Software Development Kit

Library Manual

Enrollment SDK

© INNOVATRICS s.r.o 2026

Total number of pages: 768
Latest version released by: innovatrics s.r.o.

INNOVATRICS s.r.o

All rights reserved. No part of the document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the written permission of INNOVATRICS s.r.o.

Company or product names mentioned in this document may be trademarks or registered trademarks of their respective companies.

1 Enrollment . . . 26
1.1 Overview . . . 26
1.1.1 Runtime Requirements . . . 26
1.1.2 Installation . . . 27
1.1.3 Initialization . . . 28
1.1.4 Applicant . . . 30
1.1.5 Extraction . . . 39
1.1.6 Verification . . . 42
1.1.7 Identification . . . 43
1.1.8 Similarity Score . . . 48
1.1.9 Quality . . . 50
1.1.10 Server . . . 51
1.2 Design . . . 51
1.2.1 Header and Binary parts . . . 51
1.2.2 C++ . . . 52
1.2.3 Quality . . . 52
1.2.4 Extending SDK . . . 52
2 Version History and Legal Notices . . . 53
2.1 License Agreement . . . 113
2.2 Third-Party Software . . . 116
3 Topic Documentation . . . 118
3.1 Extending SDK . . . 118
3.1.1 Detailed Description . . . 118
3.1.2 Overview . . . 118
3.2 Binding . . . 119
3.2.1 Detailed Description . . . 120
3.2.2 Java . . . 120
3.2.3 C# . . . 123
3.3 Exception . . . 125
3.3.1 Detailed Description . . . 126
3.3.2 Overview . . . 126

3.4 Face 126
3.4.1 Detailed Description 129
3.4.2 Overview 129
3.4.3 Typedef Documentation 151
3.4.4 Enumeration Type Documentation 152
3.4.5 Function Documentation 154
3.4.6 Solvers 156
3.5 File 162
3.5.1 Detailed Description 163
3.6 Image 164
3.6.1 Detailed Description 166
3.6.2 Overview 166
3.6.3 Enumeration Type Documentation 179
3.7 Manual initialisation 180
3.7.1 Detailed Description 181
3.7.2 Function Documentation 181
3.8 Iris 183
3.8.1 Detailed Description 184
3.8.2 Overview 184
3.8.3 Typedef Documentation 189
3.8.4 Enumeration Type Documentation 189
3.8.5 Function Documentation 190
3.9 Logger 193
3.9.1 Detailed Description 194
3.9.2 Typedef Documentation 196
3.10 Print 196
3.10.1 Detailed Description 199
3.10.2 Overview 199
3.10.3 Typedef Documentation 225
3.10.4 Enumeration Type Documentation 226
3.10.5 Function Documentation 230
3.11 Serialization 232

3.11.1 Detailed Description . . . 232

4 Class Documentation . . . 238

4.1 AgeReliability Class Reference . . . 238

4.1.1 Member Function Documentation . . . 238

4.2 AlgorithmicSegmentation Class Reference . . . 239

4.2.1 Constructor & Destructor Documentation . . . 240

4.3 ANSITemplate Class Reference . . . 241

4.3.1 Constructor & Destructor Documentation . . . 242

4.3.2 Member Function Documentation . . . 242

4.4 ANSITemplateAdapter Class Reference . . . 243

4.4.1 Constructor & Destructor Documentation . . . 244

4.4.2 Member Function Documentation . . . 244

4.5 Applicant Class Reference . . . 247

4.5.1 Detailed Description . . . 248

4.5.2 Member Function Documentation . . . 249

4.6 ApplicantFaceValidation Class Reference . . . 254

4.6.1 Member Function Documentation . . . 254

4.7 ApplicantMatcher Class Reference . . . 255

4.7.1 Detailed Description . . . 255

4.7.2 Member Function Documentation . . . 255

4.8 ApplicantPrintValidation Class Reference . . . 256

4.8.1 Member Function Documentation . . . 256

4.9 Image::AreaImageResampler Class Reference . . . 257

4.9.1 Constructor & Destructor Documentation . . . 258

4.9.2 Member Function Documentation . . . 258

4.10 ICSExtractor::Artifact Class Reference . . . 259

4.11 ICSVerifyMatcher::Artifact Class Reference . . . 259

4.12 ICSExtractor::Artifacts Class Reference . . . 260

4.12.1 Member Function Documentation . . . 260

4.13 ICSVerifyMatcher::Artifacts Class Reference . . . 261

4.13.1 Member Function Documentation 261

4.14 Base64Strategy Class Reference 262

4.14.1 Detailed Description 262

4.14.2 Member Function Documentation 262

4.15 Image::BicubicImageResampler Class Reference 263

4.15.1 Constructor & Destructor Documentation 264

4.15.2 Member Function Documentation 264

4.16 ICSExtractor::BinarizedImage Class Reference 265

4.16.1 Detailed Description 265

4.16.2 Member Function Documentation 266

4.17 BinaryFileAccess Class Reference 266

4.17.1 Member Function Documentation 267

4.18 BmpImageEncoder Class Reference 267

4.18.1 Member Function Documentation 268

4.19 BytesWriterStrategy Class Reference 269

4.19.1 Member Function Documentation 269

4.20 CallbackLogger Class Reference 270

4.20.1 Detailed Description 270

4.20.2 Constructor & Destructor Documentation 271

4.20.3 Member Function Documentation 271

4.21 ModalitiesGallery::Candidate Class Reference 272

4.21.1 Detailed Description 272

4.21.2 Member Function Documentation 273

4.22 Circle Class Reference 273

4.22.1 Constructor & Destructor Documentation 274

4.22.2 Member Function Documentation 274

4.23 CircleImageMask Class Reference 275

4.23.1 Detailed Description 276

4.23.2 Constructor & Destructor Documentation 276

4.23.3 Member Function Documentation 277

4.24 CircleShape Class Reference 277

4.24.1 Constructor & Destructor Documentation . . . 278
4.24.2 Member Function Documentation . . . 278
4.25 Colors Class Reference . . . 278
4.26 AlgorithmicSegmentation::Config Class Reference . . . 279
4.26.1 Constructor & Destructor Documentation . . . 280
4.26.2 Member Function Documentation . . . 280
4.27 FaceDetector::Config Class Reference . . . 282
4.27.1 Detailed Description . . . 283
4.27.2 Member Enumeration Documentation . . . 283
4.27.3 Constructor & Destructor Documentation . . . 284
4.27.4 Member Function Documentation . . . 284
4.28 ICSVerifyMatcher::Config Class Reference . . . 286
4.28.1 Constructor & Destructor Documentation . . . 287
4.28.2 Member Function Documentation . . . 287
4.29 IrisDetector::Config Class Reference . . . 288
4.29.1 Detailed Description . . . 290
4.29.2 Member Enumeration Documentation . . . 290
4.29.3 Constructor & Destructor Documentation . . . 290
4.29.4 Member Function Documentation . . . 291
4.30 ModalitiesVerifyMatcher::Config Class Reference . . . 293
4.30.1 Constructor & Destructor Documentation . . . 294
4.30.2 Member Function Documentation . . . 294
4.31 NeuralSegmentation::Config Class Reference . . . 295
4.31.1 Constructor & Destructor Documentation . . . 296
4.31.2 Member Function Documentation . . . 297
4.32 CriticalPoint Class Reference . . . 298
4.32.1 Member Enumeration Documentation . . . 300
4.32.2 Constructor & Destructor Documentation . . . 300
4.32.3 Member Function Documentation . . . 300
4.33 CriticalPointShape Class Reference . . . 301
4.33.1 Detailed Description . . . 301
4.33.2 Constructor & Destructor Documentation . . . 302

4.33.3 Member Function Documentation . . . 302

4.34 CriticalPointsShape Class Reference . . . 303

4.34.1 Constructor & Destructor Documentation . . . 303

4.34.2 Member Function Documentation . . . 304

4.35 CroppedFace Class Reference . . . 304

4.35.1 Member Function Documentation . . . 305

4.36 EmbeddingExtractor Class Reference . . . 307

4.36.1 Detailed Description . . . 308

4.36.2 Member Function Documentation . . . 308

4.37 EmbeddingMatcher Class Reference . . . 308

4.37.1 Detailed Description . . . 309

4.37.2 Member Function Documentation . . . 309

4.38 EmbeddingTemplate Class Reference . . . 309

4.38.1 Constructor & Destructor Documentation . . . 311

4.38.2 Member Function Documentation . . . 314

4.39 EmbeddingVerifyMatcher Class Reference . . . 316

4.39.1 Detailed Description . . . 316

4.39.2 Constructor & Destructor Documentation . . . 316

4.39.3 Member Function Documentation . . . 317

4.40 EnrollmentException Class Reference . . . 318

4.40.1 Constructor & Destructor Documentation . . . 318

4.40.2 Member Function Documentation . . . 319

4.41 EnrollmentInvalidArgumentException Class Reference . . . 319

4.41.1 Member Function Documentation . . . 320

4.42 EnrollmentRuntimeException Class Reference . . . 320

4.42.1 Member Function Documentation . . . 321

4.43 ExpiredLicenseException Class Reference . . . 322

4.44 ExtractedFace Class Reference . . . 323

4.44.1 Detailed Description . . . 324

4.44.2 Member Function Documentation . . . 324

4.45 ExtractedIris Class Reference . . . 325

4.45.1 Detailed Description . . . 327
4.45.2 Member Function Documentation . . . 327
4.46 ExtractedPrint Class Reference . . . 328
4.46.1 Constructor & Destructor Documentation . . . 330
4.46.2 Member Function Documentation . . . 331
4.47 ExtractedPrintEmbedding Class Reference . . . 334
4.47.1 Detailed Description . . . 335
4.47.2 Constructor & Destructor Documentation . . . 335
4.47.3 Member Function Documentation . . . 337
4.48 Face Class Reference . . . 338
4.48.1 Member Enumeration Documentation . . . 341
4.48.2 Member Function Documentation . . . 346
4.48.3 Member Data Documentation . . . 362
4.49 FaceAttributeReliability Class Reference . . . 363
4.49.1 Detailed Description . . . 363
4.49.2 Member Function Documentation . . . 363
4.50 FaceAttributes Class Reference . . . 364
4.50.1 Detailed Description . . . 365
4.50.2 Constructor & Destructor Documentation . . . 365
4.50.3 Member Function Documentation . . . 365
4.51 FaceCapture Class Reference . . . 368
4.51.1 Member Function Documentation . . . 370
4.52 FaceCaptureDeviceTypeID Class Reference . . . 374
4.52.1 Detailed Description . . . 374
4.52.2 Constructor & Destructor Documentation . . . 375
4.53 FaceCaptureDeviceVendorID Class Reference . . . 375
4.53.1 Detailed Description . . . 375
4.53.2 Constructor & Destructor Documentation . . . 376
4.54 FaceCropConfiguration Class Reference . . . 376
4.54.1 Constructor & Destructor Documentation . . . 377
4.54.2 Member Function Documentation . . . 377

4.55 FaceDetector Class Reference 379
4.55.1 Constructor & Destructor Documentation 380
4.55.2 Member Function Documentation 381
4.56 FaceExtractorConfig Class Reference 381
4.56.1 Member Enumeration Documentation 383
4.56.2 Constructor & Destructor Documentation 383
4.56.3 Member Function Documentation 383
4.57 FaceSizeReliability Class Reference 384
4.57.1 Member Function Documentation 385
4.58 FaceTemplate Class Reference 385
4.58.1 Detailed Description 386
4.58.2 Constructor & Destructor Documentation 386
4.58.3 Member Function Documentation 388
4.59 FailedFaceAttribute Struct Reference 389
4.60 FileException Class Reference 389
4.61 FilesystemAccess Class Reference 390
4.61.1 Member Function Documentation 391
4.62 ICSExtractor::FilteredImage Class Reference 392
4.62.1 Detailed Description 393
4.62.2 Member Function Documentation 393
4.63 FlatteningValidation Class Reference 393
4.63.1 Detailed Description 394
4.63.2 Constructor & Destructor Documentation 395
4.63.3 Member Function Documentation 395
4.64 FormattingComposer Class Reference 396
4.64.1 Detailed Description 397
4.64.2 Constructor & Destructor Documentation 398
4.64.3 Member Function Documentation 398
4.65 FormattingLogger Class Reference 398
4.65.1 Detailed Description 399
4.65.2 Constructor & Destructor Documentation 399

4.65.3 Member Function Documentation . . . 399

4.66 GlassStatusReliability Class Reference . . . 400

4.66.1 Member Function Documentation . . . 401

4.67 GlobalFaceExecutor Class Reference . . . 402

4.67.1 Detailed Description . . . 402

4.67.2 Member Function Documentation . . . 402

4.68 GlobalFileAccess Class Reference . . . 403

4.68.1 Detailed Description . . . 403

4.68.2 Member Function Documentation . . . 404

4.69 GlobalImageExecutor Class Reference . . . 404

4.69.1 Detailed Description . . . 405

4.69.2 Member Function Documentation . . . 405

4.70 GlobalIrisExecutor Class Reference . . . 405

4.70.1 Detailed Description . . . 406

4.70.2 Member Function Documentation . . . 406

4.71 GlobalLogger Class Reference . . . 406

4.71.1 Detailed Description . . . 407

4.71.2 Member Function Documentation . . . 407

4.72 GlobalMultiModalityExecutor Class Reference . . . 407

4.72.1 Detailed Description . . . 408

4.72.2 Member Function Documentation . . . 408

4.73 GlobalPrintExecutor Class Reference . . . 408

4.73.1 Detailed Description . . . 409

4.73.2 Member Function Documentation . . . 409

4.74 GuessedHandPosition Class Reference . . . 409

4.74.1 Detailed Description . . . 410

4.74.2 Member Function Documentation . . . 410

4.75 ICAOReliability Class Reference . . . 410

4.75.1 Member Function Documentation . . . 411

4.76 ICSExtractor Class Reference . . . 411

4.76.1 Detailed Description . . . 412

4.76.2 Constructor & Destructor Documentation . . . 412
4.76.3 Member Function Documentation . . . 413
4.77 ICSMatcher Class Reference . . . 414
4.77.1 Detailed Description . . . 414
4.77.2 Member Function Documentation . . . 415
4.78 ICSTemplate Class Reference . . . 415
4.78.1 Detailed Description . . . 416
4.78.2 Constructor & Destructor Documentation . . . 417
4.78.3 Member Function Documentation . . . 419
4.79 ICSVerifyMatcher Class Reference . . . 422
4.79.1 Detailed Description . . . 423
4.79.2 Constructor & Destructor Documentation . . . 423
4.79.3 Member Function Documentation . . . 424
4.80 IdentificationEmbeddingExtractor Class Reference . . . 425
4.80.1 Detailed Description . . . 426
4.80.2 Constructor & Destructor Documentation . . . 426
4.80.3 Member Function Documentation . . . 427
4.81 ILOSIDTemplate Class Reference . . . 427
4.81.1 Detailed Description . . . 428
4.81.2 Constructor & Destructor Documentation . . . 428
4.81.3 Member Function Documentation . . . 429
4.82 ILOSIDTemplateAdapter Class Reference . . . 430
4.82.1 Constructor & Destructor Documentation . . . 431
4.82.2 Member Function Documentation . . . 431
4.83 Image Class Reference . . . 434
4.83.1 Member Enumeration Documentation . . . 436
4.83.2 Member Function Documentation . . . 438
4.84 ImageDimensions Class Reference . . . 453
4.84.1 Detailed Description . . . 453
4.84.2 Constructor & Destructor Documentation . . . 454
4.85 ImageDpi Class Reference . . . 454
4.85.1 Constructor & Destructor Documentation . . . 455

4.85.2 Member Function Documentation 455

4.86 ImageEncoder Class Reference 455

4.86.1 Detailed Description 456

4.86.2 Member Function Documentation 456

4.87 ImageEncoderVisitor Class Reference 457

4.88 ImageException Class Reference 457

4.88.1 Member Enumeration Documentation 459

4.88.2 Constructor & Destructor Documentation 459

4.88.3 Member Function Documentation 460

4.89 ImageFileSaver Class Reference 460

4.89.1 Detailed Description 460

4.89.2 Constructor & Destructor Documentation 461

4.89.3 Member Function Documentation 461

4.90 ImageMask Class Reference 461

4.90.1 Member Function Documentation 462

4.91 ImageMaskConfiguration Class Reference 462

4.91.1 Constructor & Destructor Documentation 463

4.91.2 Member Function Documentation 463

4.92 ImageMasker Class Reference 464

4.92.1 Member Function Documentation 464

4.93 Image::ImageResampler Class Reference 466

4.93.1 Constructor & Destructor Documentation 467

4.93.2 Member Function Documentation 468

4.94 ImageSaver Class Reference 468

4.94.1 Member Function Documentation 469

4.95 ImageShape Class Reference 469

4.95.1 Member Function Documentation 470

4.96 ImageShapeAttributes Class Reference 470

4.96.1 Member Function Documentation 471

4.97 ImageShapeDrawer Class Reference 471

4.97.1 Member Function Documentation 472

4.98 ImageType Class Reference 472
4.98.1 Member Enumeration Documentation 474
4.98.2 Constructor & Destructor Documentation 474
4.98.3 Member Function Documentation 475
4.99 IncompatibleTemplateException Class Reference 475
4.100 IndexedModalitiesGallery Class Reference 476
4.100.1 Detailed Description 478
4.100.2 Member Function Documentation 480
4.101 IndexedModalitiesGalleryConfig Class Reference 482
4.101.1 Constructor & Destructor Documentation 482
4.101.2 Member Function Documentation 483
4.102 InitializerConfig Struct Reference 484
4.102.1 Detailed Description 485
4.102.2 Member Function Documentation 485
4.103 InitializerConfigBuilder Class Reference 486
4.103.1 Detailed Description 487
4.103.2 Member Function Documentation 487
4.104 InvalidHWIDException Class Reference 490
4.105 InvalidImageArgumentException Class Reference 491
4.106 InvalidLicenseException Class Reference 492
4.106.1 Detailed Description 492
4.107 InvalidTemplateException Class Reference 493
4.108 IRawImageEncoder Class Reference 494
4.108.1 Member Function Documentation 495
4.109 Iris Class Reference 495
4.109.1 Member Function Documentation 497
4.109.2 Member Data Documentation 500
4.110 IrisAttributes Class Reference 500
4.110.1 Detailed Description 501
4.110.2 Constructor & Destructor Documentation 501
4.110.3 Member Function Documentation 502

4.111 IrisCapture Class Reference 504
4.111.1 Member Function Documentation 505
4.112 IrisCaptureDeviceTypeID Class Reference 506
4.112.1 Detailed Description 507
4.112.2 Constructor & Destructor Documentation 507
4.113 IrisCaptureDeviceVendorID Class Reference 507
4.113.1 Detailed Description 508
4.113.2 Constructor & Destructor Documentation 508
4.114 IrisDetector Class Reference 508
4.114.1 Constructor & Destructor Documentation 509
4.114.2 Member Function Documentation 510
4.115 IrisExtractorConfig Class Reference 511
4.115.1 Member Enumeration Documentation 512
4.115.2 Constructor & Destructor Documentation 513
4.115.3 Member Function Documentation 513
4.116 IrisSegment Class Reference 514
4.116.1 Member Function Documentation 515
4.117 IrisTemplate Class Reference 515
4.117.1 Detailed Description 517
4.117.2 Constructor & Destructor Documentation 517
4.117.3 Member Function Documentation 520
4.118 ISOCCTemplate Class Reference 522
4.118.1 Constructor & Destructor Documentation 522
4.118.2 Member Function Documentation 523
4.119 ISOCCTemplateAdapter Class Reference 524
4.119.1 Member Enumeration Documentation 525
4.119.2 Constructor & Destructor Documentation 525
4.119.3 Member Function Documentation 526
4.120 ISOFullFrontalImageValidator Class Reference 530
4.120.1 Detailed Description 530
4.120.2 Member Function Documentation 530

4.121 ISONCTemplate Class Reference 531
4.121.1 Constructor & Destructor Documentation 532
4.121.2 Member Function Documentation 532
4.122 ISONCTemplateAdapter Class Reference 533
4.122.1 Constructor & Destructor Documentation 534
4.122.2 Member Function Documentation 534
4.123 ISOTemplate Class Reference 537
4.123.1 Constructor & Destructor Documentation 538
4.123.2 Member Function Documentation 538
4.124 ISOTemplateAdapter Class Reference 539
4.124.1 Member Enumeration Documentation 540
4.124.2 Constructor & Destructor Documentation 540
4.124.3 Member Function Documentation 541
4.125 Jp2ImageEncoder Class Reference 544
4.125.1 Member Function Documentation 545
4.126 JpgImageEncoder Class Reference 546
4.126.1 Constructor & Destructor Documentation 548
4.126.2 Member Function Documentation 548
4.127 JsonWriter Class Reference 550
4.127.1 Detailed Description 551
4.127.2 Member Function Documentation 551
4.128 Face::Keypoint Class Reference 554
4.128.1 Member Function Documentation 555
4.129 Face::Keypoints Class Reference 556
4.129.1 Detailed Description 558
4.129.2 Member Function Documentation 558
4.130 LicenseException Class Reference 566
4.131 LicenseFeatureNotAllowedException Class Reference 567
4.132 LicenseLimitReachedException Class Reference 568
4.133 LicenseNotFoundException Class Reference 569
4.134 Line Class Reference 570

4.134.1 Constructor & Destructor Documentation . . . 571
4.134.2 Member Function Documentation . . . 571
4.135 LineShape Class Reference . . . 572
4.135.1 Constructor & Destructor Documentation . . . 572
4.135.2 Member Function Documentation . . . 573
4.136 Logger Class Reference . . . 573
4.136.1 Detailed Description . . . 573
4.136.2 Member Function Documentation . . . 574
4.137 LoggerCallBacks Struct Reference . . . 574
4.137.1 Detailed Description . . . 575
4.137.2 Member Data Documentation . . . 575
4.138 LogMessage Class Reference . . . 576
4.138.1 Detailed Description . . . 576
4.138.2 Member Function Documentation . . . 576
4.139 ICSVerifyMatcher::MatchingMinutiae Class Reference . . . 577
4.139.1 Detailed Description . . . 577
4.139.2 Member Function Documentation . . . 578
4.140 Minutia Class Reference . . . 578
4.140.1 Detailed Description . . . 580
4.140.2 Member Enumeration Documentation . . . 580
4.140.3 Constructor & Destructor Documentation . . . 580
4.140.4 Member Function Documentation . . . 581
4.140.5 Member Data Documentation . . . 581
4.141 MinutiaeShape Class Reference . . . 581
4.141.1 Constructor & Destructor Documentation . . . 582
4.141.2 Member Function Documentation . . . 582
4.142 MinutiaShape Class Reference . . . 583
4.142.1 Detailed Description . . . 583
4.142.2 Constructor & Destructor Documentation . . . 584
4.142.3 Member Function Documentation . . . 584
4.143 ModalitiesGallery::ModalitiesCandidates Class Reference . . . 585
4.143.1 Member Function Documentation . . . 585

4.144 ModalitiesEmbeddingVerifyMatcher Class Reference 586
4.144.1 Detailed Description 588
4.144.2 Constructor & Destructor Documentation 588
4.144.3 Member Function Documentation 589
4.145 ModalitiesGallery Class Reference 590
4.145.1 Detailed Description 591
4.145.2 Member Function Documentation 594
4.146 ModalitiesGalleryConfig Class Reference 598
4.146.1 Constructor & Destructor Documentation 599
4.146.2 Member Function Documentation 599
4.147 ModalitiesGalleryIndexedTemplate Class Reference 600
4.147.1 Constructor & Destructor Documentation 601
4.147.2 Member Function Documentation 602
4.148 ModalitiesGalleryReference Class Reference 603
4.148.1 Member Function Documentation 603
4.149 ModalitiesICSTemplate Class Reference 604
4.149.1 Detailed Description 605
4.149.2 Constructor & Destructor Documentation 605
4.149.3 Member Function Documentation 606
4.150 ModalitiesICSTemplateDefaultFactory Class Reference 607
4.150.1 Detailed Description 608
4.150.2 Constructor & Destructor Documentation 608
4.150.3 Member Function Documentation 609
4.151 ModalitiesICSTemplateFactory Class Reference 609
4.151.1 Detailed Description 610
4.151.2 Member Function Documentation 610
4.152 IndexedModalitiesGallery::ModalitiesIdentifiers Class Reference 610
4.152.1 Member Function Documentation 611
4.153 ModalitiesMatcher Class Reference 612
4.153.1 Detailed Description 612
4.153.2 Member Function Documentation 612

4.154 ModalitiesTemplate Class Reference 613
4.154.1 Detailed Description 615
4.154.2 Member Function Documentation 615
4.155 ModalitiesVerifyMatcher Class Reference 618
4.155.1 Detailed Description 619
4.155.2 Constructor & Destructor Documentation 619
4.155.3 Member Function Documentation 620
4.156 ModalityScore Class Reference 623
4.156.1 Constructor & Destructor Documentation 624
4.156.2 Member Function Documentation 624
4.157 MouthOpennessReliability Class Reference 625
4.157.1 Member Function Documentation 626
4.158 NeuralSegmentation Class Reference 626
4.158.1 Constructor & Destructor Documentation 627
4.159 Image::NISTImageResampler Class Reference 628
4.159.1 Member Enumeration Documentation 629
4.159.2 Constructor & Destructor Documentation 630
4.159.3 Member Function Documentation 631
4.160 NotSupportedException Class Reference 632
4.161 NullArgumentException Class Reference 633
4.162 NullLogger Class Reference 634
4.163 OFIQFace Class Reference 634
4.163.1 Detailed Description 636
4.163.2 Member Function Documentation 637
4.164 PngImageEncoder Class Reference 648
4.164.1 Member Function Documentation 649
4.165 Point Class Reference 650
4.165.1 Constructor & Destructor Documentation 651
4.165.2 Member Function Documentation 651
4.166 Polygon Class Reference 651
4.166.1 Constructor & Destructor Documentation 652

4.166.2 Member Function Documentation 652

4.167 PolygonImageMask Class Reference 653

4.167.1 Detailed Description 654

4.167.2 Constructor & Destructor Documentation 654

4.167.3 Member Function Documentation 655

4.168 Print Class Reference 655

4.168.1 Member Function Documentation 658

4.169 PrintAttributes Class Reference 665

4.169.1 Detailed Description 667

4.169.2 Constructor & Destructor Documentation 667

4.169.3 Member Function Documentation 668

4.170 PrintCapture Class Reference 671

4.170.1 Detailed Description 672

4.170.2 Member Function Documentation 672

4.171 PrintCaptureDeviceTypeID Class Reference 676

4.171.1 Detailed Description 677

4.171.2 Constructor & Destructor Documentation 677

4.172 PrintCaptureDeviceVendorID Class Reference 677

4.172.1 Detailed Description 677

4.172.2 Constructor & Destructor Documentation 678

4.173 PrintCropConfiguration Class Reference 678

4.173.1 Constructor & Destructor Documentation 679

4.173.2 Member Function Documentation 679

4.174 PrintDetector Class Reference 681

4.174.1 Detailed Description 681

4.174.2 Constructor & Destructor Documentation 682

4.174.3 Member Function Documentation 682

4.175 PrintDetectorConfig Class Reference 683

4.175.1 Constructor & Destructor Documentation 684

4.175.2 Member Function Documentation 684

4.176 PrintSegment Class Reference 685

4.176.1 Detailed Description 686
4.176.2 Member Function Documentation 686
4.177 PrintTemplate Class Reference 688
4.177.1 Constructor & Destructor Documentation 690
4.177.2 Member Function Documentation 691
4.178 RawImage Class Reference 693
4.178.1 Member Enumeration Documentation 694
4.178.2 Constructor & Destructor Documentation 694
4.178.3 Member Function Documentation 695
4.179 Rectangle Class Reference 696
4.179.1 Detailed Description 697
4.179.2 Member Typedef Documentation 697
4.179.3 Constructor & Destructor Documentation 697
4.179.4 Member Function Documentation 698
4.180 RectangleShape Class Reference 700
4.180.1 Constructor & Destructor Documentation 701
4.180.2 Member Function Documentation 701
4.181 RectanglesShape Class Reference 702
4.181.1 Constructor & Destructor Documentation 702
4.181.2 Member Function Documentation 703
4.182 RGB Class Reference 703
4.182.1 Detailed Description 704
4.182.2 Member Function Documentation 704
4.183 RGBA Class Reference 704
4.183.1 Member Enumeration Documentation 706
4.183.2 Member Function Documentation 706
4.184 SegmentationMaskReliability Class Reference 706
4.184.1 Member Function Documentation 707
4.185 Serializable Class Reference 707
4.185.1 Member Function Documentation 708
4.186 Shapes Class Reference 708

4.187 ICSExtractor::SkeletonImage Class Reference 709
4.187.1 Detailed Description 709
4.187.2 Member Function Documentation 710
4.188 SlapCapture Class Reference 711
4.188.1 Detailed Description 712
4.188.2 Member Function Documentation 713
4.189 SourceLocation Class Reference 714
4.189.1 Detailed Description 715
4.189.2 Constructor & Destructor Documentation 715
4.189.3 Member Function Documentation 715
4.190 SupportedImageEncoder Class Reference 716
4.190.1 Member Function Documentation 717
4.191 TemplateReliability Class Reference 718
4.191.1 Member Function Documentation 718
4.192 ThumbsCapture Class Reference 719
4.192.1 Detailed Description 720
4.192.2 Member Function Documentation 721
4.193 TiffImageEncoder Class Reference 722
4.193.1 Member Enumeration Documentation 723
4.193.2 Constructor & Destructor Documentation 724
4.193.3 Member Function Documentation 724
4.194 TintedGlassesReliability Class Reference 725
4.194.1 Member Function Documentation 725
4.195 UniversalPassiveLivenessReliability Class Reference 726
4.195.1 Member Function Documentation 726
4.196 UnsupportedTemplateException Class Reference 727
4.197 WebpImageEncoder Class Reference 728
4.197.1 Member Function Documentation 729
4.198 Writer Class Reference 730
4.198.1 Detailed Description 731
4.198.2 Member Function Documentation 736

4.199 WsqImageEncoder Class Reference 741
4.199.1 Member Function Documentation 742

Bibliography 745

Index 746

ILLUSTRATIONS

1 FAR/FRR . . . 48
2 Face Size . . . 129
3 Face Area . . . 130
4 Face . . . 130
5 Face area visible in image. . . . 130
6 Possible face rotations (defined as in DIN9300) . . . 131
7 Face . . . 132
8 Face with crop region outside of image . . . 133
9 Face with shrunken crop . . . 133
10 Face with FULL crop configuration . . . 134
11 Face with FULL_NOT_ALIGNED crop . . . 135
12 Face with TOKEN crop . . . 135
13 Face with TOKEN_NOT_FRONTAL crop . . . 135
14 Faces . . . 136
15 Detect Faces . . . 137
16 Detected face 1 . . . 137
17 Detected face 2 . . . 137
18 Detected face 3 . . . 138
19 Detected face 4 . . . 138
20 Face . . . 139
21 Face with colored background . . . 140
22 Masked Face . . . 140
23 Face with transparent background . . . 140
24 OFIQ face . . . 145
25 Face keypoints . . . 148
26 Resize input . . . 169
27 Resize output . . . 169
28 DPI change input . . . 170
29 DPI change output . . . 170
30 Draw input . . . 170
31 Draw rectangle output . . . 170
32 Draw circle output . . . 171
33 Draw custom shape output . . . 171
34 Circle masked inside . . . 173
35 Circle masked outside . . . 174
36 Polygon masked and blurred inside . . . 174
37 Polygon masked and blurred outside . . . 175
38 Masked combination . . . 176
39 Crop aligned rectangle input . . . 176
40 Crop aligned rectangle output . . . 176
41 Crop rectangle input . . . 177
42 Crop rectangle output . . . 177
43 Crop circle input . . . 178
44 Crop circle output . . . 178
45 Partial intersection . . . 178
46 Inside intersection . . . 179
47 Outside intersection . . . 179
48 Same intersection . . . 179
49 No intersection . . . 179
50 Eye Capture . . . 185
51 Detected Iris . . . 185
52 Iris . . . 186
53 ISO/IEC 19794-6:2011 masked iris . . . 189
54 Logger design . . . 195
55 Logger design . . . 195
56 Slap . . . 200

57 Segmented Slap . . . 201
58 Slap Segment 1 . . . 201
59 Slap Segment 2 . . . 202
60 Slap Segment 3 . . . 202
61 Slap Segment 4 . . . 202
62 Modified Segmented Slap . . . 204
63 Modified Slap Segment 1 . . . 204
64 Modified Slap Segment 2 . . . 204
65 Modified Slap Segment 3 . . . 204
66 Modified Slap Segment 4 . . . 204
67 Thumbs . . . 205
68 Segmented Thumbs . . . 206
69 Thumbs Segment 1 . . . 207
70 Thumbs Segment 2 . . . 207
71 Print . . . 207
72 Segmented Print . . . 208
73 Fingerprint Segment . . . 208
74 Finger for extraction . . . 210
75 Binarized . . . 210
76 Filtered . . . 210
77 Skeleton . . . 210
78 Probe . . . 210
79 Gallery . . . 210
80 Minutiae . . . 215
81 Critical Points . . . 216
82 Use built in JSON with Base64 encoded bytes . . . 233
83 Use client specific serialization . . . 234
84 Use client specific bytes strategy . . . 237
85 Finger to binarize . . . 265
86 Binarized . . . 265
87 Critical points types . . . 300
88 Finger for filtering . . . 393
89 Filtered . . . 393
125 Partial intersection . . . 437
126 Inside intersection . . . 437
127 Outside intersection . . . 437
128 No intersection . . . 437
129 Same intersection . . . 437
130 Minutia types . . . 580
131 Finger for skeleton . . . 709
132 Skeleton . . . 710
133 Use built in JSON with Base64 encoded bytes . . . 732
134 Use client specific serialization . . . 733
135 Use client specific bytes strategy . . . 735

TABLES

1 Enrollment

1.1 Overview

The library for applicant enrollment. The main purpose is to process images of print, iris, face modalities in various formats and create Applicant object suitable for further processing. The library provides following features to create Applicant:

- Image processing
- Extraction
- 1:1 Verification
- Identification

Some features of library can be extended/adapted to user needs via implementation of provided interfaces of features. All available interfaces that can be used to customize are described in Extending SDK.

We provides wrappers for

- Java
- C#

The SDK can execute functionality in same process in which it is used or in provided server. The following functionalities can be executed on server:

- Image
- Print
- Face
- Iris

1.1.1 Runtime Requirements

1.1.1.1 Linux

- The **glibc** version MUST be **2.17 or higher** (e.g., CentOS 7 and above).
- The **processor** MUST support the **AVX2 instruction set**.
- The **system architecture** MUST be **x86_64 (64-bit)**.
- The **libusb** MUST be installed

1.1.1.2 Windows

- The **minimum supported version** is `Windows Vista SP2 or Windows Server 2008 SP2`.
- The **processor** MUST support the **AVX2 instruction set**.
- The **system architecture** MUST be **x86_64 (64-bit)** or **x86 (32-bit)**.

Note

In the case of x86 (32-bit) architecture, the Iris, Face and Print Embedding functionalities are NOT supported.

1.1.1.3 Android

- The **Android version** MUST be **5.0 or higher**, which corresponds to `API level 21`.

1.1.2 Installation

We provide isolated installation. All necessary shared libraries are installed with Enrollment SDK. The libraries are available only for Enrollment SDK and do not influence other SDKs. Also other SDKs do not influence Enrollment SDK installation.
We provide more variants of Enrollment SDK. Only one variant can be installed on device/PC. It is ensured by installation process.

1.1.2.1 Reducing installation size

The Enrollment SDK installation package includes large binary files `*.solver`, `*.nnkitsolver` that can be removed if they are not in use. For more details, refer to section Solvers.

1.1.2.2 Linux

Isolated installation

We use *rpath* to isolate Enrollment SDK.
The RPM installation uses `ldconfig` to add installed library into cache that is used by the run-time linker, ld.so or ld-linux.so.

Variant

On Linux package name is different between SDK variants. The variant is part of package name. In rpm specification file all variants provide the same name on which they all conflict. It ensures that only one variant can be installed in system. The maintainer MUST manually uninstall previous variant when he/she needs to install other one. The same variant can be updated without problem.
Let assume that maintainer already installed Enrollment SDK in DEV variant with command

```
yum install innovatrics-enrollment-sdk-dev
```

and he/she needs to install other variant, general one

```
yum install innovatrics-enrollment-sdk
```

then `yum` will report error

```
Error: innovatrics-enrollment-sdk conflicts with innovatrics-enrollment-sdk-dev-7.0.3-1.el7.x86_64
Error: innovatrics-enrollment-sdk-dev conflicts with innovatrics-enrollment-sdk-7.0.3-1.el7.x86_64
```

to fix error maintainer MUST remove previous variant manually

```
yum remove innovatrics-enrollment-sdk-dev
```

1.1.2.3 Windows

Isolated installation

We use `concepts of isolated applications` to achieve isolated Enrollment SDK. The dependant libraries contains embedded manifest to force windows to use `assembly search sequence` instead of `dll search order` to locate dependant DLLs. For troubleshooting use `sxstrace.exe`:

- `sxstrace Trace -logfile:log.bin -nostop`
- run investigated application
- `sxstrace Stoptrace`
- `sxstrace Parse -logfile:log.bin -outfile:log.log`

Then installation bin folder is added to PATH environment variable for x86_64 build. The x86 does not modify the PATH variable to not influence x86_64 windows applications. The x86_64 installation creates also INNO↩_ENROLLMENT_HOME variable and sets its value to installation folder. The x86 installation creates INNO↩_ENROLLMENT_32_HOME variable and sets its value to installation folder.

Variant

On Widows all variants have same GUID. Windows installer will ensure that only one installation is presented on system.

1.1.3 Initialization

The Enrollment SDK initialization can be

- automatic
- manual

Once automatic initialization is applied then manual initialization is not possible. Automatic initialization is done on library load when all prerequisites are fulfilled

- `iengine.lic` license file is found
- solvers (neural network models) exist at `iface/models` next to the loaded library; see Solvers for package layout and discovery

To suppress automatic initialization even if all prerequisites are satisfied set `INNO_ENROLLMENT_SDK_↩MANUAL_INITIALIZATION` to one of these case-insensitive values: `1`, `true`, `on`, `manual`, `enable`, `enabled` before your application starts.

It is uncommon for all prerequisites to be fulfilled at application startup on Android. However, if this is your use case, you can suppress automatic initialization by calling:

```
android.system.Os.setenv("INNO_ENROLLMENT_SDK_MANUAL_INITIALIZATION", "1", true);
```

before invoking:

```
enrollment.LoadLibrary();
```

A valid reason to disable automatic initialization is to gain full control over when the Enrollment SDK is initialized. This allows you to provide a custom configuration instead of relying on the default configuration used during automatic initialization.

By default, the Enrollment SDK initializes itself automatically when the shared library is loaded.

If you need to control OpenCV or ONNX Runtime thread counts before the SDK loads its models, suppress auto-init and explicitly initialize the SDK with a custom configuration. Automatic initialization may not be suitable when some prerequisites are unavailable at startup or when full control over the SDK configuration is required. In such cases, initialize the SDK explicitly using either `Initialize(std::vector<uint8_t> license, std::string modelsPath)` or `Initialize(const InitializerConfig& config)`. The initialization functions can only be called once during the lifetime of the process. The Enrollment SDK also provides the helper functions `DefaultModelsDirectory()` and `FindLicenseFile()`, which can be used to populate the default paths in the initialization configuration.

For Java-specific initialization requirements (loading native libraries before any SDK call), see the initialization section of the Java binding documentation.

C++

```cpp
const auto cfg = inno::InitializerConfigBuilder()
                    .SetLicensePath(inno::FindLicenseFile())
                    .SetModelsPath(inno::DefaultModelsDirectory())
                    .SetFaceIrisOpencvThreads(4)
                    .SetFaceIrisOrtIntraThreads(4)
                    .SetFaceIrisOrtInterThreads(1)
                    .Build();

inno::Initialize(cfg);

// SDK is now fully initialised with the requested thread counts.
// Proceed with normal face / iris / fingerprint operations.
```

java

```java
EnrollmentInitializer.LoadNativeLibrary(
        ()
                -> new InitializerConfigBuilder()
                            .SetLicensePath(enrollment.FindLicenseFile())
                            .SetModelsPath(
                                    enrollment.DefaultModelsDirectory())
                            .SetFaceIrisOpencvThreads(4)
                            .SetFaceIrisOrtIntraThreads(4)
                            .SetFaceIrisOrtInterThreads(1)
                            .Build());

// SDK is now fully initialized with the requested thread counts.
// Proceed with normal face / iris / fingerprint operations.
```

csharp

```csharp
using (var builder = new InitializerConfigBuilder())
{
    var cfg = builder
        .SetLicensePath(enrollment.FindLicenseFile())
        .SetModelsPath(enrollment.DefaultModelsDirectory())
        .SetFaceIrisOpencvThreads(4)
        .SetFaceIrisOrtIntraThreads(4)
        .SetFaceIrisOrtInterThreads(1)
        .Build();

    enrollment.Initialize(cfg);
}

// SDK is now fully initialised with the requested thread counts.
// Proceed with normal face / iris / fingerprint operations.
```

1.1.3.1 Licensing

The license file `iengine.lic` prerequisite must be added into one of following directories to be possible to initialize SDK on load time:

- Linux
  - **Per Application:** current working directory

  - **Per User:** `~/.innovatrics/`
  - **Per System:** `/etc/innovatrics/`
- Windows
  - **Per Application:** current working directory
  - **Per User:** `%LOCALAPPDATA%/Innovatrics/`
  - **Per System:** `C:/ProgramData/Innovatrics/`

The search follows this priority:

- Application directory
- User directory
- System directory

Otherwise user MUST provide content of license via function Initialize(). The HWID for generating license can be obtained by SDK function HardwareID().

1.1.3.1.1 Dongle license

If you are using dongle-based licensing, you will receive a license file that is paired with one or more dongles.

The dongle **must** be connected before starting the application or calling the Initialize() function. It **must** remain connected throughout the entire runtime of the application.

1.1.4 Applicant

Applicant represent enrolled person. It holds person biometric data, images, qualities. Once you finished Applicant enrollment you can save its ModalitiesTemplate for further processing. For details see Store,Load Applicant biometrics representations.

1.1.4.1 Prints

We can add Fingerprints/fingertips from SlapCapture,Thumbs/Palms.

C++

```
const auto dpi = 1000;
auto img = Image::Decode(BinaryFile::ReadAll("assets/slap.png"), dpi);
auto slap = SlapCapture::Create(std::move(img), HandPosition::RIGHT);

// Segment Slap
AlgorithmicSegmentation segmentation(AlgorithmicSegmentation::Config::Default());
// Create Applicant and fill it with segmented Prints
Applicant applicant;
// Create ICSExtractor for calculating Template for segmented prints
ICSExtractor extractor;

for (auto& segment : slap->SegmentWith(segmentation))
{
    if (not segment->HasPosition())
    {
        // Set position manually. May be ask operator for position.
        // It is not possible to create Print without position.
        // In this example all segments have position.
    }
    // create fingerprint
```

```
    auto fp = segment->ToPrint();
    // Put segmented fingers into applicant with their template
    applicant.AddPrint(extractor.Extract(fp));
}
applicant.AddSlap(slap);
```

java

```
final int dpi = 1000;
Image img = Image.Decode(BinaryFile.ReadAll("assets/slap.png"), dpi);

SlapCapture slap = SlapCapture.Create(img, HandPosition.RIGHT);
// Segment Slap
AlgorithmicSegmentation segmentation = new AlgorithmicSegmentation();
// Create Applicant and fill it with segmented Prints
Applicant applicant = new Applicant();
// Create ICSExtractor for calculating Template for segmented prints
ICSExtractor extractor = new ICSExtractor();

for (PrintSegment segment : slap.SegmentWith(segmentation)) {
    if (segment.HasPosition() == false) {
        // Set position manually. May be ask operator for position.
        // It is not possible to create Print without position.
        // In this example all segments have position.
    }
    // create fingerprint
    Print fp = segment.ToPrint();
    // Put segmented fingers into applicant with their template
    applicant.AddPrint(extractor.Extract(fp));
}
applicant.AddSlap(slap);
```

csharp

```
uint dpi = 1000;
Image img = Image.Decode(BinaryFile.ReadAll("assets/slap.png"), dpi);

SlapCapture slap = SlapCapture.Create(img, HandPosition.RIGHT);
// Segment Slap
AlgorithmicSegmentation segmentation = new AlgorithmicSegmentation();
// Create Applicant and fill it with segmented Prints
Applicant applicant = new Applicant();
// Create ICSExtractor for calculating Template for segmented prints
ICSExtractor extractor = new ICSExtractor();

foreach (PrintSegment segment in slap.SegmentWith(segmentation))
{
    if (segment.HasPosition() == false)
    {
        // Set position manually. May be ask operator for position.
        // It is not possible to create Print without position.
        // In this example all segments have position.
    }
    // create fingerprint
    Print fp = segment.ToPrint();
    // Put segmented fingers into applicant with their template
    applicant.AddPrint(extractor.Extract(fp));
}
applicant.AddSlap(slap);
```

1.1.4.1.1 Print flattening

We provide check for wrong enrollment. Print is valid in following scenarios:

- When has same position and Similarity Score is above or equal Similarity Threshold given in constructor for same position
- When has different position and Similarity Score is below or equal Similarity Threshold given in constructor for different position.

c++

```
auto enrolled = applicant.AddUniquePrint(enrolling);

if (enrolled != nullptr)
{
    // The print is invalid
```

```
    auto enrolledPos = enrolled->print->position;
    auto enrollingPos = enrolling->print->position;

    if (enrolledPos == enrollingPos)
    {
        std::cout « "    Dissimilar print on same position " « enrolledPos
                  « std::endl;
    }
    else
    {
        std::cout « "    Similar print on position " « enrolledPos
                  « " with position " « enrollingPos « std::endl;
    }
}
```

java

```
ExtractedPrint enrolled = applicant.AddUniquePrint(enrolling);

if (enrolled != null) {
    // The print is invalid
    PrintPosition enrolledPos = enrolled.getPrint().getPosition();
    PrintPosition enrollingPos = enrolling.getPrint().getPosition();
    if (enrolledPos == enrollingPos) {
        System.out.println("    Dissimilar print on same position " + enrolledPos);
    } else {
        System.out.println("    Similar print on position " + enrolledPos
                           + " with position " + enrollingPos);
    }
}
```

csharp

```
ExtractedPrint enrolled = applicant.AddUniquePrint(enrolling);

if (enrolled != null)
{
    // The print is invalid
    PrintPosition enrolledPos = enrolled.print.position;
    PrintPosition enrollingPos = enrolling.print.position;
    if (enrolledPos == enrollingPos)
    {
        System.Console.WriteLine("    Dissimilar print on same position " +
                                 enrolledPos);
    }
    else
    {
        System.Console.WriteLine("    Similar print on position " + enrolledPos +
                                 " with position " + enrollingPos);
    }
}
```

1.1.4.1.2 Custom Print validation

The SDK provides possibility to add only Print that pass client specific Print validation. The client can implement interface ApplicantPrintValidation and use it in Applicant::AddValidPrint.

1.1.4.2 Faces

We can add multiple faces to applicant.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/face.png"));

auto fc = FaceCapture::Create(std::move(img));
FaceDetector fd;
// Detect face
auto faces = fc->DetectWith(fd);

Applicant applicant;

// Put face into applicant
if (not faces.empty())
{
    applicant.AddFace(faces[0]->Extract());
```

```
}
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));

FaceCapture fc = new FaceCapture(img);
FaceDetector fd = new FaceDetector();
// Detect face
Faces faces = fc.DetectWith(fd);

Applicant applicant = new Applicant();

// Put face into applicant
if (faces.size() > 0) {
    applicant.AddFace(faces.get(0).Extract());
}
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));

FaceCapture fc = new FaceCapture(img);
FaceDetector fd = new FaceDetector();
// Detect face
Faces faces = fc.DetectWith(fd);

Applicant applicant = new Applicant();

// Put face into applicant
if (faces.Count > 0)
{
    applicant.AddFace(faces[0].Extract());
}
```

1.1.4.2.1 Custom Face validation

The SDK provides possibility to add only Face that pass client specific Face validation. The client can implement interface ApplicantFaceValidation and use it in Applicant::AddValidFace.

c++

```
class CustomFaceValidation : public ApplicantFaceValidation
{
public:
    bool Validate(
      const std::shared_ptr<ExtractedFace>& probe,
      const std::vector<std::shared_ptr<ExtractedFace»& /*  gallery */) final
    {
        // probe is valid when all attributes pass ICAO check
        ICAOReliability icao;
        return icao.Validate(probe->face).empty();
    }
};

static void ApplicantIcaoReliability()
{
    auto face = ExtractFace();
    auto applicant = Applicant();
    auto faceValidation = CustomFaceValidation();
    if (not applicant.AddValidFace(face, faceValidation))
    {
        std::cout « "   Face does not fullfil ICAO requirement" « std::endl;
    }
}
```

java

```
class CustomFaceValidation extends ApplicantFaceValidation {
    public boolean Validate(ExtractedFace probe, ExtractedFaces gallery) {
        // probe is valid when all attributes pass ICAO check
        ICAOReliability icao = new ICAOReliability();
        return icao.Validate(probe.getFace()).isEmpty();
    }
};

public void ApplicantIcaoReliability() {
    ExtractedFace face = ExtractFace();
    Applicant applicant = new Applicant();
    CustomFaceValidation faceValidation = new CustomFaceValidation();
```

```
    if (applicant.AddValidFace(face, faceValidation) == false) {
        System.out.println("   Face does not fullfil ICAO requirement");
    }
}
```

csharp

```
class CustomFaceValidation : ApplicantFaceValidation
{
    public override bool Validate(ExtractedFace probe, ExtractedFaces gallery)
    {
        // probe is valid when all attributes pass ICAO check
        ICAOReliability icao = new ICAOReliability();
        return icao.Validate(probe.face).Count == 0;
    }
};

public void ApplicantIcaoReliability()
{
    ExtractedFace face = ExtractFace();
    Applicant applicant = new Applicant();
    CustomFaceValidation faceValidation = new CustomFaceValidation();
    if (applicant.AddValidFace(face, faceValidation) == false)
    {
        System.Console.WriteLine("   Face does not fullfil ICAO requirement");
    }
}
```

1.1.4.3 Irises

We can add multiple irises to applicant.

c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/eye_L.png"));
const auto ic = IrisCapture::Create(std::move(img), IrisPosition::LEFT_EYE);
IrisDetector id;
// Detect iris
auto iris = ic->DetectWith(id)->iris;
auto irisTemplate = iris->Extract();

Applicant applicant;
// Put iris into applicant
applicant.AddIris(irisTemplate);
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));

IrisCapture ic = new IrisCapture(img, IrisPosition.LEFT_EYE);
IrisDetector id = new IrisDetector();
// Detect iris
IrisSegment segment = ic.DetectWith(id);
Applicant applicant = new Applicant();

// Put iris into applicant
if (segment != null) {
    applicant.AddIris(segment.getIris().Extract());
}
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));

IrisCapture ic = new IrisCapture(img, IrisPosition.LEFT_EYE);
IrisDetector id = new IrisDetector();
// Detect iris
IrisSegment segment = ic.DetectWith(id);
Applicant applicant = new Applicant();

// Put iris into applicant
if (segment != null)
{
    applicant.AddIris(segment.iris.Extract());
}
```

1.1.4.4 Verification

The probe and gallery Applicant can be verified by

- convenient function Applicant::SimilarWith() - it is not efficient when it is used to verify more applicants because it always create ModalitiesTemplate.
- converting probe and gallery Applicant to ModalitiesTemplate and using this templates for verification. Use function Applicant::ToTemplate() to prepare ModalitiesTemplate for matching. For details see 1 to 1 Modalities Verification.

You can verify Applicant with any combinations of modalities (e.g. only prints).

c++

```
auto print1Img = Image::Decode(BinaryFile::ReadAll("assets/s1_li.png"));
auto print1 = Print::Create(std::move(print1Img), PrintPosition::LEFT_INDEX);
ICSExtractor extractor;
auto extractedPrint1 = extractor.Extract(print1);

Applicant probe;
probe.AddPrint(extractedPrint1);

auto print2Img = Image::Decode(BinaryFile::ReadAll("assets/s2_li.png"));
auto print2 = Print::Create(std::move(print2Img), PrintPosition::LEFT_INDEX);
auto extractedPrint2 = extractor.Extract(print2);

Applicant gallery;
gallery.AddPrint(extractedPrint2);

ModalitiesVerifyMatcher matcher;
auto scores = probe.SimilarWith(gallery, matcher);

std::cout « "    prints " « scores.GetPrintScore() « ", faces "
          « scores.GetFaceScore() « ", irises " « scores.GetIrisScore()
          « std::endl;
```

java

```
Image printImg1 = Image.Decode(BinaryFile.ReadAll("assets/s1_li.png"));
Print print1 = new Print(printImg1, PrintPosition.LEFT_INDEX);
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint extractedPrint1 = extractor.Extract(print1);

Applicant probe = new Applicant();
probe.AddPrint(extractedPrint1);

Image printImg2 = Image.Decode(BinaryFile.ReadAll("assets/s2_li.png"));
Print print2 = new Print(printImg2, PrintPosition.LEFT_INDEX);
ExtractedPrint extractedPrint2 = extractor.Extract(print2);

Applicant gallery = new Applicant();
gallery.AddPrint(extractedPrint2);

ModalitiesVerifyMatcher matcher = new ModalitiesVerifyMatcher();
ModalityScore scores = probe.SimilarWith(gallery, matcher);

System.out.println("    prints " + scores.GetPrintScore() + ", faces "
                   + scores.GetFaceScore() + ", irises " + scores.GetIrisScore());
```

csharp

```
var printImg1 = Image.Decode(BinaryFile.ReadAll("assets/s1_li.png"));
var print1 = new Print(printImg1, PrintPosition.LEFT_INDEX);
var extractor = new ICSExtractor();
var extractedPrint1 = extractor.Extract(print1);

var probe = new Applicant();
probe.AddPrint(extractedPrint1);

var printImg2 = Image.Decode(BinaryFile.ReadAll("assets/s2_li.png"));
var print2 = new Print(printImg2, PrintPosition.LEFT_INDEX);
var extractedPrint2 = extractor.Extract(print2);

var gallery = new Applicant();
gallery.AddPrint(extractedPrint2);

var matcher = new ModalitiesVerifyMatcher();
```

```
var scores = probe.SimilarWith(gallery, matcher);

System.Console.WriteLine("    prints " + scores.GetPrintScore() + ", faces " +
                         scores.GetFaceScore() + ", irises " +
                         scores.GetIrisScore());
```

You can verify Applicant with all modalities.

c++

```
auto printImg = Image::Decode(BinaryFile::ReadAll("assets/finger.png"));
auto print = Print::Create(std::move(printImg), PrintPosition::LEFT_INDEX);
ICSExtractor extractor;
auto extractedPrint = extractor.Extract(print);

auto faceImg = Image::Decode(BinaryFile::ReadAll("assets/face.png"));
auto faceCapture = FaceCapture::Create(std::move(faceImg));
FaceDetector faceDetector;
auto face = faceCapture->DetectWith(faceDetector)[0];
auto extractedFace = face->Extract();

auto irisImg = Image::Decode(BinaryFile::ReadAll("assets/eye_L.png"));
auto irisCapture =
  IrisCapture::Create(std::move(irisImg), IrisPosition::LEFT_EYE);
IrisDetector irisDetector;
auto irisSegment = irisCapture->DetectWith(irisDetector);
auto extractedIris = irisSegment->iris->Extract();

Applicant probe;
probe.AddPrint(extractedPrint);
probe.AddFace(extractedFace);
probe.AddIris(extractedIris);

Applicant gallery;
gallery.AddPrint(extractedPrint);
gallery.AddFace(extractedFace);
gallery.AddIris(extractedIris);

ModalitiesVerifyMatcher matcher;
auto scores = probe.SimilarWith(gallery, matcher);

std::cout « "    prints " « scores.GetPrintScore() « ", faces "
          « scores.GetFaceScore() « ", irises " « scores.GetIrisScore()
          « std::endl;
```

java

```
Image printImg = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
Print print = new Print(printImg, PrintPosition.LEFT_INDEX);
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint extractedPrint = extractor.Extract(print);

Image faceImg = Image.Decode(BinaryFile.ReadAll("assets/faces.png"));
FaceCapture faceCapture = new FaceCapture(faceImg);
FaceDetector faceDetector = new FaceDetector();
Face face = faceCapture.DetectWith(faceDetector).get(0);
ExtractedFace extractedFace = face.Extract();

Image irisImg = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));
IrisCapture irisCapture = new IrisCapture(irisImg, IrisPosition.LEFT_EYE);
IrisDetector irisDetector = new IrisDetector();
IrisSegment irisSegment = irisCapture.DetectWith(irisDetector);
ExtractedIris extractedIris = irisSegment.getIris().Extract();

Applicant probe = new Applicant();
probe.AddPrint(extractedPrint);
probe.AddFace(extractedFace);
probe.AddIris(extractedIris);

Applicant gallery = new Applicant();
gallery.AddPrint(extractedPrint);
gallery.AddFace(extractedFace);
gallery.AddIris(extractedIris);

ModalitiesVerifyMatcher matcher = new ModalitiesVerifyMatcher();
ModalityScore scores = probe.SimilarWith(gallery, matcher);

System.out.println("    prints " + scores.GetPrintScore() + ", faces "
                   + scores.GetFaceScore() + ", irises " + scores.GetIrisScore());
```

csharp

```
var printImg = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
var print = new Print(printImg, PrintPosition.LEFT_INDEX);
var extractor = new ICSExtractor();
```

```
var extractedPrint = extractor.Extract(print);

var faceImg = Image.Decode(BinaryFile.ReadAll("assets/faces.png"));
var faceCapture = new FaceCapture(faceImg);
var faceDetector = new FaceDetector();
var face = faceCapture.DetectWith(faceDetector)[0];
var extractedFace = face.Extract();

var irisImg = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));
var irisCapture = new IrisCapture(irisImg, IrisPosition.LEFT_EYE);
var irisDetector = new IrisDetector();
var irisSegment = irisCapture.DetectWith(irisDetector);
var extractedIris = irisSegment.iris.Extract();

var probe = new Applicant();
probe.AddPrint(extractedPrint);
probe.AddFace(extractedFace);
probe.AddIris(extractedIris);

var gallery = new Applicant();
gallery.AddPrint(extractedPrint);
gallery.AddFace(extractedFace);
gallery.AddIris(extractedIris);

var matcher = new ModalitiesVerifyMatcher();
var scores = probe.SimilarWith(gallery, matcher);

System.Console.WriteLine("    prints " + scores.GetPrintScore() + ", faces " +
                         scores.GetFaceScore() + ", irises " +
                         scores.GetIrisScore());
```

1.1.4.5 Store,Load Applicant biometrics representations

Any type of biometric representation does not contains images or other attributes of applicant such as qualities.

Note

It is recommend also to store images for Applicant to be possible to re-extract biometric data when better extractors (by means of speed/accuracy) will be provided.

Modalities ICS Template

It is possible to convert Applicant to ModalitiesICSTemplate. It contains all modalities templates gathered during Applicant enrollment. It can be stored and loaded to be able to use biometric representation of enrolled applicant for print verification or for getting information about minutiae in the future.

Note

ICS Template does not contains even position of fingers/irises templates and it is not possible to use it for multi verification. It is recommend to save position together with template. Or you can use ModalitiesTemplate.

C++

```
Applicant app;
app.AddPrint(extractedPrint);

// Save Applicant as ICS template
auto tmpl = app.ToICSTemplate();
BinaryFile::WriteAll("output/applicant.ics",
                     tmpl->GetModalitiesICSTemplateData());
// Load Applicant as ICS template
auto loadedTmpl =
  ModalitiesICSTemplate(BinaryFile::ReadAll("output/applicant.ics"));
```

java

```
Applicant app = new Applicant();
app.AddPrint(extractedPrint);

// Save Applicant as ICS template
```

```
ModalitiesICSTemplate tmpl = app.ToICSTemplate();
BinaryFile.WriteAll("output/applicant.ics", tmpl.GetModalitiesICSTemplateData());
// Load Applicant as ICS template
ModalitiesICSTemplate loadedTmpl = new ModalitiesICSTemplate(
        BinaryFile.ReadAll("output/applicant.ics"));
```

csharp

```
var app = new Applicant();
app.AddPrint(extractedPrint);

// Save Applicant as ICS template
var tmpl = app.ToICSTemplate();
BinaryFile.WriteAll("output/applicant.ics", tmpl.GetModalitiesICSTemplateData());
// Load Applicant as ICS template
var loadedTmpl =
    new ModalitiesICSTemplate(BinaryFile.ReadAll("output/applicant.ics"));
```

Modalities Template

It is possible to convert Applicant to ModalitiesTemplate. It can be stored and loaded to be able to use enrolled applicants biometric representation for any sdk operations including Identification. It contains all modalities templates gathered during Applicant enrollment. The differences against Modalities ICS Template are:

- It contains also positions of fingers/irises templates.
- It contains data necessary for Identification.

The size is little bit bigger as Modalities ICS Template. The ModalitiesTemplate can be converted to ModalitiesICSTemplate, but ModalitiesICSTemplate cannot be converted into ModalitiesTemplate.

Note

it is recommended to store biometric representation of Applicant in this format.

c++

```
Applicant app;
app.AddPrint(extractedPrint);

// Save Applicant as template
auto tmpl = app.ToTemplate();
BinaryFile::WriteAll("output/applicant.tmpl", tmpl->GetModalitiesTemplateData());
// Load Applicant as template
auto loadedTmpl =
  ModalitiesTemplate::Create(BinaryFile::ReadAll("output/applicant.tmpl"));
```

java

```
Applicant app = new Applicant();
app.AddPrint(extractedPrint);

// Save Applicant as template
ModalitiesTemplate tmpl = app.ToTemplate();
BinaryFile.WriteAll("output/applicant.tmpl", tmpl.GetModalitiesTemplateData());
// Load Applicant as template
ModalitiesTemplate loadedTmpl = ModalitiesTemplate.Create(
        BinaryFile.ReadAll("output/applicant.tmpl"));
```

csharp

```
var app = new Applicant();
app.AddPrint(extractedPrint);

// Save Applicant as template
var tmpl = app.ToTemplate();
BinaryFile.WriteAll("output/applicant.tmpl", tmpl.GetModalitiesTemplateData());
// Load Applicant as template
var loadedTmpl =
    ModalitiesTemplate.Create(BinaryFile.ReadAll("output/applicant.tmpl"));
```

1.1.5 Extraction

The extraction is transformation of biometric sample into template suitable for further processing e.g. Verification, Identification. The library provides extraction for

- Print
- Face
- Iris

The table below lists the enrollment SDK versions and their corresponding template versions. Use this table to determine template compatibility for Verification and Identification according to the rules described in Template Compatibility Rules.

sdk version	print ics	face balanced	face accurate	face accurate server	face fast	iris accurate	iris fast
v28.2.0	2.3	53.1	54.1	-	52.1	10.1	8.1
v28.1.0	2.3	53.1	54.1	-	52.1	-	-
v28.0.0	2.3	53.1	54.1	-	52.1	-	-
v27.5.0	2.3	53.1	54.1	-	52.1	-	-
v27.4.1	2.3	48.1	49.1	-	47.1	-	-
v27.4.0	2.3	48.1	49.1	-	47.1	-	-
v27.3.0	2.3	48.1	49.1	-	47.1	-	-
v27.2.1	2.3	48.1	49.1	-	47.1	-	-
v27.2.0	2.3	48.1	49.1	-	47.1	-	-
v27.1.0	2.3	48.1	49.1	-	47.1	-	-
v27.0.1	2.3	48.1	49.1	-	47.1	-	-
v27.0.0	2.3	48.1	49.1	-	47.1	-	-
v26.4.0	2.3	48.1	49.1	-	47.1	-	-
v26.3.1	2.3	48.1	49.1	-	47.1	-	-
v26.3.0	2.3	48.1	49.1	-	47.1	-	-
v26.2.1	2.3	48.1	49.1	-	47.1	-	-
v26.2.0	2.3	48.1	49.1	-	47.1	-	-
v26.1.0	2.3	48.1	49.1	-	47.1	-	-
v26.0.0	2.3	48.1	49.1	-	47.1	-	-
v25.0.0	2.3	48.1	49.1	-	47.1	-	-
v24.2.0	2.3	48.1	49.1	-	47.1	-	-
v24.1.0	2.3	48.1	49.1	-	47.1	-	-
v24.0.0	2.3	48.1	49.1	-	47.1	-	-
v23.0.1	2.3	48.1	49.1	-	47.1	-	-
v23.0.0	2.3	48.1	49.1	-	47.1	-	-
v22.0.2	2.3	48	49	44	47	-	-
v22.0.1	2.3	48	49	44	47	-	-

sdk version	print ics	face balanced	face accurate	face accurate server	face fast	iris accurate	iris fast
v22.0.0	2.3	48	49	44	47	-	-
v21.0.1	2.3	48	49	44	47	-	-
v21.0.0	2.3	48	49	44	47	-	-
v20.0.1	2.3	1.39	1.33	1.44	1.36	-	-
v20.0.0	2.3	1.39	1.33	1.44	1.36	-	-
v19.1.2	2.3	1.39	1.33	1.44	1.36	-	-
v19.1.1	2.3	1.39	1.33	1.44	1.36	-	-
v19.1.0	2.3	1.39	1.33	1.44	1.36	-	-
v19.0.0	2.3	1.39	1.33	1.44	1.36	-	-
v18.0.2	2.3	1.39	1.33	1.44	1.36	-	-
v18.0.1	2.3	1.39	1.33	1.44	1.36	-	-
v18.0.0	2.3	1.39	1.33	1.44	1.36	-	-
v17.6.2	2.3	1.39	1.33	1.44	1.36	-	-
v17.6.1	2.3	1.39	1.33	1.44	1.36	-	-
v17.6.0	2.3	1.39	1.33	1.44	1.36	-	-
v17.5.1	2.3	1.39	1.33	1.44	1.36	-	-
v17.4.2	2.3	1.39	1.33	1.44	1.36	-	-
v17.4.1	2.3	1.39	1.33	1.44	1.36	-	-
v17.4.0	2.3	1.39	1.33	1.44	1.36	-	-
v17.3.1	2.3	1.39	1.33	1.44	1.36	-	-
v17.3.0	2.3	1.39	1.33	1.44	1.36	-	-
v17.2.2	2.3	1.39	1.33	1.44	1.36	-	-
v17.2.1	2.3	1.39	1.33	1.44	1.36	-	-
v17.2.0	2.3	1.39	1.33	1.44	1.36	-	-
v17.1.1	2.3	1.39	1.33	1.44	1.36	-	-
v17.1.0	2.3	1.39	1.33	1.44	1.36	-	-
v17.0.1	2.3	1.39	1.33	1.44	1.36	-	-
v17.0.0	2.3	1.39	1.33	1.44	1.36	-	-
v16.0.0	2.3	1.39	1.33	1.44	1.36	-	-
v15.0.0	2.3	1.39	1.33	1.44	1.36	-	-
v14.1.0	2.3	1.39	1.33	1.44	1.36	-	-
v14.0.0	2.3	1.39	1.33	1.44	1.36	-	-
v13.0.0	2.3	1.39	1.33	1.44	1.36	-	-
v12.1.0	2.3	1.39	1.33	1.44	1.36	-	-
v12.0.0	2.3	1.39	1.33	1.44	1.36	-	-
v11.1.1	2.3	1.39	1.33	1.44	1.36	-	-
v11.1.0	2.3	1.39	1.33	1.44	1.36	-	-
v11.0.1	2.3	1.39	1.33	1.44	1.36	-	-
v11.0.0	2.3	1.39	1.33	1.44	1.36	-	-
v10.1.1	2.3	1.39	1.33	1.44	1.36	-	-

sdk version	print ics	face balanced	face accurate	face accurate server	face fast	iris accurate	iris fast
v10.1.0	2.3	1.39	1.33	1.44	1.36	-	-
v10.0.0	2.3	1.39	1.33	1.44	1.36	-	-
v9.1.0	2.3	1.39	1.33	1.44	1.36	-	-
v9.0.3	2.3	1.39	1.33	1.44	1.36	-	-
v9.0.2	2.3	1.39	1.33	1.44	1.36	-	-
v9.0.1	2.3	1.39	1.33	1.44	1.36	-	-
v9.0.0	2.3	1.39	1.33	1.37	1.36	-	-
v8.0.0	2.3	1.39	1.33	1.37	1.36	-	-
v7.0.2	2.3	1.39	1.33	1.37	1.36	-	-
v7.0.1	2.3	1.39	1.33	1.37	1.36	-	-
v7.0.0	2.3	1.39	1.33	1.37	1.36	-	-
v6.0.0	2.3	1.39	1.33	1.37	1.36	-	-
v5.0.0	2.3	1.39	1.33	1.37	1.36	-	-
v4.0.0	2.3	1.39	1.33	1.37	1.36	-	-
v3.4.0	2.3	1.39	1.33	1.37	1.36	-	-
v3.3.6	2.3	1.39	1.33	1.37	1.36	-	-
v3.3.5	2.3	1.39	1.33	1.37	1.36	-	-
v3.3.4	2.3	1.39	1.33	1.37	1.36	-	-
v3.3.3	2.3	1.39	1.33	1.37	1.36	-	-
v3.3.2	2.3	1.39	1.33	1.37	1.36	-	-
v3.3.1	2.3	1.39	1.33	1.37	1.36	-	-
v3.3.0	2.3	1.39	1.33	1.37	1.36	-	-
v3.2.0	2.3	1.39	1.33	1.37	1.36	-	-
v3.1.2	2.3	1.39	1.33	1.37	1.36	-	-
v3.1.1	2.3	1.39	1.33	1.37	1.36	-	-
v3.0.1	2.3	1.39	1.33	1.37	1.36	-	-
v3.0.0	2.3	1.39	1.33	1.37	1.36	-	-
v2.0.0	2.3	1.39	1.33	1.37	1.36	-	-
v1.0.0	2.3	1.39	1.33	1.37	1.36	-	-

1.1.5.1 Template Compatibility Rules

1.1.5.1.1 Face Templates

Enrollment SDK versions up to and including 27.4.1

Face templates generated in different modes (Fast / Balanced / Accurate) are **not mutually compatible**. Templates can be used for verification and identification **only if they have the same template version**.

Enrollment SDK version 27.5.0 and later

Face templates generated in Fast / Balanced / Accurate modes are **mutually compatible** and can be used interchangeably for verification and identification.

Although each mode produces a different template version number, these templates belong to the same compatibility group and can be matched together.

This compatibility remains valid across different enrollment SDK versions, as long as they produce templates belonging to the same compatibility group.

If a future enrollment SDK version introduces a new compatibility group, templates from that group are **mutually compatible with each other**, but **not compatible with templates from previous compatibility groups**.

1.1.5.1.2 Print Templates

Print modality templates can be used for verification or identification **only if they have the same template version**. **No cross-version compatibility is supported** for print templates.

1.1.6 Verification

1.1.6.1 1 to 1 Verification

The library compares biometric samples of each modality and calculates their similarity scores. The result of verification is similarity score. The library provides verification for

- Print - provides PrintSimilarityScore
- Face - provides FaceSimilarityScore
- Iris - provides IrisSimilarityScore

1.1.6.2 1 to 1 Modalities Verification

Use function ModalitiesVerifyMatcher::Match(). It takes previously stored or actually created (from current enrolling Applicant) ModalitiesTemplate. The result of verification is ModalitiesScore that provides scores per each modality

- print - print templates on same biometric position in probe and gallery are compared with each other. The provided score is average of maximal score per biometric position. A template can have also unknown position e.g when it was loaded with no biometric position info. The unknown position is treated as special biometric position. When the probe template has unknown position it is compared with all gallery templates. When the gallery template has unknown position then each probe template with any position is compared against this template.
- face - each probe face template is compared with each gallery face template. The maximal score of all scores is provided.
- iris - the calculation is same as print calculation.

1.1.7 Identification

The SDK provides IndexedModalitiesGallery and ModalitiesGallery for identification. They differ in usage convenience and performance.

1.1.7.1 IndexedModalitiesGallery

Note

> The number of references in any IndexedModalitiesGallery is limited by license. You can have more created galleries in your code. The number limits overall references in all galleries.

The IndexedModalitiesGallery stores only the data necessary for identification. Relevant data from ModalitiesTemplate is copied into the gallery using the IndexedModalitiesGallery::Add() function.

The IndexedModalitiesGallery::IdentifyModalities() function returns reference index identifiers to the user.

Adding data to the gallery can be faster in higher-level language bindings and consumes less memory, as only the essential identification data is stored compared to ModalitiesGallery. However, the user must maintain a mapping between their reference representation and the identifier provided to IndexedModalitiesGallery::Add().

It is recommended to use IndexedModalitiesGallery for performance- and memory-critical use cases and to prefer it over the more convenient ModalitiesGallery.

C++

```
// Recreate your own data types from data stored during enrollment
auto templates =
  std::unordered_map<size_t, std::shared_ptr<ModalitiesTemplate>{};
templates.emplace(
  1, ModalitiesTemplate::Create(BinaryFile::ReadAll("output/1.tmpl")));
templates.emplace(
  2, ModalitiesTemplate::Create(BinaryFile::ReadAll("output/2.tmpl")));

// Create a gallery and add users.
auto gallery = IndexedModalitiesGallery::Create({});
gallery->Add(1, templates[1]);
gallery->Add(2, templates[2]);
auto probe = templates[1];

// Identify a probe in the gallery.
constexpr size_t MaxCandidates = 2; // size of result set of candidates
auto result = gallery->IdentifyModalities(probe, MaxCandidates);

// Output identification results.
for (const auto& id : result->GetPrintIdentifiers())
{
    std::cout « "   Found Id is " « id « std::endl;
}
```

java

```
// Recreate your own data types from data stored during enrollment
HashMap<Long, ModalitiesTemplate> templates
        = new HashMap<Long, ModalitiesTemplate>();
templates.put(1L, ModalitiesTemplate.Create(BinaryFile.ReadAll("output/1.tmpl")));
templates.put(2L, ModalitiesTemplate.Create(BinaryFile.ReadAll("output/2.tmpl")));
ModalitiesTemplate probe = templates.get(1L);

// Create a gallery and add users.
IndexedModalitiesGallery gallery = IndexedModalitiesGallery.Create(
        new IndexedModalitiesGalleryConfig());
gallery.Add(1, templates.get(1L));
gallery.Add(2, templates.get(2L));

// Identify a probe in the gallery.
long maxCandidates = 2; // size of result set of candidates
IndexedModalitiesGallery.ModalitiesIdentifiers result = gallery.IdentifyModalities(
        probe, maxCandidates);

// Output identification results.
for (long id : result.GetPrintIdentifiers()) {
```

```
    System.out.println("   Found Id is " + id);
}
```

csharp

```
// Recreate your own data types from data stored during enrollment
var templates = new Dictionary<uint, ModalitiesTemplate>();
templates.Add(1, ModalitiesTemplate.Create(BinaryFile.ReadAll("output/1.tmpl")));
templates.Add(2, ModalitiesTemplate.Create(BinaryFile.ReadAll("output/2.tmpl")));
var probe = templates[1];

// Create a gallery and add users.
var gallery =
    IndexedModalitiesGallery.Create(new IndexedModalitiesGalleryConfig());
gallery.Add(1, templates[1]);
gallery.Add(2, templates[2]);

// Identify a probe in the gallery.
var maxCandidates = 2U; // size of result set of candidates
var result = gallery.IdentifyModalities(probe, maxCandidates);

// Output identification results.
foreach (uint id in result.GetPrintIdentifiers())
{
    System.Console.WriteLine("   Found Id is " + id);
}
```

Once the candidates are identified, you can use ModalitiesGallery::Consolidate() static function to get unique list of candidates with scores.

c++

```
// Create print candidates for consolidation
auto printCandidates = std::vector<std::shared_ptr<ModalitiesGalleryReference»{};
for (const auto& id : result->GetPrintIdentifiers())
{
    printCandidates.push_back(
      std::make_shared<ModalitiesGalleryIndexedTemplate>(id, templates[id]));
}
// Create type that holds all necessary information for consolidation.
auto consolidationCandidates =
  std::make_shared<ModalitiesGallery::ModalitiesCandidates>(
    std::move(printCandidates),
    std::vector<std::shared_ptr<ModalitiesGalleryReference»{},
    std::vector<std::shared_ptr<ModalitiesGalleryReference»{},
    probe);
// Consolidate
auto candidates = ModalitiesGallery::Consolidate(consolidationCandidates, 2);
// Output identification results.
for (const auto& c : candidates)
{
    std::cout « "   Id is " « c.GetCandidate()->GetIndex() « " score is ["
              « c.GetScore().GetPrintScore() « ", "
              « c.GetScore().GetFaceScore() « ", "
              « c.GetScore().GetIrisScore() « "]" « std::endl;
}
```

java

```
// Create print candidates for consolidation
ArrayList<ModalitiesGalleryReference> printCandidates
        = new ArrayList<ModalitiesGalleryReference>();
for (long id : result.GetPrintIdentifiers()) {
    printCandidates.add(
            new ModalitiesGalleryIndexedTemplate(id, templates.get(id)));
}
// Create type that holds all necessary information for consolidation.
ModalitiesGallery.ModalitiesCandidates consolidationCandidates
        = new ModalitiesGallery.ModalitiesCandidates(printCandidates,
                new ArrayList<ModalitiesGalleryReference>(),
                new ArrayList<ModalitiesGalleryReference>(), probe);
// Consolidate
ArrayList<ModalitiesGallery.Candidate> candidates = ModalitiesGallery.Consolidate(
        consolidationCandidates, maxCandidates);
// Print result
for (ModalitiesGallery.Candidate c : candidates) {
    System.out.println("   Id is " + c.GetCandidate().GetIndex() + " score is ["
                       + c.GetScore().GetPrintScore() + ", "
                       + c.GetScore().GetFaceScore() + ", "
                       + c.GetScore().GetIrisScore() + "]");
}
```

csharp

```
// Create print candidates for consolidation
var printCandidates = new List<ModalitiesGalleryReference>();
foreach (uint id in result.GetPrintIdentifiers())
{
    printCandidates.Add(new ModalitiesGalleryIndexedTemplate(id, templates[id]));
}
// Create type that holds all necessary information for consolidation.
var consolidationCandidates = new ModalitiesGallery.ModalitiesCandidates(
    printCandidates, new List<ModalitiesGalleryReference>(),
    new List<ModalitiesGalleryReference>(), probe);
// Consolidate
var candidates = ModalitiesGallery.Consolidate(consolidationCandidates, 2);
// Print result
foreach (ModalitiesGallery.Candidate c in candidates)
{
    System.Console.WriteLine("   Id is " + c.GetCandidate().GetIndex() +
                             " score is [" + c.GetScore().GetPrintScore() +
                             ", " + c.GetScore().GetFaceScore() + ", " +
                             c.GetScore().GetIrisScore() + "]");
}
```

1.1.7.2 ModalitiesGallery

Note

> The number of references in any ModalitiesGallery is limited by license. You can have more created galleries in your code. The number limits overall references in all galleries.

The ModalitiesGallery stores ModalitiesGalleryReference objects for later identification. Adding data to this gallery in higher-level language bindings is less efficient than in IndexedModalitiesGallery. This inefficiency arises because it creates objects derived from C++ interfaces while also maintaining references to the original high-level language objects.

These high-level language references are necessary because objects returned from the C++ implementation lose their original type information. To address this, the gallery remaps types based on the ID from ModalitiesGalleryReference. However, this approach consumes more memory, as it stores both the identification related data and a reference to the user object.

Despite the increased memory usage, this gallery is more convenient to use as IndexedModalitiesGallery. Since the entire user object is stored within the gallery, users do not need to manually map objects to their identifiers in their code or reload ModalitiesTemplate for consolidation. However, users should be mindful of their object size. The object itself is not required for IdentifyModalities, only the Index and ModalitiesTemplate provided by ModalitiesGalleryReference are used. The relevant identification data is copied from ModalitiesTemplate.

Once a ModalitiesGalleryReference is added to the gallery, its associated ModalitiesTemplate is no longer relevant until consolidation. If memory usage is a critical concern, users can implement ModalitiesGalleryReference::GetModalitiesTemplates() in such a way that the function loads the template on each call instead of storing the object in memory or use IndexedModalitiesGallery.

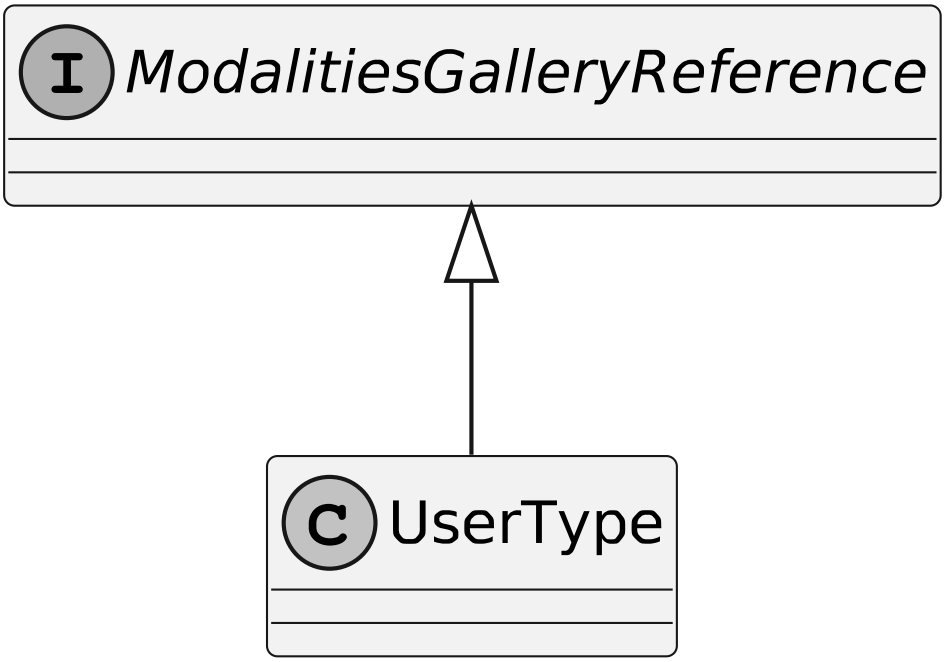
Each ModalitiesGalleryReference contains a ModalitiesTemplate used for identification. The ModalitiesGallery::Identify() function returns this reference to the SDK user, who can then downcast it to their custom type to retrieve additional information.

Possible approaches for adding custom types to the gallery include:

- **Inheritance**: Extend ModalitiesGalleryReference to include custom data.
- **Composition**: Store a reference to a custom object within ModalitiesGalleryReference.

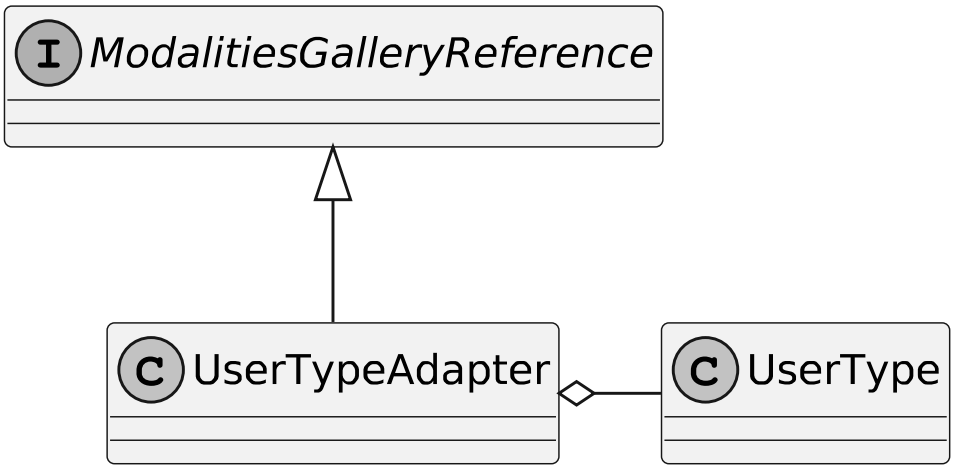
Inheritance

You can inherit our ModalitiesGalleryReference in your types.



Composition

You can embed your type into adapter of ModalitiesGalleryReference.



The identification (ModalitiesGallery::Identify()) has two phases

- searching in gallery for best candidates per modality (details in ModalitiesGallery::IdentifyModalities()).
- merging lists of candidates per modality into final candidates list (details in ModalitiesGallery::Consolidate()).

The enrollment SDK provides following types of gallery

- Print identification based on ICSTemplate

c++

```
// Recreate your own data types from data stored during enrollment
auto johnTemplate =
  ModalitiesTemplate::Create(BinaryFile::ReadAll("output/john.tmpl"));
auto john = std::make_shared<UserType>(0, "john", johnTemplate);
auto markTemplate =
  ModalitiesTemplate::Create(BinaryFile::ReadAll("output/mark.tmpl"));
auto mark = std::make_shared<UserType>(1, "mark", markTemplate);

// Create a gallery and add users.
auto gallery = ModalitiesGallery::Create();
gallery->Add(john);
gallery->Add(mark);

// Identify a probe in the gallery.
constexpr size_t MaxCandidates = 2; // size of result set of candidates
auto result = gallery->Identify(johnTemplate, MaxCandidates);

// Output identification results.
for (const auto& c : result)
{
    auto user = std::dynamic_pointer_cast<UserType>(c.GetCandidate());
    std::cout « "   Name is " « user->name « " score is ["
              « c.GetScore().GetPrintScore() « ", "
              « c.GetScore().GetFaceScore() « ", "
              « c.GetScore().GetIrisScore() « "]" « std::endl;
}
```

java

```
// Recreate your own data types from data stored during enrollment.
ModalitiesTemplate johnTemplate = ModalitiesTemplate.Create(
        BinaryFile.ReadAll("output/john.tmpl"));
UserType john = new UserType(0, "john", johnTemplate);
ModalitiesTemplate markTemplate = ModalitiesTemplate.Create(
        BinaryFile.ReadAll("output/mark.tmpl"));
UserType mark = new UserType(1, "mark", markTemplate);

// Create a gallery and add users.
ModalitiesGallery gallery = ModalitiesGallery.Create();
gallery.Add(john);
gallery.Add(mark);

// Identify a probe in the gallery.
long maxCandidates = 2; // size of result set of candidates
ArrayList<ModalitiesGallery.Candidate> result = gallery.Identify(
        johnTemplate, maxCandidates);

// Output identification results.
for (ModalitiesGallery.Candidate c : result) {
    UserType user = (UserType) c.GetCandidate();
    System.out.println("   Name is " + user.name + " score is ["
                       + c.GetScore().GetPrintScore() + ", "
                       + c.GetScore().GetFaceScore() + ", "
                       + c.GetScore().GetIrisScore() + "]");
}
```

csharp

```
// Recreate your own data types from data stored during enrollment
var johnTemplate =
    ModalitiesTemplate.Create(BinaryFile.ReadAll("output/john.tmpl"));
var john = new UserType(0, "john", johnTemplate);
var markTemplate =
    ModalitiesTemplate.Create(BinaryFile.ReadAll("output/mark.tmpl"));
var mark = new UserType(1, "mark", markTemplate);

// Create a gallery and add users.
var gallery = ModalitiesGallery.Create();
gallery.Add(john);
gallery.Add(mark);

// Identify a probe in the gallery.
var maxCandidates = 2U; // size of result set of candidates
var result = gallery.Identify(johnTemplate, maxCandidates);

// Output identification results.
foreach (ModalitiesGallery.Candidate c in result)
{
    var user = c.GetCandidate() as UserType;
    System.Console.WriteLine("   Name is " + user.name + " score is [" +
                             c.GetScore().GetPrintScore() + ", " +
                             c.GetScore().GetFaceScore() + ", " +
                             c.GetScore().GetIrisScore() + "]");
}
```

1.1.8 Similarity Score

Due to variations in positioning, deformations, lighting conditions, sensor characteristics, and biometric-specific factors such as facial expressions, eye occlusion, finger humidity, or random noise, it is impossible for two biometric samples of the same person, acquired in different sessions, to match exactly. To address this, a matching algorithm calculates a global similarity score between two biometric samples. This similarity score is then compared to a predefined similarity threshold. If the similarity score exceeds the threshold, the system considers the two samples to be from the same individual. However, biometric systems are not perfect, and matching results can sometimes be incorrect. The two primary system errors are measured in terms of:

- FAR (False Acceptance Rate) - The frequency of fraudulent accesses due to impostors claiming a false identity
- FRR (False Rejection Rate) - The frequency of rejections relative to people who should be correctly verified

FAR and FRR depend on the similarity threshold, which is set to obtain the desired security level. FAR and FRR are strictly related to each other. More precisely, FRR is an increasing function of the similarity threshold and FAR is a decreasing function of the similarity threshold. If the similarity threshold setting is increased to make it harder for impostors to gain access (lower FAR), some authorized people may find it harder to gain access (higher FRR).

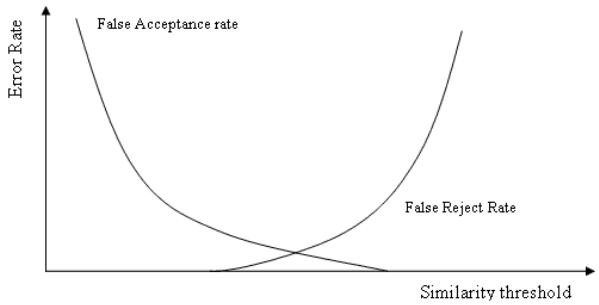


Figure 1 FAR/FRR

The range of returned similarity scores is from 0 (no similarity) to 1000 (highest similarity) for print and iris modality templates, and from 0 to 100 for face templates (see Modality Similarity Score). The correspondence between biometric templates is generally indicated by similarity scores greater than a default similarity threshold, which varies by modality (e.g., 40 for prints, which roughly corresponds to FAR = 10^-4). For more information, please refer to FAR and Threshold (see False Acceptance Rate).

1.1.8.1 Modality Similarity Score

Each biometric modality has its own similarity score scale and progression characteristics:

1.1.8.1.1 Print Similarity Score

It is similarity score between print Templates.

Score is 0-1000 where 1000 represents most similar templates.

The correspondence between print templates is generally reflected by similarity scores greater than default similarity threshold (40). For more information please refer to FAR and Threshold (see False Acceptance Rate). Default similarity threshold value (40) roughly corresponds to FAR=10^-4. Depending on the type of scanner, the size of print images, the size of the database and the desired security level of your application, you may need to select a different threshold.

See Similarity Score.

1.1.8.1.2 Iris Similarity Score

It is similarity score between iris Templates.

Score is 0-1000 where 1000 represents most similar templates.

Scores in range <0, 60> are computed based on FAR analysis. Scores above this range are approximated with linear slope to 1000. Therefore very similar irises can have matching score near maximal 1000.

See Similarity Score.

1.1.8.1.3 Face Similarity Score

It is similarity score between face Templates.

Score is 0-100 where 100 represents most similar templates.

When the matching score is higher then certain score threshold then the face images belongs to the same person with high probability. The matching score range is <0, 100>. Its values can be interpreted as follows:

- Low values of the score, i.e. range <0, 60>, are normalized using FAR values and this formula `score_L=-10*log(FAR)`. It means that score 30 is related to FAR=1:1000=10^-3, score 50 is related to FAR=1:100000=10^-5 (evaluated on our large testing non-matching pairs dataset).
- High values of the score, i.e. range <80, 100>, are normalized using FRR values and this formula `score_H=100/3*(FRR + 2)`. It means that score 80 is related to FRR=0.4, score 90 is related to FRR=0.7 (evaluated on our large testing matching pairs dataset).

Scores values in range (60, 80) are weighted average of `score_L` and `score_H`. This normalization help the users to select the score threshold according their needs. If it is too low e.g. score threshold is 30, then the chance of false accepted non-matching faces is quite high (FAR=10^-3). When it is too high e.g. score threshold is 90, then the chance of false rejected matching faces is quite high (FRR=0.7). For face recognition following recommendations are given:

- minimum face size - inter eye-pupils distance at least 40 pix
- optimal face size - inter eye-pupils distance at least 100 pix
- optimal image - sharp, contrast, frontal facial image with standard lighting conditions
- the more similar are conditions when the matched faces are captured the higher is the matching accuracy
- complex robust feature extraction invariant to partial faces occlusions / poor lighting / blurriness / etc is done. So, no further preprocessing is necessary.

See Similarity Score.

1.1.8.2 False Acceptance Rate

False Acceptance Rate (FAR) describes probability at which matching algorithm makes false acceptance errors. Similarity scores returned by this library are normalized. More precisely, approximate relationship between similarity score and False Acceptance Rate (sometimes called also False Match Rate) is given by the following formula:

$$similarityscore = -10 * \log_{10}(FAR)$$

Please note that the normalization formula is only approximation. Score distribution can vary depending on various factors such as scanner surface area, finger placement, finger quality, finger positions (index fingers tend to give higher scores compared to little fingers, etc..)

Note

> False Acceptance Rate (FAR) is one of several accuracy ratings that can be calculated for matching algorithm. False acceptance error is when matching algorithm identifies matching print in database for some probe print even though the two prints do not belong to the same person. FAR depends predominantly on the threshold selected. Increasing threshold reduces FAR, but it can also increase FRR see Similarity Scores. FAR is influenced by quality of the database being searched and quality of the probe print.

1.1.8.3 False Rejection Rate

FRR describes probability at which matching algorithm makes false rejection errors.

Note

> False Rejection Rate (FRR) is one of several accuracy ratings that can be calculated for matching algorithm. False rejection error is when matching algorithm fails to identify a probe user (print, face or iris) in database even though the database contains user with biometric data (print, face or iris) of the same person. FRR primarily depends on quality of the database being searched and quality of probe biometric data. FRR is somewhat influenced by threshold. Decreasing threshold reduces FRR, but it can increase FAR significantly.

1.1.8.4 Similarity Threshold

Similarity threshold defines minimum similarity score that is considered a match.

Note

> While similarity score provides degree of certainty information that is in line with probabilistic algorithms of biometrics, most applications eventually have to decide whether particular similarity score means a match or not. Similarity threshold is used to divide the similarity spectrum into matching and non-matching range. Default threshold setting is suitable for most applications. You can adjust the threshold to control FAR and FRR. There is an approximate mapping between FAR and threshold (see False Acceptance Rate).

1.1.9 Quality

The library provides fingerprint/iris/face qualities for evaluation of a image whether is suitable for further processing.

- Print
- Iris
- Face

1.1.10 Server

The SDK can use provided server to execute its functionality. To use server you need to run server `enrolment_server endpoint` e.g. `enrollment_server 127.0.0.1:50051`. Then it is necessary to use server executor in appropriate classes or setup global executor.
Example of usage server executor in class constructor:

c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/face.png"));
auto fc = FaceCapture::Create(std::move(img));
auto executor = std::make_shared<GrpcFaceExecutor>("127.0.0.1:50051");
FaceDetector fd(executor);
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));
FaceCapture capture = new FaceCapture(img);
GrpcFaceExecutor executor = new GrpcFaceExecutor("127.0.0.1:50051");
FaceDetector detector = new FaceDetector(executor);
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));
FaceCapture capture = new FaceCapture(img);
GrpcFaceExecutor executor = new GrpcFaceExecutor("127.0.0.1:50051");
FaceDetector detector = new FaceDetector(executor);
```

Example of usage global server executor:

c++

```
GlobalFaceExecutor::Set(std::make_shared<GrpcFaceExecutor>("127.0.0.1:50051"));
auto img = Image::Decode(BinaryFile::ReadAll("assets/face.png"));
auto fc = FaceCapture::Create(std::move(img));
FaceDetector fd;
```

java

```
FaceExecutor executor = new GrpcFaceExecutor("127.0.0.1:50051");
GlobalFaceExecutor.Set(executor);
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));
FaceCapture capture = new FaceCapture(img);
FaceDetector detector = new FaceDetector();
```

csharp

```
FaceExecutor executor = new GrpcFaceExecutor("127.0.0.1:50051");
GlobalFaceExecutor.Set(executor);
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));
FaceCapture capture = new FaceCapture(img);
FaceDetector detector = new FaceDetector();
```

1.2 Design

1.2.1 Header and Binary parts

The library is has header part and binary part. The interface to binary is "C" due to better compatibility across different compilers. The headers forwards functionality to binary, when we need to hide implementation details. The interface between headers and binary can be

- functions
- primitive data types and its pointers
- plain old structures with pragma pack 1 for compatibility between binary and compiled client code. We do not know what pragma pack is used/available in client compiler. We use 1 that is implemented in each compiler and architecture even if it is not optimal.

We are using C++20 standard. In binary part the pointer to implementation is casted to appropriate objects, and we can work with inheritance. The header part is used only to define API and transform error to exceptions.

The disadvantage is that we need copy data created in binary into vector to use safe containers. Later we can implement something similar to std::span or std::string_view.

1.2.2 C++

- We use `SWIG tool` code generator to convert C++ classes into Java / C# classes.
- We can write safe library due to containers, smart pointers and RAII principle.
- We use open/closed principle to extend our library with client implementation e.g logging, serialization.

1.2.2.1 Disadvantage

- Creating a library with a C++ interface is not feasible due to complications with the STL, `#pragma pack` directives, and various other issues. Therefore, we use a "C" interface for the library.
- The C wrapper MUST be implemented when we will need some other language as Java / C#
- It is not possible to use multiple interface inheritance, because swig implements interfaces as usual implementation classes due to director feature. Then Ruby and C# are not able to inherit multiple implementation classes.
- It is not possible to create nested classes, because SWIG does not support it for Ruby

1.2.3 Quality

To achieve better quality we use

- clang-format for formatting code
- clang-tidy for static analysis
- clang sanitizers for dynamic analysis
  - address sanitizer for detecting memory leaks and overflows
  - memory sanitizer for detecting use of unutilized memory
  - thread sanitizer for detecting race conditions
  - undefined sanitizer for detecting undefined behavior
- `valgrind` for unit, integration tests, and c++ example application
- `ABI compliance checker` to know impact of our changes regarding to previous SDK versions.
- unit tests with Google test framework
- warnings are set to the strictest level and treated as error
- CI pipeline for various architectures
- accurate documentation generated by doxygen
- C++ because of automatic memory management using smart pointers and RAII idiom

1.2.4 Extending SDK

The SDK can be extended by implementation of provided interfaces. See Extending SDK for list of customizable interfaces.

2 Version History and Legal Notices

Version	Date	
v28.2.0	2026-05-28	**Fixed** • [BSDK-614] - Fixed `remove()` silently doing nothing on face, iris, and print template lists in Java, C#, and Python bindings (Added `operator==` and `Hash()` to `FaceTemplate`, `IrisTemplate`, and `PrintTemplate`). • [BSDK-609] - Fixed a crash during parallel face and iris template verification in `FaceTemplate.SimilarWith` and `IrisTemplate.SimilarWith` by skipping iFace verification and using inno-gallery for verification. **Added** • [BSDK-606] - New iris detector modes `IrisDetector::Config::Type::ACCURATE` (previously default) and `IrisDetector::Config::Type::FAST`. • [BSDK-606] - New iris extraction mode `IrisExtractorConfig::Type::FAST`. • [BSDK-606] - Manual SDK initialization with explicit OpenCV and ONNX Runtime thread configuration. Set `INNO_ENROLLMENT_SDK_MANUAL↩_INITIALIZATION=1` before loading the library, then call `inno::↩Initialize(InitializerConfig)` to fine-control license, models path, and threading before library starts up. • [BSDK-606] - `inno::DefaultModelsDirectory()` returning the default IFace models path (`iface/models` next to the loaded library) for manual initialization. For more information see the *Initialization* section in the documentation. • [BSDK-601] - Added `HardwareIdMethod::ARM_ANDROID_MAC_ADDR` enum value to support generating a MAC-address-based licensing ID via the `HardwareID()` function (ARM Linux and Android including x86/x86_64 simulators only). **Changed** • Java samples are now maven project **Dependencies** • iface v6.27.0 (balanced 53.1, fast 52.1, accurate 54.1, iris-accurate 10.1, iris-fast 8.1) • iseglib v4.3.3 • iengine v3.16.3 (ics 2.3) • ifinger v1.44.0 (embedding 1735835209.1)

Version	Date	
v28.1.0	2026-05-11	**Fixed** • Fixed documentation errors and typos **Changed** • Onnxruntime is updated from 1.13.1 to 1.18.1 version (BSDK-256) • `OFIQ v1.1.2` is used via IFace to have one implementation with known thresholds (BSDK-513) **Dependencies** • iface v6.27.0 (balanced 53.1, fast 52.1, accurate 54.1, iris 10.1) • iseglib v4.3.3 • iengine v3.16.3 (ics 2.3) • ifinger v1.44.0 (embedding 1735835209.1)
v28.0.0	2026-03-26	**Breaking Change** • Enum value `TiffImageEncoder::Compression::JBIG` has been removed due to a licensing issue. TIFF images with `jbig` compression are no longer supported. (BSDK-482) **Changed** • Solver references now use full type names for improved clarity. • Restricted `eigen` to MPL-2.0-only modules to avoid LGPL-3.0 obligations (BSDK-482) **Dependencies** • iface v6.24.0 (balanced 53.1, fast 52.1, accurate 54.1, iris 10.1) • iseglib v4.3.2 • iengine v3.16.3 (ics 2.3) • ifinger v1.44.0 (embedding 1735835209.1)

Version	Date	
v27.5.0	2026-03-06	**Fixed** • Added missing JavaDoc for SWIG-generated Java container classes (`std::vector` wrappers). • If a Java object is used after it has been closed, the JVM will not crash. Instead, a `NullPointerException` or `RuntimeException` will be thrown. • Added third party licenses into package (BSDK-481) **Added** • Integrated Face Image Quality assessment using OFIQ open source library (BSDK-425) **Changed** • Face templates extracted by `FaceExtractorConfig::Type::FAST`, `FaceExtractorConfig::Type::BALANCED`, and `FaceExtractorConfig::Type::ACCURATE` modes are now mutually compatible **Dependencies** • iface v6.24.0 (balanced 53.1, fast 52.1, accurate 54.1, iris 10.1) • iseglib v4.3.2 • iengine v3.16.3 (ics 2.3) • ifinger v1.44.0 (embedding 1735835209.1)
v27.4.1	2026-01-26	**Changed** • Documentation layout is improved • Automatically upload Enrollment SDK to the Customer Portal on release (BSDK-18) **Dependencies** • iface v6.22.2 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.2 • iengine v3.16.3 (ics 2.3) • ifinger v1.44.0 (embedding 1735835209.1)

Version	Date	
v27.4.0	2025-12-15	**Fixed** • Fixed `FaceDetector::Detect` and `FaceCapture::DetectWith` returning many invalid faces when using the `FAST` detector and the `minFaceSize` configuration parameter was set to 0.01 or lower. (BSDK-241) **Changed** • Interval of `Face::BackgroundUniformity` in ICAO validation changed to <-1000, 10000>. (BSDK-357) **Dependencies** • iface v6.22.2 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.2 • iengine v3.16.3 (ics 2.3) • ifinger v1.44.0 (embedding 1735835209.1)

Version	Date	

v27.3.0	2025-11-25	**Added**
		• Added function `ModalitiesGallery::Remove` into Java and C# bindings (BSDK-363).
		• Enabled `Valgrind` checks for unit tests, integration tests, and the C++ example. All `memcheck reports` are treated as errors **except** leak kinds `Possibly lost` and `Still reachable`.
		• Added `ABI compliance checker` to know impact of our changes regarding to previous SDK versions.
		
		Fixed
		• Fixed an issue where creating a `ModalitiesGallery` or `IndexedModalitiesGallery` before the enrollment SDK was initialized prevented adding references to the gallery (BSDK-362).
		• Fixed an issue where `IndexedModalitiesGallery::IdentifyModalities` (for gallery type `IndexedModalitiesGalleryType::PRINT_ICS_TEMPLATE`) could return previously deleted IDs for the print modality when the requested maximum number of candidates exceeded the number of valid (non-deleted) print modality references in the gallery.
		• Fixed an issue where `ModalitiesGallery::IdentifyModalities` and `ModalitiesGallery::Identify` (for gallery type `ModalitiesGalleryType::PRINT_ICS_TEMPLATE`) could crash when the requested maximum number of candidates exceeded the number of valid (non-deleted) print modality references in the gallery.
		• Documentation:
		  – Improved HTML documentation layout and links.
		  – Fixed incorrectly generated images in the PDF documentation.
		• Updated all language samples to use the static variant of the function `SlapCapture::GuessHandPosition`.
		
		Changed
		• `Swig` is updated to version 4.4.0.
		• A single dependency tree is managed by Conan. ISeglib is included as a dependency due to an extension introduced in IEngine 3.16.3.
		
		Dependencies
		• iface v6.18.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1)
		• iseglib v4.3.2
		• iengine v3.16.3 (ics 2.3)
		• ifinger v1.44.0 (embedding 1735835209.1)

Version	Date	
v27.2.1	2025-09-30	**Changed** • The Java `Bytes` class constructor (for `byte[]`) and its getter are now faster, as data is copied between Java and native memory in a single operation. • Documentation layout is improved • Android sample is updated (BSDK-280) – target `API level 36` – use gradle 8.14.3 – split APK according to architectures **Dependencies** • iface v6.18.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.0 • iengine v3.10.2 (ics 2.3) • ifinger v1.44.0 (embedding 1735835209.1)
v27.2.0	2025-08-12	**Changed** • All android libraries now use `16 kB page sizes`, as mandated by Google for architectures `x86_64` and `arm64-v8a` (BSDK-220) **Dependencies** • iface v6.18.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.0 • iengine v3.10.2 (ics 2.3) • ifinger v1.44.0 (embedding 1735835209.1)

Version	Date	
v27.1.0	2025-07-29	**Added** • New function `ModalitiesGallery::Remove` and `IndexedModalitiesGallery::Remove` for removing user from the gallery (BSDK-208) **Changed** • Android libraries **except IFace** now use `16 kB page sizes`, as mandated by Google for architectures `x86_64` and `arm64-v8a`. The Android NDK has been upgraded to version `r28b`, which produces binaries with a 16 kB page size by default. • The matching algorithm for ICS Identification was changed from the previously used fastest algorithm (now obsolete) to the default one **Dependencies** • iface v6.14.1 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.0 • iengine v3.10.2 (ics 2.3) • ifinger v1.44.0 (embedding 1735835209.1)
v27.0.1	2025-06-11	**Fixed** • Identification now correctly accepts a maximum candidate count that is bigger than the gallery size without throwing an exception (BSDK-176) **Dependencies** • iface v6.14.1 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.0 • iengine v3.7.0 (ics 2.3) • ifinger v1.35.1 (embedding 1735835209.1)

Version	Date	
v27.0.0	2025-05-16	**Breaking Change** • The passive liveness type `Face::PassiveLivenessType::COMBINED` has been replaced by a single model offering better accuracy and similar performance (BSDK-93) – The class `SelfieCombinedPassiveLivenessDetector` is no longer needed to achieve accurate passive liveness results. Use `FaceDetector` instead – To maintain meaningful configuration, `Face::PassiveLivenessType::COMBINED` has been renamed to `Face::PassiveLivenessMode::FAST`, and `Face::PassiveLivenessType::UNIVERSAL` has been renamed to `Face::PassiveLivenessMode::ACCURATE` **Removed** • Class `SelfieCombinedPassiveLivenessDetector`, as it is no longer required for accurate passive liveness calculation (BSDK-93) • Enum `Face::PassiveLivenessType`, replaced by `Face::PassiveLivenessMode` (BSDK-93) **Fixed** • Exceptions thrown by functions returning enums are now correctly propagated to Java without causing a segmentation fault **Dependencies** • iface v6.14.1 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.0 • iengine v3.7.0 (ics 2.3) • ifinger v1.35.1 (embedding 1735835209.1)

Version	Date	
v26.4.0	2025-05-16	**Fixed** • Transparency is now correctly preserved during image decoding (BSDK-69) **Added** • Added consolidation after identification based ONLY on print embedding templates using `EmbeddingVerifyMatcher` or `ModalitiesEmbeddingVerifyMatcher`(BSDK-83) **Dependencies** • iface v6.12.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.0 • iengine v3.7.0 (ics 2.3) • ifinger v1.35.1 (embedding 1735835209.1)
v26.3.1	2025-05-12	**Fixed** • Fixed a crash in the `PRO` and `PRINT` variants on Windows machines with CPUs supporting the VNNI instruction set. **Changed** • Binary size reduced by omitting unused NN models **Dependencies** • iface v6.12.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.0 • iengine v3.7.0 (ics 2.3) • ifinger v1.35.1 (embedding 1735835209.1)

Version	Date	
v26.3.0	2025-05-02	**Added** • Configurable speed/accuracy modes for `Face::Age` and `Face::Gender` (BSDK-84) **Changed** • String representation of enrollment exceptions now includes the full location of the function, file, and line where the error occurred (BSDK-12) • IFace version is upgraded to 6.12.0 to provide age/gender accuracy modes (BSDK-84) – SDK provides same universal passive liveness scores as `DIS 1.46.1`. **Dependencies** • iface v6.12.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.0 • iengine v3.1.0 (ics 2.3) • ifinger v1.22.1 (embedding 1678264171.1)
v26.2.1	2025-03-21	**Fixed** • Fixed an issue where an exception was thrown when the probe used for identification was missing a face, iris, or print embedding. Now, instead of throwing an exception, identification returns an empty candidate list for missing modalities. (BSDK-81) **Dependencies** • iface v6.9.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.0 • iengine v3.1.0 (ics 2.3) • ifinger v1.22.1 (embedding 1678264171.1)

Version	Date	
v26.2.0	2025-03-20	**Fixed** • Minor documentation and changelog fixes • An exception is thrown in the `ModalitiesTemplate` constructor, in case of an incorrect input data format (BSDK-70) • An exception is thrown in the `ICSTemplate` constructor, in case of an incorrect input data format (BSDK-70) • An exception is thrown in the `ModalitiesICSTemplate` constructor, in case of an incorrect input data format (BSDK-70) **Added** • Function `ICSTemplate::Anonymize` is added (BSDK-70) • Functions to get templates encoded in `ModalitiesICSTemplate` (BSDK-70) **Dependencies** • iface v6.9.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.0 • iengine v3.1.0 (ics 2.3) • ifinger v1.22.1 (embedding 1678264171.1)

Version	Date	
v26.1.0	2025-03-10	**Added** • New NIST image resampling algorithm provided by NIFR library in compliance with `NIST IR7839` and `NIST SP500-289`. – New resampler interface `Image::ImageResampler` for choosing between resizing algorithms (BSDK-21) * Specialization `Image::BicubicImageResampler` providing the old resizing behaviour * Specialization `Image::AreaImageResampler` providing pixel-area-relation resampling algorithm (BSDK-53) * Specialization `Image::NISTImageResampler` providing the new `NIST` resampling algorithm – New enum `Image::NISTImageResampler::InterpolationMethod` for customizing the `NIST` resampling algorithm with parameters `BICUBIC` and `BILINEAR` (BSDK-21) – New enum `Image::NISTImageResampler::FilterType` for customizing the `NIST` resampling algorithm with parameters `GAUSSIAN` and `IDEAL` (BSDK-21) – New functions `Image::ChangeDpiTo`, `Print::ChangeDpiTo` and `PrintCapture::ChangeDpiTo` with parameter `Image::ImageResampler` for choosing the resizing algorithm (BSDK-21) **Dependencies** • iface v6.9.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.0 • iengine v3.1.0 (ics 2.3) • ifinger v1.22.1 (embedding 1678264171.1)

Version	Date	
v26.0.0	2025-03-06	**Breaking Change** • Split Face Detection and Cropping (BSDK-62) Previously, cropping was always performed when a `Face` object was created. This was unnecessary for use cases where only the given face capture needed to be examined (e.g., when only attributes should be checked). Additionally, it was unclear whether attribute values belonged to the cropped image or the original capture where the face was detected. To address these issues, we have modified the API to separate face detection from cropping. **Changes to Face Detection:** – Renamed `FaceDetector::Segment` and `FaceCapture::SegmentWith` to `FaceDetector::Detect` and `FaceCapture::DetectWith` – `FaceDetector::Detect` and `FaceCapture::DetectWith` now return fully created `Face` objects – Removed crop configuration setters and getters from `FaceDetector::Config`, as they are no longer relevant – Removed the `FaceSegment` class, as its functionality has been integrated into the new `CroppedFace` class **Changes to Face Cropping:** – Added `Face::Crop`, which crops a detected face according to a given `FaceCropConfiguration` and returns a `CroppedFace` object – Replaced `Face::image`, which represented a cropped image, with `CroppedFace::GetFaceCapture` – Replaced `FaceSegment::roundingBox` with `CroppedFace::GetCropRectangle` – Replaced `FaceSegment::isInside` with `CroppedFace::IsCropRegionInsideImage` – Replaced `FaceSegment::captureImage` and `Face::captureImage` with `Face::capture`, which now holds more metadata, including the capture image – Moved `Face::FaceAreaMask` and `Face::FaceAreaSegment` to `CroppedFace` – Removed `Face::faceAttributes`, as they became redundant after introducing `Face::capture` – Removed the unnecessary `CropableFace` interface, as it is no longer needed following the removal of `FaceSegment` • Fixed Cropping Issues (BSDK-11) – Removed alignment getters and setters from `FaceCropConfiguration`, as they became confusing after introducing the `FaceCrop::FULL_NOT_ALIGNED` cropping method – It is now possible to set a transparent background in `FaceCropConfiguration`, but **only** for the `FaceCrop::FULL_NOT_ALIGNED` method **Fixed** • Fixed missing C# documentation for several language elements (BSDK-56) • Fixed missing `ACCURATE_SERVER` detector in the PRO variant • Fixed incorrect cropping behavior for the `TOKEN`, `FULL`, and `TOKEN_NOT_FRONTAL` FaceCrop methods (BSDK-11) **Added**

Version	Date	
v25.0.0	2025-02-12	**Breaking Change** • Removed the `ModalitiesGalleryIndexedReference` class. This class was redundant and caused confusion after introducing the `ModalitiesGalleryReference::GetIndex()` function (ABIS-10819) • `ModalitiesGalleryReference` is now an interface in C# (ABIS-10819) • The `ModalitiesGallery.Consolidate` function is now static in Java, C#, Python, and Ruby (ABIS-10819) **Fixed** • Improved performance when adding a large number of references to the gallery in C#. The slowdown was due to SWIG director overhead (ABIS-10819) • The `ModalitiesGallery.Consolidate` function no longer uses data from `ModalitiesGallery` that might change between identification and consolidation in Java, C#, Ruby, and Python (ABIS-10819) **Added** • Introduced the `IndexedModalitiesGallery` class for faster data addition and lower memory usage. However, it offers less convenience compared to `ModalitiesGallery` (ABIS-10819) **Removed** • Removed the `ModalitiesGalleryIndexedReference` class (ABIS-10819) **Dependencies** • iface v6.9.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.0 • iengine v3.1.0 (ics 2.3) • ifinger v1.22.1 (embedding 1678264171.1)

Version	Date	
v24.2.0	2025-02-05	**Added** • Added Dongle licensing support (ABIS-10618) **Changed** • ISeglib is updated to version 4.3.0 to support dongle license (ABIS-10618) • IFace version is upgraded to 6.9.0 to support dongle license (ABIS-10618) – SDK provides same combined (via SelfieCombinedPassiveLiveness↩ Detector) and universal passive liveness scores as DIS 1.46.1. **Dependencies** • iface v6.9.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.3.0 • iengine v3.1.0 (ics 2.3) • ifinger v1.22.1 (embedding 1678264171.1)
v24.1.0	2025-01-22	**Fixed** • Fixed memory leak in face segmentation (ABIS-7062) • Minor documentation fixes such as trim left source code in examples • Thread sanitizer observations fixed by upgrading BS::thread_pool to version 5.↩ 0.0 **Added** • Documentation of C# and java garbage collector tuning for enrollment-sdk (ABIS-7062) • Java classes now implement AutoClosable **Changed** • IFace version is upgraded to 6.8.0 (ABIS-7062) – SDK provides same combined (via SelfieCombinedPassiveLiveness↩ Detector) and universal passive liveness scores as DIS 1.46.1. **Dependencies** • iface v6.8.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.2.1 • iengine v3.1.0 (ics 2.3) • ifinger v1.22.1 (embedding 1678264171.1)

Version	Date	
v24.0.0	2024-12-12	**Changed** • Compacting iris templates by position to samples in iris gallery to improve `Identify` performance (ABIS-10329) • NFIQ2 version is upgraded to `v2.3.0` (ABIS-10030) **Dependencies** • iface v6.5.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.2.0 • iengine v3.1.0 (ics 2.3) • ifinger v1.22.1 (embedding 1678264171.1)
v23.0.1	2024-11-26	**Changed** • Face and iris identification now respects the thread settings specified in ModalitiesGalleryConfig (ABIS-10033, ABIS-10031, ABIS-10049) **Dependencies** • iface v6.5.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171.1)

Version	Date	
v23.0.0	2024-11-18	**Added** • Face and iris templates protection (ABIS-10051) **Breaking Change** • Removed face accurate server extractor (ABIS-10075) • Removed face accurate-server detector from enrollment-sdk standard variant (ABIS-10077) • The format of FaceTemplate and IrisTemplate has changed. Templates from previous versions of the enrollment SDK are not compatible with this version. Additionally, FaceTemplate and IrisTemplate provided by this SDK will not work with older versions. (ABIS-10051) **Changed** • Speed up adding face/iris templates into ModalitiesGallery (ABIS-10130) **Dependencies** • iface v6.5.0 (balanced 48.1, fast 47.1, accurate 49.1, iris 10.1) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171.1)
v22.0.2	2024-11-08	**Changed** • Fully automated release process of Enrollment SDK (ABIS-9123, ABIS-9125, ABIS-9127) **Dependencies** • iface v6.5.0 (balanced 48, fast 47, accurate 49, accurate-server 44, iris 10) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171.1)

Version	Date	
v22.0.1	2024-10-30	**Fixed** • Added support for processors without AVX2/SSE4.1 instruction set (ABIS-9948) **Dependencies** • iface v6.5.0 (balanced 48, fast 47, accurate 49, accurate-server 44, iris 10) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171.1)
v22.0.0	2024-10-29	**Breaking Change** • Removed function `Face::ToIsoRawImage` that produces non-compliant ISO 19794-5 2011 face ISO image with raw image data (ABIS-9829) **Fixed** • Removed magic numbers in Face, FaceCapture, Iris, IrisCapture, Print that was used to create ISO images (ABIS-9698) • Fixed encode/decode of ISO Images. The encoded ISO images pass `BioCTS for ISO/IEC`. – Print ISO image encode/decode (ABIS-9829) – Iris ISO image encode/decode (ABIS-9829) – Face ISO image decode (ABIS-9829) • Python Logger error when it is not set twice (ABIS-9574) **Added** • Conversion from dot per centimeter to dot per inch ImageDpi::FromDPCM (ABIS-9829) **Dependencies** • iface v6.5.0 (balanced 48, fast 47, accurate 49, accurate-server 44, iris 10) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171.1)

Version	Date	
v21.0.1	2024-10-22	**Added** • C# documentation (ABIS-9777) **Changed** • Swig is updated to version 4.3.0 (ABIS-9777) **Dependencies** • iface v6.5.0 (balanced 48, fast 47, accurate 49, accurate-server 44, iris 10) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171.1)

Version	Date	
v21.0.0	2024-10-21	**Breaking Change** • Removed function Face::Roll, Face::Yaw, Face::Pitch because they are not supported by IFace anymore. (ABIS-8017) • Removed ModalitiesICSMatcher and ModalitiesICSVerifyMatcher because verification of multi modal ModalitiesICSTemplate is not supported due to template format change in IFace. (ABIS-8017) • Removed function ModalitiesICSTemplate::SimilarWith because multi modality verification of ICS templates is not supported due to template format change in IFace. (ABIS-8017) **Fixed** • Concurrent detection and extraction of faces and irises (ABIS-8017) **Changed** • IFace version is upgraded to 6.5.0 (ABIS-8017) – SDK provides same combined (via SelfieCombinedPassiveLiveness↩Detector) and universal passive liveness scores as `DIS 1.42.2`. **Added** • ModalitiesMatcher, ModalitiesVerifyMatcher are added to provide multi modality verification functionality. (ABIS-8017) **Dependencies** • iface v6.5.0 (balanced 48, fast 47, accurate 49, accurate-server 44, iris 10) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171.1)

Version	Date	
v20.0.1	2024-10-16	**Fixed** • Wrong values in Quality blocks and Landmark Point blocks. They are removed from ISO 19794-5 facial images produced by Face::ToIsoRawImage and Face::ToIsoImage (ABIS-9502) **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171.1)
v20.0.0	2024-10-15	**Breaking Change** • New mandatory interface method ModalitiesGalleryReference::GetIndex for providing unique id of a reference (ABIS-9273) **Changed** • It is possible to add only embedding templates to ModalitiesGallery (without ICS) (ABIS-9273) – It is possible to run ModalitiesGallery::IdentifyModalities with embedding only gallery and probe – It is NOT possible to run ModalitiesGallery::Identify neither ModalitiesGallery::Consolidate when there are prints without ICS representation in the gallery or probe **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171.1)

Version	Date	
v19.1.2	2024-10-03	**Fixed** • Wrong calculation of iris and face embeddings distance that can result to lower identification accuracy. **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171.1)
v19.1.1	2024-10-01	**Fixed** • Embeddings comparison count is calculated only for same positions in partition. (ABIS-9461) **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171.1)
v19.1.0	2024-09-27	**Added** • Embedding identification can run in threads. The thread count can be set via ModalitiesGalleryConfig::SetThreadsCount. License feature `multithreaded-galleries` is necessary to use threads in ModalitiesGallery. (ABIS-9271) **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171.1)

Version	Date	
v19.0.0	2024-09-20	**Added** • variant `PRINT` that has only print features (ABIS-9268) • license feature `multithreaded-galleries` (ABIS-9270) • print embedding protection (ABIS-9269) **Breaking Change** • The format of the EmbeddingTemplate has changed. Templates from previous versions of the enrollment SDK are not compatible with this version. Additionally, the EmbeddingTemplate provided by this SDK will not work with older versions. (ABIS-9269) **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171.1)

Version	Date	
v18.0.2	2024-09-17	**Added** • Added license to the documentation (ABIS-8746) • Added list of third-party dependencies to the documentation and to the package (ABIS-8746) **Changed** • ISegLib version is upgraded to 4.1.5 (ABIS-8746) • Improved Image::CropRectangle and Image::CropAlignedRectangle performance (ABIS-6550) • Image::CropAlignedRectangle uses bicubic interpolation strategy instead of the linear interpolation (ABIS-6550) **Fixed** • The lifespan of SDK interface implementation BinaryFileAccess/Logger/Bytes↩WriterStrategy/ImageEncoder instantiated in Python is kept while it is in use by SDK. (ABIS-9074) Till now the user has to maintain lifespan of objects while they are used in SDK. • Fixed cropped Rectangle/AlignedRectangle/Circle when input image has different number of channels as cropped one (ABIS-9220) • Fixed template conversion in samples **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.5 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171)

Version	Date	
v18.0.1	2024-09-04	**Fixed** • Python documentation is fixed • Run of C# sample on Windows x86 (ABIS-9075) **Changed** • IEngine version is upgraded to 2.96.2 • IFinger version is upgraded to 1.16.0 **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.4 • iengine v2.96.2 (ics 2.3) • ifinger v1.16.0 (embedding 1678264171)

Version	Date	
v18.0.0	2024-09-03	**Fixed** • The lifespan of SDK interface implementation BinaryFileAccess/Logger/Bytes↩WriterStrategy/ImageEncoder instantiated in Java/C# is kept while it is in use by SDK. Till now the user has to maintain lifespan of objects while they are used in SDK. **Added** • Python wrapper contains HTML documentation (ABIS-9001) • Detailed exceptions (ABIS-8358) **Breaking Change** • Exception types are changed (ABIS-8358) – Java * FileException and ImageException are removed from functions throw list * IllegalArgumentException is replaced by EnrollmentInvalidArgumentException or its subclasses, that is RuntimeException via EnrollmentException – Cpp * std::runtime_error is changed to EnrollmentRuntimeException that is std::exception via EnrollmentException * std::invalid_argument is changed to EnrollmentInvalidArgumentException or its subclasses that is std::exception via EnrollmentException * std::logic_error is changed to EnrollmentRuntimeException that is std::exception via EnrollmentException – C# * ArgumentOutOfRangeException is replaced by EnrollmentInvalidArgumentException or its subclasses, that is ApplicationException via EnrollmentException – Python * ValueError is replaced by EnrollmentInvalidArgumentException or its subclasses that is SystemError via EnrollmentException – Ruby * ArgumentError is replaced by EnrollmentInvalidArgumentException or its subclasses that is Exception via EnrollmentException **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.4 • iengine v2.95.3 (ics 2.3) • ifinger v1.12.2 (embedding 1678264171)

Version	Date	
v17.6.2	2024-08-20	**Added** • Python wrapper contains HTML documentation (ABIS-9001) **Fixed** • Identification results no longer contain face or iris modality templates when these modalities are absent in the gallery or probe (ABIS-8890) • Print verification score range is in correct range <0,1000>. The wrong score was provided from version v17.5.1 to v17.6.1 (ABIS-8910) **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.4 • iengine v2.95.3 (ics 2.3) • ifinger v1.12.2 (embedding 1678264171)
v17.6.1	2024-08-14	**Added** • FPIR at zero FNIR characteristic in the accuracy test report (ABIS-8370) **Changed** • `WsqImageEncoder` method `SetBitrate` supports bitrate <0.1,5.0> (ABIS-8448) **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.4 • iengine v2.95.3 (ics 2.3) • ifinger v1.12.2 (embedding 1678264171)

Version	Date	
v17.6.0	2024-08-12	**Added** • Added cross-referenced solver list to each respective feature / function in the documentation (ABIS-8366) • Minimal background ratio parameter for background uniformity calculation (ABIS-8361) • New`WsqImageEncoder`'s methods `SetBitrate` and `GetBitrate` for adjusting quality and size of output WSQ images (ABIS-8448) • New root page in HTML documentation hiding excessive content of `html/`↩ folder (ABIS-8366) • New C# constructor `Bytes(byte[])` for faster memory copy (ABIS-8438) **Changed** • `ResizeImage` function now uses bicubic interpolation strategy instead of the exact bilinear interpolation and the pixel area relation strategies. **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.4 • iengine v2.95.3 (ics 2.3) • ifinger v1.12.2 (embedding 1678264171)
v17.5.1	2024-08-01	**Fixed** • angles table for face detectors in documentation **Added** • Templates conversion from `ICS` and to/from `ISO/ANSI/ILO SID/ISO CC/`↩ `ISO v30/ISO NC` templates (ABIS-8354) **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.4 • iengine v2.95.3 (ics 2.3) • ifinger v1.12.2 (embedding 1678264171)

Version	Date	
v17.4.2	2024-07-01	**Added** • Class ModalitiesGalleryIndexedTemplate to simplify use cases that prefer usage of indexes instead of object references. • Identification accuracy test (ABIS-7281) – Added identification accuracy test (ABIS-7356) – Identification accuracy report provided with each version in the release notes (ABIS-7357) • Table of sdk versions and corresponding extracted template versions into documentation (ABIS-7360) **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.4 • iengine v2.89.0 (ics 2.3) • ifinger v1.12.2 (embedding 1678264171)
v17.4.1	2024-06-24	**Added** • Identification accuracy test (ABIS-7281) – Added extractor application to extract templates for accuracy tests (ABIS-7354, ABIS-7353) • IFace version is upgraded to 5.16.0 (ABIS-7596) – SDK provides same combined/universal passive liveness scores as `DIS 1.37.1` via SelfieCombinedPassiveLivenessDetector. **Fixed** • IFace version 5.16.0 recognize default hardware id on Windows (Hashed HDD ID) and allows licensed functionality (ABIS-7596) **Dependencies** • iface v5.16.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.4 • iengine v2.89.0 (ics 2.3) • ifinger v1.12.2 (embedding 1678264171)

Version	Date	
v17.4.0	2024-06-12	**Warning** • When client generates default hardware id on Windows (Hashed HDD ID) then licensed face functionality fails because iface do not recognize new hardware id. **Added** • Added descriptive exception when invalid parameter is provided to Face↩ Detector::Config::SetCropMethod method (ABIS-7753) • New API call `GetVersion()` for classes EmbeddingTemplate, ICSTemplate, ANSITemplate, ISOTemplate, FaceTemplate and IrisTemplate (ABIS-7794) **Fixed** • Fix crash of NFIQ2 quality calculation on Windows for images where NFIQ2 minutia extractor found less then 6 minutiae (ABIS-7697) **Dependencies** • iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.4 • iengine v2.89.0 (ics 2.3) • ifinger v1.12.2 (embedding 1678264171)

Version	Date	
v17.3.1	2024-06-06	**Warning** • When client generates default hardware id on Windows (Hashed HDD ID) then licensed face functionality fails because iface do not recognize new hardware id. **Changed** • Optimized memory utilization during wsq image manipulation (ABIS-6549) • Single multi-architecture package for Android in Standard and PRO variant (ABIS-6997) • ILicense is upgraded to 3.2.24 to use Hashed HDD ID as default license hardware ID `HardwareId()` on windows (ABIS-7609) **Fixed** • Fix missing Writer class documentation • Fix missing text in diagrams in pdf documentation on windows due to missing helvetica font in windows docker • Missing documentation for Minutia::quality (ABIS-7890) • Identification in ModalitiesGallery of ICS templates works also without print templates. It is possible to identify also probes with only face/iris templates. (ADEL-6505) **Dependencies** • iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.3 • iengine v2.89.0 (ics 2.3) • ifinger v1.12.2 (embedding 1678264171)

Version	Date	
v17.3.0	2024-05-15	**Changed** • ILicense is upgraded to 3.2.23 to mitigate probability of calculation of same HWID for different HDD on Windows (ABIS-7216) • ISeglib is upgraded to 4.1.2 to mitigate probability of calculation of same HWID for different HDD on Windows (ABIS-7216) • More descriptive exception is thrown when NFIQ2 is calculated for image with different DPI as 500 (ABIS-7543) **Added** • ChangeDpiTo function into classes Image, PrintCapture, Print (ABIS-7543) • Content check during creation of Android packages (ABIS-6997) **Fixed** • Clang tidy (18.1.4) observations (ABIS-7439) **Dependencies** • iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.2 • iengine v2.89.0 (ics 2.3) • ifinger v1.12.2 (embedding 1678264171)

Version	Date	
v17.2.2	2024-05-03	**Changed** • Undefined behavior is analyzed (ABIS-6484) – Clang upgraded to 18.1.4 to enable UB sanitizer suppression, for external libraries – iseglib is updated to version 4.0.2 with support for Clang 18.1.4 **Fixed** • Thread sanitizer observations – Abseil updated to 20240116.2 for Linux and Android, not possible to compile with MSVC 2019 (ABIS-7305) – Logging in cpp sample – Logging in service tests (ABIS-7304) • Fixed segmenting of images with other resolution than 500 DPI (ABIS-7420) – Updated iseglib to 4.1.1 supporting arbitrary dimensions of output cropped segments **Dependencies** • iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.1.1 • iengine v2.89.0 (ics 2.3) • ifinger v1.12.2 (embedding 1678264171)
v17.2.1	2024-04-20	**Added** • Clang thread sanitizer check (ABIS-7251) **Changed** • Added rolled print image into the extraction sample to better visualize fingerprint critical points (ABIS-7049) **Dependencies** • iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.0.0 • iengine v2.89.0 (ics 2.3) • ifinger v1.12.2 (embedding 1678264171)

Version	Date	

v17.2.0	2024-04-11	**Fixed**
		• Iris iso image quality, Image type, Image format, Representation count (ABIS-7116)
		• Create IrisCapture from iso image with raw data (ABIS-7116)
		
		Added
		• It is possible to get facial keypoints using Face::GetKeypoints functions (ABIS-7093)
		  – Individual keypoints are provided by Face::Keypoints class via functions:
		  * RightEyeOuterCorner, RightEyeCentre, RightEyeInnerCorner, LeftEyeInnerCorner, LeftEyeCentre, LeftEyeOuterCorner, NoseRoot, NoseRightBottom, NoseTip, NoseLeftBottom, NoseBottom, MouthRightCorner, MouthCenter, MouthLeftCorner, MouthUpperEdge, MouthLowerEdge, RightEyebrowOuterEnd, RightEyebrowInnerEnd, LeftEyebrowInnerEnd, LeftEyebrowOuterEnd, RightEdge, ChinTip, LeftEdge
		• Set iris image type in IrisAttributes (ABIS-7116)
		
		Dependencies
		• iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10)
		• iseglib v4.0.0
		• iengine v2.88.4 (ics 2.3)
		• ifinger v1.7.4 (embedding 1678264171)

v17.1.1	2024-04-10	**Fixed** • Raw images are same from local and server implementation (ABIS-7100) **Dependencies** • iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.0.0 • iengine v2.88.4 (ics 2.3) • ifinger v1.7.4 (embedding 1678264171)

Version	Date	
v17.1.0	2024-04-04	**Added** • It is possible to set DPI in Image::Decode function (ABIS-7140) • Pro variant of Enrollment SDK for Android **Changed** • iseglib is updated to version 4.0.0 – segmentation input supports bigger images than 4000x4000, proportionally to higher DPI than 500 (ABIS-7141) – segmentation output is preserving original resolution (ABIS-7069) **Fixed** • The rounding boxes have correct dimensions for images with DPI different as 500 when algorithmic segmentation is used (ABIS-7069) **Dependencies** • iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v4.0.0 • iengine v2.88.4 (ics 2.3) • ifinger v1.7.4 (embedding 1678264171)
v17.0.1	2024-03-26	**Fixed** • Uninitialized memory issues found by clang memory sanitizer – libjpeg-turbo is updated to version 2.1.5 (ABIS-6483) – iseglib is updated to version 3.2.16 to fix NN segmentation (ABIS-6482) – ilicense is updated to version 3.2.22 to fix hwid calculation (ABIS-6482) – iengine is updated to version 2.88.4 to fix placement quality calculation (ABIS-6482) **Dependencies** • iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.16 • iengine v2.88.4 (ics 2.3) • ifinger v1.7.4 (embedding 1678264171)

Version	Date	
v17.0.0	2024-03-15	**Added** • Identification based on ICSTemplate only, no EmbeddingTemplate is needed (ABIS-6910) • New masking interface provided by ImageMask. (ABIS-6619) – It is possible to mask image by circle or polygon. – It is possible to blur image by circle or polygon. – For detail see documentation in Image topic. **Breaking Change** • CompositeImageMask is removed. Instead of CompositeImageMask use your own type derived from ImageMask. For detail see documentation in Image topic. • ImageMaskConfiguration is changed, the blurring type is removed. Blurring type is now separate argument for CircleImageMask and PolygonImageMask. For detail see documentation in Image topic. **Changed** • Swig is updated to version 4.2.1 **Fixed** • Transition blurring of cropped and masked iris image is compliant with ISO 19794-6 2011. (ABIS-6619) **Dependencies** • iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.13 • iengine v2.84.4 (ics 2.3) • ifinger v1.7.4 (embedding 1678264171)

Version	Date	

v16.0.0	2024-03-01	**Warning**
		• The blurring of masked and cropped Iris image is not according to standard ISO 19794-6.
		
		Added
		• It is possible to add ExtractedPrintEmbedding into Applicant (ABIS-5894)
		• Applicant can be converted to ModalitiesTemplate for future operations (ABIS-5895)
		• It is possible to identify candidates in modalities gallery (ABIS-5896)
		• Get Iris Cropped and Masked image (ABIS-6619)
		  – It is possible to get cropped and masked iris image compliant with ISO 19794-6 2011 using function Iris::GetISO19794IrisImage (ABIS-6619)
		• Function Image::MaskImageBy for image masking (ABIS-6619)
		
		Breaking Change
		• To separate enrollment phase from operation phase the names of classes have been changed
		  – ApplicantICSMatcher is renamed to ModalitiesICSMatcher
		  – ApplicantICSTemplate is renamed to ModalitiesICSTemplate
		  – ApplicantICSTemplateFactory is renamed to ModalitiesICSTemplateDefaultFactory
		
		Changed
		• OpenCV exceptions are propagated to client to get better info in case of error (ABIS-6552)
		• License exception are more detailed
		
		Fixed
		• The compiler optimizations for speed are turned on in Release configuration
		
		Dependencies
		• iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10)
		• iseglib v3.2.13
		• iengine v2.84.4 (ics 2.3)
		• ifinger v1.7.4 (embedding 1678264171)

Version	Date	
v15.0.0	2024-02-01	**Added** • Image intensity into Print and PrintCapture classes (ABIS-6137) • Segmentation of fingerprints via PrintCapture class (ABIS-5537) • Image DPI value from instance of ImageDpi (ABIS-5706) • Segment Print images to requested dimensions (ABIS-5534) – It is possible to resize PrintSegment. Function PrintSegment::Resize is added. – It is possible to crop PrintSegment to image with given dimensions via setting of PrintCropConfiguration class. – It is possible to disallow aligned crop via setting of PrintCropConfiguration class. – It is possible to provide dimension of cropped image in Image::CropAlignedRectangle and Image::CropRectangle – Functions Width, Height, Resize are added to Rectangle class – PrintDetectorConfig is added into PrintDetector to search top-left vertex from fingerprint point of view. Searched top-left vertex is later use in case of cropping aligned print segment **Changed** • The member variables in rectangle class are changed to functions (ABIS-5534) • Dimension class is renamed to ImageDimension (ABIS-5534) • Image::CropAlignedRectangle behavior is changed to always align top-left vertex to [0,0] coordinate in new image (ABIS-5534) • Swig is updated to version 4.2.0 (ABIS-6487) **Fixed** • LightGray color has correct RGB value (ABIS-5534) • Print create from PrintSegment has all PrintAttributes from PrintCapture (ABIS-5534) • Images in documentation are same as in samples output (ABIS-6494) **Dependencies** • iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.13 • iengine v2.84.4 (ics 2.3) • ifinger v1.7.4 (embedding 1678264171)

Version	Date	
v14.1.0	2024-01-18	**Added** • IFinger library (ABIS-5860) • Embedding extraction of Print (ABIS-5861) • clang sanitizer checks • windows docker for build **Dependencies** • iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.13 • iengine v2.84.4 (ics 2.3) • ifinger v1.7.4 (embedding 1678264171)
v14.0.0	2023-12-06	**Added** • Universal passive liveness calculation (ABIS-5991) **Changed** • IFace version is upgraded to 5.12.0 (ABIS-5991) – SDK provides same combined/universal passive liveness scores as DIS 1.29.0 (`DIS: 1.29.0-SNAPSHOT SAM: 1.35.5 IFace:↩ 5.10.0`) via SelfieCombinedPassiveLivenessDetector. – age gender model is improved (balanced in used) • SelfiePassiveLivenessDetector is renamed to SelfieCombinedPassive↩ LivenessDetector (ABIS-5991) • In the Face::PassiveLiveness function, the default calculation for passive liveness has been updated from the old Combined method to the new Universal approach. (ABIS-5991) **Dependencies** • iface v5.12.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.13 • iengine v2.83.12 (ics 2.3)

Version	Date	
v13.0.0	2023-11-24	**Added** • Applicant vs Applicant verification (ABIS-5391) – Verify applicants – Store/Load Applicant template – Can be executed on server via MultiModalityExecutor • Add angle to print segment and possibility to calculate global angle (ABIS-5389) • Add hand position estimation after slap segmentation (ABIS-5390) **Changed** • Add iface for android x86 simulator • ISegLib is updated to 3.2.13 with extended rectangle info • Default maximal rotation in ICSVerifyMatcher is changed from 100 to 180. • Extracted print will throw when given print to constructor is null **Fixed** • Fix accepting maximal rotation in ICSVerifyMatcher up to 180. • ICSTemplate::data renamed to ICSTemplate::icsTemplateData (ABIS-5548) • Fixed sporadic crash/exception in java caused by RawImage that can be not referenced in java, therefore GC can deleted it any time. The Data() could then return reference to deleted object. (ABIS-5687) **Dependencies** • iface v5.9.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.13 • iengine v2.83.12 (ics 2.3)

Version	Date	
v12.1.0	2023-10-25	**Changed** • ISegLib is updated to 3.2.12 to upgrade NFIQ2 to version 2.2.0 (ABIS-4137) • IFace version is upgraded to 5.9.0 – SDK provides same passive liveness scores as DIS 1.26.0 (`DIS: 1.26.0 SAM: 1.36.0 IFace: 5.8.0`) via SelfiePassive↩ LivenessDetector. **Added** • Critical points (Core/Delta) from template (ABIS-2204) • It is possible to use hardware id ANDROID_ID on Android platform • Obtain license hardware ID by specific method • License is necessary to use passive liveness and adjudicator features (ABIS-5507) **Fixed** • Fix left hand finger positions in slap segmentation (ABIS-5556) • clang address sanitizer observations – wsq image encode – server shutdown • minutia shape is based on type – ridge bifurcation - square – ridge ending - circle **Dependencies** • iface v5.9.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.12 • iengine v2.79.9 (ics 2.3)

Version	Date	
v12.0.0	2023-10-03	**Added** • Face, Iris, Print can be stored as iso image (ABIS-2001). • FaceCapture, IrisCapture, Print can be loaded from iso image (ABIS-2001). • Conversion from ICS to ANSI template (ABIS-2199). • ILicense SDK is integrated. **Changed** • PrintAttributes are used to initialize Print and PrintCapture classes constructors. • IrisAttributes are used to initialize Iris and IrisCapture classes constructors. • FaceAttributes are used to initialize Face and FaceCapture classes constructors. • RawImageEncoder is renamed to IRawImageEncoder • Function Jp2ImageEncoder::SetCompressionRate is renamed to Jp2ImageEncoder::SetCompressionRatio and argument meaning is now ratio e.g. `4:1` • BinarizedImage, SkeletonImage, FilteredImage are accessible only via getters • PrintCaptureDeviceTechnology, PrintImpression values are renamed to be ISO 19794-4 (2011) compliant (ABIS-2001) **Fixed** • clang-tidy warnings **Dependencies** • iface v5.7.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.9 • iengine v2.79.9 (ics 2.3)

Version	Date	
v11.1.1	2023-08-04	**Added** • Note about not supported transparency decoding for image formats PNG, WEBP, TIFF is added into documentation. **Fixed** • IFace transparent segmentation is shown as transparent image in Android sample **Dependencies** • iface v5.7.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.9 • iengine v2.79.9 (ics 2.3)
v11.1.0	2023-08-03	**Added** • More descriptive face documentation • Function HardwareID to get hardware ID for PC/Mobile/device from application (ABIS-4891) **Fixed** • OpenCV is built without filesystem support on Android • Disable lint of ExpiredTargetSdkVersion on Android • Install native libraries under architecture folder on Android • Removed jp2 encoding from cpp Android sample due to OpenCV is compiled without filesystem support on Android • Message compose do not join messages on subsequent compose calls **Dependencies** • iface v5.7.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.9 • iengine v2.79.9 (ics 2.3)

Version	Date	
v11.0.1	2023-08-01	**Added** • Android samples • BinaryFileAccess interface to allow user to modify SDK access to filesystem **Dependencies** • iface v5.7.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.9 • iengine v2.79.9 (ics 2.3)
v11.0.0	2023-07-31	The major version is increased because Image is now used as shared_ptr in c++ due to seg.fault in bindings. However the API of bindings is not affected. **Fixed** • Wrong JNI method in ImageException • Sporadic seg. fault in bindings cased by bad JNI implementation of Image. The garbage collector deletes Image twice. **Dependencies** • iface v5.7.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.9 • iengine v2.79.9 (ics 2.3)
v10.1.1	2023-07-28	**Changed** • IFace version is upgraded to 5.7.0 **Fixed** • IFace attributes nose shadow and red eye are available on android **Dependencies** • iface v5.7.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.9 • iengine v2.79.9 (ics 2.3)

Version	Date	
v10.1.0	2023-07-26	**Warning** • IFace attributes nose shadow and red eye are not available on android **Removed** • License file `iengine.lic` is not used when it is besides SDK library (ABIS-4870) **Added** • Function Initialize to initialize SDK when license file or models path are not accessible via file system (ABIS-4870) **Dependencies** • iface v5.6.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.9 • iengine v2.79.9 (ics 2.3)

Version	Date	
v10.0.0	2023-07-24	**Warning** • IFace attributes nose shadow and red eye are not available on android **Added** • It is possible to read `iengine.lic` file that is stored side by side with SDK library (ABIS-4870) • ImageShapeAttributes class that holds attributes for drawing shapes • RBG and RBGA class instead of Color class **Changed** • IFace version is upgraded to 5.6.0 – SDK provides same passive liveness scores as DIS 1.22.0 via Selfie↩ PassiveLivenessDetector. • IEngine version is upgraded to 2.79.9 • Image – Shapes have more attributes when drawing. Added transparency and thickness. – It is possible to use transparency while cropping image **Fixed** • IFace seg. fault on Android platform for iris detection • Some OpenCV errors printed on console are redirected to log (https://↩ github.com/opencv/opencv/issues/23994) **Removed** • Not used IFace solvers (passive liveness distance, accurate server wild) **Dependencies** • iface v5.6.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.10) • iseglib v3.2.9 • iengine v2.79.9 (ics 2.3)

Version	Date	
v9.1.0	2023-06-27	**Warning** • IFace seg. fault on Android platform for iris detection • IFace attributes nose shadow and red eye are not available on android **Changed** • IEngine version is upgraded to 2.75.0 • IFace version is upgraded to 5.4.0 **Added** • Skeleton, binarized, filtered intermediate images (ABIS-4586) • Minutiae from template (ABIS-4586) • Matching minutiae (ABIS-4586) • SelfiePassiveLivenessDetector - detector for conform passive liveness calculation with precalculated thresholds. It provides same passive liveness scores as DIS 1.21.0. • DrawMinutiae function into Print class (ABIS-4586) • Resize function into Image class • ResizeToMaxDimension function into Image class • Convert image class to different color models due to Minutiae drawings • ResizeToMaxDimension function into FaceCapture class **Fixed** • Minor documentation fixes • SDK provides same passive liveness scores as DIS 1.21.0 via SelfiePassive↩ LivenessDetector. **Dependencies** • iface v5.4.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.7) • iseglib v3.2.9 • iengine v2.75.0 (ics 2.3)

Version	Date	
v9.0.3	2023-05-15	**Warning** • IFace seg. fault on Android platform for iris detection • IFace attributes nose shadow and red eye are not available on android **Fixed** • Fixed passive liveness to be same as in DIS (ABIS-4257) **Dependencies** • iface v5.2.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.7) • iseglib v3.2.9 • iengine v2.61.30 (ics 2.3)
v9.0.2	2023-05-12	**Warning** • IFace seg. fault on Android platform for iris detection • IFace attributes nose shadow and red eye are not available on android **Fixed** • EXIF orientation is taken into account on jpeg decode (ABIS-4254) **Dependencies** • iface v5.2.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.7) • iseglib v3.2.9 • iengine v2.61.30 (ics 2.3)

Version	Date	
v9.0.1	2023-05-09	**Warning** • IFace seg. fault on Android platform for iris detection • IFace attributes nose shadow and red eye are not available on android **Changed** • IFace version is upgraded to 5.2.0 – with an improved model for Passive Liveness (threshold change is needed) – with onnxruntime 1.13.1 • ISegLib version is upgraded to 3.2.9 – with onnxruntime 1.13.1 • .NET standard is changed from 2.1 to 2.0 to be compatible with more .NET frameworks **Fixed** • Enrollment on Android architecture (ABIS-4156) **Dependencies** • iface v5.2.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.44, iris 3.7) • iseglib v3.2.9 • iengine v2.61.30 (ics 2.3)
v9.0.0	2023-05-04	**Added** • Passive liveness (ABIS-4088) **Dependencies** • iface v5.0.3 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.7) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)

Version	Date	
v8.0.0	2023-04-26	**Changed** • ExtractedPrint/ExtractedFace/ExtractedIris replace ICSTemplate/Face↩ Template/IrisTemplate (ABIS-2880) • Function CreateTemplate is renamed to Extract in modality extractors (ABIS-2880) • ICSTemplate/FaceTemplate/IrisTemplate hold only template related data and operations (ABIS-2880) • Applicant function AddValidPrint is renamed to AddUniquePrint (ABIS-1988) • Applicant function AddFace accepts only ExtractedFace (ABIS-1988) **Added** • It is possible to instantiate ICSTemplate/FaceTemplate/IrisTemplate with previously created templates those do not have associated images (ABIS-2876) • Face attributes reliability checks (ABIS-1988) • Applicant is extended with AddValidFace function (ABIS-1988) **Fixed** • Documentation formatting • RPM for variants. The variant is part of RPM name • All exceptions from CPP are translated to binding language • All exceptions from binding language (in case of interface implementation in binding language) are translated to CPP exceptions **Removed** • KSA variant from build is removed **Dependencies** • iface v5.0.3 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.7) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)

Version	Date	
v7.0.2	2023-01-19	**Fixed** • JP2000 with grayscale content are correctly decoded without exception (ABIS-3041) **Dependencies** • iface v5.0.3 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.7) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)
v7.0.1	2023-01-11	**Changed** • IFace version is 5.0.3 **Fixed** • Fix samples output • Add similarity score to modality documentation **Dependencies** • iface v5.0.3 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.7) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)

Version	Date	
v7.0.0	2023-01-10	**Changed** • Change PrintPosition to be compatible with ISO print position (ABIS-2970) • Face and Iris similarity score is now in range <0,100> (ABIS-2970) • Change Iris detection confidence to float with range <0,1> (ABIS-2970) **Removed** • Removed iris extractor P1 and P2 because they are not intent to be used by clients. They are partials extractor for Accurate extractor. (ABIS-2970) • Removed face Mask attribute because iface does not provides same result for face handler and serialized face handler (ABIS-2970) **Dependencies** • iface v5.0.2 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.7) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)
v6.0.0	2023-01-09	**Changed** • Replaced Get in GetRectangle, GetCircle, GetAlignedRectangle to Crop **Added** • Default prototype for configuration classes (ABIS-2875) **Dependencies** • iface v5.0.2 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.7) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)

Version	Date	
v5.0.0	2023-01-03	**Added** • SaveAs method into Image (ABIS-2872) • To/From ISO positions conversion functions (ABIS-2873) **Changed** • Renamed Thumbs to ThumbsCapture (ABIS-2879) • Renamed Slap to SlapCapture (ABIS-2879) • Change iris detection confidence threshold to 6000 in documentation (ABIS-2870) • Renamed File to BinaryFile (ABIS-2877) **Fixed** • Fixed print position in ICS to ISO template conversion **Dependencies** • iface v5.0.2 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.7) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)
v4.0.0	2022-12-29	**Added** • Iris in service (ABIS-2870) **Changed** • The Logger can be set ONLY globally • IFace version is 5.0.2 (ABIS-2870) **Fixed** • throw on null Logger and executors **Dependencies** • iface v5.0.2 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.7) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)

Version	Date	
v3.4.0	2022-12-19	**Added** • Default Logger (ABIS-1991) **Dependencies** • iface v4.20.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)
v3.3.6	2022-12-09	**Fixed** • RPM installation - added missing .so files for bindings (ABIS-2776) **Dependencies** • iface v4.20.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)
v3.3.5	2022-12-09	**Fixed** • RPM installation (ABIS-2776) **Added** • RPM installation adds installed library into ldconfig **Dependencies** • iface v4.20.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)

Version	Date	
v3.3.4	2022-12-07	**Added** • license and readme into nupkg is added **Dependencies** • iface v4.20.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)
v3.3.3	2022-12-06	**Added** • nupkg is stored at nuget.org for public usage **Dependencies** • iface v4.20.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)
v3.3.2	2022-12-06	**Changed** • CSharp DLL binding name is changed to Innovatrics.Enrollment.dll **Added** • nupkg is stored into nexus3.ba.innovatrics.net **Dependencies** • iface v4.20.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)

Version	Date	
v3.3.1	2022-12-06	**Fixed** • Print matching in service **Dependencies** • iface v4.20.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)
v3.3.0	2022-12-05	**Added** • Service for print (ABIS-2737) • Global executors for print, face and image **Dependencies** • iface v4.20.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)
v3.2.0	2022-12-01	**Added** • Service for face: – segmentation masks – re-cropping of detected face • Face extraction and verification (ABIS-1987) **Dependencies** • iface v4.20.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)

Version	Date	
v3.1.2	2022-11-14	**Fixed** • background color when cropping aligned rectangle **Dependencies** • iface v4.20.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)
v3.1.1	2022-11-14	**Added** • possibility to crop aligned face • documentation for env. variables provided by installation **Changed** • Java bindings are published on public maven • GRPC message size limits are set to maximum **Dependencies** • iface v4.20.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)

Version	Date	
v3.0.1	2022-11-10	**Added** • INNO_ENROLLMENT_HOME,INNO_ENROLLMENT_32_HOME during windows installation **Changed** • 32bit installation is not added to PATH on windows **Fixed** • 32bit runtime linkage on windows **Dependencies** • iface v4.20.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)
v3.0.0	2022-11-09	**Added** • Service for image • Service for face except: – mask attribute – segmentation masks – re-cropping of detected face **Dependencies** • iface v4.20.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.61.30 (ics 2.3)

Version	Date	
v2.0.0	2022-09-20	**Added** • Image – Raw, Jpg, Tiff, Jpg2000, WebP image encoder/decoder • Version info (ABIS-1993) **Fix** • Image can be used after passing into FaceCapture, Slap, Thumbs in bindings (ABIS-1997) • Small fixes in documentation (ABIS-1995, ABIS-1994) **Dependencies** • iface v4.15.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2) • iseglib v3.2.5 • iengine v2.53.6 (ics 2.3)

Version	Date	
v1.0.0	2022-09-05	**Added** • Image processing based on opencv and iwsqlib (for wsq images) (SDK-3642) – Encode/Decode (bmp, png, wsq) – Draw shapes (all with lines, such as rectangles e.g. rounding boxes) – Get rectangle (to get print images from slap also with rotation to horizontal position) • Logging (customizable by client code also for bindings) (SDK-3642) • Serialization (customizable by client code also for bindings) (SDK-3642) • JSON serialization is provided by library. It supports base64 encoding for bytes (SDK-3642) • Slap/Thumbs segmentation via iseglib library (both segmentation algorithms are supported) (SDK-3642, SDK-3675, ABIS-1642) • Linux Windows installers (SDK-3667) • Binding – Java on Windows, Linux, Android – CSharp on Windows, Linux – Ruby on Linux – Python on Linux • Iris – detection (ABIS-1635, ABIS-1644) – extraction (ABIS-1643) – template quality (ABIS-1643) • Face – detection (SDK-3691, ABIS-1639) – attributes (ABIS-1640) Sharpness, Brightness, Contrast, Unique Intensity Levels, Shadow, Nose Shadow, Specularity, Right Red Eye, Left Red Eye, Roll, Eye Distance, Roll Angle, Pitch Angle, Yaw Angle, Face Size, Mask, Face Relative Area, Face Relative Area In Image, Tinted Glasses, Gender, Pitch, Yaw, Eye Gaze, Right Eye Status, Left Eye Status, Glass Status, Heavy Frame, Mouth Status, Background Uniformity, Age, Passive Liveness (Moire, Frames, Nearby, Distant) – re-crop of already detected face segment (ABIS-1645) – detection if face segment is inside image (ABIS-1645) – background elimination (ABIS-1645) • Print – Print extraction, 1:1 verification (SDK-3678) – Print extraction based on Neural Network (SDK-3745) – Print flattening (SDK-3741, SDK-3736, SDK-3746) – Print quality (SDK-3698) – Finger and palm positions (ABIS-1636) – Impression and capture technology (ABIS-1638) – Conversion from ICS to ISO template (ABIS-1641) **Dependencies** • iface v4.15.0 (balanced 1.39, fast 1.36, accurate 1.33, accurate-server 1.37, iris 3.2)

2.1 License Agreement

LICENSE AGREEMENT FOR THE INNOVATRICS ENROLLMENT SDK

ATTENTION: CAREFULLY READ THIS AGREEMENT.
This Innovatrics Enrollment SDK License Agreement ("Agreement") is made by and between Innovatrics, s.r.o., a Slovak company, having its offices at Pri vinohradoch 82, 831 06 Bratislava, Slovakia, hereinafter "Innovatrics" and you, hereinafter as "Company" for the Innovatrics software products identified above (hereinafter "Products").

By installing, copying, downloading, accessing, or otherwise using any of the Products, you agree to be bound by the terms of this Agreement.
IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, DO NOT INSTALL OR USE ANY OF THE PRODUCTS.

1. Definitions
(a) "Software" means the source code, libraries, and documentation included in the Products identified above and provided to assist the Company in building product(s) that work closely with the IEngine print recognition software.
(b) "Application" means software product(s) built by the Company through the use of the Software, which bundle the elements of the Software and connect to a database for print storage ("Print Database").
(c) "Deployment of Application" means any and all copies of the Application installed, used, distributed, leased, or sold by the Company and any of its customers, employees, distributors, resellers, affiliates, or other channel partners.
(d) "Deployment License" means the Server and, if applicable, the Client license required for each Application installation.
(e) "Client" means a workstation using the Software/Products to extract print templates from print images and/or connect to the Matching Station.
(f) "Matching Station" means a computer that uses the Software/Product to search prints in the Print Database of an Application.
(g) "User" means one user record, with one up to ten (1-10) registered prints, included in the Print Database of an Application.
(h) "Customer" means an end-user customer of the Company that licenses Applications for use in its operations and/or a reseller or other distributor of the Company's Application that is granted rights to sublicense the Application directly or indirectly to an end-user customer.

2. Grant Of 14-Day Evaluation License
Subject to the terms and conditions of this Agreement, Innovatrics grants the Company a personal, non-exclusive, non-transferable, limited license for testing purposes only for a period of fourteen (14) days following the initial installation of the Software/Product ("Trial Period"), with the following rights and limitations:
(a) Company may install one (1) copy of the Software on up to one (1) computer to test the Software Product only.
(b) Company may not copy, redistribute, or sell the Application using the Software Product. Before the end of the Trial Period, the Company must purchase the Software Product. Otherwise, the Company must refrain from using or testing the Software Product, remove the Software from its system, and erase, delete, or destroy any notes, documents, magnetic media, or other computer storage that contains the Software.
(c) Support and maintenance services are not provided during the Trial Period.

3. Grant Of Software License And Permitted Uses
In consideration of the license fees paid and subject to the terms and conditions of this Agreement, Innovatrics grants the Company a personal, worldwide, non-exclusive, non-transferable, limited license with the following rights:
(a) Company may install and use one (1) copy of the Software/Product on up to one (1) computer to design, develop, and test Applications that use the Software/Product.
(b) Solely with respect to the portions of the Software identified as sample code ("Sample Code"), Innovatrics also grants the Company the right to modify the source code version of the Sample Code only for the sole purposes of designing, developing, and testing Applications, and to reproduce and distribute the Sample Code along with any modifications thereof, in object code form only.
(c) Company may copy and redistribute the Applications, provided that:
i. The Company has purchased a sufficient number of Deployment Licenses;
ii. The Application adds significant and primary functionality to the Software; and

iii. The Company satisfies Innovatrics' branding requirements ("Branding Requirements") and includes a valid copyright notice on the Application.
(d) Innovatrics also grants the Company the right to use the documentation that accompanies the Software for internal, non-commercial reference purposes only.

4. Limitations Or Restrictions
Company agrees not to:
(a) SUBLICENSE THE SOFTWARE ON A STAND-ALONE BASIS (I.E., OTHER THAN AS PART OF AN APPLICATION);
(b) Design, develop, license, and/or distribute derivative software products;
(c) Translate, reverse engineer, decompile, disassemble, or modify the Software;
(d) Share the license, install the Software/Product, or use it concurrently on different computers;
(e) Distribute or publish all or substantially all of the Software/Product;
(f) Sell, resell, rent, lease, lend, or otherwise transfer the Software/Product, in whole or in part; or
(g) Remove any proprietary notices or labels from the Software/Product.

If the Software/Product includes recognition component(s), Company should understand that recognition is an inherently statistical process, that recognition errors are inherent in the process, and that it is Company's responsibility to handle such errors, monitor the recognition processes, and correct any errors. Neither Innovatrics nor its suppliers shall be liable for any damages arising out of errors in the recognition processes.

5. Branding Requirements
Applications must display a non-obscured bitmap of an icon provided in the Innovatrics logo program current at the time of development, in the command line or on the Application's graphical user interface. Should Innovatrics notify the Company of any amendment(s) to the logo program, the Company agrees to modify the Application immediately so that it is in compliance with the amended Innovatrics logo program. The current logo program is accessible on the website http://www.innovatrics.com/company/logo-program/.

6. Intellectual Property
Innovatrics retains all rights, title, and interest to the Software. All rights not expressly and unambiguously granted herein are reserved by Innovatrics. The Software is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The Software is licensed, not sold. In order to protect Innovatrics' intellectual property rights in the Software, the Company must reproduce on each copy of the Application a copyright notice that clearly states:

"Copyright (c) Innovatrics 2004-2024. All Rights Reserved."

In addition, the Company must distribute the Application only on the basis of a valid agreement that is at least as protective of Innovatrics' rights in the Software as this Agreement.

7. Deployment Licenses
The Company shall purchase a sufficient number of Deployment licenses for each Deployment of the Application. For each Deployment of the Application, the required quantities of Deployment Licenses are calculated as follows:
(a) ONE (1) SERVER DEPLOYMENT LICENSE PER MATCHING STATION. THE NUMBER OF USERS IN THE SERVER DEPLOYMENT LICENSE MUST BE GREATER THAN OR EQUAL TO THE NUMBER OF USERS IN THE FINGERPRINT DATABASE. The Company agrees not to use, and agrees to ensure its Customers do not use, the same Server Deployment License on two Matching Stations.
(b) ONE (1) CLIENT DEPLOYMENT LICENSE PER WORKSTATION ON WHICH THE CLIENT PART OF THE SOFTWARE IS INSTALLED.

Deployment licenses have to be ordered from Innovatrics and paid for before the Company makes any transaction involving the Deployment of the Application. Innovatrics agrees to use its best efforts to make the Deployment Licenses available to the Company on a timely basis.

8. Price And Ordering
The price of Server Deployment licenses and the price of Client Deployment licenses ordered by the Company from Innovatrics shall be in accordance with Innovatrics' Price List in effect at the time of order placement.

The Company must order and pay for Deployment Licenses prior to the Deployment of the Application. Payment for the Deployment Licenses ordered from Innovatrics, as well as support and maintenance fees and shipping costs, if any, shall be due at the time of order placement.

Deployment Licenses are available to the Company upon request at http://customerportal.↩innovatrics.com or at sales@innovatrics.com.

All payments made to Innovatrics shall be in Euros or the then equivalent amount in United States dollars. The Company's obligation to pay Innovatrics under this Agreement shall not be affected by its ability to collect payment from its Customers. In addition, the Company shall pay all charges reasonably incurred in the handling and/↩or shipment of the Software/Product to Customers, including transportation charges, insurance premiums, taxes, duties, and other applicable government assessments.

9. Support, Upgrades, Consulting
Innovatrics provides the Company with support services ("Support Services") related to the Software for an initial period of sixty (60) days ("Warranty Period") from the date of original purchase or download as shown on the Company's receipt or similar proof of payment. The Company may purchase additional Support Services for a period ("Additional Period") beyond the Warranty Period under a separate agreement. Software upgrades and enhancements ("Upgrades") and professional consulting services ("Consulting Services") are not provided as part of this Agreement but can be contracted under a separate agreement. Use of Support Services, as well as use of Upgrades and Consulting Services, is governed by the standard Innovatrics support, upgrades, and consulting policies available upon request to Innovatrics at sales@innovatrics.com.

10. Termination
All rights granted under this Agreement will automatically terminate if the Company breaches any of the terms and conditions of this Agreement. In addition, either Innovatrics or the Company may terminate this Agreement at any time upon notice to the other party, for convenience. Upon any termination of this Agreement, all licenses hereunder shall terminate, other than end-user Deployment Licenses previously entered into in accordance with the provisions and during the term of this Agreement. Upon the termination of this Agreement, the Company will promptly erase all copies of the Software in its possession and shall discontinue all reproduction and distribution of Applications developed that reference elements of the Software. The limitations of intellectual property, confidentiality, warranties, liability, and indemnity set forth in this Agreement shall continue in force even after termination.

11. Confidentiality
The Software and any information contained within, related to, or about the Software/Products, including performance and test results, are confidential and proprietary to, and trade secrets of, Innovatrics and its affiliates and suppliers. The Company agrees not to disclose such information to any third party without Innovatrics' prior written consent.

The Company consents to written and oral disclosure by Innovatrics of its name as an end user of the Software/↩Product and/or services in a factual listing of Innovatrics customers to be published within marketing and promotional materials, in presentations, on trade show signs and materials, on its external website, and to financial industry analysts. Innovatrics will seek the Company's prior written authorization for use of the Company's name for promotional activities beyond the above-mentioned, which may include press releases and brochures, or descriptions of the specific Software and/or services provided by Innovatrics.

12. Disclaimer Of Warranty
Innovatrics declares that the Software, in the form originally purchased or downloaded by the Company, will materially conform with the applicable accompanying documentation, and its media will be free from defects in material and workmanship, for the Warranty Period. **HOWEVER, INNOVATRICS DOES NOT AND CANNOT WARRANT THE ACCURACY, SPEED, PERFORMANCE, OR RESULTS THE COMPANY MAY OBTAIN BY USING THE SOFTWARE/PRODUCTS OR DOCUMENTATION.**

INNOVATRICS DISCLAIMS ALL OTHER WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WITH RESPECT TO THE SOFTWARE/PRODUCTS AND THE ACCOMPANYING WRITTEN MATERIALS.

13. Limitation Of Liability; Exclusion Of Damages
TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL INNOVATRICS AND

ITS DISTRIBUTORS OR RESELLERS BE LIABLE TO THE COMPANY OR ANY THIRD PARTY FOR ANY SPECIAL, INCIDENTAL, INDIRECT, PUNITIVE, OR CONSEQUENTIAL DAMAGES WHATSOEVER, WHETHER IN TORT, PRODUCT LIABILITY, AND/OR NEGLIGENCE (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, GOODWILL, ANTICIPATED SAVINGS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, LOSS OF DATA, PRIVACY OR CONFIDENTIALITY, BREACH OF SECURITY SYSTEMS, OR ANY OTHER PECUNIARY LOSS) ARISING OUT OF OR RELATING IN ANY WAY TO THE USE OF, OR INABILITY TO USE, THE SOFTWARE, EVEN IF INNOVATRICS AND ITS DISTRIBUTORS OR RESELLERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES OR LOSSES. THE SOFTWARE IS SUPPLIED "↩ AS IS," AND THE ENTIRE RISK OF ACCURACY, SPEED, AND SATISFACTORY PERFORMANCE IS WITH THE COMPANY. INNOVATRICS AND ITS DISTRIBUTORS OR RESELLERS DO NOT GUARANTEE THAT THE SOFTWARE WILL MEET THE COMPANY'S REQUIREMENTS OR ALL REQUIREMENTS OF THE SOFTWARE OR HARDWARE WITH WHICH IT INTERACTS. **IN ANY CASE, INNOVATRICS' ENTIRE LIABILITY UNDER ANY PROVISION OF THIS AGREEMENT SHALL BE LIMITED TO THE AMOUNT ACTUALLY PAID BY THE COMPANY FOR THIS SOFTWARE.**

14. Indemnity
The Company agrees to indemnify, hold harmless, and defend Innovatrics from and against any claims or lawsuits, including attorneys' fees, that arise from or result from the use of the Software or the distribution of the Application.

15. Company Warranty Of Authority
The Company warrants that the person who has agreed to the terms of this Agreement has full authority to bind the Company.

16. Governing Law
All matters arising from or in connection with this Agreement, its validity, existence, or termination shall be governed by the federal laws of Switzerland published in the official legal bulletin "Bundesblatt/Feuille fédérale/Foglio federale." All disputes arising out of or in connection with the present contract and not amicably settled within a period of three (3) months starting from the date the dispute arose shall be finally settled under the Rules of Arbitration of the International Chamber of Commerce by one or more arbitrators appointed in accordance with the said Rules, and shall be held in the English language and take place in Vienna (Austria).

17. Entire Agreement
This Agreement is the entire agreement between Innovatrics and the Company and supersedes any other agreement, oral or written, and may not be changed except by a written, signed agreement.

18. General
Neither party shall be responsible for delays or failure of performance resulting from acts beyond the reasonable control of such party. Should any provision or part of any provision of this Agreement be found void or unenforceable by a court of competent jurisdiction, such provision or part thereof shall be deemed severed, and the remainder of this Agreement shall remain in full force and effect. The English version of this Agreement shall be the version used when interpreting or construing this Agreement.

19. Trademarks
Innovatrics™, IEngine™, IDKIT™, ANSI-Kit™, ExpressID™, ANSI&ISO™

2.2 Third-Party Software

The table below lists third party software that is incorporated into the Enrollment SDK, along with the license associated with the software.

Included Software	Version	License
NFIQ2	2.3.0-5	NIST
abseil	20240722.0	Apache-2.0

Included Software	Version	License
benchmark	1.8.3	Apache-2.0
boost	1.83.0	BSL-1.0
bshoshany-thread-pool	5.0.0	MIT
c-ares	1.19.1	MIT
cpuinfo	cci.20231129	BSD-2-Clause
date	3.0.1	MIT
eigen	3.4.0	MPL-2.0
flatbuffers	23.5.26	Apache-2.0
gflags	2.2.2	BSD-3-Clause
glog	0.7.0	BSD-3-Clause
grpc	1.48.4	Apache-2.0
gsl-lite	0.38.1	MIT
incbin	cci.20211107	Unlicense
libcurl	8.10.1	curl
libdeflate	1.19	MIT
libjpeg-turbo	3.0.2	IJG, BSD-3-Clause, Zlib
libpng	1.6.53	libpng-2.0
libtiff	4.6.0	libtiff
libunwind	1.8.0	MIT
libwebp	1.3.2	BSD-3-Clause
matrix-lock	2.8.0-2	Matrix
mbedtls	2.27.0-4	Apache 2.0
mio	cci.20230303	MIT
mlpack	4.1.0	BSD License
ms-gsl	4.0.0	MIT
nfir	0.2.0.cci.20241122	NIST-PD
nlohmann_json	3.11.3	MIT
nsync	1.26.0	Apache-2.0
ofiq	1.1.2-2	MIT
onnx	1.16.2	Apache-2.0
onnxruntime	1.18.1-7	MIT
opencv	4.10.0	Apache-2.0
opencv	4.5.5-7	Apache-2.0
openjpeg	2.5.2	BSD-2-Clause
openssl	3.1.4	Apache-2.0
protobuf	3.21.12	BSD-3-Clause
rapidjson	cci.20200410	MIT
re2	20231101	BSD-3-Clause
safeint	3.0.28	MIT
tl-expected	20190710	CC0-1.0
xz_utils	5.4.5	Unlicense, LGPL-2.1-or-later, GPL-2.0-or-later, GPL-3.0-or-later

Included Software	Version	License
zlib	1.2.13	Zlib
zstd	1.5.5	BSD-3-Clause

Note

The `xz_utils` entry lists multiple licenses (Unlicense, LGPL-2.1+, GPL-2.0+, GPL-3.0+). The LGPL-2.1+, GPL-2.0+ and GPL-3.0+ portions apply exclusively to the `xz` command-line tools and test scripts, which are not included in or linked by the SDK.

3 Topic Documentation

3.1 Extending SDK

Classes

- class ApplicantPrintValidation

  Strategy used in Applicant to accept or reject ExtractedPrint.
- class ApplicantFaceValidation

  Strategy used in Applicant to accept or reject ExtractedFace.
- class BinaryFileAccess

  BinaryFileAccess provides abstraction on underlying file system.
- class ImageEncoder

  Interface for image formatting converting to ImageType.
- class ImageMask

  Defines the interface for applying mask to the image with a masker.
- class ImageShape

  Any type of shape that wants to be drawn.
- class Logger

  Interface for library logging.
- class ModalitiesGalleryReference

  The reference interface for ModalitiesGallery.
- class BytesWriterStrategy

  It is used by Writer class to modify bytes serialization algorithm.
- class Writer

  Provides interface for serialization of all classes.

3.1.1 Detailed Description

3.1.2 Overview

The user can make its own implementation of interfaces provided by SDK to extend SDK functionality. The interfaces are available also for binding languages.

- Logger - create your own implementation to bind SDK log messages into your preferred logger. Set it in GlobalLogger::Set.

See also

Logger

- ApplicantPrintValidation - once you pass your own implementation into function Applicant::AddValidPrint, you get control on which prints are added into Applicant.

  See also

  Custom Print validation

- ApplicantFaceValidation - once you pass your own implementation into function Applicant::AddValidFace, you get control on which faces are added into Applicant.

  See also

  Custom Face validation

- BinaryFileAccess - once you want to use file operation provided by Enrollment SDK via functions BinaryFile::ReadAll, BinaryFile::WriteAll yo can still extend their behavior via your own implementation of BinaryFileAccess interface. Set it in GlobalFileAccess::Set.

  See also

  File

- ImageEncoder - you can extend Image with your own encoder, when Enrollment SDK does not support your image type (see supported types) in function Image::DecodeRawImage.
- ImageMask - once you pass your own implementation into function Image::MaskImageBy, you get control how image is masked.

  See also

  Custom mask

- ImageShape - once you pass your own implementation into function Image::DrawShape, you get control what shape is drown in image.

  See also

  Custom shape

- ModalitiesGalleryReference - you can add your own object into ModalitiesGallery with function ModalitiesGallery::Add.

  See also

  ModalitiesGallery

- BytesWriterStrategy - once you pass your own implementation into function Writer::SetBytesStrategy, you get control on how writer represents bytes.

  See also

  Writer

- Writer - you can use your own Writer implementation to serialize Enrollment SDK classes.

3.2 Binding

Topics

- Java
- C#

3.2.1 Detailed Description

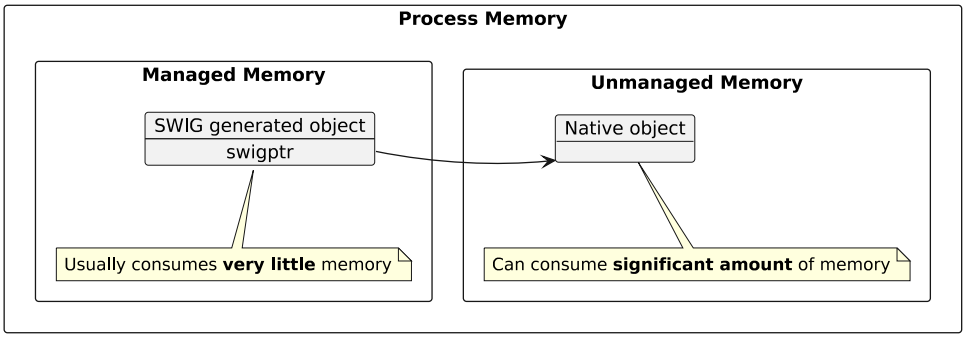
Enrollment SDK is developed in C++. To be possible to use it in customer preferred language we use `SWIG tool` code generator to convert C++ API classes. We provides wrappers for

- Java
- C#

3.2.2 Java

User is able to use Java classes generated by `SWIG tool`, using Java version 8.

3.2.2.1 Memory Management



The Java Garbage Collector (GC) manages memory in the JVM heap, ensuring efficient allocation and deallocation of resources within that space.

In the case of the enrollment-sdk, objects in the Java heap are generally small. However, these objects often create significantly larger amounts of unmanaged (native) memory, which contributes to the overall process memory usage.

Since the Java heap objects are small, the GC may not be triggered frequently, potentially leading to high process memory consumption due to unmanaged memory growth.

The following sections outline how lifetime works and which levers you can combine for your use case.

3.2.2.1.1 One generated type: GC by default, eager release when needed

All Java proxy classes generated for the enrollment-sdk are a **single** surface type per C++ type. They implement `java.lang.AutoCloseable` so you **may** use `try-with-resources` or call `close()` (which forwards to SWIG `delete()`) when you need **deterministic** release of native memory (for example under memory pressure or in tight loops creating many proxies).

In typical applications it is **enough to rely on the garbage collector**: when a proxy becomes unreachable, SWIG's finalization path can reclaim the native side as well. You are **not** required to wrap every object in `try-with-resources`; use eager release only where profiling shows native memory growth or latency requires it.

`AutoCloseable` exists so both styles are supported **without** a second wrapper type.

3.2.2.1.2 Use small heap

If your Java application uses only small objects, consider setting the maximum heap size as low as possible using the `-Xmx` option during application startup.

This will cause the GC to run more frequently, ensuring that unmanaged memory is released more effectively.

3.2.2.1.3 Use try-with-resources when you need eager cleanup

When you choose deterministic cleanup, use the `try-with-resources statement` so `close()` is invoked automatically. This is especially useful where many short-lived proxies are created and you want native memory freed promptly.

Code

```
// Please note that the sample source code
// provided here is for informational purposes
// only and should not be used for production
// purposes without proper testing and
      modification.
private static Face getFace(byte[] img) {
    Bytes bytes = new Bytes(img);
    Image image = Image.Decode(bytes);
    FaceCapture capture = new FaceCapture(image);

    FaceDetector.Config cfg = new
      FaceDetector.Config();
    FaceDetector detector = new FaceDetector(cfg);
    Faces faces = capture.DetectWith(detector);

    if (faces.isEmpty()) {
        return null;
    }

    return faces.get(0);
}
```

Code with `try-with-resources`

```
// Please note that the sample source code
// provided here is for informational purposes
// only and should not be used for production
// purposes without proper testing and
      modification.
private static Face getFace(byte[] img) {
    try (Bytes bytes = new Bytes(img); Image image
      = Image.Decode(bytes);
            FaceCapture capture = new
      FaceCapture(image);
            FaceDetector.Config cfg = new
      FaceDetector.Config();
            FaceDetector detector = new
      FaceDetector(cfg);
            Faces faces =
      capture.DetectWith(detector)) {
        if (faces.isEmpty()) {
            return null;
        }

        return faces.get(0);
    }
}
```

3.2.2.1.4 Use `close()` with caution

You can call `close()` to free the native memory immediately (same effect as successful completion of a `try-with-resources` block). **Do not use the Java object after it has been closed.** If any method is called on a closed Java object, a `RuntimeException` or `NullPointerException` is thrown.

3.2.2.1.5 IDE "resource leak" warnings and GC-only code

Eclipse JDT and editors based on it (including VS Code / Cursor with the Java language server) often warn when a local variable holds an `AutoCloseable` value and neither `try-with-resources` nor `close()` appears on all paths. That does **not** mean GC-only usage is wrong for this SDK: it is a static analysis limitation.

If you intentionally rely on GC for a scope, you can, for example:

- Narrow a `try-with-resources` block around the few objects that need eager release, or
- Add `@SuppressWarnings("resource")` on the enclosing **method** (or, where your Java level allows, on the variable) for that intentional GC lifetime, or
- Adjust your project's Java compiler "resource / closeable" diagnostics if your team policy allows.

3.2.2.1.6 Force GC

While Java GC is automatic, you can request garbage collection using `System.gc()`. This should be used cautiously, as it is a suggestion, not a guarantee.

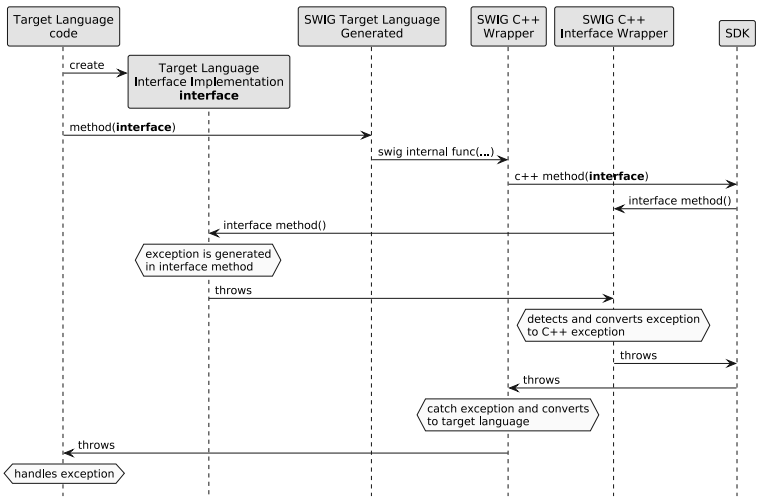
For applications with high unmanaged memory usage, consider monitoring total memory and triggering the garbage collector if needed:

```
// Please note that the sample source code provided here is for informational purposes
// only and should not be used for production purposes without proper testing and
// modification.
private static void triggerGarbageCollector() {
    // Set process memory limit to 2GB
    long maxMemoryLimit = 2L * 1024 * 1024 * 1024;

    Runtime runtime = Runtime.getRuntime();

    while (!Thread.currentThread().isInterrupted()) {
        long totalMemory = runtime.totalMemory() - runtime.freeMemory();

        if (totalMemory > maxMemoryLimit) {
            // Request garbage collection
            System.gc();
        }

        // Check every second
        try {
            Thread.sleep(1000);
        } catch (InterruptedException e) {
            Thread.currentThread().interrupt();
            break;
        }
    }
}
```

Use it in the main function of your application:

```
Thread memoryLimiter = new Thread(() -> triggerGarbageCollector());
memoryLimiter.start();
```

3.2.2.2 Exception Propagation



Exception from Java implementation of Interface	Exception in Enrollment SDK native code	Exception propagated back to Java
EnrollmentException and its sub-classes	EnrollmentException and its sub-classes	EnrollmentException and its sub-classes

Exception from Java implementation of Interface	Exception in Enrollment SDK native code	Exception propagated back to Java
IllegalArgumentException	std::invalid_argument	IllegalArgumentException
IndexOutOfBoundsException	std::out_of_range	IndexOutOfBoundsException
RuntimeException	std::runtime_exception	RuntimeException
Exception	std::runtime_exception	RuntimeException

3.2.2.3 Initialization

Because Java requires native libraries to be loaded before any SDK function can be called, `LoadNative↩Library()` must be the first SDK function invoked.

After the native library has been loaded, you can either rely on the SDK's auto-init functionality or initialize the SDK manually.

For convenience, the SDK also provides an overload:

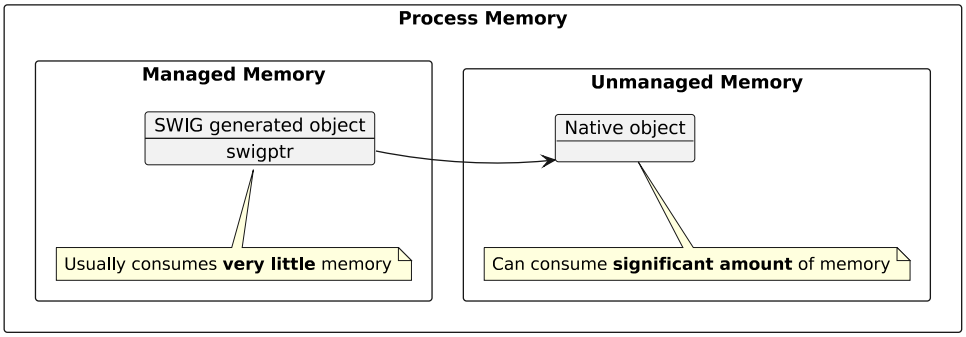
```
LoadNativeLibrary(() -> InitializerConfig)
```

This function first calls `LoadNativeLibrary()`, then invokes the supplied lambda, and finally passes the lambda's return value to `Initialize(InitializerConfig)`.

3.2.3 C#

User is able to use C# classes generated by `SWIG tool`.

3.2.3.1 Memory management



The C# Garbage Collector (GC) manages only managed memory, ensuring efficient allocation and deallocation of resources within that space.

In the case of the enrollment-sdk, objects in managed memory are typically small. However, these objects often create significantly larger amounts of unmanaged memory, contributing to the overall process memory usage.

Since managed objects are small, the GC is triggered less frequently, which can lead to high process memory consumption due to unmanaged memory growth.

The following sections outline solutions to address this challenge. You can choose one or more approaches that best suit your use case.

3.2.3.1.1 Use small heap

If your C# application uses only small objects, consider setting the maximum GC heap memory as low as possible using the `HeapHardLimit` configuration.

This will cause the GC to run more frequently, ensuring that unmanaged memory is released more effectively.

3.2.3.1.2 Use using statement

Use the `using statement` for each enrollment-sdk variable individually. Avoid chaining multiple enrollment-sdk variables in a single using statement.

Code

```
// Please note that the sample source code
// provided here is for informational purposes
// only and should not be used for production
// purposes without proper testing and
      modification.
private static Face GetFace(byte[] img)
{
    var capture = new FaceCapture(Image.Decode(new
      Bytes(img)));

    var cfg = new FaceDetector.Config();
    var detector = new FaceDetector(cfg);
    var faces = capture.DetectWith(detector);

    if (faces.Count == 0)
        return default;

    var face = faces[0];
    return face;
}
```

Code with `using`

```
// Please note that the sample source code
// provided here is for informational purposes
// only and should not be used for production
// purposes without proper testing and
      modification.
private static Face GetFace(byte[] img)
{
    using var bytes = new Bytes(img);
    using var image = Image.Decode(bytes);
    using var capture = new FaceCapture(image);

    using var cfg = new FaceDetector.Config();
    using var detector = new FaceDetector(cfg);
    using var faces = capture.DetectWith(detector);

    if (faces.Count == 0)
        return default;

    // IMPORTANT
    // Because the Face object is returned,
    // we can not use `using` because
    // `using` will delete native memory
    // on function return
    var face = faces[0];
    return face;
}
```

3.2.3.1.3 Force GC

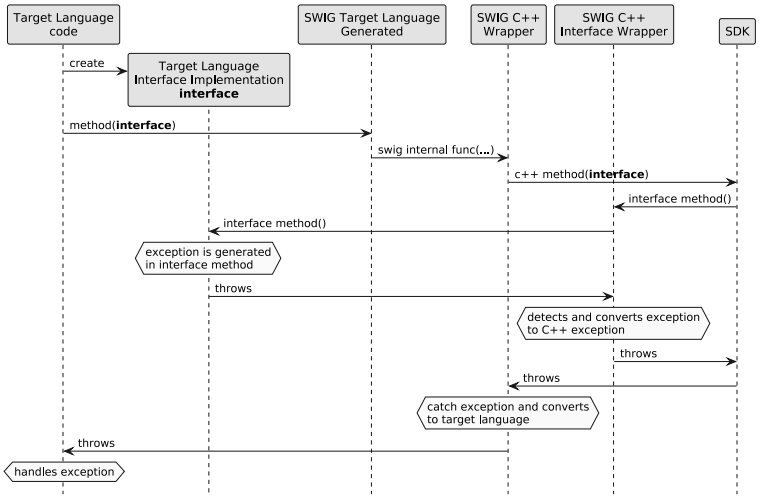
```
// Please note that the sample source code provided here is for informational purposes
// only and should not be used for production purposes without proper testing and
// modification.
static void TriggerGarbageCollector()
{
    // Set process memory limit to 2GB
    long maxMemoryLimit = 2L * 1024 * 1024 * 1024;

    while (true)
    {
        try
        {

            long totalMemory = Process.GetCurrentProcess().WorkingSet64;

            if (totalMemory > maxMemoryLimit)
            {
                // Trigger garbage collection
                GC.Collect();
            }

            // Check every second
            Thread.Sleep(1000);
        }
        catch (ThreadInterruptedException)
        {
            break;
        }
```

```
    }
}
```

Use it in main function of your application

```
var memoryLimiter = new Thread(new ThreadStart(TriggerGarbageCollector));
memoryLimiter.Start();
```

3.2.3.2 Exception Propagation



Exception from C# implementation of Interface	Exception in Enrollment SDK native code	Exception propagated back to C#
EnrollmentException and its sub-classes	EnrollmentException and its sub-classes	EnrollmentException and its sub-classes
ApplicationException	std::runtime_error	ApplicationException
ArithmeticException	std::runtime_error	ApplicationException
DivideByZeroException	std::domain_error	ArgumentException
IndexOutOfRangeException	std::out_of_range	IndexOutOfRangeException
InvalidCastException	std::runtime_error	ApplicationException
InvalidOperationException	std::logic_error	ApplicationException
IOException	std::ios_base::failure	ApplicationException
NullReferenceException	std::runtime_error	ApplicationException
OutOfMemoryException	std::runtime_error	ApplicationException
OverflowException	std::overflow_error	OverflowException
SystemException	std::runtime_error	ApplicationException
ArgumentException	std::invalid_argument	ArgumentException
ArgumentNullException	std::invalid_argument	ArgumentException
ArgumentOutOfRangeException	std::invalid_argument	ArgumentException
Exception	std::runtime_error	ApplicationException

3.3 Exception

Enrollment SDK uses exceptions to propagate errors.

Classes

- class EnrollmentException

  Base exception for all enrollment-sdk exceptions.
- class EnrollmentRuntimeException

  Base exception for all enrollment-sdk runtime exceptions.
- class EnrollmentInvalidArgumentException

  Base exception for all enrollment-sdk invalid argument exceptions.
- class InvalidTemplateException

  The template is empty or or do not contain enough distinctive data or is corrupted.
- class UnsupportedTemplateException

  Template version is not supported in this variant of Enrollment SDK.
- class IncompatibleTemplateException

  Template is incompatible in requested operation.
- class InvalidImageArgumentException

  Image has wrong format or dimensions or dpi and is not suitable for processing.
- class NullArgumentException

  Null argument is not allowed.
- class LicenseException

  Base exception for all enrollment-sdk license errors.
- class LicenseNotFoundException

  License is not found.
- class InvalidLicenseException

  License is found but is invalid.
- class ExpiredLicenseException

  License is expired.
- class LicenseFeatureNotAllowedException

  The licensed feature is not allowed.
- class InvalidHWIDException

  HWID in license is wrong or can not be read.
- class LicenseLimitReachedException

  The limit for operation/count is reached in license.
- class NotSupportedException

  When user uses not supported functionality.
- class FileException

  Exception for file processing.
- class ImageException

  Exception for image processing.

3.3.1 Detailed Description

3.3.2 Overview

Enrollment SDK uses exceptions to propagate errors.

3.4 Face

The library can be used to detect faces from face captures.

Topics

- Solvers

  The Enrollment SDK package includes large binary files called solvers.

Classes

- class FaceExtractorConfig

  It contains extraction configuration.
- class FaceTemplate

  FaceTemplate is Innovatrics proprietary face template.
- class ExtractedFace

  ExtractedFace is ICF Template created by FaceExtractor with Face that was extracted.
- class CroppedFace

  Represents a face cropped from the original face capture.
- class Face

  It holds all functions and data related to Face manipulation.
- class Face::Keypoint

  It holds all attributes related to a facial key point.
- class Face::Keypoints

  It holds all facial keypoints detected on a FaceCapture image.
- class FaceAttributes

  Face attributes Holds the face attributes.
- class FaceCaptureDeviceTypeID

  Capture device type identifier The capture device type identifier shall identify the biometric organization that owns the product that created the BDIR.
- class FaceCaptureDeviceVendorID

  Capture device vendor identifier The capture device vendor identifier shall identify the biometric organization that owns the product that created the BDIR.
- class FaceCapture

  Face capture can contains one or group of people, faces or full body.
- class FaceCropConfiguration

  Face crop configuration.
- class FaceDetector

  FaceDetector implements detection by means of Innovatrics IFace.
- class GlobalFaceExecutor

  Holds instance of FaceExecutor.
- struct FailedFaceAttribute

  It contains attribute data that express which and why attribute fails in reliability check.
- class FaceAttributeReliability

  Interface for checking reliability of attribute.
- class ICAOReliability

  Check if Face will fullfil ICAO requirements.
- class TemplateReliability

  Check if template extraction will be reliable.
- class AgeReliability

  Check if calculation of Age attribute will be reliable.
- class GlassStatusReliability

  Check if calculation of Glass Status attribute will be reliable.
- class TintedGlassesReliability

  Check if calculation of Tinted Glasses attribute will be reliable.

- class SegmentationMaskReliability

  Check if calculation of Segmentation Mask in functions Face::FaceAreaSegment, Face::FaceAreaSegment(RGB), Face::FaceAreaMask will be reliable.
- class FaceSizeReliability

  Check if calculation of Face Size attribute will be reliable.
- class MouthOpennessReliability

  Check if calculation of Mouth Openness attribute will be reliable.
- class UniversalPassiveLivenessReliability

  Check if calculation of Universal passive liveness will be reliable.
- class ISOFullFrontalImageValidator

  Check if usage of Full Frontal Images on travel documents will follow best practices, as described in Section B.3.2 Best practices for use of Full Frontal Images on travel documents [6] .
- class OFIQFace

  It provides quality measures as defined in ISO/IEC 29794-5:2025.

Typedefs

- using FaceSimilarityScore = unsigned int

  It is similarity score between face Templates.

Enumerations

- enum class FaceCaptureDeviceTechnology : uint8_t { UNKNOWN = 0 , STATIC_PHOTOGRAPH_FROM_UNKNOWN_SOURCE = 1 , STATIC_PHOTOGRAPH_FROM_DIGITAL_CAMERA = 2 , STATIC_PHOTOGRAPH_FROM_SCANNER = 3 , VIDEO_FRAME_FROM_UNKNOWN_SOURCE = 4 , VIDEO_FRAME_FROM_ANALOGUE_CAMERA = 5 , VIDEO_FRAME_FROM_DIGITAL_CAMERA = 6 }

  It indicates the class of device technology used to acquire the captured biometric sample [6] .
- enum class FaceImageType : uint8_t { BASIC = 0 , FULL_FRONTAL = 1 , TOKEN_FRONTAL = 2 , POST_PROCESSED_FRONTAL = 3 }

  Several face image types are introduced to define categories that satisfy requirements of some applications [6] .
- enum class FaceCrop : uint8_t { TOKEN_NOT_FRONTAL = 0 , FULL = 1 , TOKEN = 2 , FULL_NOT_ALIGNED = 3 }

  Defines different Crop methods.

Functions

- static constexpr int FaceToIsoCaptureDeviceTechnology (FaceCaptureDeviceTechnology printCaptureDevice↩ Technology)

  Function converts FaceCaptureDeviceTechnology to ISO capture technology.
- static constexpr FaceCaptureDeviceTechnology IsoToFaceCaptureDeviceTechnology (int isoCaptureDevice↩ Technology)

  Function converts ISO capture technology to FaceCaptureDeviceTechnology.
- static constexpr int FaceToIsoImageType (FaceImageType faceImageType)

  Function converts FaceImageType to ISO face image type as defined in iso_19794-5_2011.
- static constexpr FaceImageType IsoToFaceImageType (int faceImageType)

  Function converts ISO face image type to FaceImageType.
- static constexpr bool IsFaceCropMethodValid (FaceCrop faceCrop)

  Function validates the value of FaceCrop enum.

3.4.1 Detailed Description

3.4.2 Overview

The library can be used to detect faces from face captures. Face capture can contains one or group of people, faces or full body.

The face functionality is provided via neural networks. The solvers can be large. If some functionality is not used in your application, you can omit the corresponding solver files from your installation. Chapter Solvers contains information about which solver is required for each functionality.

3.4.2.1 Face Detection

SDK can detect faces of various sizes, orientations, facial hair types or ethnicities. Face can be partly occluded by glasses, sunglasses or hat. SDK can operate with many different lighting conditions and image qualities. Face size is defined as a maximum of values of inter eyes centers distance and distance between center of mouth and center point between eyes: `face_size = max(distance(left_eye_center, right_eye_center), distance(mouth_center,eyes_center))`

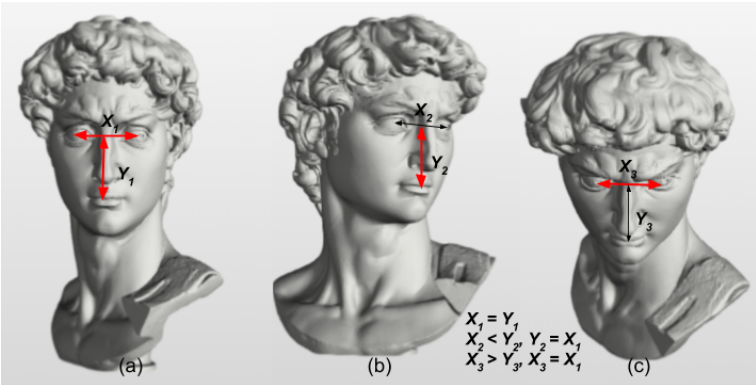


Figure 2 Face Size

Image above shows face size visualizations of various distances of inter eyes centers distances (`X=distance(left_eye_center,right_eye_center)`) and distances between center of mouth and center point between eyes (`Y=distance(mouth_center, eyes_center)`). The face size is the maximum of these two distances. It can be seen from images (a), (b) and (c) that the face size (red arrow) defined in this way is invariant to pose of the head (yaw, pitch, roll). The face size can be specified in absolute (pixel distance) or relative value (pixel distance) relative to image size (`max(image_height, image_width)`) and is important parameter for face detection. SDK can detect faces of various size but there is a lower limit of face size. Upper limit on the face size is not defined. The recommended face size is more than 50 pixels. SDK face processing or recognition functions have the most accurate results when this recommendation is fulfilled. Face area is closely related to the face size. It is an area around a face defined by a bounding rectangle with `width = 4 x face_size`, `height = width / 0.75` (according to ISO/IEC 19794-5 standard, section 9.2). The Point which is the geometric center between eyes is positioned in the face area in position `X_Pos`, `Y_Pos`, where `Y_Pos = 0.6f x width`, and `X_Pos` relies on the head yaw rotation. The main reason for this is to have the whole head in the face area no matter what the head rotation is.

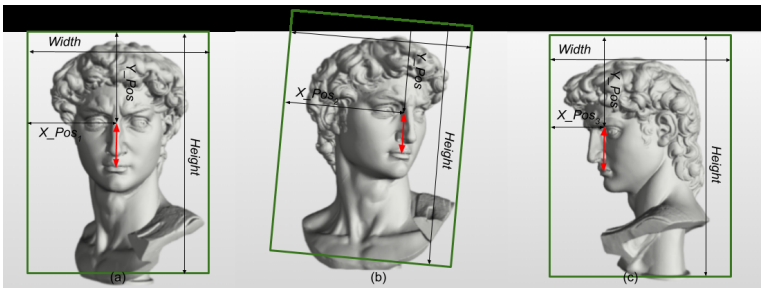


Figure 3 Face Area

Image above shows face area visualizations of face areas (green boxes) for face in various positions. The face size is defined by distance shown as the red arrow. There are shown different positions of face area according to the face yaw rotation in (a),(b) and (c). Positions of the eyes centers in the face areas are the same in the Y direction (`Y_Pos`) but changes in the X direction (`X_Pos1`, `X_Pos2`, `X_Pos3`). Width and Height is the same in all three cases. The face area size relative to image area size can specify sizes of faces that should be detected.

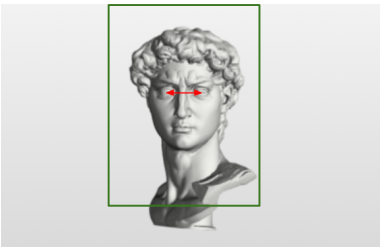


Figure 4 Face

If image above has resolution of 730x470 pixels (`image_area_size = 730 x 470 = 343100`) and face_size is 70 pixels (`face_area_width = 70 x 4 = 280`, `face_area_height = face_area_width / 0.75 = 373`, `face_area_size = 104440`), then relative image size is `relative_image_size = 104440 / 343100 = 0.304` (30.4%). Face area is marked with green box and face size with red arrow. Sometimes, when faces are close to image boundaries, their face areas can get out of the image boundaries. Some application may want to find these faces (and filter them out). Due to this reason each face has attribute related to face area visible in image (FaceRelativeAreaInImage).

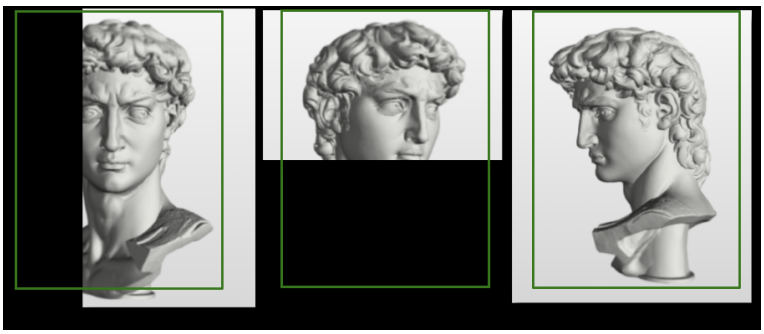


Figure 5 Face area visible in image.

In the image above relative values are (a) 0.30, (b) 0.5, (c) 1

SDK can detect faces of various rotations. However various face detection modes imply different ranges of detectable faces

	fast	accurate / balanced / accurate server
Roll	<-15, 15 >	<-45, 45>
Yaw	<-30, 30 >	<-90, 90>
Pitch	<-30, 30 >	<-45, 45>

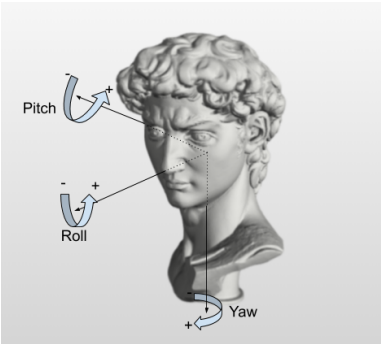


Figure 6 Possible face rotations (defined as in DIN9300)

Prior to further processing each face in FaceCapture MUST be detected via FaceDetector. SDK provides following detection modes:

- accurate Neural network face detector and facial features detector with very high accuracy.

  It is slower than `balanced` mode.

  Required Solvers

  Total size of solvers is 27 MB.
  References face-detector-accurate.solver, face-detector-landmarks-mob-0.50.solver, face-detector-eye-validation.solver

- balanced Neural network face detector and facial features detector with good accuracy.

  It is slower than `fast` mode.

  Required Solvers

  Total size of solvers is 14 MB.
  References face-detector-accurate.solver, face-detector-landmarks-mob-0.25.solver, face-detector-eye-validation.solver

- fast The fastest face detector in combination with the fastest face validators and facial features detectors are used when *fast* mode is used.

  Some faces that are partially occluded faces or faces with sunglasses may be missed. However the speed performance of the face detection is much better as when other modes are used.

  Required Solvers

  Total size of solvers is 559 kB.
  References face-detector-fast.solver, face-detector-eye-validation.solver, face-detector-landmarks-fast.solver

- accurate_server

Note

Not supported in this variant of Enrollment SDK.

Using of different modes can adjust trade-off between speed and accuracy of face detection. Face detection accuracy has two meaning:

1. ratio between false accepted faces and false rejected faces.
2. precision of facial keypoints detection.

3.4.2.1.1 Face Detection

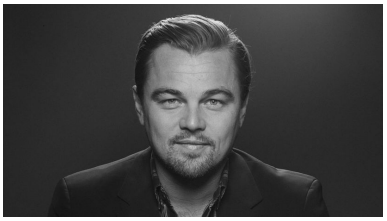


Figure 7 Face

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/face.png"));
auto fc = FaceCapture::Create(std::move(img));
auto cfg = FaceDetector::Config{};
cfg.SetMaxDetectionSize(0.9F);
FaceDetector fd(cfg);
auto face = fc->DetectWith(fd)[0];
auto cropCfg = FaceCropConfiguration();
cropCfg.SetCropMethod(FaceCrop::TOKEN);
auto croppedFace = face->Crop(cropCfg);

if (not croppedFace->IsCropRegionInsideImage())
{
    // shrink bounding box into image
    cropCfg.SetCropScale(0.90F);
    auto resizedCrop = face->Crop(cropCfg);

    auto resizedCropRegion =
      fc->Draw(resizedCrop->GetCropRectangle(), Shapes::InnoBlue);
    resizedCropRegion->SaveAs("output/face_crop_box_resized.png");
}
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));
FaceCapture capture = new FaceCapture(img);
FaceDetector.Config cfg = FaceDetector.Config.Default();
cfg.SetMaxDetectionSize(0.9F);
FaceDetector detector = new FaceDetector(cfg);
Face face = capture.DetectWith(detector).get(0);
FaceCropConfiguration cropCfg = new FaceCropConfiguration();
cropCfg.SetCropMethod(FaceCrop.TOKEN);
CroppedFace croppedFace = face.Crop(cropCfg);

if (croppedFace.IsCropRegionInsideImage() == false) {
    // shrink bounding box into image
    cropCfg.SetCropScale(0.90f);
    CroppedFace resizedCrop = face.Crop(cropCfg);

    Image resizedCropRegion = capture.Draw(
            resizedCrop.GetCropRectangle(), Shapes.InnoBlue);
    resizedCropRegion.SaveAs("output/face_crop_box_resized.png");
}
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));
FaceCapture capture = new FaceCapture(img);
```

```
FaceDetector detector = new FaceDetector();
Face face = capture.DetectWith(detector)[0];
FaceCropConfiguration cropCfg = new FaceCropConfiguration();
cropCfg.SetCropMethod(FaceCrop.TOKEN);
CroppedFace croppedFace = face.Crop(cropCfg);

if (croppedFace.IsCropRegionInsideImage() == false)
{
    // shrink bounding box into image
    cropCfg.SetCropScale(0.90f);
    CroppedFace resizedCrop = face.Crop(cropCfg);

    Image resizedCropRegion =
        capture.Draw(resizedCrop.GetCropRectangle(), Shapes.InnoBlue);
    resizedCropRegion.SaveAs("output/face_crop_box_resized.png");
}
```

The output of example

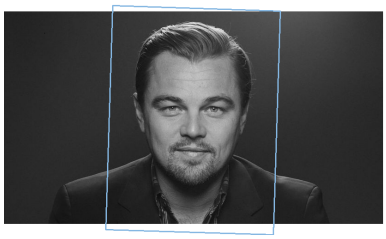


Figure 8 Face with crop region outside of image

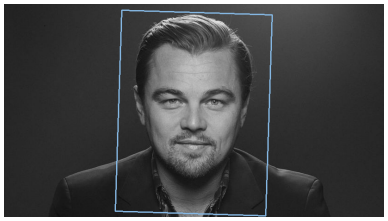


Figure 9 Face with shrunken crop

3.4.2.1.2 Face Detection with various crops

Once a face is detected, a Face object is provided. It is possible to set various crop methods via Face::Crop() method.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/face.png"));
auto fc = FaceCapture::Create(std::move(img));
auto cfg = FaceDetector::Config{};
cfg.SetMaxDetectionSize(0.9F);
FaceDetector fd(cfg);
auto face = fc->DetectWith(fd)[0];

FaceCropConfiguration cropCfg;
cropCfg.SetCropBackground(Colors::InnoBlue);
cropCfg.SetCropMethod(FaceCrop::FULL);
auto full = face->Crop(cropCfg)->GetFaceCapture();

cropCfg.SetCropMethod(FaceCrop::FULL_NOT_ALIGNED);
auto fullNotAligned = face->Crop(cropCfg)->GetFaceCapture();

cropCfg.SetCropMethod(FaceCrop::TOKEN);
auto token = face->Crop(cropCfg)->GetFaceCapture();

cropCfg.SetCropMethod(FaceCrop::TOKEN_NOT_FRONTAL);
auto tokenNotFrontal = face->Crop(cropCfg)->GetFaceCapture();
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));
FaceCapture capture = new FaceCapture(img);
FaceDetector detector = new FaceDetector();
Face face = capture.DetectWith(detector).get(0);

FaceCropConfiguration cropCfg = new FaceCropConfiguration();
cropCfg.SetCropBackground(Colors.InnoBlue);
cropCfg.SetCropMethod(FaceCrop.FULL);
FaceCapture full = face.Crop(cropCfg).GetFaceCapture();

cropCfg.SetCropMethod(FaceCrop.FULL_NOT_ALIGNED);
FaceCapture fullNotAligned = face.Crop(cropCfg).GetFaceCapture();

cropCfg.SetCropMethod(FaceCrop.TOKEN);
FaceCapture token = face.Crop(cropCfg).GetFaceCapture();

cropCfg.SetCropMethod(FaceCrop.TOKEN_NOT_FRONTAL);
FaceCapture tokenNotFrontal = face.Crop(cropCfg).GetFaceCapture();
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));
var capture = new FaceCapture(img);
var detector = new FaceDetector();
var face = capture.DetectWith(detector)[0];

var cropCfg = new FaceCropConfiguration();
cropCfg.SetCropBackground(Colors.InnoBlue);
cropCfg.SetCropMethod(FaceCrop.FULL);
var full = face.Crop(cropCfg).GetFaceCapture();

cropCfg.SetCropMethod(FaceCrop.FULL_NOT_ALIGNED);
var fullNotAligned = face.Crop(cropCfg).GetFaceCapture();

cropCfg.SetCropMethod(FaceCrop.TOKEN);
var token = face.Crop(cropCfg).GetFaceCapture();

cropCfg.SetCropMethod(FaceCrop.TOKEN_NOT_FRONTAL);
var tokenNotFrontal = face.Crop(cropCfg).GetFaceCapture();
```

The output of example

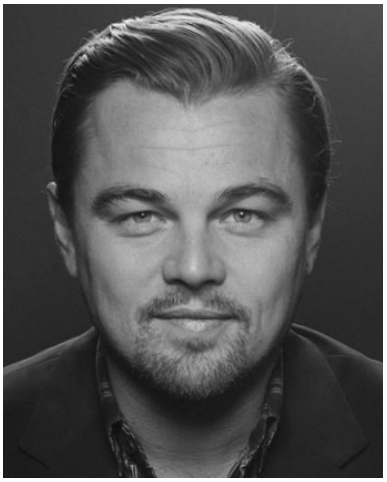


Figure 10 Face with FULL crop configuration

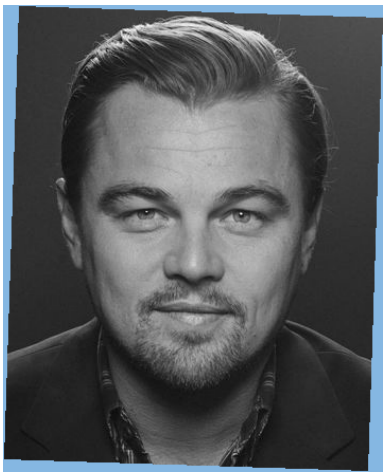


Figure 11 Face with FULL_NOT_ALIGNED crop

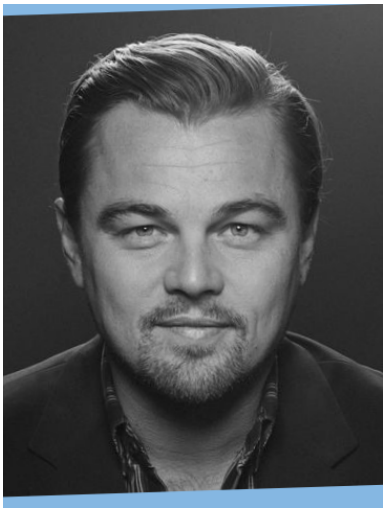


Figure 12 Face with TOKEN crop

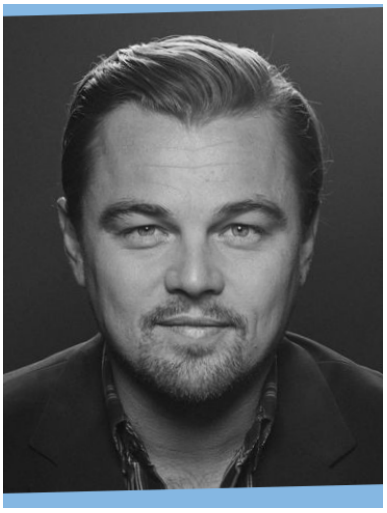


Figure 13 Face with TOKEN_NOT_FRONTAL crop

3.4.2.1.3 Faces Detection

It is possible to detect multiple faces (up to 16) in FaceCapture.

Figure 14 Faces

The faces are detected with function FaceCapture::DetectWith.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/faces.png"));
auto fc = FaceCapture::Create(std::move(img));
FaceDetector fd;
auto faces = fc->DetectWith(fd);

unsigned int index = 0;
for (const auto& face : faces)
{
    const std::string name = "output/face_" + std::to_string(index++) + ".png";
    face->Crop({})->GetFaceCapture()->image->SaveAs(name);
}

auto boundingBoxes = fc->Draw(faces, Shapes::InnoBlue);
boundingBoxes->SaveAs("output/faces_bounding_boxes.png");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/faces.png"));
FaceCapture capture = new FaceCapture(img);
FaceDetector detector = new FaceDetector();
Faces faces = capture.DetectWith(detector);

int index = 0;
for (Face face : faces) {
    face.Crop(new FaceCropConfiguration())
            .GetFaceCapture()
            .getImage()
            .SaveAs("output/face_" + index + ".png");
    index++;
}

Image boundingBoxes = capture.Draw(faces, Shapes.InnoBlue);
boundingBoxes.SaveAs("output/faces_bounding_boxes.png");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/faces.png"));
FaceCapture capture = new FaceCapture(img);
FaceDetector detector = new FaceDetector();
```

```
Faces faces = capture.DetectWith(detector);

int index = 0;
foreach (Face face in faces)
{
    face.Crop(new FaceCropConfiguration())
        .GetFaceCapture()
        .image.SaveAs("output/face_" + index + ".png");
    index++;
}

Image boundingBoxes = capture.Draw(faces, Shapes.InnoBlue);
boundingBoxes.SaveAs("output/faces_bounding_boxes.png");
```

The output of example are detected faces

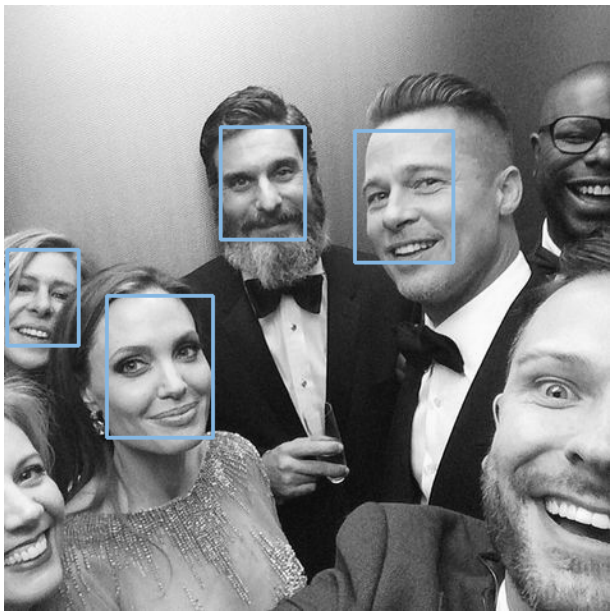


Figure 15 Detect Faces

and unique faces

Figure 16 Detected face 1

Figure 17 Detected face 2

Figure 18 Detected face 3

Figure 19 Detected face 4

3.4.2.2 Extraction

We provide Internal Innovatrics template for Face. The main goal is to create template for further processing e.g. 1:1 Verification.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/face.png"));
auto fc = FaceCapture::Create(std::move(img));
FaceDetector fd;
// Detect face
auto face = fc->DetectWith(fd)[0];
auto cfg = FaceExtractorConfig::Default();
auto faceTemplate = face->Extract(cfg);
BinaryFile::WriteAll("output/face.ics", faceTemplate->data);

std::cout « "   Template Version: " « faceTemplate->GetVersion() « std::endl;
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));

FaceCapture fc = new FaceCapture(img);
FaceDetector fd = new FaceDetector();
Face face = fc.DetectWith(fd).get(0);

FaceExtractorConfig cfg = FaceExtractorConfig.Default();
ExtractedFace faceTemplate = face.Extract(cfg);
BinaryFile.WriteAll("output/face.ics", faceTemplate.getData());

System.out.println("   Template Version: " + faceTemplate.GetVersion());
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));

FaceCapture fc = new FaceCapture(img);
FaceDetector fd = new FaceDetector();
Face face = fc.DetectWith(fd)[0];

FaceExtractorConfig cfg = FaceExtractorConfig.Default();
ExtractedFace faceTemplate = face.Extract(cfg);
BinaryFile.WriteAll("output/face.ics", faceTemplate.data);

System.Console.WriteLine("   Template Version: " + faceTemplate.GetVersion());
```

3.4.2.3 1 to 1 Verification

We provide verification of face templates. The main goal is to calculate Similarity Score for given probe and gallery templates. The examples uses same image for probe and gallery

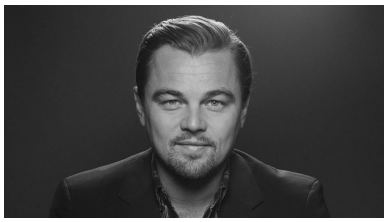


Figure 20 Face

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/face.png"));
auto fc = FaceCapture::Create(std::move(img));
FaceDetector fd;
auto face = fc->DetectWith(fd)[0];
auto probe = face->Extract();
auto gallery = face->Extract();
auto score = probe->SimilarWith(*gallery);
std::cout « "   Match Score: " « score « std::endl;
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));

FaceCapture fc = new FaceCapture(img);
FaceDetector fd = new FaceDetector();
Face face = fc.DetectWith(fd).get(0);

ExtractedFace probe = face.Extract();
ExtractedFace gallery = face.Extract();
long score = probe.SimilarWith(gallery);
System.out.println("      Match Score: " + score);
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));

FaceCapture fc = new FaceCapture(img);
FaceDetector fd = new FaceDetector();
Face face = fc.DetectWith(fd)[0];

ExtractedFace probe = face.Extract();
ExtractedFace gallery = face.Extract();
uint score = probe.SimilarWith(gallery);

System.Console.WriteLine("      Match Score: " + score);
```

3.4.2.4 Face Segmentation

It is possible to distinguish face and background once face is detected.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/face.png"));
auto fc = FaceCapture::Create(std::move(img));
const FaceDetector::Config cfg;
FaceDetector fd(cfg);
auto face = fc->DetectWith(fd)[0];
auto cropCfg = FaceCropConfiguration();
cropCfg.SetCropMethod(FaceCrop::FULL);
auto croppedFace = face->Crop(cropCfg);
auto maskImage = croppedFace->FaceAreaMask();
auto faceSegmentColor = croppedFace->FaceAreaSegment(Colors::InnoBlue.color);
auto faceSegmentTransparent = croppedFace->FaceAreaSegment();

maskImage->SaveAs("output/face_seg_mask.png");
```

```
faceSegmentColor->SaveAs("output/face_seg_color.png");
faceSegmentTransparent->SaveAs("output/face_seg_transparent.png");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));
FaceCapture capture = new FaceCapture(img);
FaceDetector detector = new FaceDetector();
Face face = capture.DetectWith(detector).get(0);
FaceCropConfiguration cropCfg = new FaceCropConfiguration();
cropCfg.SetCropMethod(FaceCrop.FULL);
CroppedFace croppedFace = face.Crop(cropCfg);
Image maskImage = croppedFace.FaceAreaMask();
Image faceSegmentColor = croppedFace.FaceAreaSegment(Colors.InnoBlue.getColor());
Image faceSegmentTransparent = croppedFace.FaceAreaSegment();

maskImage.SaveAs("output/face_seg_mask.png");
faceSegmentColor.SaveAs("output/face_seg_color.png");
faceSegmentTransparent.SaveAs("output/face_seg_transparent.png");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));
FaceCapture capture = new FaceCapture(img);
FaceDetector detector = new FaceDetector();
Face face = capture.DetectWith(detector)[0];
FaceCropConfiguration cropCfg = new FaceCropConfiguration();
cropCfg.SetCropMethod(FaceCrop.FULL);
CroppedFace croppedFace = face.Crop(cropCfg);
Image maskImage = croppedFace.FaceAreaMask();
Image faceSegmentColor = croppedFace.FaceAreaSegment(Colors.InnoBlue.color);
Image faceSegmentTransparent = croppedFace.FaceAreaSegment();

maskImage.SaveAs("output/face_seg_mask.png");
faceSegmentColor.SaveAs("output/face_seg_color.png");
faceSegmentTransparent.SaveAs("output/face_seg_transparent.png");
```

The output of example

Figure 21 Face with colored background

Figure 22 Masked Face

Figure 23 Face with transparent background

3.4.2.5 Face Attributes

3.4.2.5.1 ICAO and face image quality

The International Civil Aviation Organization (ICAO) defines global requirements for facial images used in machine-readable travel documents (MRTDs) in ICAO Doc 9303. These requirements describe how a compliant face image must be captured and presented: frontal pose, neutral expression, proper illumination, uniform background, correct head size and positioning. Compliant images ensure easy use for ID document personalization and good performance in both human verification and computer automated facial recognition.

These ICAO requirements are technically implemented and formalized in:

- ISO/IEC 39794-5 — Face image data format and capture requirements
- ISO/IEC 19794-5 — Legacy face image specifications
- ISO/IEC 29794-5 — Face image quality assessment

ISO/IEC 19794-5 defines four face image types: Basic (record format only), Frontal (with Full Frontal and Token Frontal subtypes), and specifies scene requirements (pose, expression, lighting, background, etc.), photographic requirements (exposure, focus, face position, head size), digital requirements (color profile, resolution, geometry), and format requirements (encoding, compression).

ISO/IEC 29794-5 defines a canonical face image as a face image conformant to an external standard or specification of a reference face image in most civil identity and travel document applications this corresponds to the ISO/IEC 39794-5 portrait specification, which reflects ICAO Doc 9303. [8]

3.4.2.5.2 Implementations and rationale

Enrollment provides two implementations for assessing face image quality against ICAO/ISO requirements:

- **Innovatrics** — proprietary implementation that checks mandatory and best-practice requirements of ICAO Doc 9303 and ISO/IEC 19794-5 for the full frontal face image type, with a rich set of individual attributes.
- **OFIQ (Open Face Image Quality)** — Implementation recognized by ISO/IEC 29794-5 as the reference for computing face image quality it outputs a unified quality score and standardized quality components.

When compliance with the standard or interchange with other systems is required, prefer **OFIQ**, as it is officially recognized by ISO/IEC 29794-5. The Innovatrics implementation is well-suited when fine-grained control over individual attributes or integration with other SDK features is needed.

3.4.2.5.3 Innovatrics implementation

The SDK (Innovatrics propietary implementation) is able to check all mandatory requirements and best practice recommendations of the ICAO Document 9303 specification and the ISO/IEC 19794-5 standard for the interoperable full frontal face image type, and of other relevant international standards (e.g. ANSI/INCITS 385-2004).

Once face is detected and Face object is provided, it is possible to get Face attributes.

- **Sharpness** - Sharpness attribute used to measure the sharpness level of an area of the detected face in the original face capture. See Face::Sharpness for more details.
- **Brightness** - Brightness attribute used to measure the brightness level of an area of the detected face in the original face capture. See Face::Brightness for more details.
- **Contrast** - Contrast attribute used to measure the contrast level of an area of the detected face in the original face capture. See Face::Contrast for more details.
- **Unique Intensity Levels** - Unique Intensity Levels attribute used to measure whether an area of the detected face in the original face capture has an appropriate number of unique intensity levels. See Face::UniqueIntensityLevels for more details.
- **Shadow** - Shadow attribute used to evaluate whether an area of the detected face in the original face capture is affected by shadows. See Face::Shadow for more details.
- **Nose Shadow** - Nose Shadow attribute for evaluating whether eyes or a nose don't cast sharp shadows. See Face::NoseShadow for more details.
- **Specularity** - Specularity attribute used to evaluate the presence of spotlights in an area of the detected face in the original face capture. See Face::Specularity for more details.
- **Right Red Eye** - Attribute for evaluating whether red-eye effect is not present on right eye. See Face::RightRedEye for more details.
- **Left Red Eye** - Attribute for evaluating whether red-eye effect is not present on left eye. See Face::LeftRedEye for more details.
- **Eye Distance** - Attribute used to measure the distance between the eyes in pixels for the detected face in the original face capture. See Face::EyeDistance for more details.
- **Roll Angle** - Attribute representing the head rotation angle around the Z-axis of the detected face in the original face capture, relative to the camera reference frame, as per DIN 9300. See Face::RollAngle for more details.
- **Pitch Angle** - Attribute representing the head rotation angle around the X-axis of the detected face in the original face capture, relative to the camera reference frame, as per DIN 9300. See Face::PitchAngle for more details.
- **Yaw Angle** - Attribute representing the head rotation angle around the Y-axis of the detected face in the original face capture, relative to the camera reference frame, as per DIN 9300. See Face::YawAngle for more details.
- **Face Size** - Attribute representing face size - the maximum of eye distance and eye-mouth distance. See Face::FaceSize for more details.
- **Face Relative Area** - Attribute representing the area of the detected face in the original face capture relative to the size of the original face capture. See Face::FaceRelativeArea for more details.
- **Face Relative Area In Image** - Attribute representing the visible area of the detected face in the original face capture relative to the total face area. See Face::FaceRelativeAreaInImage for more details.
- **Width To Height Ratio Of Image** - Attribute representing width to height aspect ratio of the original face capture. See Face::WidthHeightRatio for more details.

- **Tinted Glasses** - Attribute for evaluating tinted glasses presence. See Face::TintedGlasses for more details.
- **Gender** - Attribute for evaluating gender of subject. See Face::Gender for more details.
- **Eye Gaze** - Eye Gaze attribute used to evaluate whether the gaze direction of the detected face in the original face capture is frontal. See Face::EyeGaze for more details.
- **Right Eye Status** - Attribute for evaluating right eye status. See Face::RightEyeStatus for more details.
- **Left Eye Status** - Attribute for evaluating left eye status. See Face::LeftEyeStatus for more details.
- **Glass Status** - Attribute for evaluating glasses presence. See Face::GlassStatus for more details.
- **Heavy Frame** - Attribute for evaluating whether glasses with heavy frames are not present. See Face::HeavyFrame for more details.
- **Mouth Status** - Attribute for evaluating mouth status. See Face::MouthStatus for more details.
- **Background Uniformity** - Attribute used to measure background uniformity in the close area around the detected face in the original face capture. See Face::BackgroundUniformity for more details.
- **Age** - Attribute for evaluating age of subject using the face. See Face::Age for more details.

C++

```
std::cout « "   Sharpness: " « face->Sharpness() « std::endl;
std::cout « "   Brightness: " « face->Brightness() « std::endl;
std::cout « "   Contrast: " « face->Contrast() « std::endl;
std::cout « "   Unique Intensity Levels: " « face->UniqueIntensityLevels()
          « std::endl;
std::cout « "   Shadow: " « face->Shadow() « std::endl;
std::cout « "   Nose Shadow: " « face->NoseShadow() « std::endl;
std::cout « "   Specularity: " « face->Specularity() « std::endl;
std::cout « "   Right Red Eye: " « face->RightRedEye() « std::endl;
std::cout « "   Left Red Eye: " « face->LeftRedEye() « std::endl;
std::cout « "   Eye Distance: " « face->EyeDistance() « std::endl;
std::cout « "   Roll Angle: " « face->RollAngle() « std::endl;
std::cout « "   Pitch Angle: " « face->PitchAngle() « std::endl;
std::cout « "   Yaw Angle: " « face->YawAngle() « std::endl;
std::cout « "   Face Size: " « face->FaceSize() « std::endl;
std::cout « "   Face Relative Area: " « face->FaceRelativeArea() « std::endl;
std::cout « "   Face Relative Area In Image: " « face->FaceRelativeAreaInImage()
          « std::endl;
std::cout « "   Eye Gaze: " « face->EyeGaze() « std::endl;
std::cout « "   Right Eye Status: " « face->RightEyeStatus() « std::endl;
std::cout « "   Left Eye Status: " « face->LeftEyeStatus() « std::endl;
std::cout « "   Glass Status: " « face->GlassStatus() « std::endl;
std::cout « "   Heavy Frame: " « face->HeavyFrame() « std::endl;
std::cout « "   Mouth Status: " « face->MouthStatus() « std::endl;
std::cout « "   Background Uniformity: " « face->BackgroundUniformity(5)
          « std::endl;
std::cout « "   Age: " « face->Age() « std::endl;
std::cout « "   Gender: " « face->Gender() « std::endl;
std::cout « "   Tinted Glasses: " « face->TintedGlasses() « std::endl;
```

java

```
Face face = faces.get(0);
System.out.println("   Sharpness: " + face.Sharpness());
System.out.println("   Brightness: " + face.Brightness());
System.out.println("   Contrast: " + face.Contrast());
System.out.println("   Unique Intensity Levels: " + face.UniqueIntensityLevels());
System.out.println("   Shadow: " + face.Shadow());
System.out.println("   Nose Shadow: " + face.NoseShadow());
System.out.println("   Specularity: " + face.Specularity());
System.out.println("   Right Red Eye: " + face.RightRedEye());
System.out.println("   Left Red Eye: " + face.LeftRedEye());
System.out.println("   Eye Distance: " + face.EyeDistance());
System.out.println("   Roll Angle: " + face.RollAngle());
System.out.println("   Pitch Angle: " + face.PitchAngle());
System.out.println("   Yaw Angle: " + face.YawAngle());
System.out.println("   Face Size: " + face.FaceSize());
System.out.println("   Face Relative Area: " + face.FaceRelativeArea());
System.out.println(
        "   Face Relative Area In Image: " + face.FaceRelativeAreaInImage());
System.out.println("   Eye Gaze: " + face.EyeGaze());
System.out.println("   Right Eye Status: " + face.RightEyeStatus());
System.out.println("   Left Eye Status: " + face.LeftEyeStatus());
System.out.println("   Glass Status: " + face.GlassStatus());
System.out.println("   Heavy Frame: " + face.HeavyFrame());
```

```
System.out.println("   Mouth Status: " + face.MouthStatus());
System.out.println(
        "   Background Uniformity: " + face.BackgroundUniformity((byte) 5));
System.out.println("   Age: " + face.Age());
System.out.println("   Gender: " + face.Gender());
System.out.println("   Tinted Glasses: " + face.TintedGlasses());
```

csharp

```
Face face = faces[0];
System.Console.WriteLine("   Sharpness: " + face.Sharpness());
System.Console.WriteLine("   Brightness: " + face.Brightness());
System.Console.WriteLine("   Contrast: " + face.Contrast());
System.Console.WriteLine("   Unique Intensity Levels: " +
                        face.UniqueIntensityLevels());
System.Console.WriteLine("   Shadow: " + face.Shadow());
System.Console.WriteLine("   Nose Shadow: " + face.NoseShadow());
System.Console.WriteLine("   Specularity: " + face.Specularity());
System.Console.WriteLine("   Right Red Eye: " + face.RightRedEye());
System.Console.WriteLine("   Left Red Eye: " + face.LeftRedEye());
System.Console.WriteLine("   Eye Distance: " + face.EyeDistance());
System.Console.WriteLine("   Roll Angle: " + face.RollAngle());
System.Console.WriteLine("   Pitch Angle: " + face.PitchAngle());
System.Console.WriteLine("   Yaw Angle: " + face.YawAngle());
System.Console.WriteLine("   Face Size: " + face.FaceSize());
System.Console.WriteLine("   Face Relative Area: " + face.FaceRelativeArea());
System.Console.WriteLine("   Face Relative Area In Image: " +
                        face.FaceRelativeAreaInImage());
System.Console.WriteLine("   Eye Gaze: " + face.EyeGaze());
System.Console.WriteLine("   Right Eye Status: " + face.RightEyeStatus());
System.Console.WriteLine("   Left Eye Status: " + face.LeftEyeStatus());
System.Console.WriteLine("   Glass Status: " + face.GlassStatus());
System.Console.WriteLine("   Heavy Frame: " + face.HeavyFrame());
System.Console.WriteLine("   Mouth Status: " + face.MouthStatus());
System.Console.WriteLine("   Background Uniformity: " +
                        face.BackgroundUniformity(5));
System.Console.WriteLine("   Age: " + face.Age());
System.Console.WriteLine("   Gender: " + face.Gender());
System.Console.WriteLine("   Tinted Glasses: " + face.TintedGlasses());
```

3.4.2.5.4 OFIQ implementation

ISO/IEC 29794-5 recognizes OFIQ (Open Face Image Quality) as the reference implementation for computing face image quality. [8]

OFIQ computes [8] :

- A Unified Quality Score (QS) in the range 0–100
- Individual quality components, including:
  - Head pose (yaw, pitch, roll)
  - Head size and image margins
  - Inter-eye distance
  - Eye visibility and occlusion
  - Mouth closed / expression neutrality
  - Sharpness and focus
  - Illumination uniformity
  - Background uniformity
  - Exposure control
  - Natural colour
  - Compression artefacts

The unified quality score predicts the expected recognition performance of a face image under the assumption that it will be compared against a canonical ISO/ICAO-compliant portrait image. When face images are being collected from many biometric capture subjects, the unified quality score and quality components can be aggregated to summarize the effectiveness of the collection (e.g. mean or proportion with low or high value), to reveal site-specific problems or population effects, or as a response variable in A-B tests or trend analysis. [8] Higher scores indicate stronger conformance to ISO/ICAO portrait requirements and higher expected face recognition performance.

c++

```
auto image = Image::Decode(BinaryFile::ReadAll("assets/face.png"));
auto faceCapture = FaceCapture::Create(std::move(image), {});

auto ofiq = OFIQFace::Create(faceCapture);

auto bbImage = faceCapture->Draw(ofiq->GetBoundingBox(), Shapes::InnoBlue);
bbImage->SaveAs("output/ofiq_bounding_box.png");

std::cout « "   Sharpness: " « static_cast<int>(ofiq->GetSharpness())
          « std::endl;
std::cout « "   UnifiedQualityScore: "
          « static_cast<int>(ofiq->GetUnifiedQualityScore()) « std::endl;
std::cout « "   BackgroundUniformity: "
          « static_cast<int>(ofiq->GetBackgroundUniformity()) « std::endl;
// Similarly for other OFIQFace attributes
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/face.png"));
FaceCapture faceCapture = new FaceCapture(img, new FaceAttributes());

OFIQFace ofiq = OFIQFace.Create(faceCapture);

Image bbImage = faceCapture.Draw(ofiq.GetBoundingBox(), Shapes.InnoBlue);
bbImage.SaveAs("output/ofiq_bounding_box.png");

System.out.println("   Sharpness: " + ofiq.GetSharpness());
System.out.println("   UnifiedQualityScore: " + ofiq.GetUnifiedQualityScore());
System.out.println("   BackgroundUniformity: " + ofiq.GetBackgroundUniformity());
// Similarly for other OFIQFace attributes
```

csharp

```
Image image = Image.Decode(BinaryFile.ReadAll("assets/face.png"));
FaceCapture faceCapture = new FaceCapture(image, new FaceAttributes());

OFIQFace ofiq = OFIQFace.Create(faceCapture);

Image bbImage = faceCapture.Draw(ofiq.GetBoundingBox(), Shapes.InnoBlue);
bbImage.SaveAs("output/ofiq_bounding_box.png");

System.Console.WriteLine("   Sharpness: {0}", ofiq.GetSharpness());
System.Console.WriteLine("   UnifiedQualityScore: {0}",
                         ofiq.GetUnifiedQualityScore());
System.Console.WriteLine("   BackgroundUniformity: {0}",
                         ofiq.GetBackgroundUniformity());
// Similarly for other OFIQFace attributes
```

The detected face by OFIQFace from example

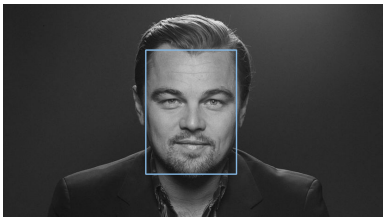


Figure 24 OFIQ face

3.4.2.6 Keypoints

The Face allows also retrieval of facial keypoints.

- **Right Eye Outer Corner** - Key point for evaluating reliability of right eye outer corner. See Face::Keypoints::RightEyeOuterCorner for more details.
- **Right Eye Centre** - Key point for evaluating reliability of right eye centre. See Face::Keypoints::RightEyeCentre for more details.
- **Left Eye Inner Corner** - Key point for evaluating reliability of inner corner of left eye. See Face::Keypoints::LeftEyeInnerCorner for more details.
- **Left Eye Centre** - Key point for evaluating reliability of left eye centre. See Face::Keypoints::LeftEyeCentre for more details.
- **Left Eye Outer Corner** - Key point for evaluating reliability of outer corner of left eye. See Face::Keypoints::LeftEyeOuterCorner for more details.
- **Nose Root** - Key point for evaluating reliability of nose root. See Face::Keypoints::NoseRoot for more details.
- **Nose Right Bottom** - Key point for evaluating reliability of right bottom of nose. See Face::Keypoints::NoseRightBottom for more details.
- **Nose Tip** - Key point for evaluating reliability of nose tip. See Face::Keypoints::NoseTip for more details.
- **Nose Left Bottom** - Key point for evaluating reliability of left bottom of nose. See Face::Keypoints::NoseLeftBottom for more details.
- **Nose Bottom** - Key point for evaluating reliability of bottom of nose. See Face::Keypoints::NoseBottom for more details.
- **Mouth Right Corner** - Key point for evaluating reliability of right corner of mouth. See Face::Keypoints::MouthRightCorner for more details.
- **Mouth Center** - Key point for evaluating reliability of center of mouth. See Face::Keypoints::MouthCenter for more details.
- **Mouth Left Corner** - Key point for evaluating reliability of left corner of mouth. See Face::Keypoints::MouthLeftCorner for more details.
- **Mouth Upper Edge** - Key point for evaluating reliability of upper edge of mouth. See Face::Keypoints::MouthUpperEdge for more details.
- **Mouth Lower Edge** - Key point for evaluating reliability of lower edge of mouth. See Face::Keypoints::MouthLowerEdge for more details.
- **Right Eyebrow Outer End** - Key point for evaluating reliability of outer end of right eyebrow. See Face::Keypoints::RightEyebrowOuterEnd for more details.
- **Right Eyebrow Inner End** - Key point for evaluating reliability of inner end of right eyebrow. See Face::Keypoints::RightEyebrowInnerEnd for more details.
- **Left Eyebrow Inner End** - Key point for evaluating reliability of inner end of left eyebrow. See Face::Keypoints::LeftEyebrowInnerEnd for more details.
- **Left Eyebrow Outer End** - Key point for evaluating reliability of outer end of left eyebrow. See Face::Keypoints::LeftEyebrowOuterEnd for more details.
- **Right Edge** - Key point for evaluating reliability of right edge of face. See Face::Keypoints::RightEdge for more details.
- **Chin Tip** - Key point for evaluating reliability of tip of chin. See Face::Keypoints::ChinTip for more details.
- **Left Edge** - Key point for evaluating reliability of left edge of face. See Face::Keypoints::LeftEdge for more details.

c++

```
auto keypoints = face->GetKeypoints();

DotsShape dots;
dots.Add(keypoints->RightEyeOuterCorner());
dots.Add(keypoints->RightEyeCentre());
dots.Add(keypoints->RightEyeInnerCorner());
dots.Add(keypoints->LeftEyeInnerCorner());
dots.Add(keypoints->LeftEyeCentre());
dots.Add(keypoints->LeftEyeOuterCorner());
dots.Add(keypoints->NoseRoot());
dots.Add(keypoints->NoseRightBottom());
dots.Add(keypoints->NoseTip());
dots.Add(keypoints->NoseLeftBottom());
dots.Add(keypoints->NoseBottom());
dots.Add(keypoints->MouthRightCorner());
dots.Add(keypoints->MouthCenter());
dots.Add(keypoints->MouthLeftCorner());
dots.Add(keypoints->MouthUpperEdge());
dots.Add(keypoints->MouthLowerEdge());
dots.Add(keypoints->RightEyebrowOuterEnd());
dots.Add(keypoints->RightEyebrowInnerEnd());
dots.Add(keypoints->LeftEyebrowInnerEnd());
dots.Add(keypoints->LeftEyebrowOuterEnd());
dots.Add(keypoints->RightEdge());
dots.Add(keypoints->ChinTip());
dots.Add(keypoints->LeftEdge());

auto keypointImg = keypoints->GetImage()->DrawShape(dots);
keypointImg->SaveAs("output/keypoints.png");
```

java

```
Face face = faces.get(0);
Face.Keypoints keypoints = face.GetKeypoints();

DotsShape dots = new DotsShape();
dots.Add(keypoints.RightEyeOuterCorner());
dots.Add(keypoints.RightEyeCentre());
dots.Add(keypoints.RightEyeInnerCorner());
dots.Add(keypoints.LeftEyeInnerCorner());
dots.Add(keypoints.LeftEyeCentre());
dots.Add(keypoints.LeftEyeOuterCorner());
dots.Add(keypoints.NoseRoot());
dots.Add(keypoints.NoseRightBottom());
dots.Add(keypoints.NoseTip());
dots.Add(keypoints.NoseLeftBottom());
dots.Add(keypoints.NoseBottom());
dots.Add(keypoints.MouthRightCorner());
dots.Add(keypoints.MouthCenter());
dots.Add(keypoints.MouthLeftCorner());
dots.Add(keypoints.MouthUpperEdge());
dots.Add(keypoints.MouthLowerEdge());
dots.Add(keypoints.RightEyebrowOuterEnd());
dots.Add(keypoints.RightEyebrowInnerEnd());
dots.Add(keypoints.LeftEyebrowInnerEnd());
dots.Add(keypoints.LeftEyebrowOuterEnd());
dots.Add(keypoints.RightEdge());
dots.Add(keypoints.ChinTip());
dots.Add(keypoints.LeftEdge());

Image keypointImg = keypoints.GetImage().DrawShape(dots);
keypointImg.SaveAs("output/keypoints.png");
```

csharp

```
var face = faces[0];
var keypoints = face.GetKeypoints();

var dots = new DotsShape();
dots.Add(keypoints.RightEyeOuterCorner());
dots.Add(keypoints.RightEyeCentre());
dots.Add(keypoints.RightEyeInnerCorner());
dots.Add(keypoints.LeftEyeInnerCorner());
dots.Add(keypoints.LeftEyeCentre());
dots.Add(keypoints.LeftEyeOuterCorner());
dots.Add(keypoints.NoseRoot());
dots.Add(keypoints.NoseRightBottom());
dots.Add(keypoints.NoseTip());
dots.Add(keypoints.NoseLeftBottom());
dots.Add(keypoints.NoseBottom());
dots.Add(keypoints.MouthRightCorner());
dots.Add(keypoints.MouthCenter());
dots.Add(keypoints.MouthLeftCorner());
dots.Add(keypoints.MouthUpperEdge());
dots.Add(keypoints.MouthLowerEdge());
```

```
dots.Add(keypoints.RightEyebrowOuterEnd());
dots.Add(keypoints.RightEyebrowInnerEnd());
dots.Add(keypoints.LeftEyebrowInnerEnd());
dots.Add(keypoints.LeftEyebrowOuterEnd());
dots.Add(keypoints.RightEdge());
dots.Add(keypoints.ChinTip());
dots.Add(keypoints.LeftEdge());

var keypointImg = keypoints.GetImage().DrawShape(dots);
keypointImg.SaveAs("output/keypoints.png");
```

Figure 25 Face keypoints

3.4.2.7 Face Passive Liveness

Note

Access to the passive liveness feature requires explicit permission in the license to enable face passive liveness.

We are providing following passive liveness calculations:

- fast - Passive liveness mode with best performance available but worse accuracy as ACCURATE mode.

The Face::PassiveLivenessMode::FAST operates with higher speed but lower accuracy, whereas the Face::PassiveLivenessMode::ACCURATE is slower but provides increased accuracy, irrespective of the chosen FaceDetector::Config.

c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/liveness_face.png"));
auto capture = FaceCapture::Create(std::move(img));
FaceDetector fd;
auto face = capture->DetectWith(fd)[0];

auto plf = face->PassiveLiveness(Face::PassiveLivenessMode::FAST);
std::cout « "   Fast PL: " « plf « std::endl;
auto pla = face->PassiveLiveness(Face::PassiveLivenessMode::ACCURATE);
std::cout « "   Accurate PL: " « pla « std::endl;
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/liveness_face.png"));
FaceCapture capture = new FaceCapture(img);
FaceDetector fd = new FaceDetector();
Face face = capture.DetectWith(fd).get(0);

System.out.println("   Fast PL: "
                   + face.PassiveLiveness(Face.PassiveLivenessMode.FAST));
System.out.println("   Accurate PL: "
                   + face.PassiveLiveness(Face.PassiveLivenessMode.ACCURATE));
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/liveness_face.png"));
FaceCapture capture = new FaceCapture(img);
FaceDetector fd = new FaceDetector();
Face face = capture.DetectWith(fd)[0];

System.Console.WriteLine("   Fast PL: " +
                         face.PassiveLiveness(Face.PassiveLivenessMode.FAST));
System.Console.WriteLine("   Accurate PL: " +
                         face.PassiveLiveness(Face.PassiveLivenessMode.ACCURATE));
```

Passive Liveness Scores, Error Rates and Accuracy

- **APCER**: Attack presentation images that are classified as bona-fide presentations are false accepts. The percentual rate of such error on a given dataset and given threshold represents the Attack Presentation Classification Error Rate (APCER, formerly FAR).
- **BPCER**: Bona-fide presentation images that are classified as attacks are false rejects. The percentual rate of such error on a given dataset and given threshold represents the Bona-fide Presentation Classification Error Rate (BPCER, formerly FRR).

Example:

Imagine a dataset of 10,000 bona-fide presentation photos (real faces) and 1,000 attack presentation photos, where measurements were made. Threshold of 89.5, which is at working point of 1% APCER results in 3.7% BPCER. That means there are 10 attack presentation photos marked as bona-fide (false accepts) and 370 bona-fide photos are marked as attacks (false rejects).

3.4.2.7.1 Fast Passive Liveness Thresholds

Use case type	Threshold	Performance
Convenience (minimum rejected attempts)	**80.↩ 0**	**2.75% APCER** @ 1 % BPCER
Balanced (equal error rate)	**83.↩ 2**	**1.76%** both APCER & BPCER
Security (minimum accepted frauds)	**86.↩ 2**	**3.62% BPCER** @ 1% APCER

3.4.2.8 Face Attributes Reliability

In some cases the meaningfulness of attribute value relies on values of other attributes. We provide following reliability checks

- ICAOReliability - Check if Face will fullfil ICAO requirements.
- TemplateReliability - Check if template extraction will be reliable.
- AgeReliability - Check if calculation of Age attribute will be reliable.
- GlassStatusReliability - Check if calculation of Glass Status attribute will be reliable.
- TintedGlassesReliability - Check if calculation of Tinted Glasses attribute will be reliable.
- SegmentationMaskReliability - Check if calculation of Segmentation Mask in functions Face::FaceArea↩ Segment, Face::FaceAreaSegment(RGB), Face::FaceAreaMask will be reliable.
- FaceSizeReliability - Check if calculation of Face Size attribute will be reliable.
- MouthOpennessReliability - Check if calculation of Mouth Openness attribute will be reliable.
- UniversalPassiveLivenessReliability - Check if calculation of Universal passive liveness will be reliable.

c++

```
ICAOReliability icao;
auto failedAttributes = icao.Validate(face);
for (const auto& a : failedAttributes)
{
    std::cout « "   Value " « a.value « " of " « a.id « " is not in range ["
              « a.min « ", " « a.max « "]" « std::endl;
}
```

java

```
ICAOReliability icao = new ICAOReliability();
for (FailedFaceAttribute a : icao.Validate(face)) {
    System.out.println("   Value " + a.getValue() + " of " + a.getId()
                       + " is not in range [" + a.getMin() + ", " + a.getMax()
                       + "]");
}
```

csharp

```
ICAOReliability icao = new ICAOReliability();
foreach (FailedFaceAttribute a in icao.Validate(face))
{
    System.Console.WriteLine("   Value " + a.value + " of " + a.id +
                             " is not in range [" + a.min + ", " + a.max + "]");
}
```

3.4.2.9 Best practices for use of Face Images

We provide following best practice checks

- ISOFullFrontalImageValidator - Check if usage of Full Frontal Images on travel documents will follow best practices, as described in Section B.3.2 Best practices for use of Full Frontal Images on travel documents [6] .

c++

```
ISOFullFrontalImageValidator icaoFullFrontal;
auto failedAttributes = icaoFullFrontal.Validate(face);
for (const auto& a : failedAttributes)
{
    std::cout « "   Value " « a.value « " of " « a.id « " is not in range ["
              « a.min « ", " « a.max « "]" « std::endl;
}
```

java

```
ISOFullFrontalImageValidator icaoFullFrontal = new ISOFullFrontalImageValidator();
for (FailedFaceAttribute a : icaoFullFrontal.Validate(face)) {
    System.out.println("   Value " + a.getValue() + " of " + a.getId()
                       + " is not in range [" + a.getMin() + ", " + a.getMax()
                       + "]");
}
```

csharp

```
ISOFullFrontalImageValidator icaoFullFrontal = new ISOFullFrontalImageValidator();
foreach (FailedFaceAttribute a in icaoFullFrontal.Validate(face))
{
    System.Console.WriteLine("   Value " + a.value + " of " + a.id +
                             " is not in range [" + a.min + ", " + a.max + "]");
}
```

3.4.2.10 ISO Conversion

3.4.2.10.1 ISO Image

It is possible to convert Face to ISO image compliant with ISO 19794-5:2011 and back from image to FaceCapture.

C++

```
auto image = Image::Decode(BinaryFile::ReadAll("assets/faces.png"));

FaceAttributes fa;
fa.SetCaptureDeviceTechnology(
  FaceCaptureDeviceTechnology::STATIC_PHOTOGRAPH_FROM_SCANNER);
fa.SetCaptureDeviceVendorID(FaceCaptureDeviceVendorID(11));
fa.SetCaptureDeviceTypeID(FaceCaptureDeviceTypeID(7));

auto faceCapture = FaceCapture::Create(std::move(image), fa);
FaceDetector faceDetector;
auto face = faceCapture->DetectWith(faceDetector)[0];

std::cout « "   Conversion Face to ISO image" « std::endl;
auto encoder = PngImageEncoder();
auto isoImage = face->ToIsoImage(encoder);

std::cout « "   Conversion ISO to Face image" « std::endl;
auto faceImage = faceCapture->FromIsoImage(isoImage);
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/faces.png"));

FaceAttributes fa = new FaceAttributes();
fa.SetCaptureDeviceTechnology(
        FaceCaptureDeviceTechnology.STATIC_PHOTOGRAPH_FROM_SCANNER);
fa.SetCaptureDeviceVendorID(new FaceCaptureDeviceVendorID(11));
fa.SetCaptureDeviceTypeID(new FaceCaptureDeviceTypeID(7));

FaceCapture faceCapture = new FaceCapture(img, fa);
FaceDetector faceDetector = new FaceDetector();
Face face = faceCapture.DetectWith(faceDetector).get(0);

System.out.println("   Conversion Face to ISO image");
PngImageEncoder encoder = new PngImageEncoder();
Bytes isoImage = face.ToIsoImage(encoder);

System.out.println("   Conversion ISO to Face image");
FaceCapture faceImage = FaceCapture.FromIsoImage(isoImage);
```

csharp

```
Image image = Image.Decode(BinaryFile.ReadAll("assets/faces.png"));

FaceAttributes fa = new FaceAttributes();
fa.SetCaptureDeviceTechnology(
    FaceCaptureDeviceTechnology.STATIC_PHOTOGRAPH_FROM_SCANNER);
fa.SetCaptureDeviceVendorID(new FaceCaptureDeviceVendorID(11));
fa.SetCaptureDeviceTypeID(new FaceCaptureDeviceTypeID(7));

FaceCapture faceCapture = new FaceCapture(image, fa);
FaceDetector faceDetector = new FaceDetector();
var face = faceCapture.DetectWith(faceDetector)[0];

System.Console.WriteLine("   Conversion Face to ISO image");
var encoder = new PngImageEncoder();
var isoImage = face.ToIsoImage(encoder);

System.Console.WriteLine("   Conversion ISO to Face image");
var faceImage = FaceCapture.FromIsoImage(isoImage);
```

3.4.3 Typedef Documentation

3.4.3.1 FaceSimilarityScore

```
using FaceSimilarityScore = unsigned int
```

Score is 0-100 where 100 represents most similar templates.

When the matching score is higher then certain score threshold then the face images belongs to the same person with high probability. The matching score range is <0, 100>. Its values can be interpreted as follows:

- Low values of the score, i.e. range <0, 60>, are normalized using FAR values and this formula `score_L=-10*log(FAR)`. It means that score 30 is related to FAR=1:1000=10^-3, score 50 is related to FAR=1:100000=10^-5 (evaluated on our large testing non-matching pairs dataset).
- High values of the score, i.e. range <80, 100>, are normalized using FRR values and this formula `score_H=100/3*(FRR + 2)`. It means that score 80 is related to FRR=0.4, score 90 is related to FRR=0.7 (evaluated on our large testing matching pairs dataset).

Scores values in range (60, 80) are weighted average of `score_L` and `score_H`. This normalization help the users to select the score threshold according their needs. If it is too low e.g. score threshold is 30, then the chance of false accepted non-matching faces is quite high (FAR=10^-3). When it is too high e.g. score threshold is 90, then the chance of false rejected matching faces is quite high (FRR=0.7). For face recognition following recommendations are given:

- minimum face size - inter eye-pupils distance at least 40 pix
- optimal face size - inter eye-pupils distance at least 100 pix
- optimal image - sharp, contrast, frontal facial image with standard lighting conditions
- the more similar are conditions when the matched faces are captured the higher is the matching accuracy
- complex robust feature extraction invariant to partial faces occlusions / poor lighting / blurriness / etc is done. So, no further preprocessing is necessary.

See Similarity Score.

3.4.4 Enumeration Type Documentation

3.4.4.1 FaceCaptureDeviceTechnology

```
enum class FaceCaptureDeviceTechnology : uint8_t [strong]
```

Enumerator

UNKNOWN	Unknown or Unspecified.
STATIC_PHOTOGRAPH_FROM_UNKNOWN_SOURCE	Static photograph from an unknown source.
STATIC_PHOTOGRAPH_FROM_DIGITAL_CAMERA	Static photograph from a digital still-image camera.
STATIC_PHOTOGRAPH_FROM_SCANNER	Static photograph from a scanner.
VIDEO_FRAME_FROM_UNKNOWN_SOURCE	Video frame(s) from an unknown source.
VIDEO_FRAME_FROM_ANALOGUE_CAMERA	Video frame(s) from an analogue video camera.
VIDEO_FRAME_FROM_DIGITAL_CAMERA	Video frame(s) from a digital video camera.

3.4.4.2 FaceCrop

```
enum class FaceCrop : uint8_t [strong]
```

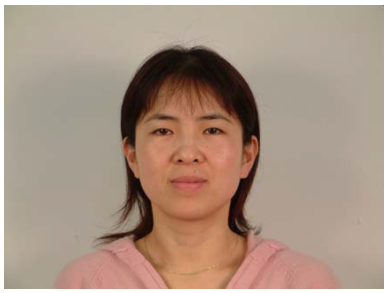
Enumerator

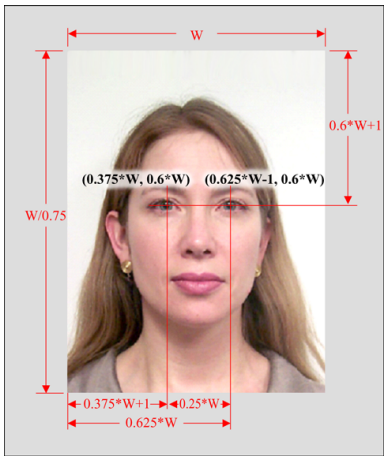
TOKEN_NOT_FRONTAL	Image cropping method similar to TOKEN but works quite well even on non-frontal images It will produce FaceImageType::BASIC.
FULL	Full Frontal Image cropping method defined in ISO/IEC 19794-5 standard. Cropping requirements specified to allow for full capture of face and shoulders. Consistent with most current mug-shot and passport standards. Recommended for use with ePassports, both on the printed passport and stored in the chip. [10] It will produce FaceImageType::FULL_FRONTAL image.
TOKEN	Token Frontal Image cropping method defined in ISO/IEC 19794-5 standard. Eye positions in fixed positions on image. The image aspect ratio is fixed. Storage size is reduced. A 90 or 120 pixels from eye to eye Token can be used in the MRTD chip. [10] It will produce FaceImageType::TOKEN_FRONTAL image.
FULL_NOT_ALIGNED	Crops bounding box provided by FaceCrop::FULL method as is from original FaceCapture. It will produce FaceImageType::BASIC. It is used to get face from original FaceCapture as is.

3.4.4.3 FaceImageType

```
enum class FaceImageType : uint8_t [strong]
```

Enumerator

BASIC	This is the fundamental Face Image Type that specifies a record format including header and representation data. All Face Image Types adhere to the properties of this type. No mandatory scene, photographic and digital requirements are specified for this image type. [6]
FULL_FRONTAL	A Face Image Type that specifies frontal images with sufficient resolution for human examination as well as reliable computer face recognition. This type of Face Image Type includes the full head with all hair in most cases, as well as neck and shoulders. This image type is suitable for permanent storage of the face information, and it is applicable to portraits for passport, driver license, and "mugshot" images. [6] 

TOKEN_FRONTAL	A Face Image Type that specifies frontal images with a specific geometric size and eye positioning based on the width and height of the image. This image type is suitable for minimizing the storage requirements for computer face recognition tasks such as verification while still offering vendor independence and human verification (versus human examination which requires more detail) capabilities. [6] 
POST_PROCESSED_FRONTAL	Applying digital post-processing to a captured image can modify this image in a way that it is more suitable for automatic face recognition. The Post-processed Frontal Face Image Type is thought of as the interchange format for these kinds of facial images. [6]

3.4.5 Function Documentation

3.4.5.1 FaceToIsoCaptureDeviceTechnology()

```
constexpr int FaceToIsoCaptureDeviceTechnology (
            FaceCaptureDeviceTechnology printCaptureDeviceTechnology) [static], [constexpr]
```

Parameters

printCaptureDeviceTechnology	FaceCaptureDeviceTechnology

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

ISO capture technology

3.4.5.2 FaceToIsoImageType()

```
constexpr int FaceToIsoImageType (
            FaceImageType faceImageType)  [static], [constexpr]
```

Parameters

faceImageType	FaceImageType

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

ISO face image type

3.4.5.3 IsFaceCropMethodValid()

```
constexpr bool IsFaceCropMethodValid (
            FaceCrop faceCrop)  [static], [constexpr]
```

Parameters

faceCrop	face crop method

Returns

Whether the face crop method is valid

3.4.5.4 IsoToFaceCaptureDeviceTechnology()

```
constexpr FaceCaptureDeviceTechnology IsoToFaceCaptureDeviceTechnology (
            int isoCaptureDeviceTechnology)  [static], [constexpr]
```

Parameters

isoCaptureDeviceTechnology	iso capture technology

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

FaceCaptureDeviceTechnology

3.4.5.5 IsoToFaceImageType()

```
constexpr FaceImageType IsoToFaceImageType (
            int faceImageType) [static], [constexpr]
```

Parameters

faceImageType	iso face image type

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

FaceImageType

3.4.6 Solvers

The Enrollment SDK package includes large binary files called *solvers*.

These solvers contain neural models utilized by specific features. They are shipped in an `iface/models` directory (for example `lib/iface/models` in the Linux/Android package layout).

At runtime the SDK expects solver files in `iface/models` next to the loaded Enrollment SDK library. Use DefaultModelsDirectory to obtain that path see Initialization.

During library load, if a license file is found and the default models directory exists on disk, the SDK can initialize automatically. Otherwise, provide the path via functions `Initialize(const std::vector<uint8_t>& license, const std::string& modelsPath)`/`Initialize(const InitializerConfig& config)`, (or, for Java, EnrollmentInitializer.LoadNativeLibrary).

3.4.6.1 Solvers Utilization

If your current use-case does not require a certain function, you can omit the corresponding solver from your installation, thereby saving space on the filesystem.

3.4.6.1.1 face-verification-accurate.solver

The solver size is 73 MB.

References

FaceExtractorConfig::Type::ACCURATE

3.4.6.1.2 face-verification-balanced.solver

The solver size is 15 MB.

References

FaceExtractorConfig::Type::BALANCED

3.4.6.1.3 face-verification-fast.solver

The solver size is 3 MB.

References

FaceExtractorConfig::Type::FAST

3.4.6.1.4 face-segmentation-mask.solver

The solver size is 52 MB.

References

CroppedFace::FaceAreaMask, CroppedFace::FaceAreaSegment

3.4.6.1.5 face-attribute-eye-gaze.solver

The solver size is 944 kB.

References

Face::EyeGaze, ICAOReliability::Validate

3.4.6.1.6 face-attribute-eye-status.solver

The solver size is 16 kB.

References

Face::RightEyeStatus, Face::LeftEyeStatus, ICAOReliability::Validate

3.4.6.1.7 face-attribute-glasses.solver

The solver size is 608 kB.

References

Face::GlassStatus, Face::TintedGlasses

3.4.6.1.8 face-attribute-heavy-frame.solver

The solver size is 498 kB.

References

Face::HeavyFrame, ICAOReliability::Validate

3.4.6.1.9 face-attribute-mouth-status.solver

The solver size is 155 kB.

References

Face::MouthStatus, ICAOReliability::Validate

3.4.6.1.10 face-attribute-background-uniformity.solver

The solver size is 1 MB.

References

Face::BackgroundUniformity, ICAOReliability::Validate

3.4.6.1.11 face-attribute-age-gender-balanced.solver

The solver size is 8 MB.

References

Face::AgeEstimationMode::BALANCED, Face::Age, Face::GenderEstimationMode::BALANCED, Face::Gender

3.4.6.1.12 face-attribute-age-gender-accurate.nnkitsolver

The solver size is 313 MB.

References

Face::AgeEstimationMode::ACCURATE, Face::GenderEstimationMode::ACCURATE

3.4.6.1.13 face-attribute-passive-liveness-universal-fast.nnkitsolver

The solver size is 5 MB.

References

Face::PassiveLivenessMode::FAST

3.4.6.1.14 face-detector-accurate.solver

The solver size is 9 MB.

References

FaceDetector::Config::Type::ACCURATE, FaceDetector::Config::Type::BALANCED

3.4.6.1.15 face-detector-landmarks-mob-0.50.solver

The solver size is 17 MB.

References

FaceDetector::Config::Type::ACCURATE

3.4.6.1.16 face-detector-eye-validation.solver

The solver size is 94 kB.

References

FaceDetector::Config::Type::ACCURATE, FaceDetector::Config::Type::BALANCED, FaceDetector::Config::Type::FAST

3.4.6.1.17 face-detector-landmarks-mob-0.25.solver

The solver size is 4 MB.

References

FaceDetector::Config::Type::BALANCED

3.4.6.1.18 face-detector-fast.solver

The solver size is 221 kB.

References

FaceDetector::Config::Type::FAST

3.4.6.1.19 face-detector-landmarks-fast.solver

The solver size is 242 kB.

References

FaceDetector::Config::Type::FAST

3.4.6.1.20 ofiq/expression_neutrality/grimmer/hse_1_2_C_adaboost.yml

The solver size is 216 MB.

References

OFIQFace::Create

3.4.6.1.21 ofiq/expression_neutrality/hsemotion/enet_b0_8_best_vgaf_embed_zeroed.onnx

The solver size is 15 MB.

References

OFIQFace::Create

3.4.6.1.22 ofiq/expression_neutrality/hsemotion/enet_b2_8_embed_zeroed.onnx

The solver size is 29 MB.

References

OFIQFace::Create

3.4.6.1.23 ofiq/face_detection/ssd_facedetect.caffemodel

The solver size is 10 MB.

References

OFIQFace::Create

3.4.6.1.24 ofiq/face_detection/ssd_facedetect.prototxt.txt

The solver size is 29 kB.

References

OFIQFace::Create

3.4.6.1.25 ofiq/face_landmark_estimation/ADNet.onnx

The solver size is 52 MB.

References

OFIQFace::Create

3.4.6.1.26 ofiq/face_occlusion_segmentation/face_occlusion_segmentation_ort.onnx

The solver size is 54 MB.

References

OFIQFace::Create

3.4.6.1.27 ofiq/face_parsing/bisenet_400.onnx

The solver size is 50 MB.

References

OFIQFace::Create

3.4.6.1.28 ofiq/head_pose_estimation/mb1_120x120.onnx

The solver size is 12 MB.

References

OFIQFace::Create

3.4.6.1.29 ofiq/no_compression_artifacts/ssim_248_model.onnx

The solver size is 22 MB.

References

OFIQFace::Create

3.4.6.1.30 ofiq/sharpness/face_sharpness_rtree.xml

The solver size is 5 MB.

References

OFIQFace::Create

3.4.6.1.31 ofiq/unified_quality_score/magface_iresnet50_norm.onnx

The solver size is 166 MB.

References

OFIQFace::Create

3.4.6.1.32 iris-verification-accurate-p1.solver

The solver size is 83 MB.

References

IrisExtractorConfig::Type::ACCURATE, IrisExtractorConfig::Type::FAST

3.4.6.1.33 iris-verification-accurate-p2.solver

The solver size is 91 MB.

References

IrisExtractorConfig::Type::ACCURATE

3.4.6.1.34 iris-detector.solver

The solver size is 9 MB.

References

IrisDetector::Config::Type::ACCURATE

3.4.6.1.35 iris-detector-fast.solver

The solver size is 627 kB.

References

IrisDetector::Config::Type::FAST

3.5 File

The library defines BinaryFileAccess interface to access files for reading and writing.

Classes

- class BinaryFileAccess

  BinaryFileAccess provides abstraction on underlying file system.
- class FilesystemAccess

  Standard filesystem implementation.
- class GlobalFileAccess

  Holds instance of BinaryFileAccess.
- class FileException

  Exception for file processing.

3.5.1 Detailed Description

The library defines BinaryFileAccess interface to access files for reading and writing. Usually default implementation (using filesystem) is sufficient. In case e.g. Android we need special access implementation due to Android restriction. The default GlobalFileAccess is filesystem implementation that is sufficient on most operating systems. The BinaryFileAccess can be implemented also in Java / CSharp bindings.

The library provide convenient static methods BinaryFile::ReadAll() and BinaryFile::WriteAll() that uses GlobalFileAccess to read, write files.

c++

```
class UserFileAccess : public BinaryFileAccess
{
    std::vector<uint8_t> ReadAll(const std::string& filePath) final
    {
        // do something useful, special
        return FilesystemAccess::Create()->ReadAll(filePath);
    }

    void WriteAll(const std::string& filePath,
                  const std::vector<uint8_t>& content) final
    {
        // do something useful, special
        FilesystemAccess::Create()->WriteAll(filePath, content);
    }
};

static void UseUserManagedFilesystem()
{
    GlobalFileAccess::Set(std::make_shared<UserFileAccess>());
    auto content = BinaryFile::ReadAll("assets/color.png");
    BinaryFile::WriteAll("output/color_copied_via_file_access.png", content);
}
```

java

```
class UserFileAccess extends BinaryFileAccess {
    public Bytes ReadAll(String filePath) {
        try {
            if (!"The Android Project".equals(
                        System.getProperty("java.specification.vendor"))) {
                return new Bytes(Files.readAllBytes(Paths.get(filePath)));
            } else {
                // Use Android approach e.g.
                // IOUtils.toByteArray(getAssets().open(filePath));
                return new Bytes();
            }
        } catch (IOException e) {
            throw new FileException(e.getMessage());
        }
    }

    public void WriteAll(String filePath, Bytes content) {
        try {
            if (!"The Android Project".equals(
                        System.getProperty("java.specification.vendor"))) {
                Path file = Paths.get(filePath);
                Files.write(file, content.toByteArray());
            } else {
                // Use Android approach
            }
        } catch (IOException e) {
            throw new FileException(e.getMessage());
        }
    }
}

public void UseUserManagedFilesystem() {
    BinaryFileAccess userFileAccess = new UserFileAccess();
    GlobalFileAccess.Set(userFileAccess);

    Bytes content = BinaryFile.ReadAll("assets/color.png");
    BinaryFile.WriteAll("output/color_copied_via_file_access.png", content);
}
```

csharp

```
class UserFileAccess : BinaryFileAccess
{
    public override Bytes ReadAll(string filePath)
```

```
    {
        // E.g. use csharp approach to read files
        return new Bytes(File.ReadAllBytes(filePath));
    }

    public override void WriteAll(string filePath, Bytes content)
    {
        // E.g. use csharp approach to write files
        File.WriteAllBytes(filePath, content.ToArray());
    }
};

public void UseUserManagedFilesystem()
{
    BinaryFileAccess userFileAccess = new UserFileAccess();
    GlobalFileAccess.Set(userFileAccess);

    Bytes content = BinaryFile.ReadAll("assets/color.png");
    BinaryFile.WriteAll("output/color_copied_via_file_access.png", content);
}
```

3.6 Image

Contains image operations.

Classes

- class Image

  Image representation.
- class Image::ImageResampler

  Image resampler for DPI-change operations, including upsampling and downsampling.
- class Image::BicubicImageResampler

  Image resampler for DPI-change operations, including upsampling and downsampling using bicubic interpolation resampling algorithm.
- class Image::AreaImageResampler

  Image resampler for DPI-change operations, including upsampling and downsampling using area interpolation resampling algorithm.
- class Image::NISTImageResampler

  Image resampler for DPI-change operations, including upsampling and downsampling using NIST-compliant resampling algorithm provided by `NFIR library`.
- class ImageDpi

  Image DPI.
- class RawImage

  Contains plain image data.
- class ImageDimensions

  It holds width and height of Image.
- class Point

  It holds X and Y coordinates of point.
- class Rectangle

  It holds rectangle vertexes, width, height.
- class Line

  It holds line points.
- class Circle

  It holds circle dimensions.
- class Polygon

  It holds polygon points.
- class RGB

It holds RGB colors.

- class RGBA

  It holds RGB color with transparency.
- class Colors

  It holds color literals.
- class ImageEncoderVisitor

  The class is used to visit concrete encoder to get concrete data.
- class ImageEncoder

  Interface for image formatting converting to ImageType.
- class SupportedImageEncoder

  Encodes raw images into specified supported format, It is used as based class for all supported formats.
- class BmpImageEncoder

  Converts image type ImageType::BMP.
- class PngImageEncoder

  Converts image type ImageType::PNG.
- class JpgImageEncoder

  Converts image type ImageType::JPG.
- class TiffImageEncoder

  Converts image type ImageType::TIFF.
- class Jp2ImageEncoder

  Converts image type ImageType::JPG2000.
- class WebpImageEncoder

  Converts image type ImageType::WEBP.
- class WsqImageEncoder

  Converts image type ImageType::WSQ.
- class IRawImageEncoder

  Saves RawImage image to memory.
- class ImageSaver

  Interface for image saving.
- class ImageFileSaver

  Saves image to file.
- class ImageException

  Exception for image processing.
- class GlobalImageExecutor

  Holds instance of ImageExecutor.
- class ImageMaskConfiguration

  Configuration parameters for image masking operations, including mask type, and color.
- class ImageMasker

  Interface for image masking operations.
- class ImageMask

  Defines the interface for applying mask to the image with a masker.
- class CircleImageMask

  Represents an image mask in the shape of a circle.
- class PolygonImageMask

  Represents an image mask defined by a polygonal shape.
- class ImageShapeAttributes

  It holds shape attributes such as color and line width.
- class Shapes

  It holds shape attributes literals.
- class ImageShapeDrawer

  It is used by ImageShape to draw its shape.

- class ImageShape

  Any type of shape that wants to be drawn.
- class RectangleShape

  It holds rectangle and draws it.
- class RectanglesShape

  It holds rectangles and draws it as one shape.
- class CircleShape

  It holds circle and draws it.
- class LineShape

  It holds line and draws it.
- class ImageType

  It contains all supported image types.

Typedefs

- using **ImageDimension** = unsigned int

  Image dimension.
- using **ImageHeight** = ImageDimension

  Image Height.
- using **ImageWidth** = ImageDimension

  Image Width.
- using **X** = int

  X coordinate.
- using **Y** = int

  Y coordinate.
- using **Angle** = float

  Angle in degrees.
- using **Radius** = unsigned int

  Radius.

Enumerations

- enum class ImageMaskType : uint8_t { OUTSIDE = 0 , INSIDE = 1 }

  Defines the polygon mask application area in image processing.
- enum class ImageBlurType : uint8_t { NO_BORDER_BLURRING = 0 , ISO_19794_BORDER_BLURRING = 1 }

  Defines the type of border blurring applied to an image.

3.6.1 Detailed Description

3.6.2 Overview

Contains image operations. Supported formats are

- BMP - Raster graphics image file format developed by Microsoft. [18]
- PNG - PNG supports palette-based images (with palettes of 24-bit RGB or 32-bit RGBA colors), grayscale images (with or without an alpha channel for transparency), and full-color non-palette-based RGB or RGBA images. [14]

- WSQ - WSQ (Wavelet Scalar Quantization), also known as FBI Print Format, is a raster image file format used for storing images of prints. It uses lossy wavelet compression, and supports only grayscale images.
- IRAW - The format is IRAW (4 bytes), width (4 bytes, big endian unsigned int), height (4 bytes, big endian unsigned int), dpi (4 bytes, big endian unsigned int), type (4 bytes, big endian unsigned int), data.
- JPG - JPEG is a commonly used method of lossy compression for digital images, particularly for those images produced by digital photography. The degree of compression can be adjusted, allowing a selectable tradeoff between storage size and image quality. [13]
- TIFF - TIF is an image file format for storing raster graphics images, popular among graphic artists, the publishing industry, and photographers. [15]
- JPG2000 - JPEG 2000 (JP2) is an image compression standard and coding system. It was developed from 1997 to 2000 by a Joint Photographic Experts Group committee with the intention of superseding their original JPEG standard (created in 1992) [12]
- WEBP - It is intended as a replacement for JPEG, PNG, and GIF file formats. It supports both lossy and lossless compression as well as animation and alpha transparency. [16]

The rest library functionality uses Image class as abstraction of supported image formats. The library extension points are

- ImageShape - implement to draw any user shape via ImageShapeDrawer
- ImageMask - implement to mask image with your own mask
- ImageEncoder - implement to write/convert Image to user format

3.6.2.1 Reading image

Function Image::Decode is used to create Image object. It automatically detects image type. For supported types see ImageType. On error the function will throw exception, e.g. for not supported ImageType.

c++

```
auto png = Image::Decode(BinaryFile::ReadAll("assets/color.png"));
auto bmp = Image::Decode(BinaryFile::ReadAll("assets/color.bmp"));
auto wsq = Image::Decode(BinaryFile::ReadAll("assets/color.wsq"));
auto raw = Image::Decode(BinaryFile::ReadAll("assets/color.iraw"));
auto jpg = Image::Decode(BinaryFile::ReadAll("assets/color.jpg"));
auto jp2 = Image::Decode(BinaryFile::ReadAll("assets/color.jp2"));
auto webp = Image::Decode(BinaryFile::ReadAll("assets/color.webp"));
auto tiff = Image::Decode(BinaryFile::ReadAll("assets/color.tiff"));
```

java

```
Image png = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
Image wsq = Image.Decode(BinaryFile.ReadAll("assets/color.wsq"));
Image bmp = Image.Decode(BinaryFile.ReadAll("assets/color.bmp"));
Image raw = Image.Decode(BinaryFile.ReadAll("assets/color.iraw"));
Image jpg = Image.Decode(BinaryFile.ReadAll("assets/color.jpg"));
Image jp2 = Image.Decode(BinaryFile.ReadAll("assets/color.jp2"));
Image tiff = Image.Decode(BinaryFile.ReadAll("assets/color.tiff"));
Image webp = Image.Decode(BinaryFile.ReadAll("assets/color.webp"));
```

csharp

```
Image png = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
Image wsq = Image.Decode(BinaryFile.ReadAll("assets/color.wsq"));
Image bmp = Image.Decode(BinaryFile.ReadAll("assets/color.bmp"));
Image raw = Image.Decode(BinaryFile.ReadAll("assets/color.iraw"));
Image jpg = Image.Decode(BinaryFile.ReadAll("assets/color.jpg"));
Image jp2 = Image.Decode(BinaryFile.ReadAll("assets/color.jp2"));
Image tiff = Image.Decode(BinaryFile.ReadAll("assets/color.tiff"));
Image webp = Image.Decode(BinaryFile.ReadAll("assets/color.webp"));
```

Function Image::DecodeRawImage is used to create Image object from provided grayscale/BGR/BGRA data.

c++

```
const RawImage grayscale = {
    3, 2, ImageDpi{ 500 }, RawImage::GRAYSCALE, { 0, 30, 40, 255, 15, 128 }
};
auto png = Image::DecodeRawImage(grayscale);
```

java

```
RawImage grayscale = new RawImage(3, 2, new ImageDpi(500),
        RawImage.Type.GRAYSCALE,
        new Bytes(new byte[] { 0, 30, 40, -1, 15, -128 }));
Image img = Image.DecodeRawImage(grayscale);
```

csharp

```
RawImage grayscale =
    new RawImage(3, 2, new ImageDpi(500), RawImage.Type.GRAYSCALE,
                 new Bytes(new byte[] { 0, 30, 40, 255, 15, 128 }));
Image img = Image.DecodeRawImage(grayscale);
```

3.6.2.2 Writing/Converting image

Function Image::EncodeIn is used to convert image via ImageEncoder.

c++

```
        auto png = Image::Decode(BinaryFile::ReadAll("assets/color.png"));
        // Use encoder of original format
        std::ignore = png->EncodeIn(*png->OriginalEncoder());
        // convert to other format and save by BinaryFile
        BmpImageEncoder bmpEncoder;
        auto encoded = png->EncodeIn(bmpEncoder);
        BinaryFile::WriteAll("output/color_converted.bmp", encoded);
        // convert and save in ImageFileSaver
        ImageFileSaver wsqFileSaver("output/color_converted.wsq");
        png->SaveIn(wsqFileSaver);
        // convert and save in file, with convenient function
        png->SaveAs("output/color_converted.iraw");
        png->SaveAs("output/color_converted.jpg");
#ifndef __ANDROID__
        // Encoding to jp2 is not working on Android,
        // due to disabled support of filesystem in OpenCV
        png->SaveAs("output/color_converted.jp2");
#endif
        png->SaveAs("output/color_converted.tiff");
        png->SaveAs("output/color_converted.webp");
```

java

```
Image png = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
// Use encoder of original format
png.EncodeIn(png.OriginalEncoder());
// convert to other format and use BinaryFile to save
BmpImageEncoder bmpEncoder = new BmpImageEncoder();
Bytes encoded = png.EncodeIn(bmpEncoder);
BinaryFile.WriteAll("output/color_converted.bmp", encoded);
// Use ImageFileSaver to convert and write
ImageFileSaver wsqSaver = new ImageFileSaver("output/color_converted.wsq");
png.SaveIn(wsqSaver);
// convert and save in file, with convenient function
png.SaveAs("output/color_converted.iraw");
png.SaveAs("output/color_converted.jpg");
if (!"The Android Project".equals(
            System.getProperty("java.specification.vendor"))) {
    // Encoding to jp2 is not working on Android,
    // due to disabled support of filesystem in OpenCV
    png.SaveAs("output/color_converted.jp2");
}
png.SaveAs("output/color_converted.tiff");
png.SaveAs("output/color_converted.webp");
```

csharp

```
Image png = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
// Use encoder of original format
png.EncodeIn(png.OriginalEncoder());
// convert to other formats and use BinaryFile to write
BmpImageEncoder bmpEncoder = new BmpImageEncoder();
Bytes encoded = png.EncodeIn(bmpEncoder);
```

```
BinaryFile.WriteAll("output/color_converted.bmp", encoded);
// Use ImageFileSaver to convert and write
ImageFileSaver wsqSaver = new ImageFileSaver("output/color_converted.wsq");
png.SaveIn(wsqSaver);
// convert and save in file, with convenient function
png.SaveAs("output/color_converted.iraw");
png.SaveAs("output/color_converted.jpg");
png.SaveAs("output/color_converted.jp2");
png.SaveAs("output/color_converted.tiff");
png.SaveAs("output/color_converted.webp");
```

3.6.2.3 Resize

Functions Image::Resize and Image::ResizeToMaxDimension are used to resize image.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/color.png"));
auto resized = img->Resize(0.5F);
auto resizedToMaxDimension = img->ResizeToMaxDimension(img->Width() / 2);

resized->SaveAs("output/color_resized_half.png");
resizedToMaxDimension->SaveAs("output/color_resized_half_dimension.png");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
Image resized = img.Resize(0.5F);
Image resizedToMaxDimension = img.ResizeToMaxDimension(img.Width() / 2);

resized.SaveAs("output/color_resized_half.png");
resizedToMaxDimension.SaveAs("output/color_resized_half_dimension.png");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
Image resized = img.Resize(0.5F);
Image resizedToMaxDimension = img.ResizeToMaxDimension(img.Width() / 2);

resized.SaveAs("output/color_resized_half.png");
resizedToMaxDimension.SaveAs("output/color_resized_half_dimension.png");
```

Figure 26 Resize input

Figure 27 Resize output

3.6.2.4 Change DPI

Function Image::ChangeDpiTo is used to change DPI of image. The original image is resized with factor `target↩DPI/originalDPI`.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/color.png"), 1000);
```

```
auto changed = img->ChangeDpiTo(ImageDpi(500));

changed->SaveAs("output/color_dpi_500.png");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"), 1000);
Image changed = img.ChangeDpiTo(new ImageDpi(500));

changed.SaveAs("output/color_dpi_500.png");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"), 1000);
Image changed = img.ChangeDpiTo(new ImageDpi(500));

changed.SaveAs("output/color_dpi_500.png");
```

Figure 28 DPI change input

Figure 29 DPI change output

3.6.2.5 Draw shape

For each shape function Image::DrawShape creates new Image to keep current instance immutable. If more shapes need to be drawn in one image, then implement new class with base ImageShape e.g RectanglesShape. Then it is possible to draw into image via ImageShapeDrawer.

Figure 30 Draw input

Figure 31 Draw rectangle output

c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/color.png"));
const Rectangle rectangle{ { 37, 20 }, { 79, -11 }, { 109, 29 }, { 67, 60 } };
const RectangleShape shape(rectangle, Shapes::InnoBlue);

auto drawnImage = img->DrawShape(shape);

drawnImage->SaveAs("output/color_drawn.bmp");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
Rectangle rectangle = new Rectangle(new Point(37, 20), new Point(79, -11),
        new Point(109, 29), new Point(67, 60));
RectangleShape shape = new RectangleShape(rectangle, Shapes.InnoBlue);

Image drawnImage = img.DrawShape(shape);

drawnImage.SaveAs("output/color_drawn.bmp");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
Rectangle rectangle = new Rectangle(new Point(37, 20), new Point(79, -11),
                                    new Point(109, 29), new Point(67, 60));
RectangleShape shape = new RectangleShape(rectangle, Shapes.InnoBlue);

Image drawnImage = img.DrawShape(shape);

drawnImage.SaveAs("output/color_drawn.bmp");
```

Figure 32 Draw circle output

c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/color.png"));
const CircleShape shape(Circle{ { 30, 45 }, 10 }, Shapes::InnoBlue);

auto drawnImage = img->DrawShape(shape);

drawnImage->SaveAs("output/color_drawn_circle.bmp");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
CircleShape shape = new CircleShape(
        new Circle(new Point(30, 45), 10), Shapes.InnoBlue);

Image drawnImage = img.DrawShape(shape);

drawnImage.SaveAs("output/color_drawn_circle.bmp");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
CircleShape shape =
    new CircleShape(new Circle(new Point(30, 45), 10), Shapes.InnoBlue);

Image drawnImage = img.DrawShape(shape);

drawnImage.SaveAs("output/color_drawn_circle.bmp");
```

Figure 33 Draw custom shape output

c++

```
// Define custom shape
class CustomShape : public ImageShape
{
public:
    void DrawIn(ImageShapeDrawer& drawer) const final
    {
        // Draw custom shapes
        drawer.DrawLine({ { 0, 0 }, { 100, 100 } }, { { { 0, 255, 0 }, 50 }, 5 });
        drawer.DrawCircle({ { 0, 0 }, 30 }, { { { 255, 0, 0 }, 100 }, 5 });
    }
};

static void DrawCustomShapeInImage()
{
    auto img = Image::Decode(BinaryFile::ReadAll("assets/color.png"));
    const CustomShape customShape{};
    auto drawnImage = img->DrawShape(customShape);
    drawnImage->SaveAs("output/color_custom_drawn.png");
}
```

java

```
// Define custom shape
public class CustomShape extends ImageShape {
    @Override
    public void DrawIn(ImageShapeDrawer drawer) {
        // Draw custom shapes
        ImageShapeAttributes attr = new ImageShapeAttributes();
        attr.setRgba(new RGBA((short) 0, (short) 255, (short) 0, (short) 50));
        attr.setThickness((short) 5);
        drawer.DrawLine(new Line(new Point(0, 0), new Point(100, 100)), attr);
        attr.setRgba(new RGBA((short) 255, (short) 0, (short) 0, (short) 100));
        drawer.DrawCircle(new Circle(new Point(0, 0), 30), attr);
    }
};

public void DrawCustomShapeInImage() {
    Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
    CustomShape shape = new CustomShape();
    Image drawnImage = img.DrawShape(shape);
    drawnImage.SaveAs("output/color_custom_drawn.png");
}
```

csharp

```
// Define custom shape
public class CustomShape : ImageShape
{
    public override void DrawIn(ImageShapeDrawer drawer)
    {
        // Draw custom shapes
        ImageShapeAttributes attr = new ImageShapeAttributes();
        attr.rgba = new RGBA(0, 255, 0, 50);
        attr.thickness = 5;
        drawer.DrawLine(new Line(new Point(0, 0), new Point(100, 100)), attr);
        attr.rgba = new RGBA(255, 0, 0, 100);
        drawer.DrawCircle(new Circle(new Point(0, 0), 30), attr);
    }
};

public void DrawCustomShapeInImage()
{
    Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
    CustomShape shape = new CustomShape();
    Image drawnImage = img.DrawShape(shape);
    drawnImage.SaveAs("output/color_custom_drawn.png");
}
```

3.6.2.6 Mask shape

For each shape function Image::MaskImageBy creates new Image to keep current instance immutable. If more shapes need to be masked in one image, then use custom mask class extending ImageMask where you can add any combination of PolygonImageMask and CircleImageMask. The each mask shape has ImageMaskAttributes

- masked area

  - INSIDE - Mask is applied to the region inside the shape, preserving the area outside.
  - OUTSIDE - Mask is applied to the region outside the shape, preserving the area inside.
- blurring mask border after applying mask
  - NO_BORDER_BLURRING - No border blurring is applied.
  - ISO_19794_BORDER_BLURRING - Applies border blurring according to ISO/IEC 19794-6 standard.

C++

```
const auto img = Image::Decode(BinaryFile::ReadAll("assets/color.png"));
const Circle circle{ { 60, 60 }, 20 };

// Mask image by circle - inside
auto insideMask = CircleImageMask(
  circle,
  ImageMaskConfiguration{ ImageMaskType::INSIDE, Colors::InnoBlue },
  ImageBlurType::NO_BORDER_BLURRING);
auto maskedInside = img->MaskImageBy(insideMask);
maskedInside->SaveAs("output/masked_circle_inside.png");

// Mask image by circle - outside
auto outsideMask = CircleImageMask(
  circle,
  ImageMaskConfiguration{ ImageMaskType::OUTSIDE, Colors::InnoBlue },
  ImageBlurType::NO_BORDER_BLURRING);
auto maskedOutside = img->MaskImageBy(outsideMask);
maskedOutside->SaveAs("output/masked_circle_outside.png");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
Circle circle = new Circle(new Point(60, 60), 20);

// Mask image by circle - inside
CircleImageMask insideMask = new CircleImageMask(circle,
        new ImageMaskConfiguration(ImageMaskType.INSIDE, Colors.InnoBlue),
        ImageBlurType.NO_BORDER_BLURRING);
Image maskedInside = img.MaskImageBy(insideMask);
maskedInside.SaveAs("output/masked_circle_inside.png");

// Mask image by circle - outside
CircleImageMask outsideMask = new CircleImageMask(circle,
        new ImageMaskConfiguration(ImageMaskType.OUTSIDE, Colors.InnoBlue),
        ImageBlurType.NO_BORDER_BLURRING);
Image maskedOutside = img.MaskImageBy(outsideMask);
maskedOutside.SaveAs("output/masked_circle_outside.png");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
var circle = new Circle(new Point(60, 60), 20);

// Mask image by circle - inside
var insideMask = new CircleImageMask(
    circle, new ImageMaskConfiguration(ImageMaskType.INSIDE, Colors.InnoBlue),
    ImageBlurType.NO_BORDER_BLURRING);
var maskedInside = img.MaskImageBy(insideMask);
maskedInside.SaveAs("output/masked_circle_inside.png");

// Mask image by circle - outside
var outsideMask = new CircleImageMask(
    circle, new ImageMaskConfiguration(ImageMaskType.OUTSIDE, Colors.InnoBlue),
    ImageBlurType.NO_BORDER_BLURRING);
var maskedOutside = img.MaskImageBy(outsideMask);
maskedOutside.SaveAs("output/masked_circle_outside.png");
```

Figure 34 Circle masked inside

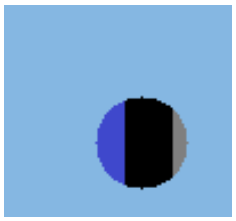


Figure 35 Circle masked outside

C++

```
const auto img = Image::Decode(BinaryFile::ReadAll("assets/color.png"));
const Polygon polygon{ { { 50, 20 }, { 80, 80 }, { 20, 80 } } };

// Mask image by polygon - inside
auto insideMask = PolygonImageMask(
  polygon,
  ImageMaskConfiguration{ ImageMaskType::INSIDE, Colors::InnoBlue },
  ImageBlurType::ISO_19794_BORDER_BLURRING);
auto maskedInside = img->MaskImageBy(insideMask);
maskedInside->SaveAs("output/masked_polygon_inside.png");

// Mask image by polygon - outside
auto outsideMask = PolygonImageMask(
  polygon,
  ImageMaskConfiguration{ ImageMaskType::OUTSIDE, Colors::InnoBlue },
  ImageBlurType::ISO_19794_BORDER_BLURRING);
auto maskedOutside = img->MaskImageBy(outsideMask);
maskedOutside->SaveAs("output/masked_polygon_outside.png");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
Polygon polygon = new Polygon(new Points(new ArrayList<>(
        Arrays.asList(new Point(50, 20), new Point(80, 80), new Point(20, 80)))));

// Mask image by polygon - inside
PolygonImageMask insideMask = new PolygonImageMask(polygon,
        new ImageMaskConfiguration(ImageMaskType.INSIDE, Colors.InnoBlue),
        ImageBlurType.ISO_19794_BORDER_BLURRING);
Image maskedInside = img.MaskImageBy(insideMask);
maskedInside.SaveAs("output/masked_polygon_inside.png");

// Mask image by polygon - outside
PolygonImageMask outsideMask = new PolygonImageMask(polygon,
        new ImageMaskConfiguration(ImageMaskType.OUTSIDE, Colors.InnoBlue),
        ImageBlurType.ISO_19794_BORDER_BLURRING);
Image maskedOutside = img.MaskImageBy(outsideMask);
maskedOutside.SaveAs("output/masked_polygon_outside.png");
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
var polygon = new Polygon(new Points(
    new List<Point> { new Point(50, 20), new Point(80, 80), new Point(20, 80) }));

// Mask image by polygon - inside
var insideMask = new PolygonImageMask(
    polygon, new ImageMaskConfiguration(ImageMaskType.INSIDE, Colors.InnoBlue),
    ImageBlurType.ISO_19794_BORDER_BLURRING);
var maskedInside = img.MaskImageBy(insideMask);
maskedInside.SaveAs("output/masked_polygon_inside.png");

// Mask image by polygon - outside
var outsideMask = new PolygonImageMask(
    polygon, new ImageMaskConfiguration(ImageMaskType.OUTSIDE, Colors.InnoBlue),
    ImageBlurType.ISO_19794_BORDER_BLURRING);
var maskedOutside = img.MaskImageBy(outsideMask);
maskedOutside.SaveAs("output/masked_polygon_outside.png");
```

Figure 36 Polygon masked and blurred inside

Figure 37 Polygon masked and blurred outside

c++

```
// Define custom mask
class CustomMask : public ImageMask
{
public:
    void MaskWith(ImageMasker& masker) const override
    {
        // Create shapes for the custom mask
        const Polygon polygon{ { { 50, 20 }, { 80, 80 }, { 20, 80 } } };
        const Circle circle{ { 60, 60 }, 20 };

        // Define custom mask behaviour
        masker.MaskCircle(
          circle, ImageMaskConfiguration{ ImageMaskType::INSIDE, Colors::White });
        masker.MaskPolygon(
          polygon,
          ImageMaskConfiguration{ ImageMaskType::OUTSIDE, Colors::InnoBlue });
        masker.AddToBlur(polygon, ImageMaskType::OUTSIDE);
        masker.ApplyBlur(ImageBlurType::ISO_19794_BORDER_BLURRING);
    }
};

static void MaskImageWithCombination()
{
    const auto img = Image::Decode(BinaryFile::ReadAll("assets/color.png"));
    const CustomMask mask;
    auto maskedImage = img->MaskImageBy(mask);
    maskedImage->SaveAs("output/masked_combination.png");
}
```

java

```
// Define custom mask
public class CustomMask extends ImageMask {
    @Override
    public void MaskWith(ImageMasker masker) {
        // Create shapes for the custom mask
        Polygon polygon = new Polygon(new Points(Arrays.asList(
                new Point(50, 20), new Point(80, 80), new Point(20, 80))));
        Circle circle = new Circle(new Point(60, 60), 20);

        // Define custom mask behaviour
        masker.MaskCircle(circle,
                new ImageMaskConfiguration(ImageMaskType.INSIDE, Colors.White));
        masker.MaskPolygon(polygon,
                new ImageMaskConfiguration(ImageMaskType.OUTSIDE, Colors.InnoBlue));
        masker.AddToBlur(polygon, ImageMaskType.OUTSIDE);
        masker.ApplyBlur(ImageBlurType.ISO_19794_BORDER_BLURRING);
    }
};

public void MaskImageWithCombination() {
    Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));

    CustomMask mask = new CustomMask();
    Image maskedImage = img.MaskImageBy(mask);
    maskedImage.SaveAs("output/masked_combination.png");
}
```

csharp

```
// Define custom mask
public class CustomMask : ImageMask
{
    public override void MaskWith(ImageMasker masker)
    {
        // Create shapes for the custom mask
```

```
        var polygon = new Polygon(
            new Points(new List<Point> { new Point(50, 20), new Point(80, 80),
                                         new Point(20, 80) }));
        var circle = new Circle(new Point(60, 60), 20);

        // Define custom mask behaviour
        masker.MaskCircle(
            circle, new ImageMaskConfiguration(ImageMaskType.INSIDE, Colors.White));
        masker.MaskPolygon(polygon, new ImageMaskConfiguration(ImageMaskType.OUTSIDE,
                                                                Colors.InnoBlue));
        masker.AddToBlur(polygon, ImageMaskType.OUTSIDE);
        masker.ApplyBlur(ImageBlurType.ISO_19794_BORDER_BLURRING);
    }
}

public void MaskImageWithCombination()
{
    var img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));

    var mask = new CustomMask();
    var maskedImage = img.MaskImageBy(mask);
    maskedImage.SaveAs("output/masked_combination.png");
}
```

Figure 38 Masked combination

3.6.2.7 Crop aligned rectangle

Use function Image::CropAlignedRectangle to crop and align rectangle. It Crop rectangle and align given top left rectangle point to coordinates [0,0] in new image.

Figure 39 Crop aligned rectangle input

Figure 40 Crop aligned rectangle output

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/color.png"));
// The points MUST be ordered as top left, top right, bottom right, bottom left
// the rectangle will be aligned to transform top left to coordinate [0,0] in new
// image
const Rectangle rectangle{ { 79, -11 }, { 109, 29 }, { 67, 60 }, { 37, 20 } };

auto copiedImage = img->CropAlignedRectangle(rectangle);

copiedImage->SaveAs("output/color_copied_align.bmp");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
// The points MUST be ordered as top left, top right, bottom right, bottom left
// the rectangle will be aligned to transform top left to coordinate [0,0] in
// new image
Rectangle rectangle = new Rectangle(new Point(79, -11), new Point(109, 29),
        new Point(67, 60), new Point(37, 20));

Image copiedImage = img.CropAlignedRectangle(rectangle);

copiedImage.SaveAs("output/color_copied_align.bmp");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
// The points MUST be ordered as top left, top right, bottom right, bottom left
// the rectangle will be aligned to transform top left to coordinate [0,0] in new
// image
Rectangle rectangle = new Rectangle(new Point(79, -11), new Point(109, 29),
                                    new Point(67, 60), new Point(37, 20));

Image copiedImage = img.CropAlignedRectangle(rectangle);

copiedImage.SaveAs("output/color_copied_align.bmp");
```

3.6.2.8 Crop rectangle

Use function Image::CropRectangle to crop rectangle. It Crops a rectangle from the image, using a white background for any extra space.

Figure 41 Crop rectangle input

Figure 42 Crop rectangle output

c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/color.png"));
const Rectangle rectangle{ { 37, 20 }, { 79, -11 }, { 109, 29 }, { 67, 60 } };

auto copiedImage = img->CropRectangle(rectangle, Colors::InnoBlue);

copiedImage->SaveAs("output/color_copied.bmp");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
Rectangle rectangle = new Rectangle(new Point(37, 20), new Point(79, -11),
        new Point(109, 29), new Point(67, 60));

Image copiedImage = img.CropRectangle(rectangle, Colors.InnoBlue);

copiedImage.SaveAs("output/color_copied.bmp");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
Rectangle rectangle = new Rectangle(new Point(37, 20), new Point(79, -11),
                                    new Point(109, 29), new Point(67, 60));

Image copiedImage = img.CropRectangle(rectangle, Colors.InnoBlue);

copiedImage.SaveAs("output/color_copied.bmp");
```

3.6.2.9 Crop circle

For given circle function Image::CropCircle cut circle from image.

Figure 43 Crop circle input

Figure 44 Crop circle output

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/color.png"));
auto copiedImage = img->CropCircle(Circle{ { 30, 45 }, 10 });
copiedImage->SaveAs("output/color_copied_circle.bmp");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
Image copiedImage = img.CropCircle(new Circle(new Point(30, 45), 10));
copiedImage.SaveAs("output/color_copied_circle.bmp");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/color.png"));
Image copiedImage = img.CropCircle(new Circle(new Point(30, 45), 10));
copiedImage.SaveAs("output/color_copied_circle.bmp");
```

3.6.2.10 Intersection

Library provides functions Image::RectangleIntersection and Image::CircleIntersection that calculates intersection type with Image.

- **Partial** - The Object and Image has intersection.

Figure 45 Partial intersection

- **Inside** - Object is fully in Image.

Figure 46 Inside intersection

- **Outside** - Image is fully in Object.

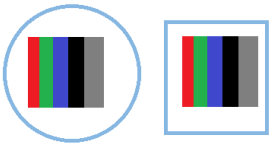


Figure 47 Outside intersection

- **Same** - Image has same dimensions and vertices coordinates as Object.

Figure 48 Same intersection

- **No** - Image and Object has no intersection.

Figure 49 No intersection

3.6.3 Enumeration Type Documentation

3.6.3.1 ImageBlurType

```
enum class ImageBlurType : uint8_t [strong]
```

Enumerator

NO_BORDER_BLURRING	No border blurring is applied.

ISO_19794_BORDER_BLURRING	Applies border blurring according to ISO/IEC 19794-6 standard. The borders of mask region is smoothed by low-pass filtering. Each image pixel, whose centred 7 x 7 neighbourhood contains at least one mask pixel, is replaced by a weighted sum of a 7 x 7 binomial kernel. The coefficients of this kernel are defined by the outer product, [7] `K = 1/(64x64) UUT where: U = [ 1 6 15 20 15 6 1 ]T`

3.6.3.2 ImageMaskType

```
enum class ImageMaskType : uint8_t [strong]
```

Enumerator

OUTSIDE	Mask is applied to the region outside the shape, preserving the area inside.
INSIDE	Mask is applied to the region inside the shape, preserving the area outside.

3.7 Manual initialisation

By default, the Enrollment SDK initializes itself automatically when the shared library is loaded.

Classes

- struct InitializerConfig

  SDK initialisation configuration for manual init.
- class InitializerConfigBuilder

  Fluent builder for InitializerConfig.

Functions

- static std::string FindLicenseFile ()

  Finds the installed license file using the same search order as auto-init.
- static std::string DefaultModelsDirectory ()

  Returns the default IFace models directory.
- static void Initialize (const InitializerConfig &config)

  Initialize SDK from InitializerConfig (license, models path, and thread counts).
- static void Initialize (std::vector< uint8_t > license, std::string modelsPath)

  Initialize SDK.

3.7.1 Detailed Description

If you need to control OpenCV or ONNX Runtime thread counts before the SDK loads its models, suppress auto-init and explicitly initialize the SDK with a custom configuration. Automatic initialization may not be suitable when some prerequisites are unavailable at startup or when full control over the SDK configuration is required. In such cases, initialize the SDK explicitly using either `Initialize(std::vector<uint8_t> license, std::string modelsPath)` or `Initialize(const InitializerConfig& config)`. The initialization functions can only be called once during the lifetime of the process. The Enrollment SDK also provides the helper functions `DefaultModelsDirectory()` and `FindLicenseFile()`, which can be used to populate the default paths in the initialization configuration.

For Java-specific initialization requirements (loading native libraries before any SDK call), see the initialization section of the Java binding documentation.

C++

```
const auto cfg = inno::InitializerConfigBuilder()
                   .SetLicensePath(inno::FindLicenseFile())
                   .SetModelsPath(inno::DefaultModelsDirectory())
                   .SetFaceIrisOpencvThreads(4)
                   .SetFaceIrisOrtIntraThreads(4)
                   .SetFaceIrisOrtInterThreads(1)
                   .Build();

inno::Initialize(cfg);

// SDK is now fully initialised with the requested thread counts.
// Proceed with normal face / iris / fingerprint operations.
```

java

```
EnrollmentInitializer.LoadNativeLibrary(
        ()
                -> new InitializerConfigBuilder()
                           .SetLicensePath(enrollment.FindLicenseFile())
                           .SetModelsPath(
                                   enrollment.DefaultModelsDirectory())
                           .SetFaceIrisOpencvThreads(4)
                           .SetFaceIrisOrtIntraThreads(4)
                           .SetFaceIrisOrtInterThreads(1)
                           .Build());

// SDK is now fully initialized with the requested thread counts.
// Proceed with normal face / iris / fingerprint operations.
```

csharp

```
using (var builder = new InitializerConfigBuilder())
{
    var cfg = builder
        .SetLicensePath(enrollment.FindLicenseFile())
        .SetModelsPath(enrollment.DefaultModelsDirectory())
        .SetFaceIrisOpencvThreads(4)
        .SetFaceIrisOrtIntraThreads(4)
        .SetFaceIrisOrtInterThreads(1)
        .Build();

    enrollment.Initialize(cfg);
}

// SDK is now fully initialised with the requested thread counts.
// Proceed with normal face / iris / fingerprint operations.
```

3.7.2 Function Documentation

3.7.2.1 DefaultModelsDirectory()

`std::string DefaultModelsDirectory () [inline], [static]`

The path is `iface/models` next to the loaded Enrollment SDK library (the directory containing `libinnoenrollment.so` or `innoenrollment.dll`). The directory is not guaranteed to exist on disk; this function does not verify the path.

Returns

path to the default models directory For package layout and auto-initialization, see Initialization and Solvers.

3.7.2.2 FindLicenseFile()

```
std::string FindLicenseFile () [inline], [static]
```

Search order:

1. Current working directory (iengine.lic)
2. User home directory (~/.innovatrics/iengine.lic on Linux/macOS, LOCALAPPDATA%\Innovatrics\iengine.lic on Windows)
3. System directory (/etc/innovatrics/iengine.lic on Linux/macOS, C:\ProgramData\Innovatrics\iengine.lic on Windows)

Returns

path to the license file, or empty string if not found

Note

The result is cached after the first call. Ensure the license file is present before calling this function for the first time.

3.7.2.3 Initialize() [1/2]

```
void Initialize (
            const InitializerConfig & config) [inline], [static]
```

Use this overload when you need to control thread counts before the SDK loads its models (e.g. on multi-core Android devices to reduce iris extraction latency). Set `INNO_ENROLLMENT_SDK_MANUAL_INITIALIZATION=1` before loading the library to suppress auto-init and preserve the parameter window for this call. Accepted values are case-insensitive: `1`, `true`, `on`, `manual`, `enable`, `enabled`.

Parameters

config	initialisation configuration built with InitializerConfigBuilder

Exceptions

EnrollmentInvalidArgumentException	when `config` has empty or missing license data
EnrollmentException	when Enrollment SDK is already initialized

3.7.2.4 Initialize() [2/2]

```
void Initialize (
            std::vector< uint8_t > license,
            std::string modelsPath)  [inline], [static]
```

SDK is initialized automatically when library is called. It uses models delivered side by side with library. The licence is used from known paths. When SDK is used on system like Android, that do not allow to access files, user can use this function to initialize library.

Parameters

license	license content
modelsPath	path to provided solvers

Exceptions

EnrollmentException	when Enrollment SDK is already initialized

3.8 Iris

The library can be used to detect iris from iris captures.

Classes

- class IrisExtractorConfig

  It contains extraction configuration.
- class IrisTemplate

  IrisTemplate is Innovatrics proprietary template created by IrisExtractor.
- class ExtractedIris

  ExtractedIris is template created by IrisExtractor with Iris that belongs to template.
- class Iris

  It holds all functions and data related to iris manipulation.
- class IrisAttributes

  Iris attributes Holds the iris attributes.
- class IrisCaptureDeviceTypeID

  The capture device type identifier shall identify the product type that created the BDIR.
- class IrisCaptureDeviceVendorID

  Capture device vendor identifier The capture device vendor identifier shall identify the biometric organisation that owns the product that created the BDIR.
- class IrisCapture

  Iris capture contains one iris.
- class IrisDetector

  IrisDetector implements detection by means of Innovatrics IFace.
- class GlobalIrisExecutor

  Holds instance of IrisExecutor.
- class IrisSegment

  It describes Iris segmented from original IrisCapture (one that contain this IrisSegment) with IrisDetector.

Typedefs

- using IrisSimilarityScore = unsigned int

  It is similarity score between iris Templates.

Enumerations

- enum class IrisCaptureDeviceTechnology : uint8_t { UNKNOWN = 0 , CMOS_CCD = 1 }

  It shall indicate the class of capture device technology used to acquire the captured biometric sample [7] .
- enum class IrisImageType : uint8_t { UNCROPPED = 1 , VGA = 2 , CROPPED = 3 , CROPPED_AND_MASKED = 7 }

  It shall indicate type of iris image [7] .
- enum class IrisPosition : uint8_t { RIGHT_EYE = 1 , LEFT_EYE = 2 }

  It is position of iris on face [7] .

Functions

- static constexpr int IrisToIsoCaptureDeviceTechnology (IrisCaptureDeviceTechnology printCaptureDevice↩ Technology)

  Function converts IrisCaptureDeviceTechnology to ISO capture technology.
- static constexpr IrisCaptureDeviceTechnology IsoToIrisCaptureDeviceTechnology (int isoCaptureDevice↩ Technology)

  Function converts ISO capture technology to IrisCaptureDeviceTechnology.
- static constexpr int IrisImageTypeToIso (IrisImageType type)

  Function converts IrisImageType to ISO iris image type.
- static constexpr IrisImageType IsoToIrisImageType (int type)

  Function converts ISO iris image type to IrisImageType.
- static constexpr int IrisPositionToIsoPosition (IrisPosition irisPosition)

  Function converts IrisPosition to ISO position.
- static constexpr IrisPosition IsoToIrisPosition (int irisPosition)

  Function converts ISO position to IrisPosition.

3.8.1 Detailed Description

3.8.2 Overview

The library can be used to detect iris from iris captures. Iris capture can contains entire eye. There MUST be only one iris in capture. Each capture MUST be passed to detector that recognize Iris.

3.8.2.1 Iris Detection

Prior to further processing the Iris in IrisCapture MUST be detected

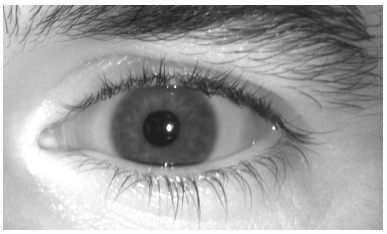


Figure 50 Eye Capture

The iris is detected with function IrisCapture::DetectWith.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/eye_L.png"));
const auto ic = IrisCapture::Create(std::move(img), IrisPosition::LEFT_EYE);
auto cfg = IrisDetector::Config::Default();
IrisDetector id(cfg);
auto segment = ic->DetectWith(id);

const std::string name = "output/iris_L.png";
segment->iris->image->SaveAs(name);
auto roundingSegments =
  ic->DrawSegment(*segment, Shapes::InnoBlue, Shapes::InnoBlue);
roundingSegments->SaveAs("output/iris_rounding_box.png");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));
IrisCapture capture = new IrisCapture(img, IrisPosition.LEFT_EYE);
IrisDetector.Config cfg = IrisDetector.Config.Default();
IrisDetector detector = new IrisDetector(cfg);
IrisSegment segment = capture.DetectWith(detector);
Iris iris = segment.getIris();
iris.getImage().SaveAs("output/iris_L.png");
Image roundigCircle = capture.DrawSegment(
        segment, Shapes.InnoBlue, Shapes.InnoBlue);
roundigCircle.SaveAs("output/iris_rounding_box.png");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));
IrisCapture capture = new IrisCapture(img, IrisPosition.LEFT_EYE);
IrisDetector.Config cfg = IrisDetector.Config.Default();
IrisDetector detector = new IrisDetector(cfg);
IrisSegment segment = capture.DetectWith(detector);
Iris iris = segment.iris;
iris.image.SaveAs("output/iris_L.png");
Image roundigCircle =
    capture.DrawSegment(segment, Shapes.InnoBlue, Shapes.InnoBlue);
roundigCircle.SaveAs("output/iris_rounding_box.png");
```

The output of example is detected iris

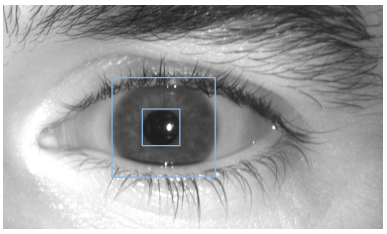


Figure 51 Detected Iris

3.8.2.2 Extraction

We provide Internal Innovatrics template for Iris. The main goal is to create template for further processing e.g. 1:1 Verification.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/eye_L.png"));
const auto ic = IrisCapture::Create(std::move(img), IrisPosition::LEFT_EYE);
auto detectorCfg = IrisDetector::Config();
detectorCfg.SetDetector(IrisDetector::Config::Type::ACCURATE);
IrisDetector id(detectorCfg);
// Detect iris
auto iris = ic->DetectWith(id)->iris;
auto cfg = IrisExtractorConfig();
cfg.SetExtractor(IrisExtractorConfig::Type::ACCURATE);
auto irisTemplate = iris->Extract(cfg);
BinaryFile::WriteAll("output/iris.ics", irisTemplate->data);

std::cout « "   Template Version: " « irisTemplate->GetVersion() « std::endl;
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));

IrisCapture ic = new IrisCapture(img, IrisPosition.LEFT_EYE);
IrisDetector.Config detectorCfg = new IrisDetector.Config();
detectorCfg.SetDetector(IrisDetector.Config.Type.ACCURATE);
IrisDetector id = new IrisDetector(detectorCfg);
// Detect iris
Iris iris = ic.DetectWith(id).getIris();
IrisExtractorConfig cfg = new IrisExtractorConfig();
cfg.SetExtractor(IrisExtractorConfig.Type.ACCURATE);
ExtractedIris irisTemplate = iris.Extract(cfg);
BinaryFile.WriteAll("output/iris.ics", irisTemplate.getData());

System.out.println("   Template Version: " + irisTemplate.GetVersion());
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));

var ic = new IrisCapture(img, IrisPosition.LEFT_EYE);
var detectorCfg = new IrisDetector.Config();
detectorCfg.SetDetector(IrisDetector.Config.Type.ACCURATE);
var id = new IrisDetector(detectorCfg);
// Detect iris
var iris = ic.DetectWith(id).iris;
var cfg = new IrisExtractorConfig();
cfg.SetExtractor(IrisExtractorConfig.Type.ACCURATE);
var irisTemplate = iris.Extract(cfg);
BinaryFile.WriteAll("output/iris.ics", irisTemplate.data);

System.Console.WriteLine("   Template Version: " + irisTemplate.GetVersion());
```

3.8.2.3 1 to 1 Verification

We provide verification of iris templates. The main goal is to calculate Similarity Score for given probe and gallery templates. The examples uses same image for probe and gallery

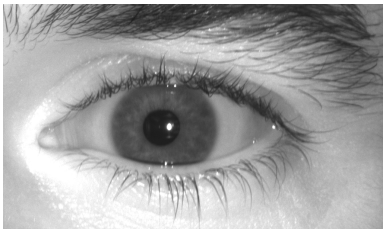


Figure 52 Iris

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/eye_L.png"));
const auto ic = IrisCapture::Create(std::move(img), IrisPosition::LEFT_EYE);
IrisDetector id;
auto iris = ic->DetectWith(id)->iris;
auto probe = iris->Extract();
auto gallery = iris->Extract();
auto score = probe->SimilarWith(*gallery);
std::cout « "   Match Score: " « score « std::endl;
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));
IrisCapture ic = new IrisCapture(img, IrisPosition.LEFT_EYE);
IrisDetector id = new IrisDetector();
Iris iris = ic.DetectWith(id).getIris();
ExtractedIris probe = iris.Extract();
ExtractedIris gallery = iris.Extract();
long score = probe.SimilarWith(gallery);
System.out.println("   Match Score: " + score);
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));
IrisCapture ic = new IrisCapture(img, IrisPosition.LEFT_EYE);
IrisDetector id = new IrisDetector();
Iris iris = ic.DetectWith(id).iris;
ExtractedIris probe = iris.Extract();
ExtractedIris gallery = iris.Extract();
uint score = probe.SimilarWith(gallery);
System.Console.WriteLine("   Match Score: " + score);
```

3.8.2.4 Quality

Template quality is calculated from a iris template. The quality is an integer number between 1 (blank image) and 100 (maximum quality).

c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/eye_L.png"));
const auto ic = IrisCapture::Create(std::move(img), IrisPosition::LEFT_EYE);
IrisDetector id;
auto iris = ic->DetectWith(id)->iris;
auto irisTemplate = iris->Extract();

std::cout « "   Template Quality: " « +irisTemplate->CalculateQuality()
          « std::endl;
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));
IrisCapture ic = new IrisCapture(img, IrisPosition.LEFT_EYE);
IrisDetector id = new IrisDetector();
Iris iris = ic.DetectWith(id).getIris();
ExtractedIris irisTemplate = iris.Extract();

System.out.println("   Template Quality: " + irisTemplate.CalculateQuality());
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));
IrisCapture ic = new IrisCapture(img, IrisPosition.LEFT_EYE);
IrisDetector id = new IrisDetector();
Iris iris = ic.DetectWith(id).iris;
ExtractedIris irisTemplate = iris.Extract();

System.Console.WriteLine("   Template Quality: " +
                         irisTemplate.CalculateQuality());
```

3.8.2.5 ISO Conversion

3.8.2.5.1 ISO Image format

It is possible to convert Iris to ISO image compliant with ISO 19794-6:2011 and back from image to IrisCapture.

C++

```
auto image = Image::Decode(BinaryFile::ReadAll("assets/eye_L.png"));

IrisAttributes ia;
ia.SetCaptureDeviceTechnology(IrisCaptureDeviceTechnology::CMOS_CCD);
ia.SetCaptureDeviceVendorID(IrisCaptureDeviceVendorID(11));
ia.SetCaptureDeviceTypeID(IrisCaptureDeviceTypeID(7));

const auto irisCapture =
  IrisCapture::Create(std::move(image), IrisPosition::LEFT_EYE, ia);
IrisDetector irisDetector;
auto iris = irisCapture->DetectWith(irisDetector)->iris;

std::cout « "   Conversion Iris to ISO image" « std::endl;
auto encoder = PngImageEncoder();
auto isoImage = iris->ToIsoImage(encoder);

std::cout « "   Conversion ISO to Iris image" « std::endl;
auto irisImage = irisCapture->FromIsoImage(isoImage);
```

java

```
Image image = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));

IrisAttributes ia = new IrisAttributes();
ia.SetCaptureDeviceTechnology(IrisCaptureDeviceTechnology.CMOS_CCD);
ia.SetCaptureDeviceVendorID(new IrisCaptureDeviceVendorID(11));
ia.SetCaptureDeviceTypeID(new IrisCaptureDeviceTypeID(7));

IrisCapture irisCapture = new IrisCapture(image, IrisPosition.LEFT_EYE, ia);
IrisDetector irisDetector = new IrisDetector();
IrisSegment segment = irisCapture.DetectWith(irisDetector);
Iris iris = segment.getIris();

System.out.println("   Conversion Iris to ISO image");
PngImageEncoder encoder = new PngImageEncoder();
Bytes isoImage = iris.ToIsoImage(encoder);

System.out.println("   Conversion ISO to Iris image");
IrisCapture irisImage = IrisCapture.FromIsoImage(isoImage);
```

csharp

```
Image image = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));

IrisAttributes ia = new IrisAttributes();
ia.SetCaptureDeviceTechnology(IrisCaptureDeviceTechnology.CMOS_CCD);
ia.SetCaptureDeviceVendorID(new IrisCaptureDeviceVendorID(11));
ia.SetCaptureDeviceTypeID(new IrisCaptureDeviceTypeID(7));

IrisCapture irisCapture = new IrisCapture(image, IrisPosition.LEFT_EYE, ia);
IrisDetector irisDetector = new IrisDetector();
Iris iris = irisCapture.DetectWith(irisDetector).iris;

System.Console.WriteLine("   Conversion Iris to ISO image");
var encoder = new PngImageEncoder();
var isoImage = iris.ToIsoImage(encoder);

System.Console.WriteLine("   Conversion ISO to Iris image");
var irisImage = IrisCapture.FromIsoImage(isoImage);
```

3.8.2.5.2 ISO Masked and Cropped Image

The SDK support masking and cropping detected iris into image that is compliant with ISO/IEC 19794-6:2011. After iris detection use function Iris::GetISO19794IrisImage().

C++

```
auto image = Image::Decode(BinaryFile::ReadAll("assets/eye_L.png"));
```

```
auto irisCapture = IrisCapture::Create(std::move(image), IrisPosition::LEFT_EYE);
IrisDetector irisDetector;
auto iris = irisCapture->DetectWith(irisDetector)->iris;

auto maskedImage = iris->GetISO19794IrisImage();
maskedImage->SaveAs("output/eye_L_iso197946.png");
```

java

```
Image image = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));
IrisCapture irisCapture = new IrisCapture(image, IrisPosition.LEFT_EYE);
IrisDetector irisDetector = new IrisDetector();
IrisSegment segment = irisCapture.DetectWith(irisDetector);
Iris iris = segment.getIris();

Image maskedImage = iris.GetISO19794IrisImage();
maskedImage.SaveAs("output/eye_L_iso197946.png");
```

csharp

```
var image = Image.Decode(BinaryFile.ReadAll("assets/eye_L.png"));
var irisCapture = new IrisCapture(image, IrisPosition.LEFT_EYE);
var irisDetector = new IrisDetector();
Iris iris = irisCapture.DetectWith(irisDetector).iris;

Image maskedImage = iris.GetISO19794IrisImage();
maskedImage.SaveAs("output/eye_L_iso197946.png");
```

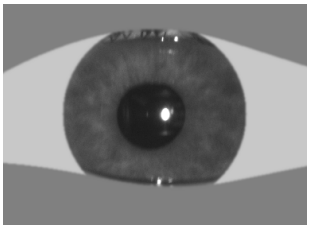


Figure 53 ISO/IEC 19794-6:2011 masked iris

3.8.3 Typedef Documentation

3.8.3.1 IrisSimilarityScore

```
using IrisSimilarityScore = unsigned int
```

Score is 0-1000 where 1000 represents most similar templates.

Scores in range <0, 60> are computed based on FAR analysis. Scores above this range are approximated with linear slope to 1000. Therefore very similar irises can have matching score near maximal 1000.

See Similarity Score.

3.8.4 Enumeration Type Documentation

3.8.4.1 IrisCaptureDeviceTechnology

```
enum class IrisCaptureDeviceTechnology : uint8_t [strong]
```

Enumerator

UNKNOWN	Unknown or Unspecified.
CMOS_CCD	Static photograph from an unknown source.

3.8.4.2 IrisImageType

```
enum class IrisImageType : uint8_t [strong]
```

Enumerator

UNCROPPED	An uncropped rectilinear iris image. Reference: ISO/IEC 19794-6:2011(E), Sections 7.4 table 4, Cropped and Masked Iris Image specifications [7] .
VGA	A rectilinear iris image in VGA (640x480) format. Reference: ISO/IEC 19794-6:2011(E), Sections 7.4 table 4, Cropped and Masked Iris Image specifications [7] .
CROPPED	A cropped, centred, iris image with (0,6R 0,2R) margins. Reference: ISO/IEC 19794-6:2011(E), Sections 7.4 table 4, Cropped and Masked Iris Image specifications [7] .
CROPPED_AND_MASKED	A cropped and region-of-interest masked, centred, iris image with (0,6R 0,2R) margins. Reference: ISO/IEC 19794-6:2011(E), Sections 7.4 table 4, Cropped and Masked Iris Image specifications [7] .

3.8.4.3 IrisPosition

```
enum class IrisPosition : uint8_t [strong]
```

Enumerator

RIGHT_EYE	Iris of right eye.
LEFT_EYE	Iris of left eye.

3.8.5 Function Documentation

3.8.5.1 IrisImageTypeToIso()

```
constexpr int IrisImageTypeToIso (
            IrisImageType type) [static], [constexpr]
```

Parameters

type	IrisImageType

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

ISO iris image type

3.8.5.2 IrisPositionToIsoPosition()

```
constexpr int IrisPositionToIsoPosition (
            IrisPosition irisPosition)  [static], [constexpr]
```

[7]

Parameters

irisPosition	IrisPosition

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

ISO position

3.8.5.3 IrisToIsoCaptureDeviceTechnology()

```
constexpr int IrisToIsoCaptureDeviceTechnology (
            IrisCaptureDeviceTechnology printCaptureDeviceTechnology)  [static], [constexpr]
```

Parameters

printCaptureDeviceTechnology	IrisCaptureDeviceTechnology

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

ISO capture technology

3.8.5.4 IsoToIrisCaptureDeviceTechnology()

```
constexpr IrisCaptureDeviceTechnology IsoToIrisCaptureDeviceTechnology (
            int isoCaptureDeviceTechnology) [static], [constexpr]
```

Parameters

isoCaptureDeviceTechnology	iso capture technology

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

IrisCaptureDeviceTechnology

3.8.5.5 IsoToIrisImageType()

```
constexpr IrisImageType IsoToIrisImageType (
            int type) [static], [constexpr]
```

Parameters

type	ISO iris image type

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

IrisImageType

3.8.5.6 IsoToIrisPosition()

```
constexpr IrisPosition IsoToIrisPosition (
            int irisPosition) [static], [constexpr]
```

[7]

Parameters

irisPosition	iso position

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

IrisPosition

3.9 Logger

The library defines Logger interface to log messages.

Classes

- class FormattingComposer

  Compose message from variadic arguments.
- class FormattingLogger

  It provides functions with format specifier to compose and log message via provided Logger.
- class LogMessage

  Compose Logger message.
- class Logger

  Interface for library logging.
- class NullLogger

  The logger implementation that do not logs.
- struct LoggerCallBacks

  LoggerCallBacks contains logger handle and function pointers as callbacks.
- class CallbackLogger

  CallbackLogger is logger for binary part of libary due to C boundary.
- class GlobalLogger

  Holds instance of Logger.

Typedefs

- using LogMessageCallback = const char∗ (∗)(void∗ composerData)

  Function pointer to compose message in binary.
- using LoggerCallback = void (∗)(void∗ logger, LogMessageCallback composer, void∗ composerData)

  Function pointer to log from binary.

3.9.1 Detailed Description

The library defines Logger interface to log messages. The library knows when and what to log, but how it leaves on library user. The default GlobalLogger is dummy implementation, that ignores all messages on all logging levels. The client can set global Logger for whole library.

c++

```
class ConsoleLogger : public Logger
{
    void LogInfo(LogMessage& msg) final
    {
        std::cout « "   " « msg.Compose() « std::endl;
    }
    void LogError(LogMessage& msg) final
    {
        std::cout « "   error: " « msg.Compose() « std::endl;
    }
    void LogDebug(LogMessage& msg) final
    {
        std::cout « "   debug: " « msg.Compose() « std::endl;
    }
};

static void Log()
{
    GlobalLogger::Set(std::make_shared<ConsoleLogger>());
}
```

java

```
class ConsoleLogger extends Logger {
    public void LogInfo(LogMessage msg) {
        System.out.println("   native: " + msg.Compose());
    }

    public void LogError(LogMessage msg) {
        System.out.println("   native:error: " + msg.Compose());
    }

    public void LogDebug(LogMessage msg) {
        System.out.println("   native:debug: " + msg.Compose());
    }
}

public void Log() {
    Logger log = new ConsoleLogger();
    GlobalLogger.Set(log);
}
```

csharp

```
class ConsoleLogger : Logger
{
    public override void LogInfo(LogMessage msg)
    {
        System.Console.WriteLine("   native: " + msg.Compose());
    }
    public override void LogError(LogMessage msg)
    {
        System.Console.WriteLine("   native:error: " + msg.Compose());
    }
    public override void LogDebug(LogMessage msg)
    {
        System.Console.WriteLine("   native:debug: " + msg.Compose());
    }
}

public void Log()
{
    Logger log = new ConsoleLogger();
    GlobalLogger.Set(log);
}
```

The Logger can be implemented also in Java / CSharp bindings.

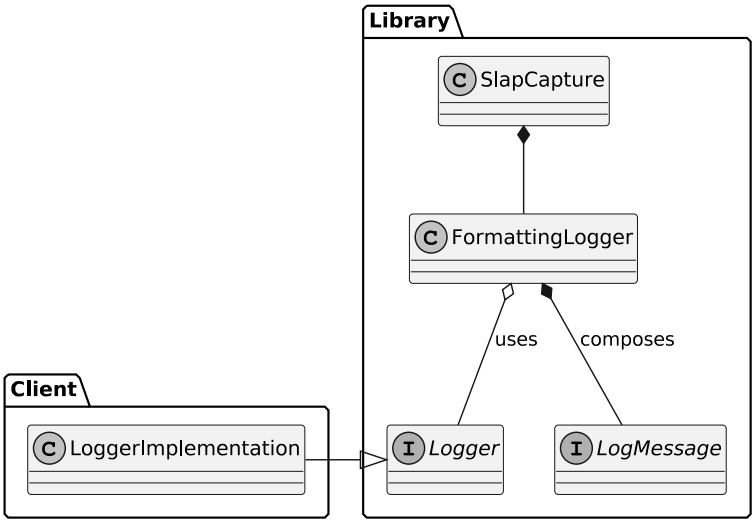


Figure 54 Logger design

The logger has 3 levels

- Info - on interface boundaries.
- Error - when error is reported by used libraries of in wrong arguments
- Debug - messages that helps with bug hunting in this library

Client decides which level is logged. The LogMessage is used to avoid unnecessary composition of logger message. LogMessage is composed only when client wants to log it.

Warning

Client MUST NOT store the LogMessage and use it outside Logger::LogInfo, Logger::LogError, Logger::LogDebug call. The library stores only references to data in LogMessage. The references can be invalid when Logger functions end.

Note

Only for C++ the library provides also FormattingComposer, FormattingLogger that extends (and use) Logger with functionality similar as `printf` function.

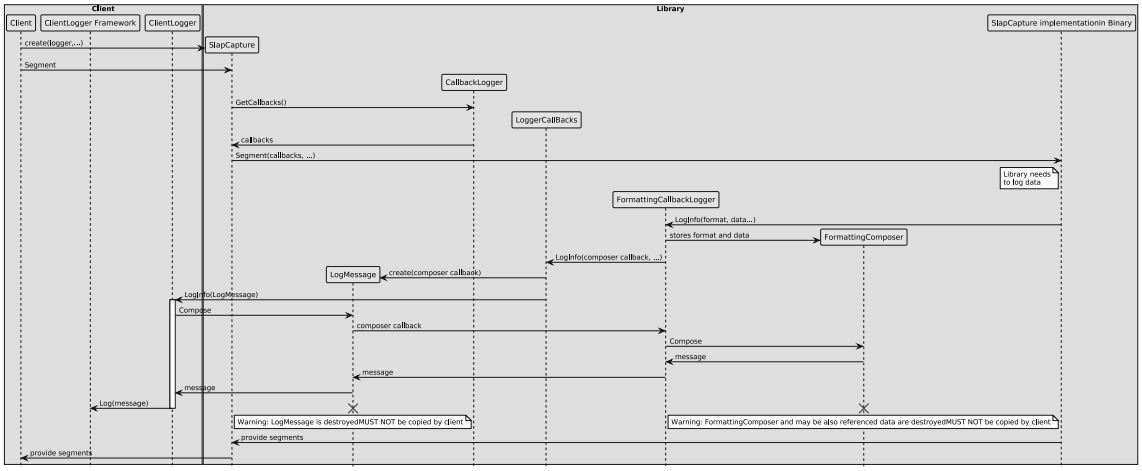


Figure 55 Logger design

For more details see Logging overview.

3.9.2 Typedef Documentation

3.9.2.1 LoggerCallback

```
using LoggerCallback = void (*)(void* logger, LogMessageCallback composer, void* composerData)
```

It is provided by header part and used in binary to log messages. The parameter logger is created and used ONLY in header part.

Parameters

logger	is pointer to Logger class used in header part. It is provide to binary in LoggerCallBacks and provide it back from binary to header part.
composer	is LogMessageCallback called when header part wants to compose message in binary part of library
composerData	data necessary for composer, it is interpreted and used ONLY in binary part of library

Note

Only for C++

3.9.2.2 LogMessageCallback

```
using LogMessageCallback = const char* (*)(void* composerData)
```

It is called by header part of library and executed in binary part of library. The parameter composerData is created and used ONLY in binary part.

Parameters

composerData	data that is interpreted and used ONLY in binary part of library

Returns

message

Note

Only for C++

3.10 Print

The library can be used to segment slaps and extract templates from prints.

Classes

- class ModalitiesEmbeddingVerifyMatcher

  ModalitiesEmbeddingVerifyMatcher is template matcher that uses Innovatrics proprietary verification algorithm.
- class ANSITemplate

  ANSI INCITS 378 print compliant template.
- class ANSITemplateAdapter

  Adapts Template to ANSITemplate.
- class CriticalPoint

  It represents a particular CriticalPoint.
- class CriticalPointShape

  It draws CriticalPoint into image.
- class CriticalPointsShape

  It draws collection of CriticalPoint into image.
- class EmbeddingExtractor
- class EmbeddingMatcher

  It is general matcher for Embedding templates.
- class EmbeddingVerifyMatcher

  EmbeddingVerifyMatcher is template matcher that uses Innovatrics proprietary verification algorithm.
- class ICSExtractor

  ICSExtractor creates ICS, Innovatrics proprietary print template.
- class ICSVerifyMatcher

  ICSVerifyMatcher is template matcher that uses Innovatrics proprietary verification algorithm.
- class IdentificationEmbeddingExtractor

  EmbeddingExtractor creates Innovatrics proprietary embedding print template.
- class ILOSIDTemplate

  International Labour Organisation Seafarers Identity Document template.
- class ILOSIDTemplateAdapter

  Adapts Template to ILOSIDTemplate.
- class ISOCCTemplate

  ISO/IEC 19794-2:2005 template, template version 2.0, 3-byte format for minutiae (compact card encoding).
- class ISOCCTemplateAdapter

  Adapts Template to ISOCCTemplate.
- class ISONCTemplate

  ISO/IEC 19794-2:2005 template, template version 2.0, 5-byte format for minutiae (normal card encoding).
- class ISONCTemplateAdapter

  Adapts Template to ISONCTemplate.
- class ISOTemplate

  ISO/IEC 19794-2 print compliant template.
- class ISOTemplateAdapter

  Adapts Template to ISOTemplate.
- class Minutia

  It represents a particular Minutia (distinctive fingerprint feature point found in fingerprint skeleton, such as a bifurcation or an ending).
- class MinutiaShape

  It draws Minutia into image.
- class MinutiaeShape

  It draws Minutia into image.
- class Print

  It holds all functions and data related to print manipulation.
- class PrintAttributes

Print attributes Holds the print attributes.

- class PrintCaptureDeviceTypeID

  Capture device type identifier The capture device type identifier shall identify the biometric organization that owns the product that created the BDIR.
- class PrintCaptureDeviceVendorID

  Capture device vendor identifier The capture device vendor identifier shall identify the biometric organization that owns the product that created the BDIR.
- class PrintCapture

  It holds all functions and data related to print manipulation.
- class GuessedHandPosition

  It represents guessed hand position of Slap.
- class SlapCapture

  It represents PrintCapture with Slap.
- class ThumbsCapture

  It represents PrintCapture with ThumbsCapture.
- class PrintDetectorConfig

  General configuration of PrintDetector.
- class PrintDetector

  PrintDetector implements segmentation by means of Innovatrics ISegLib library.
- class AlgorithmicSegmentation

  AlgorithmicSegmentation implements algorithmic segmentation from Innovatrics ISegLib library.
- class NeuralSegmentation

  NeuralSegmentation implements segmentation based on neural network from Innovatrics ISegLib library.
- class GlobalPrintExecutor

  Holds instance of PrintExecutor.
- class PrintCropConfiguration

  Holds print cropping configuration.
- class PrintSegment

  Segment detected in print capture Usually the segment has also position that is assigned by segmentation algorithm.
- class PrintTemplate

  Holds various biometric representation types for print.

Typedefs

- using PrintSimilarityScore = unsigned int

  It is similarity score between print Templates.

Enumerations

- enum class PrintCaptureDeviceTechnology : uint8_t {
  UNKNOWN = 0 , WHITE_LIGHT_OPTICAL_TIR = 1 , WHITE_LIGHT_OPTICAL_DIRECT = 2 , WHITE_LIGHT_OPTICAL_TOUCHLESS = 3 ,
  MONOCHROMATIC_VISIBLE_OPTICAL_TIR = 4 , MONOCHROMATIC_VISIBLE_OPTICAL_DIRECT = 5 , MONOCHROMATIC_VISIBLE_OPTICAL_TOUCHLESS = 6 , MONOCHROMATIC_VISIBLE_IR_OPTICAL_TIR = 7 ,
  MONOCHROMATIC_VISIBLE_IR_OPTICALTOUCHLESS = 8 , MONOCHROMATIC_VISIBLE_IR_OPTICAL = 9 , MULTISPECTRAL_OPTICAL_TIR = 10 , MULTISPECTRAL_OPTICAL_DIRECT = 11 ,
  MULTISPECTRAL_OPTICAL_TOUCHLESS = 12 , ELECTRO_LUMINESCENT = 13 , SEMICONDUCTOR_CAPACITIVE = 14 , SEMICONDUCTOR_RF = 15 ,
  SEMICONDUCTOR_THERMAL = 16 , PRESSURE_SENSITIVE = 17 , ULTRASOUND = 18 , MECHANICAL = 19 ,
  GLASS_FIBER = 20 }

It shall indicate the class of capture device technology used to acquire the captured biometric sample [5] .

- enum class PrintImpression : uint8_t {
  LIVE_SCAN_PLAIN = 0 , LIVE_SCAN_ROLLED = 1 , NONLIVE_SCAN_PLAIN = 2 , NONLIVE_SCAN_ROLLED = 3 ,
  LATENT_IMPRESSION = 4 , LATENT_TRACING = 5 , LATENT_PHOTO = 6 , LATENT_LIFT = 7 ,
  LIVE_SCAN_SWIPE = 8 , LIVE_SCAN_VERTICAL_ROLL = 9 , LIVE_SCAN_PALM = 10 , NONLIVE_SCAN_PALM = 11 ,
  LATENT_PALM_IMPRESSION = 12 , LATENT_PALM_TRACING = 13 , LATENT_PALM_PHOTO = 14 , LATENT_PALM_LIFT = 15 ,
  LIVE_SCAN_OPTICAL_CONCTACTLESS_PLAIN = 24 , OTHER = 28 , UNKNOWN = 29 }

  It is impression of prints [5] .
- enum class PrintPosition : uint8_t {
  RIGHT_THUMB = 1 , RIGHT_INDEX = 2 , RIGHT_MIDDLE = 3 , RIGHT_RING = 4 ,
  RIGHT_LITTLE = 5 , LEFT_THUMB = 6 , LEFT_INDEX = 7 , LEFT_MIDDLE = 8 ,
  LEFT_RING = 9 , LEFT_LITTLE = 10 , RIGHT_FULL_PALM = 21 , RIGHT_WRITERS_PALM = 22 ,
  LEFT_FULL_PALM = 23 , LEFT_WRITERS_PALM = 24 , RIGHT_LOWER_PALM = 25 , RIGHT_UPPER_PALM = 26 ,
  LEFT_LOWER_PALM = 27 , LEFT_UPPER_PALM = 28 , RIGHT_INTERDIGITAL = 31 , RIGHT_THENAR = 32 ,
  RIGHT_HYPOTHENAR = 33 , LEFT_INTERDIGITAL = 34 , LEFT_THENAR = 35 , LEFT_HYPOTHENAR = 36 }

  It is position of finger on hands.

Functions

- static constexpr int PrintToIsoCaptureDeviceTechnology (PrintCaptureDeviceTechnology printCaptureDeviceTechnology)

  Function converts PrintCaptureDeviceTechnology to ISO capture technology.
- static constexpr PrintCaptureDeviceTechnology IsoToPrintCaptureDeviceTechnology (int isoCaptureDeviceTechnology)

  Function converts ISO capture technology to PrintCaptureDeviceTechnology.
- static constexpr int PrintToIsoImpression (PrintImpression printImpression)

  Function converts PrintImpression to ISO impression.
- static constexpr PrintImpression IsoToPrintImpression (int isoImpression)

  Function converts ISO impression to PrintImpression.
- static constexpr PrintPosition IsoToPrintPosition (int isoPosition)

  Function converts ISO position to PrintPosition.
- static constexpr int PrintToIsoPosition (PrintPosition position)

  Function converts Print position to ISO position.

3.10.1 Detailed Description

3.10.2 Overview

The library can be used to segment slaps and extract templates from prints. The library can process following print types

- Slap
- Thumbs
- Print (types are different only by PrintPosition)
  - Fingerprint
  - Palm

We are able to segment Slap and provide segmented Prints for further processing. Usually the processing is template extraction from Print. The library provides ICSExtractor to extract Print.

3.10.2.1 Slap

Slap is image of usually 4 fingers. It can be segmented and result of segmentation is PrintSegment. Usually the fingertip has also position that is assigned by segmentation algorithm. The position of found fingertip on hand is estimated by comparing the x-coordinate of the center of the finger's bounding box rectangle. As a result, the estimation may not be precise for slaps with an angle greater than 45 degrees. When the segmentation algorithm does not detect position then it is necessary that user sets correct position to be possible to create Print for further processing.

We provides following algorithm for segmentation process:

- Neural
- Algorithmic

Figure 56 Slap

It can be segmented with function Slap::SegmentWith that return segmented rectangles.

c++

```
const auto dpi = 1000;
auto img = Image::Decode(BinaryFile::ReadAll("assets/slap.png"), dpi);
auto slap = SlapCapture::Create(std::move(img), HandPosition::RIGHT);

auto cfg = AlgorithmicSegmentation::Config::Default();
cfg.SetMaxRotation(10);
AlgorithmicSegmentation segmentation(cfg);

unsigned int fingerIndex = 0;
auto segments = slap->SegmentWith(segmentation);
for (auto& segment : segments)
{
    const std::string file =
      "output/slap_segment_" + std::to_string(fingerIndex) + ".png";
    segment->ToPrint()->image->SaveAs(file);
    fingerIndex++;
}
slap->Draw(segments, Shapes::InnoBlue)->SaveAs("output/slap_segments.png");
```

java

```
final int dpi = 1000;
Image img = Image.Decode(BinaryFile.ReadAll("assets/slap.png"), dpi);
SlapCapture slap = SlapCapture.Create(img, HandPosition.RIGHT);

AlgorithmicSegmentation.Config cfg = AlgorithmicSegmentation.Config.Default();
cfg.SetMaxRotation(10);
AlgorithmicSegmentation segmentation = new AlgorithmicSegmentation(cfg);

int fingerIndex = 0;
PrintSegments segments = slap.SegmentWith(segmentation);
for (PrintSegment segment : segments) {
    segment.ToPrint().getImage().SaveAs(
            "output/slap_segment_" + fingerIndex + ".png");
```

```
    fingerIndex++;
}

slap.Draw(segments, Shapes.InnoBlue).SaveAs("output/slap_segments.png");
```

csharp

```
const int dpi = 1000;
Image img = Image.Decode(BinaryFile.ReadAll("assets/slap.png"), dpi);
SlapCapture slap = SlapCapture.Create(img, HandPosition.RIGHT);

uint fingerIndex = 0;
AlgorithmicSegmentation.Config cfg = AlgorithmicSegmentation.Config.Default();
cfg.SetMaxRotation(10);
AlgorithmicSegmentation segmentation = new AlgorithmicSegmentation(cfg);

PrintSegments segments = slap.SegmentWith(segmentation);
foreach (PrintSegment segment in segments)
{
    segment.ToPrint().image.SaveAs("output/slap_segment_" + fingerIndex + ".png");
    fingerIndex++;
}

slap.Draw(segments, Shapes.InnoBlue).SaveAs("output/slap_segments.png");
```

The output of example is segmented slap

Figure 57 Segmented Slap

and cropped segments

Figure 58 Slap Segment 1

Figure 59 Slap Segment 2

Figure 60 Slap Segment 3

Figure 61 Slap Segment 4

3.10.2.1.1 Modify segment dimensions and crop

It is possible to use crop configuration (PrintCropConfiguration) when creating prints from PrintSegment. It is also possible to modified segment dimension via function PrintSegment::Resize.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/canted_slap_left.png"));
auto slap = SlapCapture::Create(std::move(img), HandPosition::LEFT);

NeuralSegmentation segmentation;
PrintCropConfiguration cropCfg;
cropCfg.SetBackgroundColor(Colors::InnoBlue);
cropCfg.SetImageDimensions(400, 500);
auto segments = slap->SegmentWith(segmentation);
auto modifiedSegments = segments;

// Get aligned fingerprint into image with dimension 400x500
segments[0]->ToPrint(cropCfg)->image->SaveAs("output/slap_segment_crop_0.png");
// Resize rounding box to absolute values and get aligned fingerprint into image
// with dimension 400x500
modifiedSegments[1] = segments[1]->Resize(400.0F, 500.F);
modifiedSegments[1]->ToPrint(cropCfg)->image->SaveAs(
  "output/slap_segment_crop_1.png");
// Resize rounding box relatively to original one and get not aligned fingerprint
// into image with dimension 400x500
```

```
cropCfg.EnableAlignedCropping(false);
auto segment2 = segments[2];
modifiedSegments[2] = segment2->Resize(segment2->roundingBox.GetWidth() * 1.3F,
                                       segment2->roundingBox.GetHeight() * 1.5F);
modifiedSegments[2]->ToPrint(cropCfg)->image->SaveAs(
  "output/slap_segment_crop_2.png");
// Resize rounding box to absolute values and get aligned fingerprint into image
// with dimensions same as rounding box. The default crop configuration is used in
// ToPrint() function.
modifiedSegments[3] = segments[3]->Resize(300.0F, 400.0F);
modifiedSegments[3]->ToPrint()->image->SaveAs("output/slap_segment_crop_3.png");

// Draw segments into image. Original one as light gray and modified as blue
auto segmentsImage =
  slap->image->DrawShape(SegmentsShape(segments, modifiedSegments));
segmentsImage->SaveAs("output/slap_segment_crop.png");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/canted_slap_left.png"));
SlapCapture slap = SlapCapture.Create(img, HandPosition.LEFT);

NeuralSegmentation segmentation = new NeuralSegmentation();
PrintCropConfiguration cropCfg = new PrintCropConfiguration();
cropCfg.SetBackgroundColor(Colors.InnoBlue);
cropCfg.SetImageDimensions(400, 500);
PrintSegments segments = slap.SegmentWith(segmentation);
PrintSegments modifiedSegments = new PrintSegments(segments);

// Get aligned fingerprint into image with dimension 400x500
segments.get(0).ToPrint(cropCfg).getImage().SaveAs(
        "output/slap_segment_crop_0.png");
// Resize rounding box to absolute values and get aligned fingerprint into image
// with dimension 400x500
modifiedSegments.set(1, segments.get(1).Resize(400.0F, 500.0F));
modifiedSegments.get(1).ToPrint(cropCfg).getImage().SaveAs(
        "output/slap_segment_crop_1.png");
// Resize rounding box relatively to original one and get not aligned
// fingerprint
// into image with dimension 400x500
cropCfg.EnableAlignedCropping(false);
PrintSegment segment2 = segments.get(2);
modifiedSegments.set(
        2, segment2.Resize(segment2.getRoundingBox().GetWidth() * 1.3F,
                   segment2.getRoundingBox().GetHeight() * 1.5F));
modifiedSegments.get(2).ToPrint(cropCfg).getImage().SaveAs(
        "output/slap_segment_crop_2.png");
// Resize rounding box to absolute values and get aligned fingerprint into image
// with dimensions same as rounding box. The default crop configuration is used
// in ToPrint() function.
modifiedSegments.set(3, segments.get(3).Resize(300.0F, 400.0F));
modifiedSegments.get(3).ToPrint().getImage().SaveAs(
        "output/slap_segment_crop_3.png");

// Draw segments into image. Original one as light gray and modified as blue
Image segmentsImage = slap.getImage().DrawShape(
        new SegmentsShape(segments, modifiedSegments));
segmentsImage.SaveAs("output/slap_segment_crop.png");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/canted_slap_left.png"));
SlapCapture slap = SlapCapture.Create(img, HandPosition.LEFT);

NeuralSegmentation segmentation = new NeuralSegmentation();
PrintCropConfiguration cropCfg = new PrintCropConfiguration();
cropCfg.SetBackgroundColor(Colors.InnoBlue);
cropCfg.SetImageDimensions(400, 500);
PrintSegments segments = slap.SegmentWith(segmentation);
PrintSegments modifiedSegments = new PrintSegments(segments);

// Get aligned fingerprint into image with dimension 400x500
segments[0].ToPrint(cropCfg).image.SaveAs("output/slap_segment_crop_0.png");
// Resize rounding box to absolute values and get aligned fingerprint into image
// with dimension 400x500
modifiedSegments[1] = segments[1].Resize(400.0F, 500.0F);
modifiedSegments[1].ToPrint(cropCfg).image.SaveAs(
    "output/slap_segment_crop_1.png");
// Resize rounding box relatively to original one and get not aligned fingerprint
// into image with dimension 400x500
cropCfg.EnableAlignedCropping(false);
PrintSegment segment2 = segments[2];
modifiedSegments[2] = segment2.Resize(segment2.roundingBox.GetWidth() * 1.3F,
                                      segment2.roundingBox.GetHeight() * 1.5F);
modifiedSegments[2].ToPrint(cropCfg).image.SaveAs(
    "output/slap_segment_crop_2.png");
```

```
// Resize rounding box to absolute values and get aligned fingerprint into image
// with dimensions same as rounding box. The default crop configuration is used in
// ToPrint() function.
modifiedSegments[3] = segments[3].Resize(300.0F, 400.0F);
modifiedSegments[3].ToPrint().image.SaveAs("output/slap_segment_crop_3.png");

// Draw segments into image. Original one as light gray and modified as blue
Image segmentsImage =
    slap.image.DrawShape(new SegmentsShape(segments, modifiedSegments));
segmentsImage.SaveAs("output/slap_segment_crop.png");
```

The output of example is segmented slap (Original segments are light gray and modified are blue)

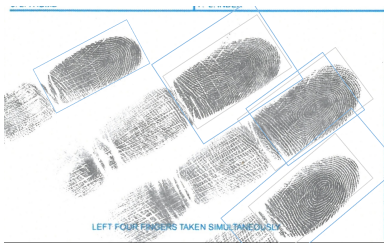


Figure 62 Modified Segmented Slap

and cropped segments

Figure 63 Modified Slap Segment 1

Figure 64 Modified Slap Segment 2

Figure 65 Modified Slap Segment 3

Figure 66 Modified Slap Segment 4

3.10.2.1.2 Hand position

Guess hand position. It is possible to guess hand position when slap contains at least 3 segments. The hand position can be obtained by function SlapCapture::GuessHandPosition.

C++

```
const auto dpi = 1000;
auto img = Image::Decode(BinaryFile::ReadAll("assets/slap.png"), dpi);
auto slap = SlapCapture::Create(std::move(img), HandPosition::RIGHT);

AlgorithmicSegmentation segmentation{};
auto segments = slap->SegmentWith(segmentation);

auto guessedHand = SlapCapture::GuessHandPosition(segments);
if (guessedHand.IsGuessSuccessful())
{
    std::cout « "   Guessed hand position " « guessedHand.GetHandPosition()
              « std::endl;
}
```

java

```
final int dpi = 1000;
Image img = Image.Decode(BinaryFile.ReadAll("assets/slap.png"), dpi);
SlapCapture slap = SlapCapture.Create(img, HandPosition.RIGHT);

AlgorithmicSegmentation segmentation = new AlgorithmicSegmentation();
PrintSegments segments = slap.SegmentWith(segmentation);

GuessedHandPosition guessedHand = SlapCapture.GuessHandPosition(segments);
if (guessedHand.IsGuessSuccessful()) {
    System.out.println(
            "   Guessed hand position " + guessedHand.GetHandPosition());
}
```

csharp

```
const int dpi = 1000;
Image img = Image.Decode(BinaryFile.ReadAll("assets/slap.png"), dpi);
SlapCapture slap = SlapCapture.Create(img, HandPosition.RIGHT);

AlgorithmicSegmentation segmentation = new AlgorithmicSegmentation();
PrintSegments segments = slap.SegmentWith(segmentation);

GuessedHandPosition guessedHand = SlapCapture.GuessHandPosition(segments);
if (guessedHand.IsGuessSuccessful())
{
    System.Console.WriteLine("   Guessed hand position " +
                             guessedHand.GetHandPosition());
}
```

3.10.2.2 Thumbs

Thumbs is image of both thumbs. When 2 segments are detected then first is left and second is right thumb. When only one thumb is detected then position is unknown and user MUST set correct position to be possible to create Print for further processing.

We provides following algorithm for segmentation process:

- Neural
- Algorithmic

Figure 67 Thumbs

It can be segmented with function Thumbs::SegmentWith that return segmented rectangles.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/thumbs.png"));
auto thumbs = ThumbsCapture::Create(std::move(img));

auto cfg = AlgorithmicSegmentation::Config::Default();
cfg.SetMaxRotation(10);
AlgorithmicSegmentation segmentation(cfg);

unsigned int fingerIndex = 0;
auto segments = thumbs->SegmentWith(segmentation);
for (auto& segment : segments)
{
    const std::string file =
      "output/thumbs_segment_" + std::to_string(fingerIndex) + ".png";
    segment->ToPrint()->image->SaveAs(file);
    fingerIndex++;
}

thumbs->Draw(segments, Shapes::InnoBlue)->SaveAs("output/thumbs_segments.png");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/thumbs.png"));
ThumbsCapture thumbs = ThumbsCapture.Create(img);

AlgorithmicSegmentation.Config cfg = AlgorithmicSegmentation.Config.Default();
cfg.SetMaxRotation(10);
AlgorithmicSegmentation segmentation = new AlgorithmicSegmentation(cfg);

int fingerIndex = 0;
PrintSegments segments = thumbs.SegmentWith(segmentation);
for (PrintSegment segment : segments) {
    segment.ToPrint().getImage().SaveAs(
            "output/thumbs_segment_" + fingerIndex + ".png");
    fingerIndex++;
}

thumbs.Draw(segments, Shapes.InnoBlue).SaveAs("output/thumbs_segments.png");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/thumbs.png"));
ThumbsCapture thumbs = ThumbsCapture.Create(img);

uint fingerIndex = 0;
AlgorithmicSegmentation.Config cfg = AlgorithmicSegmentation.Config.Default();
cfg.SetMaxRotation(10);
AlgorithmicSegmentation segmentation = new AlgorithmicSegmentation(cfg);

PrintSegments segments = thumbs.SegmentWith(segmentation);
foreach (PrintSegment segment in segments)
{
    segment.ToPrint().image.SaveAs("output/thumbs_segment_" + fingerIndex +
                                   ".png");
    fingerIndex++;
}

thumbs.Draw(segments, Shapes.InnoBlue).SaveAs("output/thumbs_segments.png");
```

The output of example are segmented thumbs

Figure 68 Segmented Thumbs

and cropped segments

Figure 69 Thumbs Segment 1

Figure 70 Thumbs Segment 2

3.10.2.3 Prints

We can segment images with one or more fingerprints on it (up to 4 fingerprints). The position of segmented fingerprints are always unknown and user MUST set correct position to be possible to create Print for further processing.

We provides following algorithm for segmentation process:

- Neural
- Algorithmic

Figure 71 Print

It can be segmented with function PrintCapture::SegmentWith that return segmented rectangles.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/print.png"));
auto print = PrintCapture::Create(std::move(img));
NeuralSegmentation segmentation;
auto segments = print->SegmentWith(segmentation);
// We know that there is only one finger segmented
auto segment = segments[0];
// The segment has no position, we have to set it for
// further processing
segment->SetPosition(PrintPosition::LEFT_MIDDLE);
segment->ToPrint()->image->SaveAs("output/print_nn_segment.png");

print->Draw(segments, Shapes::InnoBlue)->SaveAs("output/print_nn_segments.png");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/print.png"));
PrintCapture print = PrintCapture.Create(img);
NeuralSegmentation segmentation = new NeuralSegmentation();
PrintSegments segments = print.SegmentWith(segmentation);
// We know that there is only one finger segmented
PrintSegment segment = segments.get(0);
// The segment has no position, we have to set it for
// further processing
segment.SetPosition(PrintPosition.LEFT_MIDDLE);
segment.ToPrint().getImage().SaveAs("output/print_nn_segment.png");

print.Draw(segments, Shapes.InnoBlue).SaveAs("output/print_nn_segments.png");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/print.png"));
PrintCapture print = PrintCapture.Create(img);
NeuralSegmentation segmentation = new NeuralSegmentation();
PrintSegments segments = print.SegmentWith(segmentation);
// We know that there is only one finger segmented
PrintSegment segment = segments[0];
// The segment has no position, we have to set it for
// further processing
segment.SetPosition(PrintPosition.LEFT_MIDDLE);
segment.ToPrint().image.SaveAs("output/print_nn_segment.png");

print.Draw(segments, Shapes.InnoBlue).SaveAs("output/print_nn_segments.png");
```

The output of example is segmented fingerprint

Figure 72 Segmented Print

and its cropp

Figure 73 Fingerprint Segment

3.10.2.4 Extraction

The SDK provides following extraction types

- ICS Extraction

3.10.2.4.1 ICS Extraction

We provide ICS (Internal Innovatrics template) for Print via ICSExtractor. The main goal is to create template for further processing e.g. 1:1 Verification. During the extraction some intermediate artifacts can be provided.

Note

The adjudicator feature MUST be allowed in license to obtain intermediate artifacts.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/finger.png"));
auto fp = Print::Create(std::move(img), PrintPosition::LEFT_INDEX);

ICSExtractor::Artifacts artifacts;
ICSExtractor::SkeletonImage skeleton;
artifacts.Add(skeleton);
ICSExtractor::FilteredImage filtered;
artifacts.Add(filtered);
ICSExtractor::BinarizedImage binarized;
artifacts.Add(binarized);

ICSExtractor extractor;
auto tmpl = extractor.Extract(fp, artifacts);

skeleton.GetImage()->SaveAs("output/skeleton.png");
filtered.GetImage()->SaveAs("output/filtered.png");
binarized.GetImage()->SaveAs("output/binarized.png");
BinaryFile::WriteAll("output/fp.ics", tmpl->icsTemplateData);
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
Print fp = new Print(img, PrintPosition.LEFT_INDEX);

ICSExtractor.Artifacts artifacts = new ICSExtractor.Artifacts();
ICSExtractor.SkeletonImage skeleton = new ICSExtractor.SkeletonImage();
artifacts.Add(skeleton);
ICSExtractor.FilteredImage filtered = new ICSExtractor.FilteredImage();
artifacts.Add(filtered);
ICSExtractor.BinarizedImage binarized = new ICSExtractor.BinarizedImage();
artifacts.Add(binarized);

ICSExtractor extractor = new ICSExtractor();
ExtractedPrint tmpl = extractor.Extract(fp, artifacts);

skeleton.GetImage().SaveAs("output/skeleton.png");
filtered.GetImage().SaveAs("output/filtered.png");
binarized.GetImage().SaveAs("output/binarized.png");
BinaryFile.WriteAll("output/fp.ics", tmpl.getIcsTemplateData());
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
Print fp = new Print(img, PrintPosition.LEFT_INDEX);

ICSExtractor.Artifacts artifacts = new ICSExtractor.Artifacts();
ICSExtractor.SkeletonImage skeleton = new ICSExtractor.SkeletonImage();
artifacts.Add(skeleton);
ICSExtractor.FilteredImage filtered = new ICSExtractor.FilteredImage();
artifacts.Add(filtered);
ICSExtractor.BinarizedImage binarized = new ICSExtractor.BinarizedImage();
artifacts.Add(binarized);

ICSExtractor extractor = new ICSExtractor();
ExtractedPrint tmpl = extractor.Extract(fp, artifacts);

skeleton.GetImage().SaveAs("output/skeleton.png");
filtered.GetImage().SaveAs("output/filtered.png");
binarized.GetImage().SaveAs("output/binarized.png");
BinaryFile.WriteAll("output/fp.ics", tmpl.icsTemplateData);
```

The output of example is template and required intermediate images.

Figure 74 Finger for extraction

Figure 75 Binarized

Figure 76 Filtered

Figure 77 Skeleton

3.10.2.5 1 to 1 Verification

We provide verification of ICS (Internal Innovatrics template) for Print via ICSVerifyMatcher. The main goal is to calculate Similarity Score for given probe and gallery templates. The examples uses following images:

Figure 78 Probe

Figure 79 Gallery

c++

```
auto img1 = Image::Decode(BinaryFile::ReadAll("assets/s1_li.png"));
auto p1 = Print::Create(std::move(img1), PrintPosition::LEFT_INDEX);
auto img2 = Image::Decode(BinaryFile::ReadAll("assets/s2_li.png"));
auto p2 = Print::Create(std::move(img2), PrintPosition::LEFT_INDEX);
ICSExtractor extractor;
auto tmpl1 = extractor.Extract(p1);
auto tmpl2 = extractor.Extract(p2);

auto cfg = ICSVerifyMatcher::Config::Default();
const ICSVerifyMatcher matcher(cfg);
auto score = matcher.Match(*tmpl1, *tmpl2);
std::cout « "   Similarity score == " « score « std::endl;
```

java

```
Image img_1 = Image.Decode(BinaryFile.ReadAll("assets/s1_li.png"));
Print p_1 = Print.Create(img_1, PrintPosition.LEFT_INDEX);
Image img_2 = Image.Decode(BinaryFile.ReadAll("assets/s2_li.png"));
Print p_2 = Print.Create(img_2, PrintPosition.LEFT_INDEX);
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint tmpl_1 = extractor.Extract(p_1);
ExtractedPrint tmpl_2 = extractor.Extract(p_2);

ICSVerifyMatcher matcher = new ICSVerifyMatcher(ICSVerifyMatcher.Config.Default());
long score = matcher.Match(tmpl_1, tmpl_2);
System.out.println("   Similarity score == " + score);
```

csharp

```
Image img_1 = Image.Decode(BinaryFile.ReadAll("assets/s1_li.png"));
Print p_1 = Print.Create(img_1, PrintPosition.LEFT_INDEX);
Image img_2 = Image.Decode(BinaryFile.ReadAll("assets/s2_li.png"));
Print p_2 = Print.Create(img_2, PrintPosition.LEFT_INDEX);
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint tmpl_1 = extractor.Extract(p_1);
ExtractedPrint tmpl_2 = extractor.Extract(p_2);

ICSVerifyMatcher matcher =
    new ICSVerifyMatcher(ICSVerifyMatcher.Config.Default());
uint score = matcher.Match(tmpl_1, tmpl_2);
System.Console.WriteLine("   Similarity score == " + score);
```

3.10.2.5.1 Matching Minutiae

During verification process SDK can provide matching minutiae between verified probe against gallery ICSTemplate. The matching minutia has same index in provided probe and gallery minutiae.

Note

The adjudicator feature MUST be allowed in license to obtain matching minutiae.

c++

```
auto img1 = Image::Decode(BinaryFile::ReadAll("assets/s1_li.png"));
auto p1 = Print::Create(std::move(img1), PrintPosition::LEFT_INDEX);
auto img2 = Image::Decode(BinaryFile::ReadAll("assets/s2_li.png"));
auto p2 = Print::Create(std::move(img2), PrintPosition::LEFT_INDEX);
ICSExtractor extractor;
auto probe = extractor.Extract(p1);
auto gallery = extractor.Extract(p2);

auto matchingMinutiae = ICSVerifyMatcher::MatchingMinutiae();
const ICSVerifyMatcher matcher;
std::ignore = matcher.Match(*probe, *gallery, matchingMinutiae);

auto p1Minutiae = probe->print->DrawMinutiae(matchingMinutiae.GetProbeMinutiae(),
                                             Shapes::InnoBlue);
auto p2Minutiae = gallery->print->DrawMinutiae(
  matchingMinutiae.GetGalleryMinutiae(), Shapes::InnoBlue);

p1Minutiae->SaveAs("output/s1_li_minutiae.png");
p2Minutiae->SaveAs("output/s2_li_minutiae.png");
```

java

```
Image img_1 = Image.Decode(BinaryFile.ReadAll("assets/s1_li.png"));
Print p_1 = Print.Create(img_1, PrintPosition.LEFT_INDEX);
Image img_2 = Image.Decode(BinaryFile.ReadAll("assets/s2_li.png"));
Print p_2 = Print.Create(img_2, PrintPosition.LEFT_INDEX);
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint probe = extractor.Extract(p_1);
ExtractedPrint gallery = extractor.Extract(p_2);

ICSVerifyMatcher.MatchingMinutiae matchingMinutiae = new ICSVerifyMatcher
                                                        .MatchingMinutiae();
ICSVerifyMatcher matcher = new ICSVerifyMatcher();
long score = matcher.Match(probe, gallery, matchingMinutiae);

Image p_1_minutiae = probe.getPrint().DrawMinutiae(
        matchingMinutiae.GetProbeMinutiae(), Shapes.InnoBlue);
Image p_2_minutiae = gallery.getPrint().DrawMinutiae(
        matchingMinutiae.GetGalleryMinutiae(), Shapes.InnoBlue);

p_1_minutiae.SaveAs("output/s1_li_minutiae.png");
p_2_minutiae.SaveAs("output/s2_li_minutiae.png");
```

csharp

```
Image img_1 = Image.Decode(BinaryFile.ReadAll("assets/s1_li.png"));
Print p_1 = Print.Create(img_1, PrintPosition.LEFT_INDEX);
Image img_2 = Image.Decode(BinaryFile.ReadAll("assets/s2_li.png"));
Print p_2 = Print.Create(img_2, PrintPosition.LEFT_INDEX);
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint probe = extractor.Extract(p_1);
ExtractedPrint gallery = extractor.Extract(p_2);

ICSVerifyMatcher.MatchingMinutiae matchingMinutiae =
    new ICSVerifyMatcher.MatchingMinutiae();
ICSVerifyMatcher matcher = new ICSVerifyMatcher();
uint score = matcher.Match(probe, gallery, matchingMinutiae);

Image p_1_minutiae = probe.print.DrawMinutiae(matchingMinutiae.GetProbeMinutiae(),
                                    Shapes.InnoBlue);
Image p_2_minutiae = gallery.print.DrawMinutiae(
    matchingMinutiae.GetGalleryMinutiae(), Shapes.InnoBlue);

p_1_minutiae.SaveAs("output/s1_li_minutiae.png");
p_2_minutiae.SaveAs("output/s2_li_minutiae.png");
```

Output of example are verified prints with drown matching minutiae.

3.10.2.6 Quality

Qualities provided for a fingerprint image and template:

- Template quality is calculated from a fingerprint template. The quality is an integer number between 0 (blank image) and 100 (maximum quality). For enrollment, it is recommended to use only templates with quality $>$ 30.

  c++

  ```
  ICSExtractor extractor;
  auto tmpl = extractor.Extract(
  ```

```
  Print::Create(Image::Decode(BinaryFile::ReadAll("assets/finger.png")),
                PrintPosition::LEFT_INDEX));
const uint8_t quality = tmpl->CalculateQuality();
std::cout « "  Template Quality == " « +quality « std::endl;
```

java

```
ICSExtractor extractor = new ICSExtractor();
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
ExtractedPrint tmpl = extractor.Extract(
        Print.Create(img, PrintPosition.LEFT_INDEX));
int quality = tmpl.CalculateQuality();
System.out.println("    Template Quality == " + quality);
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint tmpl =
    extractor.Extract(Print.Create(img, PrintPosition.LEFT_INDEX));
var quality = tmpl.CalculateQuality();
System.Console.WriteLine("  Template Quality == {0}", quality);
```

- Presence image quality number is calculated as an area where presence of fingerprint is detected. The calculated value does not measure quality of the presented fingerprint pattern. Presence value is calculated from the fingerprint image. The range is from 0 (lowest quality) to 100 (highest quality).

  c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/finger.png"));
auto fp = Print::Create(std::move(img), PrintPosition::LEFT_INDEX);
const uint8_t presence = fp->CalculatePresence();
std::cout « "   Presence quality == " « +presence « std::endl;
```

  java

```
// Read file content via provided File class
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
Print fp = Print.Create(img, PrintPosition.LEFT_INDEX);
byte presence = fp.CalculatePresence();
System.out.println("    Presence Quality == " + presence);
```

  csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
var fp = Print.Create(img, PrintPosition.LEFT_INDEX);
var presence = fp.CalculatePresence();
System.Console.WriteLine("  Presence Quality == {0}", presence);
```

- NFIQ image quality number is calculated in accordance with the general guidelines contained in Section 7.↩ 2.5 of ANSI/INCITS 381 standard (Section 8.3.7.3 Quality score of ISO/IEC 19794-4 international standard). Presence value is calculated from the fingerprint image. The range is from 1 (highest quality) to 5 (lowest quality).

  c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/finger.png"));
auto fp = Print::Create(std::move(img), PrintPosition::LEFT_INDEX);
const uint8_t nfiq1 = fp->CalculateNFIQ1();
std::cout « "   NFIQ quality == " « +nfiq1 « std::endl;
```

  java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
Print fp = Print.Create(img, PrintPosition.LEFT_INDEX);
byte nfiq1 = fp.CalculateNFIQ1();
System.out.println("    NFIQ1 Quality == " + nfiq1);
```

  csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
var fp = Print.Create(img, PrintPosition.LEFT_INDEX);
var nfiq1 = fp.CalculateNFIQ1();
System.Console.WriteLine("  NFIQ1 Quality == {0}", nfiq1);
```

- NFIQ2 image quality number is calculated in accordance with the general guidelines contained in Section 7.↩ 2.5 of ANSI/INCITS 381 standard (Section 8.3.7.3 Quality score of ISO/IEC 19794-4 international standard). Presence value is calculated from the fingerprint image. The range is from 0 (lowest quality) to 100 (highest quality).

  c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/finger.png"));
auto fp = Print::Create(std::move(img), PrintPosition::LEFT_INDEX);
const uint8_t nfiq2 = fp->CalculateNFIQ2();
std::cout « "   NFIQ2 quality == " « +nfiq2 « std::endl;
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
Print fp = Print.Create(img, PrintPosition.LEFT_INDEX);
byte nfiq2 = fp.CalculateNFIQ2();
System.out.println("    NFIQ2 Quality == " + nfiq2);
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
var fp = Print.Create(img, PrintPosition.LEFT_INDEX);
var nfiq2 = fp.CalculateNFIQ2();
System.Console.WriteLine("  NFIQ2 Quality == {0}", nfiq2);
```

- Image quality number is calculated in accordance with the general guidelines contained in Section 7.2.5 of ANSI/INCITS 381 standard Section 8.3.7.3 Quality score of ISO/IEC 19794-4 international standard). The range is from 0 (lowest quality) to 100 (highest quality).

c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/finger.png"));
auto fp = Print::Create(std::move(img), PrintPosition::LEFT_INDEX);
const uint8_t imageQuality = fp->CalculateImageQuality();
std::cout « "   Image quality == " « +imageQuality « std::endl;
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
Print fp = Print.Create(img, PrintPosition.LEFT_INDEX);
byte imageQuality = fp.CalculateImageQuality();
System.out.println("    Image Quality == " + imageQuality);
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
var fp = Print.Create(img, PrintPosition.LEFT_INDEX);
var imageQuality = fp.CalculateImageQuality();
System.Console.WriteLine("  Image Quality == {0}", imageQuality);
```

- Placement The placement score number ranges from 0 to 100 and represents the fitness of the finger placement, evaluated with respect to the detected fingerprint classification (arch, loop, whorl).

c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/finger.png"));
auto fp = Print::Create(std::move(img), PrintPosition::LEFT_INDEX);
const uint8_t placement = fp->CalculatePlacement();
std::cout « "   Placement quality == " « +placement « std::endl;
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
Print fp = Print.Create(img, PrintPosition.LEFT_INDEX);
byte placement = fp.CalculatePlacement();
System.out.println("    Placement Quality == " + placement);
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
var fp = Print.Create(img, PrintPosition.LEFT_INDEX);
var placement = fp.CalculatePlacement();
System.Console.WriteLine("  Placement Quality == {0}", placement);
```

- Intensity The intensity value is from 0 (lowest intensity, dry finger, low pressure) to 100 (highest intensity, wet finger, high pressure).

c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/finger.png"));
auto fp = Print::Create(std::move(img), PrintPosition::LEFT_INDEX);
const uint8_t intensity = fp->CalculateIntensity();
std::cout « "   Image intensity == " « +intensity « std::endl;
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
Print fp = Print.Create(img, PrintPosition.LEFT_INDEX);
byte intensity = fp.CalculateIntensity();
System.out.println("    Intensity Quality == " + intensity);
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
var fp = Print.Create(img, PrintPosition.LEFT_INDEX);
var intensity = fp.CalculateIntensity();
System.Console.WriteLine("  Image intensity == {0}", intensity);
```

3.10.2.7 Minutiae

The SDK provides extracted minutiae from ICSTemplate and draw them into Image.

- The Minutia::Type::BIFURCATION is drawn as square.
- The Minutia::Type::RIDGE_ENDING is drawn as circle.

When default minutiae shape does not fit, you can draw own via image functionality.

Note

The adjudicator feature MUST be allowed in license to obtain minutiae from template.

c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/finger.png"));
auto fp = Print::Create(std::move(img), PrintPosition::LEFT_INDEX);

ICSExtractor::SkeletonImage skeleton;
ICSExtractor extractor;
auto tmpl = extractor.Extract(fp, skeleton);
auto minutiae = tmpl->GetMinutiae();
auto skeletonWithMinutiae = skeleton.DrawMinutiae(minutiae, Shapes::InnoBlue);

skeletonWithMinutiae->SaveAs("output/skeleton_with_minutiae.png");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
Print fp = new Print(img, PrintPosition.LEFT_INDEX);

ICSExtractor.SkeletonImage skeleton = new ICSExtractor.SkeletonImage();
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint tmpl = extractor.Extract(fp, skeleton);
Minutiae minutiae = tmpl.GetMinutiae();
Image skeletonWithMinutiae = skeleton.DrawMinutiae(minutiae, Shapes.InnoBlue);

skeletonWithMinutiae.SaveAs("output/skeleton_with_minutiae.png");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
Print fp = new Print(img, PrintPosition.LEFT_INDEX);

ICSExtractor.SkeletonImage skeleton = new ICSExtractor.SkeletonImage();
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint tmpl = extractor.Extract(fp, skeleton);
Minutiae minutiae = tmpl.GetMinutiae();
Image skeletonWithMinutiae = skeleton.DrawMinutiae(minutiae, Shapes.InnoBlue);

skeletonWithMinutiae.SaveAs("output/skeleton_with_minutiae.png");
```

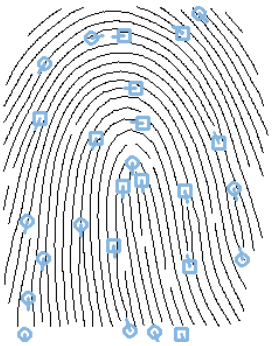


Figure 80 Minutiae

3.10.2.8 Critical Points

The SDK provides extracted core/deltas from ICSTemplate and draw them into Image.

- CriticalPoint::Type::CORE is drawn as circle.
- CriticalPoint::Type::DELTA is drawn as square.

When default critical point shape does not fit, you can draw own via image functionality.

Note

The adjudicator feature MUST be allowed in license to obtain critical points from template.

c++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/rolled_print.png"));
auto fp = Print::Create(std::move(img), PrintPosition::LEFT_INDEX);

ICSExtractor extractor;
auto tmpl = extractor.Extract(fp);
auto points = tmpl->GetCriticalPoints();
auto printWithPoints = fp->DrawCriticalPoints(points, Shapes::InnoBlue);

printWithPoints->SaveAs("output/rolled_print_with_critical_points.png");
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/rolled_print.png"));
Print fp = new Print(img, PrintPosition.LEFT_INDEX);

ICSExtractor extractor = new ICSExtractor();
ExtractedPrint tmpl = extractor.Extract(fp);
CriticalPoints points = tmpl.GetCriticalPoints();
Image printWithPoints = fp.DrawCriticalPoints(points, Shapes.InnoBlue);

printWithPoints.SaveAs("output/rolled_print_with_critical_points.png");
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/rolled_print.png"));
Print fp = new Print(img, PrintPosition.LEFT_INDEX);

ICSExtractor extractor = new ICSExtractor();
ExtractedPrint tmpl = extractor.Extract(fp);
CriticalPoints points = tmpl.GetCriticalPoints();
Image printWithPoints = fp.DrawCriticalPoints(points, Shapes.InnoBlue);

printWithPoints.SaveAs("output/rolled_print_with_critical_points.png");
```

Figure 81 Critical Points

3.10.2.9 Template Conversions

3.10.2.9.1 ISO Template

It is possible to convert proprietary ICSTemplate to ISOTemplate version 2 with PrintAttributes via function ExtractedPrint::ToISO().

c++

```
ICSExtractor extractor;
auto tmpl = extractor.Extract(
  Print::Create(Image::Decode(BinaryFile::ReadAll("assets/finger.png")),
                PrintPosition::LEFT_INDEX));
auto iso = tmpl->ToISO();
std::cout « "ISO template version: " « iso->GetVersion() « std::endl;
BinaryFile::WriteAll("output/template.iso", iso->data);
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint tmpl = extractor.Extract(
        Print.Create(img, PrintPosition.LEFT_INDEX));
ISOTemplate iso = tmpl.ToISO();
System.out.println("ISO template version: " + iso.GetVersion());
BinaryFile.WriteAll("output/template.iso", iso.getData());
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint tmpl =
    extractor.Extract(Print.Create(img, PrintPosition.LEFT_INDEX));
ISOTemplate iso = tmpl.ToISO();
Console.WriteLine("ISO template version: " + iso.GetVersion());
BinaryFile.WriteAll("output/template.iso", iso.data);
```

It is possible to convert ICSTemplate / ANSITemplate / ISOTemplate / ISONCTemplate / ISOCCTemplate /↩ ILOSIDTemplate to ISOTemplate version 2/3 via ISOTemplateAdapter. The attributes are copied from adapted template if they are present in adapted template.

c++

```
ICSExtractor extractor;
auto extractedPrint = extractor.Extract(
  Print::Create(Image::Decode(BinaryFile::ReadAll("assets/finger.png")),
                PrintPosition::LEFT_INDEX));

const ISOTemplateAdapter adapter(ISOTemplateAdapter::Version::VERSION_3);
// ICS to ISO v30
auto fromIcs = adapter.Adapt(*extractedPrint);
BinaryFile::WriteAll("output/template_from_ics.isov3", fromIcs.data);
// ANSI to ISO v30
auto fromAnsi = adapter.Adapt(*extractedPrint->ToANSI());
BinaryFile::WriteAll("output/template_from_ansi.isov3", fromAnsi.data);
// ISO to ISO v30
auto fromIso = adapter.Adapt(*extractedPrint->ToISO());
BinaryFile::WriteAll("output/template_from_iso.isov3", fromIso.data);
// ILOSID to ISO v30
const ILOSIDTemplateAdapter ilosidAdapter;
auto ilosid = ilosidAdapter.Adapt(*extractedPrint);
auto fromIlosid = adapter.Adapt(ilosid);
BinaryFile::WriteAll("output/template_from_ilosid.isov3", fromIlosid.data);
// ISO CC to ISO
const ISOCCTemplateAdapter isoccAdapter;
auto isocc = isoccAdapter.Adapt(*extractedPrint);
auto fromIsocc = adapter.Adapt(isocc);
BinaryFile::WriteAll("output/template_from_isocc.isov3", fromIsocc.data);
// ISO NC to ISO v30
const ISONCTemplateAdapter isoncAdapter;
auto isonc = isoncAdapter.Adapt(*extractedPrint);
auto fromIsonc = adapter.Adapt(isonc);
BinaryFile::WriteAll("output/template_from_isonc.isov3", fromIsonc.data);
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint extractedPrint = extractor.Extract(
```

```
        Print.Create(img, PrintPosition.LEFT_INDEX));

ISOTemplateAdapter adapter = new ISOTemplateAdapter(
        ISOTemplateAdapter.Version.VERSION_3);
// ICS to ISO v30
ISOTemplate fromIcs = adapter.Adapt(extractedPrint);
BinaryFile.WriteAll("output/template_from_ics.isov3", fromIcs.getData());
// ANSI to ISO v30
ISOTemplate fromAnsi = adapter.Adapt(extractedPrint.ToANSI());
BinaryFile.WriteAll("output/template_from_ansi.isov3", fromAnsi.getData());
// ISO to ISO v30
ISOTemplate fromIso = adapter.Adapt(extractedPrint.ToISO());
BinaryFile.WriteAll("output/template_from_iso.isov3", fromIso.getData());
// ILOSID to ISO v30
ILOSIDTemplateAdapter ilosidAdapter = new ILOSIDTemplateAdapter();
ILOSIDTemplate ilosid = ilosidAdapter.Adapt(extractedPrint);
ISOTemplate fromIlosid = adapter.Adapt(ilosid);
BinaryFile.WriteAll("output/template_from_ilosid.isov3", fromIlosid.getData());
// ISO CC to ISO v30
ISOCCTemplateAdapter isoccAdapter = new ISOCCTemplateAdapter();
ISOCCTemplate isocc = isoccAdapter.Adapt(extractedPrint);
ISOTemplate fromIsocc = adapter.Adapt(isocc);
BinaryFile.WriteAll("output/template_from_isocc.isov3", fromIsocc.getData());
// ISO NC to ISO v30
ISONCTemplateAdapter isoncAdapter = new ISONCTemplateAdapter();
ISONCTemplate isonc = isoncAdapter.Adapt(extractedPrint);
ISOTemplate fromIsonc = adapter.Adapt(isonc);
BinaryFile.WriteAll("output/template_from_isonc.isov3", fromIsonc.getData());
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
var extractor = new ICSExtractor();
var extractedPrint =
    extractor.Extract(Print.Create(img, PrintPosition.LEFT_INDEX));

var adapter = new ISOTemplateAdapter(ISOTemplateAdapter.Version.VERSION_3);
// ICS to ISO v30
var fromIcs = adapter.Adapt(extractedPrint);
BinaryFile.WriteAll("output/template_from_ics.isov3", fromIcs.data);
// ANSI to ISO v30
var fromAnsi = adapter.Adapt(extractedPrint.ToANSI());
BinaryFile.WriteAll("output/template_from_ansi.isov3", fromAnsi.data);
// ISO to ISO v30
var fromIso = adapter.Adapt(extractedPrint.ToISO());
BinaryFile.WriteAll("output/template_from_iso.isov3", fromIso.data);
// ILOSID to ISO v30
var ilosidAdapter = new ILOSIDTemplateAdapter();
var ilosid = ilosidAdapter.Adapt(extractedPrint);
var fromIlosid = adapter.Adapt(ilosid);
BinaryFile.WriteAll("output/template_from_ilosid.isov3", fromIlosid.data);
// ISO CC to ISO v30
var isoccAdapter = new ISOCCTemplateAdapter();
var isocc = isoccAdapter.Adapt(extractedPrint);
var fromIsocc = adapter.Adapt(isocc);
BinaryFile.WriteAll("output/template_from_isocc.isov3", fromIsocc.data);
// ISO NC to ISO v30
var isoncAdapter = new ISONCTemplateAdapter();
var isonc = isoncAdapter.Adapt(extractedPrint);
var fromIsonc = adapter.Adapt(isonc);
BinaryFile.WriteAll("output/template_from_isonc.isov3", fromIsonc.data);
```

3.10.2.9.2 ISO Image

It is possible to convert Print to ISO image compliant with ISO 19794-4:2011 and back from image to Print.

C++

```
auto img = Image::Decode(BinaryFile::ReadAll("assets/finger.png"));

PrintAttributes pa;
pa.SetImpression(PrintImpression::LIVE_SCAN_PLAIN);
pa.SetCaptureDeviceTechnology(
  PrintCaptureDeviceTechnology::SEMICONDUCTOR_CAPACITIVE);
pa.SetCaptureDeviceVendorID(PrintCaptureDeviceVendorID(11));
pa.SetCaptureDeviceTypeID(PrintCaptureDeviceTypeID(7));
pa.SetCertificationFlag(true);

auto fp = Print::Create(img, PrintPosition::RIGHT_THUMB, pa);

std::cout « "   Conversion Print to ISO image" « std::endl;
```

```
auto encoder = PngImageEncoder();
auto isoImage = fp->ToIsoImage(encoder);

std::cout « "   Conversion ISO to Print image" « std::endl;
auto printImage = fp->FromIsoImage(isoImage);
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));

PrintAttributes pa = new PrintAttributes();
pa.SetImpression(PrintImpression.LIVE_SCAN_PLAIN);
pa.SetCaptureDeviceTechnology(
        PrintCaptureDeviceTechnology.SEMICONDUCTOR_CAPACITIVE);
pa.SetCaptureDeviceVendorID(new PrintCaptureDeviceVendorID(11));
pa.SetCaptureDeviceTypeID(new PrintCaptureDeviceTypeID(7));
pa.SetCertificationFlag(true);

Print fp = Print.Create(img, PrintPosition.RIGHT_THUMB, pa);

System.out.println("   Conversion Print to ISO image");
PngImageEncoder encoder = new PngImageEncoder();
Bytes isoImage = fp.ToIsoImage(encoder);

System.out.println("   Conversion ISO to Print image");
Print printImage = Print.FromIsoImage(isoImage);
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));

PrintAttributes pa = new PrintAttributes();
pa.SetImpression(PrintImpression.LIVE_SCAN_PLAIN);
pa.SetCaptureDeviceTechnology(
    PrintCaptureDeviceTechnology.SEMICONDUCTOR_CAPACITIVE);
pa.SetCaptureDeviceVendorID(new PrintCaptureDeviceVendorID(11));
pa.SetCaptureDeviceTypeID(new PrintCaptureDeviceTypeID(7));
pa.SetCertificationFlag(true);

var fp = Print.Create(img, PrintPosition.RIGHT_THUMB, pa);

System.Console.WriteLine("   Conversion Print to ISO image");
var encoder = new PngImageEncoder();
var isoImage = fp.ToIsoImage(encoder);

System.Console.WriteLine("   Conversion ISO to Print image");
var printImage = Print.FromIsoImage(isoImage);
```

3.10.2.9.3 ANSI Template

It is possible to convert proprietary ICSTemplate to ANSITemplate with PrintAttributes via function ExtractedPrint::ToANSI().

c++

```
ICSExtractor extractor;
auto tmpl = extractor.Extract(
  Print::Create(Image::Decode(BinaryFile::ReadAll("assets/finger.png")),
                PrintPosition::LEFT_INDEX));
auto ansi = tmpl->ToANSI();
std::cout « "ANSI template version: " « ansi->GetVersion() « std::endl;
BinaryFile::WriteAll("output/template.ansi", ansi->data);
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint tmpl = extractor.Extract(
        Print.Create(img, PrintPosition.LEFT_INDEX));
ANSITemplate ansi = tmpl.ToANSI();
System.out.println("ANSI template version: " + ansi.GetVersion());
BinaryFile.WriteAll("output/template.ansi", ansi.getData());
```

csharp

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint tmpl =
    extractor.Extract(Print.Create(img, PrintPosition.LEFT_INDEX));
```

```
ANSITemplate ansi = tmpl.ToANSI();
Console.WriteLine("ANSI template version: " + ansi.GetVersion());
BinaryFile.WriteAll("output/template.ansi", ansi.data);
```

It is possible to convert ICSTemplate / ISOTemplate / ISONCTemplate / ISOCCTemplate / ILOSIDTemplate to ANSITemplate via ANSITemplateAdapter. The attributes are copied from adapted template if they are present in adapted template.

C++

```
ICSExtractor extractor;
auto extractedPrint = extractor.Extract(
  Print::Create(Image::Decode(BinaryFile::ReadAll("assets/finger.png")),
                PrintPosition::LEFT_INDEX));

const ANSITemplateAdapter adapter;
// ICS to ANSI
auto fromIcs = adapter.Adapt(*extractedPrint);
BinaryFile::WriteAll("output/template_from_ics.ansi", fromIcs.data);
// ISO to ANSI
auto fromIso = adapter.Adapt(*extractedPrint->ToISO());
BinaryFile::WriteAll("output/template_from_iso.ansi", fromIso.data);
// ILOSID to ANSI
const ILOSIDTemplateAdapter ilosidAdapter;
auto ilosid = ilosidAdapter.Adapt(*extractedPrint);
auto fromIlosid = adapter.Adapt(ilosid);
BinaryFile::WriteAll("output/template_from_ilosid.ansi", fromIlosid.data);
// ISO CC to ANSI
const ISOCCTemplateAdapter isoccAdapter;
auto isocc = isoccAdapter.Adapt(*extractedPrint);
auto fromIsocc = adapter.Adapt(isocc);
BinaryFile::WriteAll("output/template_from_isocc.ansi", fromIsocc.data);
// ISO NC to ANSI
const ISONCTemplateAdapter isoncAdapter;
auto isonc = isoncAdapter.Adapt(*extractedPrint);
auto fromIsonc = adapter.Adapt(isonc);
BinaryFile::WriteAll("output/template_from_isonc.ansi", fromIsonc.data);
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint extractedPrint = extractor.Extract(
        Print.Create(img, PrintPosition.LEFT_INDEX));

ANSITemplateAdapter adapter = new ANSITemplateAdapter();
// ICS to ANSI
ANSITemplate fromIcs = adapter.Adapt(extractedPrint);
BinaryFile.WriteAll("output/template_from_ics.ansi", fromIcs.getData());
// ISO to ANSI
ANSITemplate fromIso = adapter.Adapt(extractedPrint.ToISO());
BinaryFile.WriteAll("output/template_from_iso.ansi", fromIso.getData());
// ILOSID to ANSI
ILOSIDTemplateAdapter ilosidAdapter = new ILOSIDTemplateAdapter();
ILOSIDTemplate ilosid = ilosidAdapter.Adapt(extractedPrint);
ANSITemplate fromIlosid = adapter.Adapt(ilosid);
BinaryFile.WriteAll("output/template_from_ilosid.ansi", fromIlosid.getData());
// ISO CC to ANSI
ISOCCTemplateAdapter isoccAdapter = new ISOCCTemplateAdapter();
ISOCCTemplate isocc = isoccAdapter.Adapt(extractedPrint);
ANSITemplate fromIsocc = adapter.Adapt(isocc);
BinaryFile.WriteAll("output/template_from_isocc.ansi", fromIsocc.getData());
// ISO NC to ANSI
ISONCTemplateAdapter isoncAdapter = new ISONCTemplateAdapter();
ISONCTemplate isonc = isoncAdapter.Adapt(extractedPrint);
ANSITemplate fromIsonc = adapter.Adapt(isonc);
BinaryFile.WriteAll("output/template_from_isonc.ansi", fromIsonc.getData());
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
var extractor = new ICSExtractor();
var extractedPrint =
    extractor.Extract(Print.Create(img, PrintPosition.LEFT_INDEX));

var adapter = new ANSITemplateAdapter();
// ICS to ANSI
var fromIcs = adapter.Adapt(extractedPrint);
BinaryFile.WriteAll("output/template_from_ics.ansi", fromIcs.data);
// ISO to ANSI
var fromIso = adapter.Adapt(extractedPrint.ToISO());
BinaryFile.WriteAll("output/template_from_iso.ansi", fromIso.data);
// ILOSID to ANSI
var ilosidAdapter = new ILOSIDTemplateAdapter();
```

```
var ilosid = ilosidAdapter.Adapt(extractedPrint);
var fromIlosid = adapter.Adapt(ilosid);
BinaryFile.WriteAll("output/template_from_ilosid.ansi", fromIlosid.data);
// ISO CC to ANSI
var isoccAdapter = new ISOCCTemplateAdapter();
var isocc = isoccAdapter.Adapt(extractedPrint);
var fromIsocc = adapter.Adapt(isocc);
BinaryFile.WriteAll("output/template_from_isocc.ansi", fromIsocc.data);
// ISO NC to ANSI
var isoncAdapter = new ISONCTemplateAdapter();
var isonc = isoncAdapter.Adapt(extractedPrint);
var fromIsonc = adapter.Adapt(isonc);
BinaryFile.WriteAll("output/template_from_isonc.ansi", fromIsonc.data);
```

3.10.2.9.4 ILO SID Template

International Labour Organisation Seafarers Identity Document template.

It is possible to convert ICSTemplate / ANSITemplate / ISOTemplate / ISONCTemplate / ISOCCTemplate to ILOSIDTemplate via ILOSIDTemplateAdapter.

C++

```
ICSExtractor extractor;
auto extractedPrint = extractor.Extract(
  Print::Create(Image::Decode(BinaryFile::ReadAll("assets/finger.png")),
                PrintPosition::LEFT_INDEX));

const ILOSIDTemplateAdapter adapter;
// ICS to ILO SID
auto fromIcs = adapter.Adapt(*extractedPrint);
BinaryFile::WriteAll("output/template_from_ics.ilosid", fromIcs.GetData());
// ANSI to ILO SID
auto fromAnsi = adapter.Adapt(*extractedPrint->ToANSI());
BinaryFile::WriteAll("output/template_from_ansi.ilosid", fromAnsi.GetData());
// ISO to ILO SID
auto fromIso = adapter.Adapt(*extractedPrint->ToISO());
BinaryFile::WriteAll("output/template_from_iso.ilosid", fromIso.GetData());
// ISO NC to ILO SID
const ISONCTemplateAdapter isoncAdapter;
auto isonc = isoncAdapter.Adapt(*extractedPrint);
auto fromIsonc = adapter.Adapt(isonc);
BinaryFile::WriteAll("output/template_from_isonc.ilosid", fromIsonc.GetData());
// ISO CC to ILO SID
const ISOCCTemplateAdapter isoccAdapter;
auto isocc = isoccAdapter.Adapt(*extractedPrint);
auto fromIsocc = adapter.Adapt(isocc);
BinaryFile::WriteAll("output/template_from_isocc.ilosid", fromIsocc.GetData());
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint extractedPrint = extractor.Extract(
        Print.Create(img, PrintPosition.LEFT_INDEX));

ILOSIDTemplateAdapter adapter = new ILOSIDTemplateAdapter();
// ICS to ILO SID
ILOSIDTemplate fromIcs = adapter.Adapt(extractedPrint);
BinaryFile.WriteAll("output/template_from_ics.ilosid", fromIcs.GetData());
// ANSI to ILO SID
ILOSIDTemplate fromAnsi = adapter.Adapt(extractedPrint.ToANSI());
BinaryFile.WriteAll("output/template_from_ansi.ilosid", fromAnsi.GetData());
// ISO to ILO SID
ILOSIDTemplate fromIso = adapter.Adapt(extractedPrint.ToISO());
BinaryFile.WriteAll("output/template_from_iso.ilosid", fromIso.GetData());
// ISO NC to ILO SID
ISONCTemplateAdapter isoncAdapter = new ISONCTemplateAdapter();
ISONCTemplate isonc = isoncAdapter.Adapt(extractedPrint);
ILOSIDTemplate fromIlosid = adapter.Adapt(isonc);
BinaryFile.WriteAll("output/template_from_isonc.ilosid", fromIlosid.GetData());
// ISO CC to ILOSID
ISOCCTemplateAdapter isoccAdapter = new ISOCCTemplateAdapter();
ISOCCTemplate isocc = isoccAdapter.Adapt(extractedPrint);
ILOSIDTemplate fromIsocc = adapter.Adapt(isocc);
BinaryFile.WriteAll("output/template_from_isocc.ilosid", fromIsocc.GetData());
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
var extractor = new ICSExtractor();
var extractedPrint =
    extractor.Extract(Print.Create(img, PrintPosition.LEFT_INDEX));

var adapter = new ILOSIDTemplateAdapter();
// ICS to ILO SID
var fromIcs = adapter.Adapt(extractedPrint);
BinaryFile.WriteAll("output/template_from_ics.ilosid", fromIcs.GetData());
// ANSI to ILO SID
var fromAnsi = adapter.Adapt(extractedPrint.ToANSI());
BinaryFile.WriteAll("output/template_from_ansi.ilosid", fromAnsi.GetData());
// ISO to ILO SID
var fromIso = adapter.Adapt(extractedPrint.ToISO());
BinaryFile.WriteAll("output/template_from_iso.ilosid", fromIso.GetData());
// ISO NC to ILO SID
var isoncAdapter = new ISONCTemplateAdapter();
var isonc = isoncAdapter.Adapt(extractedPrint);
var fromIsonc = adapter.Adapt(isonc);
BinaryFile.WriteAll("output/template_from_isonc.ilosid", fromIsonc.GetData());
// ISO CC to ILO SID
var isoccAdapter = new ISOCCTemplateAdapter();
var isocc = isoccAdapter.Adapt(extractedPrint);
var fromIsocc = adapter.Adapt(isocc);
BinaryFile.WriteAll("output/template_from_isocc.ilosid", fromIsocc.GetData());
```

3.10.2.9.5 ISO NC Template

ISO/IEC 19794-2:2005 template, template version 2.0, 5-byte format for minutiae (normal card encoding).

It is possible to convert ICSTemplate / ANSITemplate / ISOTemplate / ILOSIDTemplate / ISOCCTemplate to ISONCTemplate via ISONCTemplateAdapter.

C++

```
ICSExtractor extractor;
auto extractedPrint = extractor.Extract(
  Print::Create(Image::Decode(BinaryFile::ReadAll("assets/finger.png")),
                PrintPosition::LEFT_INDEX));

const ISONCTemplateAdapter adapter;
// ICS to ISO NC
auto fromIcs = adapter.Adapt(*extractedPrint);
BinaryFile::WriteAll("output/template_from_ics.isonc", fromIcs.GetData());
// ANSI to ISO NC
auto fromAnsi = adapter.Adapt(*extractedPrint->ToANSI());
BinaryFile::WriteAll("output/template_from_ansi.isonc", fromAnsi.GetData());
// ISO to ISO NC
auto fromIso = adapter.Adapt(*extractedPrint->ToISO());
BinaryFile::WriteAll("output/template_from_iso.isonc", fromIso.GetData());
// ILOSID to ISO NC
const ILOSIDTemplateAdapter ilosidAdapter;
auto ilosid = ilosidAdapter.Adapt(*extractedPrint);
auto fromIlosid = adapter.Adapt(ilosid);
BinaryFile::WriteAll("output/template_from_ilosid.isonc", fromIlosid.GetData());
// ISO CC to ISO NC
const ISOCCTemplateAdapter isoccAdapter;
auto isocc = isoccAdapter.Adapt(*extractedPrint);
auto fromIsocc = adapter.Adapt(isocc);
BinaryFile::WriteAll("output/template_from_isocc.isonc", fromIsocc.GetData());
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint extractedPrint = extractor.Extract(
        Print.Create(img, PrintPosition.LEFT_INDEX));

ISONCTemplateAdapter adapter = new ISONCTemplateAdapter();
// ICS to ISO NC
ISONCTemplate fromIcs = adapter.Adapt(extractedPrint);
BinaryFile.WriteAll("output/template_from_ics.isonc", fromIcs.GetData());
// ANSI to ISO NC
ISONCTemplate fromAnsi = adapter.Adapt(extractedPrint.ToANSI());
BinaryFile.WriteAll("output/template_from_ansi.isonc", fromAnsi.GetData());
// ISO to ISO NC
ISONCTemplate fromIso = adapter.Adapt(extractedPrint.ToISO());
BinaryFile.WriteAll("output/template_from_iso.isonc", fromIso.GetData());
// ILOSID to ISO NC
ILOSIDTemplateAdapter ilosidAdapter = new ILOSIDTemplateAdapter();
```

```
ILOSIDTemplate ilosid = ilosidAdapter.Adapt(extractedPrint);
ISONCTemplate fromIlosid = adapter.Adapt(ilosid);
BinaryFile.WriteAll("output/template_from_ilosid.isonc", fromIlosid.GetData());
// ISO CC to ISO NC
ISOCCTemplateAdapter isoccAdapter = new ISOCCTemplateAdapter();
ISOCCTemplate isocc = isoccAdapter.Adapt(extractedPrint);
ISONCTemplate fromIsocc = adapter.Adapt(isocc);
BinaryFile.WriteAll("output/template_from_isocc.isonc", fromIsocc.GetData());
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
var extractor = new ICSExtractor();
var extractedPrint =
    extractor.Extract(Print.Create(img, PrintPosition.LEFT_INDEX));

var adapter = new ISONCTemplateAdapter();
// ICS to ISO NC
var fromIcs = adapter.Adapt(extractedPrint);
BinaryFile.WriteAll("output/template_from_ics.isonc", fromIcs.GetData());
// ANSI to ISO NC
var fromAnsi = adapter.Adapt(extractedPrint.ToANSI());
BinaryFile.WriteAll("output/template_from_ansi.isonc", fromAnsi.GetData());
// ISO to ISO NC
var fromIso = adapter.Adapt(extractedPrint.ToISO());
BinaryFile.WriteAll("output/template_from_iso.isonc", fromIso.GetData());
// ILOSID to ISO NC
var ilosidAdapter = new ILOSIDTemplateAdapter();
var ilosid = ilosidAdapter.Adapt(extractedPrint);
var fromIlosid = adapter.Adapt(ilosid);
BinaryFile.WriteAll("output/template_from_ilosid.isonc", fromIlosid.GetData());
// ISO CC to ISO NC
var isoccAdapter = new ISOCCTemplateAdapter();
var isocc = isoccAdapter.Adapt(extractedPrint);
var fromIsocc = adapter.Adapt(isocc);
BinaryFile.WriteAll("output/template_from_isocc.isonc", fromIsocc.GetData());
```

3.10.2.9.6 ISO CC Template

ISO/IEC 19794-2:2005 template, template version 2.0, 3-byte format for minutiae (compact card encoding).

It is possible to convert ICSTemplate / ANSITemplate / ISOTemplate / ILOSIDTemplate / ISONCTemplate to ISOCCTemplate via ISOCCTemplateAdapter.

c++

```
ICSExtractor extractor;
auto extractedPrint = extractor.Extract(
  Print::Create(Image::Decode(BinaryFile::ReadAll("assets/finger.png")),
                PrintPosition::LEFT_INDEX));

const ISOCCTemplateAdapter adapter;
// ICS to ISO CC
auto fromIcs = adapter.Adapt(*extractedPrint);
BinaryFile::WriteAll("output/template_from_ics.isocc", fromIcs.GetData());
// ANSI to ISO CC
auto fromAnsi = adapter.Adapt(*extractedPrint->ToANSI());
BinaryFile::WriteAll("output/template_from_ansi.isocc", fromAnsi.GetData());
// ISO to ISO CC
auto fromIso = adapter.Adapt(*extractedPrint->ToISO());
BinaryFile::WriteAll("output/template_from_iso.isocc", fromIso.GetData());
// ILOSID to ISO CC
const ILOSIDTemplateAdapter ilosidAdapter;
auto ilosid = ilosidAdapter.Adapt(*extractedPrint);
auto fromIlosid = adapter.Adapt(ilosid);
BinaryFile::WriteAll("output/template_from_ilosid.isocc", fromIlosid.GetData());
// ISO NC to ISO CC
const ISONCTemplateAdapter isoncAdapter;
auto isonc = isoncAdapter.Adapt(*extractedPrint);
auto fromIsonc = adapter.Adapt(isonc);
BinaryFile::WriteAll("output/template_from_isonc.isocc", fromIsonc.GetData());
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint extractedPrint = extractor.Extract(
        Print.Create(img, PrintPosition.LEFT_INDEX));

ISOCCTemplateAdapter adapter = new ISOCCTemplateAdapter();
```

```
// ICS to ISO CC
ISOCCTemplate fromIcs = adapter.Adapt(extractedPrint);
BinaryFile.WriteAll("output/template_from_ics.isocc", fromIcs.GetData());
// ANSI to ISO CC
ISOCCTemplate fromAnsi = adapter.Adapt(extractedPrint.ToANSI());
BinaryFile.WriteAll("output/template_from_ansi.isocc", fromAnsi.GetData());
// ISO to ISO CC
ISOCCTemplate fromIso = adapter.Adapt(extractedPrint.ToISO());
BinaryFile.WriteAll("output/template_from_iso.isocc", fromIso.GetData());
// ILOSID to ISO CC
ILOSIDTemplateAdapter ilosidAdapter = new ILOSIDTemplateAdapter();
ILOSIDTemplate ilosid = ilosidAdapter.Adapt(extractedPrint);
ISOCCTemplate fromIlosid = adapter.Adapt(ilosid);
BinaryFile.WriteAll("output/template_from_ilosid.isocc", fromIlosid.GetData());
// ISO NC to ISO CC
ISONCTemplateAdapter isoncAdapter = new ISONCTemplateAdapter();
ISONCTemplate isonc = isoncAdapter.Adapt(extractedPrint);
ISOCCTemplate fromIsonc = adapter.Adapt(isonc);
BinaryFile.WriteAll("output/template_from_isonc.isocc", fromIsonc.GetData());
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
var extractor = new ICSExtractor();
var extractedPrint =
    extractor.Extract(Print.Create(img, PrintPosition.LEFT_INDEX));

var adapter = new ISOCCTemplateAdapter();
// ICS to ISO CC
var fromIcs = adapter.Adapt(extractedPrint);
BinaryFile.WriteAll("output/template_from_ics.isocc", fromIcs.GetData());
// ANSI to ISO CC
var fromAnsi = adapter.Adapt(extractedPrint.ToANSI());
BinaryFile.WriteAll("output/template_from_ansi.isocc", fromAnsi.GetData());
// ISO to ISO CC
var fromIso = adapter.Adapt(extractedPrint.ToISO());
BinaryFile.WriteAll("output/template_from_iso.isocc", fromIso.GetData());
// ILOSID to ISO CC
var ilosidAdapter = new ILOSIDTemplateAdapter();
var ilosid = ilosidAdapter.Adapt(extractedPrint);
var fromIlosid = adapter.Adapt(ilosid);
BinaryFile.WriteAll("output/template_from_ilosid.isocc", fromIlosid.GetData());
// ISO NC to ISO CC
var isoncAdapter = new ISONCTemplateAdapter();
var isonc = isoncAdapter.Adapt(extractedPrint);
var fromIsonc = adapter.Adapt(isonc);
BinaryFile.WriteAll("output/template_from_isonc.isocc", fromIsonc.GetData());
```

It is possible to limit and modify minutiae order in ISOCCTemplate.

c++

```
ICSExtractor extractor;
auto extractedPrint = extractor.Extract(
  Print::Create(Image::Decode(BinaryFile::ReadAll("assets/finger.png")),
                PrintPosition::LEFT_INDEX));

const ISOCCTemplateAdapter adapter(5,
                                   ISOCCTemplateAdapter::MinutiaSortOrder::X_ASC,
                                   ISOCCTemplateAdapter::MinutiaSortOrder::NONE);
// ICS to Limited ISO CC
auto fromIcs = adapter.Adapt(*extractedPrint);
BinaryFile::WriteAll("output/template_from_ics.limited_isocc", fromIcs.GetData());
```

java

```
Image img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
ICSExtractor extractor = new ICSExtractor();
ExtractedPrint extractedPrint = extractor.Extract(
        Print.Create(img, PrintPosition.LEFT_INDEX));

ISOCCTemplateAdapter adapter = new ISOCCTemplateAdapter(5,
        ISOCCTemplateAdapter.MinutiaSortOrder.X_ASC,
        ISOCCTemplateAdapter.MinutiaSortOrder.NONE);

// ICS to Limited ISO CC
ISOCCTemplate fromIcs = adapter.Adapt(extractedPrint);
BinaryFile.WriteAll("output/template_from_ics.limited_isocc", fromIcs.GetData());
```

csharp

```
var img = Image.Decode(BinaryFile.ReadAll("assets/finger.png"));
var extractor = new ICSExtractor();
var extractedPrint =
```

```
    extractor.Extract(Print.Create(img, PrintPosition.LEFT_INDEX));

var adapter =
    new ISOCCTemplateAdapter(5, ISOCCTemplateAdapter.MinutiaSortOrder.X_ASC,
                             ISOCCTemplateAdapter.MinutiaSortOrder.NONE);
// ICS to Limited ISO CC
var fromIcs = adapter.Adapt(extractedPrint);
BinaryFile.WriteAll("output/template_from_ics.limited_isocc", fromIcs.GetData());
```

3.10.2.10 Load template

It is possible to load previously created template to make e.g. verification against enrolling Applicant.

c++

```
auto tmpl = ICSTemplate(BinaryFile::ReadAll("assets/template.ics"));
std::cout « "   ICS template version: " « tmpl.GetVersion() « std::endl;
// Template was exported from IDKit with function IEngine_ExportUserTemplate with
// FORMAT_ICS and contains only one fingerprint
auto tmplFromIDkit =
  ICSTemplate(BinaryFile::ReadAll("assets/idkit_template.ics"));
std::cout « "   IDKit ICS template version: " « tmplFromIDkit.GetVersion()
          « std::endl;
```

java

```
ICSTemplate tmpl = new ICSTemplate(BinaryFile.ReadAll("assets/template.ics"));
System.out.println("   ICS template version: " + tmpl.GetVersion());
// Template was exported from IDKit with function IEngine_ExportUserTemplate with
// FORMAT_ICS and contains only one fingerprint
ICSTemplate tmplFromIDkit = new ICSTemplate(
        BinaryFile.ReadAll("assets/idkit_template.ics"));
System.out.println("   IDKit ICS template version: " + tmplFromIDkit.GetVersion());
```

csharp

```
ICSTemplate tmpl = new ICSTemplate(BinaryFile.ReadAll("assets/template.ics"));
Console.WriteLine("   ICS template version: " + tmpl.GetVersion());
// Template was exported from IDKit with function IEngine_ExportUserTemplate with
// FORMAT_ICS and contains only one fingerprint
ICSTemplate tmplFromIDkit =
    new ICSTemplate(BinaryFile.ReadAll("assets/idkit_template.ics"));
Console.WriteLine("   IDKit ICS template version: " + tmplFromIDkit.GetVersion());
```

3.10.3 Typedef Documentation

3.10.3.1 PrintSimilarityScore

```
using PrintSimilarityScore = unsigned int
```

Score is 0-1000 where 1000 represents most similar templates.

The correspondence between print templates is generally reflected by similarity scores greater than default similarity threshold (40). For more information please refer to FAR and Threshold (see False Acceptance Rate). Default similarity threshold value (40) roughly corresponds to FAR=10^-4. Depending on the type of scanner, the size of print images, the size of the database and the desired security level of your application, you may need to select a different threshold.

See Similarity Score.

3.10.4 Enumeration Type Documentation

3.10.4.1 PrintCaptureDeviceTechnology

```
enum class PrintCaptureDeviceTechnology : uint8_t [strong]
```

Enumerator

UNKNOWN	Unknown or unspecified.
WHITE_LIGHT_OPTICAL_TIR	White light optical TIR.
WHITE_LIGHT_OPTICAL_DIRECT	White light optical direct view on platen. Card scanner should encode their technology type as white light optical direct view on platen.
WHITE_LIGHT_OPTICAL_TOUCHLESS	White light optical touchless.
MONOCHROMATIC_VISIBLE_OPTICAL_TIR	Monochromatic visible optical TIR.
MONOCHROMATIC_VISIBLE_OPTICAL_DIRECT	Monochromatic visible optical direct view on platen.
MONOCHROMATIC_VISIBLE_OPTICAL_TOUCHLESS	Monochromatic visible optical touchless.
MONOCHROMATIC_VISIBLE_IR_OPTICAL_TIR	Monochromatic IR optical TIR.
MONOCHROMATIC_VISIBLE_IR_OPTICALTOUCHLESS	Monochromatic IR optical direct view on platen.
MONOCHROMATIC_VISIBLE_IR_OPTICAL	Monochromatic IR optical touchless.
MULTISPECTRAL_OPTICAL_TIR	Multispectral optical TIR.
MULTISPECTRAL_OPTICAL_DIRECT	Multispectral optical direct view on platen.
MULTISPECTRAL_OPTICAL_TOUCHLESS	Multispectral optical touchless.
ELECTRO_LUMINESCENT	Electro luminescent.
SEMICONDUCTOR_CAPACITIVE	Semiconductor capacitive.
SEMICONDUCTOR_RF	Semiconductor RF.
SEMICONDUCTOR_THERMAL	Semiconductor thermal.
PRESSURE_SENSITIVE	Pressure sensitive.
ULTRASOUND	Ultrasound.
MECHANICAL	Mechanical.
GLASS_FIBER	Glass fiber.

3.10.4.2 PrintImpression

```
enum class PrintImpression : uint8_t [strong]
```

Depending on the acquisition methodology, fingerprints can be classified in three types:

- rolled
- plain
- latent

Rolled

A fingerprint image collected by rolling the finger across a livescan platen or paper fingerprint card from nail to nail. Rolls may be from livescan devices or scanned from paper fingerprint cards

Plain

A fingerprint image resulting from the touching of one or more fingers to a livescan platen or paper fingerprint card without any rolling motion

Latent

An impression or image of friction ridge skin left on a surface. Latent impressions are only partial fingerprints and normally have poor quality and strong distortions

Nonlive

Nonlive entries refer to images scanned from cards or other media.

Live-scan

Live-scan contactless refers to image capture devices that do not depend upon the surface of a finger making contact with a scanner platen.

Enumerator

LIVE_SCAN_PLAIN	Live-scan plain.
LIVE_SCAN_ROLLED	Live-scan rolled.
NONLIVE_SCAN_PLAIN	Nonlive-scan plain.
NONLIVE_SCAN_ROLLED	Nonlive-scan rolled.
LATENT_IMPRESSION	Latent impression.
LATENT_TRACING	Latent tracing.
LATENT_PHOTO	Latent photo.
LATENT_LIFT	Latent lift.
LIVE_SCAN_SWIPE	Live-scan swipe.
LIVE_SCAN_VERTICAL_ROLL	Live-scan vertical roll.
LIVE_SCAN_PALM	Live-scan palm.
NONLIVE_SCAN_PALM	Nonlive-scan palm.
LATENT_PALM_IMPRESSION	Latent palm impression.
LATENT_PALM_TRACING	Latent palm tracing.
LATENT_PALM_PHOTO	Latent palm photo.
LATENT_PALM_LIFT	Latent palm lift.
LIVE_SCAN_OPTICAL_CONCTACTLESS_PLAIN	Live-scan optical contactless plain.
OTHER	Other.
UNKNOWN	Unknown.

3.10.4.3 PrintPosition

```
enum class PrintPosition : uint8_t [strong]
```

The library supports conversion to and from ISO positions [5] .

C++

```
auto isoPosition = PrintToIsoPosition(PrintPosition::LEFT_FULL_PALM);
auto printPosition = IsoToPrintPosition(isoPosition);

std::cout « "   Print to ISO position " « PrintPosition::LEFT_FULL_PALM
          « " -> " « isoPosition « std::endl;
std::cout « "   ISO to print position " « isoPosition « " -> " « printPosition
          « std::endl;
```

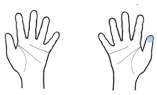
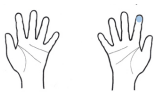
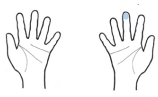
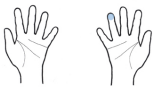
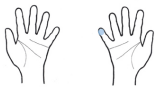
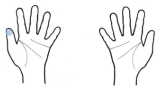
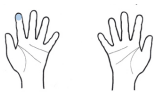
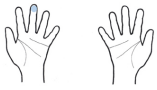
java

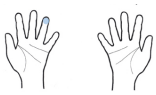
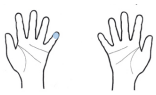
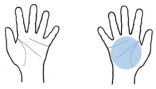
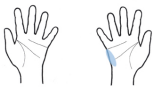
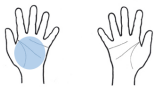
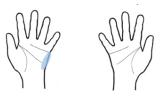
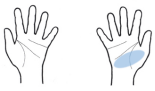
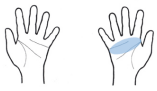
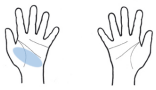
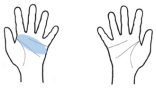
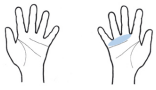
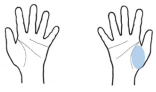
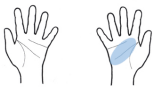
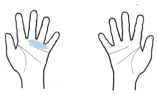
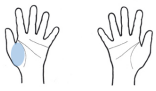
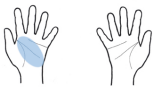
```
int isoPosition = enrollment.PrintToIsoPosition(PrintPosition.LEFT_FULL_PALM);
PrintPosition printPosition = enrollment.IsoToPrintPosition(isoPosition);

System.out.println("   Print to ISO position " + PrintPosition.LEFT_FULL_PALM
                   + " -> " + isoPosition);
System.out.println("   ISO to print position " + isoPosition + " -> "
                   + printPosition);
```

csharp

```
var isoPosition = enrollment.PrintToIsoPosition(PrintPosition.LEFT_FULL_PALM);
var printPosition = enrollment.IsoToPrintPosition(isoPosition);

System.Console.WriteLine("   Print to ISO position " +
                         PrintPosition.LEFT_FULL_PALM + " -> " + isoPosition);
System.Console.WriteLine("   ISO to print position " + isoPosition + " -> " +
                         printPosition);
```

Enumerator

RIGHT_THUMB	Thumb on right hand. 
RIGHT_INDEX	Index on right hand. 
RIGHT_MIDDLE	Middle on right hand. 
RIGHT_RING	Ring on right hand. 
RIGHT_LITTLE	Little on right hand. 
LEFT_THUMB	Thumb on left hand. 
LEFT_INDEX	Index on left hand. 
LEFT_MIDDLE	Middle on left hand. 

LEFT_RING	Ring on left hand. 
LEFT_LITTLE	Little finger on left hand. 
RIGHT_FULL_PALM	Right full palm. 
RIGHT_WRITERS_PALM	Right writer palm. 
LEFT_FULL_PALM	Left full palm. 
LEFT_WRITERS_PALM	Left writer palm. 
RIGHT_LOWER_PALM	Right lower palm. 
RIGHT_UPPER_PALM	Right upper palm. 
LEFT_LOWER_PALM	Left lower palm. 
LEFT_UPPER_PALM	Left upper palm. 
RIGHT_INTERDIGITAL	Right interdigital palm. 
RIGHT_THENAR	Right thenar palm. 
RIGHT_HYPOTHENAR	Right hypo thenar palm. 
LEFT_INTERDIGITAL	Left interdigital palm. 
LEFT_THENAR	Left thenar palm. 
LEFT_HYPOTHENAR	Left hypo thenar palm. 

3.10.5 Function Documentation

3.10.5.1 IsoToPrintCaptureDeviceTechnology()

```
constexpr PrintCaptureDeviceTechnology IsoToPrintCaptureDeviceTechnology (
            int isoCaptureDeviceTechnology) [static], [constexpr]
```

Parameters

isoCaptureDeviceTechnology	iso capture technology

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

PrintCaptureDeviceTechnology

3.10.5.2 IsoToPrintImpression()

```
constexpr PrintImpression IsoToPrintImpression (
            int isoImpression) [static], [constexpr]
```

Parameters

isoImpression	iso impression

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

PrintImpression

3.10.5.3 IsoToPrintPosition()

```
constexpr PrintPosition IsoToPrintPosition (
            int isoPosition) [static], [constexpr]
```

Parameters

isoPosition	iso position

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

PrintPosition

3.10.5.4 PrintToIsoCaptureDeviceTechnology()

```
constexpr int PrintToIsoCaptureDeviceTechnology (
            PrintCaptureDeviceTechnology printCaptureDeviceTechnology) [static], [constexpr]
```

Parameters

printCaptureDeviceTechnology	PrintCaptureDeviceTechnology

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

ISO capture technology

3.10.5.5 PrintToIsoImpression()

```
constexpr int PrintToIsoImpression (
            PrintImpression printImpression) [static], [constexpr]
```

Parameters

printImpression	PrintImpression

Exceptions

EnrollmentInvalidArgumentException	for invalid conversion

Returns

ISO impression

3.10.5.6 PrintToIsoPosition()

```
constexpr int PrintToIsoPosition (
            PrintPosition position)  [static], [constexpr]
```

Parameters

position	PrintPosition

Exceptions

EnrollmentInvalidArgumentException	when conversion is not possible

Returns

ISO Position

3.11 Serialization

The serialization is used to transform internal classes into user defined output.

Classes

- class Base64Strategy

  It writes bytes represented as std::vector<uint8_t> as base64 coded string.
- class JsonWriter

  JSON implementation of Writer abstraction.
- class Serializable

  Interface that MUST be implemented by all classes those want to be serialized.
- class BytesWriterStrategy

  It is used by Writer class to modify bytes serialization algorithm.
- class Writer

  Provides interface for serialization of all classes.

3.11.1 Detailed Description

The serialization is used to transform internal classes into user defined output.

The library serialize itself via Writer, Serializable and BytesWriterStrategy interfaces.

Interfaces Writer,BytesWriterStrategy are defined for client extension. When client needs own serialization then client implements Writer interface and provide it as Writer for library classes. When client needs to modify only bytes serialization, client implements BytesWriterStrategy interface and sets it into Writer. The library itself provides JSON implementation in class JsonWriter. Each class in library that can be serialized must inherit Serializable interface.

Use built in JSON

c++

```
static void UseJson()
{
    auto fp = CreatePrint();
    JsonWriter writer;
    writer.WriteRoot(*fp);
    // print serialized output
    std::cout « "    " « writer.Output() « std::endl;
}
```

java

```
public void useJson() {
    Print fp = createPrint();
    JsonWriter writer = new JsonWriter();
    writer.WriteRoot(fp);
    // print serialized output
    System.out.println("    " + writer.Output());
}
```

csharp

```
public void useJson()
{
    Print fp = createPrint();
    JsonWriter writer = new JsonWriter();
    writer.WriteRoot(fp);
    // print serialized output
    System.Console.WriteLine("    " + writer.Output());
}
```

Use built in JSON with Base64 encoded bytes

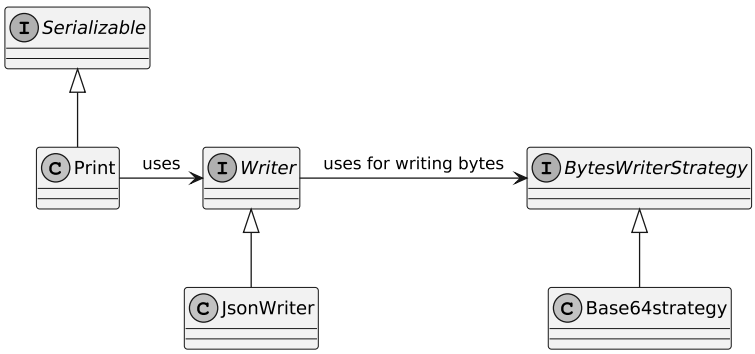


Figure 82 Use built in JSON with Base64 encoded bytes

c++

```
void UseJsonWithBase64()
{
    auto fp = CreatePrint();
    JsonWriter writer;
    Base64Strategy base64;
    writer.SetBytesStrategy(base64);
    writer.WriteRoot(*fp);
    // print serialized output
    std::cout « "    " « writer.Output() « std::endl;
}
```

java

```
public void useJsonWithBase64() {
    Print fp = createPrint();
    JsonWriter writer = new JsonWriter();
```

```
    BytesWriterStrategy base64 = new Base64Strategy();
    writer.SetBytesStrategy(base64);
    writer.WriteRoot(fp);
    // print serialized output
    System.out.println("   " + writer.Output());
}
```

csharp

```
public void useJsonWithBase64()
{
    Print fp = createPrint();
    JsonWriter writer = new JsonWriter();
    BytesWriterStrategy base64 = new Base64Strategy();
    writer.SetBytesStrategy(base64);
    writer.WriteRoot(fp);
    // print serialized output
    System.Console.WriteLine("   " + writer.Output());
}
```

Use client specific serialization

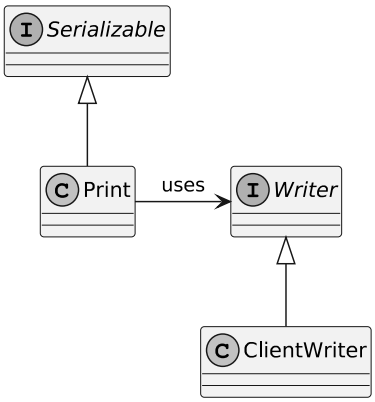


Figure 83 Use client specific serialization

C++

```
class ClientWriter : public Writer
{
public:
    std::string text;

    void Write(const char* name, const Serializable& s) final
    {
        text += name;
        text += "(";
        s.WriteTo(*this);
        text += ")";
    }
    void WriteRoot(Serializable& s) final
    {
        s.WriteTo(*this);
    }
    void WriteNull(const char* name) final
    {
        text += name;
        text += "(";
        text += "null";
        text += ")";
    }
    void Write(const char* name, const char* value) final
    {
        text += name;
        text += "(";
        text += value;
        text += ")";
    }

    void Write(const char* name, double value) final
```

```
    {
        Write(name, std::to_string(value).c_str());
    }
    void Write(const char* name, int64_t value) final
    {
        Write(name, std::to_string(value).c_str());
    }
    void Write(const char* name, uint64_t value) final
    {
        Write(name, std::to_string(value).c_str());
    }
    void Write(const char* name, int value) final
    {
        Write(name, std::to_string(value).c_str());
    }
    void Write(const char* name, unsigned int value) final
    {
        Write(name, std::to_string(value).c_str());
    }
    void Write(const char* name, const std::shared_ptr<Image>& value) final
    {
        auto w = std::to_string(value->Width());
        auto h = std::to_string(value->Height());
        const std::string desc = "w:" + w + ",h:" + h;
        Write(name, desc.c_str());
    }

private:
    void StartArray(const char* /*name*/) final
    {
        // for simplicity we ignore arrays in example
    }
    void EndArray() final
    {
        // for simplicity we ignore arrays in example
    }
};

static void UseClientSpecificWriter()
{
    auto fp = CreatePrint();
    ClientWriter writer;
    writer.WriteRoot(*fp);
    // print serialized output
    std::cout « "   " « writer.text « std::endl;
}
```

java

```
class ClientWriter extends com.innovatrics.enrollment.Writer {
    public String text;

    public ClientWriter() {
        text = "";
    }

    public void Write(String name, com.innovatrics.enrollment.Serializable s) {
        text += name + "(";
        s.WriteTo(this);
        text += ")";
    }

    public void WriteRoot(com.innovatrics.enrollment.Serializable s) {
        s.WriteTo(this);
    }

    public void Write(String name, String value) {
        text += name + "(" + value + ")";
    }

    public void Write(String name, int value) {
        text += name + "(" + value + ")";
    }

    public void Write(String name, long value) {
        text += name + "(" + value + ")";
    }

    public void Write(String name, Image value) {
        Write(name, "w:" + value.Width() + ",h:" + value.Height());
    }

    protected void StartArray(String name) {
        // for simplicity we ignore arrays in example
    }

    protected void EndArray() {
```

```
        // for simplicity we ignore arrays in example
    }
}

public void useClientSpecificWriter() {
    Print fp = createPrint();
    ClientWriter writer = new ClientWriter();
    writer.WriteRoot(fp);
    // print serialized output
    System.out.println("   " + writer.text);
}
```

csharp

```
class ClientWriter : Writer
{
    public string text;

    public ClientWriter()
    {
    }

    public override void Write(string name, Serializable s)
    {
        text += name + "(";
        s.WriteTo(this);
        text += ")";
    }
    public override void WriteRoot(Serializable s)
    {
        s.WriteTo(this);
    }

    public override void Write(string name, string value)
    {
        text += name + "(" + value + ")";
    }

    public override void Write(string name, int value)
    {
        text += name + "(" + value + ")";
    }

    public override void Write(string name, uint value)
    {
        text += name + "(" + value + ")";
    }

    public override void Write(string name, long value)
    {
        text += name + "(" + value + ")";
    }

    public override void Write(string name, ulong value)
    {
        text += name + "(" + value + ")";
    }

    public override void Write(string name, Image value)
    {
        Write(name, "w:" + value.Width() + ",h:" + value.Height());
    }

    protected override void StartArray(string name)
    {
        // for simplicity we ignore arrays in example
    }
    protected override void EndArray()
    {
        // for simplicity we ignore arrays in example
    }
}

public void useClientSpecificWriter()
{
    Print fp = createPrint();
    ClientWriter writer = new ClientWriter();
    writer.WriteRoot(fp);
    // print serialized output
    System.Console.WriteLine("   " + writer.text);
}
```

Use client specific bytes strategy

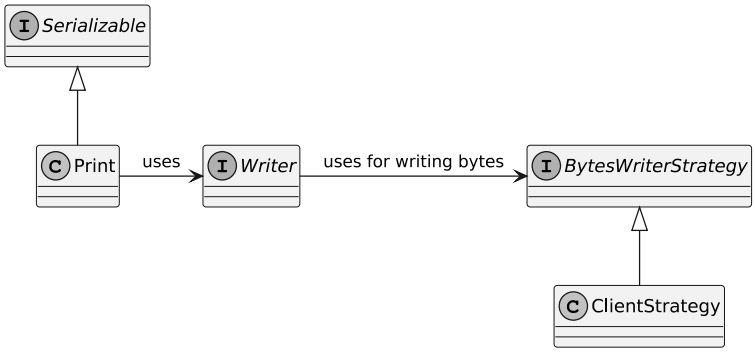


Figure 84 Use client specific bytes strategy

C++

```
class ClientStrategy : public BytesWriterStrategy
{
public:
    void WriteBytes(Writer& writer,
                    const char* name,
                    const std::vector<uint8_t>& value) final
    {
        auto size = std::to_string(value.size());
        auto v0 = std::to_string(value[0]);
        auto v1 = std::to_string(value[1]);
        const std::string text = "Reduced 2/" + size + " " + v0 + "," + v1 + ", ...";
        writer.Write(name, text.c_str());
    }
};

void UseClientSpecificByteStrategy()
{
    auto fp = CreatePrint();
    JsonWriter writer;
    ClientStrategy bytesStrategy;
    writer.SetBytesStrategy(bytesStrategy);
    writer.WriteRoot(*fp);
    // print serialized output
    std::cout « "    " « writer.Output() « std::endl;
}
```

java

```
class ClientStrategy extends BytesWriterStrategy {
    public void WriteBytes(
            com.innovatrics.enrollment.Writer writer, String name, Bytes value) {
        int s = value.size();
        int v0 = value.get(0);
        int v1 = value.get(1);
        writer.Write(name, "Reduced 2/" + s + " " + v0 + "," + v1 + ", ...");
    }
}

public void useClientSpecificByteStrategy() {
    Print fp = createPrint();
    JsonWriter writer = new JsonWriter();
    ClientStrategy bytesStrategy = new ClientStrategy();
    writer.SetBytesStrategy(bytesStrategy);
    writer.WriteRoot(fp);
    // print serialized output
    System.out.println("    " + writer.Output());
}
```

csharp

```
class ClientStrategy : BytesWriterStrategy
{
    public override void WriteBytes(Writer writer, string name, Bytes value)
    {
        int s = value.Count;
        int v0 = value[0];
        int v1 = value[1];
        writer.Write(name, "Reduced 2/" + s + " " + v0 + "," + v1 + ", ...");
    }
}

public void useClientSpecificByteStrategy()
```

```
{
    Print fp = createPrint();
    JsonWriter writer = new JsonWriter();
    ClientStrategy bytesStrategy = new ClientStrategy();
    writer.SetBytesStrategy(bytesStrategy);
    writer.WriteRoot(fp);
    // print serialized output
    System.Console.WriteLine("    " + writer.Output());
}
```

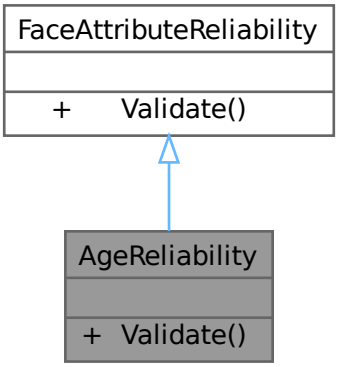
4 Class Documentation

4.1 AgeReliability Class Reference

Check if calculation of Age attribute will be reliable.

```
#include <FaceValidation.h>
```

Collaboration diagram for AgeReliability:



Public Member Functions

- std::vector< FailedFaceAttribute > Validate (const std::shared_ptr< Face > &probe) final

  Check if calculation of Age attribute will be reliable.

4.1.1 Member Function Documentation

4.1.1.1 Validate()

```
std::vector< FailedFaceAttribute > Validate (
            const std::shared_ptr< Face > & probe)  [inline], [final], [virtual]
```

Parameters

probe	Face to be checked

Returns

list of attributes that will cause not reliable Age attribute

Exceptions

NullArgumentException	when probe is null

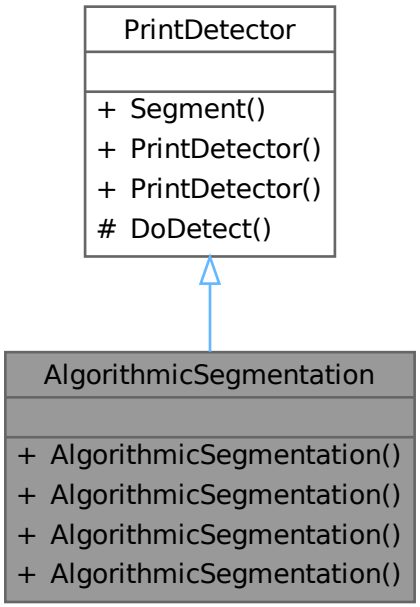
Implements FaceAttributeReliability.

4.2 AlgorithmicSegmentation Class Reference

AlgorithmicSegmentation implements algorithmic segmentation from Innovatrics ISegLib library.

```
#include <PrintDetector.h>
```

Collaboration diagram for AlgorithmicSegmentation:



Classes

- class Config

  Configuration for AlgorithmicSegmentation.

Public Member Functions

- AlgorithmicSegmentation (Config config)

  Construct a new AlgorithmicSegmentation object.
- **AlgorithmicSegmentation** ()=default

  Construct a new Algorithmic Segmentation with default config.
- AlgorithmicSegmentation (Config config, std::shared_ptr< PrintExecutor > executor)

  Construct a new AlgorithmicSegmentation object.
- AlgorithmicSegmentation (std::shared_ptr< PrintExecutor > executor)

  Construct a new Algorithmic Segmentation with default config.

Public Member Functions inherited from PrintDetector

- std::vector< std::shared_ptr< PrintSegment > > Segment (const std::shared_ptr< PrintCapture > &capture) final

  Detects segments in given image The Function scans the provided capture to locate fingers.
- **PrintDetector** ()

  Construct a new PrintDetector object GlobalPrintExecutor is used for PrintSegment detection.
- PrintDetector (std::shared_ptr< PrintExecutor > executor)

  Construct a new PrintDetector object.

4.2.1 Constructor & Destructor Documentation

4.2.1.1 AlgorithmicSegmentation() [1/3]

```
AlgorithmicSegmentation (
            Config config)  [inline], [explicit]
```

Parameters

config	Config

4.2.1.2 AlgorithmicSegmentation() [2/3]

```
AlgorithmicSegmentation (
            Config config,
            std::shared_ptr< PrintExecutor > executor)  [inline]
```

Parameters

config	Config
executor	PrintExecutor to set where functionality is executed

Exceptions

NullArgumentException	when executor is null

4.2.1.3 AlgorithmicSegmentation() [3/3]

```
AlgorithmicSegmentation (
            std::shared_ptr< PrintExecutor > executor)  [inline], [explicit]
```

Parameters

executor	PrintExecutor to set where functionality is executed

Exceptions

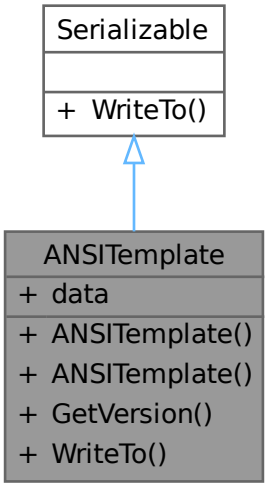
NullArgumentException	when executor is null

4.3 ANSITemplate Class Reference

ANSI INCITS 378 print compliant template.

```
#include <ANSITemplate.h>
```

Collaboration diagram for ANSITemplate:



Public Member Functions

- ANSITemplate (std::vector< uint8_t > &&d) noexcept

  Create ANSITemplate object.
- ANSITemplate (const std::vector< uint8_t > &d)

  Create ANSITemplate object, copy data to ANSI template.
- std::string GetVersion () const

  Provides version information of Innovatrics proprietary algorithm used to create the ANSI template.
- void WriteTo (Writer &writer) const final

  Write itself into writer.

Public Attributes

- const std::vector< uint8_t > **data**

  Template data.

4.3.1 Constructor & Destructor Documentation

4.3.1.1 ANSITemplate() [1/2]

```
ANSITemplate (
            std::vector< uint8_t > && d)  [inline], [explicit], [noexcept]
```

Parameters

d	template data

4.3.1.2 ANSITemplate() [2/2]

```
ANSITemplate (
            const std::vector< uint8_t > & d)  [inline], [explicit]
```

Parameters

d	template data

4.3.2 Member Function Documentation

4.3.2.1 GetVersion()

```
std::string GetVersion () const  [inline], [nodiscard]
```

Returns

version string

Exceptions

EnrollmentRuntimeException	when it is not possible to read version from the template.

4.3.2.2 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

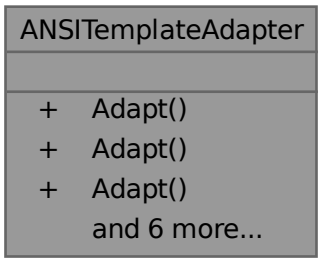
Implements Serializable.

4.4 ANSITemplateAdapter Class Reference

Adapts Template to ANSITemplate.

```
#include <ANSITemplateAdapter.h>
```

Collaboration diagram for ANSITemplateAdapter:



Public Member Functions

- ANSITemplate Adapt (const ISONCTemplate &tmpl) const

  Adapts ISONCTemplate to ANSITemplate.
- ANSITemplate Adapt (const ILOSIDTemplate &tmpl) const

  Adapts ILOSIDTemplate to ANSITemplate.
- ANSITemplate Adapt (const ISOCCTemplate &tmpl) const

  Adapts ISOCCTemplate to ANSITemplate.
- ANSITemplate Adapt (const ISOTemplate &tmpl) const

  Adapts ISOTemplate to ANSITemplate.
- ANSITemplate Adapt (const ICSTemplate &tmpl) const

  Adapts ICSTemplate to ANSITemplate.
- ANSITemplate Adapt (const ExtractedPrint &tmpl) const

  Adapts ExtractedPrint to ANSITemplate.
- ANSITemplate Adapt (const PrintTemplate &tmpl) const

  Adapts PrintTemplate to ANSITemplate.
- **ANSITemplateAdapter** ()

  Creates ANSITemplateAdapter object with GlobalPrintExecutor used to execute template functionality.
- ANSITemplateAdapter (std::shared_ptr< PrintExecutor > executor)

  Creates ANSITemplateAdapter object.

4.4.1 Constructor & Destructor Documentation

4.4.1.1 ANSITemplateAdapter()

```
ANSITemplateAdapter (
            std::shared_ptr< PrintExecutor > executor)  [inline], [explicit]
```

Parameters

executor	PrintExecutor used to execute template functionality.

Exceptions

NullArgumentException	when executor is null

4.4.2 Member Function Documentation

4.4.2.1 Adapt() [1/7]

```
ANSITemplate Adapt (
            const ExtractedPrint & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ANSITemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.4.2.2 Adapt() [2/7]

```
ANSITemplate Adapt (
            const ICSTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ANSITemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.4.2.3 Adapt() [3/7]

```
ANSITemplate Adapt (
            const ILOSIDTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ANSITemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.4.2.4 Adapt() [4/7]

```
ANSITemplate Adapt (
            const ISOCCTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ANSITemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.4.2.5 Adapt() [5/7]

```
ANSITemplate Adapt (
            const ISONCTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ANSITemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.4.2.6 Adapt() [6/7]

```
ANSITemplate Adapt (
            const ISOTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ANSITemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.4.2.7 Adapt() [7/7]

```
ANSITemplate Adapt (
            const PrintTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ANSITemplate

Exceptions

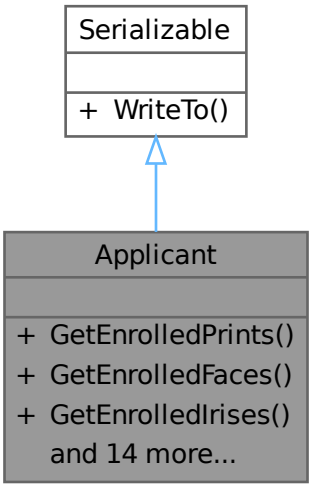
EnrollmentException	when it is not possible to adapt template

4.5 Applicant Class Reference

Applicant represent enrolled person.

```
#include <Applicant.h>
```

Collaboration diagram for Applicant:



Public Member Functions

- std::vector< std::shared_ptr< ExtractedPrint > > GetEnrolledPrints () const

  Function provides enrolled Prints.
- std::vector< std::shared_ptr< ExtractedFace > > GetEnrolledFaces () const

  Function provides enrolled Faces.
- std::vector< std::shared_ptr< ExtractedIris > > GetEnrolledIrises () const

  Function provides enrolled Irises.
- std::shared_ptr< ModalitiesTemplate > ToTemplate () const

  Converts Applicant to ModalitiesTemplate.
- std::shared_ptr< ModalitiesICSTemplate > ToICSTemplate () const

  Converts Applicant to ModalitiesICSTemplate.
- std::shared_ptr< ModalitiesICSTemplate > ToICSTemplate (ModalitiesICSTemplateFactory &factory) const

  Converts Applicant to ModalitiesICSTemplate.

- ModalityScore SimilarWith (const Applicant &gallery, ApplicantMatcher &matcher) const

  Calculates SimilarityScore for each modality.
- std::shared_ptr< ExtractedPrint > AddUniquePrint (std::shared_ptr< ExtractedPrint > print)

  Validates print according to default FlatteningValidation and adds Print when is valid.
- bool AddValidPrint (std::shared_ptr< ExtractedPrint > print, ApplicantPrintValidation &validation)

  Validates print according to given ApplicantPrintValidation and adds Print when is valid See Custom Print validation.
- void AddPrint (std::shared_ptr< ExtractedPrint > print)

  Adds Print to applicant.
- void AddSlap (std::shared_ptr< SlapCapture > slap)

  It adds SlapCapture to applicant.
- void AddThumbs (std::shared_ptr< ThumbsCapture > thumbs)

  It adds ThumbsCapture to applicant.
- bool AddValidFace (std::shared_ptr< ExtractedFace > face, ApplicantFaceValidation &validation)

  Validates Face according to given ApplicantFaceValidation and adds Face when is valid.
- void AddFace (std::shared_ptr< ExtractedFace > face)

  It adds Face to applicant.
- void AddIris (std::shared_ptr< ExtractedIris > iris)

  It adds Iris to applicant.
- void WriteTo (Writer &writer) const final

  It serializes applicant via provided Writer.
- **Applicant** ()

  Constructor uses GLobalLogger.

4.5.1 Detailed Description

It holds person biometric data. It provides features:

- Add print
- Add Face
- Add Iris

Print

The Print can be added via functions

- AddPrint - use no validation
- AddUniquePrint - uses FlatteningValidation
- AddValidPrint - use customized implementation of ApplicantPrintValidation, see Custom Print validation.

Face

The Face can be added via functions

- AddFace - use no validation
- AddValidFace - use customized implementation of ApplicantFaceValidation see Custom Face validation.

Iris

The Iris can be added via functions

- AddIris - use no validation

4.5.2 Member Function Documentation

4.5.2.1 AddFace()

```
void AddFace (
            std::shared_ptr< ExtractedFace > face)  [inline]
```

Parameters

face	ExtractedFace detected by FaceDetector and extracted

Exceptions

NullArgumentException	when face is null object

4.5.2.2 AddIris()

```
void AddIris (
            std::shared_ptr< ExtractedIris > iris)  [inline]
```

Parameters

iris	Iris detected by IrisDetector

Exceptions

NullArgumentException	when iris is null object

4.5.2.3 AddPrint()

```
void AddPrint (
            std::shared_ptr< ExtractedPrint > print)  [inline]
```

Parameters

print	print template calculated by PrintExtractor.

Exceptions

NullArgumentException	when print is null

4.5.2.4 AddSlap()

```
void AddSlap (
            std::shared_ptr< SlapCapture > slap)  [inline]
```

Parameters

slap	SlapCapture

Exceptions

NullArgumentException	when slap is null

4.5.2.5 AddThumbs()

```
void AddThumbs (
            std::shared_ptr< ThumbsCapture > thumbs)  [inline]
```

Parameters

thumbs	ThumbsCapture

Exceptions

NullArgumentException	when thumbs is null

4.5.2.6 AddUniquePrint()

```
std::shared_ptr< ExtractedPrint > AddUniquePrint (
            std::shared_ptr< ExtractedPrint > print)  [inline]
```

See Print flattening.

Parameters

print	ExtractedPrint to be checked and added.

Returns

enrolled print that causes invalidity of currently enrolling Print

null when print is valid

Exceptions

NullArgumentException	when FlatteningValidation throws

4.5.2.7 AddValidFace()

```
bool AddValidFace (
            std::shared_ptr< ExtractedFace > face,
            ApplicantFaceValidation & validation)  [inline]
```

See Custom Face validation.

Parameters

face	ExtractedFace to be checked and added
validation	ApplicantFaceValidation to check Face validity

Returns

`true` when Face was added

`false` when Face do not pass ApplicantFaceValidation::Validate

Exceptions

NullArgumentException	when probe is null

4.5.2.8 AddValidPrint()

```
bool AddValidPrint (
            std::shared_ptr< ExtractedPrint > print,
            ApplicantPrintValidation & validation)  [inline]
```

Parameters

print	ExtractedPrint to be checked and added
validation	ApplicantPrintValidation to check print validity

Returns

`true` when print was added

`false` when print do not pass ApplicantPrintValidation::Validate

Exceptions

NullArgumentException	when print is null

4.5.2.9 GetEnrolledFaces()

```
std::vector< std::shared_ptr< ExtractedFace > > GetEnrolledFaces () const  [inline]
```

Returns

enrolled faces

4.5.2.10 GetEnrolledIrises()

```
std::vector< std::shared_ptr< ExtractedIris > > GetEnrolledIrises () const  [inline]
```

Returns

enrolled irises

4.5.2.11 GetEnrolledPrints()

```
std::vector< std::shared_ptr< ExtractedPrint > > GetEnrolledPrints () const  [inline]
```

Returns

enrolled prints

4.5.2.12 SimilarWith()

```
ModalityScore SimilarWith (
            const Applicant & gallery,
            ApplicantMatcher & matcher) const  [inline], [nodiscard]
```

This is convenient function to check similarity of applicants. It is not efficient because it always creates probe and gallery ModalitiesTemplate.

Parameters

gallery	applicant to compare with
matcher	is ApplicantMatcher used to calculate ModalityScore between probe (this Applicant) and given Applicant `gallery`.

Returns

ModalityScore which contains the similarity score for each modality

Exceptions

EnrollmentException	when it is not possible to match applicants

4.5.2.13 ToICSTemplate() [1/2]

```
std::shared_ptr< ModalitiesICSTemplate > ToICSTemplate () const [inline]
```

It gets all enrolled prints, faces, irises and converts them to ModalityICSTemplate. It uses default ModalitiesICSTemplateDefaultFactory for conversion.

Returns

ModalitiesICSTemplate with all enrolled modalities

4.5.2.14 ToICSTemplate() [2/2]

```
std::shared_ptr< ModalitiesICSTemplate > ToICSTemplate (
            ModalitiesICSTemplateFactory & factory) const [inline]
```

It gets all enrolled prints, faces, irises and converts them to ModalityICSTemplate.

Parameters

factory	ModalitiesICSTemplateFactory used for conversion

Returns

ModalitiesICSTemplate with all enrolled modalities

4.5.2.15 ToTemplate()

```
std::shared_ptr< ModalitiesTemplate > ToTemplate () const [inline]
```

It gets all enrolled prints, faces, irises and converts them to ModalityTemplate. The ModalityTemplate can be later used in ModalitiesGallery as probe or candidate. The ModalityTemplate can be later used in ModalitiesVerifyMatcher as probe or reference for Applicant verification.

Returns

ModalitiesTemplate with all enrolled modalities

4.5.2.16 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Applicant.

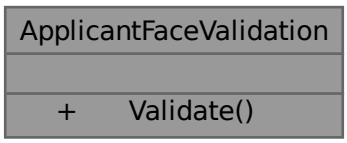
Implements Serializable.

4.6 ApplicantFaceValidation Class Reference

Strategy used in Applicant to accept or reject ExtractedFace.

```
#include <Applicant.h>
```

Collaboration diagram for ApplicantFaceValidation:



Public Member Functions

- virtual bool Validate (const std::shared_ptr< ExtractedFace > &probe, const std::vector< std::shared_ptr< ExtractedFace > > &gallery)=0

  Check if Face is valid.

4.6.1 Member Function Documentation

4.6.1.1 Validate()

```
virtual bool Validate (
            const std::shared_ptr< ExtractedFace > & probe,
            const std::vector< std::shared_ptr< ExtractedFace > > & gallery)  [pure virtual]
```

Parameters

probe	ExtractedFace to be checked

gallery	ExtractedFace that are already validated

Returns

`true` when Face is valid

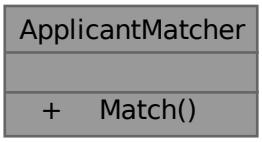
`false` when Face is not valid

4.7 ApplicantMatcher Class Reference

It is general matcher for Applicant.

```
#include <ApplicantMatcher.h>
```

Collaboration diagram for ApplicantMatcher:



Public Member Functions

- virtual ModalityScore Match (const Applicant &probe, const Applicant &gallery) const =0

  Calculates similarity score of probe and gallery applicant ICS template for each modality.

4.7.1 Detailed Description

It compares probe and gallery applicant. Result of comparison is ModalityScore.

4.7.2 Member Function Documentation

4.7.2.1 Match()

```
virtual ModalityScore Match (
            const Applicant & probe,
            const Applicant & gallery) const  [nodiscard], [pure virtual]
```

Parameters

probe	probe Applicant
gallery	gallery Applicant

Returns

similarity ModalityScore

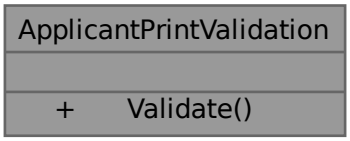
Implemented in ModalitiesMatcher, and ModalitiesVerifyMatcher.

4.8 ApplicantPrintValidation Class Reference

Strategy used in Applicant to accept or reject ExtractedPrint.

```
#include <Applicant.h>
```

Collaboration diagram for ApplicantPrintValidation:



Public Member Functions

- virtual bool Validate (const std::shared_ptr< ExtractedPrint > &probe, const std::vector< std::shared_ptr< ExtractedPrint > > &gallery)=0

  Check if print is valid.

4.8.1 Member Function Documentation

4.8.1.1 Validate()

```
virtual bool Validate (
            const std::shared_ptr< ExtractedPrint > & probe,
            const std::vector< std::shared_ptr< ExtractedPrint > > & gallery)  [pure virtual]
```

Parameters

probe	ExtractedPrint to be checked
gallery	ExtractedPrint that are already validated

Returns

`true` when Print is valid

`false` when Print is not valid

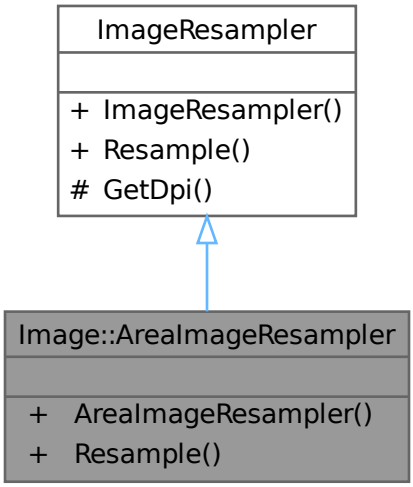
Implemented in FlatteningValidation.

4.9 Image::AreaImageResampler Class Reference

Image resampler for DPI-change operations, including upsampling and downsampling using area interpolation resampling algorithm.

```
#include <Image.h>
```

Collaboration diagram for Image::AreaImageResampler:



Public Member Functions

- AreaImageResampler (ImageDpi targetDpi)

  Constructs pixel-area-relation interpolation image resampler for a target DPI.
- std::shared_ptr< Image > Resample (const inno::Image &image) const final

  Resample image to a target DPI using pixel-area-relation interpolation.

Public Member Functions inherited from Image::ImageResampler

- ImageResampler (ImageDpi targetDpi)
  Constructs an image resampler for a target DPI.

Additional Inherited Members

Protected Member Functions inherited from Image::ImageResampler

- ImageDpi GetDpi () const
  Retrieves target DPI of the resampler.

4.9.1 Constructor & Destructor Documentation

4.9.1.1 AreaImageResampler()

```
AreaImageResampler (
            ImageDpi targetDpi)  [inline], [explicit]
```

This constructor initializes the image resampler to resample an image to the specified DPI using pixel-area-relation interpolation. In most cases, this resampler provides the fastest resampling among the Enrollment SDK resamplers.

Parameters

targetDpi	The DPI (dots per inch) of the new image.

Exceptions

EnrollmentInvalidArgumentException	when targetDpi is 0

4.9.2 Member Function Documentation

4.9.2.1 Resample()

```
std::shared_ptr< Image > Resample (
            const inno::Image & image) const  [inline], [nodiscard], [final], [virtual]
```

It creates new image as result of resampling with factor target DPI/original DPI. In most cases, this resampling method provides the fastest resampling among the Enrollment SDK resamplers.

Parameters

image	Image to resize.

Returns

image with given DPI

Exceptions

ImageException	when there is an processing error

Implements Image::ImageResampler.

4.10 ICSExtractor::Artifact Class Reference

Interface that generalize storing of intermediate images and other artifacts that are produced during extraction.

```
#include <ICSExtractor.h>
```

Collaboration diagram for ICSExtractor::Artifact:

Friends

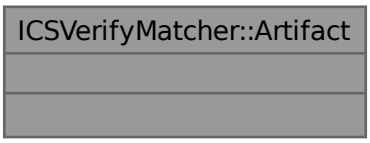
- class **ICSExtractor**

4.11 ICSVerifyMatcher::Artifact Class Reference

Interface that generalize storing of artifacts that are produced during matching.

```
#include <ICSVerifyMatcher.h>
```

Collaboration diagram for ICSVerifyMatcher::Artifact:



Friends

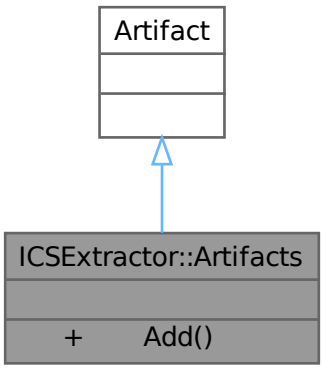
- class **ICSVerifyMatcher**

4.12 ICSExtractor::Artifacts Class Reference

It contains all added artifacts that were created during extraction.

```
#include <ICSExtractor.h>
```

Collaboration diagram for ICSExtractor::Artifacts:



Public Member Functions

- void Add (Artifact &artifact)

  Add artifact to be created during extraction.

4.12.1 Member Function Documentation

4.12.1.1 Add()

```
void Add (
            Artifact & artifact)  [inline]
```

Parameters

artifact	Artifact to be created during extraction

Exceptions

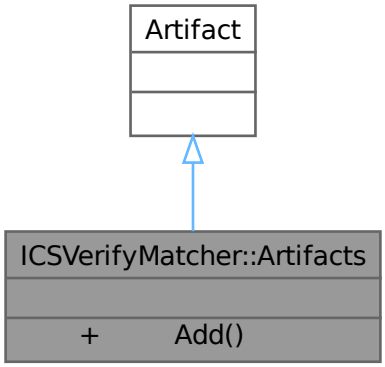
EnrollmentRuntimeException	when artifact is already enabled or is not compatible with already added artifacts
EnrollmentRuntimeException	when extraction is in progress

4.13 ICSVerifyMatcher::Artifacts Class Reference

It contains all added artifacts that were created during matching.

```
#include <ICSVerifyMatcher.h>
```

Collaboration diagram for ICSVerifyMatcher::Artifacts:



Public Member Functions

- void Add (Artifact &artifact)

  Add artifact to be created during matching.

4.13.1 Member Function Documentation

4.13.1.1 Add()

```
void Add (
            Artifact & artifact)  [inline]
```

Parameters

artifact	Artifact to be created during matching

Exceptions

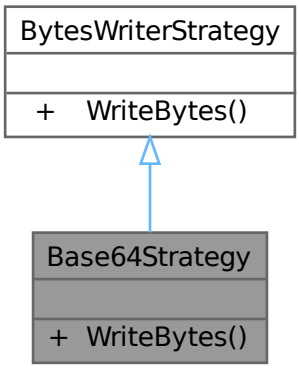
EnrollmentRuntimeException	when artifact is already enabled or is not compatible with already added artifacts
EnrollmentRuntimeException	when matching is in progress

4.14 Base64Strategy Class Reference

It writes bytes represented as std::vector<uint8_t> as base64 coded string.

```
#include <Base64Strategy.h>
```

Collaboration diagram for Base64Strategy:



Public Member Functions

- void WriteBytes (Writer &writer, const char *name, const std::vector< uint8_t > &value) final

  It writes bytes in given writer.

4.14.1 Detailed Description

It uses Writer to write converted bytes to string. The serialization output id provide by used Writer.

4.14.2 Member Function Documentation

4.14.2.1 WriteBytes()

```
void WriteBytes (
            Writer & writer,
            const char * name,
            const std::vector< uint8_t > & value)  [inline], [final], [virtual]
```

Parameters

writer	writer to use to write bytes
name	element name
value	bytes to write

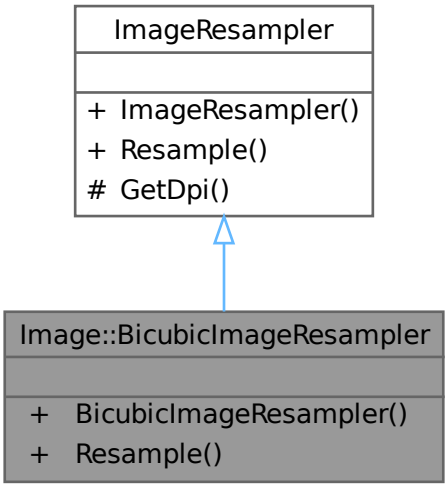
Implements BytesWriterStrategy.

4.15 Image::BicubicImageResampler Class Reference

Image resampler for DPI-change operations, including upsampling and downsampling using bicubic interpolation resampling algorithm.

```
#include <Image.h>
```

Collaboration diagram for Image::BicubicImageResampler:



Public Member Functions

- BicubicImageResampler (ImageDpi targetDpi)

  Constructs bicubic interpolation image resampler for a target DPI.
- std::shared_ptr< Image > Resample (const inno::Image &image) const final

  Resample image to a target DPI using bicubic interpolation.

Public Member Functions inherited from Image::ImageResampler

- ImageResampler (ImageDpi targetDpi)

  Constructs an image resampler for a target DPI.

Additional Inherited Members

Protected Member Functions inherited from Image::ImageResampler

- ImageDpi GetDpi () const
  Retrieves target DPI of the resampler.

4.15.1 Constructor & Destructor Documentation

4.15.1.1 BicubicImageResampler()

```
BicubicImageResampler (
            ImageDpi targetDpi)  [inline], [explicit]
```

This constructor initializes the image resampler to resample an image to the specified DPI using bicubic interpolation.

Parameters

targetDpi	The DPI (dots per inch) of the new image.

Exceptions

EnrollmentInvalidArgumentException	when targetDpi is 0

4.15.2 Member Function Documentation

4.15.2.1 Resample()

```
std::shared_ptr< Image > Resample (
            const inno::Image & image) const  [inline], [nodiscard], [final], [virtual]
```

It creates new image as result of resampling with factor target DPI/original DPI.

Parameters

image	Image to resize.

Returns

image with given DPI

Exceptions

ImageException	when there is an processing error

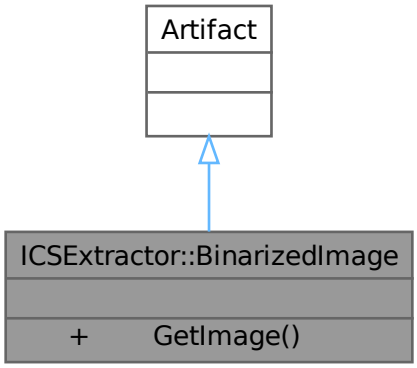
Implements Image::ImageResampler.

4.16 ICSExtractor::BinarizedImage Class Reference

It enables creation of binarized image during extraction.

```
#include <ICSExtractor.h>
```

Collaboration diagram for ICSExtractor::BinarizedImage:



Public Member Functions

- std::shared_ptr< Image > GetImage () const

  Get created Image To create Image, extraction must be finished.

4.16.1 Detailed Description

Note

The adjudicator feature MUST be allowed in license to get binarized image.

Figure 85 Finger to binarize

Figure 86 Binarized

4.16.2 Member Function Documentation

4.16.2.1 GetImage()

```
std::shared_ptr< Image > GetImage () const  [inline], [nodiscard]
```

Exceptions

EnrollmentRuntimeException	when extraction is not finished

Returns

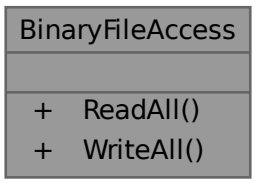
Binarized Image

4.17 BinaryFileAccess Class Reference

BinaryFileAccess provides abstraction on underlying file system.

```
#include <BinaryFile.h>
```

Collaboration diagram for BinaryFileAccess:



Public Member Functions

- virtual std::vector< uint8_t > ReadAll (const std::string &path)=0

  Reads whole file as binary to memory.
- virtual void WriteAll (const std::string &path, const std::vector< uint8_t > &content)=0

  Writes whole given memory to file as binary.

4.17.1 Member Function Documentation

4.17.1.1 ReadAll()

```
virtual std::vector< uint8_t > ReadAll (
            const std::string & path)  [pure virtual]
```

Parameters

path	file path

Returns

file content

Exceptions

FileException	on problem with file system

Implemented in FilesystemAccess.

4.17.1.2 WriteAll()

```
virtual void WriteAll (
            const std::string & path,
            const std::vector< uint8_t > & content)  [pure virtual]
```

Parameters

path	file path
content	content to write

Exceptions

FileException	on problem with file system

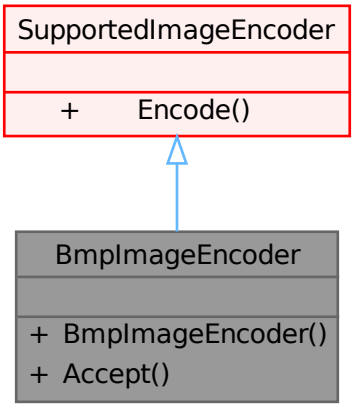
Implemented in FilesystemAccess.

4.18 BmpImageEncoder Class Reference

Converts image type ImageType::BMP.

```
#include <ImageEncoder.h>
```

Collaboration diagram for BmpImageEncoder:



Public Member Functions

- **BmpImageEncoder** ()

  Construct a new BMP Image Encoder object.
- void Accept (ImageEncoderVisitor &visitor) const final

  Allow visitor to visit concrete type of encoder.

Public Member Functions inherited from SupportedImageEncoder

- std::vector< uint8_t > Encode (const RawImage &raw) const final

  Saves given RawImage as formatted image.

Additional Inherited Members

Static Public Member Functions inherited from ImageEncoder

- static std::shared_ptr< ImageEncoder > Create (ImageType type)

  Create concrete default encoder for given type.

4.18.1 Member Function Documentation

4.18.1.1 Accept()

```
void Accept (
            ImageEncoderVisitor & visitor) const [inline], [final], [virtual]
```

Parameters

visitor	ImageEncoderVisitor to call

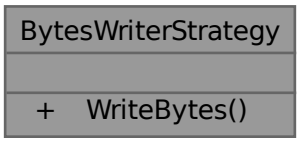
Implements ImageEncoder.

4.19 BytesWriterStrategy Class Reference

It is used by Writer class to modify bytes serialization algorithm.

```
#include <Writer.h>
```

Collaboration diagram for BytesWriterStrategy:



Public Member Functions

- virtual void WriteBytes (Writer &writer, const char ∗name, const std::vector< uint8_t > &value)=0

  It writes bytes in given writer.

4.19.1 Member Function Documentation

4.19.1.1 WriteBytes()

```
virtual void WriteBytes (
            Writer & writer,
            const char * name,
            const std::vector< uint8_t > & value)  [pure virtual]
```

Parameters

writer	writer to use to write bytes
name	element name
value	bytes to write

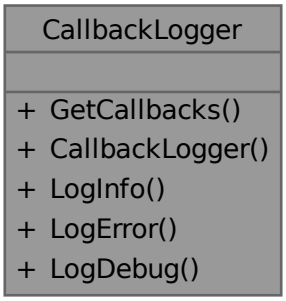
Implemented in Base64Strategy.

4.20 CallbackLogger Class Reference

CallbackLogger is logger for binary part of libary due to C boundary.

```
#include <Logger.h>
```

Collaboration diagram for CallbackLogger:



Public Member Functions

- LoggerCallBacks GetCallbacks () const

  Get the Callbacks object.
- CallbackLogger (std::shared_ptr< Logger > logger)

  Construct a new Callback Logger object.

Static Public Member Functions

- static void LogInfo (void *log, LogMessageCallback composer, void *composerData)

  Log message on info level.
- static void LogError (void *log, LogMessageCallback composer, void *composerData)

  Log message on error level.
- static void LogDebug (void *log, LogMessageCallback composer, void *composerData)

  Log message on debug level.

4.20.1 Detailed Description

It provides static functions that can be called from binary part to log messages.

Note

Only for C++

4.20.2 Constructor & Destructor Documentation

4.20.2.1 CallbackLogger()

```
CallbackLogger (
            std::shared_ptr< Logger > logger)  [inline], [explicit]
```

Parameters

logger	Logger

4.20.3 Member Function Documentation

4.20.3.1 GetCallbacks()

```
LoggerCallBacks GetCallbacks () const  [inline], [nodiscard]
```

Returns

LoggerCallBacks

4.20.3.2 LogDebug()

```
void LogDebug (
            void * log,
            LogMessageCallback composer,
            void * composerData)  [inline], [static]
```

Parameters

log	logger handle provided to binary in LoggerCallBacks::log
composer	callback created in binary that is used by header part when message needs to be composed
composerData	necessary data created in binary for composer

4.20.3.3 LogError()

```
void LogError (
            void * log,
            LogMessageCallback composer,
            void * composerData)  [inline], [static]
```

Parameters

log	logger handle provided to binary in LoggerCallBacks::log
composer	callback created in binary that is used by header part when message needs to be composed

composerData	necessary data created in binary for composer

4.20.3.4 LogInfo()

```
void LogInfo (
            void * log,
            LogMessageCallback composer,
            void * composerData)  [inline], [static]
```

Parameters

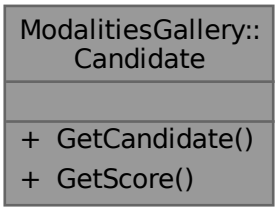
log	logger handle provided to binary in LoggerCallBacks::log
composer	callback created in binary that is used by header part when message needs to be composed
composerData	necessary data created in binary for composer

4.21 ModalitiesGallery::Candidate Class Reference

Found candidate.

```
#include <ModalitiesGallery.h>
```

Collaboration diagram for ModalitiesGallery::Candidate:



Public Member Functions

- std::shared_ptr< ModalitiesGalleryReference > GetCandidate () const

  Return identified candidate.
- ModalityScore GetScore () const

  Return score for candidate.

4.21.1 Detailed Description

It contains candidate and score per modality

4.21.2 Member Function Documentation

4.21.2.1 GetCandidate()

```
std::shared_ptr< ModalitiesGalleryReference > GetCandidate () const [inline], [nodiscard]
```

Returns

candidate

4.21.2.2 GetScore()

```
ModalityScore GetScore () const [inline], [nodiscard]
```

Returns

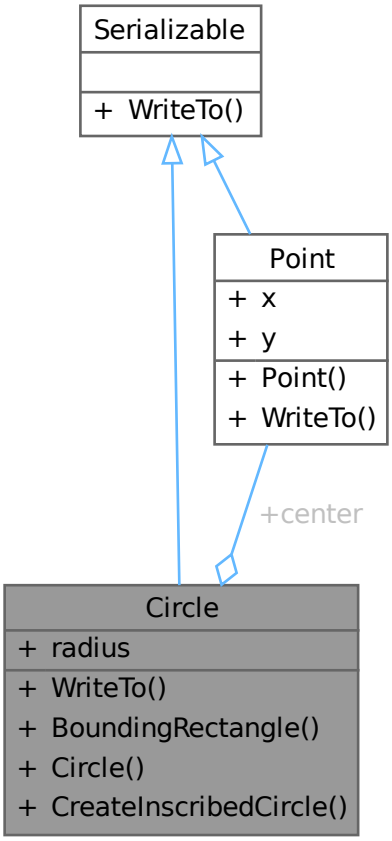
candidate score

4.22 Circle Class Reference

It holds circle dimensions.

`#include <ImageAttribute.h>`

Collaboration diagram for Circle:



Public Member Functions

- void WriteTo (Writer &writer) const final

  Write itself into writer.
- Rectangle BoundingRectangle () const

  It provides bounding rectangle for circle.
- Circle (Point centerPoint, Radius circleRadius)

  Constructor store given coordinates.

Static Public Member Functions

- static Circle CreateInscribedCircle (const Rectangle &rect)

  Creates the largest circle that fits entirely within a given rectangle.

Public Attributes

- Point **center**

  top left
- Radius **radius** = 0

  top right

4.22.1 Constructor & Destructor Documentation

4.22.1.1 Circle()

```
Circle (
            Point centerPoint,
            Radius circleRadius)  [inline]
```

Parameters

centerPoint	center of circle
circleRadius	radius of circle

4.22.2 Member Function Documentation

4.22.2.1 BoundingRectangle()

```
Rectangle BoundingRectangle () const  [inline], [nodiscard]
```

Returns

Rectangle bounding rectangle of circle

4.22.2.2 CreateInscribedCircle()

```
Circle CreateInscribedCircle (
            const Rectangle & rect)  [inline], [static], [nodiscard]
```

Parameters

rect	The rectangle from which to create the inner circle.

Returns

Circle representing the inner circle of the rectangle.

4.22.2.3 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

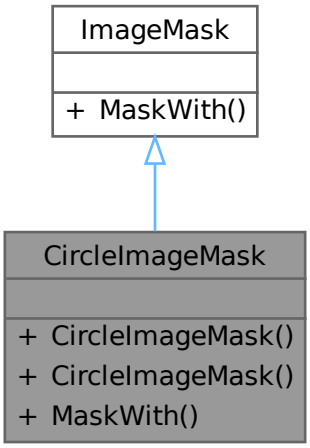
Implements Serializable.

4.23 CircleImageMask Class Reference

Represents an image mask in the shape of a circle.

```
#include <ImageMask.h>
```

Collaboration diagram for CircleImageMask:



Public Member Functions

- CircleImageMask (Circle circle, ImageMaskConfiguration config)

  Constructs a CircleImageMask with a specified circle and configuration.
- CircleImageMask (Circle circle, ImageMaskConfiguration config, ImageBlurType blur)

  Constructs a CircleImageMask with a specified circle and configuration and blurring.
- void MaskWith (ImageMasker &masker) const override

  Applies the circular mask to an image using the provided ImageMasker.

4.23.1 Detailed Description

It is convenient class for masking and blurring circle.

4.23.2 Constructor & Destructor Documentation

4.23.2.1 CircleImageMask() [1/2]

```
CircleImageMask (
            Circle circle,
            ImageMaskConfiguration config)  [inline]
```

Parameters

circle	The Circle object that defines the shape and bounds of the mask.
config	The ImageMaskConfiguration object that contains masking parameters.

4.23.2.2 CircleImageMask() [2/2]

```
CircleImageMask (
            Circle circle,
            ImageMaskConfiguration config,
            ImageBlurType blur)  [inline]
```

Parameters

circle	The Circle object that defines the shape and bounds of the mask.
blur	Type of blurring to by applied on mask-to-image transition.
config	The ImageMaskConfiguration object that contains masking parameters.

4.23.3 Member Function Documentation

4.23.3.1 MaskWith()

```
void MaskWith (
            ImageMasker & masker) const  [inline], [override], [virtual]
```

Parameters

masker	The ImageMasker instance used to apply the mask to an image.

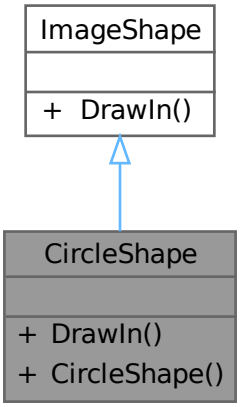
Implements ImageMask.

4.24 CircleShape Class Reference

It holds circle and draws it.

```
#include <ImageShape.h>
```

Collaboration diagram for CircleShape:



Public Member Functions

- void DrawIn (ImageShapeDrawer &drawer) const final

  It draws circle in ImageShapeDrawer.
- CircleShape (Circle circle, ImageShapeAttributes attr)

  Constructor sets given circle for drawing.

4.24.1 Constructor & Destructor Documentation

4.24.1.1 CircleShape()

```
CircleShape (
            Circle circle,
            ImageShapeAttributes attr)  [inline]
```

Parameters

circle	Circle
attr	ImageShapeAttributes to draw circle

4.24.2 Member Function Documentation

4.24.2.1 DrawIn()

```
void DrawIn (
            ImageShapeDrawer & drawer) const  [inline], [final], [virtual]
```

Parameters

drawer	the implementation of ImageShapeDrawer

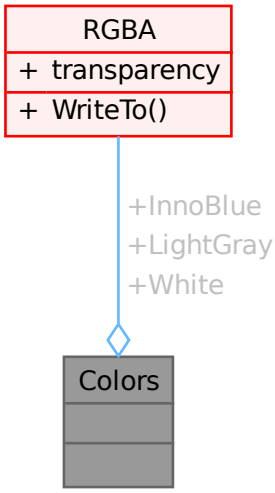
Implements ImageShape.

4.25 Colors Class Reference

It holds color literals.

`#include <ImageAttribute.h>`

Collaboration diagram for Colors:

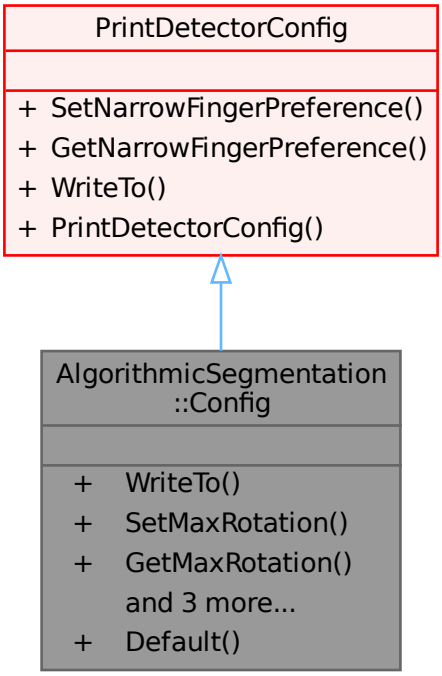


4.26 AlgorithmicSegmentation::Config Class Reference

Configuration for AlgorithmicSegmentation.

```
#include <PrintDetector.h>
```

Collaboration diagram for AlgorithmicSegmentation::Config:



Public Member Functions

- void WriteTo (Writer &writer) const final

  Function serializes configuration via provided Writer.
- void SetMaxRotation (unsigned int value)

  It sets maximum rotation of fingers in the slap image.
- unsigned int GetMaxRotation () const

  It gets maximum rotation of fingers in the slap image.
- void **EnableNoiseReduction** ()

  It enables noise reduction before segmentation.
- bool IsNoiseReduction () const

  It returns `true` when noise reduction before segmentation is enabled.
- Config ()=default

  Constructs default configuration.

Public Member Functions inherited from PrintDetectorConfig

- void SetNarrowFingerPreference (bool expectNarrow)

  Informs the print detector to anticipate narrow fingers.
- bool GetNarrowFingerPreference () const

  Retrieves information about the expected narrowness of detected fingers.
- void WriteTo (Writer &writer) const override

  Function serializes configuration via provided Writer.
- PrintDetectorConfig ()=default

  Default PrintDetector configuration is.

Static Public Member Functions

- static Config Default ()

  Provides default config.

4.26.1 Constructor & Destructor Documentation

4.26.1.1 Config()

```
Config () [default]
```

Default Algorithmic segmentation configuration is

- maxRotation - 45
- reduceNoise - false
- expectNarrowFinger - true

4.26.2 Member Function Documentation

4.26.2.1 Default()

```
Config Default () [inline], [static]
```

Default Algorithmic segmentation configuration is

- maxRotation - 45
- reduceNoise - false
- expectNarrowFinger - true

Returns

default config

4.26.2.2 GetMaxRotation()

```
unsigned int GetMaxRotation () const  [inline], [nodiscard]
```

Returns

maximum rotation of fingers in the slap image

4.26.2.3 IsNoiseReduction()

```
bool IsNoiseReduction () const  [inline], [nodiscard]
```

Returns

`true` when noise reduction is enabled

4.26.2.4 SetMaxRotation()

```
void SetMaxRotation (
            unsigned int value)  [inline]
```

Parameters

value	indicating maximum rotation of fingers in the slap image. Value is in degrees. Valid range <0,45>.

Exceptions

EnrollmentInvalidArgumentException	when value is not in range

4.26.2.5 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Config.

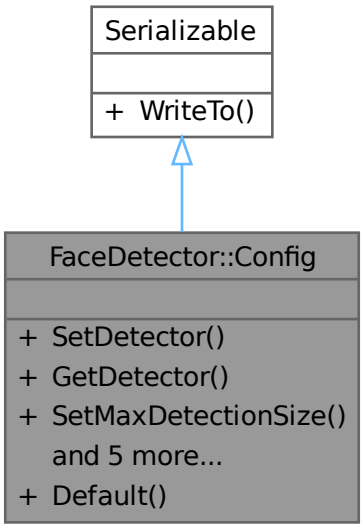
Implements Serializable.

4.27 FaceDetector::Config Class Reference

Face detector configuration.

```
#include <FaceDetector.h>
```

Collaboration diagram for FaceDetector::Config:



Public Types

- enum class Type : uint8_t { ACCURATE , BALANCED , FAST , ACCURATE_SERVER }

  Type of neural network detector.

Public Member Functions

- void SetDetector (Type value)

  Function sets type of detector.
- Type GetDetector () const

  Function gets detection type.
- void SetMaxDetectionSize (float value)

  Function sets max detection size.
- float GetMaxDetectionSize () const

  Function gets max detection size.
- void SetMinDetectionSize (float value)

  Function sets min detection size.
- float GetMinDetectionSize () const

  Function gets min detection size.
- void WriteTo (Writer &writer) const final

  Function serializes configuration via provided Writer.
- Config ()=default

  Creates Default configuration.

Static Public Member Functions

- static Config Default ()

  Provides default Config.

4.27.1 Detailed Description

It is input for detection.

4.27.2 Member Enumeration Documentation

4.27.2.1 Type

```
enum class Type : uint8_t [strong]
```

Enumerator

ACCURATE	Neural network face detector and facial features detector with very high accuracy. It is slower than `balanced` mode. **Required Solvers** Total size of solvers is 27 MB. References face-detector-accurate.solver, face-detector-landmarks-mob-0.50.solver, face-detector-eye-validation.solver
BALANCED	Neural network face detector and facial features detector with good accuracy. It is slower than `fast` mode. **Required Solvers** Total size of solvers is 14 MB. References face-detector-accurate.solver, face-detector-landmarks-mob-0.25.solver, face-detector-eye-validation.solver
FAST	The fastest face detector in combination with the fastest face validators and facial features detectors are used when *fast* mode is used. Some faces that are partially occluded faces or faces with sunglasses may be missed. However the speed performance of the face detection is much better as when other modes are used. **Required Solvers** Total size of solvers is 559 kB. References face-detector-fast.solver, face-detector-eye-validation.solver, face-detector-landmarks-fast.solver
ACCURATE_SERVER	**Note** Not supported in this variant of Enrollment SDK.

4.27.3 Constructor & Destructor Documentation

4.27.3.1 Config()

```
Config () [default]
```

The default configuration is:

- type - BALANCED
- min detection size - 0.05
- max detection size - 0.95

4.27.4 Member Function Documentation

4.27.4.1 Default()

```
Config Default () [inline], [static]
```

The default configuration is:

- type - BALANCED
- min detection size - 0.05
- max detection size - 0.95

Returns

default config

4.27.4.2 GetDetector()

```
Type GetDetector () const [inline], [nodiscard]
```

Returns

detection type

4.27.4.3 GetMaxDetectionSize()

```
float GetMaxDetectionSize () const [inline], [nodiscard]
```

Returns

max detection size.

4.27.4.4 GetMinDetectionSize()

```
float GetMinDetectionSize () const  [inline], [nodiscard]
```

Returns

min detection size.

4.27.4.5 SetDetector()

```
void SetDetector (
            Type value)  [inline]
```

Parameters

value	type from Config::Type.

4.27.4.6 SetMaxDetectionSize()

```
void SetMaxDetectionSize (
            float value)  [inline]
```

It specifies maximal valid size of face which is detectable in current detection mode. Face size is given by maximum of eyes distance and eyes to mouth distance in image.

Parameters

value	In case of (0., 1> range, the value is taken with respect to maximum of image dimensions. Otherwise it is considered as size in pixels.

4.27.4.7 SetMinDetectionSize()

```
void SetMinDetectionSize (
            float value)  [inline]
```

It specifies minimal valid size of face which is detectable in current detection mode. Face size is given by maximum of eyes distance and eyes to mouth distance in image.

Parameters

value	In case of (0., 1> range, the value is taken with respect to maximum of image dimensions. Otherwise it is considered as size in pixels.

4.27.4.8 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Config.

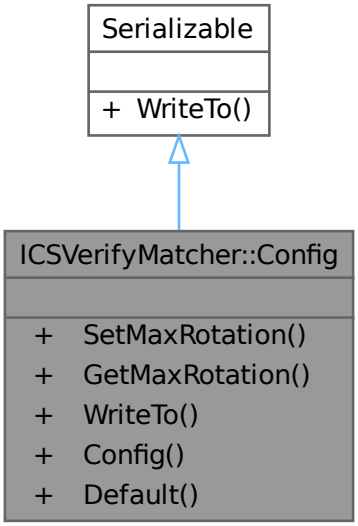
Implements Serializable.

4.28 ICSVerifyMatcher::Config Class Reference

ICSVerifyMatcher configuration.

```
#include <ICSVerifyMatcher.h>
```

Collaboration diagram for ICSVerifyMatcher::Config:



Public Member Functions

- void SetMaxRotation (unsigned int value)

  Set the Max Rotation.
- unsigned int GetMaxRotation () const

  Get the Max Rotation.
- void WriteTo (Writer &writer) const final

  Function serializes configuration via provided Writer.
- Config ()=default

  Default configuration is.

Static Public Member Functions

- static Config Default ()

  Provides default config.

4.28.1 Constructor & Destructor Documentation

4.28.1.1 Config()

```
Config () [default]
```

- max rotation - 180 degrees

4.28.2 Member Function Documentation

4.28.2.1 Default()

```
Config Default () [inline], [static]
```

- max rotation - 180 degrees

Returns

default config

4.28.2.2 GetMaxRotation()

```
unsigned int GetMaxRotation () const [inline], [nodiscard]
```

Returns

maximum rotation

4.28.2.3 SetMaxRotation()

```
void SetMaxRotation (
            unsigned int value)  [inline]
```

When taking print images, it is very likely that fingers might not be aligned exactly at the same angle every time. It defines maximum allowed rotation between two print images.Valid range is between 0 and 180, where [0, 180] is closed interval.

Default value is 180.

Parameters

value	in range <0, 180>

Exceptions

EnrollmentInvalidArgumentException	when max rotation is not in range

4.28.2.4 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Config.

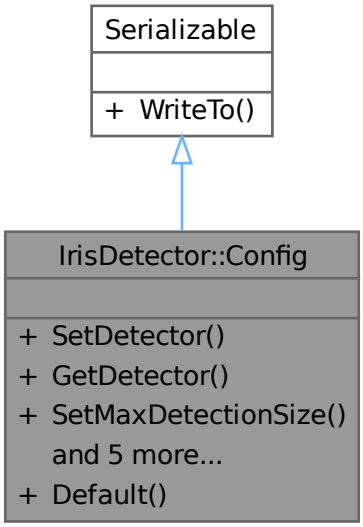
Implements Serializable.

4.29 IrisDetector::Config Class Reference

Iris detector configuration.

```
#include <IrisDetector.h>
```

Collaboration diagram for IrisDetector::Config:



Public Types

- enum class Type : uint8_t { ACCURATE , FAST }

  Type of neural network detector.

Public Member Functions

- void SetDetector (Type value)

  Function sets type of detector.
- Type GetDetector () const

  Function gets detection type.
- void SetMaxDetectionSize (float value)

  Function sets max detection size.
- float GetMaxDetectionSize () const

  Function gets max detection size.
- void SetMinDetectionSize (float value)

  Function sets min detection size.
- float GetMinDetectionSize () const

  Function gets min detection size.
- void WriteTo (Writer &writer) const final

  Function serializes configuration via provided Writer.
- Config ()=default

  Creates Default configuration.

Static Public Member Functions

- static Config Default ()

  Provides default configuration.

4.29.1 Detailed Description

It is input for detection.

4.29.2 Member Enumeration Documentation

4.29.2.1 Type

```
enum class Type : uint8_t [strong]
```

Enumerator

ACCURATE	Neural network iris detector with very high accuracy. It is slower than `fast` mode. **Required Solvers** Total size of solvers is 9 MB. References iris-detector.solver
FAST	The fastest iris detector is used when *fast* mode is used. The speed performance of the iris detection is much better as when other modes are used. **Required Solvers** Total size of solvers is 627 kB. References iris-detector-fast.solver

4.29.3 Constructor & Destructor Documentation

4.29.3.1 Config()

```
Config () [default]
```

The default configuration is:

- type - ACCURATE
- min detection size - 0.05
- max detection size - 0.95

4.29.4 Member Function Documentation

4.29.4.1 Default()

```
Config Default () [inline], [static]
```

The default configuration is:

- type - ACCURATE
- min detection size - 0.05
- max detection size - 0.95

Returns

default configuration

4.29.4.2 GetDetector()

```
Type GetDetector () const [inline], [nodiscard]
```

Returns

detection type

4.29.4.3 GetMaxDetectionSize()

```
float GetMaxDetectionSize () const [inline], [nodiscard]
```

Returns

max detection size.

4.29.4.4 GetMinDetectionSize()

```
float GetMinDetectionSize () const [inline], [nodiscard]
```

Returns

min detection size.

4.29.4.5 SetDetector()

```
void SetDetector (
            Type value)  [inline]
```

Parameters

value	type from Config::Type.

4.29.4.6 SetMaxDetectionSize()

```
void SetMaxDetectionSize (
            float value)  [inline]
```

It specifies maximal valid size of iris which is detectable in current detection mode. Iris size is given by the iris bounding box size in the image.

Parameters

value	In case of (0., 1> range, the value is taken with respect to maximum of image dimensions. Otherwise it is considered as size in pixels.

4.29.4.7 SetMinDetectionSize()

```
void SetMinDetectionSize (
            float value)  [inline]
```

It specifies minimal valid size of iris which is detectable in current detection mode. Iris size is given by the iris bounding box size in the image.

Parameters

value	In case of (0., 1> range, the value is taken with respect to maximum of image dimensions. Otherwise it is considered as size in pixels.

4.29.4.8 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Config.

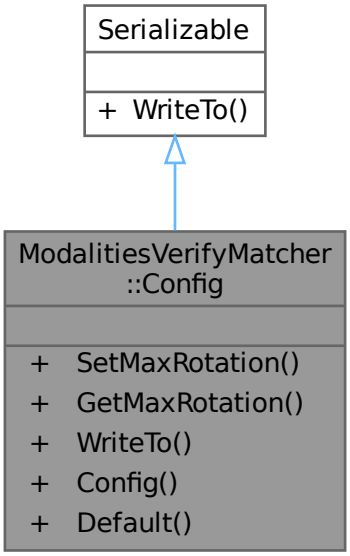
Implements Serializable.

4.30 ModalitiesVerifyMatcher::Config Class Reference

ModalitiesVerifyMatcher configuration.

```
#include <ModalitiesVerifyMatcher.h>
```

Collaboration diagram for ModalitiesVerifyMatcher::Config:



Public Member Functions

- void SetMaxRotation (unsigned int value)

  Set the Max Rotation.
- unsigned int GetMaxRotation () const

  Get the Max Rotation.
- void WriteTo (Writer &writer) const final

  Function serializes configuration via provided Writer.
- Config ()=default

  Default configuration is.

Static Public Member Functions

- static Config Default ()

  Provides default config.

4.30.1 Constructor & Destructor Documentation

4.30.1.1 Config()

```
Config () [default]
```

- max rotation - 180 degrees

4.30.2 Member Function Documentation

4.30.2.1 Default()

```
Config Default () [inline], [static]
```

- max rotation - 180 degrees

Returns

default config

4.30.2.2 GetMaxRotation()

```
unsigned int GetMaxRotation () const [inline], [nodiscard]
```

Returns

maximum rotation

4.30.2.3 SetMaxRotation()

```
void SetMaxRotation (
            unsigned int value) [inline]
```

When taking print images, it is very likely that fingers might not be aligned exactly at the same angle every time. It defines maximum allowed rotation between two print images.Valid range is between 0 and 180, where [0, 180] is closed interval.

Default value is 180.

Parameters

value	in range <0, 180>

Exceptions

EnrollmentInvalidArgumentException	when value is not in range <0,180>

4.30.2.4 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Config.

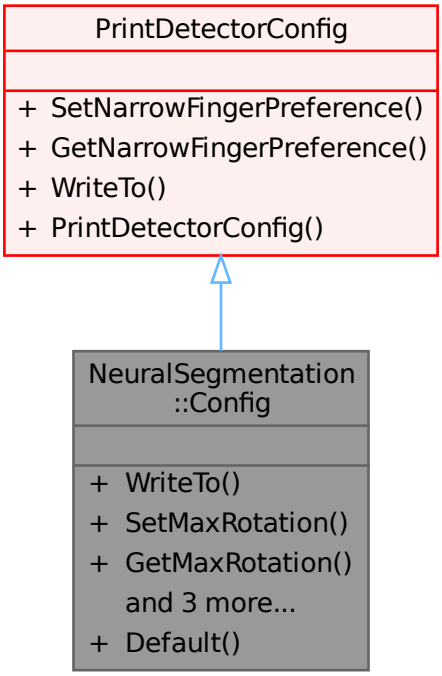
Implements Serializable.

4.31 NeuralSegmentation::Config Class Reference

Configuration for NeuralSegmentation.

```
#include <PrintDetector.h>
```

Collaboration diagram for NeuralSegmentation::Config:



Public Member Functions

- void WriteTo (Writer &writer) const final

  Function serializes configuration via provided Writer.
- void SetMaxRotation (unsigned int value)

  It sets maximum rotation of fingers in the slap image.
- unsigned int GetMaxRotation () const

  It gets maximum rotation of fingers in the slap image.
- void SetConfidenceThreshold (unsigned int value)

  It sets confidence threshold.
- unsigned int GetConfidenceThreshold () const

  It gets confidence threshold.
- Config ()=default

  Constructs default configuration.

Public Member Functions inherited from PrintDetectorConfig

- void SetNarrowFingerPreference (bool expectNarrow)

  Informs the print detector to anticipate narrow fingers.
- bool GetNarrowFingerPreference () const

  Retrieves information about the expected narrowness of detected fingers.
- void WriteTo (Writer &writer) const override

  Function serializes configuration via provided Writer.
- PrintDetectorConfig ()=default

  Default PrintDetector configuration is.

Static Public Member Functions

- static Config Default ()

  Provides default config.

4.31.1 Constructor & Destructor Documentation

4.31.1.1 Config()

```
Config () [default]
```

Default Neural segmentation configuration is

- maxRotation - 45
- confidenceThreshold - 9000
- expectNarrowFinger - true

4.31.2 Member Function Documentation

4.31.2.1 Default()

```
Config Default () [inline], [static]
```

Default Neural segmentation configuration is

- maxRotation - 45
- confidenceThreshold - 9000
- expectNarrowFinger - true

Returns

default config

4.31.2.2 GetConfidenceThreshold()

```
unsigned int GetConfidenceThreshold () const [inline], [nodiscard]
```

Returns

confidence threshold

4.31.2.3 GetMaxRotation()

```
unsigned int GetMaxRotation () const [inline], [nodiscard]
```

Returns

maximum rotation of fingers in the slap image

4.31.2.4 SetConfidenceThreshold()

```
void SetConfidenceThreshold (
            unsigned int value)  [inline]
```

Parameters

value	indicating how sure MUST be neural network to segment. Valid range is [0-10000], where 10000 means absolutely sure.

Exceptions

EnrollmentInvalidArgumentException	when value is not in range

4.31.2.5 SetMaxRotation()

```
void SetMaxRotation (
            unsigned int value)  [inline]
```

Parameters

value	indicating maximum rotation of fingers in the slap image. Value is in degrees. Valid range <0,45>.

Exceptions

EnrollmentInvalidArgumentException	when value is not in range

4.31.2.6 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Config.

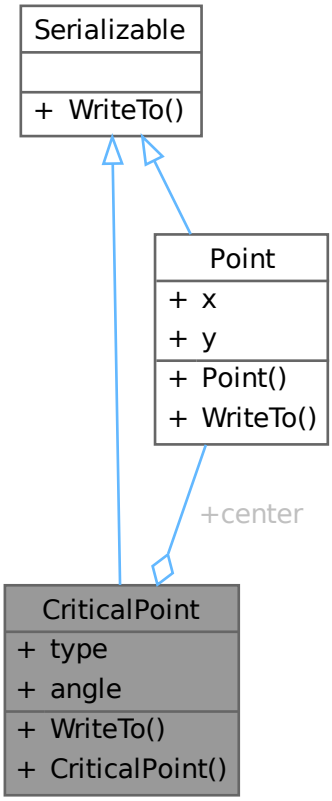
Implements Serializable.

4.32 CriticalPoint Class Reference

It represents a particular CriticalPoint.

```
#include <CriticalPoint.h>
```

Collaboration diagram for CriticalPoint:



Public Types

- enum class Type : uint8_t { CORE = 1 , DELTA = 2 }

  Type of CriticalPoint.

Public Member Functions

- void WriteTo (Writer &writer) const final

  Write itself into writer.
- CriticalPoint (Point pointArg, Type typeArg, Angle angleArg)

  It creates CriticalPoint with given parameters.

Public Attributes

- Point **center**

  CriticalPoint point.
- Type **type**

  CriticalPoint type (core/delta).
- Angle **angle**

  The orientation of the point is represented in degrees, with zero degrees pointing horizontally and to the right and increasing degrees proceeding counter-clockwise [9] .

4.32.1 Member Enumeration Documentation

4.32.1.1 Type

```
enum class Type : uint8_t [strong]
```

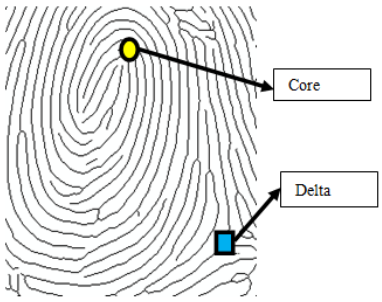


Figure 87 Critical points types

Enumerator

CORE	Core is topmost point on the innermost recurving ridge line of a fingerprint [4] .
DELTA	Delta is point on a ridge at or nearest to the point of divergence of two type lines (one of the two innermost friction ridges and located at or directly in front of the point of divergence [4] .

4.32.2 Constructor & Destructor Documentation

4.32.2.1 CriticalPoint()

```
CriticalPoint (
            Point pointArg,
            Type typeArg,
            Angle angleArg) [inline]
```

Parameters

pointArg	CriticalPoint point
typeArg	CriticalPoint type (core/delta)
angleArg	The orientation of the point is represented in degrees, with zero degrees pointing horizontally and to the right and increasing degrees proceeding counter-clockwise [9] .

4.32.3 Member Function Documentation

4.32.3.1 WriteTo()

```
void WriteTo (
            Writer & writer) const [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

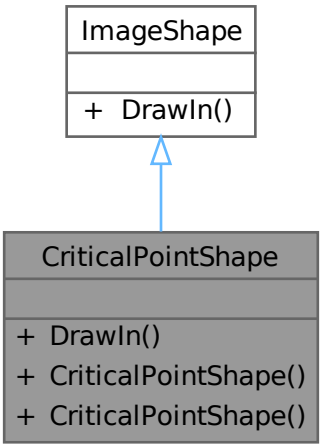
Implements Serializable.

4.33 CriticalPointShape Class Reference

It draws CriticalPoint into image.

`#include <CriticalPoint.h>`

Collaboration diagram for CriticalPointShape:



Public Member Functions

- void DrawIn (ImageShapeDrawer &drawer) const final

  It draws CriticalPoint into image.
- CriticalPointShape (std::shared_ptr< CriticalPoint > point, Radius radius, ImageShapeAttributes attr, unsigned int angleLength)

  Create CriticalPoint shape.
- CriticalPointShape (std::shared_ptr< CriticalPoint > point, ImageShapeAttributes attr)

  Create CriticalPoint shape.

4.33.1 Detailed Description

The shapes of critical points are

- CriticalPoint::Type::CORE - circle
- CriticalPoint::Type::DELTA - square

4.33.2 Constructor & Destructor Documentation

4.33.2.1 CriticalPointShape() [1/2]

```
CriticalPointShape (
            std::shared_ptr< CriticalPoint > point,
            Radius radius,
            ImageShapeAttributes attr,
            unsigned int angleLength)  [inline]
```

Parameters

point	point to draw
radius	radius of point
attr	ImageShapeAttributes of point
angleLength	length of angle line

4.33.2.2 CriticalPointShape() [2/2]

```
CriticalPointShape (
            std::shared_ptr< CriticalPoint > point,
            ImageShapeAttributes attr)  [inline]
```

Parameters

point	point to draw
attr	ImageShapeAttributes of point

4.33.3 Member Function Documentation

4.33.3.1 DrawIn()

```
void DrawIn (
            ImageShapeDrawer & drawer) const  [inline], [final], [virtual]
```

Parameters

drawer	drawer to draw into

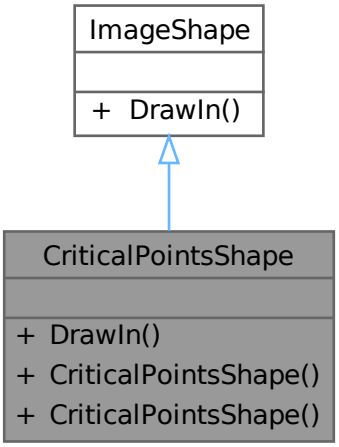
Implements ImageShape.

4.34 CriticalPointsShape Class Reference

It draws collection of CriticalPoint into image.

```
#include <CriticalPoint.h>
```

Collaboration diagram for CriticalPointsShape:



Public Member Functions

- void DrawIn (ImageShapeDrawer &drawer) const final

  It draws all CriticalPoint into image.
- CriticalPointsShape (const std::vector< std::shared_ptr< CriticalPoint > > &point, Radius radius, ImageShapeAttributes attr, unsigned int angleLength)

  Create CriticalPoint shape.
- CriticalPointsShape (const std::vector< std::shared_ptr< CriticalPoint > > &point, ImageShapeAttributes attr)

  Create CriticalPoint shape.

4.34.1 Constructor & Destructor Documentation

4.34.1.1 CriticalPointsShape() [1/2]

```
CriticalPointsShape (
            const std::vector< std::shared_ptr< CriticalPoint > > & point,
            Radius radius,
            ImageShapeAttributes attr,
            unsigned int angleLength)  [inline]
```

Parameters

point	point to draw
radius	radius of point
attr	ImageShapeAttributes of point
angleLength	length of angle line

4.34.1.2 CriticalPointsShape() [2/2]

```
CriticalPointsShape (
            const std::vector< std::shared_ptr< CriticalPoint > > & point,
            ImageShapeAttributes attr)  [inline]
```

Parameters

point	point to draw
attr	ImageShapeAttributes of point

4.34.2 Member Function Documentation

4.34.2.1 DrawIn()

```
void DrawIn (
            ImageShapeDrawer & drawer) const  [inline], [final], [virtual]
```

Parameters

drawer	drawer to draw into

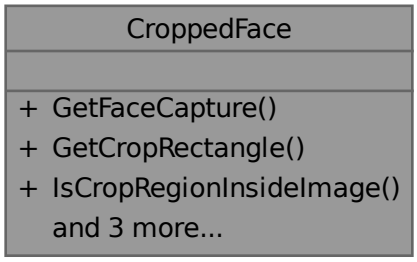
Implements ImageShape.

4.35 CroppedFace Class Reference

Represents a face cropped from the original face capture.

```
#include <Face.h>
```

Collaboration diagram for CroppedFace:



Public Member Functions

- std::shared_ptr< FaceCapture > GetFaceCapture () const

  Retrieves the face capture from the cropped region.
- Rectangle GetCropRectangle () const

  Gets the crop region of the face within the image.
- bool IsCropRegionInsideImage () const

  Checks if the crop region is entirely inside the image.
- std::shared_ptr< Image > FaceAreaMask () const

  Creates an image representing the mask of the detected face in the original FaceCapture, considering the applied FaceCropConfiguration.
- std::shared_ptr< Image > FaceAreaSegment (const RGB &color) const

  Creates an image of the detected face in the original FaceCapture, with the surrounding area replaced by a given color, considering the applied FaceCropConfiguration.
- std::shared_ptr< Image > FaceAreaSegment () const

  Creates an image of the detected face in the original FaceCapture, with a transparent surrounding area, considering the applied FaceCropConfiguration.

Friends

- class **Face**

4.35.1 Member Function Documentation

4.35.1.1 FaceAreaMask()

```
std::shared_ptr< Image > FaceAreaMask () const  [inline], [nodiscard]
```

Each pixel that belongs to face has white color, the surrounding has black color.

Returns

Image image with mask

Exceptions

EnrollmentException	when it is not possible to get mask

Required Solvers

Total size of solvers is 52 MB.

References face-segmentation-mask.solver

4.35.1.2 FaceAreaSegment() [1/2]

```
std::shared_ptr< Image > FaceAreaSegment () const  [inline], [nodiscard]
```

Returns

Image image with face and transparent surrounding

Exceptions

EnrollmentException	when it is not possible to get segmented face area

Required Solvers

Total size of solvers is 52 MB.

References face-segmentation-mask.solver

4.35.1.3 FaceAreaSegment() [2/2]

```
std::shared_ptr< Image > FaceAreaSegment (
            const RGB & color) const  [inline], [nodiscard]
```

Parameters

color	surrounding RGB

Returns

Image image with face and colored surrounding

Exceptions

EnrollmentException	when it is not possible to get segmented face area

Required Solvers

Total size of solvers is 52 MB.

References face-segmentation-mask.solver

4.35.1.4 GetCropRectangle()

```
Rectangle GetCropRectangle () const [inline], [nodiscard]
```

Returns

A Rectangle defining the cropped region.

4.35.1.5 GetFaceCapture()

```
std::shared_ptr< FaceCapture > GetFaceCapture () const [inline], [nodiscard]
```

Returns

FaceCapture instance of cropped image.

4.35.1.6 IsCropRegionInsideImage()

```
bool IsCropRegionInsideImage () const [inline], [nodiscard]
```

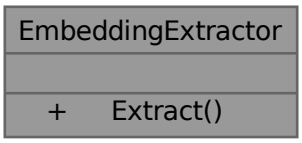
Returns

True if the crop region is within the image bounds, false otherwise.

4.36 EmbeddingExtractor Class Reference

```
#include <EmbeddingExtractor.h>
```

Collaboration diagram for EmbeddingExtractor:



Public Member Functions

- virtual std::shared_ptr< ExtractedPrintEmbedding > Extract (const std::shared_ptr< Print > &print) const =0

  Function creates embedding template for given Print.

4.36.1 Detailed Description

Note

Not supported in this variant of Enrollment SDK.

It is general extractor for Embedding templates.

4.36.2 Member Function Documentation

4.36.2.1 Extract()

```
virtual std::shared_ptr< ExtractedPrintEmbedding > Extract (
            const std::shared_ptr< Print > & print) const  [nodiscard], [pure virtual]
```

Note

Not supported in this variant of Enrollment SDK.

Parameters

print	Print to extract

Returns

extracted embedding

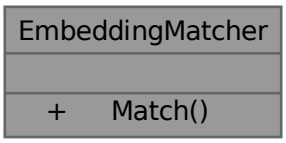
Implemented in IdentificationEmbeddingExtractor.

4.37 EmbeddingMatcher Class Reference

It is general matcher for Embedding templates.

```
#include <EmbeddingMatcher.h>
```

Collaboration diagram for EmbeddingMatcher:



Public Member Functions

- virtual PrintSimilarityScore Match (const EmbeddingTemplate &probe, const EmbeddingTemplate &gallery) const =0

  Calculates similarity score of probe and gallery template.

4.37.1 Detailed Description

It compares probe and gallery template. Result of comparison is SimilarityScore.

4.37.2 Member Function Documentation

4.37.2.1 Match()

```
virtual PrintSimilarityScore Match (
            const EmbeddingTemplate & probe,
            const EmbeddingTemplate & gallery) const  [nodiscard], [pure virtual]
```

Parameters

probe	probe Template
gallery	gallery Template

Returns

similarity SimilarityScore

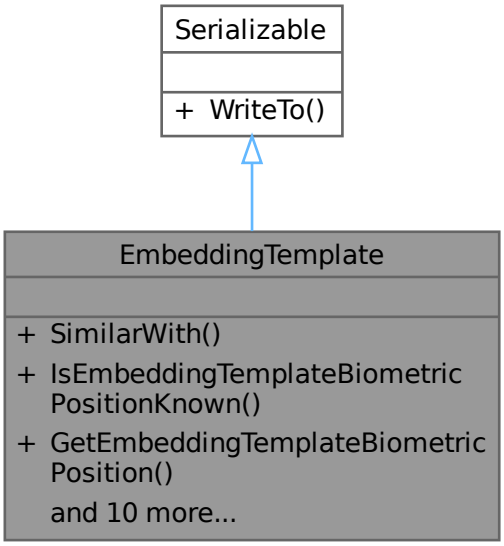
Implemented in EmbeddingVerifyMatcher.

4.38 EmbeddingTemplate Class Reference

Print template in proprietary embedding format used for biometric operations.

```
#include <EmbeddingTemplate.h>
```

Collaboration diagram for EmbeddingTemplate:



Public Member Functions

- PrintSimilarityScore SimilarWith (const EmbeddingTemplate &tmpl, const EmbeddingMatcher &matcher) const

  It calculates similarity score between this and given EmbeddingTemplate.
- bool IsEmbeddingTemplateBiometricPositionKnown () const

  It checks if template contains known biometric position.
- PrintPosition GetEmbeddingTemplateBiometricPosition () const

  It returns biometric position to which template belongs.
- std::string GetVersion () const

  Provides version information of Innovatrics proprietary algorithm used to create the embedding template.
- EmbeddingTemplate (std::vector< uint8_t > &&d)

  Create EmbeddingTemplate object.
- EmbeddingTemplate (const std::vector< uint8_t > &d)

  Create EmbeddingTemplate object that copies the data.
- EmbeddingTemplate (std::vector< uint8_t > &&d, std::shared_ptr< PrintExecutor > executor)

  Create EmbeddingTemplate object.
- EmbeddingTemplate (const std::vector< uint8_t > &d, std::shared_ptr< PrintExecutor > executor)

  Create EmbeddingTemplate object that copies data.
- EmbeddingTemplate (std::vector< uint8_t > &&d, PrintPosition position)

  Create EmbeddingTemplate object.
- EmbeddingTemplate (const std::vector< uint8_t > &d, PrintPosition position)

  Create EmbeddingTemplate object that copies the data.
- EmbeddingTemplate (std::vector< uint8_t > &&d, PrintPosition position, std::shared_ptr< PrintExecutor > executor)

  Create EmbeddingTemplate object.

- EmbeddingTemplate (const std::vector< uint8_t > &d, PrintPosition position, std::shared_ptr< PrintExecutor > executor)

  Create EmbeddingTemplate object copy data.
- void WriteTo (Writer &w) const override

  Write itself into writer.

4.38.1 Constructor & Destructor Documentation

4.38.1.1 EmbeddingTemplate() [1/8]

```
EmbeddingTemplate (
            std::vector< uint8_t > && d)  [inline], [explicit]
```

GlobalPrintExecutor is used to execute template functionality. Biometric position is unknown.

Parameters

d	template data

Exceptions

InvalidTemplateException	when Embedding template has no data

4.38.1.2 EmbeddingTemplate() [2/8]

```
EmbeddingTemplate (
            const std::vector< uint8_t > & d)  [inline], [explicit]
```

GlobalPrintExecutor is used to execute template functionality. Biometric position is unknown.

Parameters

d	template data

Exceptions

InvalidTemplateException	when Embedding template has no data

4.38.1.3 EmbeddingTemplate() [3/8]

```
EmbeddingTemplate (
            std::vector< uint8_t > && d,
            std::shared_ptr< PrintExecutor > executor)  [inline]
```

Biometric position is unknown.

Parameters

d	template data
executor	PrintExecutor used to execute template functionality.

Exceptions

NullArgumentException	when executor is null
InvalidTemplateException	when Embedding template has no data

4.38.1.4 EmbeddingTemplate() [4/8]

```
EmbeddingTemplate (
            const std::vector< uint8_t > & d,
            std::shared_ptr< PrintExecutor > executor)  [inline]
```

Biometric position is unknown.

Parameters

d	template data
executor	PrintExecutor used to execute template functionality.

Exceptions

NullArgumentException	when executor is null
InvalidTemplateException	when Embedding template has no data

4.38.1.5 EmbeddingTemplate() [5/8]

```
EmbeddingTemplate (
            std::vector< uint8_t > && d,
            PrintPosition position)  [inline], [explicit]
```

GlobalPrintExecutor is used to execute template functionality.

Parameters

d	template data
position	biometric position to which template belongs

Exceptions

InvalidTemplateException	when Embedding template has no data

4.38.1.6 EmbeddingTemplate() [6/8]

```
EmbeddingTemplate (
            const std::vector< uint8_t > & d,
            PrintPosition position)  [inline], [explicit]
```

GlobalPrintExecutor is used to execute template functionality.

Parameters

d	template data
position	biometric position to which template belongs

Exceptions

InvalidTemplateException	when Embedding template has no data

4.38.1.7 EmbeddingTemplate() [7/8]

```
EmbeddingTemplate (
            std::vector< uint8_t > && d,
            PrintPosition position,
            std::shared_ptr< PrintExecutor > executor)  [inline]
```

Parameters

d	template data
position	biometric position to which template belongs
executor	PrintExecutor used to execute template functionality.

Exceptions

NullArgumentException	when executor is null
InvalidTemplateException	when Embedding template has no data

4.38.1.8 EmbeddingTemplate() [8/8]

```
EmbeddingTemplate (
            const std::vector< uint8_t > & d,
            PrintPosition position,
            std::shared_ptr< PrintExecutor > executor)  [inline]
```

Parameters

d	template data
position	biometric position to which template belongs
executor	PrintExecutor used to execute template functionality.

Exceptions

NullArgumentException	when executor is null
InvalidTemplateException	when Embedding template has no data

4.38.2 Member Function Documentation

4.38.2.1 GetEmbeddingTemplateBiometricPosition()

```
PrintPosition GetEmbeddingTemplateBiometricPosition () const  [inline], [nodiscard]
```

Returns

biometric position

Exceptions

EnrollmentRuntimeException	when biometric position is unknown

4.38.2.2 GetVersion()

```
std::string GetVersion () const  [inline], [nodiscard]
```

Returns

version string

Exceptions

InvalidTemplateException	when it is not possible to extract version from Embedding template data

4.38.2.3 IsEmbeddingTemplateBiometricPositionKnown()

```
bool IsEmbeddingTemplateBiometricPositionKnown () const [inline], [nodiscard]
```

Returns

`true` when biometric position is known and valid

`false` when biometric position is unknown

4.38.2.4 SimilarWith()

```
PrintSimilarityScore SimilarWith (
            const EmbeddingTemplate & tmpl,
            const EmbeddingMatcher & matcher) const [inline], [nodiscard]
```

It uses given EmbeddingMatcher to calculate score.

Parameters

tmpl	gallery EmbeddingTemplate.
matcher	is EmbeddingMatcher used to calculate similarity score between probe (this template) and given template `tmpl`

Returns

similarity score

4.38.2.5 WriteTo()

```
void WriteTo (
            Writer & writer) const [inline], [override], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

Implements Serializable.

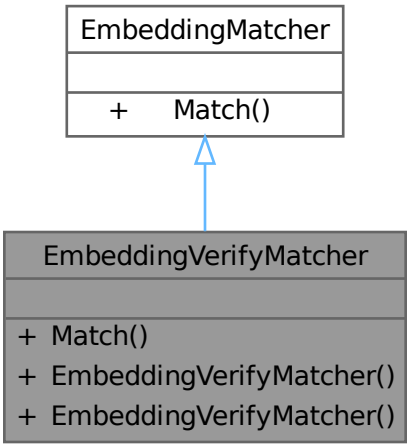
Reimplemented in ExtractedPrintEmbedding.

4.39 EmbeddingVerifyMatcher Class Reference

EmbeddingVerifyMatcher is template matcher that uses Innovatrics proprietary verification algorithm.

`#include <EmbeddingVerifyMatcher.h>`

Collaboration diagram for EmbeddingVerifyMatcher:



Public Member Functions

- PrintSimilarityScore Match (const EmbeddingTemplate &probe, const EmbeddingTemplate &gallery) const final

  Calculates similarity score of probe and gallery template.
- EmbeddingVerifyMatcher ()

  Constructor with default logger, executor and matcher module GlobalPrintExecutor is used to execute matching.
- EmbeddingVerifyMatcher (std::shared_ptr< PrintExecutor > executor)

  Constructor with custom executor .

4.39.1 Detailed Description

Note

Not supported in this variant of Enrollment SDK.

The accuracy of the matcher is approximately half that of the standard ICSVerifyMatcher. We strongly recommend using the more accurate ICSVerifyMatcher for consolidating identification results.

4.39.2 Constructor & Destructor Documentation

4.39.2.1 EmbeddingVerifyMatcher() [1/2]

`EmbeddingVerifyMatcher () [inline]`

Note

Not supported in this variant of Enrollment SDK.

4.39.2.2 EmbeddingVerifyMatcher() [2/2]

```
EmbeddingVerifyMatcher (
            std::shared_ptr< PrintExecutor > executor)  [inline], [explicit]
```

Note

Not supported in this variant of Enrollment SDK.

Parameters

executor	PrintExecutor used for matching

Exceptions

NullArgumentException	when executor is null

4.39.3 Member Function Documentation

4.39.3.1 Match()

```
PrintSimilarityScore Match (
            const EmbeddingTemplate & probe,
            const EmbeddingTemplate & gallery) const  [inline], [nodiscard], [final],
[virtual]
```

Note

Not supported in this variant of Enrollment SDK.

The embedding extraction feature MUST be allowed in license to match embeddings.

The accuracy of the matcher is approximately half that of the standard ICSVerifyMatcher. We strongly recommend using the more accurate ICSVerifyMatcher for consolidating identification results.

Parameters

probe	probe EmbeddingTemplate
gallery	gallery EmbeddingTemplate

Returns

embedding similarity score PrintSimilarityScore

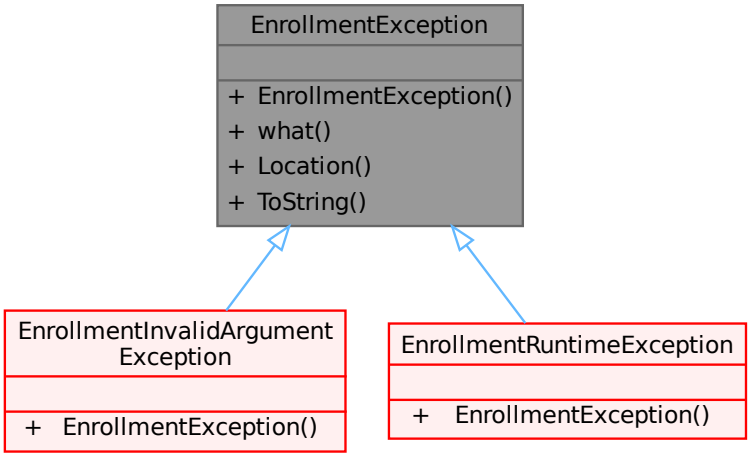
Implements EmbeddingMatcher.

4.40 EnrollmentException Class Reference

Base exception for all enrollment-sdk exceptions.

```
#include <EnrollmentException.h>
```

Inheritance diagram for EnrollmentException:



Public Member Functions

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

  Provides message with full info about exception.

4.40.1 Constructor & Destructor Documentation

4.40.1.1 EnrollmentException()

```
EnrollmentException (
            std::string reason,
            SourceLocation location = SourceLocation())  [inline], [explicit]
```

Constructor of EnrollmentException exception. This exception is used for general errors or as a base class for custom exceptions of the enrollment SDK.

Parameters

reason	error reason
location	Source location of where the exception was thrown

4.40.2 Member Function Documentation

4.40.2.1 Location()

```
SourceLocation Location () const  [inline], [nodiscard], [noexcept]
```

Returns

error location

4.40.2.2 ToString()

```
std::string ToString () const  [inline], [nodiscard]
```

Returns

exception data as string

4.40.2.3 what()

```
const char * what () const  [inline], [nodiscard], [override], [noexcept]
```

Standard exception error message interface which is usually used for printing message when exception is caught in user application.

Returns

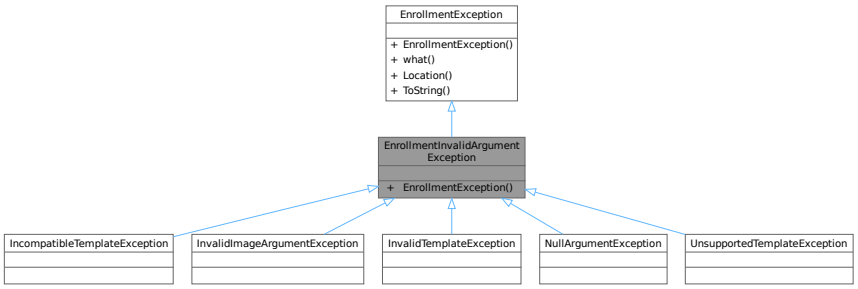
error message

4.41 EnrollmentInvalidArgumentException Class Reference

Base exception for all enrollment-sdk invalid argument exceptions.

```
#include <EnrollmentException.h>
```

Inheritance diagram for EnrollmentInvalidArgumentException:



Public Member Functions

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

  Provides message with full info about exception.

4.41.1 Member Function Documentation

4.41.1.1 EnrollmentException()

```
EnrollmentException (
            std::string reason,
            SourceLocation location = SourceLocation())  [inline], [explicit]
```

Constructor of EnrollmentException exception. This exception is used for general errors or as a base class for custom exceptions of the enrollment SDK.

Parameters

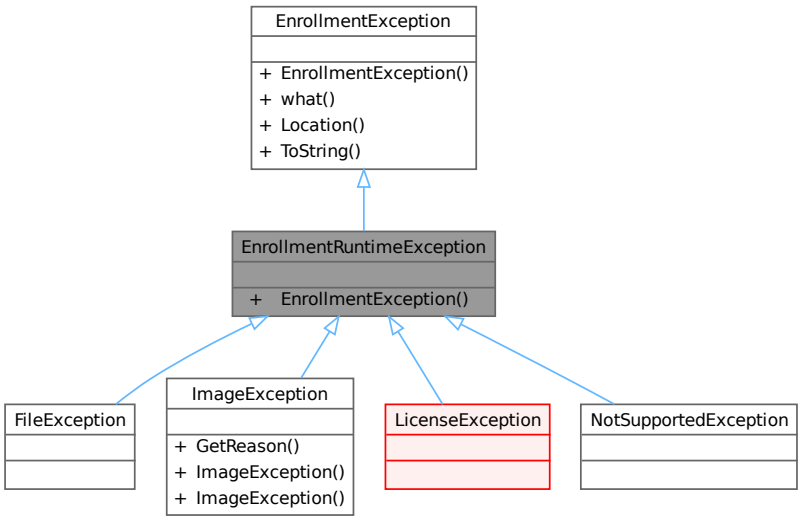
reason	error reason
location	Source location of where the exception was thrown

4.42 EnrollmentRuntimeException Class Reference

Base exception for all enrollment-sdk runtime exceptions.

```
#include <EnrollmentException.h>
```

Inheritance diagram for EnrollmentRuntimeException:



Public Member Functions

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char ∗ what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

  Provides message with full info about exception.

4.42.1 Member Function Documentation

4.42.1.1 EnrollmentException()

```
EnrollmentException (
            std::string reason,
            SourceLocation location = SourceLocation())  [inline], [explicit]
```

Constructor of EnrollmentException exception. This exception is used for general errors or as a base class for custom exceptions of the enrollment SDK.

Parameters

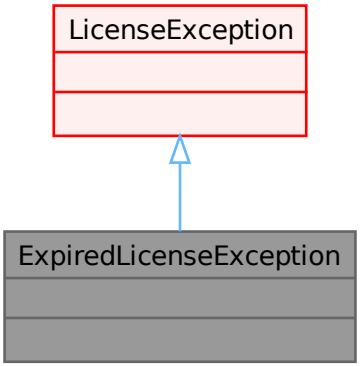
reason	error reason
location	Source location of where the exception was thrown

4.43 ExpiredLicenseException Class Reference

License is expired.

```
#include <LicenseException.h>
```

Inheritance diagram for ExpiredLicenseException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentRuntimeException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char ∗ what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

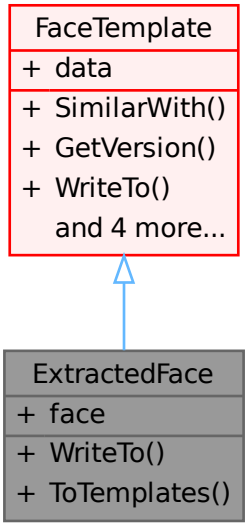
  Provides message with full info about exception.

4.44 ExtractedFace Class Reference

ExtractedFace is ICF Template created by FaceExtractor with Face that was extracted.

```
#include <Face.h>
```

Collaboration diagram for ExtractedFace:



Public Member Functions

- void WriteTo (Writer &writer) const final

  Function serializes ExtractedFace via provided Writer.

Public Member Functions inherited from FaceTemplate

- FaceSimilarityScore SimilarWith (const FaceTemplate &gallery) const

  Calculates FaceSimilarityScore for given FaceTemplate.
- std::string GetVersion () const

  Provides version information of Innovatrics proprietary algorithm used to create the face template.
- void WriteTo (Writer &writer) const override

  Function serializes FaceTemplate via provided Writer.
- FaceTemplate (std::vector< uint8_t > &&templ, std::shared_ptr< FaceExecutor > executor)

  Construct a new FaceTemplate object.
- FaceTemplate (std::vector< uint8_t > &&templ)

  Construct a new FaceTemplate object GlobalFaceExecutor is used.
- FaceTemplate (const std::vector< uint8_t > &templ, std::shared_ptr< FaceExecutor > executor)

  Construct a new FaceTemplate object, copy template data.
- FaceTemplate (const std::vector< uint8_t > &templ)

  Construct a new FaceTemplate object, copy template data GlobalFaceExecutor is used.

Static Public Member Functions

- static std::vector< std::shared_ptr< FaceTemplate > > ToTemplates (const std::vector< std::shared_ptr< ExtractedFace > > &faces)

  Converts a vector of ExtractedFace objects into a vector of FaceTemplate objects.

Public Attributes

- const std::shared_ptr< Face > **face**

  Face that is used to template extraction.

Public Attributes inherited from FaceTemplate

- const std::vector< uint8_t > **data**

  Template data.

Friends

- class **Face**

4.44.1 Detailed Description

The template can be used later for verification.

4.44.2 Member Function Documentation

4.44.2.1 ToTemplates()

```
std::vector< std::shared_ptr< FaceTemplate > > ToTemplates (
            const std::vector< std::shared_ptr< ExtractedFace > > & faces) [inline],
[static]
```

This function takes a vector of ExtractedFace objects and converts each element into an FaceTemplate object, then returns a vector of FaceTemplate objects.

Parameters

faces	A vector of ExtractedFace objects to be converted.

Returns

A vector of FaceTemplate objects converted from the ExtractedFace objects.

4.44.2.2 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Face.

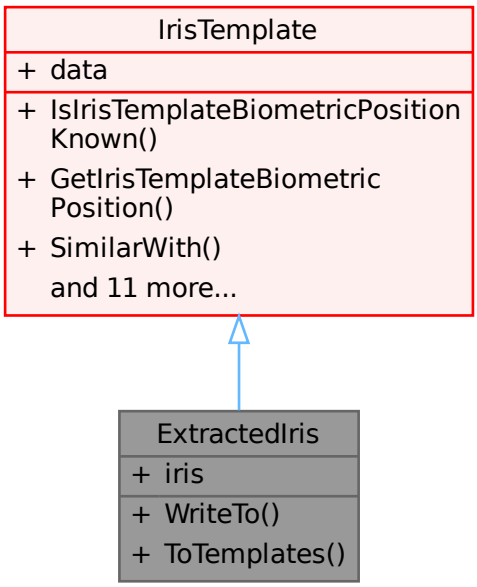
Implements Serializable.

4.45 ExtractedIris Class Reference

ExtractedIris is template created by IrisExtractor with Iris that belongs to template.

```
#include <Iris.h>
```

Collaboration diagram for ExtractedIris:



Public Member Functions

- void WriteTo (Writer &writer) const final

  Function serializes ExtractedIris via provided Writer.

Public Member Functions inherited from IrisTemplate

- bool IsIrisTemplateBiometricPositionKnown () const

  It checks if template contains known biometric position.
- IrisPosition GetIrisTemplateBiometricPosition () const

  It returns biometric position to which template belongs.
- IrisSimilarityScore SimilarWith (const IrisTemplate &gallery) const

  Calculates IrisSimilarityScore for given IrisTemplate.
- unsigned int CalculateQuality ()

  Calculates quality of IrisTemplate.
- std::string GetVersion () const

  Provides version information of Innovatrics proprietary algorithm used to create the iris template.
- void WriteTo (Writer &writer) const override

  Function serializes IrisTemplate via provided Writer.
- IrisTemplate (std::vector< uint8_t > &&templ, std::shared_ptr< IrisExecutor > executor)

  Construct a new IrisTemplate object Biometric position is unknown.
- IrisTemplate (std::vector< uint8_t > &&templ)

  Construct a new IrisTemplate object Biometric position is unknown.
- IrisTemplate (const std::vector< uint8_t > &templ, std::shared_ptr< IrisExecutor > executor)

  Construct a new IrisTemplate object, copy template data Biometric position is unknown.
- IrisTemplate (const std::vector< uint8_t > &templ)

  Construct a new IrisTemplate object, copy template data Biometric position is unknown.
- IrisTemplate (std::vector< uint8_t > &&templ, IrisPosition position, std::shared_ptr< IrisExecutor > executor)

  Construct a new IrisTemplate object.
- IrisTemplate (std::vector< uint8_t > &&templ, IrisPosition position)

  Construct a new IrisTemplate object GlobalIrisExecutor is used.
- IrisTemplate (const std::vector< uint8_t > &templ, IrisPosition position, std::shared_ptr< IrisExecutor > executor)

  Construct a new IrisTemplate object, copy template data.
- IrisTemplate (const std::vector< uint8_t > &templ, IrisPosition position)

  Construct a new IrisTemplate object, copy template data GlobalIrisExecutor is used.

Static Public Member Functions

- static std::vector< std::shared_ptr< IrisTemplate > > ToTemplates (const std::vector< std::shared_ptr< ExtractedIris > > &irises)

  Converts a vector of ExtractedIris objects into a vector of IrisTemplate objects.

Public Attributes

- const std::shared_ptr< Iris > **iris**

  Iris that is used to template extraction.

Public Attributes inherited from IrisTemplate

- const std::vector< uint8_t > **data**

  Template data.

Friends

- class **Iris**

4.45.1 Detailed Description

The template can be used later for verification.

4.45.2 Member Function Documentation

4.45.2.1 ToTemplates()

```
std::vector< std::shared_ptr< IrisTemplate > > ToTemplates (
            const std::vector< std::shared_ptr< ExtractedIris > > & irises) [inline],
[static]
```

This function takes a vector of ExtractedIris objects and converts each element into an IrisTemplate object, then returns a vector of IrisTemplate objects.

Parameters

irises	A vector of ExtractedIris objects to be converted.

Returns

A vector of IrisTemplate objects converted from the ExtractedIris objects.

4.45.2.2 WriteTo()

```
void WriteTo (
            Writer & writer) const [inline], [final], [virtual]
```

Parameters

writer	used to serialize Iris.

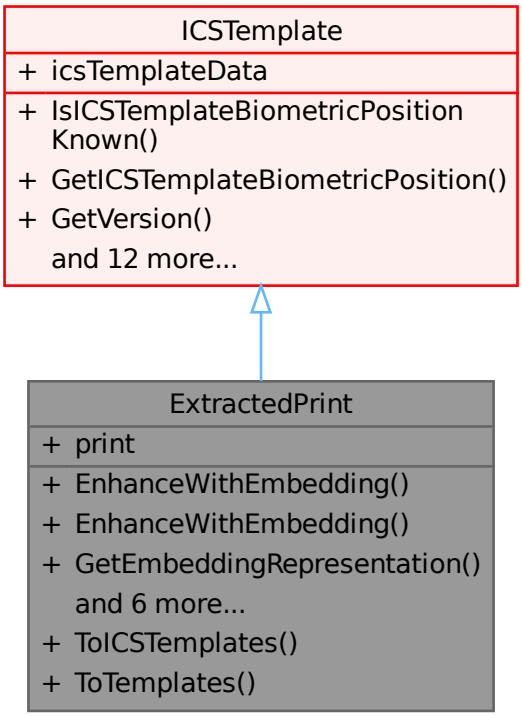
Implements Serializable.

4.46 ExtractedPrint Class Reference

Print template in ICS format used for biometric operations with Print that was extracted.

```
#include <ExtractedPrint.h>
```

Collaboration diagram for ExtractedPrint:



Public Member Functions

- void EnhanceWithEmbedding (std::shared_ptr< ExtractedPrintEmbedding > embedding)

  Enhances the object with an embedding representation.
- void EnhanceWithEmbedding (const EmbeddingExtractor &extractor)

  Enhances the object with an embedding representation using an embedding extractor.
- std::shared_ptr< ExtractedPrintEmbedding > GetEmbeddingRepresentation () const

  Retrieves the embedding representation associated with the object.
- bool HasEmbeddingRepresentation () const

  Checks if the object has an embedding representation.
- ExtractedPrint (std::vector< uint8_t > &&d, std::shared_ptr< Print > fp)

  Create ExtractedPrint object.
- ExtractedPrint (std::vector< uint8_t > &&d, std::shared_ptr< Print > fp, std::shared_ptr< PrintExecutor > executor)

  Create ExtractedPrint object.
- std::shared_ptr< ISOTemplate > ToISO () const

Convert this ExtractedPrint to into ISO/IEC 19794-2 compliant template.

- std::shared_ptr< ANSITemplate > ToANSI () const

  Convert this ExtractedPrint to into ANSI/INCITS 378 compliant template.
- void WriteTo (Writer &w) const final

  Write itself into writer.

Public Member Functions inherited from ICSTemplate

- bool IsICSTemplateBiometricPositionKnown () const

  It checks if template contains known biometric position.
- PrintPosition GetICSTemplateBiometricPosition () const

  It returns biometric position to which template belongs.
- std::string GetVersion () const

  Provides version information of Innovatrics proprietary algorithm used to create the ICS template.
- std::shared_ptr< ICSTemplate > Anonymize () const

  Creates ICSTemplate without position.
- ICSTemplate (std::vector< uint8_t > d)

  Create ICSTemplate object GlobalPrintExecutor is used to execute template functionality.
- ICSTemplate (std::vector< uint8_t > d, std::shared_ptr< PrintExecutor > executor)

  Create ICSTemplate object Biometric position is loaded from template is possible.
- ICSTemplate (std::vector< uint8_t > d, PrintPosition position)

  Create ICSTemplate object GlobalPrintExecutor is used to execute template functionality.
- ICSTemplate (std::vector< uint8_t > d, PrintPosition position, std::shared_ptr< PrintExecutor > executor)

  Create ICSTemplate object The template format is validated.
- PrintSimilarityScore SimilarWith (const ICSTemplate &tmpl, const ICSMatcher &matcher) const

  It calculates similarity score between this and given ICSTemplate.
- uint8_t CalculateQuality ()

  Function calculates a fingerprint template quality.
- std::shared_ptr< ISOTemplate > ToISO (const std::shared_ptr< const Print > &print) const

  Convert this ICSTemplate to into ISO/IEC 19794-2 compliant template.
- std::shared_ptr< ANSITemplate > ToANSI (const std::shared_ptr< const Print > &print) const

  Convert this ICSTemplate to into ANSI/INCITS 378 compliant template.
- std::vector< std::shared_ptr< Minutia > > GetMinutiae () const

  Get minutiae from template.
- std::vector< std::shared_ptr< CriticalPoint > > GetCriticalPoints () const

  Get critical points (core/delta) from template.
- void WriteTo (Writer &w) const override

  Write itself into writer.

Static Public Member Functions

- static std::vector< std::shared_ptr< ICSTemplate > > ToICSTemplates (const std::vector< std::shared_ptr< ExtractedPrint > > &prints)

  Converts a vector of ExtractedPrint objects into a vector of ICSTemplate objects.
- static std::vector< std::shared_ptr< PrintTemplate > > ToTemplates (const std::vector< std::shared_ptr< ExtractedPrint > > &prints)

  Converts a vector of ExtractedPrint objects into a vector of PrintTemplate objects.

Public Attributes

- const std::shared_ptr< Print > **print**

  Print that was extracted.

Public Attributes inherited from ICSTemplate

- const std::vector< uint8_t > **icsTemplateData**

  template data

4.46.1 Constructor & Destructor Documentation

4.46.1.1 ExtractedPrint() [1/2]

```
ExtractedPrint (
            std::vector< uint8_t > && d,
            std::shared_ptr< Print > fp)  [inline]
```

Parameters

d	template data
fp	fingerprint that belongs to template

Exceptions

NullArgumentException	when Print is null

4.46.1.2 ExtractedPrint() [2/2]

```
ExtractedPrint (
            std::vector< uint8_t > && d,
            std::shared_ptr< Print > fp,
            std::shared_ptr< PrintExecutor > executor)  [inline]
```

Parameters

d	template data
fp	fingerprint that belongs to template
executor	PrintExecutor used to execute template functionality.

Exceptions

NullArgumentException	when executor is null
EnrollmentRuntimeException	when Print is null

4.46.2 Member Function Documentation

4.46.2.1 EnhanceWithEmbedding() [1/2]

```
void EnhanceWithEmbedding (
            const EmbeddingExtractor & extractor)  [inline]
```

Note

Not supported in this variant of Enrollment SDK.

This function enhances the object with an embedding representation obtained from the provided embedding extractor.

Parameters

extractor	An instance of the EmbeddingExtractor used to extract the embedding.

Exceptions

EnrollmentException	Thrown if an error occurs during the extraction process.

4.46.2.2 EnhanceWithEmbedding() [2/2]

```
void EnhanceWithEmbedding (
            std::shared_ptr< ExtractedPrintEmbedding > embedding)  [inline]
```

Note

Not supported in this variant of Enrollment SDK.

This function enhances the object with the provided embedding representation.

Parameters

embedding	A ExtractedPrintEmbedding to be associated with the object.

Exceptions

NullArgumentException	thrown if the embedding is null
EnrollmentInvalidArgumentException	thrown if the print associated with the embedding does not match the print associated with the object

4.46.2.3 GetEmbeddingRepresentation()

```
std::shared_ptr< ExtractedPrintEmbedding > GetEmbeddingRepresentation () const [inline]
```

This function retrieves the embedding representation associated with the object.

Returns

A ExtractedPrintEmbedding associated with the object.

A nullptr if no embedding representation is associated.

4.46.2.4 HasEmbeddingRepresentation()

```
bool HasEmbeddingRepresentation () const [inline]
```

This function checks whether the object has an embedding representation associated with it.

Returns

`true` if the object has an embedding representation, `false` otherwise.

4.46.2.5 ToANSI()

```
std::shared_ptr< ANSITemplate > ToANSI () const [inline]
```

Returns

ANSITemplate

Exceptions

EnrollmentException	when it is not possible to convert template

4.46.2.6 ToICSTemplates()

```
std::vector< std::shared_ptr< ICSTemplate > > ToICSTemplates (
            const std::vector< std::shared_ptr< ExtractedPrint > > & prints) [inline],
[static]
```

This function takes a vector of ExtractedPrint objects and converts each element into an ICSTemplate object, then returns a vector of ICSTemplate objects.

Parameters

prints	A vector of ExtractedPrint objects to be converted.

Returns

A vector of ICSTemplate objects converted from the ExtractedPrint objects.

4.46.2.7 ToISO()

```
std::shared_ptr< ISOTemplate > ToISO () const  [inline]
```

Returns

ISOTemplate

Exceptions

EnrollmentException	when it is not possible to convert template

4.46.2.8 ToTemplates()

```
std::vector< std::shared_ptr< PrintTemplate > > ToTemplates (
            const std::vector< std::shared_ptr< ExtractedPrint > > & prints)  [inline],
[static]
```

This function takes a vector of ExtractedPrint objects and converts each element into an PrintTemplate object, then returns a vector of PrintTemplate objects.

Parameters

prints	A vector of ExtractedPrint objects to be converted.

Returns

A vector of PrintTemplate objects converted from the ExtractedPrint objects.

4.46.2.9 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

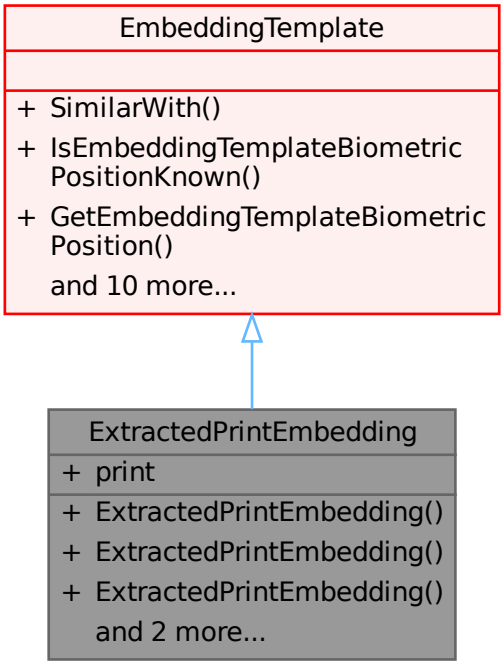
Implements Serializable.

4.47 ExtractedPrintEmbedding Class Reference

Print template in proprietary embedding format used for biometric operations with Print that was extracted.

```
#include <ExtractedPrintEmbedding.h>
```

Collaboration diagram for ExtractedPrintEmbedding:



Public Member Functions

- ExtractedPrintEmbedding (const std::vector< uint8_t > &serializedTmpl, std::shared_ptr< Print > fp)

  Create ExtractedPrintEmbedding object .
- ExtractedPrintEmbedding (const std::vector< uint8_t > &tmpl, std::shared_ptr< Print > fp, std::shared_ptr< PrintExecutor > executor)

  Create ExtractedPrintEmbedding object .
- ExtractedPrintEmbedding (std::vector< uint8_t > &&tmpl, std::shared_ptr< Print > fp)

  Create ExtractedPrintEmbedding object .
- ExtractedPrintEmbedding (std::vector< uint8_t > &&tmpl, std::shared_ptr< Print > fp, std::shared_ptr< Print↩ Executor > executor)

  Create ExtractedPrintEmbedding object .
- void WriteTo (Writer &w) const final

  Write itself into writer.

Public Member Functions inherited from EmbeddingTemplate

- PrintSimilarityScore SimilarWith (const EmbeddingTemplate &tmpl, const EmbeddingMatcher &matcher) const

  It calculates similarity score between this and given EmbeddingTemplate.
- bool IsEmbeddingTemplateBiometricPositionKnown () const

  It checks if template contains known biometric position.
- PrintPosition GetEmbeddingTemplateBiometricPosition () const

  It returns biometric position to which template belongs.
- std::string GetVersion () const

  Provides version information of Innovatrics proprietary algorithm used to create the embedding template.
- EmbeddingTemplate (std::vector< uint8_t > &&d)

  Create EmbeddingTemplate object.
- EmbeddingTemplate (const std::vector< uint8_t > &d)

  Create EmbeddingTemplate object that copies the data.
- EmbeddingTemplate (std::vector< uint8_t > &&d, std::shared_ptr< PrintExecutor > executor)

  Create EmbeddingTemplate object.
- EmbeddingTemplate (const std::vector< uint8_t > &d, std::shared_ptr< PrintExecutor > executor)

  Create EmbeddingTemplate object that copies data.
- EmbeddingTemplate (std::vector< uint8_t > &&d, PrintPosition position)

  Create EmbeddingTemplate object.
- EmbeddingTemplate (const std::vector< uint8_t > &d, PrintPosition position)

  Create EmbeddingTemplate object that copies the data.
- EmbeddingTemplate (std::vector< uint8_t > &&d, PrintPosition position, std::shared_ptr< PrintExecutor > executor)

  Create EmbeddingTemplate object.
- EmbeddingTemplate (const std::vector< uint8_t > &d, PrintPosition position, std::shared_ptr< PrintExecutor > executor)

  Create EmbeddingTemplate object copy data.

Public Attributes

- const std::shared_ptr< Print > **print**

  Print that was extracted.

4.47.1 Detailed Description

Note

Not supported in this variant of Enrollment SDK.

4.47.2 Constructor & Destructor Documentation

4.47.2.1 ExtractedPrintEmbedding() [1/4]

```
ExtractedPrintEmbedding (
            const std::vector< uint8_t > & serializedTmpl,
            std::shared_ptr< Print > fp)  [inline]
```

Note

Not supported in this variant of Enrollment SDK.

GlobalPrintExecutor is used to execute template functionality.

Parameters

serializedTmpl	serialized template data
fp	fingerprint that belongs to template

Exceptions

EnrollmentRuntimeException	when Print is null

4.47.2.2 ExtractedPrintEmbedding() [2/4]

```
ExtractedPrintEmbedding (
            const std::vector< uint8_t > & tmpl,
            std::shared_ptr< Print > fp,
            std::shared_ptr< PrintExecutor > executor)  [inline]
```

Note

Not supported in this variant of Enrollment SDK.

Parameters

tmpl	pointer to template
fp	fingerprint that belongs to template
executor	PrintExecutor used to execute template functionality.

Exceptions

NullArgumentException	when executor is null
EnrollmentRuntimeException	when Print is null

4.47.2.3 ExtractedPrintEmbedding() [3/4]

```
ExtractedPrintEmbedding (
            std::vector< uint8_t > && tmpl,
            std::shared_ptr< Print > fp)  [inline]
```

Note

Not supported in this variant of Enrollment SDK.

GlobalPrintExecutor is used to execute template functionality.

Parameters

tmpl	pointer to template
fp	fingerprint that belongs to template

Exceptions

EnrollmentRuntimeException	when Print is null

4.47.2.4 ExtractedPrintEmbedding() [4/4]

```
ExtractedPrintEmbedding (
            std::vector< uint8_t > && tmpl,
            std::shared_ptr< Print > fp,
            std::shared_ptr< PrintExecutor > executor)  [inline]
```

Note

Not supported in this variant of Enrollment SDK.

Parameters

tmpl	pointer to template
fp	fingerprint that belongs to template
executor	PrintExecutor used to execute template functionality.

Exceptions

NullArgumentException	when executor is null
EnrollmentRuntimeException	when Print is null

4.47.3 Member Function Documentation

4.47.3.1 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

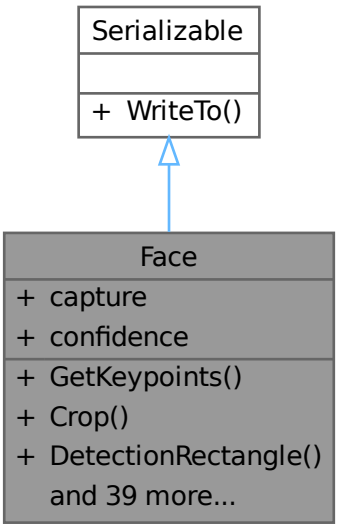
Reimplemented from EmbeddingTemplate.

4.48 Face Class Reference

It holds all functions and data related to Face manipulation.

```
#include <Face.h>
```

Collaboration diagram for Face:



Classes

- class Keypoint

  It holds all attributes related to a facial key point.
- class Keypoints

  It holds all facial keypoints detected on a FaceCapture image.

Public Types

- enum class AgeEstimationMode : uint8_t { BALANCED = 1 , ACCURATE = 2 }

  Specifies the level of compromise between age estimation speed and accuracy.
- enum class GenderEstimationMode : uint8_t { BALANCED = 1 , ACCURATE = 2 }

  Specifies the level of compromise between age estimation speed and accuracy.
- enum class PassiveLivenessMode : uint8_t { FAST , ACCURATE }

  Mode of passive liveness calculation.
- enum class AttributeID : uint8_t { CONFIDENCE , SHARPNESS , BRIGHTNESS , CONTRAST , UNIQUE_INTENSITY_LEVELS , SHADOW , NOSE_SHADOW , SPECULARITY , EYE_GAZE , RIGHT_EYE_STATUS , LEFT_EYE_STATUS , GLASS_STATUS , HEAVY_FRAME , MOUTH_STATUS , BACKGROUND_UNIFORMITY , AGE , RIGHT_RED_EYE , LEFT_RED_EYE , GENDER , EYE_DISTANCE , ROLL_ANGLE , PITCH_ANGLE , YAW_ANGLE , FACE_SIZE , FACE_RELATIVE_AREA , FACE_RELATIVE_AREA_IN_IMAGE , TINTED_GLASSES , WIDTH_HEIGHT_RATIO }

  Face attributes IDs.

Public Member Functions

- std::shared_ptr< Keypoints > GetKeypoints () const

  Provides the facial keypoints detected on the face in the original face capture.
- std::shared_ptr< CroppedFace > Crop (FaceCropConfiguration cfg) const

  Crops the detected face from the original face capture.
- Rectangle DetectionRectangle () const

  Retrieves the bounding box of the detected face.
- std::vector< uint8_t > ToIsoImage (ImageEncoder &encoder) const

  Function creates ISO image based on ISO/IEC 19794-5 (Face image data).
- std::shared_ptr< ExtractedFace > Extract (const FaceExtractorConfig &cfg)

  This function creates a template for the detected face.
- std::shared_ptr< ExtractedFace > Extract ()

  This function creates a template for the detected face with default FaceExtractorConfig.
- void WriteTo (Writer &writer) const final

  Function serializes Face via provided Writer.
- bool IsSameAs (const Face &face) const

  Returns `true` when argument is same object.
- bool operator== (const Face &f) const

  Compares if Face is equal with other one.
- bool operator!= (const Face &f) const

  Compares if Face is not equal with other one.
- float Sharpness ()

  Sharpness attribute used to measure the sharpness level of an area of the detected face in the original face capture.
- float Brightness ()

  Brightness attribute used to measure the brightness level of an area of the detected face in the original face capture.
- float Contrast ()

  Contrast attribute used to measure the contrast level of an area of the detected face in the original face capture.
- float UniqueIntensityLevels ()

  Unique Intensity Levels attribute used to measure whether an area of the detected face in the original face capture has an appropriate number of unique intensity levels.
- float Shadow ()

Shadow attribute used to evaluate whether an area of the detected face in the original face capture is affected by shadows.

- float NoseShadow ()

  Nose Shadow attribute for evaluating whether eyes or a nose don't cast sharp shadows.
- float Specularity ()

  Specularity attribute used to evaluate the presence of spotlights in an area of the detected face in the original face capture.
- float EyeGaze ()

  Eye Gaze attribute used to evaluate whether the gaze direction of the detected face in the original face capture is frontal.
- float RightEyeStatus ()

  Attribute for evaluating right eye status.
- float LeftEyeStatus ()

  Attribute for evaluating left eye status.
- float GlassStatus ()

  Attribute for evaluating glasses presence.
- float HeavyFrame ()

  Attribute for evaluating whether glasses with heavy frames are not present.
- float MouthStatus ()

  Attribute for evaluating mouth status.
- float BackgroundUniformity (uint8_t minimalBackgroundRatio)

  Attribute used to measure background uniformity in the close area around the detected face in the original face capture.
- float BackgroundUniformity ()

  Attribute used to measure background uniformity in the close area around the detected face in the original face capture.
- float Age ()

  Attribute for evaluating age of subject using the face.
- float Age (AgeEstimationMode mode)

  Attribute for evaluating age of subject using the face.
- float RightRedEye ()

  Attribute for evaluating whether red-eye effect is not present on right eye.
- float LeftRedEye ()

  Attribute for evaluating whether red-eye effect is not present on left eye.
- float Gender ()

  Attribute for evaluating gender of subject.
- float Gender (GenderEstimationMode mode)

  Attribute for evaluating gender of subject.
- float EyeDistance ()

  Attribute used to measure the distance between the eyes in pixels for the detected face in the original face capture.
- float RollAngle ()

  Attribute representing the head rotation angle around the Z-axis of the detected face in the original face capture, relative to the camera reference frame, as per DIN 9300.
- float PitchAngle ()

  Attribute representing the head rotation angle around the X-axis of the detected face in the original face capture, relative to the camera reference frame, as per DIN 9300.
- float YawAngle ()

  Attribute representing the head rotation angle around the Y-axis of the detected face in the original face capture, relative to the camera reference frame, as per DIN 9300.
- float FaceSize ()

  Attribute representing face size - the maximum of eye distance and eye-mouth distance.
- float FaceRelativeArea ()

  Attribute representing the area of the detected face in the original face capture relative to the size of the original face capture.
- float FaceRelativeAreaInImage ()

Attribute representing the visible area of the detected face in the original face capture relative to the total face area.

- float TintedGlasses ()

  Attribute for evaluating tinted glasses presence.
- float WidthHeightRatio () const

  Attribute representing width to height aspect ratio of the original face capture.
- float PassiveLiveness ()

  Attribute used to assess whether a detected face in the original face capture qualifies as live, using the PassiveLivenessMode::ACCURATE calculation method.
- float PassiveLiveness (PassiveLivenessMode mode)

  Attribute used to assess whether a detected face in the original face capture qualifies as live.

Public Attributes

- const std::shared_ptr< FaceCapture > capture

  Provides the original FaceCapture.
- const unsigned int confidence

  Provides confidence score of the face related to face detection.

Friends

- class **FaceDetector**

4.48.1 Member Enumeration Documentation

4.48.1.1 AgeEstimationMode

```
enum class AgeEstimationMode : uint8_t [strong]
```

Enumerator

BALANCED	Provides the fastest processing speed but the lowest age estimation accuracy compared to ACCURATE mode. **Required Solvers** Total size of solvers is 8 MB. References face-attribute-age-gender-balanced.solver
ACCURATE	Provides better age estimation accuracy compared to BALANCED mode. However the performance of the age estimation is not as good as when BALANCED mode is used. **Note** The age estimation feature MUST be allowed in license to get accurate age estimation. **Required Solvers** Total size of solvers is 313 MB. References face-attribute-age-gender-accurate.nnkitsolver

4.48.1.2 AttributeID

```
enum class AttributeID : uint8_t [strong]
```

Enumerator

CONFIDENCE	Provides confidence score of the face related to face detection. **See also** Face::confidence
SHARPNESS	Sharpness attribute used to measure the sharpness level of an area of the detected face in the original face capture. **See also** Face::Sharpness
BRIGHTNESS	Brightness attribute used to measure the brightness level of an area of the detected face in the original face capture. **See also** Face::Brightness
CONTRAST	Contrast attribute used to measure the contrast level of an area of the detected face in the original face capture. **See also** Face::Contrast
UNIQUE_INTENSITY_LEVELS	Unique Intensity Levels attribute used to measure whether an area of the detected face in the original face capture has an appropriate number of unique intensity levels. **See also** Face::UniqueIntensityLevels
SHADOW	Shadow attribute used to evaluate whether an area of the detected face in the original face capture is affected by shadows. **See also** Face::Shadow
NOSE_SHADOW	Nose Shadow attribute for evaluating whether eyes or a nose don't cast sharp shadows. **See also** Face::NoseShadow
SPECULARITY	Specularity attribute used to evaluate the presence of spotlights in an area of the detected face in the original face capture. **See also** Face::Specularity

EYE_GAZE	Eye Gaze attribute used to evaluate whether the gaze direction of the detected face in the original face capture is frontal. **See also** Face::EyeGaze
RIGHT_EYE_STATUS	Attribute for evaluating right eye status. **See also** Face::RightEyeStatus
LEFT_EYE_STATUS	Attribute for evaluating left eye status. **See also** Face::LeftEyeStatus
GLASS_STATUS	Attribute for evaluating glasses presence. **See also** Face::GlassStatus
HEAVY_FRAME	Attribute for evaluating whether glasses with heavy frames are not present. **See also** Face::HeavyFrame
MOUTH_STATUS	Attribute for evaluating mouth status. **See also** Face::MouthStatus
BACKGROUND_UNIFORMITY	Attribute used to measure background uniformity in the close area around the detected face in the original face capture. **See also** Face::BackgroundUniformity
AGE	Attribute for evaluating age of subject using the face. **See also** Face::Age
RIGHT_RED_EYE	Attribute for evaluating whether red-eye effect is not present on right eye. **See also** Face::RightRedEye

LEFT_RED_EYE	Attribute for evaluating whether red-eye effect is not present on left eye. **See also** Face::LeftRedEye
GENDER	Attribute for evaluating gender of subject. **See also** Face::Gender
EYE_DISTANCE	Attribute used to measure the distance between the eyes in pixels for the detected face in the original face capture. **See also** Face::EyeDistance
ROLL_ANGLE	Attribute representing the head rotation angle around the Z-axis of the detected face in the original face capture, relative to the camera reference frame, as per DIN 9300. **See also** Face::RollAngle
PITCH_ANGLE	Attribute representing the head rotation angle around the X-axis of the detected face in the original face capture, relative to the camera reference frame, as per DIN 9300. **See also** Face::PitchAngle
YAW_ANGLE	Attribute representing the head rotation angle around the Y-axis of the detected face in the original face capture, relative to the camera reference frame, as per DIN 9300. **See also** Face::YawAngle
FACE_SIZE	Attribute representing face size - the maximum of eye distance and eye-mouth distance. **See also** Face::FaceSize
FACE_RELATIVE_AREA	Attribute representing the area of the detected face in the original face capture relative to the size of the original face capture. **See also** Face::FaceRelativeArea

FACE_RELATIVE_AREA_IN_IMAGE	Attribute representing the visible area of the detected face in the original face capture relative to the total face area. **See also** Face::FaceRelativeAreaInImage
TINTED_GLASSES	Attribute for evaluating tinted glasses presence. **See also** Face::TintedGlasses
WIDTH_HEIGHT_RATIO	Attribute representing width to height aspect ratio of the original face capture. **See also** Face::WidthHeightRatio

4.48.1.3 GenderEstimationMode

```
enum class GenderEstimationMode : uint8_t [strong]
```

Enumerator

BALANCED	Provides the fastest processing speed but the lowest gender estimation accuracy compared to ACCURATE mode. **Required Solvers** Total size of solvers is 8 MB. References face-attribute-age-gender-balanced.solver
ACCURATE	Provides better gender estimation accuracy compared to BALANCED mode. However the performance of the gender estimation is not as good as when BALANCED mode is used. **Note** The gender estimation feature MUST be allowed in license to get accurate gender estimation. **Required Solvers** Total size of solvers is 313 MB. References face-attribute-age-gender-accurate.nnkitsolver

4.48.1.4 PassiveLivenessMode

```
enum class PassiveLivenessMode : uint8_t [strong]
```

Enumerator

FAST	Passive liveness mode with best performance available but worse accuracy as ACCURATE mode. **Required Solvers** Total size of solvers is 5 MB. References face-attribute-passive-liveness-universal-fast.nnkitsolver
ACCURATE	Passive liveness mode with highest accuracy available but significantly slower as FAST mode. **Note** Not supported in this variant of Enrollment SDK.

4.48.2 Member Function Documentation

4.48.2.1 Age() [1/2]

```
float Age () [inline]
```

It uses BALANCED estimation.

Returns

estimated age

Exceptions

EnrollmentException	when attribute can not be calculated

Required Solvers

Total size of solvers is 8 MB.

References face-attribute-age-gender-balanced.solver

4.48.2.2 Age() [2/2]

```
float Age (
            AgeEstimationMode mode) [inline]
```

Parameters

mode	mode of age estimation

Returns

estimated age

Exceptions

EnrollmentException	when attribute can not be calculated

Required Solvers

See Face::AgeEstimationMode

4.48.2.3 BackgroundUniformity() [1/2]

```
float BackgroundUniformity () [inline]
```

The original face capture is stored in the Face::capture member variable.

The attribute can be taken as an ICAO feature.

The default background percentage of the whole image is 5%.

Returns

Background uniformity values are from range <-10000,10000>. Values near -10000 indicates 'very non-uniform background present', values near 10000 indicates 'uniform background present'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

Required Solvers

Total size of solvers is 1 MB.

References face-attribute-background-uniformity.solver

4.48.2.4 BackgroundUniformity() [2/2]

```
float BackgroundUniformity (
            uint8_t minimalBackgroundRatio) [inline]
```

The original face capture is stored in the Face::capture member variable.

The attribute can be taken as an ICAO feature.

Parameters

minimalBackgroundRatio	Specifies minimal background percentage of the whole image which needs to be available for background uniformity solver. The values lower as 5% resolves into less accurate result.

Returns

Background uniformity values are from range <-10000,10000>. Values near -10000 indicates 'very non-uniform background present', values near 10000 indicates 'uniform background present'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated
EnrollmentInvalidArgumentException	when minimalBackgroundRatio is not in range <0,100>

Required Solvers

Total size of solvers is 1 MB.

References face-attribute-background-uniformity.solver

4.48.2.5 Brightness()

```
float Brightness () [inline]
```

The original face capture is stored in the Face::capture member variable.

The attribute can be taken as an ICAO feature.

Returns

Brightness values are from range <-10000,10000>. Values near -10000 indicates 'too dark', values near 10000 indicates 'too light', values around 0 indicates OK. The decision thresholds are around -5000 and 5000.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.6 Contrast()

```
float Contrast () [inline]
```

The original face capture is stored in the Face::capture member variable.

The attribute can be taken as an ICAO feature.

Returns

Contrast values are from range <-10000,10000>. Values near -10000 indicates 'very low contrast', values near 10000 indicates 'very high contrast', values around 0 indicates OK. The decision thresholds are around -5000 and 5000.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.7 Crop()

```
std::shared_ptr< CroppedFace > Crop (
            FaceCropConfiguration cfg) const  [inline], [nodiscard]
```

This method extracts a cropped version of the detected face based on the given cropping configuration. The original face capture is stored in the Face::capture member variable.

Parameters

cfg	The configuration specifying how the face should be cropped.

Returns

CroppedFace object containing the cropped face.

4.48.2.8 DetectionRectangle()

```
Rectangle DetectionRectangle () const  [inline], [nodiscard]
```

This method returns the rectangular region that encloses the detected face within the original face capture.

Returns

A Rectangle representing the bounding box of the detected face.

4.48.2.9 Extract() [1/2]

```
std::shared_ptr< ExtractedFace > Extract ()  [inline]
```

Returns

template calculated template

Exceptions

EnrollmentException	when it is not possible to extract templates

4.48.2.10 Extract() [2/2]

```
std::shared_ptr< ExtractedFace > Extract (
            const FaceExtractorConfig & cfg)  [inline]
```

Parameters

cfg	FaceExtractorConfig

Returns

template calculated template

Exceptions

EnrollmentException	when it is not possible to extract templates

Required Solvers

See FaceExtractorConfig::Type

4.48.2.11 EyeDistance()

```
float EyeDistance ()  [inline]
```

The original face capture is stored in the Face::capture member variable.

The attribute can be taken as an ICAO feature.

Returns

Distance in pixels.

Exceptions

EnrollmentException	when attribute can not be calculated

Required Solvers

See FaceDetector::Config::Type

4.48.2.12 EyeGaze()

```
float EyeGaze () [inline]
```

The original face capture is stored in the Face::capture member variable.

The attribute can be taken as an ICAO feature.

Returns

Eye gaze values are from range <-10000,10000>. Values near -10000 indicates 'sideway ahead eyes gaze', values near 10000 indicates 'straight ahead eyes gaze'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

Required Solvers

Total size of solvers is 944 kB.

References face-attribute-eye-gaze.solver

4.48.2.13 FaceRelativeArea()

```
float FaceRelativeArea () [inline]
```

The original face capture is stored in the Face::capture member variable.

Returns

area of face relative to image size in percentage. It can be more that 100% due to borders added by face detector.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.14 FaceRelativeAreaInImage()

```
float FaceRelativeAreaInImage () [inline]
```

This value indicates the percentage of the face area that is outside the image. The original face capture is stored in the Face::capture member variable.

Returns

area of face visible in image relative to total area of face. This value implies the percentage of face area outside the image.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.15 FaceSize()

```
float FaceSize () [inline]
```

Returns

face size - the maximum of eye distance and eye-mouth distance in pixels.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.16 Gender() [1/2]

```
float Gender () [inline]
```

It uses BALANCED estimation.

Returns

Gender values are from range <-10000, 10000>. Values near -10000 indicates 'male', values near 10000 indicates 'female'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

Required Solvers

Total size of solvers is 8 MB.

References face-attribute-age-gender-balanced.solver

4.48.2.17 Gender() [2/2]

```
float Gender (
            GenderEstimationMode mode)  [inline]
```

Parameters

mode	mode of gender estimation

Returns

Gender values are from range <-10000, 10000>. Values near -10000 indicates 'male', values near 10000 indicates 'female'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

Required Solvers

Total size of solvers is 8 MB.

References face-attribute-age-gender-balanced.solver

4.48.2.18 GetKeypoints()

```
std::shared_ptr< Keypoints > GetKeypoints () const  [inline], [nodiscard]
```

The original face capture is stored in the Face::capture member variable.

Returns

Keypoints instance.

4.48.2.19 GlassStatus()

```
float GlassStatus ()  [inline]
```

Returns

Glasses values are from range <-10000,10000>. Values near -10000 indicates 'no glasses present', values near 10000 indicates 'glasses present'. The decision threshold is around 0

Exceptions

EnrollmentException	when attribute can not be calculated

Required Solvers

Total size of solvers is 608 kB.

References face-attribute-glasses.solver

4.48.2.20 HeavyFrame()

```
float HeavyFrame () [inline]
```

The attribute can be taken as an ICAO feature.

Returns

Heavy frame glasses values are from range <-10000,10000>. Values near -10000 indicates 'no heavy frame glasses present', values near 10000 indicates 'heavy frame glasses present'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

Required Solvers

Total size of solvers is 498 kB.

References face-attribute-heavy-frame.solver

4.48.2.21 IsSameAs()

```
bool IsSameAs (
            const Face & face) const [inline]
```

Parameters

face	to check if it is reference to same object

Returns

`true` when given Face is the same object

`false` when given Face is a different object

4.48.2.22 LeftEyeStatus()

```
float LeftEyeStatus () [inline]
```

The attribute can be taken as an ICAO feature.

Returns

Left eye values are from range <-10000,10000>. Values near -10000 indicates 'closed, narrowed or bulged eye', values near 10000 indicates 'normally opened eye'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

Required Solvers

Total size of solvers is 16 kB.

References face-attribute-eye-status.solver

4.48.2.23 LeftRedEye()

```
float LeftRedEye () [inline]
```

The attribute can be taken as an ICAO feature.

Returns

Left red-eye effect values are from range <-10000,10000>. Values near -10000 indicates 'no red-eye effect present', values near 10000 indicates 'red-eye effect present'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.24 MouthStatus()

```
float MouthStatus () [inline]
```

The attribute can be taken as an ICAO feature.

Returns

Mouth status values are from range <-10000,10000>. Values near -10000 indicates 'open mouth, smile showing teeth or round lips present', values near 10000 indicates 'mouth with no expression'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

Required Solvers

Total size of solvers is 155 kB.

References face-attribute-mouth-status.solver

4.48.2.25 NoseShadow()

```
float NoseShadow () [inline]
```

The attribute can be taken as an ICAO feature.

Returns

Nose shadow values are from range <-10000,10000> Values near -10000 indicates 'very strong local (eyes/nose) shadows present', values near 10000 indicates 'no local shadows present'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.26 operator"!=()

```
bool operator!= (
            const Face & f) const [inline]
```

Parameters

f	compared Face

Returns

`true` when data of Face are different

`false` when data of Face are same

4.48.2.27 operator==()

```
bool operator== (
            const Face & f) const [inline]
```

Parameters

f	compared Face

Returns

`true` when data of Face are same

`false` when data of Face are different

4.48.2.28 PassiveLiveness() [1/2]

```
float PassiveLiveness () [inline]
```

The original face capture is stored in the Face::capture member variable.

Returns

Passive Liveness score from range <0,100>.

Note

Not supported in this variant of Enrollment SDK.

4.48.2.29 PassiveLiveness() [2/2]

```
float PassiveLiveness (
            PassiveLivenessMode mode) [inline]
```

The original face capture is stored in the Face::capture member variable.

Exceptions

EnrollmentException	when attribute can not be calculated

Parameters

mode	mode of passive liveness calculation. For more information see Face Passive Liveness.

Note

The passive liveness feature MUST be allowed in license to get face passive liveness.

Returns

Passive Liveness score from range <0,100>.

When `mode` is PassiveLivenessMode::FAST then Fast Passive Liveness Thresholds can be used to check live and non-live faces.

Required Solvers

See Face::PassiveLivenessMode

4.48.2.30 PitchAngle()

```
float PitchAngle () [inline]
```

The original face capture is stored in the Face::capture member variable.

Returns

angle rotation of head towards camera reference frame around X-axis as per DIN9300.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.31 RightEyeStatus()

```
float RightEyeStatus () [inline]
```

The attribute can be taken as an ICAO feature.

Returns

Right eye values are from range <-10000,10000>. Values near -10000 indicates 'closed, narrowed or bulged eye', values near 10000 indicates 'normally opened eye'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

Required Solvers

Total size of solvers is 16 kB.

References face-attribute-eye-status.solver

4.48.2.32 RightRedEye()

```
float RightRedEye () [inline]
```

The attribute can be taken as an ICAO feature.

Returns

Right red-eye effect values are from range <-10000,10000>. Values near -10000 indicates 'no red-eye effect present', values near 10000 indicates 'red-eye effect present'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.33 RollAngle()

```
float RollAngle () [inline]
```

The original face capture is stored in the Face::capture member variable.

Returns

angle rotation of head towards camera reference frame around Z-axis as per DIN9300.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.34 Shadow()

```
float Shadow () [inline]
```

The original face capture is stored in the Face::capture member variable.

The attribute can be taken as an ICAO feature.

Returns

Shadow values are from range <-10000,10000>. Values near -10000 indicates 'very strong global shadows present', values near 10000 indicates 'no global shadows present'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.35 Sharpness()

```
float Sharpness () [inline]
```

The original face capture is stored in the Face::capture member variable.

The attribute can be taken as an ICAO feature.

Returns

Sharpness values are from range <-10000,10000>. Values near -10000 indicates 'very blurred', values near 10000 indicates 'very sharp'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.36 Specularity()

```
float Specularity () [inline]
```

The original face capture is stored in the Face::capture member variable.

The attribute can be taken as an ICAO feature.

Returns

Specularity values are from range <-10000,10000> Values near -10000 indicates 'very strong specularity present', values near 10000 indicates 'no specularity present'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.37 TintedGlasses()

```
float TintedGlasses () [inline]
```

Returns

Tinted glasses values are from range <-10000,10000>. Values near -10000 indicates 'no tinted glasses present', values near 10000 indicates 'tinted glasses present'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

Required Solvers

Total size of solvers is 608 kB.

References face-attribute-glasses.solver

4.48.2.38 ToIsoImage()

```
std::vector< uint8_t > ToIsoImage (
            ImageEncoder & encoder) const  [inline], [nodiscard]
```

ISO standard defines a standard data format of digital face images to allow interoperability among vendors.

Parameters

encoder	specific encoder for encoding Print image. Supported JPEG 2000 (lossy & lossless), PNG

Returns

ISO image from ISO/IEC 19794-5 (Face image data)

Exceptions

ImageException	when it is not possible to encode image
EnrollmentInvalidArgumentException	when encoder is not supported
EnrollmentRuntimeException	for invalid conversion size

4.48.2.39 UniqueIntensityLevels()

```
float UniqueIntensityLevels ()  [inline]
```

The original face capture is stored in the Face::capture member variable.

The attribute can be taken as an ICAO feature.

Returns

Unique intensity levels values are from range <-10000,10000>. Values near -10000 indicates 'very few unique intensity levels', values near 10000 indicates 'enough unique intensity levels'. The decision threshold is around 0.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.2.40 WidthHeightRatio()

```
float WidthHeightRatio () const  [inline]
```

This is calculated as an image width divided image height, as defined in B.3.2.2 Width to height ratio of the image [6] .

The original face capture is stored in the Face::capture member variable.

Returns

width to height aspect ratio of the original Face::capture

4.48.2.41 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Face.

Implements Serializable.

4.48.2.42 YawAngle()

```
float YawAngle ()  [inline]
```

The original face capture is stored in the Face::capture member variable.

Returns

angle rotation of head towards camera reference frame around Y-axis as per DIN9300.

Exceptions

EnrollmentException	when attribute can not be calculated

4.48.3 Member Data Documentation

4.48.3.1 capture

```
const std::shared_ptr<FaceCapture> capture
```

It contains the image in which this face was detected.

4.48.3.2 confidence

```
const unsigned int confidence
```

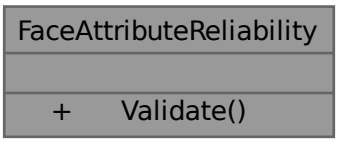
It can be taken as an ICAO feature. Confidence values are from range <0,10000>. The higher the value of the attribute the better quality of the face. The decision thresholds are around 600, but it depends on the face image quality / camera angle etc.

4.49 FaceAttributeReliability Class Reference

Interface for checking reliability of attribute.

```
#include <FaceValidation.h>
```

Collaboration diagram for FaceAttributeReliability:



Public Member Functions

- virtual std::vector< FailedFaceAttribute > Validate (const std::shared_ptr< Face > &probe)=0
  Check if face is reliable to calculate attribute.

4.49.1 Detailed Description

In some cases the meaningfulness of attribute value relies on values of other attributes.

4.49.2 Member Function Documentation

4.49.2.1 Validate()

```
virtual std::vector< FailedFaceAttribute > Validate (
            const std::shared_ptr< Face > & probe)  [pure virtual]
```

When function returns empty string then face is usable for attribute calculation.

Parameters

probe	Face to be checked

Returns

list of FailedFaceAttribute

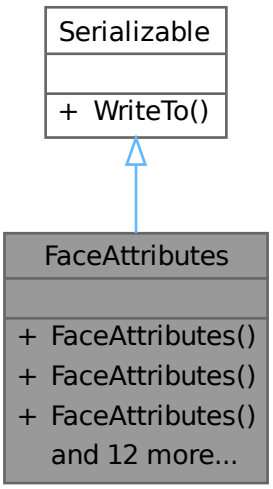
Implemented in AgeReliability, FaceSizeReliability, GlassStatusReliability, ICAOReliability, ISOFullFrontalImageValidator, MouthOpennessReliability, SegmentationMaskReliability, TemplateReliability, TintedGlassesReliability, and UniversalPassiveLivenessReliability.

4.50 FaceAttributes Class Reference

Face attributes Holds the face attributes.

```
#include <FaceAttributes.h>
```

Collaboration diagram for FaceAttributes:



Public Member Functions

- **FaceAttributes** ()=default

  Default constructor of face attributes.
- FaceAttributes (const FaceAttributes &fa)

  Copy constructor of face attributes.
- FaceAttributes (FaceAttributes &&fa) noexcept

  Move constructor of face attributes.
- FaceAttributes & operator= (const FaceAttributes &fa)

  Assign the face attributes.
- FaceAttributes & operator= (FaceAttributes &&fa) noexcept

  Assign the face attributes.
- constexpr bool operator== (const FaceAttributes &fa) const

  Compares the face attributes.
- void WriteTo (Writer &writer) const final

  Write itself into writer.
- void SetCaptureDeviceTechnology (FaceCaptureDeviceTechnology technology)

  Sets capture device technology.
- FaceCaptureDeviceTechnology GetCaptureDeviceTechnology () const

  Provides capture device technology.
- void SetCaptureDeviceVendorID (FaceCaptureDeviceVendorID vendor)

  Sets capture device vendor ID.

- FaceCaptureDeviceVendorID GetCaptureDeviceVendorID () const
  Provides capture device vendor ID.
- void SetCaptureDeviceTypeID (FaceCaptureDeviceTypeID type)
  Sets capture device type ID.
- FaceCaptureDeviceTypeID GetCaptureDeviceTypeID () const
  Provides capture device type ID.
- void SetFaceImageType (FaceImageType type)
  Sets face image type.
- FaceImageType GetFaceImageType () const
  Provides face image type.

4.50.1 Detailed Description

Enables to set each face attribute separately. In case, the attributes are not set the default values are used:

- capture device technology = FaceCaptureDeviceTechnology::UNKNOWN
- capture device vendor ID = 0
- capture device type ID = 0

4.50.2 Constructor & Destructor Documentation

4.50.2.1 FaceAttributes() [1/2]

```
FaceAttributes (
            const FaceAttributes & fa)  [inline]
```

Parameters

fa	Face attributes

4.50.2.2 FaceAttributes() [2/2]

```
FaceAttributes (
            FaceAttributes && fa)  [inline], [noexcept]
```

Parameters

fa	Face attributes

4.50.3 Member Function Documentation

4.50.3.1 GetCaptureDeviceTechnology()

```
FaceCaptureDeviceTechnology GetCaptureDeviceTechnology () const  [inline], [nodiscard]
```

Returns

Capture device technology

4.50.3.2 GetCaptureDeviceTypeID()

```
FaceCaptureDeviceTypeID GetCaptureDeviceTypeID () const  [inline], [nodiscard]
```

Returns

Capture device type ID

4.50.3.3 GetCaptureDeviceVendorID()

```
FaceCaptureDeviceVendorID GetCaptureDeviceVendorID () const  [inline], [nodiscard]
```

Returns

Capture device vendor ID

4.50.3.4 GetFaceImageType()

```
FaceImageType GetFaceImageType () const  [inline], [nodiscard]
```

Returns

FaceImageType

4.50.3.5 operator=() [1/2]

```
FaceAttributes & operator= (
            const FaceAttributes & fa)  [inline]
```

Parameters

fa	Face attributes

Returns

this modified object

4.50.3.6 operator=() [2/2]

```
FaceAttributes & operator= (
            FaceAttributes && fa)  [inline], [noexcept]
```

Parameters

fa	Face attributes

Returns

this modified object

4.50.3.7 operator==()

```
bool operator== (
            const FaceAttributes & fa) const  [inline], [constexpr]
```

Parameters

fa	Face attributes

Returns

`true` when this and given FaceAttributes are equal

`false` otherwise

4.50.3.8 SetCaptureDeviceTechnology()

```
void SetCaptureDeviceTechnology (
            FaceCaptureDeviceTechnology technology)  [inline]
```

Parameters

technology	Capture device technology

4.50.3.9 SetCaptureDeviceTypeID()

```
void SetCaptureDeviceTypeID (
            FaceCaptureDeviceTypeID type)  [inline]
```

Parameters

type	Capture device type ID

4.50.3.10 SetCaptureDeviceVendorID()

```
void SetCaptureDeviceVendorID (
            FaceCaptureDeviceVendorID vendor)  [inline]
```

Parameters

vendor	Capture device vendor ID

4.50.3.11 SetFaceImageType()

```
void SetFaceImageType (
            FaceImageType type)  [inline]
```

Parameters

type	FaceImageType

4.50.3.12 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

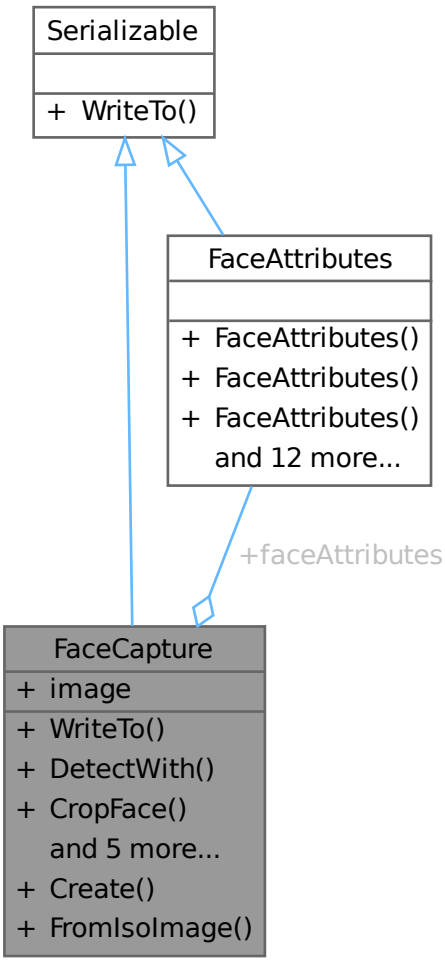
Implements Serializable.

4.51 FaceCapture Class Reference

Face capture can contains one or group of people, faces or full body.

```
#include <FaceCapture.h>
```

Collaboration diagram for FaceCapture:



Public Member Functions

- void WriteTo (Writer &writer) const final

  Write itself into writer.
- std::vector< std::shared_ptr< Face > > DetectWith (FaceCaptureSegmentation &detector)

  Detects faces in the current face capture.
- std::shared_ptr< Image > CropFace (const Rectangle &rectangle, const RGBA &backgroundColor) const

  It crops image with given rectangle and background color.
- bool IsWholeCroppedRectangleInside (const Rectangle &rectangle) const

  Calculates intersection of image and cropping rectangle.
- std::shared_ptr< Image > Draw (const std::vector< std::shared_ptr< Face > > &faces, const ImageShapeAttributes &attr) const

  It draws rounding boxes into FaceCapture.
- std::shared_ptr< FaceCapture > ResizeToMaxDimension (ImageDimension maxDimension)

  Returns new FaceCapture with resized image to given max dimension.

- ImageDimension MaxImageDimension () const

  Returns max dimension of image in FaceCapture.
- std::shared_ptr< Image > Draw (const Rectangle &boundingBox, const ImageShapeAttributes &attr) const

  It draws bounding box into FaceCapture.

Static Public Member Functions

- static std::shared_ptr< FaceCapture > Create (std::shared_ptr< Image > img, const FaceAttributes &fa=FaceAttributes())

  Constructor of FaceCapture.
- static std::shared_ptr< FaceCapture > FromIsoImage (const std::vector< uint8_t > &isoImage)

  Function creates Face object from ISO image based on ISO/IEC 19794-5 (Face image data).

Public Attributes

- const std::shared_ptr< Image > **image**

  Provides FaceCapture Image.
- const FaceAttributes **faceAttributes**

  Provides FaceAttributes.

4.51.1 Member Function Documentation

4.51.1.1 Create()

```
std::shared_ptr< FaceCapture > Create (
            std::shared_ptr< Image > img,
            const FaceAttributes & fa = FaceAttributes())  [inline], [static], [nodiscard]
```

Parameters

img	face capture image, will be moved into this class
fa	FaceAttributes

Returns

created FaceCapture object

Exceptions

NullArgumentException	when image is null

4.51.1.2 CropFace()

```
std::shared_ptr< Image > CropFace (
            const Rectangle & rectangle,
            const RGBA & backgroundColor) const  [inline], [nodiscard]
```

Can be used for manual segmentation.

Parameters

rectangle	Rectangle to be cropped
backgroundColor	background color for area cropped outside image

Returns

Image cropped image

4.51.1.3 DetectWith()

```
std::vector< std::shared_ptr< Face > > DetectWith (
            FaceCaptureSegmentation & detector)  [inline], [nodiscard]
```

This method uses the provided face detection algorithm to find faces within the current face capture. The detector must be an implementation of FaceCaptureSegmentation, such as FaceDetector.

Parameters

detector	A reference to a FaceCaptureSegmentation implementation used for face detection.

Returns

detected Face objects.

Exceptions

EnrollmentException	If the detector encounters an error during face detection.

4.51.1.4 Draw() [1/2]

```
std::shared_ptr< Image > Draw (
            const Rectangle & boundingBox,
            const ImageShapeAttributes & attr) const  [inline], [nodiscard]
```

Parameters

boundingBox	bounding box to draw

attr	ShapeAttributes of rounding boxes

Returns

Image with drawn rounding boxes

4.51.1.5 Draw() [2/2]

```
std::shared_ptr< Image > Draw (
            const std::vector< std::shared_ptr< Face > > & faces,
            const ImageShapeAttributes & attr) const  [inline], [nodiscard]
```

Parameters

faces	detected faces
attr	ShapeAttributes of rounding boxes

Returns

Image with drawn bounding boxes

4.51.1.6 FromIsoImage()

```
std::shared_ptr< FaceCapture > FromIsoImage (
            const std::vector< uint8_t > & isoImage)  [inline], [static], [nodiscard]
```

[6]

Parameters

isoImage	ISO image from ISO/IEC 19794-5 (Face image data)

Returns

decoded FaceCapture

Exceptions

ImageException	when type is not supported or image data are corrupted
EnrollmentInvalidArgumentException	when not possible to decode ISO image

4.51.1.7 IsWholeCroppedRectangleInside()

```
bool IsWholeCroppedRectangleInside (
            const Rectangle & rectangle) const  [inline], [nodiscard]
```

Parameters

rectangle	Rectangle to crop

Returns

`true` when whole rectangle is inside image

`false` when whole rectangle is not inside image

4.51.1.8 MaxImageDimension()

```
ImageDimension MaxImageDimension () const  [inline], [nodiscard]
```

Returns

max dimension of image in FaceCapture.

4.51.1.9 ResizeToMaxDimension()

```
std::shared_ptr< FaceCapture > ResizeToMaxDimension (
            ImageDimension maxDimension)  [inline], [nodiscard]
```

When any image dimension does not exceed given maximum dimension then original FaceCapture is returned. Otherwise new FaceCapture with resized image is returned.

Parameters

maxDimension	max dimension of new Image in FaceCapture

Returns

new FaceCapture with resized image

4.51.1.10 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

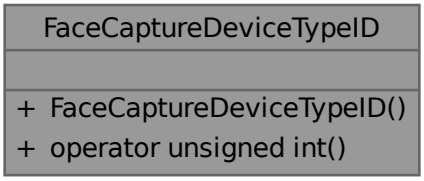
Implements Serializable.

4.52 FaceCaptureDeviceTypeID Class Reference

Capture device type identifier The capture device type identifier shall identify the biometric organization that owns the product that created the BDIR.

```
#include <FaceCaptureDeviceTypeID.h>
```

Collaboration diagram for FaceCaptureDeviceTypeID:



Public Member Functions

- FaceCaptureDeviceTypeID (unsigned int i)

  Construct a new Face Capture Equipment ID.
- **operator unsigned int** () const

  Conversion to capture equipment id number.

4.52.1 Detailed Description

The capture device algorithm type identifier shall be encoded in two bytes carrying a CBEFF biometric organization identifier (registered by IBIA or other approved registration authority). A value of all zeros shall indicate that the capture device type is unreported. [6]

4.52.2 Constructor & Destructor Documentation

4.52.2.1 FaceCaptureDeviceTypeID()

```
FaceCaptureDeviceTypeID (
            unsigned int i)  [inline], [explicit]
```

Parameters

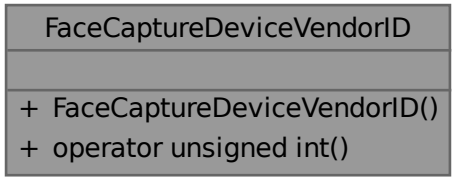
i	equipment ID determined by vendor. Only 16 bits are stored.

4.53 FaceCaptureDeviceVendorID Class Reference

Capture device vendor identifier The capture device vendor identifier shall identify the biometric organization that owns the product that created the BDIR.

```
#include <FaceCaptureDeviceVendorID.h>
```

Collaboration diagram for FaceCaptureDeviceVendorID:



Public Member Functions

- FaceCaptureDeviceVendorID (unsigned int i)

  Construct a new Face Capture Equipment ID.
- **operator unsigned int** () const

  Conversion to capture equipment id number.

4.53.1 Detailed Description

The capture device algorithm vendor identifier shall be encoded in two bytes carrying a CBEFF biometric organization identifier (registered by IBIA or other approved registration authority). A value of all zeros shall indicate that the capture device vendor is unreported. [6]

4.53.2 Constructor & Destructor Documentation

4.53.2.1 FaceCaptureDeviceVendorID()

```
FaceCaptureDeviceVendorID (
            unsigned int i)  [inline], [explicit]
```

Parameters

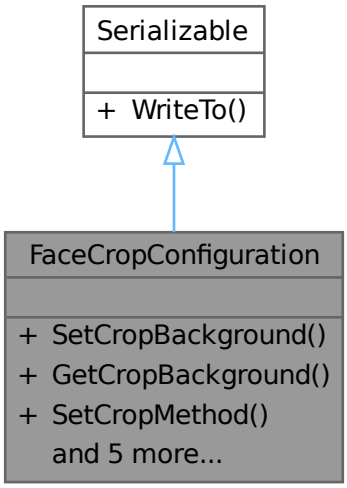
i	equipment ID determined by vendor. Only 16 bits are stored.

4.54 FaceCropConfiguration Class Reference

Face crop configuration.

```
#include <FaceCropConfiguration.h>
```

Collaboration diagram for FaceCropConfiguration:



Public Member Functions

- void SetCropBackground (const RGBA &background)

  Set the Crop Background color Background with transparency is only allowed when method is FaceCrop::FULL_NOT_ALIGNED.
- RGBA GetCropBackground () const

  Get the Crop Background object.
- void SetCropMethod (FaceCrop method)

  Set the Crop method.
- FaceCrop GetCropMethod () const

  Get the Crop method.

- void SetCropScale (float scale)

  Set the Crop scale.
- float GetCropScale () const

  Get the Crop scale.
- void WriteTo (Writer &writer) const final

  Function serializes configuration via provided Writer.
- FaceCropConfiguration ()=default

  Default configuration is.

4.54.1 Constructor & Destructor Documentation

4.54.1.1 FaceCropConfiguration()

```
FaceCropConfiguration () [default]
```

- cropped background - Colors::White
- crop method - FaceCrop::TOKEN
- cropScale - 1.0 (no scale)

4.54.2 Member Function Documentation

4.54.2.1 GetCropBackground()

```
RGBA GetCropBackground () const [inline], [nodiscard]
```

Returns

RGBA cropped background color

4.54.2.2 GetCropMethod()

```
FaceCrop GetCropMethod () const [inline], [nodiscard]
```

Returns

Crop

4.54.2.3 GetCropScale()

```
float GetCropScale () const [inline], [nodiscard]
```

Returns

crop scale

4.54.2.4 SetCropBackground()

```
void SetCropBackground (
            const RGBA & background)  [inline]
```

Parameters

background	cropped background color

Exceptions

EnrollmentInvalidArgumentException	when `background` is transparent and aligned crop is `true` and method is FaceCrop::FULL.
EnrollmentInvalidArgumentException	when `background` is transparent and method is FaceCrop::TOKEN.
EnrollmentInvalidArgumentException	when `background` is transparent and method is FaceCrop::TOKEN_NOT_FRONTAL.

4.54.2.5 SetCropMethod()

```
void SetCropMethod (
            FaceCrop method)  [inline]
```

Parameters

method	Crop method

Exceptions

EnrollmentInvalidArgumentException	when Crop::TOKEN_NOT_FRONTAL method needs to be set with other crop scale as 1.0.
EnrollmentInvalidArgumentException	when unknown crop method is going to be set.
EnrollmentInvalidArgumentException	when scale is already set and `method` is FaceCrop::TOKEN_NOT_FRONTAL.
EnrollmentInvalidArgumentException	when transparent background is already set and `method` is not FaceCrop::FULL_NOT_ALIGNED.

4.54.2.6 SetCropScale()

```
void SetCropScale (
            float scale)  [inline]
```

It shrink or enlarge rounding box according to scale.

Parameters

scale	Scale is in range <0.3,2>. When scale is 0.3 then rounding box is shrunk to 30% of original size. When scale is 2 then rounding box is enlarged to double size.

Exceptions

EnrollmentInvalidArgumentException	when scale is not in range or when crop method is Crop::TOKEN_↩ NOT_FRONTAL

4.54.2.7 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Config.

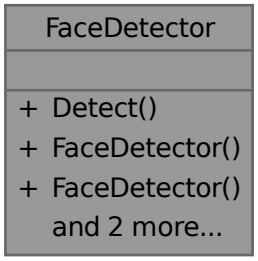
Implements Serializable.

4.55 FaceDetector Class Reference

FaceDetector implements detection by means of Innovatrics IFace.

```
#include <FaceDetector.h>
```

Collaboration diagram for FaceDetector:



Classes

- class Config

  Face detector configuration.

Public Member Functions

- std::vector< std::shared_ptr< Face > > Detect (std::shared_ptr< FaceCapture > capture) final

  Detect faces in given image.
- FaceDetector ()

  Construct a new FaceDetector object GlobalFaceExecutor is used for execution of face functionality.
- FaceDetector (Config config)

  Construct a new FaceDetector object GlobalFaceExecutor is used for execution of face functionality.
- FaceDetector (Config config, std::shared_ptr< FaceExecutor > executor)

  Construct a new FaceDetector object.
- FaceDetector (std::shared_ptr< FaceExecutor > executor)

  Construct a new FaceDetector object Default config is used for detection.

4.55.1 Constructor & Destructor Documentation

4.55.1.1 FaceDetector() [1/4]

```
FaceDetector () [inline]
```

Default config is used for detection

4.55.1.2 FaceDetector() [2/4]

```
FaceDetector (
            Config config) [inline], [explicit]
```

Parameters

config	configuration

4.55.1.3 FaceDetector() [3/4]

```
FaceDetector (
            Config config,
            std::shared_ptr< FaceExecutor > executor) [inline]
```

Parameters

config	configuration
executor	is strategy for execution of face functionality

Exceptions

NullArgumentException	when executor is null

4.55.1.4 FaceDetector() [4/4]

```
FaceDetector (
            std::shared_ptr< FaceExecutor > executor)  [inline], [explicit]
```

Parameters

executor	is strategy for execution of face functionality

Exceptions

NullArgumentException	when executor is null

4.55.2 Member Function Documentation

4.55.2.1 Detect()

```
std::vector< std::shared_ptr< Face > > Detect (
            std::shared_ptr< FaceCapture > capture)  [inline], [final]
```

Parameters

capture	FaceCapture where faces are detected

Returns

array of detected Face objects

Exceptions

EnrollmentException	when IFace returns error

Required Solvers

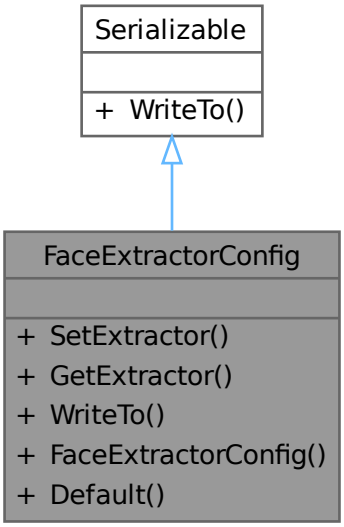
See FaceDetector::Config::Type

4.56 FaceExtractorConfig Class Reference

It contains extraction configuration.

```
#include <Face.h>
```

Collaboration diagram for FaceExtractorConfig:



Public Types

- enum class Type : uint8_t { ACCURATE , BALANCED , FAST }

  Type of neural network extractor.

Public Member Functions

- void SetExtractor (Type value)

  Function sets type of extractor.
- Type GetExtractor () const

  Function gets extractor type.
- void WriteTo (Writer &writer) const final

  Function serializes configuration via provided Writer.
- FaceExtractorConfig ()=default

  Constructs default configuration.

Static Public Member Functions

- static FaceExtractorConfig Default ()

  Provides default configuration.

4.56.1 Member Enumeration Documentation

4.56.1.1 Type

```
enum class Type : uint8_t [strong]
```

Enumerator

ACCURATE	Face templates suitable for verification/identification of very high accuracy are created when *accurate* mode is used. However the performance of the face template creation is not as good as when `balanced` or `fast` mode is used. **Required Solvers** Total size of solvers is 73 MB. References face-verification-accurate.solver
BALANCED	Face templates suitable for verification/identification of high accuracy are created when *balanced* mode is used. The performance of the face template creation is somewhere in between of `accurate` and `fast` modes. **Required Solvers** Total size of solvers is 15 MB. References face-verification-balanced.solver
FAST	Face templates suitable for verification of fairly good accuracy are created when *fast* mode is used. The performance of face template creation is very fast. Suitable for mobile/embedded devices. **Required Solvers** Total size of solvers is 3 MB. References face-verification-fast.solver

4.56.2 Constructor & Destructor Documentation

4.56.2.1 FaceExtractorConfig()

```
FaceExtractorConfig () [default]
```

- type - Type::BALANCED

4.56.3 Member Function Documentation

4.56.3.1 Default()

```
FaceExtractorConfig Default () [inline], [static]
```

- type - Type::BALANCED

Returns

default configuration.

4.56.3.2 GetExtractor()

```
Type GetExtractor () const [inline], [nodiscard]
```

Returns

extractor type

4.56.3.3 SetExtractor()

```
void SetExtractor (
            Type value) [inline]
```

Parameters

value	type from FaceExtractorConfig::Type.

4.56.3.4 WriteTo()

```
void WriteTo (
            Writer & writer) const [inline], [final], [virtual]
```

Parameters

writer	used to serialize Config.

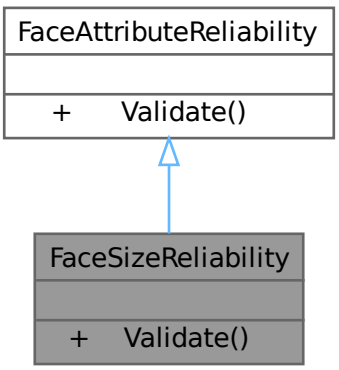
Implements Serializable.

4.57 FaceSizeReliability Class Reference

Check if calculation of Face Size attribute will be reliable.

```
#include <FaceValidation.h>
```

Collaboration diagram for FaceSizeReliability:



Public Member Functions

- std::vector< FailedFaceAttribute > Validate (const std::shared_ptr< Face > &probe) final

  Check if calculation of Face Size attribute will be reliable.

4.57.1 Member Function Documentation

4.57.1.1 Validate()

```
std::vector< FailedFaceAttribute > Validate (
            const std::shared_ptr< Face > & probe)  [inline], [final], [virtual]
```

Parameters

probe	Face to be checked

Returns

list of attributes that will cause not reliable Face Size attribute

Exceptions

NullArgumentException	when probe is null

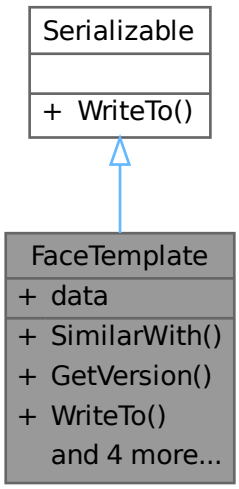
Implements FaceAttributeReliability.

4.58 FaceTemplate Class Reference

FaceTemplate is Innovatrics proprietary face template.

`#include <Face.h>`

Collaboration diagram for FaceTemplate:



Public Member Functions

- FaceSimilarityScore SimilarWith (const FaceTemplate &gallery) const

  Calculates FaceSimilarityScore for given FaceTemplate.
- std::string GetVersion () const

  Provides version information of Innovatrics proprietary algorithm used to create the face template.
- void WriteTo (Writer &writer) const override

  Function serializes FaceTemplate via provided Writer.
- FaceTemplate (std::vector< uint8_t > &&templ, std::shared_ptr< FaceExecutor > executor)

  Construct a new FaceTemplate object.
- FaceTemplate (std::vector< uint8_t > &&templ)

  Construct a new FaceTemplate object GlobalFaceExecutor is used.
- FaceTemplate (const std::vector< uint8_t > &templ, std::shared_ptr< FaceExecutor > executor)

  Construct a new FaceTemplate object, copy template data.
- FaceTemplate (const std::vector< uint8_t > &templ)

  Construct a new FaceTemplate object, copy template data GlobalFaceExecutor is used.

Public Attributes

- const std::vector< uint8_t > **data**

  Template data.

4.58.1 Detailed Description

The template can be used later for verification.

4.58.2 Constructor & Destructor Documentation

4.58.2.1 FaceTemplate() [1/4]

```
FaceTemplate (
            std::vector< uint8_t > && templ,
            std::shared_ptr< FaceExecutor > executor)  [inline]
```

Parameters

templ	template data
executor	is FaceExecutor

Exceptions

NullArgumentException	when executor is null
InvalidTemplateException	when Face template has no data

4.58.2.2 FaceTemplate() [2/4]

```
FaceTemplate (
            std::vector< uint8_t > && templ)  [inline], [explicit]
```

Parameters

templ	template data

Exceptions

InvalidTemplateException	when Face template has no data

4.58.2.3 FaceTemplate() [3/4]

```
FaceTemplate (
            const std::vector< uint8_t > & templ,
            std::shared_ptr< FaceExecutor > executor)  [inline]
```

Parameters

templ	template data
executor	is FaceExecutor

Exceptions

NullArgumentException	when executor is null
InvalidTemplateException	when Face template has no data

4.58.2.4 FaceTemplate() [4/4]

```
FaceTemplate (
            const std::vector< uint8_t > & templ)  [inline], [explicit]
```

Parameters

templ	template data

Exceptions

InvalidTemplateException	when Face template has no data

4.58.3 Member Function Documentation

4.58.3.1 GetVersion()

```
std::string GetVersion () const  [inline], [nodiscard]
```

Returns

version string

4.58.3.2 SimilarWith()

```
FaceSimilarityScore SimilarWith (
            const FaceTemplate & gallery) const  [inline], [nodiscard]
```

Parameters

gallery	FaceTemplate for score calculation

Returns

FaceSimilarityScore

Exceptions

EnrollmentException	when it is not possible to match templates

4.58.3.3 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [override], [virtual]
```

Parameters

writer	used to serialize Face.

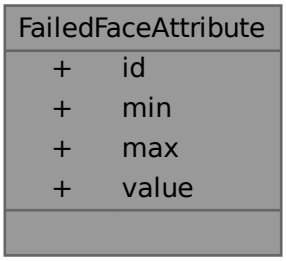
Implements Serializable.

4.59 FailedFaceAttribute Struct Reference

It contains attribute data that express which and why attribute fails in reliability check.

```
#include <FaceValidation.h>
```

Collaboration diagram for FailedFaceAttribute:



Public Attributes

- Face::AttributeID **id**

  FaceAttributeID of failed attribute.
- float **min**

  Minimal value of attribute.
- float **max**

  Maximal value of attribute.
- float **value**

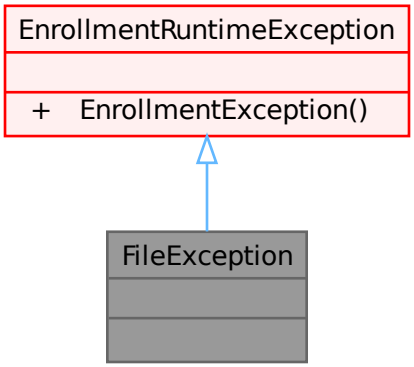
  Current value of attribute.

4.60 FileException Class Reference

Exception for file processing.

```
#include <FileException.h>
```

Collaboration diagram for FileException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentRuntimeException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

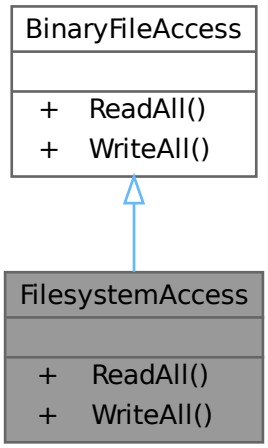
  Provides message with full info about exception.

4.61 FilesystemAccess Class Reference

Standard filesystem implementation.

```
#include <BinaryFile.h>
```

Collaboration diagram for FilesystemAccess:



Public Member Functions

- std::vector< uint8_t > ReadAll (const std::string &path) final

  Reads whole file as binary to memory.
- void WriteAll (const std::string &path, const std::vector< uint8_t > &content) final

  Writes whole given memory to file as binary.

4.61.1 Member Function Documentation

4.61.1.1 ReadAll()

```
std::vector< uint8_t > ReadAll (
            const std::string & path)  [inline], [final], [virtual]
```

Parameters

path	file path

Returns

file content

Exceptions

FileException	on problem with file system

Implements BinaryFileAccess.

4.61.1.2 WriteAll()

```
void WriteAll (
            const std::string & path,
            const std::vector< uint8_t > & content)  [inline], [final], [virtual]
```

Parameters

path	file path
content	content to write

Exceptions

FileException	on problem with file system

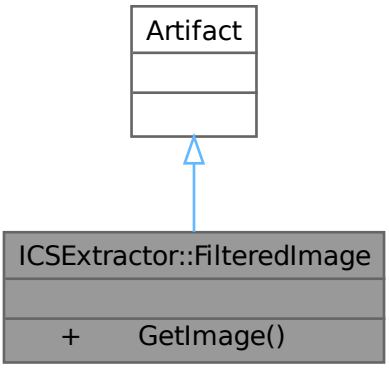
Implements BinaryFileAccess.

4.62 ICSExtractor::FilteredImage Class Reference

It enables creation of filtered image during extraction.

```
#include <ICSExtractor.h>
```

Collaboration diagram for ICSExtractor::FilteredImage:



Public Member Functions

- std::shared_ptr< Image > GetImage () const

  Get created Image To create Image, extraction must be finished.

4.62.1 Detailed Description

Note

The adjudicator feature MUST be allowed in license to get filtered image.

Figure 88 Finger for filtering

Figure 89 Filtered

4.62.2 Member Function Documentation

4.62.2.1 GetImage()

```
std::shared_ptr< Image > GetImage () const  [inline], [nodiscard]
```

Exceptions

EnrollmentRuntimeException	when extraction is not finished

Returns

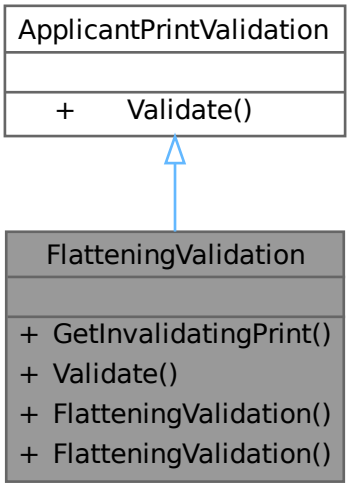
FilteredImage Image

4.63 FlatteningValidation Class Reference

Validates prints against gallery.

```
#include <Applicant.h>
```

Collaboration diagram for FlatteningValidation:



Public Member Functions

- std::shared_ptr< ExtractedPrint > GetInvalidatingPrint () const

  Function provides ExtractedPrint that causes invalidity of probe.
- bool Validate (const std::shared_ptr< ExtractedPrint > &probe, const std::vector< std::shared_ptr< ExtractedPrint > > &gallery) final

  Check if print is valid.
- FlatteningValidation (SimilarityThreshold samePositionThreshold, SimilarityThreshold differentPosition↩ Threshold, ICSMatcher &printMatcher)

  Construct a new Flattening Validation object.
- FlatteningValidation ()=default

  Construct a new Flattening Validation default object.

4.63.1 Detailed Description

Print is valid in following scenarios:

- When has same position and Similarity Score is above or equal Similarity Threshold given in constructor for same position
- When has different position and Similarity Score is below or equal Similarity Threshold given in constructor for different position.

4.63.2 Constructor & Destructor Documentation

4.63.2.1 FlatteningValidation() [1/2]

```
FlatteningValidation (
            SimilarityThreshold samePositionThreshold,
            SimilarityThreshold differentPositionThreshold,
            ICSMatcher & printMatcher)  [inline]
```

Parameters

samePositionThreshold	is SimilarityThreshold below which is validated Fingerprint considered as invalid when verified fingerprints have same position.
differentPositionThreshold	is SimilarityThreshold above which is validated Fingerprint considered as invalid when verified fingerprints have different position.
printMatcher	PrintMatcher to find already enrolled print

4.63.2.2 FlatteningValidation() [2/2]

```
FlatteningValidation ()  [default]
```

- same position threshold is 40
- different position threshold is 15

The default thresholds are choosen with regard to measured False Acceptance Rate.

Probe Fingers	Enrolled Fingers	Threshold at FAR=%1	Threshold at FAR=%0.1	Threshold at FAR=%0.01
1	1	20	30	40
1	10	30	40	50

4.63.3 Member Function Documentation

4.63.3.1 GetInvalidatingPrint()

```
std::shared_ptr< ExtractedPrint > GetInvalidatingPrint () const  [inline]
```

Returns

ExtractedPrint that causes invalidity of probe

4.63.3.2 Validate()

```
bool Validate (
            const std::shared_ptr< ExtractedPrint > & probe,
            const std::vector< std::shared_ptr< ExtractedPrint > > & gallery)  [inline],
[final], [virtual]
```

Fingerprint is valid in following scenarios:

- When has same position and Similarity Score is above or equal Similarity Threshold given in constructor for same position
- When has different position and Similarity Score is below or equal Similarity Threshold given in constructor for different position.

It returns enrolled print that causes invalidity of currently enrolling Print To realize in which scenario fails, you can check prints position.

Parameters

probe	Print to be checked
gallery	Prints that are already validated

Returns

`true` when print is valid

`false` when there is a dissimilar ExtractedPrint with same position as ExtractedPrint in gallery

`false` when there is a similar ExtractedPrint with different position as ExtractedPrint in gallery

Exceptions

NullArgumentException	when probe is null
NullArgumentException	when probe print is null

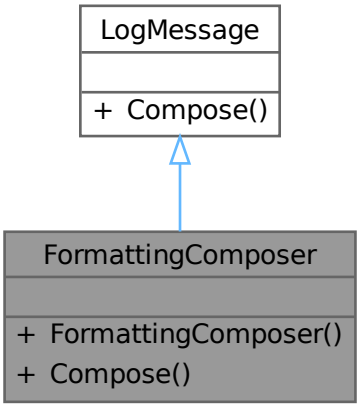
Implements ApplicantPrintValidation.

4.64 FormattingComposer Class Reference

Compose message from variadic arguments.

```
#include <FormattingLogger.h>
```

Collaboration diagram for FormattingComposer:



Public Member Functions

- template<typename... Args>
  FormattingComposer (const char *format, Args &&... arguments)

  Constructs FormattingComposer.
- const char * Compose () final

  Compose message from stored references to variadic arguments.

4.64.1 Detailed Description

It stores reference to data to be logged in FormattingComposer::FormattingComposer and compose message when function FormattingComposer::Compose is called.

Warning

Stored references MUST exists in time of Compose call.

The implementation is not thread safe. It should be instantiated once per each thread or on function stack.

Note

Only for C++

4.64.2 Constructor & Destructor Documentation

4.64.2.1 FormattingComposer()

```
template<typename... Args>
FormattingComposer (
            const char * format,
            Args &&... arguments) [inline], [explicit]
```

Parameters

format	accepts all specifiers as defined for `printf function`. Plus it defines own one 'v'. Argument specified with 'v' specifier is passed to Writer.
arguments	variadic arguments that MUST correspond to format

4.64.3 Member Function Documentation

4.64.3.1 Compose()

```
const char * Compose () [inline], [final], [virtual]
```

Returns

message

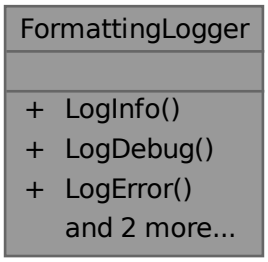
Implements LogMessage.

4.65 FormattingLogger Class Reference

It provides functions with format specifier to compose and log message via provided Logger.

```
#include <FormattingLogger.h>
```

Collaboration diagram for FormattingLogger:



Public Member Functions

- template<typename... Args>
  void LogInfo (const char *format, Args &&... args) const

  Formats and logs message as info message.
- template<typename... Args>
  void LogDebug (const char *format, Args &&... args) const

  Formats and logs message as debug message.
- template<typename... Args>
  void LogError (const char *format, Args &&... args) const

  Formats and logs message as error message.
- FormattingLogger (std::shared_ptr< Logger > logger)

  Constructs FormattingLogger.
- std::shared_ptr< Logger > GetLogger () const

  Get the Logger object.

4.65.1 Detailed Description

Function formats message with FormattingComposer and logs it. It accepts all specifiers as defined for `printf function`. Plus it defines own one 'v'. Argument specified with 'v' specifier is passed to Writer.

Note

Only for C++

4.65.2 Constructor & Destructor Documentation

4.65.2.1 FormattingLogger()

```
FormattingLogger (
            std::shared_ptr< Logger > logger)  [inline], [explicit]
```

Parameters

logger	Logger to log formater text

4.65.3 Member Function Documentation

4.65.3.1 GetLogger()

```
std::shared_ptr< Logger > GetLogger () const  [inline], [nodiscard]
```

Returns

Logger

4.65.3.2 LogDebug()

```
template<typename... Args>
void LogDebug (
            const char * format,
            Args &&... args) const [inline]
```

Parameters

format	It accepts all specifiers as defined for `printf function`. Plus it defines own one 'v'. Argument specified with 'v' specifier is passed to Writer.
args	variadic arguments that corresponds to format

4.65.3.3 LogError()

```
template<typename... Args>
void LogError (
            const char * format,
            Args &&... args) const [inline]
```

Parameters

format	It accepts all specifiers as defined for `printf function`. Plus it defines own one 'v'. Argument specified with 'v' specifier is passed to Writer.
args	variadic arguments that corresponds to format

4.65.3.4 LogInfo()

```
template<typename... Args>
void LogInfo (
            const char * format,
            Args &&... args) const [inline]
```

Parameters

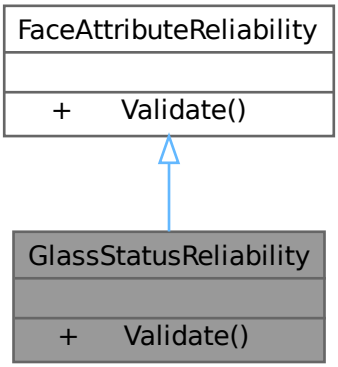
format	It accepts all specifiers as defined for `printf function`. Plus it defines own one 'v'. Argument specified with 'v' specifier is passed to Writer.
args	variadic arguments that corresponds to format

4.66 GlassStatusReliability Class Reference

Check if calculation of Glass Status attribute will be reliable.

```
#include <FaceValidation.h>
```

Collaboration diagram for GlassStatusReliability:



Public Member Functions

- std::vector< FailedFaceAttribute > Validate (const std::shared_ptr< Face > &probe) final

  Check if calculation of Glass Status attribute will be reliable.

4.66.1 Member Function Documentation

4.66.1.1 Validate()

```
std::vector< FailedFaceAttribute > Validate (
            const std::shared_ptr< Face > & probe)  [inline], [final], [virtual]
```

Parameters

probe	Face to be checked

Returns

list of attributes that will cause not reliable Glass Status attribute

Exceptions

NullArgumentException	when probe is null

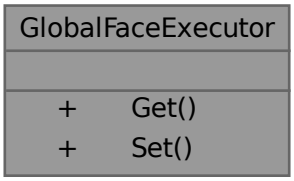
Implements FaceAttributeReliability.

4.67 GlobalFaceExecutor Class Reference

Holds instance of FaceExecutor.

```
#include <FaceExecutor.h>
```

Collaboration diagram for GlobalFaceExecutor:



Static Public Member Functions

- static std::shared_ptr< FaceExecutor > Get ()

  Provides current executor.
- static void Set (std::shared_ptr< FaceExecutor > executor)

  Sets new global executor.

4.67.1 Detailed Description

When executor is not set then default one is LocalFaceExecutor. The class is singletone.

4.67.2 Member Function Documentation

4.67.2.1 Get()

```
std::shared_ptr< FaceExecutor > Get () [inline], [static]
```

Returns

current executor

4.67.2.2 Set()

```
void Set (
            std::shared_ptr< FaceExecutor > executor)  [inline], [static]
```

Parameters

executor	new executor to store

Exceptions

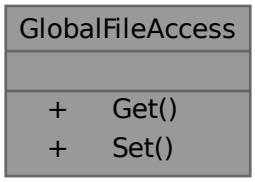
NullArgumentException	when given executor is null

4.68 GlobalFileAccess Class Reference

Holds instance of BinaryFileAccess.

```
#include <BinaryFile.h>
```

Collaboration diagram for GlobalFileAccess:



Static Public Member Functions

- static std::shared_ptr< BinaryFileAccess > Get ()

  Provides current BinaryFileAccess.
- static void Set (const std::shared_ptr< BinaryFileAccess > &fileAccess)

  Sets new global BinaryFileAccess.

4.68.1 Detailed Description

When file access is not set then default one is FileSystemAccess. The class is singleton.

4.68.2 Member Function Documentation

4.68.2.1 Get()

```
std::shared_ptr< BinaryFileAccess > Get () [inline], [static]
```

Returns

current BinaryFileAccess

4.68.2.2 Set()

```
void Set (
            const std::shared_ptr< BinaryFileAccess > & fileAccess) [inline], [static]
```

Parameters

fileAccess	new BinaryFileAccess to store

Exceptions

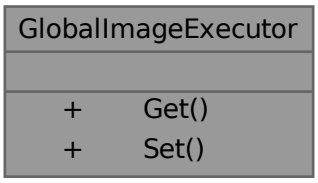
NullArgumentException	when given fileAccess is null

4.69 GlobalImageExecutor Class Reference

Holds instance of ImageExecutor.

```
#include <ImageExecutor.h>
```

Collaboration diagram for GlobalImageExecutor:



Static Public Member Functions

- static std::shared_ptr< ImageExecutor > Get ()

  Provides current executor.
- static void Set (std::shared_ptr< ImageExecutor > executor)

  Sets new global executor.

4.69.1 Detailed Description

When executor is not set then default one is LocalImageExecutor. The class is singletone.

4.69.2 Member Function Documentation

4.69.2.1 Get()

```
std::shared_ptr< ImageExecutor > Get ()  [inline], [static]
```

Returns

current executor

4.69.2.2 Set()

```
void Set (
            std::shared_ptr< ImageExecutor > executor)  [inline], [static]
```

Parameters

executor	new executor to store

Exceptions

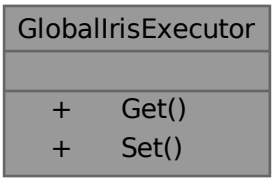
NullArgumentException	when given executor is null

4.70 GlobalIrisExecutor Class Reference

Holds instance of IrisExecutor.

```
#include <IrisExecutor.h>
```

Collaboration diagram for GlobalIrisExecutor:



Static Public Member Functions

- static std::shared_ptr< IrisExecutor > Get ()

  Provides current executor.
- static void Set (std::shared_ptr< IrisExecutor > executor)

  Sets new global executor.

4.70.1 Detailed Description

When executor is not set then default one is LocalIrisExecutor. The class is singletone.

4.70.2 Member Function Documentation

4.70.2.1 Get()

```
std::shared_ptr< IrisExecutor > Get () [inline], [static]
```

Returns

current executor

4.70.2.2 Set()

```
void Set (
            std::shared_ptr< IrisExecutor > executor) [inline], [static]
```

Parameters

executor	new executor to store

Exceptions

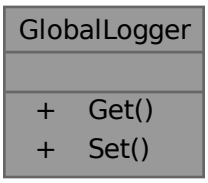
NullArgumentException	when given executor is null

4.71 GlobalLogger Class Reference

Holds instance of Logger.

```
#include <Logger.h>
```

Collaboration diagram for GlobalLogger:



Static Public Member Functions

- static std::shared_ptr< Logger > Get ()

  Provides current logger.
- static void Set (std::shared_ptr< Logger > logger)

  Sets new global logger.

4.71.1 Detailed Description

When logger is not set then default one is NullLogger. The class is singleton.

4.71.2 Member Function Documentation

4.71.2.1 Get()

```
std::shared_ptr< Logger > Get () [inline], [static]
```

Returns

current logger

4.71.2.2 Set()

```
void Set (
            std::shared_ptr< Logger > logger) [inline], [static]
```

Parameters

logger	new Logger to store

Exceptions

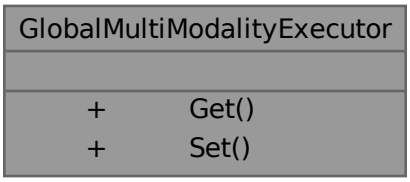
NullArgumentException	when given logger is null

4.72 GlobalMultiModalityExecutor Class Reference

Holds instance of MultiModalityExecutor.

```
#include <MultiModalityExecutor.h>
```

Collaboration diagram for GlobalMultiModalityExecutor:



Static Public Member Functions

- static std::shared_ptr< MultiModalityExecutor > Get ()

  Provides current executor.
- static void Set (std::shared_ptr< MultiModalityExecutor > executor)

  Sets new global executor.

4.72.1 Detailed Description

When executor is not set then default one is LocalMultiModalityExecutor. The class is singleton.

4.72.2 Member Function Documentation

4.72.2.1 Get()

```
std::shared_ptr< MultiModalityExecutor > Get () [inline], [static]
```

Returns

current executor

4.72.2.2 Set()

```
void Set (
            std::shared_ptr< MultiModalityExecutor > executor) [inline], [static]
```

Parameters

executor	new executor to store

Exceptions

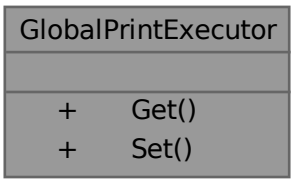
EnrollmentInvalidArgumentException	when given executor is null

4.73 GlobalPrintExecutor Class Reference

Holds instance of PrintExecutor.

```
#include <PrintExecutor.h>
```

Collaboration diagram for GlobalPrintExecutor:



Static Public Member Functions

- static std::shared_ptr< PrintExecutor > Get ()

  Provides current executor.
- static void Set (std::shared_ptr< PrintExecutor > executor)

  Sets new global executor.

4.73.1 Detailed Description

When executor is not set then default one is LocalPrintExecutor. The class is singleton.

4.73.2 Member Function Documentation

4.73.2.1 Get()

```
std::shared_ptr< PrintExecutor > Get () [inline], [static]
```

Returns

current executor

4.73.2.2 Set()

```
void Set (
            std::shared_ptr< PrintExecutor > executor) [inline], [static]
```

Parameters

executor	new executor to store

Exceptions

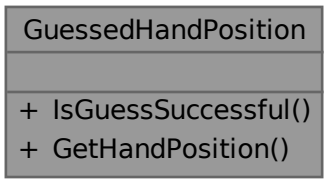
NullArgumentException	when given executor is null

4.74 GuessedHandPosition Class Reference

It represents guessed hand position of Slap.

```
#include <PrintCapture.h>
```

Collaboration diagram for GuessedHandPosition:



Public Member Functions

- bool IsGuessSuccessful () const
- HandPosition GetHandPosition () const

4.74.1 Detailed Description

See Description with examples.

4.74.2 Member Function Documentation

4.74.2.1 GetHandPosition()

```
HandPosition GetHandPosition () const  [inline], [nodiscard]
```

Returns

Guessed hand position.

Exceptions

EnrollmentRuntimeException	When it is not possible to guess hand position

4.74.2.2 IsGuessSuccessful()

```
bool IsGuessSuccessful () const  [inline], [nodiscard]
```

Returns

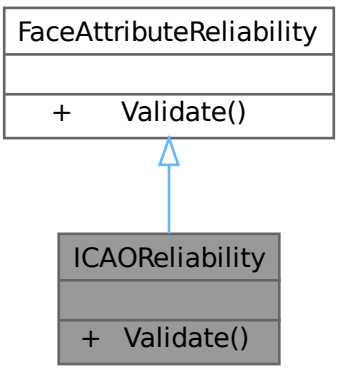
`true` if it is possible to guess.

4.75 ICAOReliability Class Reference

Check if Face will fullfil ICAO requirements.

`#include <FaceValidation.h>`

Collaboration diagram for ICAOReliability:



Public Member Functions

- std::vector< FailedFaceAttribute > Validate (const std::shared_ptr< Face > &probe) final

  Check if Face will fullfil ICAO requirements.

4.75.1 Member Function Documentation

4.75.1.1 Validate()

```
std::vector< FailedFaceAttribute > Validate (
            const std::shared_ptr< Face > & probe)  [inline], [final], [virtual]
```

Parameters

probe	Face to be checked

Returns

list of attributes that will not pass ICAO requirements

Exceptions

NullArgumentException	when probe is null

Required Solvers

Total size of solvers is 2 MB.

References face-attribute-heavy-frame.solver, face-attribute-mouth-status.solver, face-attribute-eye-status.solver, face-attribute-eye-gaze.solver, face-attribute-background-uniformity.solver

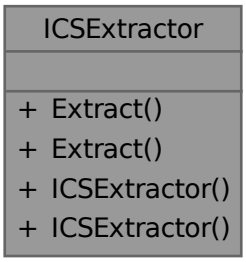
Implements FaceAttributeReliability.

4.76 ICSExtractor Class Reference

ICSExtractor creates ICS, Innovatrics proprietary print template.

```
#include <ICSExtractor.h>
```

Collaboration diagram for ICSExtractor:



Classes

- class Artifact

  Interface that generalize storing of intermediate images and other artifacts that are produced during extraction.
- class Artifacts

  It contains all added artifacts that were created during extraction.
- class SkeletonImage

  It enables creation of skeleton image during extraction.
- class FilteredImage

  It enables creation of filtered image during extraction.
- class BinarizedImage

  It enables creation of binarized image during extraction.

Public Member Functions

- std::shared_ptr< ExtractedPrint > Extract (std::shared_ptr< Print > print, Artifact &artifact)

  Function creates ICS template for given Print.
- std::shared_ptr< ExtractedPrint > Extract (const std::shared_ptr< Print > &print)

  Function creates ICS template for given Print.
- ICSExtractor ()

  Constructs new ICSExtractor.
- ICSExtractor (std::shared_ptr< PrintExecutor > executor)

  Constructs new ICSExtractor.

4.76.1 Detailed Description

The template always contains also critical points.

4.76.2 Constructor & Destructor Documentation

4.76.2.1 ICSExtractor() [1/2]

```
ICSExtractor () [inline]
```

GlobalPrintExecutor is used extraction execution.

4.76.2.2 ICSExtractor() [2/2]

```
ICSExtractor (
            std::shared_ptr< PrintExecutor > executor)  [inline], [explicit]
```

Parameters

executor	PrintExecutor to use for extraction execution

Exceptions

NullArgumentException	when executor is null

4.76.3 Member Function Documentation

4.76.3.1 Extract() [1/2]

```
std::shared_ptr< ExtractedPrint > Extract (
            const std::shared_ptr< Print > & print)  [inline]
```

The IEngine library is used to create ICS template. The critical points are always extracted.

Parameters

print	Print to extract

Returns

template calculated template

Exceptions

EnrollmentException	when extraction fails
NullArgumentException	when print is null

4.76.3.2 Extract() [2/2]

```
std::shared_ptr< ExtractedPrint > Extract (
            std::shared_ptr< Print > print,
            Artifact & artifact)  [inline]
```

The IEngine library is used to create ICS template. The critical points are always extracted.

Parameters

print	Print to extract
artifact	Artifact to store during extraction

Returns

template calculated template

Exceptions

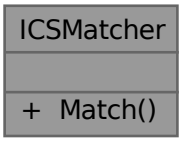
EnrollmentException	when extraction fails
NullArgumentException	when print is null

4.77 ICSMatcher Class Reference

It is general matcher for ICS templates.

```
#include <ICSMatcher.h>
```

Collaboration diagram for ICSMatcher:



Public Member Functions

- virtual PrintSimilarityScore Match (const ICSTemplate &probe, const ICSTemplate &gallery) const =0

  Calculates similarity score of probe and gallery template.

4.77.1 Detailed Description

It compares probe and gallery template. Result of comparison is SimilarityScore.

4.77.2 Member Function Documentation

4.77.2.1 Match()

```
virtual PrintSimilarityScore Match (
            const ICSTemplate & probe,
            const ICSTemplate & gallery) const  [nodiscard], [pure virtual]
```

Parameters

probe	probe Template
gallery	gallery Template

Returns

similarity SimilarityScore

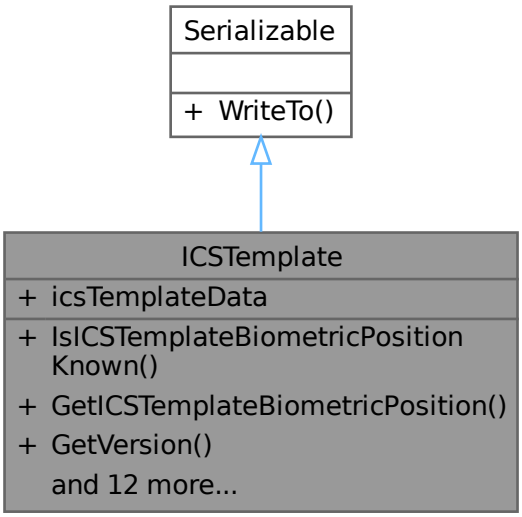
Implemented in ICSVerifyMatcher.

4.78 ICSTemplate Class Reference

Print template in ICS format used for biometric operations.

```
#include <ICSTemplate.h>
```

Collaboration diagram for ICSTemplate:



Public Member Functions

- bool IsICSTemplateBiometricPositionKnown () const

  It checks if template contains known biometric position.
- PrintPosition GetICSTemplateBiometricPosition () const

  It returns biometric position to which template belongs.
- std::string GetVersion () const

  Provides version information of Innovatrics proprietary algorithm used to create the ICS template.
- std::shared_ptr< ICSTemplate > Anonymize () const

  Creates ICSTemplate without position.
- ICSTemplate (std::vector< uint8_t > d)

  Create ICSTemplate object GlobalPrintExecutor is used to execute template functionality.
- ICSTemplate (std::vector< uint8_t > d, std::shared_ptr< PrintExecutor > executor)

  Create ICSTemplate object Biometric position is loaded from template is possible.
- ICSTemplate (std::vector< uint8_t > d, PrintPosition position)

  Create ICSTemplate object GlobalPrintExecutor is used to execute template functionality.
- ICSTemplate (std::vector< uint8_t > d, PrintPosition position, std::shared_ptr< PrintExecutor > executor)

  Create ICSTemplate object The template format is validated.
- PrintSimilarityScore SimilarWith (const ICSTemplate &tmpl, const ICSMatcher &matcher) const

  It calculates similarity score between this and given ICSTemplate.
- uint8_t CalculateQuality ()

  Function calculates a fingerprint template quality.
- std::shared_ptr< ISOTemplate > ToISO (const std::shared_ptr< const Print > &print) const

  Convert this ICSTemplate to into ISO/IEC 19794-2 compliant template.
- std::shared_ptr< ANSITemplate > ToANSI (const std::shared_ptr< const Print > &print) const

  Convert this ICSTemplate to into ANSI/INCITS 378 compliant template.
- std::vector< std::shared_ptr< Minutia > > GetMinutiae () const

  Get minutiae from template.
- std::vector< std::shared_ptr< CriticalPoint > > GetCriticalPoints () const

  Get critical points (core/delta) from template.
- void WriteTo (Writer &w) const override

  Write itself into writer.

Public Attributes

- const std::vector< uint8_t > **icsTemplateData**

  template data

4.78.1 Detailed Description

The template contains one and only one print template.

4.78.2 Constructor & Destructor Documentation

4.78.2.1 ICSTemplate() [1/4]

```
ICSTemplate (
            std::vector< uint8_t > d)  [inline], [explicit]
```

Biometric position is loaded from template is possible. The template format is validated.

Parameters

d	template data

Exceptions

InvalidTemplateException	when ICS template has no data
EnrollmentException	when ICS template has no print templates
EnrollmentException	when ICS template has more print templates
EnrollmentException	when ICS template has templates of other modalities
EnrollmentException	when it is not possible to parse template

4.78.2.2 ICSTemplate() [2/4]

```
ICSTemplate (
            std::vector< uint8_t > d,
            std::shared_ptr< PrintExecutor > executor)  [inline]
```

The template format is validated.

Parameters

d	template data
executor	PrintExecutor used to execute template functionality.

Exceptions

NullArgumentException	when executor is null
InvalidTemplateException	when ICS template has no data
EnrollmentException	when ICS template has no print templates
EnrollmentException	when ICS template has more print templates
EnrollmentException	when ICS template has templates of other modalities
EnrollmentException	when it is not possible to parse template

4.78.2.3 ICSTemplate() [3/4]

```
ICSTemplate (
            std::vector< uint8_t > d,
            PrintPosition position)  [inline], [explicit]
```

The template format is validated. The given position is stored in template data is possible.

Parameters

d	template data
position	biometric position to which template belongs

Exceptions

InvalidTemplateException	when ICS template has no data
EnrollmentException	when ICS template has no print templates
EnrollmentException	when ICS template has more print templates
EnrollmentException	when ICS template has templates of other modalities
EnrollmentException	when it is not possible to parse template

4.78.2.4 ICSTemplate() [4/4]

```
ICSTemplate (
            std::vector< uint8_t > d,
            PrintPosition position,
            std::shared_ptr< PrintExecutor > executor)  [inline]
```

The given position is stored in template data is possible.

Parameters

d	template data
position	biometric position to which template belongs
executor	PrintExecutor used to execute template functionality.

Exceptions

NullArgumentException	when executor is null
InvalidTemplateException	when ICS template has no data
EnrollmentException	when ICS template has no print templates
EnrollmentException	when ICS template has more print templates
EnrollmentException	when ICS template has templates of other modalities
EnrollmentException	when it is not possible to parse template

4.78.3 Member Function Documentation

4.78.3.1 Anonymize()

```
std::shared_ptr< ICSTemplate > Anonymize () const [inline], [nodiscard]
```

Returns

ICSTemplate without position.

4.78.3.2 CalculateQuality()

```
uint8_t CalculateQuality () [inline]
```

This function returns quality of a fingerprint template. For enrollment, it is recommended to use only templates with quality > 30.

Returns

the output range is from 0(blank image) and 100(maximum quality)

Exceptions

EnrollmentException	when fails to calculate template quality

4.78.3.3 GetCriticalPoints()

```
std::vector< std::shared_ptr< CriticalPoint > > GetCriticalPoints () const [inline]
```

The extraction of critical points is optional. The Enrollment SDK always extracts critical points. Other Innovatrics SDKs can produce template without critical points.

Note

The adjudicator feature MUST be allowed in license to get critical points.

Returns

CriticalPoint

4.78.3.4 GetICSTemplateBiometricPosition()

```
PrintPosition GetICSTemplateBiometricPosition () const [inline], [nodiscard]
```

Returns

biometric position

Exceptions

EnrollmentRuntimeException	when biometric position is unknown

4.78.3.5 GetMinutiae()

```
std::vector< std::shared_ptr< Minutia > > GetMinutiae () const [inline]
```

Note

The adjudicator feature MUST be allowed in license to get minutiae.

Returns

Minutiae

4.78.3.6 GetVersion()

```
std::string GetVersion () const [inline], [nodiscard]
```

Returns

version string

4.78.3.7 IsICSTemplateBiometricPositionKnown()

```
bool IsICSTemplateBiometricPositionKnown () const [inline], [nodiscard]
```

Returns

`true` when biometric position is known and valid

`false` when biometric position is unknown

4.78.3.8 SimilarWith()

```
PrintSimilarityScore SimilarWith (
            const ICSTemplate & tmpl,
            const ICSMatcher & matcher) const  [inline]
```

It uses given ICSMatcher to calculate score.

Parameters

tmpl	gallery ICSTemplate.
matcher	is ICSMatcher used to calculate similarity score between probe (this template) and given template `tmpl`

Returns

similarity score

4.78.3.9 ToANSI()

```
std::shared_ptr< ANSITemplate > ToANSI (
            const std::shared_ptr< const Print > & print) const  [inline]
```

Returns

ANSITemplate

Parameters

print	it is Print from which ICSTemplate was extracted.

Exceptions

EnrollmentException	when it is not possible to convert template

4.78.3.10 ToISO()

```
std::shared_ptr< ISOTemplate > ToISO (
            const std::shared_ptr< const Print > & print) const  [inline]
```

Returns

ISOTemplate

Parameters

print	it is Print from which ICSTemplate was extracted.

Exceptions

EnrollmentException	when it is not possible to convert template

4.78.3.11 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [override], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

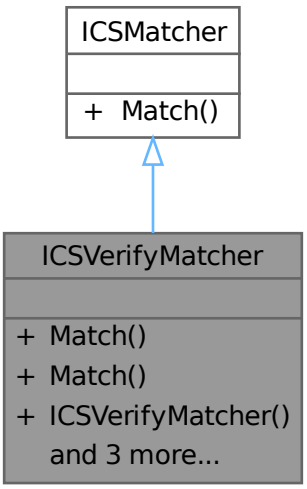
Implements Serializable.

4.79 ICSVerifyMatcher Class Reference

ICSVerifyMatcher is template matcher that uses Innovatrics proprietary verification algorithm.

```
#include <ICSVerifyMatcher.h>
```

Collaboration diagram for ICSVerifyMatcher:



Classes

- class Artifact

  Interface that generalize storing of artifacts that are produced during matching.
- class Artifacts

  It contains all added artifacts that were created during matching.
- class MatchingMinutiae

  It enables creation of matching minutiae during matching.
- class Config

  ICSVerifyMatcher configuration.

Public Member Functions

- PrintSimilarityScore Match (const ICSTemplate &probe, const ICSTemplate &gallery) const final

  Calculates similarity score of probe and gallery template.
- PrintSimilarityScore Match (const ICSTemplate &probe, const ICSTemplate &gallery, Artifact &artifact) const

  Calculates similarity score of probe and gallery template and provide artifacts.
- **ICSVerifyMatcher** ()

  Constructor with default Config GlobalPrintExecutor is used to execute matching.
- ICSVerifyMatcher (Config config)

  Constructor uses Config to setup matching algorithm.
- ICSVerifyMatcher (Config config, std::shared_ptr< PrintExecutor > executor)

  Constructor uses Config to setup matching algorithm.
- ICSVerifyMatcher (std::shared_ptr< PrintExecutor > executor)

  Constructor with default Config.

4.79.1 Detailed Description

It is configurable via ICSVerifyMatcher::Config verifies probe against gallery template.

4.79.2 Constructor & Destructor Documentation

4.79.2.1 ICSVerifyMatcher() [1/3]

```
ICSVerifyMatcher (
            Config config)  [inline], [explicit]
```

GlobalPrintExecutor is used to execute matching.

Parameters

config	is matching algorithm configuration.

4.79.2.2 ICSVerifyMatcher() [2/3]

```
ICSVerifyMatcher (
            Config config,
            std::shared_ptr< PrintExecutor > executor)  [inline]
```

Parameters

config	is matching algorithm configuration.
executor	PrintExecutor used for matching

Exceptions

NullArgumentException	when executor is null

4.79.2.3 ICSVerifyMatcher() [3/3]

```
ICSVerifyMatcher (
            std::shared_ptr< PrintExecutor > executor)  [inline], [explicit]
```

Parameters

executor	PrintExecutor used for matching

Exceptions

NullArgumentException	when executor is null

4.79.3 Member Function Documentation

4.79.3.1 Match() [1/2]

```
PrintSimilarityScore Match (
            const ICSTemplate & probe,
            const ICSTemplate & gallery) const  [inline], [nodiscard], [final], [virtual]
```

Parameters

probe	probe ICSTemplate
gallery	gallery ICSTemplate

Returns

similarity SimilarityScore

Implements ICSMatcher.

4.79.3.2 Match() [2/2]

```
PrintSimilarityScore Match (
            const ICSTemplate & probe,
            const ICSTemplate & gallery,
            Artifact & artifact) const  [inline], [nodiscard]
```

Parameters

probe	probe ICSTemplate
gallery	gallery ICSTemplate
artifact	Artifact

Returns

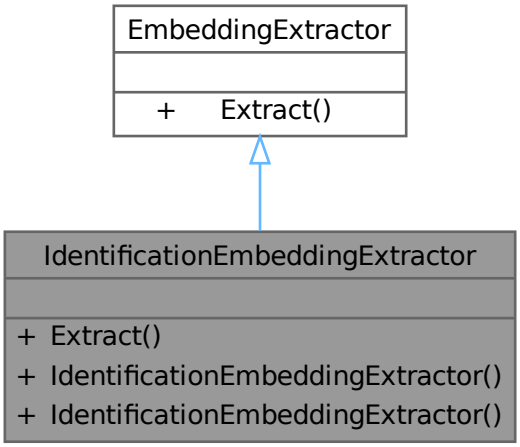
similarity SimilarityScore

4.80 IdentificationEmbeddingExtractor Class Reference

EmbeddingExtractor creates Innovatrics proprietary embedding print template.

`#include <IdentificationEmbeddingExtractor.h>`

Collaboration diagram for IdentificationEmbeddingExtractor:



Public Member Functions

- std::shared_ptr< ExtractedPrintEmbedding > Extract (const std::shared_ptr< Print > &print) const final

  Function creates embedding template for given Print.
- IdentificationEmbeddingExtractor ()

  Constructs new IdentificationEmbeddingExtractor.
- IdentificationEmbeddingExtractor (std::shared_ptr< PrintExecutor > executor)

  Constructs new IdentificationEmbeddingExtractor.

4.80.1 Detailed Description

Note

Not supported in this variant of Enrollment SDK.

The embedding extraction feature MUST be allowed in license to extract embedding.

4.80.2 Constructor & Destructor Documentation

4.80.2.1 IdentificationEmbeddingExtractor() [1/2]

```
IdentificationEmbeddingExtractor () [inline]
```

GlobalPrintExecutor is used extraction execution.

Note

Not supported in this variant of Enrollment SDK.

Exceptions

EnrollmentException	when cannot initialize module

4.80.2.2 IdentificationEmbeddingExtractor() [2/2]

```
IdentificationEmbeddingExtractor (
            std::shared_ptr< PrintExecutor > executor) [inline], [explicit]
```

Note

Not supported in this variant of Enrollment SDK.

Parameters

executor	PrintExecutor to use for extraction execution

Exceptions

NullArgumentException	when executor is null
EnrollmentException	when cannot initialize module

4.80.3 Member Function Documentation

4.80.3.1 Extract()

```
std::shared_ptr< ExtractedPrintEmbedding > Extract (
            const std::shared_ptr< Print > & print) const  [inline], [nodiscard], [final],
[virtual]
```

The IFinger library is used to create embedding template.

Note

Not supported in this variant of Enrollment SDK.

The embedding extraction feature MUST be allowed in license to extract embedding.

Parameters

print	Print to extract

Returns

extracted embedding

Exceptions

NullArgumentException	when print is null

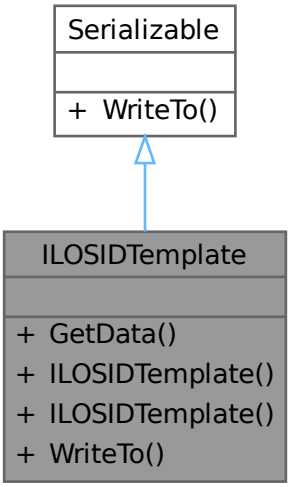
Implements EmbeddingExtractor.

4.81 ILOSIDTemplate Class Reference

International Labour Organisation Seafarers Identity Document template.

```
#include <ILOSIDTemplate.h>
```

Collaboration diagram for ILOSIDTemplate:



Public Member Functions

- std::vector< uint8_t > GetData () const

  Provides template data.
- ILOSIDTemplate (std::vector< uint8_t > &&d)

  Create ILOSIDTemplate object move data to template.
- ILOSIDTemplate (const std::vector< uint8_t > &d)

  Create ILOSIDTemplate object, copy data to template.
- void WriteTo (Writer &writer) const final

  Write itself into writer.

4.81.1 Detailed Description

The template contains data compliant with the format specified in Seafarers Identity Documents convention (Revised), 2003 (No. 185) in Annex B [3] .

4.81.2 Constructor & Destructor Documentation

4.81.2.1 ILOSIDTemplate() [1/2]

```
ILOSIDTemplate (
            std::vector< uint8_t > && d)  [inline], [explicit]
```

Parameters

d	template data

Exceptions

InvalidTemplateException	when template data are empty

4.81.2.2 ILOSIDTemplate() [2/2]

```
ILOSIDTemplate (
            const std::vector< uint8_t > & d)  [inline], [explicit]
```

Parameters

d	template data

Exceptions

InvalidTemplateException	when template data are empty

4.81.3 Member Function Documentation

4.81.3.1 GetData()

```
std::vector< uint8_t > GetData () const  [inline], [nodiscard]
```

Returns

template data

4.81.3.2 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

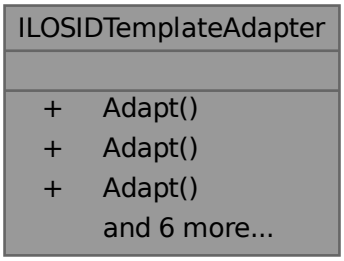
Implements Serializable.

4.82 ILOSIDTemplateAdapter Class Reference

Adapts Template to ILOSIDTemplate.

```
#include <ILOSIDTemplateAdapter.h>
```

Collaboration diagram for ILOSIDTemplateAdapter:



Public Member Functions

- ILOSIDTemplate Adapt (const ANSITemplate &tmpl) const

  Adapts ANSITemplate to ILOSIDTemplate.
- ILOSIDTemplate Adapt (const ISONCTemplate &tmpl) const

  Adapts ISONCTemplate to ILOSIDTemplate.
- ILOSIDTemplate Adapt (const ISOCCTemplate &tmpl) const

  Adapts ISOCCTemplate to ILOSIDTemplate.
- ILOSIDTemplate Adapt (const ISOTemplate &tmpl) const

  Adapts ISOTemplate to ILOSIDTemplate.
- ILOSIDTemplate Adapt (const ICSTemplate &tmpl) const

  Adapts ICSTemplate to ILOSIDTemplate.
- ILOSIDTemplate Adapt (const ExtractedPrint &tmpl) const

  Adapts ExtractedPrint to ILOSIDTemplate.
- ILOSIDTemplate Adapt (const PrintTemplate &tmpl) const

  Adapts PrintTemplate to ILOSIDTemplate.
- **ILOSIDTemplateAdapter** ()

  Creates ILOSIDTemplateAdapter object with GlobalPrintExecutor used to execute template functionality.
- ILOSIDTemplateAdapter (std::shared_ptr< PrintExecutor > executor)

  Creates ILOSIDTemplateAdapter object.

4.82.1 Constructor & Destructor Documentation

4.82.1.1 ILOSIDTemplateAdapter()

```
ILOSIDTemplateAdapter (
            std::shared_ptr< PrintExecutor > executor)  [inline], [explicit]
```

Parameters

executor	PrintExecutor used to execute template functionality.

Exceptions

NullArgumentException	when executor is null

4.82.2 Member Function Documentation

4.82.2.1 Adapt() [1/7]

```
ILOSIDTemplate Adapt (
            const ANSITemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ILOSIDTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.82.2.2 Adapt() [2/7]

```
ILOSIDTemplate Adapt (
            const ExtractedPrint & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ILOSIDTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.82.2.3 Adapt() [3/7]

```
ILOSIDTemplate Adapt (
            const ICSTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ILOSIDTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.82.2.4 Adapt() [4/7]

```
ILOSIDTemplate Adapt (
            const ISOCCTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ILOSIDTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.82.2.5 Adapt() [5/7]

```
ILOSIDTemplate Adapt (
            const ISONCTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ILOSIDTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.82.2.6 Adapt() [6/7]

```
ILOSIDTemplate Adapt (
            const ISOTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ILOSIDTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.82.2.7 Adapt() [7/7]

```
ILOSIDTemplate Adapt (
            const PrintTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ILOSIDTemplate

Exceptions

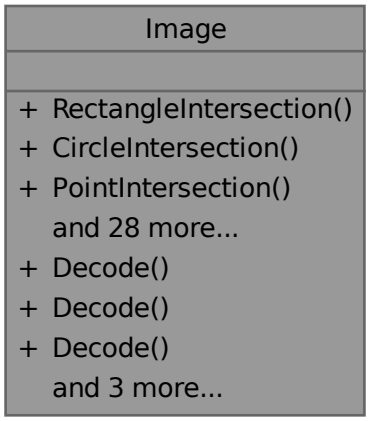
EnrollmentException	when it is not possible to adapt template

4.83 Image Class Reference

Image representation.

```
#include <Image.h>
```

Collaboration diagram for Image:



Classes

- class ImageResampler

  Image resampler for DPI-change operations, including upsampling and downsampling.
- class BicubicImageResampler

  Image resampler for DPI-change operations, including upsampling and downsampling using bicubic interpolation resampling algorithm.
- class AreaImageResampler

  Image resampler for DPI-change operations, including upsampling and downsampling using area interpolation resampling algorithm.
- class NISTImageResampler

  Image resampler for DPI-change operations, including upsampling and downsampling using NIST-compliant resampling algorithm provided by `NFIR library`.

Public Types

- enum class Intersection : uint8_t {
  PARTIAL , INSIDE , OUTSIDE , NO ,
  SAME }

  Enum define possible intersection of Image and some Object e.g. Rectangle, Circle.

Public Member Functions

- Intersection RectangleIntersection (const Rectangle &rectangle) const

  Check intersection between rectangle and image.
- Intersection CircleIntersection (const Circle &circle) const

  Check intersection between circle and image.
- Intersection PointIntersection (const Point &point) const

  Checks intersection between point and image.
- ImageDpi Dpi () const

  Returns image dpi.
- ImageWidth Width () const

  Returns image width.
- ImageHeight Height () const

  Returns image height.
- std::shared_ptr< ImageEncoder > OriginalEncoder () const

  Provides original encoder to encode image to original format.
- RawImage AsRawImage () const

  Converts image to RAW format that has been originally used.
- RawImage AsRawGrayscale () const

  Converts image to raw grayscale format.
- RawImage AsRawBGR () const

  Converts image to raw BGR format.
- std::shared_ptr< Image > AsBGR () const

  Converts image to BGR format.
- RawImage AsRawBGRA () const

  Converts image to raw BGRA format.
- std::shared_ptr< Image > AsBGRA () const

  Converts image to BGRA format.
- std::vector< uint8_t > EncodeIn (ImageEncoder &encoder) const

  Encode image in given ImageEncoder.
- void SaveIn (ImageSaver &saver) const

  Save Image in given ImageSaver.
- void SaveAs (const std::string &path) const

  Save Image in given path.
- bool operator== (const Image &right) const

  Images are equal when has same dimensions and same content.
- std::shared_ptr< Image > DrawShape (const ImageShape &shape) const

  Draws shape to new image.
- std::shared_ptr< Image > CropRectangle (const Rectangle &r) const

  Crops a rectangle from the image, using a white background for any extra space.
- std::shared_ptr< Image > CropRectangle (const Rectangle &r, const RGBA &background) const

  Crops a rectangle from the image, using a white background for any extra space.
- std::shared_ptr< Image > CropRectangle (const Rectangle &r, const RGBA &background, const ImageDimensions &dimension) const

Crops a rectangle and places it on a canvas of specified dimensions, using a specified background color.

- std::shared_ptr< Image > ChangeDpiTo (const ImageResampler &resampler) const

  Create image using provided DPI-change configuration.
- std::shared_ptr< Image > ChangeDpiTo (ImageDpi targetDpi) const

  Create image with given DPI.
- std::shared_ptr< Image > Resize (float factor) const

  Resize image to new image.
- std::shared_ptr< Image > ResizeToMaxDimension (ImageDimension maxDimension) const

  Resize image to new image with given max dimension.
- std::shared_ptr< Image > CropAlignedRectangle (const Rectangle &r) const

  Crop rectangle and align given top left rectangle point to coordinates [0,0] in new image.
- std::shared_ptr< Image > CropAlignedRectangle (const Rectangle &r, const RGBA &background) const

  Crop rectangle and align given top left rectangle point to coordinates [0,0] in new image.
- std::shared_ptr< Image > CropAlignedRectangle (const Rectangle &r, const RGBA &background, const ImageDimensions &dimension) const

  Crop rectangle and align given top left rectangle point to coordinates [0,0] in new image.
- std::shared_ptr< Image > CropCircle (const Circle &circle, const RGBA &background=Colors::White) const

  Copy circle to new image.
- std::shared_ptr< Image > MaskImageBy (const ImageMask &mask) const

  Applies the specified mask to the image, creating a new masked image.
- size_t Channels () const

  Returns the number of channels in the image.

Static Public Member Functions

- static std::shared_ptr< Image > Decode (const std::vector< uint8_t > &imageData)

  Creates Image from image data with 500 DPI.
- static std::shared_ptr< Image > Decode (const std::vector< uint8_t > &imageData, unsigned int dpi)

  Creates Image from image data with given dpi.
- static std::shared_ptr< Image > Decode (const std::vector< uint8_t > &imageData, std::shared_ptr< Image↩ Executor > executor)

  Creates Image from image data with 500 DPI.
- static std::shared_ptr< Image > Decode (const std::vector< uint8_t > &imageData, unsigned int dpi, std::↩ shared_ptr< ImageExecutor > executor)

  Creates Image from image data with given DPI.
- static std::shared_ptr< Image > DecodeRawImage (const RawImage &raw, std::shared_ptr< ImageEncoder > originalEncoder=nullptr, std::shared_ptr< ImageExecutor > executor=GlobalImageExecutor::Get())

  Creates Image from raw data.
- static ImageType DeduceType (const std::vector< uint8_t > &imageData)

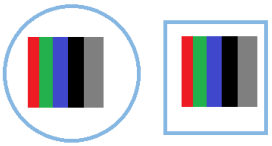
  Deduce image type from image content.

4.83.1 Member Enumeration Documentation

4.83.1.1 Intersection

```
enum class Intersection : uint8_t [strong]
```

Enumerator

PARTIAL	The Object and Image has intersection. **Figure 125 Partial intersection**
INSIDE	Object is fully in Image. **Figure 126 Inside intersection**
OUTSIDE	Image is fully in Object.  **Figure 127 Outside intersection**
NO	Image and Object has no intersection. **Figure 128 No intersection**
SAME	Image has same dimensions and vertices coordinates as Object. **Figure 129 Same intersection**

4.83.2 Member Function Documentation

4.83.2.1 AsBGR()

```
std::shared_ptr< Image > AsBGR () const [inline], [nodiscard]
```

Returns

Image.

Exceptions

ImageException	when there is an processing error

4.83.2.2 AsBGRA()

```
std::shared_ptr< Image > AsBGRA () const [inline], [nodiscard]
```

Returns

Image.

Exceptions

ImageException	when there is an processing error

4.83.2.3 AsRawBGR()

```
RawImage AsRawBGR () const [inline], [nodiscard]
```

Returns

RawImage.

Exceptions

ImageException	when there is an processing error

4.83.2.4 AsRawBGRA()

```
RawImage AsRawBGRA () const [inline], [nodiscard]
```

Returns

RawImage.

Exceptions

ImageException	when there is an processing error

4.83.2.5 AsRawGrayscale()

```
RawImage AsRawGrayscale () const [inline], [nodiscard]
```

Returns

RawImage.

Exceptions

ImageException	when there is an processing error

4.83.2.6 AsRawImage()

```
RawImage AsRawImage () const [inline], [nodiscard]
```

Returns

RawImage.

Exceptions

ImageException	when there is an processing error

4.83.2.7 ChangeDpiTo() [1/2]

```
std::shared_ptr< Image > ChangeDpiTo (
            const ImageResampler & resampler) const  [inline], [nodiscard]
```

It creates new image as result of resizing this image using a customized resizing strategy with factor `target↩Dpi/originalDPI`.

Parameters

resampler	image resampler

Returns

image with given DPI

Exceptions

ImageException	when there is an processing error
EnrollmentInvalidArgumentException	when resize algorithm is NIST and image is not single-channel grayscale

4.83.2.8 ChangeDpiTo() [2/2]

```
std::shared_ptr< Image > ChangeDpiTo (
            ImageDpi targetDpi) const  [inline], [nodiscard]
```

It creates new image as result of resizing this image with factor `newDPI/originalDPI`. The function uses Standard resizing algorithm. See BicubicImageResampler.

Parameters

targetDpi	DPI of new image.

Returns

image with given DPI

Exceptions

ImageException	when there is an processing error
EnrollmentInvalidArgumentException	when targetDpi is 0

4.83.2.9 Channels()

```
size_t Channels () const  [inline], [nodiscard]
```

- A value of 1 typically represents a grayscale image.
- A value of 3 typically represents a color image (e.g., RGB).
- A value of 4 typically represents a color image with transparency (e.g., RGBA).

Returns

The number of channels in the image.

4.83.2.10 CircleIntersection()

```
Intersection CircleIntersection (
            const Circle & circle) const  [inline], [nodiscard]
```

Parameters

circle	Circle

Returns

Image::Intersection type

Exceptions

ImageException	when there is an processing error

4.83.2.11 CropAlignedRectangle() [1/3]

```
std::shared_ptr< Image > CropAlignedRectangle (
            const Rectangle & r) const  [inline], [nodiscard]
```

For background outside image it will use Color::White. The dimensions of the output image will be equal to those of the rectangle.

See Description with examples.

Parameters

r	Rectangle

Returns

copied and aligned Image

Exceptions

ImageException	when there is an processing error

4.83.2.12 CropAlignedRectangle() [2/3]

```
std::shared_ptr< Image > CropAlignedRectangle (
            const Rectangle & r,
            const RGBA & background) const  [inline], [nodiscard]
```

The function will utilize the provided color for the background outside the image. The dimensions of the output image will be equal to those of the rectangle.

See Description with examples.

Parameters

r	Rectangle
background	color of background when rectangle is out of image

Returns

copied and aligned Image

Exceptions

ImageException	when there is an processing error

4.83.2.13 CropAlignedRectangle() [3/3]

```
std::shared_ptr< Image > CropAlignedRectangle (
            const Rectangle & r,
            const RGBA & background,
            const ImageDimensions & dimension) const  [inline], [nodiscard]
```

The function will render an image using specified dimensions, incorporating a cropped aligned rectangle within. For areas outside the cropped rectangle, the specified background color will be applied. If the dimensions of the cropped rectangle differ from those of the output image, the function will center the cropped rectangle within the output image. Any additional space in the output image, resulting from a size discrepancy with the cropped rectangle, will be filled with the chosen background color.

See Description with examples.

Parameters

r	Rectangle
background	color of background when rectangle is out of image
dimension	requested dimension of new image.

Returns

copied and aligned Image

Exceptions

ImageException	when there is an processing error

4.83.2.14 CropCircle()

```
std::shared_ptr< Image > CropCircle (
            const Circle & circle,
            const RGBA & background = Colors::White) const  [inline], [nodiscard]
```

See Description with examples.

Parameters

circle	circle to cut.
background	color of background around circle

Returns

copied Image

Exceptions

EnrollmentInvalidArgumentException	when given circle and image has no intersection
ImageException	when there is an processing error

4.83.2.15 CropRectangle() [1/3]

```
std::shared_ptr< Image > CropRectangle (
            const Rectangle & r) const  [inline], [nodiscard]
```

This method crops a rectangle from the image. If the rectangle extends beyond the image boundaries, the additional space is filled with a white background. The size of the resulting image matches the dimensions of the specified rectangle.

See Description with examples.

Parameters

r	Rectangle defining the area to crop.

Returns

A cropped image

Exceptions

EnrollmentInvalidArgumentException	if the rectangle does not intersect with the image.
ImageException	when there is an processing error

4.83.2.16 CropRectangle() [2/3]

```
std::shared_ptr< Image > CropRectangle (
            const Rectangle & r,
            const RGBA & background) const  [inline], [nodiscard]
```

This method crops a rectangle from the image. If the rectangle extends beyond the image boundaries, the additional space is filled with a specified background. The size of the resulting image matches the dimensions of the specified rectangle.

See Description with examples.

Parameters

r	Rectangle defining the area to crop.
background	Color for the background surrounding the rectangle.

Returns

A cropped image.

Exceptions

EnrollmentInvalidArgumentException	if the rectangle does not intersect with the image.

4.83.2.17 CropRectangle() [3/3]

```
std::shared_ptr< Image > CropRectangle (
            const Rectangle & r,
            const RGBA & background,
            const ImageDimensions & dimension) const  [inline], [nodiscard]
```

This function crops a rectangle and centers it on a new canvas with the specified dimensions. If the new canvas is larger than the cropped rectangle, the extra space is filled with the specified background color. If the rectangle is larger than the original image, the background color fills the extra space.

See Description with examples.

Parameters

r	Rectangle defining the area to crop.
background	Color for the background.
dimension	Dimensions of the new canvas. If width or height is 0, dimensions are set based on the rectangle.

Returns

A cropped image on a new canvas.

Exceptions

EnrollmentInvalidArgumentException	if the rectangle does not intersect with the image.

4.83.2.18 Decode() [1/4]

```
std::shared_ptr< Image > Decode (
            const std::vector< uint8_t > & imageData)  [inline], [static]
```

See Description with examples. It uses default dpi (500). It uses GlobalImageExecutor for execution of image functionality.

Parameters

imageData	image content

Returns

Image object of this class

Exceptions

ImageException	when type is not supported or image data are corrupted
NullArgumentException	when executor is null

4.83.2.19 Decode() [2/4]

```
std::shared_ptr< Image > Decode (
            const std::vector< uint8_t > & imageData,
            std::shared_ptr< ImageExecutor > executor)  [inline], [static]
```

See Description with examples. It uses default dpi (500).

Parameters

imageData	image content
executor	image functionality executor (the execution strategy)

Returns

Image object of this class

Exceptions

ImageException	when type is not supported or image data are corrupted
NullArgumentException	when executor is null

4.83.2.20 Decode() [3/4]

```
std::shared_ptr< Image > Decode (
            const std::vector< uint8_t > & imageData,
            unsigned int dpi)  [inline], [static]
```

See Description with examples. It uses GlobalImageExecutor for execution of image functionality.

Parameters

imageData	image content
dpi	dpi of image

Returns

Image object of this class

Exceptions

ImageException	when type is not supported or image data are corrupted
NullArgumentException	when executor is null

4.83.2.21 Decode() [4/4]

```
std::shared_ptr< Image > Decode (
            const std::vector< uint8_t > & imageData,
            unsigned int dpi,
            std::shared_ptr< ImageExecutor > executor)  [inline], [static]
```

See Description with examples.

Parameters

imageData	image content
dpi	dpi of image
executor	image functionality executor (the execution strategy)

Returns

Image object of this class

Exceptions

ImageException	when type is not supported or image data are corrupted
NullArgumentException	when executor is null

4.83.2.22 DecodeRawImage()

```
std::shared_ref< Image > DecodeRawImage (
            const RawImage & raw,
            std::shared_ptr< ImageEncoder > originalEncoder = nullptr,
            std::shared_ptr< ImageExecutor > executor = GlobalImageExecutor::Get())
[inline], [static]
```

See Description with examples.

Parameters

raw	RawImage
originalEncoder	original encoder
executor	image functionality executor (the execution strategy)

Returns

Image object of this class

Exceptions

ImageException	when type is not supported or image data are corrupted

4.83.2.23 DeduceType()

```
ImageType DeduceType (
            const std::vector< uint8_t > & imageData)  [inline], [static]
```

Parameters

imageData	image content

Returns

ImageType image type

Exceptions

ImageException	when image type is unknown
ImageException	when there is an processing error

4.83.2.24 Dpi()

```
ImageDpi Dpi () const  [inline], [nodiscard]
```

Returns

ImageDpi

4.83.2.25 DrawShape()

```
std::shared_ptr< Image > DrawShape (
            const ImageShape & shape) const  [inline], [nodiscard]
```

See Description with examples.

Parameters

shape	ImageShape

Returns

Image image with drown shape

Exceptions

ImageException	when there is an processing error

4.83.2.26 EncodeIn()

```
std::vector< uint8_t > EncodeIn (
            ImageEncoder & encoder) const  [inline], [nodiscard]
```

Parameters

encoder	specific ImageEncoder

Returns

encoded image

Exceptions

ImageException	when ImageEncoder throws
ImageException	when it is not possible convert image to raw data

4.83.2.27 Height()

```
ImageHeight Height () const  [inline], [nodiscard]
```

Returns

ImageHeight

4.83.2.28 MaskImageBy()

```
std::shared_ptr< Image > MaskImageBy (
            const ImageMask & mask) const  [inline], [nodiscard]
```

Parameters

mask	The ImageMask to apply to the current image. This mask defines the areas of the image to be modified or preserved according to the mask's specific logic.

Returns

New image instance that represents the result of applying the mask to the current image.

4.83.2.29 operator==()

```
bool operator== (
            const Image & right) const  [inline]
```

Parameters

right	right image in comparison

Returns

true - images have same dimensions and content

false - images have different dimensions or content

Exceptions

ImageException	when there is an processing error

4.83.2.30 OriginalEncoder()

```
std::shared_ptr< ImageEncoder > OriginalEncoder () const  [inline], [nodiscard]
```

Returns

ImageEncoder

4.83.2.31 PointIntersection()

```
Intersection PointIntersection (
            const Point & point) const  [inline], [nodiscard]
```

Parameters

point	point

Returns

Image::Intersection type

4.83.2.32 RectangleIntersection()

```
Intersection RectangleIntersection (
            const Rectangle & rectangle) const  [inline], [nodiscard]
```

Parameters

rectangle	Rectangle

Returns

Image::Intersection type

Exceptions

ImageException	when there is an processing error

4.83.2.33 Resize()

```
std::shared_ptr< Image > Resize (
            float factor) const  [inline], [nodiscard]
```

Parameters

factor	resize factor. Valid values are positive real number except 0.

Returns

resized Image

Exceptions

EnrollmentInvalidArgumentException	when factor is not positive real number except 0.
ImageException	when there is an processing error

4.83.2.34 ResizeToMaxDimension()

```
std::shared_ptr< Image > ResizeToMaxDimension (
            ImageDimension maxDimension) const  [inline], [nodiscard]
```

When image has smaller dimension than maxDimension, then image is not resized. Otherwise image is resized to maxDimension with keeping aspect ratio.

Parameters

maxDimension	max dimension of new image

Returns

resized Image or copy of original image when original image has smaller dimension than maxDimension.

Exceptions

ImageException	when there is an processing error

4.83.2.35 SaveAs()

```
void SaveAs (
            const std::string & path) const  [inline]
```

The image type is deduced from file extension. The default image encoder parameters are used. If you want to change encoding parameters, then use function EncodeIn with appropriate ImageEncoder.

Parameters

path	path and file name where to save image on filesystem

Exceptions

ImageException	when ImageEncoder throws
ImageException	when it is not possible convert image to raw data

4.83.2.36 SaveIn()

```
void SaveIn (
            ImageSaver & saver) const  [inline]
```

Parameters

saver	specific ImageSaver to use to save file

Exceptions

ImageException	when ImageSaver throws

4.83.2.37 Width()

```
ImageWidth Width () const  [inline], [nodiscard]
```

Returns

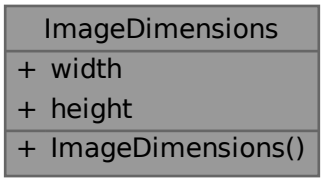
ImageWidth

4.84 ImageDimensions Class Reference

It holds width and height of Image.

```
#include <ImageAttribute.h>
```

Collaboration diagram for ImageDimensions:



Public Member Functions

- ImageDimensions (ImageWidth w, ImageHeight h)

  Constructor store given dimensions.

Public Attributes

- ImageWidth **width** = 0

  width
- ImageHeight **height** = 0

  height

4.84.1 Detailed Description

The values are always positive integers.

4.84.2 Constructor & Destructor Documentation

4.84.2.1 ImageDimensions()

```
ImageDimensions (
            ImageWidth w,
            ImageHeight h)  [inline]
```

Parameters

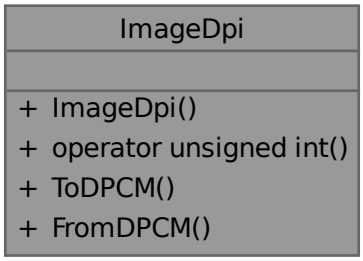
w	width
h	height

4.85 ImageDpi Class Reference

Image DPI.

```
#include <ImageAttribute.h>
```

Collaboration diagram for ImageDpi:



Public Member Functions

- constexpr ImageDpi (unsigned int i)

  Construct a new DPI.
- **operator unsigned int** () const

  Conversion to capture equipment id number.
- unsigned int ToDPCM () const

  Converts DPI to dot per centimeter.

Static Public Member Functions

- static ImageDpi FromDPCM (unsigned int dpcm)

  Converts dot per centimeter to dot per inch.

4.85.1 Constructor & Destructor Documentation

4.85.1.1 ImageDpi()

```
ImageDpi (
            unsigned int i)  [inline], [explicit], [constexpr]
```

Parameters

i	dpi

4.85.2 Member Function Documentation

4.85.2.1 FromDPCM()

```
ImageDpi FromDPCM (
            unsigned int dpcm)  [inline], [static], [nodiscard]
```

Parameters

dpcm	value of dot per centimeter

Returns

ImageDpi

4.85.2.2 ToDPCM()

```
unsigned int ToDPCM () const  [inline], [nodiscard]
```

Returns

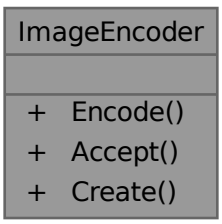
dot per centimeter

4.86 ImageEncoder Class Reference

Interface for image formatting converting to ImageType.

`#include <ImageEncoder.h>`

Collaboration diagram for ImageEncoder:



Public Member Functions

- virtual std::vector< uint8_t > Encode (const RawImage &raw) const =0

  Encodes given RawImage as encoded image.
- virtual void Accept (ImageEncoderVisitor &visitor) const =0

  Allow visitor to visit concrete type of encoder.

Static Public Member Functions

- static std::shared_ptr< ImageEncoder > Create (ImageType type)

  Create concrete default encoder for given type.

4.86.1 Detailed Description

It is used to extend library with user specific image types.

4.86.2 Member Function Documentation

4.86.2.1 Accept()

```
virtual void Accept (
            ImageEncoderVisitor & visitor) const  [pure virtual]
```

Parameters

visitor	ImageEncoderVisitor to call

Implemented in BmpImageEncoder, IRawImageEncoder, Jp2ImageEncoder, JpgImageEncoder, PngImageEncoder, TiffImageEncoder, WebpImageEncoder, and WsqImageEncoder.

4.86.2.2 Create()

```
std::shared_ptr< ImageEncoder > Create (
            ImageType type)  [inline], [static]
```

Parameters

type	ImageType

Exceptions

EnrollmentException	when image type is unknown

Returns

ImageEncoder

4.86.2.3 Encode()

```
virtual std::vector< uint8_t > Encode (
            const RawImage & raw) const  [nodiscard], [pure virtual]
```

Parameters

raw	RawImage

Returns

encoded image

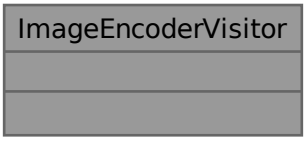
Implemented in IRawImageEncoder, and SupportedImageEncoder.

4.87 ImageEncoderVisitor Class Reference

The class is used to visit concrete encoder to get concrete data.

```
#include <ImageEncoder.h>
```

Collaboration diagram for ImageEncoderVisitor:

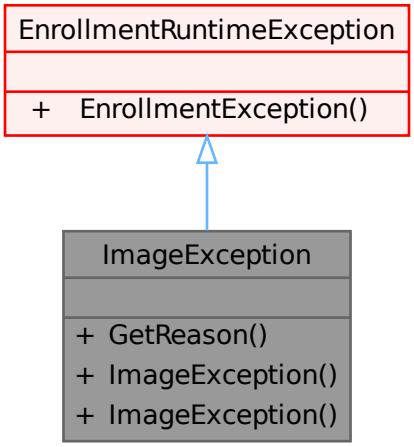


4.88 ImageException Class Reference

Exception for image processing.

```
#include <ImageException.h>
```

Collaboration diagram for ImageException:



Public Types

- enum Reason : uint8_t { UNKNOWN_IMAGE , DECODE_IMAGE , ENCODE_IMAGE , PROCESS_IMAGE }

  Contains all possible reasons of image exception.

Public Member Functions

- Reason GetReason () const

  ImageException reason.
- ImageException (Reason r, SourceLocation loc=SourceLocation())

  Constructor creates exception with given reason and information about source location.
- ImageException (Reason r, std::string d, SourceLocation loc=SourceLocation())

  Constructor creates exception with given reason and more specific message.

Public Member Functions inherited from EnrollmentRuntimeException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

  Provides message with full info about exception.

4.88.1 Member Enumeration Documentation

4.88.1.1 Reason

```
enum Reason : uint8_t
```

Enumerator

UNKNOWN_IMAGE	Image type is not supported. For supported image types **See also** ImageType
DECODE_IMAGE	Image type is supported but image format is corrupted.
ENCODE_IMAGE	Problem encode image to given format.
PROCESS_IMAGE	Problem to process an image.

4.88.2 Constructor & Destructor Documentation

4.88.2.1 ImageException() [1/2]

```
ImageException (
            Reason r,
            SourceLocation loc = SourceLocation())  [inline], [explicit]
```

Parameters

r	Reason of exception
loc	(optional) Source location of where the exception was thrown, providing file name, function name, and line information. Defaults to a default-constructed SourceLocation instance if not provided.

4.88.2.2 ImageException() [2/2]

```
ImageException (
            Reason r,
            std::string d,
            SourceLocation loc = SourceLocation())  [inline]
```

Parameters

r	Reason of exception
d	Description
loc	Source location

4.88.3 Member Function Documentation

4.88.3.1 GetReason()

```
Reason GetReason () const  [inline], [nodiscard]
```

Returns

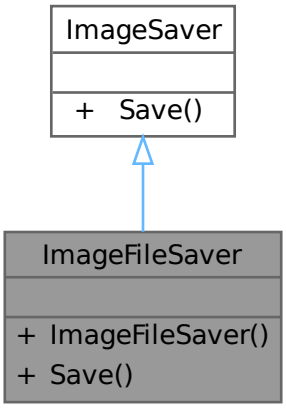
exception reason

4.89 ImageFileSaver Class Reference

Saves image to file.

```
#include <ImageEncoder.h>
```

Collaboration diagram for ImageFileSaver:



Public Member Functions

- ImageFileSaver (std::string path)

  Construct a new Image File Encoder object.
- void Save (const RawImage &raw) const final

  Saves given RawImage as encoded image.

4.89.1 Detailed Description

The image type is deduced from its extension.

4.89.2 Constructor & Destructor Documentation

4.89.2.1 ImageFileSaver()

```
ImageFileSaver (
            std::string path)  [inline], [explicit]
```

Parameters

path	file path

4.89.3 Member Function Documentation

4.89.3.1 Save()

```
void Save (
            const RawImage & raw) const  [inline], [final], [virtual]
```

It uses default encoder according to file extension.

Parameters

raw	RawImage

Exceptions

ImageException	when it is not possible to encode image
FileException	when it is not possible to write image file
EnrollmentException	when it is not possible to create encoder

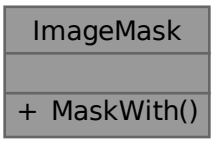
Implements ImageSaver.

4.90 ImageMask Class Reference

Defines the interface for applying mask to the image with a masker.

```
#include <ImageMask.h>
```

Collaboration diagram for ImageMask:



Public Member Functions

- virtual void MaskWith (ImageMasker &masker) const =0

  Applies a mask to the image using the provided ImageMasker.

4.90.1 Member Function Documentation

4.90.1.1 MaskWith()

```
virtual void MaskWith (
            ImageMasker & masker) const  [pure virtual]
```

Parameters

masker	image masker instance used to apply the mask.

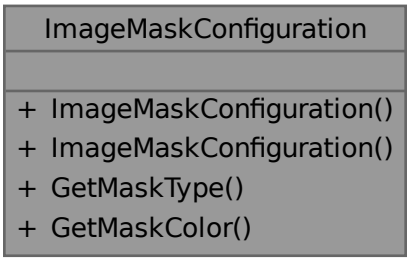
Implemented in CircleImageMask, and PolygonImageMask.

4.91 ImageMaskConfiguration Class Reference

Configuration parameters for image masking operations, including mask type, and color.

```
#include <ImageMask.h>
```

Collaboration diagram for ImageMaskConfiguration:



Public Member Functions

- ImageMaskConfiguration ()

  Constructs a default image mask configuration.
- ImageMaskConfiguration (ImageMaskType maskType, RGBA maskColor)

  Constructs a configuration with detailed mask settings including mask type and color.
- ImageMaskType GetMaskType () const

  Retrieves the mask type.
- RGBA GetMaskColor () const

  Retrieves the mask color.

4.91.1 Constructor & Destructor Documentation

4.91.1.1 ImageMaskConfiguration() [1/2]

```
ImageMaskConfiguration () [inline]
```

This default constructor initializes the mask configuration with default values. The default values are:

- Mask Type: OUTSIDE (applies the mask outside the specified area)
- Mask Color: White

4.91.1.2 ImageMaskConfiguration() [2/2]

```
ImageMaskConfiguration (
            ImageMaskType maskType,
            RGBA maskColor) [inline]
```

Allows for comprehensive customization of the mask's appearance and behavior.

Parameters

maskType	The masking approach defined by ImageMaskType.
maskColor	The color used for the mask, represented as RGBA.

4.91.2 Member Function Documentation

4.91.2.1 GetMaskColor()

```
RGBA GetMaskColor () const [inline], [nodiscard]
```

Returns

An RGBA object representing the color of the mask.

4.91.2.2 GetMaskType()

```
ImageMaskType GetMaskType () const [inline], [nodiscard]
```

Returns

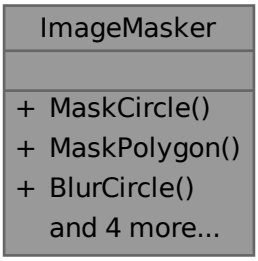
An ImageMaskType value representing the type of mask applied.

4.92 ImageMasker Class Reference

Interface for image masking operations.

```
#include <ImageMask.h>
```

Collaboration diagram for ImageMasker:



Public Member Functions

- virtual void MaskCircle (const Circle &mask, const class ImageMaskConfiguration &conf)=0

  Applies a circular image mask to an image.
- virtual void MaskPolygon (const Polygon &mask, const ImageMaskConfiguration &conf)=0

  Applies a specified polygonal mask to the image.
- virtual void BlurCircle (const Circle &circle, ImageMaskType maskType, ImageBlurType blurType)=0

  Applies a blur effect to a specified circular area in the image.
- virtual void BlurPolygon (const Polygon &polygon, ImageMaskType maskType, ImageBlurType blurType)=0

  Applies a blur effect to a specified polygonal area in the image.
- virtual void AddToBlur (const Circle &circle, ImageMaskType maskType)=0

  Adds a circular area for a future blur operation.
- virtual void AddToBlur (const Polygon &polygon, ImageMaskType maskType)=0

  Adds a polygonal area for a future blur operation.
- virtual void ApplyBlur (ImageBlurType blurType)=0

  Executes the blur effect on all added shapes with the specified blur type.

4.92.1 Member Function Documentation

4.92.1.1 AddToBlur() [1/2]

```
virtual void AddToBlur (
            const Circle & circle,
            ImageMaskType maskType)  [pure virtual]
```

Parameters

circle	The Circle object representing the area to be added to the blur queue.
maskType	The type of mask to apply the blurring on.

4.92.1.2 AddToBlur() [2/2]

```
virtual void AddToBlur (
            const Polygon & polygon,
            ImageMaskType maskType)  [pure virtual]
```

Parameters

polygon	The Polygon object representing the area to be added to the blur queue.
maskType	The type of mask to apply the blurring on.

4.92.1.3 ApplyBlur()

```
virtual void ApplyBlur (
            ImageBlurType blurType)  [pure virtual]
```

Parameters

blurType	The type of blur to apply to the queued shapes.

4.92.1.4 BlurCircle()

```
virtual void BlurCircle (
            const Circle & circle,
            ImageMaskType maskType,
            ImageBlurType blurType)  [pure virtual]
```

Parameters

circle	The Circle object defining the area to blur.
maskType	The type of mask to apply the blurring on.
blurType	The type of blur effect to apply.

4.92.1.5 BlurPolygon()

```
virtual void BlurPolygon (
            const Polygon & polygon,
            ImageMaskType maskType,
            ImageBlurType blurType)  [pure virtual]
```

Parameters

polygon	The Polygon object defining the area to blur.
maskType	The type of mask to apply the blurring on.
blurType	The type of blur effect to apply.

4.92.1.6 MaskCircle()

```
virtual void MaskCircle (
            const Circle & mask,
            const class ImageMaskConfiguration & conf)  [pure virtual]
```

Parameters

mask	The Circle object defining the mask shape.
conf	Configuration properties of the mask.

4.92.1.7 MaskPolygon()

```
virtual void MaskPolygon (
            const Polygon & mask,
            const ImageMaskConfiguration & conf)  [pure virtual]
```

Parameters

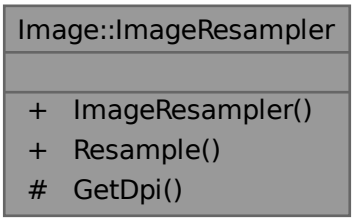
mask	The Polygon object defining the mask shape.
conf	Configuration properties of the mask.

4.93 Image::ImageResampler Class Reference

Image resampler for DPI-change operations, including upsampling and downsampling.

```
#include <Image.h>
```

Collaboration diagram for Image::ImageResampler:



Public Member Functions

- ImageResampler (ImageDpi targetDpi)

  Constructs an image resampler for a target DPI.
- virtual std::shared_ptr< Image > Resample (const inno::Image &image) const =0

  Resample image to a target DPI.

Protected Member Functions

- ImageDpi GetDpi () const

  Retrieves target DPI of the resampler.

4.93.1 Constructor & Destructor Documentation

4.93.1.1 ImageResampler()

```
ImageResampler (
            ImageDpi targetDpi)  [inline], [explicit]
```

This constructor initializes the image resampler to resample an image to the specified DPI.

Parameters

targetDpi	The DPI (dots per inch) of the new image.

Exceptions

EnrollmentInvalidArgumentException	when targetDpi is 0

4.93.2 Member Function Documentation

4.93.2.1 GetDpi()

```
ImageDpi GetDpi () const  [inline], [nodiscard], [protected]
```

Returns

The target DPI (dots per inch) of the resampler

4.93.2.2 Resample()

```
virtual std::shared_ptr< Image > Resample (
            const inno::Image & image) const  [nodiscard], [pure virtual]
```

It creates new image as result of resampling with factor target DPI/original DPI.

Parameters

image	Image to resize.

Returns

image with given DPI

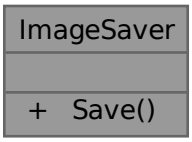
Implemented in Image::AreaImageResampler, Image::BicubicImageResampler, and Image::NISTImageResampler.

4.94 ImageSaver Class Reference

Interface for image saving.

```
#include <ImageEncoder.h>
```

Collaboration diagram for ImageSaver:



Public Member Functions

- virtual void Save (const RawImage &raw) const =0

  Encodes and saves given image.

4.94.1 Member Function Documentation

4.94.1.1 Save()

```
virtual void Save (
            const RawImage & raw) const  [pure virtual]
```

Parameters

raw	RawImage

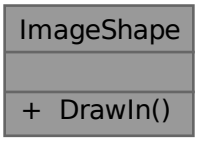
Implemented in ImageFileSaver.

4.95 ImageShape Class Reference

Any type of shape that wants to be drawn.

```
#include <ImageShape.h>
```

Collaboration diagram for ImageShape:



Public Member Functions

- virtual void DrawIn (ImageShapeDrawer &drawer) const =0

  It draws shape in ImageShapeDrawer.

4.95.1 Member Function Documentation

4.95.1.1 DrawIn()

```
virtual void DrawIn (
            ImageShapeDrawer & drawer) const  [pure virtual]
```

Parameters

drawer	the implementation of ImageShapeDrawer

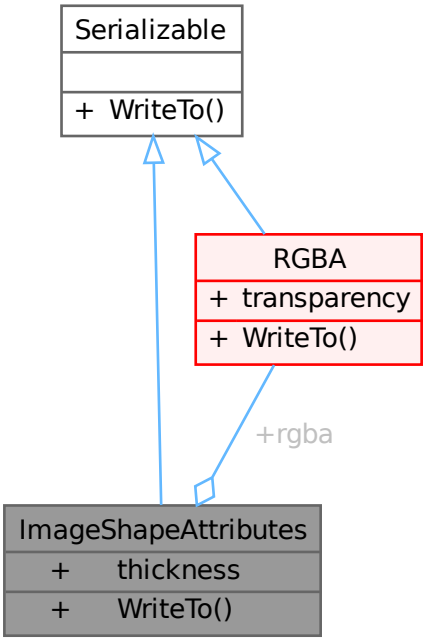
Implemented in CircleShape, CriticalPointShape, CriticalPointsShape, LineShape, MinutiaeShape, MinutiaShape, RectangleShape, and RectanglesShape.

4.96 ImageShapeAttributes Class Reference

It holds shape attributes such as color and line width.

```
#include <ImageShape.h>
```

Collaboration diagram for ImageShapeAttributes:



Public Member Functions

- void WriteTo (Writer &writer) const final

  Write itself into writer.

Public Attributes

- RGBA **rgba**

  RGBA color.
- unsigned char **thickness** = DefaultThickness

  Line thickness in pixels.

4.96.1 Member Function Documentation

4.96.1.1 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

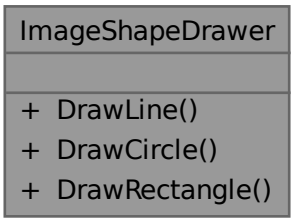
Implements Serializable.

4.97 ImageShapeDrawer Class Reference

It is used by ImageShape to draw its shape.

```
#include <ImageShape.h>
```

Collaboration diagram for ImageShapeDrawer:



Public Member Functions

- virtual void DrawLine (const Line &line, const ImageShapeAttributes &attr)=0

  Draws line.
- virtual void DrawCircle (const Circle &circle, const ImageShapeAttributes &attr)=0

  Draws circle.
- virtual void DrawRectangle (const Rectangle &rectangle, const ImageShapeAttributes &attr)=0

  Draws rectangle.

4.97.1 Member Function Documentation

4.97.1.1 DrawCircle()

```
virtual void DrawCircle (
            const Circle & circle,
            const ImageShapeAttributes & attr)  [pure virtual]
```

Parameters

circle	circle to draw
attr	ImageShapeAttributes of circle

4.97.1.2 DrawLine()

```
virtual void DrawLine (
            const Line & line,
            const ImageShapeAttributes & attr)  [pure virtual]
```

Parameters

line	line to draw
attr	ImageShapeAttributes of line

4.97.1.3 DrawRectangle()

```
virtual void DrawRectangle (
            const Rectangle & rectangle,
            const ImageShapeAttributes & attr)  [pure virtual]
```

Parameters

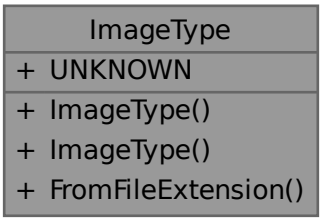
rectangle	Rectangle to draw
attr	ImageShapeAttributes of rectangle

4.98 ImageType Class Reference

It contains all supported image types.

```
#include <ImageType.h>
```

Collaboration diagram for ImageType:



Public Types

- enum Type : uint8_t {
  BMP , PNG , WSQ , IRAW ,
  JPG , TIFF , JPG2000 , WEBP }

  Type of supported images.

Public Member Functions

- constexpr ImageType (Type t)

  Construct ImageType from enum Type.
- constexpr ImageType (uint8_t t)

  Construct ImageType from basic type When value does not match enum value then UNKNOWN is set.

Static Public Member Functions

- static ImageType FromFileExtension (const std::string &extension)

  Construct ImageType from file extension.

Static Public Attributes

- static constexpr uint8_t **UNKNOWN** = 255

  Unknown image type.

4.98.1 Member Enumeration Documentation

4.98.1.1 Type

```
enum Type : uint8_t
```

Enumerator

BMP	Windows bitmaps - bmp, dib. Raster graphics image file format developed by Microsoft. [18]
PNG	Portable Network Graphics - png. PNG supports palette-based images (with palettes of 24-bit RGB or 32-bit RGBA colors), grayscale images (with or without an alpha channel for transparency), and full-color non-palette-based RGB or RGBA images. [14]
WSQ	Wavelet Scalar Quantization - wsq. WSQ (Wavelet Scalar Quantization), also known as FBI Print Format, is a raster image file format used for storing images of prints. It uses lossy wavelet compression, and supports only grayscale images.
IRAW	Innovatrics proprietary RAW image - iraw. The format is IRAW (4 bytes), width (4 bytes, big endian unsigned int), height (4 bytes, big endian unsigned int), dpi (4 bytes, big endian unsigned int), type (4 bytes, big endian unsigned int), data.
JPG	Joint Photographic Experts Group - jpeg, jpg, jpe. JPEG is a commonly used method of lossy compression for digital images, particularly for those images produced by digital photography. The degree of compression can be adjusted, allowing a selectable tradeoff between storage size and image quality. [13]
TIFF	Tagged Image File Format - tiff, tif. TIF is an image file format for storing raster graphics images, popular among graphic artists, the publishing industry, and photographers. [15]
JPG2000	JPEG 2000 is an image compression standard and coding system - jp2. JPEG 2000 (JP2) is an image compression standard and coding system. It was developed from 1997 to 2000 by a Joint Photographic Experts Group committee with the intention of superseding their original JPEG standard (created in 1992) [12]
WEBP	WebP is an image file format developed by Google - webp. It is intended as a replacement for JPEG, PNG, and GIF file formats. It supports both lossy and lossless compression as well as animation and alpha transparency. [16]

4.98.2 Constructor & Destructor Documentation

4.98.2.1 ImageType() [1/2]

```
ImageType (
            Type t) [inline], [explicit], [constexpr]
```

Parameters

t	enum Type value

4.98.2.2 ImageType() [2/2]

```
ImageType (
            uint8_t t) [inline], [explicit], [constexpr]
```

Parameters

t	enum value

Exceptions

ImageException	when type is unknown

4.98.3 Member Function Documentation

4.98.3.1 FromFileExtension()

```
ImageType FromFileExtension (
            const std::string & extension)  [inline], [static]
```

Exceptions

ImageException	when file extension is unknown

Parameters

extension	extension with '.' e.g. '.bmp'

Returns

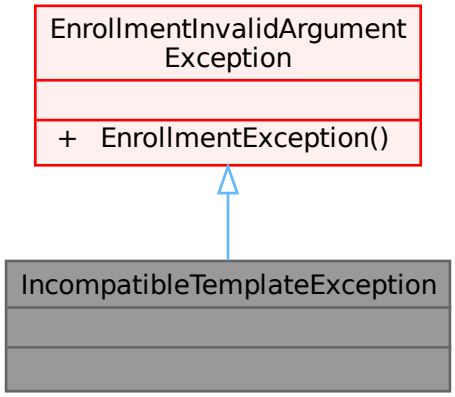
ImageType

4.99 IncompatibleTemplateException Class Reference

Template is incompatible in requested operation.

```
#include <EnrollmentException.h>
```

Inheritance diagram for IncompatibleTemplateException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentInvalidArgumentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char ∗ what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

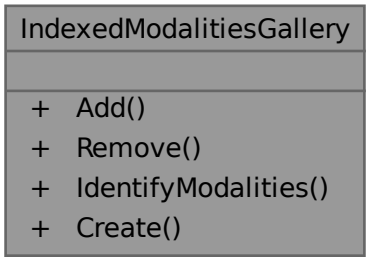
  Provides message with full info about exception.

4.100 IndexedModalitiesGallery Class Reference

Gallery of references identifiers.

```
#include <IndexedModalitiesGallery.h>
```

Collaboration diagram for IndexedModalitiesGallery:



Classes

- class ModalitiesIdentifiers

  Contains candidates per each modality.

Public Types

- using **Identifier** = size_t

  Index identifier of reference in gallery.

Public Member Functions

- virtual void Add (Identifier identifier, const std::shared_ptr< ModalitiesTemplate > &templ)

  Adds reference template under index identifier into gallery.
- virtual void Remove (Identifier identifier)

  Removes reference template under provided index identifier from the gallery.
- virtual std::shared_ptr< ModalitiesIdentifiers > IdentifyModalities (const std::shared_ptr< ModalitiesTemplate > &probe, size_t maxCandidates) const

  Search for given probe.

Static Public Member Functions

- static std::shared_ptr< IndexedModalitiesGallery > Create (IndexedModalitiesGalleryConfig config)

  Create indexed gallery of references.

4.100.1 Detailed Description

Note

The number of references in any IndexedModalitiesGallery is limited by license. You can have more created galleries in your code. The number limits overall references in all galleries.

The IndexedModalitiesGallery stores only the data necessary for identification. Relevant data from ModalitiesTemplate is copied into the gallery using the IndexedModalitiesGallery::Add() function.

The IndexedModalitiesGallery::IdentifyModalities() function returns reference index identifiers to the user.

Adding data to the gallery can be faster in higher-level language bindings and consumes less memory, as only the essential identification data is stored compared to ModalitiesGallery. However, the user must maintain a mapping between their reference representation and the identifier provided to IndexedModalitiesGallery::Add().

It is recommended to use IndexedModalitiesGallery for performance- and memory-critical use cases and to prefer it over the more convenient ModalitiesGallery.

c++

```
// Recreate your own data types from data stored during enrollment
auto templates =
  std::unordered_map<size_t, std::shared_ptr<ModalitiesTemplate»{};
templates.emplace(
  1, ModalitiesTemplate::Create(BinaryFile::ReadAll("output/1.tmpl")));
templates.emplace(
  2, ModalitiesTemplate::Create(BinaryFile::ReadAll("output/2.tmpl")));

// Create a gallery and add users.
auto gallery = IndexedModalitiesGallery::Create({});
gallery->Add(1, templates[1]);
gallery->Add(2, templates[2]);
auto probe = templates[1];

// Identify a probe in the gallery.
constexpr size_t MaxCandidates = 2; // size of result set of candidates
auto result = gallery->IdentifyModalities(probe, MaxCandidates);

// Output identification results.
for (const auto& id : result->GetPrintIdentifiers())
{
    std::cout « "   Found Id is " « id « std::endl;
}
```

java

```
// Recreate your own data types from data stored during enrollment
HashMap<Long, ModalitiesTemplate> templates
        = new HashMap<Long, ModalitiesTemplate>();
templates.put(1L, ModalitiesTemplate.Create(BinaryFile.ReadAll("output/1.tmpl")));
templates.put(2L, ModalitiesTemplate.Create(BinaryFile.ReadAll("output/2.tmpl")));
ModalitiesTemplate probe = templates.get(1L);

// Create a gallery and add users.
IndexedModalitiesGallery gallery = IndexedModalitiesGallery.Create(
        new IndexedModalitiesGalleryConfig());
gallery.Add(1, templates.get(1L));
gallery.Add(2, templates.get(2L));

// Identify a probe in the gallery.
long maxCandidates = 2; // size of result set of candidates
IndexedModalitiesGallery.ModalitiesIdentifiers result = gallery.IdentifyModalities(
        probe, maxCandidates);

// Output identification results.
for (long id : result.GetPrintIdentifiers()) {
    System.out.println("   Found Id is " + id);
}
```

csharp

```
// Recreate your own data types from data stored during enrollment
var templates = new Dictionary<uint, ModalitiesTemplate>();
templates.Add(1, ModalitiesTemplate.Create(BinaryFile.ReadAll("output/1.tmpl")));
templates.Add(2, ModalitiesTemplate.Create(BinaryFile.ReadAll("output/2.tmpl")));
```

```
var probe = templates[1];

// Create a gallery and add users.
var gallery =
    IndexedModalitiesGallery.Create(new IndexedModalitiesGalleryConfig());
gallery.Add(1, templates[1]);
gallery.Add(2, templates[2]);

// Identify a probe in the gallery.
var maxCandidates = 2U; // size of result set of candidates
var result = gallery.IdentifyModalities(probe, maxCandidates);

// Output identification results.
foreach (uint id in result.GetPrintIdentifiers())
{
    System.Console.WriteLine("   Found Id is " + id);
}
```

Once the candidates are identified, you can use ModalitiesGallery::Consolidate() static function to get unique list of candidates with scores.

c++

```
// Create print candidates for consolidation
auto printCandidates = std::vector<std::shared_ptr<ModalitiesGalleryReference»{};
for (const auto& id : result->GetPrintIdentifiers())
{
    printCandidates.push_back(
      std::make_shared<ModalitiesGalleryIndexedTemplate>(id, templates[id]));
}
// Create type that holds all necessary information for consolidation.
auto consolidationCandidates =
  std::make_shared<ModalitiesGallery::ModalitiesCandidates>(
    std::move(printCandidates),
    std::vector<std::shared_ptr<ModalitiesGalleryReference»{},
    std::vector<std::shared_ptr<ModalitiesGalleryReference»{},
    probe);
// Consolidate
auto candidates = ModalitiesGallery::Consolidate(consolidationCandidates, 2);
// Output identification results.
for (const auto& c : candidates)
{
    std::cout « "   Id is " « c.GetCandidate()->GetIndex() « " score is ["
              « c.GetScore().GetPrintScore() « ", "
              « c.GetScore().GetFaceScore() « ", "
              « c.GetScore().GetIrisScore() « "]" « std::endl;
}
```

java

```
// Create print candidates for consolidation
ArrayList<ModalitiesGalleryReference> printCandidates
        = new ArrayList<ModalitiesGalleryReference>();
for (long id : result.GetPrintIdentifiers()) {
    printCandidates.add(
            new ModalitiesGalleryIndexedTemplate(id, templates.get(id)));
}
// Create type that holds all necessary information for consolidation.
ModalitiesGallery.ModalitiesCandidates consolidationCandidates
        = new ModalitiesGallery.ModalitiesCandidates(printCandidates,
                new ArrayList<ModalitiesGalleryReference>(),
                new ArrayList<ModalitiesGalleryReference>(), probe);
// Consolidate
ArrayList<ModalitiesGallery.Candidate> candidates = ModalitiesGallery.Consolidate(
        consolidationCandidates, maxCandidates);
// Print result
for (ModalitiesGallery.Candidate c : candidates) {
    System.out.println("   Id is " + c.GetCandidate().GetIndex() + " score is ["
                       + c.GetScore().GetPrintScore() + ", "
                       + c.GetScore().GetFaceScore() + ", "
                       + c.GetScore().GetIrisScore() + "]");
}
```

csharp

```
// Create print candidates for consolidation
var printCandidates = new List<ModalitiesGalleryReference>();
foreach (uint id in result.GetPrintIdentifiers())
{
    printCandidates.Add(new ModalitiesGalleryIndexedTemplate(id, templates[id]));
}
// Create type that holds all necessary information for consolidation.
var consolidationCandidates = new ModalitiesGallery.ModalitiesCandidates(
    printCandidates, new List<ModalitiesGalleryReference>(),
```

```
    new List<ModalitiesGalleryReference>(), probe);
// Consolidate
var candidates = ModalitiesGallery.Consolidate(consolidationCandidates, 2);
// Print result
foreach (ModalitiesGallery.Candidate c in candidates)
{
    System.Console.WriteLine("   Id is " + c.GetCandidate().GetIndex() +
                             " score is [" + c.GetScore().GetPrintScore() +
                             ", " + c.GetScore().GetFaceScore() + ", " +
                             c.GetScore().GetIrisScore() + "]");
}
```

4.100.2 Member Function Documentation

4.100.2.1 Add()

```
virtual void Add (
            Identifier identifier,
            const std::shared_ptr< ModalitiesTemplate > & templ)  [inline], [virtual]
```

It copies necessary data for identification into gallery. It is not necessary to keep given `templ`.

Parameters

identifier	reference identifier
templ	reference ModalitiesTemplate

Exceptions

EnrollmentException	when it is not possible to add reference
NullArgumentException	when templ is null
EnrollmentInvalidArgumentException	when at least one probe print has position different as PrintPosition::LEFT_INDEX, PrintPosition::LEFT_LITTLE, PrintPosition::LEFT_MIDDLE, PrintPosition::LEFT_THUMB, PrintPosition::LEFT_RING, PrintPosition::RIGHT_INDEX, PrintPosition::RIGHT_LITTLE, PrintPosition::RIGHT_MIDDLE, PrintPosition::RIGHT_THUMB, PrintPosition::RIGHT_RING or no position.
NullArgumentException	when some print in probe prints is null.
NullArgumentException	when some face in probe faces is null.
NullArgumentException	when some iris in probe irises is null.

4.100.2.2 Create()

```
std::shared_ptr< IndexedModalitiesGallery > Create (
            IndexedModalitiesGalleryConfig config)  [inline], [static]
```

Parameters

config	Config of gallery

Returns

created gallery

Exceptions

EnrollmentException	when it is not possible to create gallery

4.100.2.3 IdentifyModalities()

```
virtual std::shared_ptr< ModalitiesIdentifiers > IdentifyModalities (
            const std::shared_ptr< ModalitiesTemplate > & probe,
            size_t maxCandidates) const  [inline], [nodiscard], [virtual]
```

Parameters

probe	probe that is searched in gallery
maxCandidates	maximum number of returned identified candidates

Returns

identified candidates per modalities without score. To get scores and unique candidates list use function ModalitiesGallery::Consolidate().

Exceptions

EnrollmentException	when it is not possible to identify probe
NullArgumentException	when probe is null
NullArgumentException	when some print in probe prints is null.
NullArgumentException	when some face in probe faces is null.
NullArgumentException	when some iris in probe irises is null.

4.100.2.4 Remove()

```
virtual void Remove (
            Identifier identifier)  [inline], [virtual]
```

Parameters

identifier	reference identifier

Exceptions

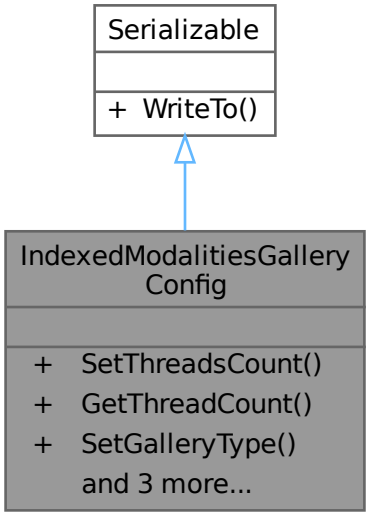
EnrollmentException	when the provided identifier is not in the gallery

4.101 IndexedModalitiesGalleryConfig Class Reference

IndexedModalitiesGallery configuration.

```
#include <IndexedModalitiesGallery.h>
```

Collaboration diagram for IndexedModalitiesGalleryConfig:



Public Member Functions

- void SetThreadsCount (unsigned int value)

  Set number of threads used by gallery.
- unsigned int GetThreadCount () const

  Get number of processing threads.
- void SetGalleryType (IndexedModalitiesGalleryType value)

  Set type of Modalities Gallery.
- IndexedModalitiesGalleryType GetGalleryType () const

  Get type of Modalities Gallery.
- void WriteTo (Writer &writer) const final

  Function serializes configuration via provided Writer.
- IndexedModalitiesGalleryConfig ()=default

  Default configuration is.

4.101.1 Constructor & Destructor Documentation

4.101.1.1 IndexedModalitiesGalleryConfig()

```
IndexedModalitiesGalleryConfig () [default]
```

- identification threads - 1 thread
- gallery type - PRINT_ICS_TEMPLATE

4.101.2 Member Function Documentation

4.101.2.1 GetGalleryType()

```
IndexedModalitiesGalleryType GetGalleryType () const  [inline], [nodiscard]
```

Default type is PRINT_ICS_TEMPLATE.

Returns

type of gallery

4.101.2.2 GetThreadCount()

```
unsigned int GetThreadCount () const  [inline], [nodiscard]
```

Returns

number of threads

4.101.2.3 SetGalleryType()

```
void SetGalleryType (
            IndexedModalitiesGalleryType value)  [inline]
```

Default type is PRINT_ICS_TEMPLATE.

Parameters

value	from the enum IndexedModalitiesGalleryType

Exceptions

EnrollmentInvalidArgumentException	when value is not in range of the enum IndexedModalitiesGalleryType

4.101.2.4 SetThreadsCount()

```
void SetThreadsCount (
            unsigned int value)  [inline]
```

Note

The 'multithreaded-galleries' feature MUST be allowed in license to allow using threads. Default value is 1.

Parameters

value	it MUST be greater as 0

Exceptions

EnrollmentInvalidArgumentException	when value is 0

4.101.2.5 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Config.

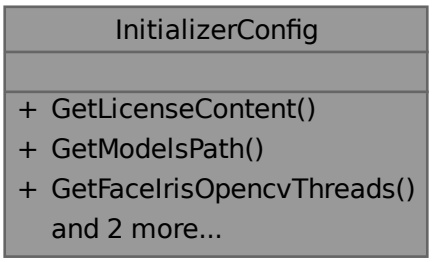
Implements Serializable.

4.102 InitializerConfig Struct Reference

SDK initialisation configuration for manual init.

```
#include <InitializerConfig.h>
```

Collaboration diagram for InitializerConfig:



Public Member Functions

- std::vector< uint8_t > GetLicenseContent () const

  Returns license content.
- std::string GetModelsPath () const

  Returns the IFace models directory path.
- int GetFaceIrisOpencvThreads () const

  Returns the configured OpenCV thread count for face and iris processing.
- int GetFaceIrisOrtIntraThreads () const

  Returns the configured ONNX Runtime intra-op thread count for face and iris processing.
- int GetFaceIrisOrtInterThreads () const

  Returns the configured ONNX Runtime inter-op thread count for face and iris processing.

Friends

- class **InitializerConfigBuilder**

4.102.1 Detailed Description

Built via InitializerConfigBuilder. Thread pools must be set explicitly in the builder (InitializerConfigBuilder::UseAllAvailableCores(), InitializerConfigBuilder::UseSingleCore(), or the `SetFace↩Iris*Threads` setters). Zero in a thread field means use all available cores for that pool.

Set license data via the builder. Models path may be left unset to use DefaultModelsDirectory().

4.102.2 Member Function Documentation

4.102.2.1 GetFaceIrisOpencvThreads()

```
int GetFaceIrisOpencvThreads () const  [inline], [nodiscard]
```

Returns

thread count, or 0 to use all available cores

4.102.2.2 GetFaceIrisOrtInterThreads()

```
int GetFaceIrisOrtInterThreads () const  [inline], [nodiscard]
```

Returns

thread count, or 0 to use all available cores

4.102.2.3 GetFaceIrisOrtIntraThreads()

```
int GetFaceIrisOrtIntraThreads () const  [inline], [nodiscard]
```

Returns

thread count, or `0` to use all available cores

4.102.2.4 GetLicenseContent()

```
std::vector< uint8_t > GetLicenseContent () const  [inline], [nodiscard]
```

Returns

license file content, or an empty vector when not set in the builder

4.102.2.5 GetModelsPath()

```
std::string GetModelsPath () const  [inline], [nodiscard]
```

Returns

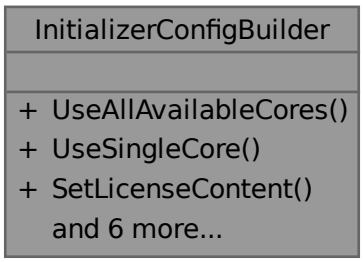
path to models

4.103 InitializerConfigBuilder Class Reference

Fluent builder for InitializerConfig.

```
#include <InitializerConfig.h>
```

Collaboration diagram for InitializerConfigBuilder:



Public Member Functions

- InitializerConfigBuilder & UseAllAvailableCores ()

  Use all available CPU cores for each face/iris thread pool.
- InitializerConfigBuilder & UseSingleCore ()

  Limit each face/iris thread pool to a single thread.
- InitializerConfigBuilder & SetLicenseContent (std::vector< uint8_t > content)

  Sets embedded license bytes.
- InitializerConfigBuilder & SetLicensePath (const std::string &path)

  Sets license content from given license file.
- InitializerConfigBuilder & SetModelsPath (std::string path)

  Sets the IFace models directory path.
- InitializerConfigBuilder & SetFaceIrisOpencvThreads (int count)

  Sets the OpenCV thread count for face and iris processing.
- InitializerConfigBuilder & SetFaceIrisOrtIntraThreads (int count)

  Sets the ONNX Runtime intra-op thread count for face and iris processing.
- InitializerConfigBuilder & SetFaceIrisOrtInterThreads (int count)

  Sets the ONNX Runtime inter-op thread count for face and iris processing.
- InitializerConfig Build ()

  Builds and returns the configuration.

4.103.1 Detailed Description

InitializerConfigBuilder::Build() fails unless every thread pool was configured via InitializerConfigBuilder::UseAllAvailableCores(), InitializerConfigBuilder::UseSingleCore(), or the three `Set↩FaceIris*Threads` methods.

4.103.2 Member Function Documentation

4.103.2.1 Build()

```
InitializerConfig Build () [inline], [nodiscard]
```

Returns

validated initialisation configuration

Exceptions

EnrollmentInvalidArgumentException	when license content is not set
EnrollmentInvalidArgumentException	when model path is not set
EnrollmentInvalidArgumentException	when not all face/iris thread pools were configured
EnrollmentInvalidArgumentException	when a thread count is negative
FileException	if given license file is not accessible

4.103.2.2 SetFaceIrisOpencvThreads()

```
InitializerConfigBuilder & SetFaceIrisOpencvThreads (
            int count) [inline]
```

Parameters

count	thread count (0 means all available cores)

Returns

reference to this builder

4.103.2.3 SetFaceIrisOrtInterThreads()

```
InitializerConfigBuilder & SetFaceIrisOrtInterThreads (
            int count) [inline]
```

Parameters

count	thread count (0 means all available cores)

Returns

reference to this builder

4.103.2.4 SetFaceIrisOrtIntraThreads()

```
InitializerConfigBuilder & SetFaceIrisOrtIntraThreads (
            int count) [inline]
```

Parameters

count	thread count (0 means all available cores)

Returns

reference to this builder

4.103.2.5 SetLicenseContent()

```
InitializerConfigBuilder & SetLicenseContent (
            std::vector< uint8_t > content)  [inline]
```

Parameters

content	raw license file content

Returns

reference to this builder

4.103.2.6 SetLicensePath()

```
InitializerConfigBuilder & SetLicensePath (
            const std::string & path)  [inline]
```

Parameters

path	Path to license file

Returns

reference to this builder

4.103.2.7 SetModelsPath()

```
InitializerConfigBuilder & SetModelsPath (
            std::string path)  [inline]
```

Parameters

path	path to the models directory

Returns

reference to this builder

4.103.2.8 UseAllAvailableCores()

```
InitializerConfigBuilder & UseAllAvailableCores ()  [inline]
```

Returns

reference to this builder

4.103.2.9 UseSingleCore()

```
InitializerConfigBuilder & UseSingleCore () [inline]
```

Returns

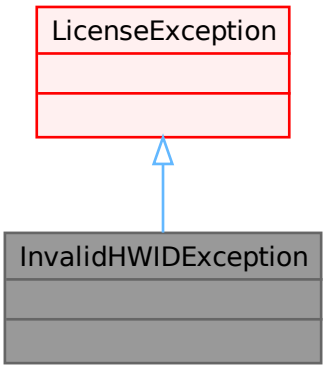
reference to this builder

4.104 InvalidHWIDException Class Reference

HWID in license is wrong or can not be read.

```
#include <LicenseException.h>
```

Inheritance diagram for InvalidHWIDException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentRuntimeException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

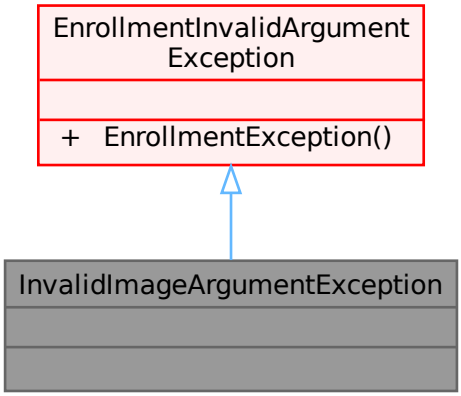
  Provides message with full info about exception.

4.105 InvalidImageArgumentException Class Reference

Image has wrong format or dimensions or dpi and is not suitable for processing.

```
#include <EnrollmentException.h>
```

Inheritance diagram for InvalidImageArgumentException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentInvalidArgumentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

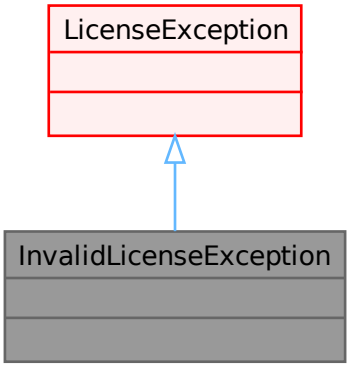
  Provides message with full info about exception.

4.106 InvalidLicenseException Class Reference

License is found but is invalid.

```
#include <LicenseException.h>
```

Inheritance diagram for InvalidLicenseException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentRuntimeException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

  Provides message with full info about exception.

4.106.1 Detailed Description

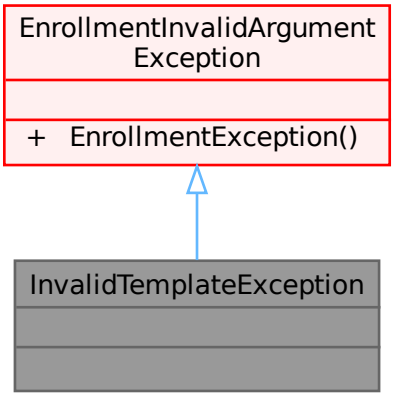
It happens when license content does not match hash in file.

4.107 InvalidTemplateException Class Reference

The template is empty or or do not contain enough distinctive data or is corrupted.

```
#include <EnrollmentException.h>
```

Inheritance diagram for InvalidTemplateException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentInvalidArgumentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

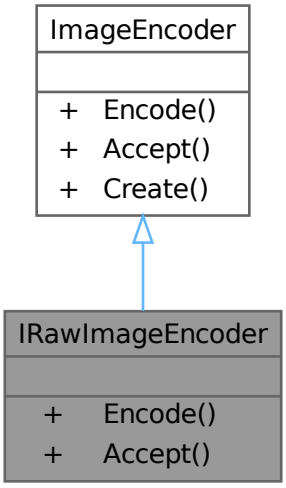
  Provides message with full info about exception.

4.108 IRawImageEncoder Class Reference

Saves RawImage image to memory.

```
#include <ImageEncoder.h>
```

Collaboration diagram for IRawImageEncoder:



Public Member Functions

- std::vector< uint8_t > Encode (const RawImage &raw) const final

  Saves given RawImage in innovatrics raw image format.
- void Accept (ImageEncoderVisitor &visitor) const final

  Allow visitor to visit concrete type of encoder.

Additional Inherited Members

Static Public Member Functions inherited from ImageEncoder

- static std::shared_ptr< ImageEncoder > Create (ImageType type)

  Create concrete default encoder for given type.

4.108.1 Member Function Documentation

4.108.1.1 Accept()

```
void Accept (
            ImageEncoderVisitor & visitor) const  [inline], [final], [virtual]
```

Parameters

visitor	ImageEncoderVisitor to call

Implements ImageEncoder.

4.108.1.2 Encode()

```
std::vector< uint8_t > Encode (
            const RawImage & raw) const  [inline], [nodiscard], [final], [virtual]
```

Parameters

raw	RawImage

Returns

encoded image

Exceptions

InvalidImageArgumentException	when image is empty

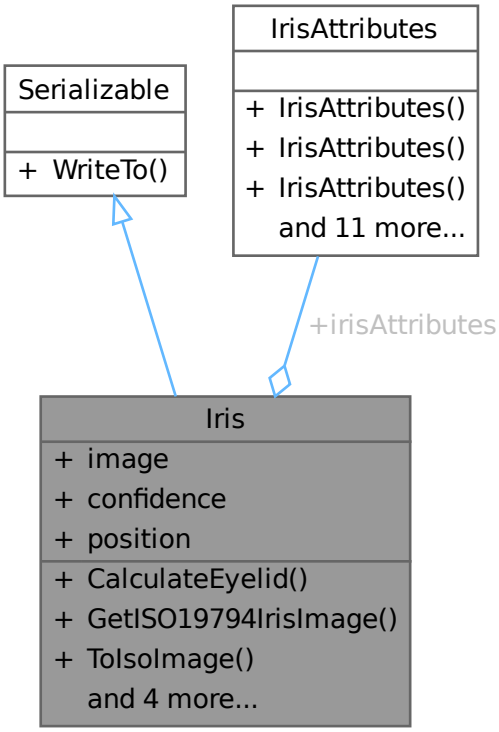
Implements ImageEncoder.

4.109 Iris Class Reference

It holds all functions and data related to iris manipulation.

```
#include <Iris.h>
```

Collaboration diagram for Iris:



Public Member Functions

- Polygon CalculateEyelid () const

  Provides polygon of lower and upper eyelid.
- std::shared_ptr< Image > GetISO19794IrisImage () const

  Provides a cropped and masked iris image compliant with ISO/IEC 19794-6:2011 standards.
- std::vector< uint8_t > ToIsoImage (ImageEncoder &encoder) const

  Function creates ISO image based on ISO/IEC 19794-6 (Iris image data).
- std::vector< uint8_t > ToIsoRawImage () const

  Function creates ISO image based on ISO/IEC 19794-6 (Iris image data) use RawImage.
- std::shared_ptr< ExtractedIris > Extract ()

  Function creates template for given image with default IrisExtractorConfig.
- std::shared_ptr< ExtractedIris > Extract (const IrisExtractorConfig &cfg)

  Function creates template for given image.
- void WriteTo (Writer &writer) const final

  Function serializes Iris via provided Writer.

Public Attributes

- const std::shared_ptr< Image > **image**

  Provides Iris Image.
- const float confidence

  Provides confidence score of the iris related to iris detection.
- const IrisPosition **position**

  Provides IrisPosition.
- const IrisAttributes **irisAttributes**

  Provides IrisAttributes.

Friends

- class **IrisDetector**

4.109.1 Member Function Documentation

4.109.1.1 CalculateEyelid()

```
Polygon CalculateEyelid () const  [inline]
```

Returns

Polygon eyelids polygon.

4.109.1.2 Extract() [1/2]

```
std::shared_ptr< ExtractedIris > Extract ()  [inline]
```

Returns

template calculated template

Exceptions

EnrollmentException	when it is not possible to extract templates

Required Solvers

See IrisExtractorConfig::Type

4.109.1.3 Extract() [2/2]

```
std::shared_ptr< ExtractedIris > Extract (
            const IrisExtractorConfig & cfg)  [inline]
```

Parameters

cfg	IrisExtractorConfig

Returns

template calculated template

Exceptions

EnrollmentException	when it is not possible to extract templates

Required Solvers

See IrisExtractorConfig::Type

4.109.1.4 GetISO19794IrisImage()

```
std::shared_ptr< Image > GetISO19794IrisImage () const  [inline]
```

This function generates a cropped rectilinear iris image with applied masks to the upper and lower eyelids, and the sclera, enhancing image compressibility. The masking operation assigns a single grey value to these regions: 128 for eyelids and 200 for sclera, facilitating compression. Additionally, it incorporates mask transition blurring to minimize boundary impacts on compression, employing a low-pass filter to smooth transitions. This process aligns with the ISO standard's guidance for creating highly compressible iris images while maintaining the normative requirements for image compression specified in the standard.

Reference: ISO/IEC 19794-6:2011(E), Sections 6.5.1 - 6.5.4, Cropped and Masked Iris Image specifications [7] .

Returns

A masked iris image, prepared according to ISO/IEC standards.

Exceptions

EnrollmentInvalidArgumentException	when iris is not fully contained within the image

4.109.1.5 ToIsoImage()

```
std::vector< uint8_t > ToIsoImage (
            ImageEncoder & encoder) const  [inline]
```

ISO standard defines a standard for exchange of iris image information. It contains a specific definition of attributes, a data record format for storing and transmitting the iris image and certain attributes, and conformance criteria

Parameters

encoder	specific encoder for encoding Print image. Supported JPG (lossy), JPEG 2000 (lossy & lossless), PNG

Returns

ISO image from ISO/IEC 19794-6 (Iris image data)

Exceptions

ImageException	when it is not possible to encode image
EnrollmentInvalidArgumentException	when encoder is not supported
EnrollmentRuntimeException	for invalid conversion size

4.109.1.6 ToIsoRawImage()

```
std::vector< uint8_t > ToIsoRawImage () const  [inline], [nodiscard]
```

Returns

ISO raw image from ISO/IEC 19794-6 (Iris image data)

Exceptions

EnrollmentRuntimeException	for invalid conversion size

4.109.1.7 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Iris.

Implements Serializable.

4.109.2 Member Data Documentation

4.109.2.1 confidence

```
const float confidence
```

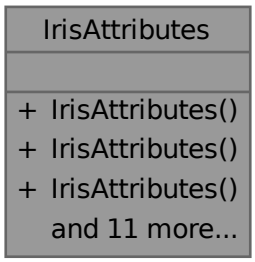
Confidence values are from range <0,1>. The higher the value of the attribute the better quality of the iris. The decision threshold is around 0.6.

4.110 IrisAttributes Class Reference

Iris attributes Holds the iris attributes.

```
#include <IrisAttributes.h>
```

Collaboration diagram for IrisAttributes:



Public Member Functions

- **IrisAttributes** ()=default

  Default constructor of iris attributes.
- IrisAttributes (const IrisAttributes &ia)

  Copy constructor of iris attributes.
- IrisAttributes (IrisAttributes &&ia) noexcept

  Move constructor of iris attributes.
- IrisAttributes & operator= (const IrisAttributes &ia)

  Assign the iris attributes.
- IrisAttributes & operator= (IrisAttributes &&ia) noexcept

  Assign the iris attributes.
- constexpr bool operator== (const IrisAttributes &ia) const

  Compares the iris attributes.
- void SetCaptureDeviceTechnology (IrisCaptureDeviceTechnology technology)

  Sets capture device technology.
- IrisCaptureDeviceTechnology GetCaptureDeviceTechnology () const

  Provides capture device technology.

- void SetCaptureDeviceVendorID (IrisCaptureDeviceVendorID vendor)

  Sets capture device vendor ID.
- IrisCaptureDeviceVendorID GetCaptureDeviceVendorID () const

  Provides capture device vendor ID.
- void SetCaptureDeviceTypeID (IrisCaptureDeviceTypeID type)

  Sets capture device type ID.
- IrisCaptureDeviceTypeID GetCaptureDeviceTypeID () const

  Provides capture device type ID.
- IrisImageType GetImageType () const

  Provides capture image type.
- void SetImageType (IrisImageType type)

  Set capture image type.

4.110.1 Detailed Description

Enables to set each iris attribute separately. In case, the attributes are not set the default values are used:

- capture device technology = IrisCaptureDeviceTechnology::UNKNOWN
- capture device vendor ID = 0
- capture device type ID = 0
- iris image type = IrisImageType::UNCROPPED

4.110.2 Constructor & Destructor Documentation

4.110.2.1 IrisAttributes() [1/2]

```
IrisAttributes (
            const IrisAttributes & ia)  [inline]
```

Parameters

ia	Iris attributes

4.110.2.2 IrisAttributes() [2/2]

```
IrisAttributes (
            IrisAttributes && ia)  [inline], [noexcept]
```

Parameters

ia	Iris attributes

4.110.3 Member Function Documentation

4.110.3.1 GetCaptureDeviceTechnology()

```
IrisCaptureDeviceTechnology GetCaptureDeviceTechnology () const  [inline], [nodiscard]
```

Returns

Capture device technology

4.110.3.2 GetCaptureDeviceTypeID()

```
IrisCaptureDeviceTypeID GetCaptureDeviceTypeID () const  [inline], [nodiscard]
```

Returns

Capture device type ID

4.110.3.3 GetCaptureDeviceVendorID()

```
IrisCaptureDeviceVendorID GetCaptureDeviceVendorID () const  [inline], [nodiscard]
```

Returns

Capture device vendor ID

4.110.3.4 GetImageType()

```
IrisImageType GetImageType () const  [inline], [nodiscard]
```

Returns

IrisImageType

4.110.3.5 operator=() [1/2]

```
IrisAttributes & operator= (
            const IrisAttributes & ia)  [inline]
```

Parameters

ia	Iris attributes

Returns

this modified object

4.110.3.6 operator=() [2/2]

```
IrisAttributes & operator= (
            IrisAttributes && ia)  [inline], [noexcept]
```

Parameters

ia	Iris attributes

Returns

this modified object

4.110.3.7 operator==()

```
bool operator== (
            const IrisAttributes & ia) const  [inline], [constexpr]
```

Parameters

ia	Iris attributes

Returns

`true` when given and this object is same

`false` otherwise

4.110.3.8 SetCaptureDeviceTechnology()

```
void SetCaptureDeviceTechnology (
            IrisCaptureDeviceTechnology technology)  [inline]
```

Parameters

technology	Capture device technology

4.110.3.9 SetCaptureDeviceTypeID()

```
void SetCaptureDeviceTypeID (
            IrisCaptureDeviceTypeID type)  [inline]
```

Parameters

type	Capture device type ID

4.110.3.10 SetCaptureDeviceVendorID()

```
void SetCaptureDeviceVendorID (
            IrisCaptureDeviceVendorID vendor)  [inline]
```

Parameters

vendor	Capture device vendor ID

4.110.3.11 SetImageType()

```
void SetImageType (
            IrisImageType type)  [inline]
```

Parameters

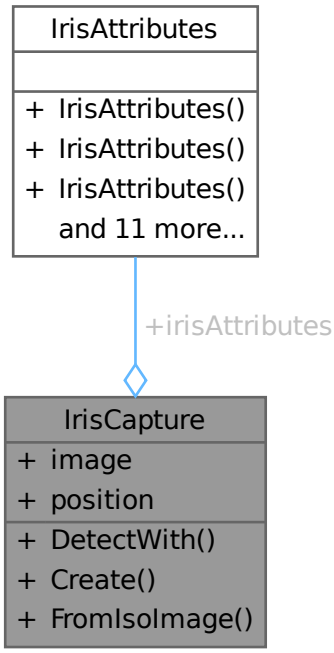
type	IrisImageType to set

4.111 IrisCapture Class Reference

Iris capture contains one iris.

```
#include <IrisCapture.h>
```

Collaboration diagram for IrisCapture:



Public Member Functions

- std::shared_ptr< IrisSegment > DetectWith (IrisDetector &detector) const

  It detects iris.

Static Public Member Functions

- static std::shared_ptr< IrisCapture > Create (std::shared_ptr< Image > img, const IrisPosition &pos, const IrisAttributes &ia=IrisAttributes())

  Constructor of IrisCapture.
- static std::shared_ptr< IrisCapture > FromIsoImage (const std::vector< uint8_t > &isoImage)

  Function creates shared pointer of the Iris object from ISO image based on ISO/IEC 19794-6 (Iris image data).

Public Attributes

- const std::shared_ptr< Image > **image**

  Provides IrisCapture Image.
- const IrisPosition **position**

  Provides IrisPosition.
- const IrisAttributes **irisAttributes**

  Provides IrisAttributes.

4.111.1 Member Function Documentation

4.111.1.1 Create()

```
std::shared_ptr< IrisCapture > Create (
            std::shared_ptr< Image > img,
            const IrisPosition & pos,
            const IrisAttributes & ia = IrisAttributes())  [inline], [static]
```

Parameters

img	iris capture image, will be moved into this class
pos	iris position
ia	IrisAttributes

Returns

created IrisCapture object

Exceptions

NullArgumentException	when image is null

4.111.1.2 DetectWith()

```
std::shared_ptr< IrisSegment > DetectWith (
            IrisDetector & detector) const  [inline], [nodiscard]
```

Parameters

detector	is IrisDetector

Returns

detected IrisSegment

Exceptions

EnrollmentException	when Detector throws

4.111.1.3 FromIsoImage()

```
std::shared_ptr< IrisCapture > FromIsoImage (
            const std::vector< uint8_t > & isoImage)  [inline], [static], [nodiscard]
```

[7]

Parameters

isoImage	ISO image from ISO/IEC 19794-6 (Iris image data)

Returns

shared pointer of the Iris object

Exceptions

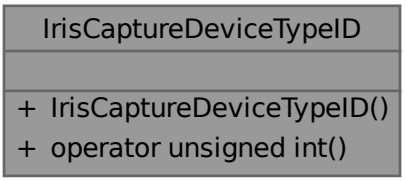
ImageException	when type is not supported or image data are corrupted
EnrollmentInvalidArgumentException	when not possible to decode ISO image

4.112 IrisCaptureDeviceTypeID Class Reference

The capture device type identifier shall identify the product type that created the BDIR.

```
#include <IrisCaptureDeviceTypeID.h>
```

Collaboration diagram for IrisCaptureDeviceTypeID:



Public Member Functions

- IrisCaptureDeviceTypeID (unsigned int i)

  Construct a new Iris Capture Equipment ID.
- **operator unsigned int** () const

  Conversion to capture equipment id number.

4.112.1 Detailed Description

It shall be assigned by the registered product owner or other approved registration authority. A value of all zeros shall indicate that the capture device type is unreported. [7]

4.112.2 Constructor & Destructor Documentation

4.112.2.1 IrisCaptureDeviceTypeID()

```
IrisCaptureDeviceTypeID (
            unsigned int i)  [inline], [explicit]
```

Parameters

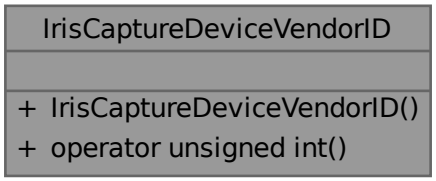
i	equipment ID determined by vendor. Only 16 bits are stored.

4.113 IrisCaptureDeviceVendorID Class Reference

Capture device vendor identifier The capture device vendor identifier shall identify the biometric organisation that owns the product that created the BDIR.

```
#include <IrisCaptureDeviceVendorID.h>
```

Collaboration diagram for IrisCaptureDeviceVendorID:



Public Member Functions

- IrisCaptureDeviceVendorID (unsigned int i)

  Construct a new Iris Capture Equipment ID.
- **operator unsigned int** () const

  Conversion to capture equipment id number.

4.113.1 Detailed Description

The capture device vendor identifier shall be encoded in two bytes carrying a CBEFF biometric organization identifier (registered by IBIA or other approved registration authority). A value of all zeros shall indicate that the capture device vendor is unreported. [7]

4.113.2 Constructor & Destructor Documentation

4.113.2.1 IrisCaptureDeviceVendorID()

```
IrisCaptureDeviceVendorID (
            unsigned int i)  [inline], [explicit]
```

Parameters

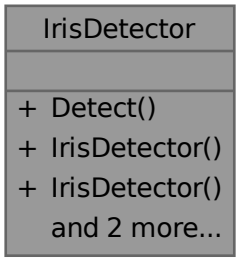
i	equipment ID determined by vendor. Only 16 bits are stored.

4.114 IrisDetector Class Reference

IrisDetector implements detection by means of Innovatrics IFace.

```
#include <IrisDetector.h>
```

Collaboration diagram for IrisDetector:



Classes

- class Config

  Iris detector configuration.

Public Member Functions

- std::shared_ptr< IrisSegment > Detect (const std::shared_ptr< Image > &image, const IrisPosition &position, const IrisAttributes &ia=IrisAttributes()) const

  Detect iris in given image.
- IrisDetector ()

  Construct a new IrisDetector object GlobalIrisExecutor is used for execution of iris functionality.
- IrisDetector (Config config)

  Construct a new IrisDetector object GlobalIrisExecutor is used for execution of iris functionality.
- IrisDetector (Config config, std::shared_ptr< IrisExecutor > executor)

  Construct a new IrisDetector object.
- IrisDetector (std::shared_ptr< IrisExecutor > executor)

  Construct a new IrisDetector object Default Config is used.

4.114.1 Constructor & Destructor Documentation

4.114.1.1 IrisDetector() [1/4]

```
IrisDetector () [inline]
```

Default Config is used.

4.114.1.2 IrisDetector() [2/4]

```
IrisDetector (
            Config config) [inline], [explicit]
```

Parameters

config	configuration

4.114.1.3 IrisDetector() [3/4]

```
IrisDetector (
            Config config,
            std::shared_ptr< IrisExecutor > executor) [inline]
```

Parameters

config	configuration
executor	is strategy for execution of iris functionality

Exceptions

NullArgumentException	when executor is null

4.114.1.4 IrisDetector() [4/4]

```
IrisDetector (
            std::shared_ptr< IrisExecutor > executor) [inline], [explicit]
```

Parameters

executor	is strategy for execution of iris functionality

Exceptions

NullArgumentException	when executor is null

4.114.2 Member Function Documentation

4.114.2.1 Detect()

```
std::shared_ptr< IrisSegment > Detect (
            const std::shared_ptr< Image > & image,
```

```
const IrisPosition & position,
const IrisAttributes & ia = IrisAttributes()) const  [inline], [nodiscard]
```

Parameters

image	Image where iris is detected
position	iris position as defined in IrisPosition.
ia	IrisAttributes

Returns

detected IrisSegment

Exceptions

EnrollmentException	when IFace returns error
EnrollmentRuntimeException	when no iris is detected
EnrollmentRuntimeException	when more than one iris is detected

Required Solvers

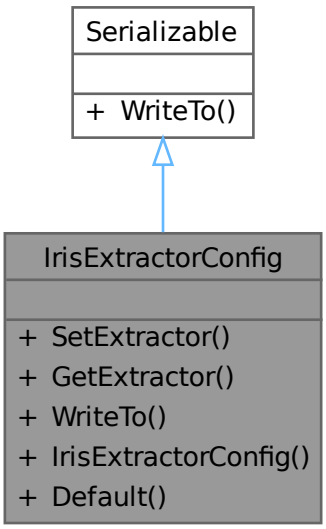
See IrisDetector::Config::Type

4.115 IrisExtractorConfig Class Reference

It contains extraction configuration.

`#include <Iris.h>`

Collaboration diagram for IrisExtractorConfig:



Public Types

- enum class Type : uint8_t { ACCURATE , FAST }

  Type of neural network extractor.

Public Member Functions

- void SetExtractor (Type value)

  Function sets type of extractor.
- Type GetExtractor () const

  Function gets extractor type.
- void WriteTo (Writer &writer) const final

  Function serializes configuration via provided Writer.
- IrisExtractorConfig ()=default

  default configuration.

Static Public Member Functions

- static IrisExtractorConfig Default ()

  Provides default configuration.

4.115.1 Member Enumeration Documentation

4.115.1.1 Type

```
enum class Type : uint8_t [strong]
```

Enumerator

ACCURATE	Iris templates suitable for verification/identification of very high accuracy are created when *accurate* mode is used. However the performance of the iris template creation is not as good as when `fast` mode is used. **Required Solvers** Total size of solvers is 174 MB. References iris-verification-accurate-p1.solver, iris-verification-accurate-p2.solver
FAST	Iris templates suitable for verification of fairly good accuracy are created when *fast* mode is used. The performance of iris template creation is very fast. Suitable for mobile/embedded devices. **Required Solvers** Total size of solvers is 83 MB. References iris-verification-accurate-p1.solver

4.115.2 Constructor & Destructor Documentation

4.115.2.1 IrisExtractorConfig()

```
IrisExtractorConfig () [default]
```

- type - Type::ACCURATE

4.115.3 Member Function Documentation

4.115.3.1 Default()

```
IrisExtractorConfig Default () [inline], [static]
```

default configuration.

- type - Type::ACCURATE

Returns

default configuration

4.115.3.2 GetExtractor()

```
Type GetExtractor () const [inline], [nodiscard]
```

Returns

extractor type

4.115.3.3 SetExtractor()

```
void SetExtractor (
            Type value) [inline]
```

Parameters

value	type from IrisExtractorConfig::Type.

4.115.3.4 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Config.

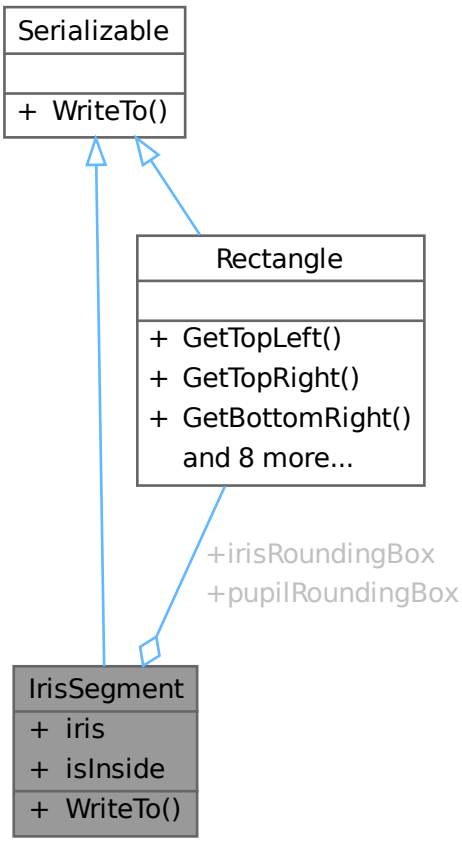
Implements Serializable.

4.116 IrisSegment Class Reference

It describes Iris segmented from original IrisCapture (one that contain this IrisSegment) with IrisDetector.

```
#include <IrisSegment.h>
```

Collaboration diagram for IrisSegment:



Public Member Functions

- void WriteTo (Writer &writer) const final

  Function serializes Iris segment.

Public Attributes

- const std::shared_ptr< Iris > **iris**

  Detected Iris image in original Iris capture.
- const Rectangle **pupilRoundingBox**

  Provides rounding box for pupil in image.
- const Rectangle **irisRoundingBox**

  Provides rounding box for iris in image.
- const bool **isInside**

  `true` *when rounding circle around iris of detected Iris is completely inside of original capture*

Friends

- class **IrisDetector**

4.116.1 Member Function Documentation

4.116.1.1 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize IrisSegment.

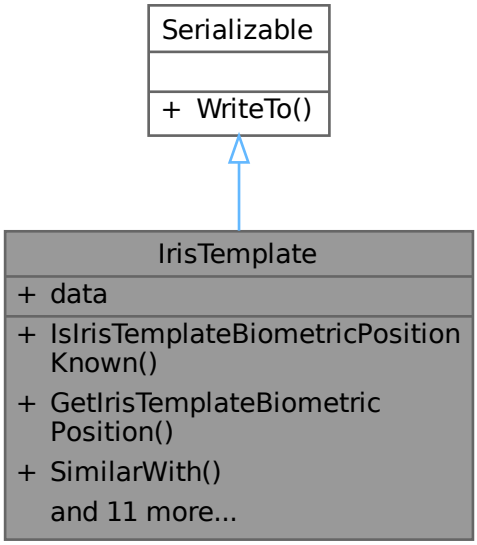
Implements Serializable.

4.117 IrisTemplate Class Reference

IrisTemplate is Innovatrics proprietary template created by IrisExtractor.

```
#include <Iris.h>
```

Collaboration diagram for IrisTemplate:



Public Member Functions

- bool IsIrisTemplateBiometricPositionKnown () const

  It checks if template contains known biometric position.
- IrisPosition GetIrisTemplateBiometricPosition () const

  It returns biometric position to which template belongs.
- IrisSimilarityScore SimilarWith (const IrisTemplate &gallery) const

  Calculates IrisSimilarityScore for given IrisTemplate.
- unsigned int CalculateQuality ()

  Calculates quality of IrisTemplate.
- std::string GetVersion () const

  Provides version information of Innovatrics proprietary algorithm used to create the iris template.
- void WriteTo (Writer &writer) const override

  Function serializes IrisTemplate via provided Writer.
- IrisTemplate (std::vector< uint8_t > &&templ, std::shared_ptr< IrisExecutor > executor)

  Construct a new IrisTemplate object Biometric position is unknown.
- IrisTemplate (std::vector< uint8_t > &&templ)

  Construct a new IrisTemplate object Biometric position is unknown.
- IrisTemplate (const std::vector< uint8_t > &templ, std::shared_ptr< IrisExecutor > executor)

  Construct a new IrisTemplate object, copy template data Biometric position is unknown.
- IrisTemplate (const std::vector< uint8_t > &templ)

  Construct a new IrisTemplate object, copy template data Biometric position is unknown.
- IrisTemplate (std::vector< uint8_t > &&templ, IrisPosition position, std::shared_ptr< IrisExecutor > executor)

  Construct a new IrisTemplate object.
- IrisTemplate (std::vector< uint8_t > &&templ, IrisPosition position)

  Construct a new IrisTemplate object GlobalIrisExecutor is used.

- IrisTemplate (const std::vector< uint8_t > &templ, IrisPosition position, std::shared_ptr< IrisExecutor > executor)

  Construct a new IrisTemplate object, copy template data.
- IrisTemplate (const std::vector< uint8_t > &templ, IrisPosition position)

  Construct a new IrisTemplate object, copy template data GlobalIrisExecutor is used.

Public Attributes

- const std::vector< uint8_t > **data**

  Template data.

4.117.1 Detailed Description

The template can be used later for verification.

4.117.2 Constructor & Destructor Documentation

4.117.2.1 IrisTemplate() [1/8]

```
IrisTemplate (
            std::vector< uint8_t > && templ,
            std::shared_ptr< IrisExecutor > executor)  [inline]
```

Parameters

templ	template data
executor	is IrisExecutor

Exceptions

NullArgumentException	when executor is null
InvalidTemplateException	when Iris template has no data

4.117.2.2 IrisTemplate() [2/8]

```
IrisTemplate (
            std::vector< uint8_t > && templ)  [inline], [explicit]
```

GlobalIrisExecutor is used.

Parameters

templ	template data

Exceptions

InvalidTemplateException	when Iris template has no data

4.117.2.3 IrisTemplate() [3/8]

```
IrisTemplate (
            const std::vector< uint8_t > & templ,
            std::shared_ptr< IrisExecutor > executor)  [inline]
```

Parameters

templ	template data
executor	is IrisExecutor

Exceptions

NullArgumentException	when executor is null
InvalidTemplateException	when Iris template has no data

4.117.2.4 IrisTemplate() [4/8]

```
IrisTemplate (
            const std::vector< uint8_t > & templ)  [inline], [explicit]
```

GlobalIrisExecutor is used.

Parameters

templ	template data

Exceptions

InvalidTemplateException	when Iris template has no data

4.117.2.5 IrisTemplate() [5/8]

```
IrisTemplate (
            std::vector< uint8_t > && templ,
            IrisPosition position,
            std::shared_ptr< IrisExecutor > executor)  [inline]
```

Parameters

templ	template data
executor	is IrisExecutor
position	biometric position to which template belongs

Exceptions

NullArgumentException	when executor is null
InvalidTemplateException	when Iris template has no data

4.117.2.6 IrisTemplate() [6/8]

```
IrisTemplate (
            std::vector< uint8_t > && templ,
            IrisPosition position)  [inline], [explicit]
```

Parameters

templ	template data
position	biometric position to which template belongs

Exceptions

InvalidTemplateException	when Iris template has no data

4.117.2.7 IrisTemplate() [7/8]

```
IrisTemplate (
            const std::vector< uint8_t > & templ,
            IrisPosition position,
            std::shared_ptr< IrisExecutor > executor)  [inline]
```

Parameters

templ	template data
position	biometric position to which template belongs
executor	is IrisExecutor

Exceptions

NullArgumentException	when executor is null
InvalidTemplateException	when Iris template has no data

4.117.2.8 IrisTemplate() [8/8]

```
IrisTemplate (
            const std::vector< uint8_t > & templ,
            IrisPosition position)  [inline], [explicit]
```

Parameters

templ	template data
position	biometric position to which template belongs

Exceptions

InvalidTemplateException	when Iris template has no data

4.117.3 Member Function Documentation

4.117.3.1 CalculateQuality()

```
unsigned int CalculateQuality ()  [inline]
```

Returns

quality of the template. It is based just on sharpness in visible area. It is normalized in range <0, 100> where 0 - min quality, 100 - max quality.

Exceptions

EnrollmentException	when it is not possible to get template quality

4.117.3.2 GetIrisTemplateBiometricPosition()

```
IrisPosition GetIrisTemplateBiometricPosition () const  [inline]
```

Returns

biometric position

Exceptions

EnrollmentRuntimeException	when biometric position is unknown

4.117.3.3 GetVersion()

```
std::string GetVersion () const  [inline], [nodiscard]
```

Returns

version string

4.117.3.4 IsIrisTemplateBiometricPositionKnown()

```
bool IsIrisTemplateBiometricPositionKnown () const  [inline]
```

Returns

`true` when biometric position is known

`false` when biometric position is unknown

4.117.3.5 SimilarWith()

```
IrisSimilarityScore SimilarWith (
            const IrisTemplate & gallery) const  [inline]
```

Parameters

gallery	IrisTemplate for score calculation

Returns

IrisSimilarityScore

Exceptions

EnrollmentException	when it is not possible to match templates

4.117.3.6 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [override], [virtual]
```

Parameters

writer	used to serialize Iris.

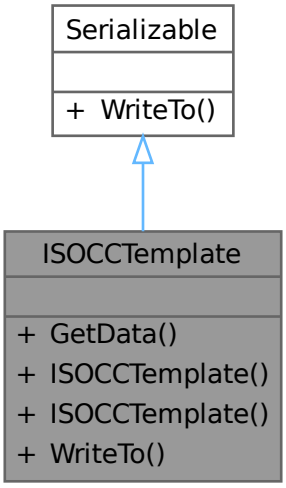
Implements Serializable.

4.118 ISOCCTemplate Class Reference

ISO/IEC 19794-2:2005 template, template version 2.0, 3-byte format for minutiae (compact card encoding).

```
#include <ISOCCTemplate.h>
```

Collaboration diagram for ISOCCTemplate:



Public Member Functions

- std::vector< uint8_t > GetData () const

  Provides template data.
- ISOCCTemplate (std::vector< uint8_t > &&d)

  Create ISOCCTemplate object move data to template.
- ISOCCTemplate (const std::vector< uint8_t > &d)

  Create ISOCCTemplate object, copy data to template.
- void WriteTo (Writer &writer) const final

  Write itself into writer.

4.118.1 Constructor & Destructor Documentation

4.118.1.1 ISOCCTemplate() [1/2]

```
ISOCCTemplate (
            std::vector< uint8_t > && d)  [inline], [explicit]
```

Parameters

d	template data

Exceptions

InvalidTemplateException	when template data are empty

4.118.1.2 ISOCCTemplate() [2/2]

```
ISOCCTemplate (
            const std::vector< uint8_t > & d)  [inline], [explicit]
```

Parameters

d	template data

Exceptions

InvalidTemplateException	when template data are empty

4.118.2 Member Function Documentation

4.118.2.1 GetData()

```
std::vector< uint8_t > GetData () const  [inline], [nodiscard]
```

Returns

template data

4.118.2.2 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

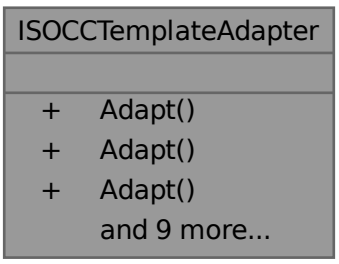
Implements Serializable.

4.119 ISOCCTemplateAdapter Class Reference

Adapts Template to ISOCCTemplate.

```
#include <ISOCCTemplateAdapter.h>
```

Collaboration diagram for ISOCCTemplateAdapter:



Public Types

- enum class MinutiaSortOrder : uint8_t {
  NONE = 0 , X_ASC = 1 , X_DESC = 2 , Y_ASC = 3 ,
  Y_DESC = 4 }

  Defines sort order of minutiae points in ISO CC template.

Public Member Functions

- ISOCCTemplate Adapt (const ANSITemplate &tmpl) const

  Adapts ANSITemplate to ISOCCTemplate.
- ISOCCTemplate Adapt (const ILOSIDTemplate &tmpl) const

  Adapts ILOSIDTemplate to ISOCCTemplate.
- ISOCCTemplate Adapt (const ISONCTemplate &tmpl) const

  Adapts ISONCTemplate to ISOCCTemplate.
- ISOCCTemplate Adapt (const ISOCCTemplate &tmpl) const

  Adapts ISOCCTemplate to ISOCCTemplate.
- ISOCCTemplate Adapt (const ISOTemplate &tmpl) const

  Adapts ISOTemplate to ISOCCTemplate.
- ISOCCTemplate Adapt (const ICSTemplate &tmpl) const

  Adapts ICSTemplate to ISOCCTemplate.
- ISOCCTemplate Adapt (const ExtractedPrint &tmpl) const

  Adapts ExtractedPrint to ISOCCTemplate.
- ISOCCTemplate Adapt (const PrintTemplate &tmpl) const

  Adapts PrintTemplate to ISOCCTemplate.
- **ISOCCTemplateAdapter** ()

  Creates ISOCCTemplateAdapter object with GlobalPrintExecutor used to execute template functionality.

- ISOCCTemplateAdapter (int maximumMinutiaeCount, MinutiaSortOrder primarySortOrder, MinutiaSortOrder secondarySortOrder)

  Creates ISOCCTemplateAdapter object with minutiae modifications.
- ISOCCTemplateAdapter (std::shared_ptr< PrintExecutor > executor)

  Creates ISOCCTemplateAdapter object.
- ISOCCTemplateAdapter (std::shared_ptr< PrintExecutor > executor, int maximumMinutiaeCount, MinutiaSortOrder primarySortOrder, MinutiaSortOrder secondarySortOrder)

  Creates ISOCCTemplateAdapter object with minutiae limitations and given PrintExecutor.

4.119.1 Member Enumeration Documentation

4.119.1.1 MinutiaSortOrder

```
enum class MinutiaSortOrder : uint8_t [strong]
```

Enumerator

NONE	No ordering required.
X_ASC	Cartesian x-coordinate is used for ordering, ascending order.
X_DESC	Cartesian x-coordinate is used for ordering, descending order.
Y_ASC	Cartesian y-coordinate is used for ordering, ascending order.
Y_DESC	Cartesian y-coordinate is used for ordering, descending order.

4.119.2 Constructor & Destructor Documentation

4.119.2.1 ISOCCTemplateAdapter() [1/3]

```
ISOCCTemplateAdapter (
            int maximumMinutiaeCount,
            MinutiaSortOrder primarySortOrder,
            MinutiaSortOrder secondarySortOrder) [inline]
```

GlobalPrintExecutor is used to execute template functionality.

Parameters

maximumMinutiaeCount	The maximal number of minutiae that will be stored in the output template. If the input template contains more minutiae points than the provided limit, extra minutiae point are removed from the template. The truncation is made by peeling off minutiae that are farthest from the point of gravity of the minutiae set.
primarySortOrder	Defines the primary ordering criteria for minutiae
secondarySortOrder	Defines the secondary ordering criteria for minutiae (used when primary ordering criteria gives equality).

4.119.2.2 ISOCCTemplateAdapter() [2/3]

```
ISOCCTemplateAdapter (
            std::shared_ptr< PrintExecutor > executor)  [inline], [explicit]
```

Parameters

executor	PrintExecutor used to execute template functionality.

Exceptions

NullArgumentException	when executor is null

4.119.2.3 ISOCCTemplateAdapter() [3/3]

```
ISOCCTemplateAdapter (
            std::shared_ptr< PrintExecutor > executor,
            int maximumMinutiaeCount,
            MinutiaSortOrder primarySortOrder,
            MinutiaSortOrder secondarySortOrder)  [inline]
```

Parameters

executor	PrintExecutor used to execute template functionality.
maximumMinutiaeCount	The maximal number of minutiae that will be stored in the output template. If the input template contains more minutiae points than the provided limit, extra minutiae point are removed from the template. The truncation is made by peeling off minutiae that are farthest from the point of gravity of the minutiae set.
primarySortOrder	Defines the primary ordering criteria for minutiae
secondarySortOrder	Defines the secondary ordering criteria for minutiae (used when primary ordering criteria gives equality).

Exceptions

NullArgumentException	when executor is null

4.119.3 Member Function Documentation

4.119.3.1 Adapt() [1/8]

```
ISOCCTemplate Adapt (
            const ANSITemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOCCTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.119.3.2 Adapt() [2/8]

```
ISOCCTemplate Adapt (
            const ExtractedPrint & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOCCTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.119.3.3 Adapt() [3/8]

```
ISOCCTemplate Adapt (
            const ICSTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOCCTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.119.3.4 Adapt() [4/8]

```
ISOCCTemplate Adapt (
            const ILOSIDTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOCCTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.119.3.5 Adapt() [5/8]

```
ISOCCTemplate Adapt (
            const ISOCCTemplate & tmpl) const  [inline], [nodiscard]
```

When minutiae modifications are used in in class constructor then function modifies given ISOCCTemplate otherwise it creates same new ISOCCTemplate.

Parameters

tmpl	template to adapt

Returns

adapted ISOCCTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.119.3.6 Adapt() [6/8]

```
ISOCCTemplate Adapt (
            const ISONCTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOCCTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.119.3.7 Adapt() [7/8]

```
ISOCCTemplate Adapt (
            const ISOTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOCCTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.119.3.8 Adapt() [8/8]

```
ISOCCTemplate Adapt (
            const PrintTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOCCTemplate

Exceptions

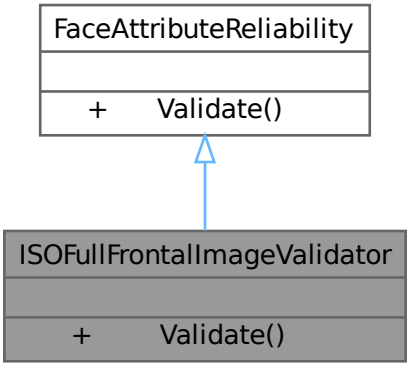
EnrollmentException	when it is not possible to adapt template

4.120 ISOFullFrontalImageValidator Class Reference

Check if usage of Full Frontal Images on travel documents will follow best practices, as described in Section B.3.2 Best practices for use of Full Frontal Images on travel documents [6] .

```
#include <FaceValidation.h>
```

Collaboration diagram for ISOFullFrontalImageValidator:



Public Member Functions

- std::vector< FailedFaceAttribute > Validate (const std::shared_ptr< Face > &probe) final

  Check if usage of Full Frontal image on travel documents will be reliable.

4.120.1 Detailed Description

For a Full Frontal Image, the (Image Width: Image Height) aspect ratio should be between 1:1,25 and 1:1,34. This allows for ratio of 1:1,25 specified by NIST best practices for mug shots, 1:1,28 used in many passport images, and 1:1,33 used in many driver's license images. [6]

4.120.2 Member Function Documentation

4.120.2.1 Validate()

```
std::vector< FailedFaceAttribute > Validate (
            const std::shared_ptr< Face > & probe)  [inline], [final], [virtual]
```

Parameters

probe	Face to be checked

Returns

list of attributes that will cause not reliable usage the image on travel documents

Exceptions

NullArgumentException	when probe is null

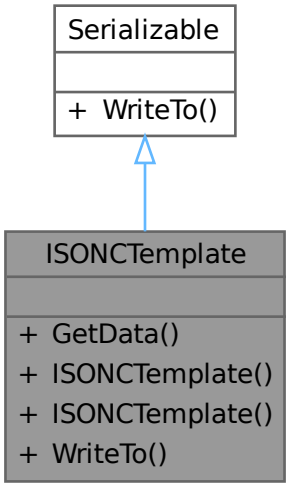
Implements FaceAttributeReliability.

4.121 ISONCTemplate Class Reference

ISO/IEC 19794-2:2005 template, template version 2.0, 5-byte format for minutiae (normal card encoding).

```
#include <ISONCTemplate.h>
```

Collaboration diagram for ISONCTemplate:



Public Member Functions

- std::vector< uint8_t > GetData () const

  Provides template data.
- ISONCTemplate (std::vector< uint8_t > &&d)

  Create ISONCTemplate object move data to template.
- ISONCTemplate (const std::vector< uint8_t > &d)

  Create ISONCTemplate object, copy data to template.
- void WriteTo (Writer &writer) const final

  Write itself into writer.

4.121.1 Constructor & Destructor Documentation

4.121.1.1 ISONCTemplate() [1/2]

```
ISONCTemplate (
            std::vector< uint8_t > && d)  [inline], [explicit]
```

Parameters

d	template data

Exceptions

InvalidTemplateException	when template data are empty

4.121.1.2 ISONCTemplate() [2/2]

```
ISONCTemplate (
            const std::vector< uint8_t > & d)  [inline], [explicit]
```

Parameters

d	template data

Exceptions

InvalidTemplateException	when template data are empty

4.121.2 Member Function Documentation

4.121.2.1 GetData()

```
std::vector< uint8_t > GetData () const  [inline], [nodiscard]
```

Returns

template data

4.121.2.2 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

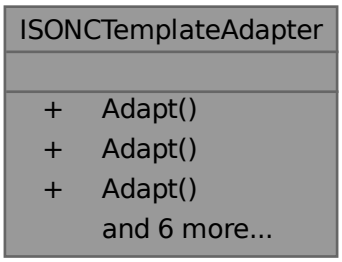
Implements Serializable.

4.122 ISONCTemplateAdapter Class Reference

Adapts Template to ISONCTemplate.

```
#include <ISONCTemplateAdapter.h>
```

Collaboration diagram for ISONCTemplateAdapter:



Public Member Functions

- ISONCTemplate Adapt (const ANSITemplate &tmpl) const

  Adapts ANSITemplate to ISONCTemplate.
- ISONCTemplate Adapt (const ILOSIDTemplate &tmpl) const

  Adapts ILOSIDTemplate to ISONCTemplate.
- ISONCTemplate Adapt (const ISOCCTemplate &tmpl) const

  Adapts ISOCCTemplate to ISONCTemplate.
- ISONCTemplate Adapt (const ISOTemplate &tmpl) const

  Adapts ISOTemplate to ISONCTemplate.
- ISONCTemplate Adapt (const ICSTemplate &tmpl) const

  Adapts ICSTemplate to ISONCTemplate.
- ISONCTemplate Adapt (const ExtractedPrint &tmpl) const

  Adapts ExtractedPrint to ISONCTemplate.
- ISONCTemplate Adapt (const PrintTemplate &tmpl) const

  Adapts PrintTemplate to ISONCTemplate.
- **ISONCTemplateAdapter** ()

  Creates ISONCTemplateAdapter object with GlobalPrintExecutor used to execute template functionality.
- ISONCTemplateAdapter (std::shared_ptr< PrintExecutor > executor)

  Creates ISONCTemplateAdapter object.

4.122.1 Constructor & Destructor Documentation

4.122.1.1 ISONCTemplateAdapter()

```
ISONCTemplateAdapter (
            std::shared_ptr< PrintExecutor > executor)  [inline], [explicit]
```

Parameters

executor	PrintExecutor used to execute template functionality.

Exceptions

NullArgumentException	when executor is null

4.122.2 Member Function Documentation

4.122.2.1 Adapt() [1/7]

```
ISONCTemplate Adapt (
            const ANSITemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISONCTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.122.2.2 Adapt() [2/7]

```
ISONCTemplate Adapt (
            const ExtractedPrint & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISONCTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.122.2.3 Adapt() [3/7]

```
ISONCTemplate Adapt (
            const ICSTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISONCTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.122.2.4 Adapt() [4/7]

```
ISONCTemplate Adapt (
            const ILOSIDTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISONCTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.122.2.5 Adapt() [5/7]

```
ISONCTemplate Adapt (
            const ISOCCTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISONCTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.122.2.6 Adapt() [6/7]

```
ISONCTemplate Adapt (
            const ISOTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISONCTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.122.2.7 Adapt() [7/7]

```
ISONCTemplate Adapt (
            const PrintTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISONCTemplate

Exceptions

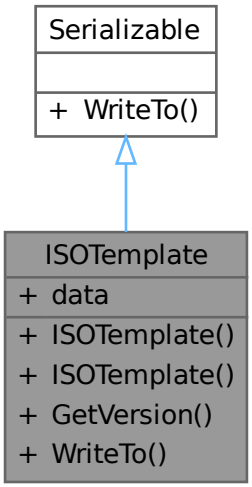
EnrollmentException	when it is not possible to adapt template

4.123 ISOTemplate Class Reference

ISO/IEC 19794-2 print compliant template.

```
#include <ISOTemplate.h>
```

Collaboration diagram for ISOTemplate:



Public Member Functions

- ISOTemplate (std::vector< uint8_t > &&d) noexcept

  Create ISOTemplate object.
- ISOTemplate (const std::vector< uint8_t > &d)

  Create ISOTemplate object, copy data to ISO template.
- std::string GetVersion () const

  Provides version information of Innovatrics proprietary algorithm used to create the ISO template.
- void WriteTo (Writer &writer) const final

  Write itself into writer.

Public Attributes

- const std::vector< uint8_t > **data**

  Template data.

4.123.1 Constructor & Destructor Documentation

4.123.1.1 ISOTemplate() [1/2]

```
ISOTemplate (
            std::vector< uint8_t > && d)  [inline], [explicit], [noexcept]
```

Parameters

d	template data

4.123.1.2 ISOTemplate() [2/2]

```
ISOTemplate (
            const std::vector< uint8_t > & d)  [inline], [explicit]
```

Parameters

d	template data

4.123.2 Member Function Documentation

4.123.2.1 GetVersion()

```
std::string GetVersion () const  [inline], [nodiscard]
```

Returns

version string

Exceptions

EnrollmentRuntimeException	when it is not possible to read version from the template.

4.123.2.2 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

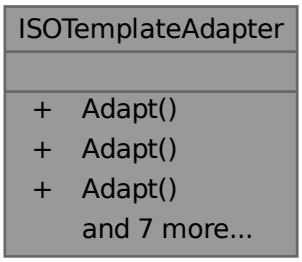
Implements Serializable.

4.124 ISOTemplateAdapter Class Reference

Adapts Template to ISOTemplate.

```
#include <ISOTemplateAdapter.h>
```

Collaboration diagram for ISOTemplateAdapter:



Public Types

- enum class Version : uint8_t { VERSION_2 = 0 , VERSION_3 = 1 }

  ISO template version.

Public Member Functions

- ISOTemplate Adapt (const ANSITemplate &tmpl) const

  Adapts ANSITemplate to ISOTemplate.
- ISOTemplate Adapt (const ILOSIDTemplate &tmpl) const

  Adapts ILOSIDTemplate to ISOTemplate.
- ISOTemplate Adapt (const ISONCTemplate &tmpl) const

  Adapts ISONCTemplate to ISOTemplate.
- ISOTemplate Adapt (const ISOCCTemplate &tmpl) const

  Adapts ISOCCTemplate to ISOTemplate.
- ISOTemplate Adapt (const ISOTemplate &tmpl) const

  Adapts ISOTemplate to ISOTemplate.
- ISOTemplate Adapt (const ICSTemplate &tmpl) const

  Adapts ICSTemplate to ISOTemplate.
- ISOTemplate Adapt (const ExtractedPrint &tmpl) const

  Adapts ExtractedPrint to ISOTemplate.
- ISOTemplate Adapt (const PrintTemplate &tmpl) const

  Adapts PrintTemplate to ISOTemplate.
- ISOTemplateAdapter (Version version)

  Creates ISOTemplateAdapter object with GlobalPrintExecutor used to execute template functionality.
- ISOTemplateAdapter (Version version, std::shared_ptr< PrintExecutor > executor)

  Creates ISOTemplateAdapter object.

4.124.1 Member Enumeration Documentation

4.124.1.1 Version

```
enum class Version : uint8_t [strong]
```

Enumerator

VERSION_2	ISO/IEC 19794-2:2005 template, template version 2.0, 6-byte format for minutiae.
VERSION_3	ISO/IEC 19794-2:2011 template, template version 3.0, 6-byte format for minutiae.

4.124.2 Constructor & Destructor Documentation

4.124.2.1 ISOTemplateAdapter() [1/2]

```
ISOTemplateAdapter (
            Version version) [inline], [explicit]
```

Parameters

version	requested ISO template version

4.124.2.2 ISOTemplateAdapter() [2/2]

```
ISOTemplateAdapter (
            Version version,
            std::shared_ptr< PrintExecutor > executor) [inline], [explicit]
```

Parameters

executor	PrintExecutor used to execute template functionality.
version	requested ISO template version

Exceptions

NullArgumentException	when executor is null

4.124.3 Member Function Documentation

4.124.3.1 Adapt() [1/8]

```
ISOTemplate Adapt (
            const ANSITemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.124.3.2 Adapt() [2/8]

```
ISOTemplate Adapt (
            const ExtractedPrint & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.124.3.3 Adapt() [3/8]

```
ISOTemplate Adapt (
            const ICSTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.124.3.4 Adapt() [4/8]

```
ISOTemplate Adapt (
            const ILOSIDTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.124.3.5 Adapt() [5/8]

```
ISOTemplate Adapt (
            const ISOCCTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.124.3.6 Adapt() [6/8]

```
ISOTemplate Adapt (
            const ISONCTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.124.3.7 Adapt() [7/8]

```
ISOTemplate Adapt (
            const ISOTemplate & tmpl) const  [inline], [nodiscard]
```

it converts ISO template to template version given in constructor.

Parameters

tmpl	template to adapt

Returns

adapted ISOTemplate

Exceptions

EnrollmentException	when it is not possible to adapt template

4.124.3.8 Adapt() [8/8]

```
ISOTemplate Adapt (
            const PrintTemplate & tmpl) const  [inline], [nodiscard]
```

Parameters

tmpl	template to adapt

Returns

adapted ISOTemplate

Exceptions

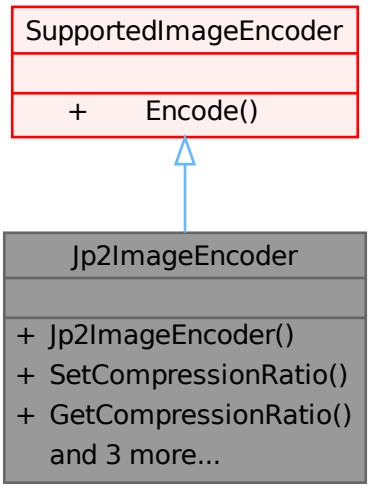
EnrollmentException	when it is not possible to adapt template

4.125 Jp2ImageEncoder Class Reference

Converts image type ImageType::JPG2000.

```
#include <ImageEncoder.h>
```

Collaboration diagram for Jp2ImageEncoder:



Public Member Functions

- **Jp2ImageEncoder** ()

  Construct a new Jpeg 2000 Image Encoder object The default compression ratio is 4:1.
- void SetCompressionRatio (int ratio)

  Set compression ratio.
- int GetCompressionRatio () const

  Provides compression ratio.
- bool IsLossy () const

  Check if encoder setup is lossy.
- bool IsLossless () const

  Check if encoder setup is lossless.
- void Accept (ImageEncoderVisitor &visitor) const final

  Allow visitor to visit concrete type of encoder.

Public Member Functions inherited from SupportedImageEncoder

- std::vector< uint8_t > Encode (const RawImage &raw) const final

  Saves given RawImage as formatted image.

Additional Inherited Members

Static Public Member Functions inherited from ImageEncoder

- static std::shared_ptr< ImageEncoder > Create (ImageType type)

  Create concrete default encoder for given type.

4.125.1 Member Function Documentation

4.125.1.1 Accept()

```
void Accept (
            ImageEncoderVisitor & visitor) const [inline], [final], [virtual]
```

Parameters

visitor	ImageEncoderVisitor to call

Implements ImageEncoder.

4.125.1.2 GetCompressionRatio()

```
int GetCompressionRatio () const [inline]
```

Returns

compression ratio, the value 1 has meaning lossless compression

4.125.1.3 IsLossless()

```
bool IsLossless () const [inline]
```

Returns

true lossless encoding

false lossy encoding

4.125.1.4 IsLossy()

```
bool IsLossy () const  [inline]
```

Returns

true lossy encoding

false lossless encoding

4.125.1.5 SetCompressionRatio()

```
void SetCompressionRatio (
            int ratio)  [inline]
```

In JPEG2000, the compression ratio is defined as the ratio of the size of the uncompressed, raw image to the size of the compressed image. $raw_size = width * height * number_of_channels * bit_depth/8$ where `bit_depth` is number of bits in channel (e.g. 8,16). For example, a compression ratio of 20:1 indicates that the compressed image is 1/20th the size of the raw image. Be aware that even if compression ratio is `1` (lossless) the image will not have `raw_size` because of used compaction.

To get for example half (ratio 2:1) of image size (the encoded lossless size) use formula $raw_size/lossless_encoded_jp2_size * 2$. The size will be closed to half due to additional size of JP2 overhead.

Parameters

ratio	specifies compression ratio `ratio:1`. It means that image size will be `1/ratio` of uncompressed, raw image size. valid range is <1,1000>. The value 1 mean no compression, lossless.

Exceptions

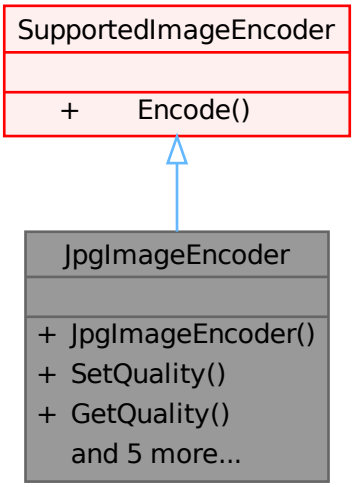
EnrollmentInvalidArgumentException	if the bitrate is not in the valid range <1,1000>.

4.126 JpgImageEncoder Class Reference

Converts image type ImageType::JPG.

```
#include <ImageEncoder.h>
```

Collaboration diagram for JpgImageEncoder:



Public Member Functions

- JpgImageEncoder ()

  Construct a new JPG Image Encoder object.
- void SetQuality (int quality)

  Set quality.
- int GetQuality () const

  Provides quality.
- void SetProgressive (bool isOn)

  Set the Progressive Feature A progressive JPEG image is encoded differently than a standard or baseline JPEG image.
- bool GetProgressive () const

  Provides Progressive Feature.
- void SetOptimize (bool isOn)

  Set the Optimize Feature.
- bool GetOptimize () const

  Provides Optimize Feature.
- void Accept (ImageEncoderVisitor &visitor) const final

  Allow visitor to visit concrete type of encoder.

Public Member Functions inherited from SupportedImageEncoder

- std::vector< uint8_t > Encode (const RawImage &raw) const final

  Saves given RawImage as formatted image.

Additional Inherited Members

Static Public Member Functions inherited from ImageEncoder

- static std::shared_ptr< ImageEncoder > Create (ImageType type)
  Create concrete default encoder for given type.

4.126.1 Constructor & Destructor Documentation

4.126.1.1 JpgImageEncoder()

```
JpgImageEncoder () [inline]
```

- Default quality is 95
- progressive feature is off
- optimize feature is off

4.126.2 Member Function Documentation

4.126.2.1 Accept()

```
void Accept (
            ImageEncoderVisitor & visitor) const [inline], [final], [virtual]
```

Parameters

visitor	ImageEncoderVisitor to call

Implements ImageEncoder.

4.126.2.2 GetOptimize()

```
bool GetOptimize () const [inline]
```

Returns

`true` when optimize Feature is on
`false` when optimize Feature is off

4.126.2.3 GetProgressive()

```
bool GetProgressive () const [inline]
```

Returns

`true` when progressive Feature is on
`false` when progressive Feature is off

4.126.2.4 GetQuality()

```
int GetQuality () const  [inline]
```

Returns

quality

4.126.2.5 SetOptimize()

```
void SetOptimize (
            bool isOn)  [inline]
```

Parameters

isOn	when true the feature is enabled

4.126.2.6 SetProgressive()

```
void SetProgressive (
            bool isOn)  [inline]
```

It loads in successive waves until a clear picture is formed. This can improve a website's performance as the images seems to be loading faster.

Parameters

isOn	when true the feature is enabled

4.126.2.7 SetQuality()

```
void SetQuality (
            int quality)  [inline]
```

Parameters

quality	from 0 to 100 (the higher is the better).

Exceptions

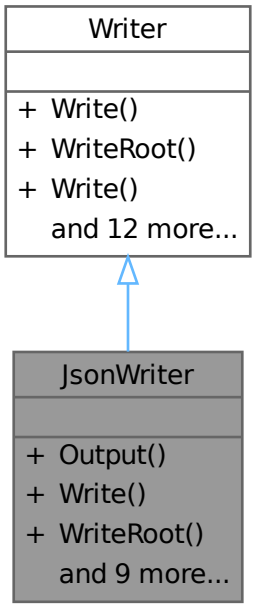
EnrollmentInvalidArgumentException	when quality is not in range

4.127 JsonWriter Class Reference

JSON implementation of Writer abstraction.

```
#include <JsonWriter.h>
```

Collaboration diagram for JsonWriter:



Public Member Functions

- const char ∗ Output () const

  Provides serialization output.
- void Write (const char ∗name, const Serializable &s) final

  Serialize serializable.
- void WriteRoot (Serializable &s) final

  Serialize serializable as root object.
- void WriteNull (const char ∗name) final

  Serialize null object.
- void Write (const char ∗name, const char ∗value) final

  Serialize string.
- void Write (const char ∗name, int value) final

  Serialize int.
- void Write (const char ∗name, unsigned int value) final

  Serialize unsigned int.
- void Write (const char ∗name, double value) final

  Serialize double.
- void Write (const char ∗name, int64_t value) final

Serialize int64_t integer.

- void Write (const char *name, uint64_t value) final

  Serialize uint64_t integer.
- void Write (const char *name, const std::shared_ptr< Image > &image) final

  Serialize Image.
- **JsonWriter** ()

  Constructor sets up serialization backend.

Public Member Functions inherited from Writer

- void WriteArray (const char *name, const std::vector< uint8_t > &value)

  Serialize bytes from vector.
- void SetBytesStrategy (BytesWriterStrategy &s)

  Set bytes serialization strategy .
- template<typename T>
  void WriteArray (const char *name, const T &array)

  Serialize array of elements that are suitable for `range-based for loop`.
- void Write (const char *name, const std::shared_ptr< const Serializable > &s)

  Serialize const serializable given as shared_ptr.
- template<typename T>
  void Write (const char *name, const std::optional< T > &s)

  Serialize serializable given as shared_ptr.

4.127.1 Detailed Description

It uses RapidJSON

4.127.2 Member Function Documentation

4.127.2.1 Output()

```
const char * Output () const  [inline], [nodiscard]
```

Returns

serialization output

4.127.2.2 Write() [1/8]

```
void Write (
            const char * name,
            const char * value)  [inline], [final], [virtual]
```

Parameters

name	name of value (key)

value	string value

Implements Writer.

4.127.2.3 Write() [2/8]

```
void Write (
            const char * name,
            const Serializable & s)  [inline], [final], [virtual]
```

Parameters

name	name of serializable
s	class that implements Serializable interface

Implements Writer.

4.127.2.4 Write() [3/8]

```
void Write (
            const char * name,
            const std::shared_ptr< Image > & value)  [inline], [final], [virtual]
```

Once the image serialization needs to be modified e.g. all images are serialized in specific format, then strategy can be used similar as BytesWriterStrategy.

Parameters

name	name of value (key)
value	Image

Implements Writer.

4.127.2.5 Write() [4/8]

```
void Write (
            const char * name,
            double value)  [inline], [final], [virtual]
```

Parameters

name	name of value (key)
value	double value

Implements Writer.

4.127.2.6 Write() [5/8]

```
void Write (
            const char * name,
            int value)  [inline], [final], [virtual]
```

Parameters

name	name of value (key)
value	integer value

Implements Writer.

4.127.2.7 Write() [6/8]

```
void Write (
            const char * name,
            int64_t value)  [inline], [final], [virtual]
```

Parameters

name	name of value (key)
value	unsigned integer value

Implements Writer.

4.127.2.8 Write() [7/8]

```
void Write (
            const char * name,
            uint64_t value)  [inline], [final], [virtual]
```

Parameters

name	name of value (key)
value	unsigned integer value

Implements Writer.

4.127.2.9 Write() [8/8]

```
void Write (
            const char * name,
            unsigned int value)  [inline], [final], [virtual]
```

Parameters

name	name of value (key)
value	unsigned integer value

Implements Writer.

4.127.2.10 WriteNull()

```
void WriteNull (
            const char * name)  [inline], [final], [virtual]
```

Parameters

name	name of value (key)

Implements Writer.

4.127.2.11 WriteRoot()

```
void WriteRoot (
            Serializable & s)  [inline], [final], [virtual]
```

Parameters

s	class that implements Serializable interface

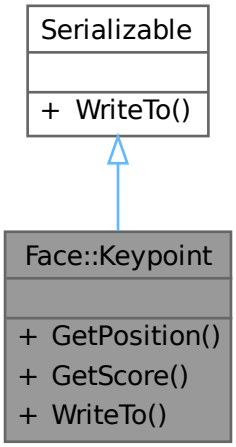
Implements Writer.

4.128 Face::Keypoint Class Reference

It holds all attributes related to a facial key point.

```
#include <Face.h>
```

Collaboration diagram for Face::Keypoint:



Public Member Functions

- Point GetPosition () const

  Retrieve position coordinates of the key point in the image.
- float GetScore () const

  Retrieve reliability score of the key point.
- void WriteTo (Writer &writer) const final

  Write itself into writer.

4.128.1 Member Function Documentation

4.128.1.1 GetPosition()

```
Point GetPosition () const  [inline], [nodiscard]
```

Returns

Position of the key point in the image.

Remarks

Position of output facial feature is always relative to the input image origin. Image origin is pixel[0,0] is upper left corner.

4.128.1.2 GetScore()

```
float GetScore () const [inline], [nodiscard]
```

Returns

Score of the key point.

Remarks

Score values are from range <0,10000>. Values near 10000 indicates the facial key point as reliable. The decision threshold is around 3000.

4.128.1.3 WriteTo()

```
void WriteTo (
            Writer & writer) const [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

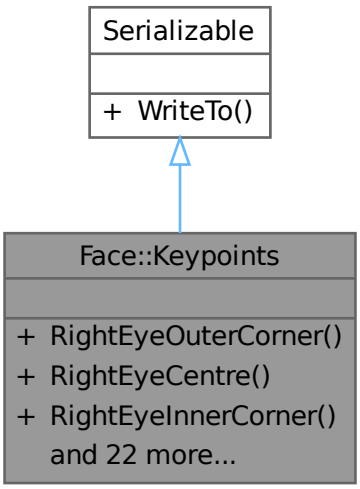
Implements Serializable.

4.129 Face::Keypoints Class Reference

It holds all facial keypoints detected on a FaceCapture image.

```
#include <Face.h>
```

Collaboration diagram for Face::Keypoints:



Public Member Functions

- Keypoint RightEyeOuterCorner ()

  Key point for evaluating reliability of right eye outer corner.
- Keypoint RightEyeCentre ()

  Key point for evaluating reliability of right eye centre.
- Keypoint RightEyeInnerCorner ()

  Key point for evaluating reliability of inner corner of right eye.
- Keypoint LeftEyeInnerCorner ()

  Key point for evaluating reliability of inner corner of left eye.
- Keypoint LeftEyeCentre ()

  Key point for evaluating reliability of left eye centre.
- Keypoint LeftEyeOuterCorner ()

  Key point for evaluating reliability of outer corner of left eye.
- Keypoint NoseRoot ()

  Key point for evaluating reliability of nose root.
- Keypoint NoseRightBottom ()

  Key point for evaluating reliability of right bottom of nose.
- Keypoint NoseTip ()

  Key point for evaluating reliability of nose tip.
- Keypoint NoseLeftBottom ()

  Key point for evaluating reliability of left bottom of nose.
- Keypoint NoseBottom ()

  Key point for evaluating reliability of bottom of nose.
- Keypoint MouthRightCorner ()

  Key point for evaluating reliability of right corner of mouth.
- Keypoint MouthCenter ()

  Key point for evaluating reliability of center of mouth.
- Keypoint MouthLeftCorner ()

  Key point for evaluating reliability of left corner of mouth.
- Keypoint MouthUpperEdge ()

  Key point for evaluating reliability of upper edge of mouth.
- Keypoint MouthLowerEdge ()

  Key point for evaluating reliability of lower edge of mouth.
- Keypoint RightEyebrowOuterEnd ()

  Key point for evaluating reliability of outer end of right eyebrow.
- Keypoint RightEyebrowInnerEnd ()

  Key point for evaluating reliability of inner end of right eyebrow.
- Keypoint LeftEyebrowInnerEnd ()

  Key point for evaluating reliability of inner end of left eyebrow.
- Keypoint LeftEyebrowOuterEnd ()

  Key point for evaluating reliability of outer end of left eyebrow.
- Keypoint RightEdge ()

  Key point for evaluating reliability of right edge of face.
- Keypoint ChinTip ()

  Key point for evaluating reliability of tip of chin.
- Keypoint LeftEdge ()

  Key point for evaluating reliability of left edge of face.
- std::shared_ptr< Image > GetImage () const

  Get image related to the detected keypoints.
- void WriteTo (Writer &writer) const final

  Function serializes keypoints via provided Writer.

Friends

- class **Face**

4.129.1 Detailed Description

The keypoint coordinates are always relative to the input image origin (one from which face was detected and segmented). Image origin is pixel[0,0] is upper left corner.

4.129.2 Member Function Documentation

4.129.2.1 ChinTip()

```
Keypoint ChinTip () [inline]
```

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.2 GetImage()

```
std::shared_ptr< Image > GetImage () const [inline], [nodiscard]
```

Returns

Face image

4.129.2.3 LeftEdge()

```
Keypoint LeftEdge () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's left side of face.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.4 LeftEyebrowInnerEnd()

```
Keypoint LeftEyebrowInnerEnd () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's left eye.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.5 LeftEyebrowOuterEnd()

```
Keypoint LeftEyebrowOuterEnd () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's left eye.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.6 LeftEyeCentre()

```
Keypoint LeftEyeCentre () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's left eye.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.7 LeftEyeInnerCorner()

```
Keypoint LeftEyeInnerCorner () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's left eye.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.8 LeftEyeOuterCorner()

```
Keypoint LeftEyeOuterCorner () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's left eye.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.9 MouthCenter()

```
Keypoint MouthCenter () [inline]
```

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.10 MouthLeftCorner()

```
Keypoint MouthLeftCorner () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's left side of face.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.11 MouthLowerEdge()

```
Keypoint MouthLowerEdge () [inline]
```

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.12 MouthRightCorner()

```
Keypoint MouthRightCorner () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's right side of face.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.13 MouthUpperEdge()

```
Keypoint MouthUpperEdge () [inline]
```

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.14 NoseBottom()

```
Keypoint NoseBottom () [inline]
```

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.15 NoseLeftBottom()

```
Keypoint NoseLeftBottom () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's left side of face.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.16 NoseRightBottom()

```
Keypoint NoseRightBottom () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's right side of face.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.17 NoseRoot()

```
Keypoint NoseRoot () [inline]
```

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.18 NoseTip()

```
Keypoint NoseTip () [inline]
```

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.19 RightEdge()

```
Keypoint RightEdge () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's right side of face.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.20 RightEyebrowInnerEnd()

```
Keypoint RightEyebrowInnerEnd () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's right eye.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.21 RightEyebrowOuterEnd()

```
Keypoint RightEyebrowOuterEnd () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's right eye.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.22 RightEyeCentre()

```
Keypoint RightEyeCentre () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's right eye.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.23 RightEyeInnerCorner()

```
Keypoint RightEyeInnerCorner () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's right eye.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.24 RightEyeOuterCorner()

```
Keypoint RightEyeOuterCorner () [inline]
```

The key point represents the anatomical position on the face (not the position as seen in the image), i.e. the person's right eye.

Returns

Instance of the key point defined by its position in the image and reliability score.

Exceptions

EnrollmentException	when key point can not be calculated

4.129.2.25 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize keypoints.

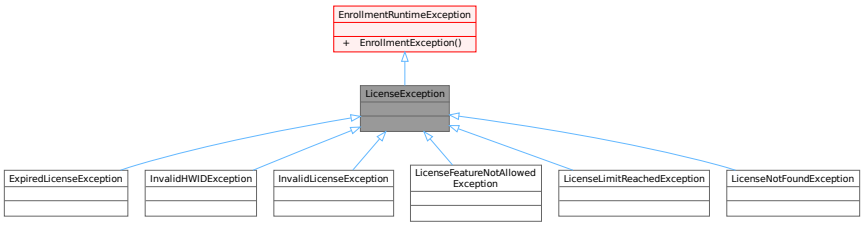
Implements Serializable.

4.130 LicenseException Class Reference

Base exception for all enrollment-sdk license errors.

```
#include <LicenseException.h>
```

Inheritance diagram for LicenseException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentRuntimeException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

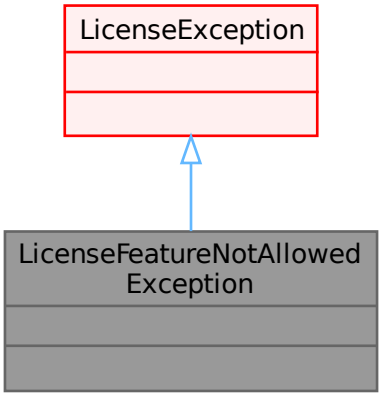
  Provides message with full info about exception.

4.131 LicenseFeatureNotAllowedException Class Reference

The licensed feature is not allowed.

```
#include <LicenseException.h>
```

Inheritance diagram for LicenseFeatureNotAllowedException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentRuntimeException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

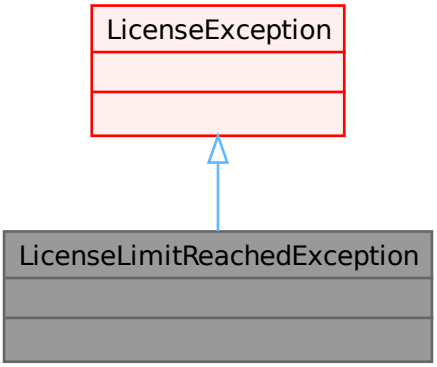
  Provides message with full info about exception.

4.132 LicenseLimitReachedException Class Reference

The limit for operation/count is reached in license.

```
#include <LicenseException.h>
```

Inheritance diagram for LicenseLimitReachedException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentRuntimeException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

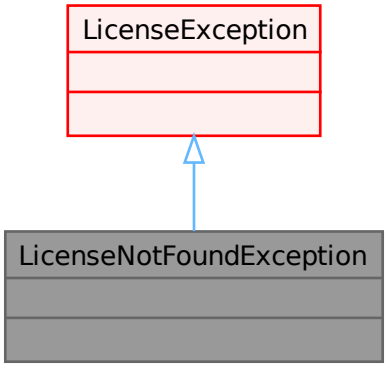
  Provides message with full info about exception.

4.133 LicenseNotFoundException Class Reference

License is not found.

```
#include <LicenseException.h>
```

Inheritance diagram for LicenseNotFoundException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentRuntimeException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

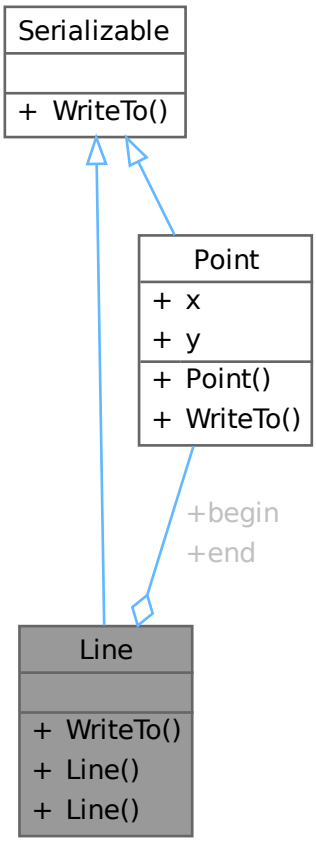
  Provides message with full info about exception.

4.134 Line Class Reference

It holds line points.

`#include <ImageAttribute.h>`

Collaboration diagram for Line:



Public Member Functions

- void WriteTo (Writer &writer) const final

  Write itself into writer.
- Line (Point beginPoint, Point endPoint)

  Constructor store given coordinates.
- Line (Point b, float angle, unsigned int length)

  Constructor calculates end point based on angle and length.

Public Attributes

- Point **begin**

  line begin point
- Point **end**

  line end point

4.134.1 Constructor & Destructor Documentation

4.134.1.1 Line() [1/2]

```
Line (
            Point beginPoint,
            Point endPoint)  [inline]
```

Parameters

beginPoint	line begin point
endPoint	line end point

4.134.1.2 Line() [2/2]

```
Line (
            Point b,
            float angle,
            unsigned int length)  [inline]
```

Parameters

b	line begin
angle	angle of line in degrees, with zero degrees pointing horizontally and to the right and increasing degrees proceeding counter-clockwise.
length	line length

4.134.2 Member Function Documentation

4.134.2.1 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

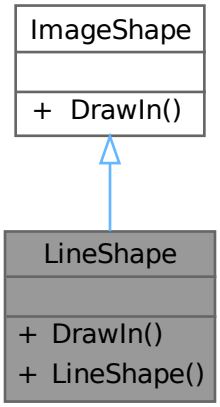
Implements Serializable.

4.135 LineShape Class Reference

It holds line and draws it.

```
#include <ImageShape.h>
```

Collaboration diagram for LineShape:



Public Member Functions

- void DrawIn (ImageShapeDrawer &drawer) const final

  It draws line in ImageShapeDrawer.
- LineShape (Line line, ImageShapeAttributes attr)

  Constructor sets given line for drawing.

4.135.1 Constructor & Destructor Documentation

4.135.1.1 LineShape()

```
LineShape (
            Line line,
            ImageShapeAttributes attr)  [inline]
```

Parameters

line	line that has to be draw
attr	ImageShapeAttributes to draw line

4.135.2 Member Function Documentation

4.135.2.1 DrawIn()

```
void DrawIn (
            ImageShapeDrawer & drawer) const  [inline], [final], [virtual]
```

Parameters

drawer	the implementation of ImageShapeDrawer

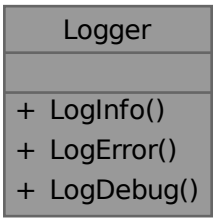
Implements ImageShape.

4.136 Logger Class Reference

Interface for library logging.

```
#include <Logger.h>
```

Collaboration diagram for Logger:



Public Member Functions

- virtual void LogInfo (LogMessage &composer)=0

  It logs info message.
- virtual void LogError (LogMessage &composer)=0

  It logs error message.
- virtual void LogDebug (LogMessage &composer)=0

  It logs debug message.

4.136.1 Detailed Description

For more details see Logging overview.

4.136.2 Member Function Documentation

4.136.2.1 LogDebug()

```
virtual void LogDebug (
            LogMessage & composer)  [pure virtual]
```

Parameters

composer	LogMessage to compose message

Implemented in NullLogger.

4.136.2.2 LogError()

```
virtual void LogError (
            LogMessage & composer)  [pure virtual]
```

Parameters

composer	LogMessage to compose message

Implemented in NullLogger.

4.136.2.3 LogInfo()

```
virtual void LogInfo (
            LogMessage & composer)  [pure virtual]
```

Parameters

composer	LogMessage to compose message

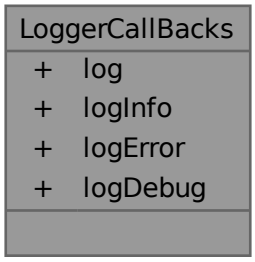
Implemented in NullLogger.

4.137 LoggerCallBacks Struct Reference

LoggerCallBacks contains logger handle and function pointers as callbacks.

```
#include <Logger.h>
```

Collaboration diagram for LoggerCallBacks:



Public Attributes

- void * log

  Logger handle.
- LoggerCallback **logInfo**

  Callback to log info level.
- LoggerCallback **logError**

  Callback to error info level.
- LoggerCallback **logDebug**

  Callback to log debug level.

4.137.1 Detailed Description

Note

Only for C++

4.137.2 Member Data Documentation

4.137.2.1 log

```
void* log
```

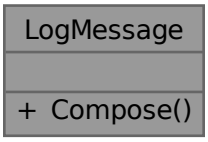
It is not used in binary. Binary provides this handle back to header part of library, that cast it safely to already created Logger class.

4.138 LogMessage Class Reference

Compose Logger message.

```
#include <Logger.h>
```

Collaboration diagram for LogMessage:



Public Member Functions

- virtual const char * Compose ()=0

  It composes message.

4.138.1 Detailed Description

It is used as callback from client code to compose message only when clients really wants to log it. When client does not want to log messages, then it is not necessary to create and not use the message.

4.138.2 Member Function Documentation

4.138.2.1 Compose()

```
virtual const char * Compose () [pure virtual]
```

Returns

composed message

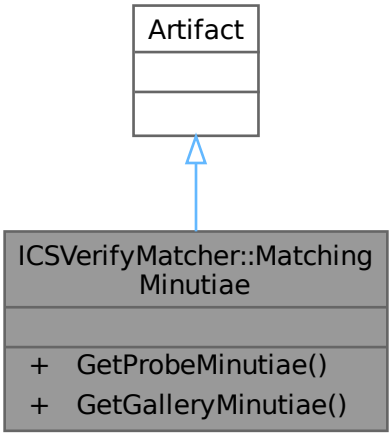
Implemented in FormattingComposer.

4.139 ICSVerifyMatcher::MatchingMinutiae Class Reference

It enables creation of matching minutiae during matching.

```
#include <ICSVerifyMatcher.h>
```

Collaboration diagram for ICSVerifyMatcher::MatchingMinutiae:



Public Member Functions

- std::vector< std::shared_ptr< Minutia > > GetProbeMinutiae () const

  Get probe matching minutiae.
- std::vector< std::shared_ptr< Minutia > > GetGalleryMinutiae () const

  Get gallery matching minutiae.

4.139.1 Detailed Description

The probe and gallery matching minutia has same index in their vectors.

Note

The adjudicator feature MUST be allowed in license to get matching minutiae.

4.139.2 Member Function Documentation

4.139.2.1 GetGalleryMinutiae()

```
std::vector< std::shared_ptr< Minutia > > GetGalleryMinutiae () const [inline]
```

Exceptions

EnrollmentRuntimeException	when matching is in progress

Returns

matching gallery Print Minutia

4.139.2.2 GetProbeMinutiae()

```
std::vector< std::shared_ptr< Minutia > > GetProbeMinutiae () const [inline]
```

Exceptions

EnrollmentRuntimeException	when matching is in progress

Returns

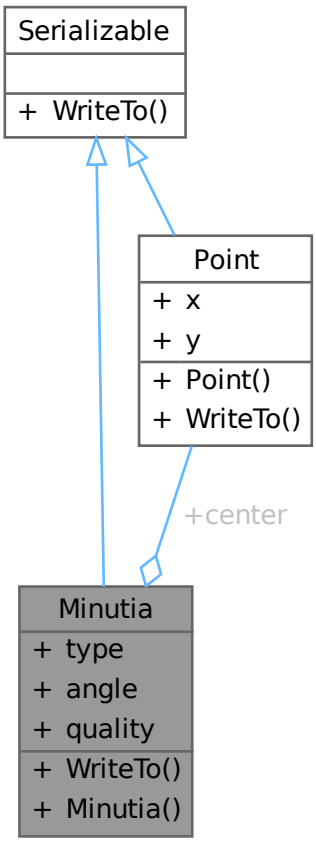
matching probe Print Minutia

4.140 Minutia Class Reference

It represents a particular Minutia (distinctive fingerprint feature point found in fingerprint skeleton, such as a bifurcation or an ending).

```
#include <Minutia.h>
```

Collaboration diagram for Minutia:



Public Types

- enum class Type : uint8_t { BIFURCATION = 0 , RIDGE_ENDING = 1 }

  Type of Minutia.

Public Member Functions

- void WriteTo (Writer &writer) const final

  Write itself into writer.
- Minutia (Point pointArg, Type typeArg, Angle angleArg, uint8_t qualityArg)

  It creates Minutia with given parameters.

Public Attributes

- Point **center**

  Minutia point.
- Type **type**

Minutia type (bifurcation/ending).

- Angle **angle**

  The orientation of the minutia is represented in degrees, with zero degrees pointing horizontally and to the right and increasing degrees proceeding counter-clockwise [9] .

- uint8_t quality

  Minutia quality in range 0-100.

4.140.1 Detailed Description

It is defined by its position (x, y), type (bifurcation or ending), angle (in radians) and quality (0-100).

4.140.2 Member Enumeration Documentation

4.140.2.1 Type

```
enum class Type : uint8_t [strong]
```

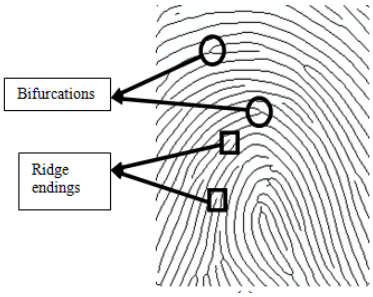


Figure 130 Minutia types

Enumerator

BIFURCATION	Bifurcation Point where a friction ridge forks and becomes two separate ridges.
RIDGE_ENDING	End of ridge.

4.140.3 Constructor & Destructor Documentation

4.140.3.1 Minutia()

```
Minutia (
            Point pointArg,
            Type typeArg,
            Angle angleArg,
            uint8_t qualityArg) [inline]
```

Parameters

pointArg	Minutia point

typeArg	Minutia type (bifurcation/ending)
angleArg	The orientation of the minutia is represented in degrees, with zero degrees pointing horizontally and to the right and increasing degrees proceeding counter-clockwise [9] .
qualityArg	Minutia quality in range 0-100. 0.. undefined, 1 - lowest, 255 - highest

4.140.4 Member Function Documentation

4.140.4.1 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

Implements Serializable.

4.140.5 Member Data Documentation

4.140.5.1 quality

```
uint8_t quality
```

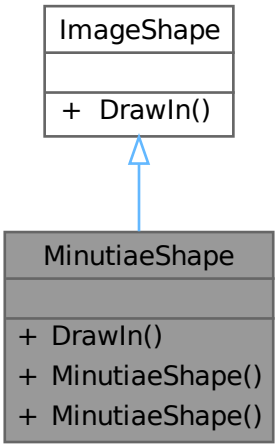
0.. undefined, 1 - lowest, 255 - highest

4.141 MinutiaeShape Class Reference

It draws Minutia into image.

`#include <Minutia.h>`

Collaboration diagram for MinutiaeShape:



Public Member Functions

- void DrawIn (ImageShapeDrawer &drawer) const final

  It draws all Minutia into image.
- MinutiaeShape (const std::vector< std::shared_ptr< Minutia > > &minutia, Radius radius, ImageShapeAttributes attr, unsigned int angleLength)

  Create Minutia shape.
- MinutiaeShape (const std::vector< std::shared_ptr< Minutia > > &minutia, ImageShapeAttributes attr)

  Create Minutia shape.

4.141.1 Constructor & Destructor Documentation

4.141.1.1 MinutiaeShape() [1/2]

```
MinutiaeShape (
            const std::vector< std::shared_ptr< Minutia > > & minutia,
            Radius radius,
            ImageShapeAttributes attr,
            unsigned int angleLength)  [inline]
```

Parameters

minutia	minutia to draw
radius	radius of minutia
attr	ImageShapeAttributes of minutia
angleLength	length of angle line

4.141.1.2 MinutiaeShape() [2/2]

```
MinutiaeShape (
            const std::vector< std::shared_ptr< Minutia > > & minutia,
            ImageShapeAttributes attr)  [inline]
```

Parameters

minutia	minutia to draw
attr	ImageShapeAttributes of minutia

4.141.2 Member Function Documentation

4.141.2.1 DrawIn()

```
void DrawIn (
            ImageShapeDrawer & drawer) const  [inline], [final], [virtual]
```

Parameters

drawer	drawer to draw into

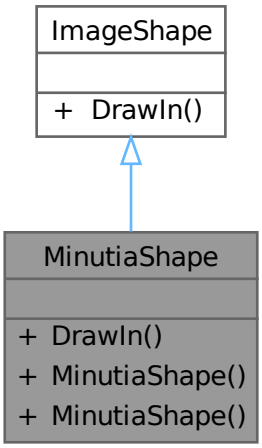
Implements ImageShape.

4.142 MinutiaShape Class Reference

It draws Minutia into image.

`#include <Minutia.h>`

Collaboration diagram for MinutiaShape:



Public Member Functions

- void DrawIn (ImageShapeDrawer &drawer) const final

  It draws Minutia into image.
- MinutiaShape (std::shared_ptr< Minutia > minutia, Radius radius, ImageShapeAttributes attr, unsigned int angleLength)

  Create Minutia shape.
- MinutiaShape (std::shared_ptr< Minutia > minutia, ImageShapeAttributes attr)

  Create Minutia shape.

4.142.1 Detailed Description

The shapes of minutiae are conform with NIST

- The Minutia::Type::BIFURCATION is drawn as square.
- The Minutia::Type::RIDGE_ENDING is drawn as circle.

4.142.2 Constructor & Destructor Documentation

4.142.2.1 MinutiaShape() [1/2]

```
MinutiaShape (
            std::shared_ptr< Minutia > minutia,
            Radius radius,
            ImageShapeAttributes attr,
            unsigned int angleLength)  [inline]
```

Parameters

minutia	minutia to draw
radius	radius of minutia
attr	ImageShapeAttributes of minutia
angleLength	length of angle line

4.142.2.2 MinutiaShape() [2/2]

```
MinutiaShape (
            std::shared_ptr< Minutia > minutia,
            ImageShapeAttributes attr)  [inline]
```

Parameters

minutia	minutia to draw
attr	ImageShapeAttributes of minutia

4.142.3 Member Function Documentation

4.142.3.1 DrawIn()

```
void DrawIn (
            ImageShapeDrawer & drawer) const  [inline], [final], [virtual]
```

Parameters

drawer	drawer to draw into

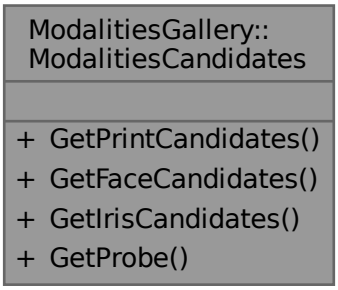
Implements ImageShape.

4.143 ModalitiesGallery::ModalitiesCandidates Class Reference

Contains candidates per each modality.

```
#include <ModalitiesGallery.h>
```

Collaboration diagram for ModalitiesGallery::ModalitiesCandidates:



Public Member Functions

- std::vector< std::shared_ptr< ModalitiesGalleryReference > > GetPrintCandidates () const

  Returns best candidates for print modality.
- std::vector< std::shared_ptr< ModalitiesGalleryReference > > GetFaceCandidates () const

  Returns best candidates for face modality.
- std::vector< std::shared_ptr< ModalitiesGalleryReference > > GetIrisCandidates () const

  Returns best candidates for iris modality.
- std::shared_ptr< ModalitiesTemplate > GetProbe () const

  Returns probe for which candidates are searched.

4.143.1 Member Function Documentation

4.143.1.1 GetFaceCandidates()

```
std::vector< std::shared_ptr< ModalitiesGalleryReference > > GetFaceCandidates () const
[inline], [nodiscard]
```

It will return best candidates for face modality in descending order.

Returns

identified candidates

4.143.1.2 GetIrisCandidates()

```
std::vector< std::shared_ptr< ModalitiesGalleryReference > > GetIrisCandidates () const
[inline], [nodiscard]
```

It will return best candidates for iris modality in descending order.

Returns

identified candidates

4.143.1.3 GetPrintCandidates()

```
std::vector< std::shared_ptr< ModalitiesGalleryReference > > GetPrintCandidates () const
[inline], [nodiscard]
```

It will return best candidates for print modality in descending order.

Returns

identified candidates

4.143.1.4 GetProbe()

```
std::shared_ptr< ModalitiesTemplate > GetProbe () const  [inline], [nodiscard]
```

Returns

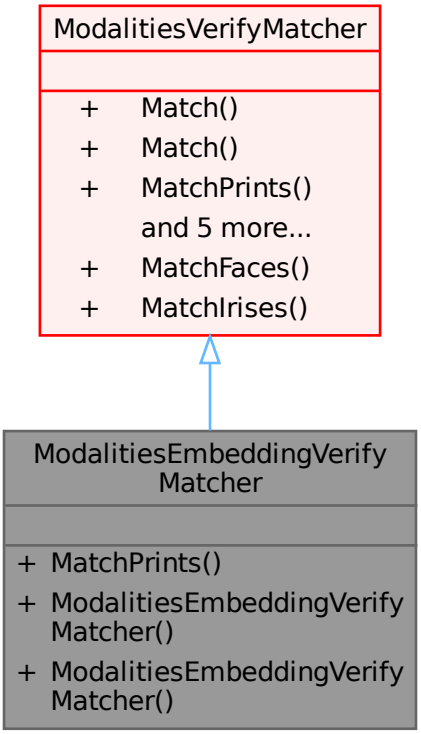
probe

4.144 ModalitiesEmbeddingVerifyMatcher Class Reference

ModalitiesEmbeddingVerifyMatcher is template matcher that uses Innovatrics proprietary verification algorithm.

```
#include <ModalitiesEmbeddingVerifyMatcher.h>
```

Collaboration diagram for ModalitiesEmbeddingVerifyMatcher:



Public Member Functions

- PrintSimilarityScore MatchPrints (const std::vector< std::shared_ptr< PrintTemplate > > &probe, const std::vector< std::shared_ptr< PrintTemplate > > &gallery) const final

  Calculates prints similarity score of probe and gallery that contains only embedding prints.
- ModalitiesEmbeddingVerifyMatcher ()

  Constructor with default logger and executor GlobalMultiModalityExecutor is used to execute matching.
- ModalitiesEmbeddingVerifyMatcher (std::shared_ptr< MultiModalityExecutor > executor)

  Constructor with custom executor.

Public Member Functions inherited from ModalitiesVerifyMatcher

- ModalityScore Match (const Applicant &probe, const Applicant &gallery) const final

  Calculates similarity score of probe and gallery applicant template for each modality.
- ModalityScore Match (const ModalitiesTemplate &probe, const ModalitiesTemplate &gallery) const final

  Calculates similarity score of probe and gallery applicant for each applicant modality.
- **ModalitiesVerifyMatcher** ()

  Constructor with default Config GlobalMultiModalityExecutor is used to execute matching.
- ModalitiesVerifyMatcher (Config config)

  Constructor uses Config to setup matching algorithm.

- ModalitiesVerifyMatcher (Config config, std::shared_ptr< MultiModalityExecutor > executor)

  Constructor uses Config to setup matching algorithm.
- ModalitiesVerifyMatcher (Config config, std::shared_ptr< MultiModalityExecutor > executor, std::shared_ptr< Logger > logger)

  Constructor uses Config to setup matching algorithm.
- ModalitiesVerifyMatcher (std::shared_ptr< MultiModalityExecutor > executor)

  Constructor with default Config.

Additional Inherited Members

Static Public Member Functions inherited from ModalitiesVerifyMatcher

- static FaceSimilarityScore MatchFaces (const std::vector< std::shared_ptr< FaceTemplate > > &probe, const std::vector< std::shared_ptr< FaceTemplate > > &gallery)

  Calculates face similarity score of probe and gallery that contains only faces.
- static IrisSimilarityScore MatchIrises (const std::vector< std::shared_ptr< IrisTemplate > > &probe, const std::vector< std::shared_ptr< IrisTemplate > > &gallery)

  Calculates face similarity score of probe and gallery that contains only irises.

4.144.1 Detailed Description

Note

Not supported in this variant of Enrollment SDK.

4.144.2 Constructor & Destructor Documentation

4.144.2.1 ModalitiesEmbeddingVerifyMatcher() [1/2]

```
ModalitiesEmbeddingVerifyMatcher () [inline]
```

Note

Not supported in this variant of Enrollment SDK.

4.144.2.2 ModalitiesEmbeddingVerifyMatcher() [2/2]

```
ModalitiesEmbeddingVerifyMatcher (
            std::shared_ptr< MultiModalityExecutor > executor) [inline], [explicit]
```

Note

Not supported in this variant of Enrollment SDK.

Parameters

executor	MultiModalityExecutor used for matching

Exceptions

NullArgumentException	when executor is null

4.144.3 Member Function Documentation

4.144.3.1 MatchPrints()

```
PrintSimilarityScore MatchPrints (
            const std::vector< std::shared_ptr< PrintTemplate > > & probe,
            const std::vector< std::shared_ptr< PrintTemplate > > & gallery) const  [inline],
[nodiscard], [final], [virtual]
```

Note

Not supported in this variant of Enrollment SDK.

The embedding extraction feature MUST be allowed in license to match print embeddings.

Parameters

probe	probe prints
gallery	gallery prints

Returns

similarity PrintSimilarityScore

Exceptions

EnrollmentException	when it is not possible to match templates

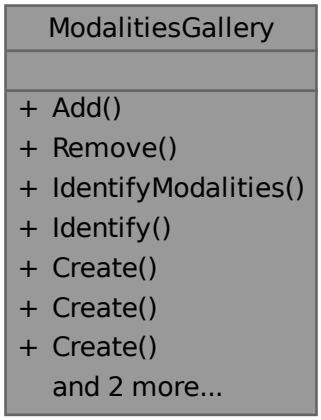
Reimplemented from ModalitiesVerifyMatcher.

4.145 ModalitiesGallery Class Reference

Gallery of references.

```
#include <ModalitiesGallery.h>
```

Collaboration diagram for ModalitiesGallery:



Classes

- class ModalitiesCandidates

  Contains candidates per each modality.
- class Candidate

  Found candidate.

Public Member Functions

- void Add (std::shared_ptr< ModalitiesGalleryReference > reference)

  Adds ModalitiesGalleryReference into gallery.
- void Remove (size_t index)

  Removes the ModalitiesGalleryReference stored at the specified index from the gallery.
- std::shared_ptr< ModalitiesCandidates > IdentifyModalities (const std::shared_ptr< ModalitiesTemplate > &probe, size_t maxCandidates) const

  Search for given probe.
- std::vector< Candidate > Identify (const std::shared_ptr< ModalitiesTemplate > &probe, size_t max↩ Candidates) const

  Search for given probe.

Static Public Member Functions

- static std::shared_ptr< ModalitiesGallery > Create ()

  Create gallery of references.
- static std::shared_ptr< ModalitiesGallery > Create (ModalitiesGalleryType type)

  Create gallery of references.
- static std::shared_ptr< ModalitiesGallery > Create (const ModalitiesGalleryConfig &config)

  Create gallery of references.
- static std::vector< Candidate > Consolidate (const std::shared_ptr< ModalitiesCandidates > &identified↩ ModalityCandidates, size_t maxCandidates)

  Merges previously identified modality candidates into list of overall best candidates.
- static std::vector< Candidate > Consolidate (const std::shared_ptr< ModalitiesCandidates > &identified↩ ModalityCandidates, size_t maxCandidates, const ModalitiesMatcher &matcher)

  Merges previously identified modality candidates into list of overall best candidates.

4.145.1 Detailed Description

Note

The number of references in any ModalitiesGallery is limited by license. You can have more created galleries in your code. The number limits overall references in all galleries.

The ModalitiesGallery stores ModalitiesGalleryReference objects for later identification. Adding data to this gallery in higher-level language bindings is less efficient than in IndexedModalitiesGallery. This inefficiency arises because it creates objects derived from C++ interfaces while also maintaining references to the original high-level language objects.

These high-level language references are necessary because objects returned from the C++ implementation lose their original type information. To address this, the gallery remaps types based on the ID from ModalitiesGalleryReference. However, this approach consumes more memory, as it stores both the identification related data and a reference to the user object.

Despite the increased memory usage, this gallery is more convenient to use as IndexedModalitiesGallery. Since the entire user object is stored within the gallery, users do not need to manually map objects to their identifiers in their code or reload ModalitiesTemplate for consolidation. However, users should be mindful of their object size. The object itself is not required for IdentifyModalities, only the Index and ModalitiesTemplate provided by ModalitiesGalleryReference are used. The relevant identification data is copied from ModalitiesTemplate.

Once a ModalitiesGalleryReference is added to the gallery, its associated ModalitiesTemplate is no longer relevant until consolidation. If memory usage is a critical concern, users can implement ModalitiesGalleryReference::GetModalitiesTemplates() in such a way that the function loads the template on each call instead of storing the object in memory or use IndexedModalitiesGallery.

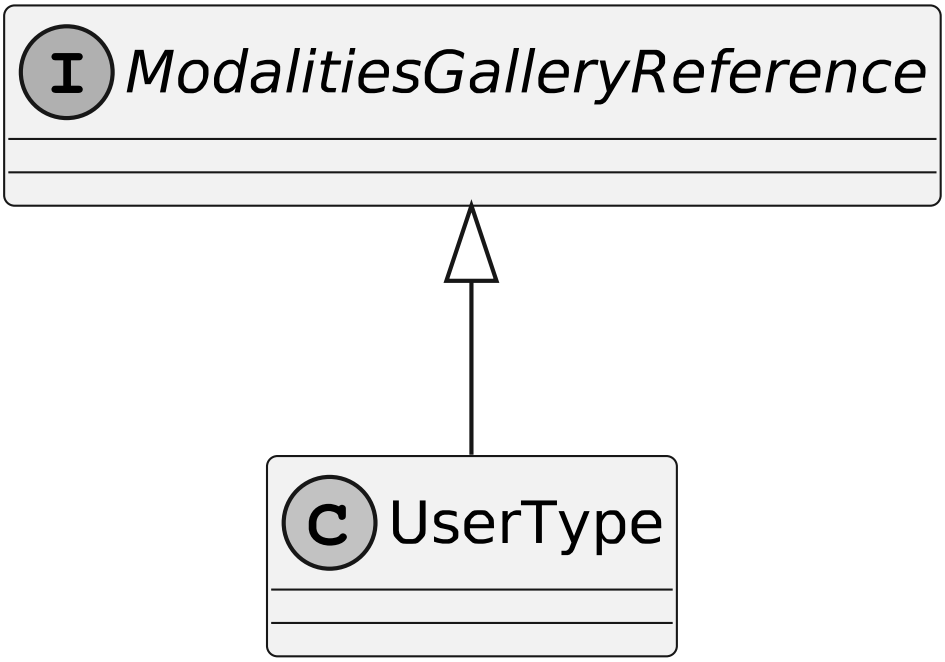
Each ModalitiesGalleryReference contains a ModalitiesTemplate used for identification. The ModalitiesGallery::Identify() function returns this reference to the SDK user, who can then downcast it to their custom type to retrieve additional information.

Possible approaches for adding custom types to the gallery include:

- **Inheritance**: Extend ModalitiesGalleryReference to include custom data.
- **Composition**: Store a reference to a custom object within ModalitiesGalleryReference.

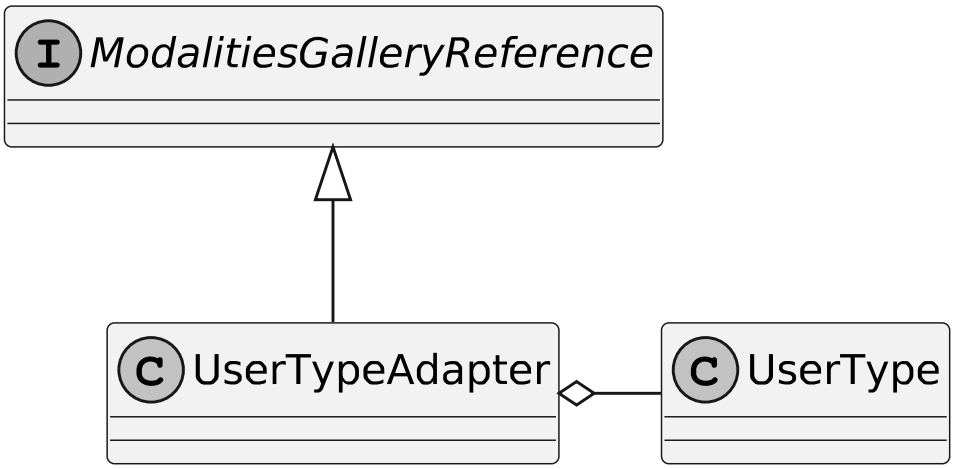
Inheritance

You can inherit our ModalitiesGalleryReference in your types.



Composition

You can embed your type into adapter of ModalitiesGalleryReference.



The identification (ModalitiesGallery::Identify()) has two phases

- searching in gallery for best candidates per modality (details in ModalitiesGallery::IdentifyModalities()).
- merging lists of candidates per modality into final candidates list (details in ModalitiesGallery::Consolidate()).

The enrollment SDK provides following types of gallery

- Print identification based on ICSTemplate

c++

```
// Recreate your own data types from data stored during enrollment
auto johnTemplate =
  ModalitiesTemplate::Create(BinaryFile::ReadAll("output/john.tmpl"));
auto john = std::make_shared<UserType>(0, "john", johnTemplate);
auto markTemplate =
  ModalitiesTemplate::Create(BinaryFile::ReadAll("output/mark.tmpl"));
auto mark = std::make_shared<UserType>(1, "mark", markTemplate);

// Create a gallery and add users.
auto gallery = ModalitiesGallery::Create();
gallery->Add(john);
gallery->Add(mark);

// Identify a probe in the gallery.
constexpr size_t MaxCandidates = 2; // size of result set of candidates
auto result = gallery->Identify(johnTemplate, MaxCandidates);

// Output identification results.
for (const auto& c : result)
{
    auto user = std::dynamic_pointer_cast<UserType>(c.GetCandidate());
    std::cout « "   Name is " « user->name « " score is ["
              « c.GetScore().GetPrintScore() « ", "
              « c.GetScore().GetFaceScore() « ", "
              « c.GetScore().GetIrisScore() « "]" « std::endl;
}
```

java

```
// Recreate your own data types from data stored during enrollment.
ModalitiesTemplate johnTemplate = ModalitiesTemplate.Create(
        BinaryFile.ReadAll("output/john.tmpl"));
UserType john = new UserType(0, "john", johnTemplate);
ModalitiesTemplate markTemplate = ModalitiesTemplate.Create(
        BinaryFile.ReadAll("output/mark.tmpl"));
UserType mark = new UserType(1, "mark", markTemplate);

// Create a gallery and add users.
ModalitiesGallery gallery = ModalitiesGallery.Create();
gallery.Add(john);
gallery.Add(mark);

// Identify a probe in the gallery.
long maxCandidates = 2; // size of result set of candidates
ArrayList<ModalitiesGallery.Candidate> result = gallery.Identify(
        johnTemplate, maxCandidates);

// Output identification results.
for (ModalitiesGallery.Candidate c : result) {
    UserType user = (UserType) c.GetCandidate();
    System.out.println("   Name is " + user.name + " score is ["
                       + c.GetScore().GetPrintScore() + ", "
                       + c.GetScore().GetFaceScore() + ", "
                       + c.GetScore().GetIrisScore() + "]");
}
```

csharp

```
// Recreate your own data types from data stored during enrollment
var johnTemplate =
    ModalitiesTemplate.Create(BinaryFile.ReadAll("output/john.tmpl"));
var john = new UserType(0, "john", johnTemplate);
var markTemplate =
    ModalitiesTemplate.Create(BinaryFile.ReadAll("output/mark.tmpl"));
var mark = new UserType(1, "mark", markTemplate);

// Create a gallery and add users.
var gallery = ModalitiesGallery.Create();
gallery.Add(john);
gallery.Add(mark);

// Identify a probe in the gallery.
var maxCandidates = 2U; // size of result set of candidates
var result = gallery.Identify(johnTemplate, maxCandidates);

// Output identification results.
foreach (ModalitiesGallery.Candidate c in result)
{
    var user = c.GetCandidate() as UserType;
    System.Console.WriteLine("   Name is " + user.name + " score is [" +
                             c.GetScore().GetPrintScore() + ", " +
                             c.GetScore().GetFaceScore() + ", " +
                             c.GetScore().GetIrisScore() + "]");
}
```

4.145.2 Member Function Documentation

4.145.2.1 Add()

```
void Add (
            std::shared_ptr< ModalitiesGalleryReference > reference)  [inline]
```

The reference contains ModalitiesTemplate used for identification.

Parameters

reference	ModalitiesGalleryReference to add

Exceptions

EnrollmentException	when it is not possible to add reference
NullArgumentException	when reference is null
NullArgumentException	when reference template is null
EnrollmentInvalidArgumentException	when at least one probe print has position different as PrintPosition::LEFT_INDEX, PrintPosition::LEFT_LITTLE, PrintPosition::LEFT_MIDDLE, PrintPosition::LEFT_THUMB, PrintPosition::LEFT_RING, PrintPosition::RIGHT_INDEX, PrintPosition::RIGHT_LITTLE, PrintPosition::RIGHT_MIDDLE, PrintPosition::RIGHT_THUMB, PrintPosition::RIGHT_RING or no position.
NullArgumentException	when some print in probe prints is null.
NullArgumentException	when some face in probe faces is null.
NullArgumentException	when some iris in probe irises is null.

4.145.2.2 Consolidate() [1/2]

```
std::vector< Candidate > Consolidate (
            const std::shared_ptr< ModalitiesCandidates > & identifiedModalityCandidates,
            size_t maxCandidates)  [inline], [static]
```

It uses ModalitiesVerifyMatcher with default ModalitiesVerifyMatcher::Config for merging identified candidates in gallery.

It creates list of unique candidates and verifies each of them with probe stored in ModalitiesCandidates. It uses 1 to 1 Modalities Verification. Once modalities scores are calculated it sorts candidates according to maximal score of all modalities scores.

Parameters

identifiedModalityCandidates	modality candidates identified by IdentifyModalities().
maxCandidates	maximum candidates to return

Returns

merged candidates

4.145.2.3 Consolidate() [2/2]

```
std::vector< Candidate > Consolidate (
            const std::shared_ptr< ModalitiesCandidates > & identifiedModalityCandidates,
            size_t maxCandidates,
            const ModalitiesMatcher & matcher)  [inline], [static]
```

It create list of unique candidates and verify each of them with probe stored in ModalitiesCandidates. It uses 1 to 1 Modalities Verification. Once modalities scores are calculated it sorts candidates according to maximal score of all modalities scores.

Parameters

identifiedModalityCandidates	modality candidates identified by IdentifyModalities().
maxCandidates	maximum candidates to return
matcher	ModalitiesMatcher to make multi modal verification

Returns

merged candidates

4.145.2.4 Create() [1/3]

```
std::shared_ptr< ModalitiesGallery > Create ()  [inline], [static]
```

It will create gallery with default configuration.

Returns

created gallery

Exceptions

EnrollmentException	when it is not possible to create gallery

4.145.2.5 Create() [2/3]

```
std::shared_ptr< ModalitiesGallery > Create (
            const ModalitiesGalleryConfig & config)  [inline], [static]
```

Parameters

config	Config of gallery

Returns

created gallery

Exceptions

EnrollmentException	when it is not possible to create gallery

4.145.2.6 Create() [3/3]

```
std::shared_ptr< ModalitiesGallery > Create (
            ModalitiesGalleryType type)  [inline], [static]
```

Parameters

type	ModalitiesGalleryType of gallery

Returns

created gallery

Exceptions

EnrollmentException	when it is not possible to create gallery

4.145.2.7 Identify()

```
std::vector< Candidate > Identify (
            const std::shared_ptr< ModalitiesTemplate > & probe,
            size_t maxCandidates) const  [inline], [nodiscard]
```

It uses ModalitiesVerifyMatcher with default ModalitiesVerifyMatcher::Config for merging identified candidates in gallery.

The identification has two phases

- searching in gallery for best candidates per modality (details in IdentifyModalities()).
- merging lists of candidates per modality into final candidates list (details in Consolidate()).

Parameters

probe	probe that is searched in gallery
maxCandidates	maximum number of returned identified candidates

Returns

identified candidates

Exceptions

EnrollmentException	when it is not possible to identify probe
NullArgumentException	when probe is null
NullArgumentException	when some print in probe prints is null.
NullArgumentException	when some face in probe faces is null.
NullArgumentException	when some iris in probe irises is null.

4.145.2.8 IdentifyModalities()

```
std::shared_ptr< ModalitiesCandidates > IdentifyModalities (
            const std::shared_ptr< ModalitiesTemplate > & probe,
            size_t maxCandidates) const  [inline], [nodiscard]
```

Parameters

probe	probe that is searched in gallery
maxCandidates	maximum number of returned identified candidates

Returns

identified candidates per modalities without score. To get scores and unique candidates list use function Consolidate().

Exceptions

EnrollmentException	when it is not possible to identify probe
NullArgumentException	when probe is null
NullArgumentException	when some print in probe prints is null.
NullArgumentException	when some face in probe faces is null.
NullArgumentException	when some iris in probe irises is null.

4.145.2.9 Remove()

```
void Remove (
            size_t index)  [inline]
```

The provided `index` MUST match the one returned by ModalitiesGalleryReference::GetIndex() when the reference was originally added to the gallery.

Parameters

index	The reference index to remove.

Exceptions

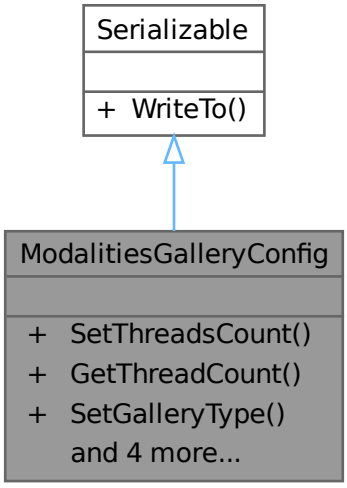
EnrollmentException	If the provided index does not exist in the gallery.

4.146 ModalitiesGalleryConfig Class Reference

ModalitiesGallery configuration.

`#include <ModalitiesGallery.h>`

Collaboration diagram for ModalitiesGalleryConfig:



Public Member Functions

- void SetThreadsCount (unsigned int value)

  Set number of threads used by gallery.
- unsigned int GetThreadCount () const

  Get number of processing threads.
- void SetGalleryType (ModalitiesGalleryType value)

  Set type of Modalities Gallery.
- ModalitiesGalleryType GetGalleryType () const

  Get type of Modalities Gallery.
- void WriteTo (Writer &writer) const final

  Function serializes configuration via provided Writer.
- IndexedModalitiesGalleryConfig ToIndexedModalitiesGalleryConfig () const

  Function converts configuration to IndexedModalitiesGalleryConfig.
- ModalitiesGalleryConfig ()=default

  Default configuration is.

4.146.1 Constructor & Destructor Documentation

4.146.1.1 ModalitiesGalleryConfig()

```
ModalitiesGalleryConfig () [default]
```

- identification threads - 1 thread
- gallery type - PRINT_ICS_TEMPLATE

4.146.2 Member Function Documentation

4.146.2.1 GetGalleryType()

```
ModalitiesGalleryType GetGalleryType () const [inline], [nodiscard]
```

Default type is PRINT_ICS_TEMPLATE.

Returns

type of gallery

4.146.2.2 GetThreadCount()

```
unsigned int GetThreadCount () const [inline], [nodiscard]
```

Returns

number of threads

4.146.2.3 SetGalleryType()

```
void SetGalleryType (
            ModalitiesGalleryType value) [inline]
```

Default type is PRINT_ICS_TEMPLATE.

Parameters

value	from the enum ModalitiesGalleryType

Exceptions

EnrollmentInvalidArgumentException	when value is not in range of the enum ModalitiesGalleryType

4.146.2.4 SetThreadsCount()

```
void SetThreadsCount (
            unsigned int value)  [inline]
```

Note

The 'multithreaded-galleries' feature MUST be allowed in license to allow using threads. Default value is 1.

Parameters

value	it MUST be greater as 0

Exceptions

EnrollmentInvalidArgumentException	when value is 0

4.146.2.5 ToIndexedModalitiesGalleryConfig()

```
IndexedModalitiesGalleryConfig ToIndexedModalitiesGalleryConfig () const  [inline], [nodiscard]
```

Returns

IndexedModalitiesGalleryConfig

4.146.2.6 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Config.

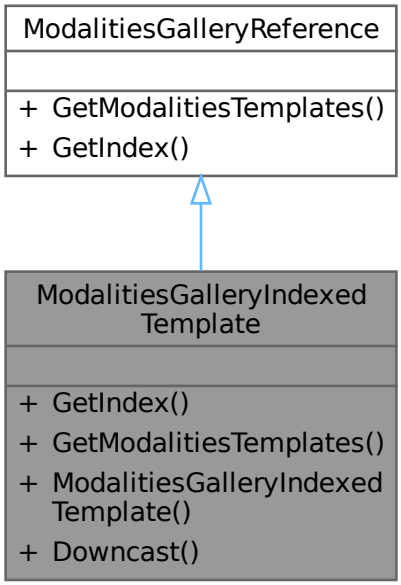
Implements Serializable.

4.147 ModalitiesGalleryIndexedTemplate Class Reference

Modalities gallery reference with number identifier and template.

```
#include <ModalitiesGallery.h>
```

Collaboration diagram for ModalitiesGalleryIndexedTemplate:



Public Member Functions

- size_t GetIndex () const final

  Provides reference index.
- std::shared_ptr< ModalitiesTemplate > GetModalitiesTemplates () const final

  Provides reference ModalitiesTemplate.
- ModalitiesGalleryIndexedTemplate (size_t index, std::shared_ptr< ModalitiesTemplate > reference)

  Create indexed reference with given index and template.

Static Public Member Functions

- static std::shared_ptr< ModalitiesGalleryIndexedTemplate > Downcast (std::shared_ptr< ModalitiesGalleryReference > &reference)

  Downcasts ModalitiesGalleryIndexedTemplate interface into indexed reference.

4.147.1 Constructor & Destructor Documentation

4.147.1.1 ModalitiesGalleryIndexedTemplate()

```
ModalitiesGalleryIndexedTemplate (
            size_t index,
            std::shared_ptr< ModalitiesTemplate > reference)  [inline]
```

Parameters

index	index to store
reference	reference ModalitiesTemplate

4.147.2 Member Function Documentation

4.147.2.1 Downcast()

```
std::shared_ptr< ModalitiesGalleryIndexedTemplate > Downcast (
            std::shared_ptr< ModalitiesGalleryReference > & reference)  [inline], [static]
```

Parameters

reference	to be downcasted

Returns

ModalitiesGalleryIndexedReference

4.147.2.2 GetIndex()

```
size_t GetIndex () const  [inline], [nodiscard], [final], [virtual]
```

Returns

reference index

Implements ModalitiesGalleryReference.

4.147.2.3 GetModalitiesTemplates()

```
std::shared_ptr< ModalitiesTemplate > GetModalitiesTemplates () const  [inline], [nodiscard],
[final], [virtual]
```

Returns

reference ModalitiesTemplate

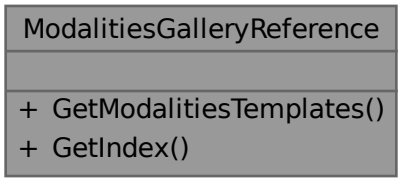
Implements ModalitiesGalleryReference.

4.148 ModalitiesGalleryReference Class Reference

The reference interface for ModalitiesGallery.

```
#include <ModalitiesGallery.h>
```

Collaboration diagram for ModalitiesGalleryReference:



Public Member Functions

- virtual std::shared_ptr< ModalitiesTemplate > GetModalitiesTemplates () const =0

  Provides reference ModalitiesTemplate.
- virtual size_t GetIndex () const =0

  Provides unique reference index.

4.148.1 Member Function Documentation

4.148.1.1 GetIndex()

```
virtual size_t GetIndex () const  [nodiscard], [pure virtual]
```

Returns

reference index

Implemented in ModalitiesGalleryIndexedTemplate.

4.148.1.2 GetModalitiesTemplates()

```
virtual std::shared_ptr< ModalitiesTemplate > GetModalitiesTemplates () const  [nodiscard],
[pure virtual]
```

Returns

reference ModalitiesTemplate

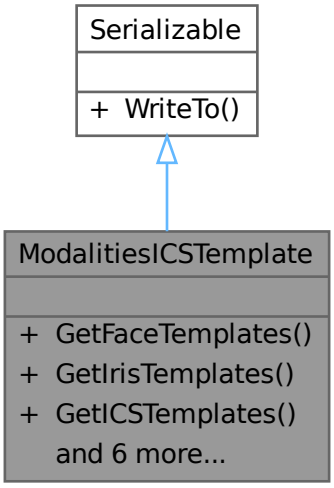
Implemented in ModalitiesGalleryIndexedTemplate.

4.149 ModalitiesICSTemplate Class Reference

Applicant template in ICS format used for biometric operations.

```
#include <ModalitiesICSTemplate.h>
```

Collaboration diagram for ModalitiesICSTemplate:



Public Member Functions

- std::vector< std::shared_ptr< FaceTemplate > > GetFaceTemplates () const

  Provides FaceTemplate objects.
- std::vector< std::shared_ptr< IrisTemplate > > GetIrisTemplates () const

  Provides IrisTemplate objects.
- std::vector< std::shared_ptr< ICSTemplate > > GetICSTemplates () const

  Provides PrintTemplate objects.
- std::vector< uint8_t > GetModalitiesICSTemplateData () const

  It provides ICS template data.
- ModalitiesICSTemplate (std::vector< uint8_t > data)

  Create ModalitiesICSTemplate object It uses default executor.
- ModalitiesICSTemplate (std::vector< uint8_t > data, std::shared_ptr< MultiModalityExecutor > executor)

  Create ModalitiesICSTemplate object.
- ModalitiesICSTemplate (std::vector< std::shared_ptr< ICSTemplate > > icsTemplates, std::vector< std::↩ shared_ptr< FaceTemplate > > faceTemplates, std::vector< std::shared_ptr< IrisTemplate > > irisTemplates)

  Create ModalitiesICSTemplate object It uses default executor.
- ModalitiesICSTemplate (std::vector< std::shared_ptr< ICSTemplate > > icsTemplates, std::vector< std::↩ shared_ptr< FaceTemplate > > faceTemplates, std::vector< std::shared_ptr< IrisTemplate > > irisTemplates, std::shared_ptr< MultiModalityExecutor > executor)

  Create ModalitiesICSTemplate object.
- void WriteTo (Writer &w) const override

  Write itself into writer.

4.149.1 Detailed Description

The template contains ICS templates for print, face, iris modalities.

4.149.2 Constructor & Destructor Documentation

4.149.2.1 ModalitiesICSTemplate() [1/4]

```
ModalitiesICSTemplate (
            std::vector< uint8_t > data)  [inline], [explicit]
```

Parameters

data	template data

Exceptions

NullArgumentException	when executor is null

4.149.2.2 ModalitiesICSTemplate() [2/4]

```
ModalitiesICSTemplate (
            std::vector< uint8_t > data,
            std::shared_ptr< MultiModalityExecutor > executor)  [inline], [explicit]
```

Parameters

data	template data
executor	MultiModalityExecutor used for matching

Exceptions

NullArgumentException	when executor is null

4.149.2.3 ModalitiesICSTemplate() [3/4]

```
ModalitiesICSTemplate (
            std::vector< std::shared_ptr< ICSTemplate > > icsTemplates,
            std::vector< std::shared_ptr< FaceTemplate > > faceTemplates,
            std::vector< std::shared_ptr< IrisTemplate > > irisTemplates)  [inline]
```

Parameters

icsTemplates	ics templates

faceTemplates	face templates
irisTemplates	iris templates

4.149.2.4 ModalitiesICSTemplate() [4/4]

```
ModalitiesICSTemplate (
            std::vector< std::shared_ptr< ICSTemplate > > icsTemplates,
            std::vector< std::shared_ptr< FaceTemplate > > faceTemplates,
            std::vector< std::shared_ptr< IrisTemplate > > irisTemplates,
            std::shared_ptr< MultiModalityExecutor > executor)  [inline]
```

Parameters

icsTemplates	ics templates
faceTemplates	face templates
irisTemplates	iris templates
executor	MultiModalityExecutor used for matching

4.149.3 Member Function Documentation

4.149.3.1 GetFaceTemplates()

```
std::vector< std::shared_ptr< FaceTemplate > > GetFaceTemplates () const  [inline],
[nodiscard]
```

Returns

face templates

4.149.3.2 GetICSTemplates()

```
std::vector< std::shared_ptr< ICSTemplate > > GetICSTemplates () const  [inline], [nodiscard]
```

Returns

print templates

4.149.3.3 GetIrisTemplates()

```
std::vector< std::shared_ptr< IrisTemplate > > GetIrisTemplates () const  [inline],
[nodiscard]
```

Returns

iris templates

4.149.3.4 GetModalitiesICSTemplateData()

```
std::vector< uint8_t > GetModalitiesICSTemplateData () const  [inline], [nodiscard]
```

Returns

all modalities merged to ICS template (IEngine user record)

4.149.3.5 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [override], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

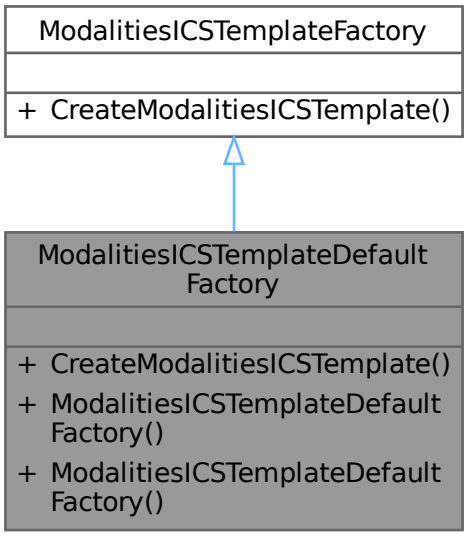
Implements Serializable.

4.150 ModalitiesICSTemplateDefaultFactory Class Reference

ModalitiesICSTemplateDefaultFactory creates biometric representations from all modality templates.

```
#include <ModalitiesICSTemplateDefaultFactory.h>
```

Collaboration diagram for ModalitiesICSTemplateDefaultFactory:



Public Member Functions

- std::shared_ptr< ModalitiesICSTemplate > CreateModalitiesICSTemplate (const std::vector< std::shared_ptr< ICSTemplate > > &printTemplates, const std::vector< std::shared_ptr< FaceTemplate > > &faceTemplates, const std::vector< std::shared_ptr< IrisTemplate > > &irisTemplates) const final

  It creates ModalitiesICSTemplate.
- ModalitiesICSTemplateDefaultFactory ()

  It constructs new ModalitiesICSTemplateDefaultFactory object.
- ModalitiesICSTemplateDefaultFactory (std::shared_ptr< MultiModalityExecutor > executor)

  It constructs new ModalitiesICSTemplateDefaultFactory object.

4.150.1 Detailed Description

It will convert all print, face, iris templates into one ModalitiesICSTemplate. The biometric representation (template) is used in biometric functions e.g. Verification. It contains no other Applicant data such as images, attributes and so on.

4.150.2 Constructor & Destructor Documentation

4.150.2.1 ModalitiesICSTemplateDefaultFactory() [1/2]

```
ModalitiesICSTemplateDefaultFactory () [inline]
```

GlobalMultiModalityExecutor is used to execute matching.

4.150.2.2 ModalitiesICSTemplateDefaultFactory() [2/2]

```
ModalitiesICSTemplateDefaultFactory (
            std::shared_ptr< MultiModalityExecutor > executor) [inline], [explicit]
```

Parameters

executor	MultiModalityExecutor used for matching

Exceptions

NullArgumentException	when executor is null

4.150.3 Member Function Documentation

4.150.3.1 CreateModalitiesICSTemplate()

```
std::shared_ptr< ModalitiesICSTemplate > CreateModalitiesICSTemplate (
            const std::vector< std::shared_ptr< ICSTemplate > > & printTemplates,
            const std::vector< std::shared_ptr< FaceTemplate > > & faceTemplates,
            const std::vector< std::shared_ptr< IrisTemplate > > & irisTemplates) const
[inline], [nodiscard], [final], [virtual]
```

Parameters

printTemplates	print templates to add to modalities ics template.
faceTemplates	face templates to add to modalities ics template.
irisTemplates	iris templates to add to modalities ics template.

Returns

created ModalitiesICSTemplate

Exceptions

EnrollmentException	when it is not possible to create template

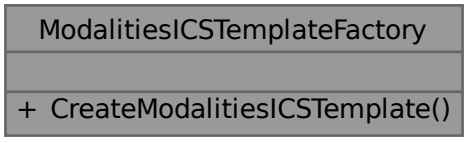
Implements ModalitiesICSTemplateFactory.

4.151 ModalitiesICSTemplateFactory Class Reference

ModalitiesICSTemplateFactory creates biometric representations from all modality templates.

```
#include <ModalitiesICSTemplateFactory.h>
```

Collaboration diagram for ModalitiesICSTemplateFactory:



Public Member Functions

- virtual std::shared_ptr< ModalitiesICSTemplate > CreateModalitiesICSTemplate (const std::vector< std::↩ shared_ptr< ICSTemplate > > &printTemplates, const std::vector< std::shared_ptr< FaceTemplate > > &faceTemplates, const std::vector< std::shared_ptr< IrisTemplate > > &irisTemplates) const =0

  It creates ModalitiesICSTemplate.

4.151.1 Detailed Description

The biometric representation (template) is used in biometric functions e.g. Verification.

4.151.2 Member Function Documentation

4.151.2.1 CreateModalitiesICSTemplate()

```
virtual std::shared_ptr< ModalitiesICSTemplate > CreateModalitiesICSTemplate (
            const std::vector< std::shared_ptr< ICSTemplate > > & printTemplates,
            const std::vector< std::shared_ptr< FaceTemplate > > & faceTemplates,
            const std::vector< std::shared_ptr< IrisTemplate > > & irisTemplates) const
[nodiscard], [pure virtual]
```

Parameters

printTemplates	print templates to add to modalities ics template.
faceTemplates	face templates to add to modalities ics template.
irisTemplates	iris templates to add to modalities ics template.

Returns

created ModalitiesICSTemplate

Exceptions

EnrollmentException	when it is not possible to create template

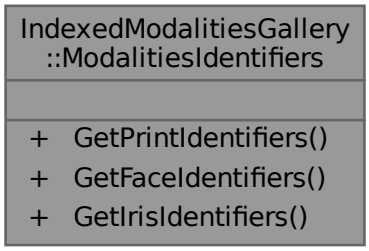
Implemented in ModalitiesICSTemplateDefaultFactory.

4.152 IndexedModalitiesGallery::ModalitiesIdentifiers Class Reference

Contains candidates per each modality.

```
#include <IndexedModalitiesGallery.h>
```

Collaboration diagram for IndexedModalitiesGallery::ModalitiesIdentifiers:



Public Member Functions

- std::vector< Identifier > GetPrintIdentifiers () const

  Returns best candidates for print modality.
- std::vector< Identifier > GetFaceIdentifiers () const

  Returns best candidates for face modality.
- std::vector< Identifier > GetIrisIdentifiers () const

  Returns best candidates for iris modality.

4.152.1 Member Function Documentation

4.152.1.1 GetFaceIdentifiers()

```
std::vector< Identifier > GetFaceIdentifiers () const  [inline], [nodiscard]
```

It will return best candidates for face modality in descending order.

Returns

identified candidates

4.152.1.2 GetIrisIdentifiers()

```
std::vector< Identifier > GetIrisIdentifiers () const  [inline], [nodiscard]
```

It will return best candidates for iris modality in descending order.

Returns

identified candidates

4.152.1.3 GetPrintIdentifiers()

```
std::vector< Identifier > GetPrintIdentifiers () const  [inline], [nodiscard]
```

It will return best candidates for print modality in descending order.

Returns

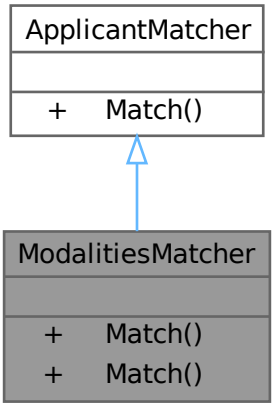
identified candidates

4.153 ModalitiesMatcher Class Reference

It is general matcher for ModalitiesTemplate - Applicant template in serialized form used for biometric operations.

```
#include <ModalitiesMatcher.h>
```

Collaboration diagram for ModalitiesMatcher:



Public Member Functions

- virtual ModalityScore Match (const ModalitiesTemplate &probe, const ModalitiesTemplate &gallery) const =0

  Calculates similarity score of probe and gallery template for each modality.
- virtual ModalityScore Match (const Applicant &probe, const Applicant &gallery) const=0

  Calculates similarity score of probe and gallery applicant ICS template for each modality.

4.153.1 Detailed Description

It compares probe and gallery applicant. Result of comparison is ModalityScore.

4.153.2 Member Function Documentation

4.153.2.1 Match() [1/2]

```
virtual ModalityScore Match (
            const Applicant & probe,
            const Applicant & gallery) const  [nodiscard], [virtual]
```

Parameters

probe	probe Applicant
gallery	gallery Applicant

Returns

similarity ModalityScore

Implements ApplicantMatcher.

Reimplemented in ModalitiesVerifyMatcher.

4.153.2.2 Match() [2/2]

```
virtual ModalityScore Match (
            const ModalitiesTemplate & probe,
            const ModalitiesTemplate & gallery) const  [nodiscard], [pure virtual]
```

Parameters

probe	probe ModalitiesTemplate
gallery	gallery ModalitiesTemplate

Returns

similarity ModalityScore

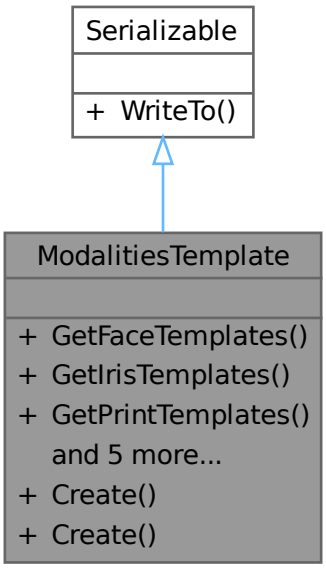
Implemented in ModalitiesVerifyMatcher.

4.154 ModalitiesTemplate Class Reference

Applicant template in serialized form used for biometric operations.

```
#include <ModalitiesTemplate.h>
```

Collaboration diagram for ModalitiesTemplate:



Public Member Functions

- std::vector< std::shared_ptr< FaceTemplate > > GetFaceTemplates () const

  Provides FaceTemplate objects.
- std::vector< std::shared_ptr< IrisTemplate > > GetIrisTemplates () const

  Provides IrisTemplate objects.
- std::vector< std::shared_ptr< PrintTemplate > > GetPrintTemplates () const

  Provides PrintTemplate objects.
- std::vector< uint8_t > GetModalitiesTemplateData () const

  It provides Applicant template data.
- std::shared_ptr< ModalitiesICSTemplate > ToModalitiesICSTemplate () const

  Converts template to Innovatrics template format useful for verification.
- std::shared_ptr< ModalitiesICSTemplate > ToModalitiesICSTemplate (const ModalitiesICSTemplateFactory &factory) const

  Converts template to Innovatrics template format useful for verification.
- ModalityScore SimilarWith (const ModalitiesTemplate &gallery, const ModalitiesMatcher &matcher) const

  It calculates similarity score between this and other applicant template.
- void WriteTo (Writer &w) const override

  Write itself into writer.

Static Public Member Functions

- static std::shared_ptr< ModalitiesTemplate > Create (const std::vector< uint8_t > &data)

  Create ModalitiesTemplate object from serialized data.
- static std::shared_ptr< ModalitiesTemplate > Create (std::vector< std::shared_ptr< PrintTemplate > > prints, std::vector< std::shared_ptr< FaceTemplate > > faces, std::vector< std::shared_ptr< IrisTemplate > > irises)

  Create ModalitiesTemplate object from templates.

4.154.1 Detailed Description

The serialized data contains templates for print, embedding print, face and iris modalities.

4.154.2 Member Function Documentation

4.154.2.1 Create() [1/2]

```
std::shared_ptr< ModalitiesTemplate > Create (
            const std::vector< uint8_t > & data)  [inline], [static]
```

Parameters

data	template data

Returns

created modality template

4.154.2.2 Create() [2/2]

```
std::shared_ptr< ModalitiesTemplate > Create (
            std::vector< std::shared_ptr< PrintTemplate > > prints,
            std::vector< std::shared_ptr< FaceTemplate > > faces,
            std::vector< std::shared_ptr< IrisTemplate > > irises)  [inline], [static]
```

Parameters

prints	print templates
faces	face templates
irises	iris templates

Returns

created modality template

4.154.2.3 GetFaceTemplates()

```
std::vector< std::shared_ptr< FaceTemplate > > GetFaceTemplates () const  [inline],
[nodiscard]
```

Returns

face templates

4.154.2.4 GetIrisTemplates()

```
std::vector< std::shared_ptr< IrisTemplate > > GetIrisTemplates () const [inline],
[nodiscard]
```

Returns

iris templates

4.154.2.5 GetModalitiesTemplateData()

```
std::vector< uint8_t > GetModalitiesTemplateData () const [inline], [nodiscard]
```

Returns

Applicant template data

Exceptions

EnrollmentException	when it is not possible to serialize template

4.154.2.6 GetPrintTemplates()

```
std::vector< std::shared_ptr< PrintTemplate > > GetPrintTemplates () const [inline],
[nodiscard]
```

Returns

print templates

4.154.2.7 SimilarWith()

```
ModalityScore SimilarWith (
            const ModalitiesTemplate & gallery,
            const ModalitiesMatcher & matcher) const [inline], [nodiscard]
```

It uses given ModalitiesTemplateMatcher to calculate the score.

Parameters

gallery	ModalitiesTemplate.
matcher	is ModalitiesTemplateMatcher used to calculate modality score between probe (this template) and given template `gallery`

Returns

modality score

4.154.2.8 ToModalitiesICSTemplate() [1/2]

```
std::shared_ptr< ModalitiesICSTemplate > ToModalitiesICSTemplate () const [inline],
[nodiscard]
```

It uses ModalitiesICSTemplateDefaultFactory for conversion to ModalitiesICSTemplate.

Returns

converted template

4.154.2.9 ToModalitiesICSTemplate() [2/2]

```
std::shared_ptr< ModalitiesICSTemplate > ToModalitiesICSTemplate (
            const ModalitiesICSTemplateFactory & factory) const [inline], [nodiscard]
```

It uses given factory for conversion to ModalitiesICSTemplate.

Parameters

factory	ModalitiesICSTemplateFactory for conversion

Returns

converted template

4.154.2.10 WriteTo()

```
void WriteTo (
            Writer & writer) const [inline], [override], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

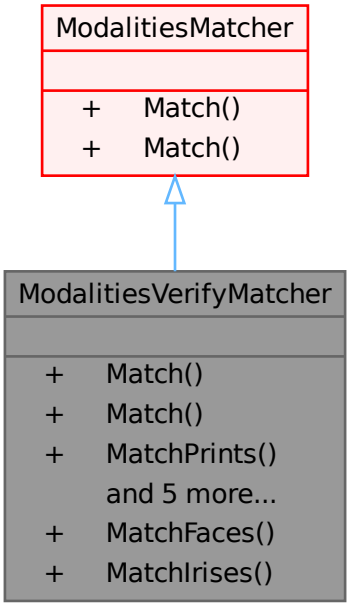
Implements Serializable.

4.155 ModalitiesVerifyMatcher Class Reference

ModalitiesVerifyMatcher is template matcher that uses Innovatrics proprietary verification algorithm.

```
#include <ModalitiesVerifyMatcher.h>
```

Collaboration diagram for ModalitiesVerifyMatcher:



Classes

- class Config

  ModalitiesVerifyMatcher configuration.

Public Member Functions

- ModalityScore Match (const Applicant &probe, const Applicant &gallery) const final

  Calculates similarity score of probe and gallery applicant template for each modality.
- ModalityScore Match (const ModalitiesTemplate &probe, const ModalitiesTemplate &gallery) const final

  Calculates similarity score of probe and gallery applicant for each applicant modality.
- virtual PrintSimilarityScore MatchPrints (const std::vector< std::shared_ptr< PrintTemplate > > &probe, const std::vector< std::shared_ptr< PrintTemplate > > &gallery) const

  Calculates prints similarity score of probe and gallery that contains only prints.
- **ModalitiesVerifyMatcher** ()

  Constructor with default Config GlobalMultiModalityExecutor is used to execute matching.
- ModalitiesVerifyMatcher (Config config)

  Constructor uses Config to setup matching algorithm.

- ModalitiesVerifyMatcher (Config config, std::shared_ptr< MultiModalityExecutor > executor)

  Constructor uses Config to setup matching algorithm.
- ModalitiesVerifyMatcher (Config config, std::shared_ptr< MultiModalityExecutor > executor, std::shared_ptr< Logger > logger)

  Constructor uses Config to setup matching algorithm.
- ModalitiesVerifyMatcher (std::shared_ptr< MultiModalityExecutor > executor)

  Constructor with default Config.

Static Public Member Functions

- static FaceSimilarityScore MatchFaces (const std::vector< std::shared_ptr< FaceTemplate > > &probe, const std::vector< std::shared_ptr< FaceTemplate > > &gallery)

  Calculates face similarity score of probe and gallery that contains only faces.
- static IrisSimilarityScore MatchIrises (const std::vector< std::shared_ptr< IrisTemplate > > &probe, const std::vector< std::shared_ptr< IrisTemplate > > &gallery)

  Calculates face similarity score of probe and gallery that contains only irises.

4.155.1 Detailed Description

It is configurable via ModalitiesVerifyMatcher::Config verifies probe against gallery template.

4.155.2 Constructor & Destructor Documentation

4.155.2.1 ModalitiesVerifyMatcher() [1/4]

```
ModalitiesVerifyMatcher (
            Config config) [inline], [explicit]
```

GlobalMultiModalityExecutor is used to execute matching.

Parameters

config	is matching algorithm configuration.

4.155.2.2 ModalitiesVerifyMatcher() [2/4]

```
ModalitiesVerifyMatcher (
            Config config,
            std::shared_ptr< MultiModalityExecutor > executor) [inline]
```

Parameters

config	is matching algorithm configuration.
executor	MultiModalityExecutor used for matching

Exceptions

NullArgumentException	when executor is null

4.155.2.3 ModalitiesVerifyMatcher() [3/4]

```
ModalitiesVerifyMatcher (
            Config config,
            std::shared_ptr< MultiModalityExecutor > executor,
            std::shared_ptr< Logger > logger)  [inline]
```

Parameters

config	is matching algorithm configuration.
executor	MultiModalityExecutor used for matching
logger	logger used for logging

Exceptions

NullArgumentException	when executor is null

4.155.2.4 ModalitiesVerifyMatcher() [4/4]

```
ModalitiesVerifyMatcher (
            std::shared_ptr< MultiModalityExecutor > executor)  [inline], [explicit]
```

Parameters

executor	MultiModalityExecutor used for matching

Exceptions

NullArgumentException	when executor is null

4.155.3 Member Function Documentation

4.155.3.1 Match() [1/2]

```
ModalityScore Match (
            const Applicant & probe,
            const Applicant & gallery) const  [inline], [nodiscard], [final], [virtual]
```

This is convenient function to check similarity of applicants. It is not efficient because it always creates probe and gallery ModalitiesTemplate.

Parameters

probe	probe Applicant
gallery	gallery Applicant

Returns

similarity ModalityScore

Exceptions

EnrollmentException	when it is not possible to create template or match created templates

Reimplemented from ModalitiesMatcher.

4.155.3.2 Match() [2/2]

```
ModalityScore Match (
            const ModalitiesTemplate & probe,
            const ModalitiesTemplate & gallery) const  [inline], [nodiscard], [final],
[virtual]
```

Parameters

probe	probe ModalitiesTemplate
gallery	gallery ModalitiesTemplate

Returns

similarity ModalityScore

Exceptions

EnrollmentException	when it is not possible to match templates

Implements ModalitiesMatcher.

4.155.3.3 MatchFaces()

```
FaceSimilarityScore MatchFaces (
            const std::vector< std::shared_ptr< FaceTemplate > > & probe,
            const std::vector< std::shared_ptr< FaceTemplate > > & gallery)  [inline],
[static], [nodiscard]
```

It compares each probe face against each gallery face and return max score from all comparisons.

Parameters

probe	probe faces
gallery	gallery faces

Returns

similarity FaceSimilarityScore

Exceptions

EnrollmentException	when it is not possible to match templates

4.155.3.4 MatchIrises()

```
IrisSimilarityScore MatchIrises (
            const std::vector< std::shared_ptr< IrisTemplate > > & probe,
            const std::vector< std::shared_ptr< IrisTemplate > > & gallery)  [inline],
[static], [nodiscard]
```

Parameters

probe	probe irises
gallery	gallery irises

Returns

similarity IrisSimilarityScore

Exceptions

EnrollmentException	when it is not possible to match templates

4.155.3.5 MatchPrints()

```
virtual PrintSimilarityScore MatchPrints (
            const std::vector< std::shared_ptr< PrintTemplate > > & probe,
            const std::vector< std::shared_ptr< PrintTemplate > > & gallery) const  [inline],
[nodiscard], [virtual]
```

Parameters

probe	probe prints
gallery	gallery prints

Returns

similarity PrintSimilarityScore

Exceptions

EnrollmentException	when it is not possible to match templates

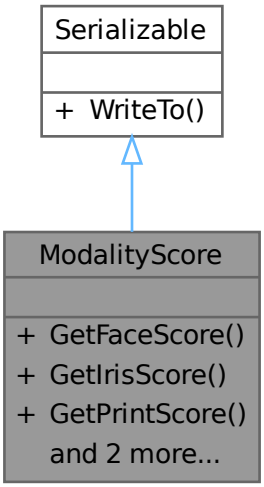
Reimplemented in ModalitiesEmbeddingVerifyMatcher.

4.156 ModalityScore Class Reference

Contains scores of multiple modalities.

```
#include <SimilarityScore.h>
```

Collaboration diagram for ModalityScore:



Public Member Functions

- FaceSimilarityScore GetFaceScore () const

  It provides face similarity score.
- IrisSimilarityScore GetIrisScore () const

  It provides iris similarity score.
- PrintSimilarityScore GetPrintScore () const

  It provides print similarity score.
- void WriteTo (Writer &writer) const final

  Write scores into writer.
- ModalityScore (FaceSimilarityScore face, IrisSimilarityScore iris, PrintSimilarityScore print)

  Constructs new ModalityScore object.

4.156.1 Constructor & Destructor Documentation

4.156.1.1 ModalityScore()

```
ModalityScore (
            FaceSimilarityScore face,
            IrisSimilarityScore iris,
            PrintSimilarityScore print)  [inline]
```

Parameters

face	face similarity score
iris	iris similarity score
print	Print similarity score

4.156.2 Member Function Documentation

4.156.2.1 GetFaceScore()

```
FaceSimilarityScore GetFaceScore () const  [inline], [nodiscard]
```

Returns

face similarity score

4.156.2.2 GetIrisScore()

```
IrisSimilarityScore GetIrisScore () const  [inline], [nodiscard]
```

Returns

iris similarity score

4.156.2.3 GetPrintScore()

```
PrintSimilarityScore GetPrintScore () const  [inline], [nodiscard]
```

Returns

Print similarity score

4.156.2.4 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize ModalityScore

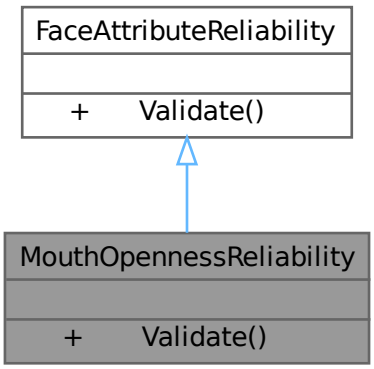
Implements Serializable.

4.157 MouthOpennessReliability Class Reference

Check if calculation of Mouth Openness attribute will be reliable.

```
#include <FaceValidation.h>
```

Collaboration diagram for MouthOpennessReliability:



Public Member Functions

- std::vector< FailedFaceAttribute > Validate (const std::shared_ptr< Face > &probe) final

  Check if calculation of Mouth Openness attribute will be reliable.

4.157.1 Member Function Documentation

4.157.1.1 Validate()

```
std::vector< FailedFaceAttribute > Validate (
            const std::shared_ptr< Face > & probe)  [inline], [final], [virtual]
```

Parameters

probe	Face to be checked

Returns

list of attributes that will cause not reliable Mouth Openness attribute

Exceptions

NullArgumentException	when probe is null

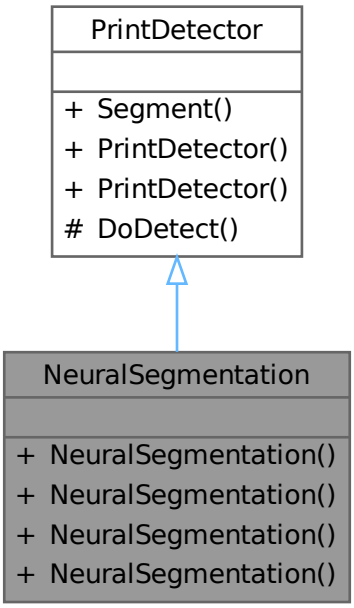
Implements FaceAttributeReliability.

4.158 NeuralSegmentation Class Reference

NeuralSegmentation implements segmentation based on neural network from Innovatrics ISegLib library.

```
#include <PrintDetector.h>
```

Collaboration diagram for NeuralSegmentation:



Classes

- class Config

  Configuration for NeuralSegmentation.

Public Member Functions

- NeuralSegmentation (Config config)

  Construct a new NeuralSegmentation object.
- **NeuralSegmentation** ()

  Construct a new NeuralSegmentation with default config.
- NeuralSegmentation (Config config, std::shared_ptr< PrintExecutor > executor)

  Construct a new NeuralSegmentation object.
- NeuralSegmentation (std::shared_ptr< PrintExecutor > executor)

  Construct a new NeuralSegmentation with default config.

Public Member Functions inherited from PrintDetector

- std::vector< std::shared_ptr< PrintSegment > > Segment (const std::shared_ptr< PrintCapture > &capture) final

  Detects segments in given image The Function scans the provided capture to locate fingers.
- **PrintDetector** ()

  Construct a new PrintDetector object GlobalPrintExecutor is used for PrintSegment detection.
- PrintDetector (std::shared_ptr< PrintExecutor > executor)

  Construct a new PrintDetector object.

4.158.1 Constructor & Destructor Documentation

4.158.1.1 NeuralSegmentation() [1/3]

```
NeuralSegmentation (
            Config config) [inline], [explicit]
```

Parameters

config	Config

4.158.1.2 NeuralSegmentation() [2/3]

```
NeuralSegmentation (
            Config config,
            std::shared_ptr< PrintExecutor > executor) [inline]
```

Parameters

config	Config

executor	PrintExecutor to set where functionality is executed

Exceptions

NullArgumentException	when executor is null

4.158.1.3 NeuralSegmentation() [3/3]

```
NeuralSegmentation (
            std::shared_ptr< PrintExecutor > executor)  [inline], [explicit]
```

Parameters

executor	PrintExecutor to set where functionality is executed

Exceptions

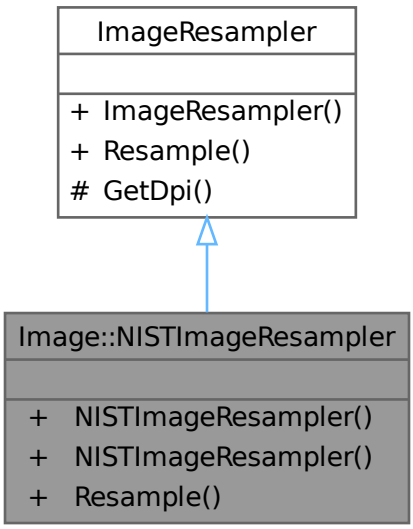
NullArgumentException	when executor is null

4.159 Image::NISTImageResampler Class Reference

Image resampler for DPI-change operations, including upsampling and downsampling using NIST-compliant resampling algorithm provided by `NFIR library`.

`#include <Image.h>`

Collaboration diagram for Image::NISTImageResampler:



Public Types

- enum class InterpolationMethod : uint8_t { BICUBIC , BILINEAR }

  Enum defines possible types of interpolation methods to apply during image resampling.
- enum class FilterType : uint8_t { GAUSSIAN , IDEAL }

  Enum defines possible types of filtering methods to apply before image resampling.

Public Member Functions

- NISTImageResampler (ImageDpi targetDpi)

  Constructs a NIST-compliant image resampler for a target DPI.
- NISTImageResampler (ImageDpi targetDpi, InterpolationMethod interpolationMethod, FilterType filterType)

  Constructs a NIST-compliant image resampler for a target DPI using a custom interpolation method and filter.
- std::shared_ptr< Image > Resample (const inno::Image &image) const final

  Resample image to a target DPI using NIST resamling algorithm.

Public Member Functions inherited from Image::ImageResampler

- ImageResampler (ImageDpi targetDpi)

  Constructs an image resampler for a target DPI.

Additional Inherited Members

Protected Member Functions inherited from Image::ImageResampler

- ImageDpi GetDpi () const

  Retrieves target DPI of the resampler.

4.159.1 Member Enumeration Documentation

4.159.1.1 FilterType

```
enum class FilterType : uint8_t [strong]
```

Enumerator

GAUSSIAN	Apply a Gaussian filter for smoothing.
IDEAL	Apply an ideal filter for sharpness.

4.159.1.2 InterpolationMethod

```
enum class InterpolationMethod : uint8_t [strong]
```

Enumerator

BICUBIC	Apply bicubic interpolation.
BILINEAR	Apply bilinear interpolation.

4.159.2 Constructor & Destructor Documentation

4.159.2.1 NISTImageResampler() [1/2]

```
NISTImageResampler (
            ImageDpi targetDpi) [inline], [explicit]
```

This constructor initializes the image resampler to resample an image to the specified DPI using a NIST-compliant resampling algorithm with default interpolation and filtering settings.

Note

It is recommended to explicitly specify the interpolation method and filter type by using the constructor `NISTImageResampler(ImageDpi targetDpi, InterpolationMethod interpolationMethod, FilterType filterType)` rather than relying on the default settings. The best settings for your use case are likely to be determined through trial and error.

Parameters

targetDpi	The DPI (dots per inch) of the new image.

Exceptions

EnrollmentInvalidArgumentException	when targetDpi is 0

4.159.2.2 NISTImageResampler() [2/2]

```
NISTImageResampler (
            ImageDpi targetDpi,
            InterpolationMethod interpolationMethod,
            FilterType filterType) [inline]
```

This constructor initializes the image resampler to resample an image to the specified DPI using a NIST-compliant resampling algorithm and the provided interpolation and filtering methods.

Depending on the resize factor, the optimal configuration of the filter type and interpolation method for achieving the best extraction accuracy was experimentally determined, as shown in the table below:

Source DPI	Interpolation Method	Filter Type
600	Bicubic	Ideal
1000	Bilinear	Ideal
1200	Bilinear	Gaussian
Other	Bicubic	Ideal

This is also the default behavior of a resampler constructed using NISTImageResampler(ImageDpi targetDpi).

Note

It is recommended to explicitly specify the interpolation method and downsampling filter mask rather than relying on the default settings. The best settings for your use case are likely to be determined through trial and error.

Parameters

targetDpi	The DPI (dots per inch) of the new image.
interpolationMethod	The interpolation method to be applied during resampling.
filterType	The type of filtering to be applied before resampling.

Exceptions

EnrollmentInvalidArgumentException	when targetDpi is 0

4.159.3 Member Function Documentation

4.159.3.1 Resample()

```
std::shared_ptr< Image > Resample (
            const inno::Image & image) const  [inline], [nodiscard], [final], [virtual]
```

It creates new image as result of resampling with factor target DPI/original DPI.

Parameters

image	Image to resize.

Returns

image with given DPI

Exceptions

ImageException	when there is an processing error

EnrollmentInvalidArgumentException	when the image is not single-channel grayscale

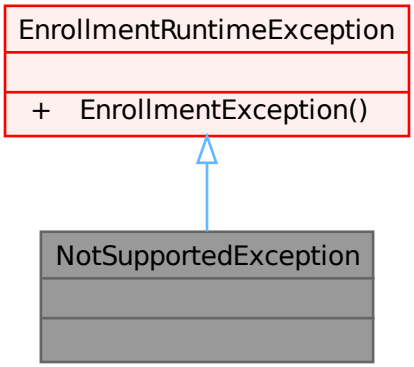
Implements Image::ImageResampler.

4.160 NotSupportedException Class Reference

When user uses not supported functionality.

```
#include <NotSupportedException.h>
```

Inheritance diagram for NotSupportedException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentRuntimeException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

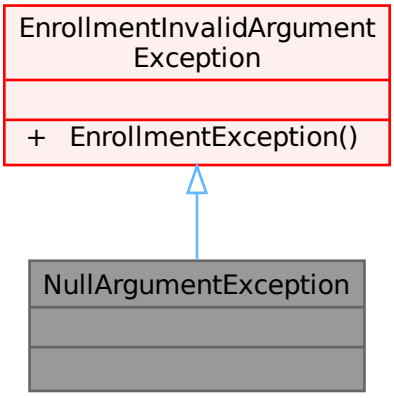
  Provides message with full info about exception.

4.161 NullArgumentException Class Reference

Null argument is not allowed.

```
#include <EnrollmentException.h>
```

Inheritance diagram for NullArgumentException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentInvalidArgumentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

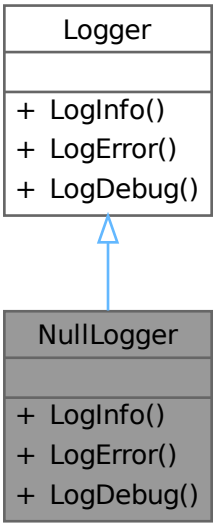
  Provides message with full info about exception.

4.162 NullLogger Class Reference

The logger implementation that do not logs.

```
#include <Logger.h>
```

Collaboration diagram for NullLogger:



Public Member Functions

- void **LogInfo** (LogMessage &) final

  It ignores info messages.
- void **LogError** (LogMessage &) final

  It ignores error messages.
- void **LogDebug** (LogMessage &) final

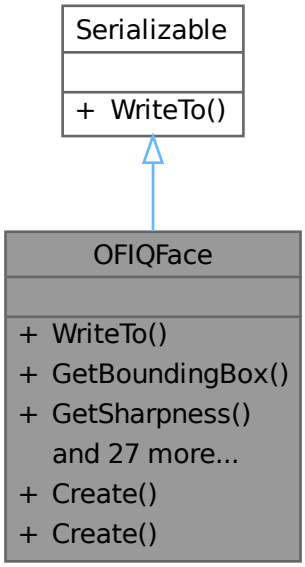
  It ignores debug messages.

4.163 OFIQFace Class Reference

It provides quality measures as defined in ISO/IEC 29794-5:2025.

```
#include <OFIQFace.h>
```

Collaboration diagram for OFIQFace:



Public Member Functions

- void WriteTo (Writer &writer) const final

  Serialize OFIQFace to Writer.
- Rectangle GetBoundingBox () const

  Provides bounding box of detected face.
- uint8_t GetSharpness () const

  Provides sharpness of the face.
- uint8_t GetUnifiedQualityScore () const

  Unified quality score (QS) mapping native OFIQ quality.
- uint8_t GetBackgroundUniformity () const

  Background uniformity of the face image.
- uint8_t GetIlluminationUniformity () const

  Illumination uniformity across the face.
- uint8_t GetLuminanceMean () const

  Luminance mean of the face image.
- uint8_t GetLuminanceVariance () const

  Luminance variance in the face image.
- uint8_t GetUnderExposurePrevention () const

  Under-exposure prevention.
- uint8_t GetOverExposurePrevention () const

  Over-exposure prevention.
- uint8_t GetDynamicRange () const

  Dynamic range of the face image.
- uint8_t GetCompressionArtifacts () const

Absence of compression artefacts.

- uint8_t GetNaturalColour () const

  Natural colour of the face image.
- uint8_t GetSingleFacePresent () const

  Single face present in the image.
- uint8_t GetEyesOpen () const

  Eyes open.
- uint8_t GetMouthClosed () const

  Mouth closed.
- uint8_t GetEyesVisible () const

  Eyes visible.
- uint8_t GetMouthOcclusionPrevention () const

  Mouth occlusion prevention.
- uint8_t GetFaceOcclusionPrevention () const

  Face occlusion prevention.
- uint8_t GetInterEyeDistance () const

  Inter-eye distance quality.
- uint8_t GetHeadSize () const

  Head size in the image.
- uint8_t GetLeftwardCropOfTheFaceImage () const

  Leftward crop of the face image.
- uint8_t GetRightwardCropOfTheFaceImage () const

  Rightward crop of the face image.
- uint8_t GetMarginAboveOfTheFaceImage () const

  Margin above the face image.
- uint8_t GetMarginBelowOfTheFaceImage () const

  Margin below the face image.
- uint8_t GetHeadPoseYaw () const

  Head pose yaw (left-right rotation).
- uint8_t GetHeadPosePitch () const

  Head pose pitch (up-down tilt).
- uint8_t GetHeadPoseRoll () const

  Head pose roll (tilt about the nose axis).
- uint8_t GetExpressionNeutrality () const

  Expression neutrality.
- uint8_t GetNoHeadCoverings () const

  No head coverings.

Static Public Member Functions

- static std::shared_ptr< OFIQFace > Create (const std::shared_ptr< FaceCapture > &capture, const std::↩ shared_ptr< FaceExecutor > &executor)

  Create OFIQ from FaceCapture using the given executor.
- static std::shared_ptr< OFIQFace > Create (const std::shared_ptr< FaceCapture > &capture)

  Create OFIQ from FaceCapture using default executor set in GlobalFaceExecutor.

4.163.1 Detailed Description

To calculate qualities we are using `OFIQ-project`.

4.163.2 Member Function Documentation

4.163.2.1 Create() [1/2]

```
std::shared_ptr< OFIQFace > Create (
            const std::shared_ptr< FaceCapture > & capture)  [inline], [static]
```

It creates this instance and calculates all qualities at once.

Required Solvers

Total size of solvers is 635 MB.

References ofiq/expression_neutrality/grimmer/hse_1_2_C_adaboost.yml, ofiq/expression_neutrality/hsemotion/enet_b0_8_bes ofiq/expression_neutrality/hsemotion/enet_b2_8_embed_zeroed.onnx, ofiq/face_detection/ssd_facedetect.caffemodel, ofiq/face_detection/ssd_facedetect.prototxt.txt, ofiq/face_landmark_estimation/ADNet.onnx, ofiq/face_occlusion_segmentation/face_occlusion_segmentation_ort.onnx, ofiq/face_parsing/bisenet_400.onnx, ofiq/head_pose_estimation/mb1_120x120.onnx, ofiq/no_compression_artifacts/ssim_248_model.onnx, ofiq/sharpness/face_sharpness_rtree.xml, ofiq/unified_quality_score/magface_iresnet50_norm.onnx

Parameters

capture	FaceCapture

Returns

OFIQFace instance with calculated qualities

Exceptions

NullArgumentException	when capture is null
EnrollmentException	when OFIQ calculation fails

4.163.2.2 Create() [2/2]

```
std::shared_ptr< OFIQFace > Create (
            const std::shared_ptr< FaceCapture > & capture,
            const std::shared_ptr< FaceExecutor > & executor)  [inline], [static]
```

It creates this instance and calculates all qualities at once.

Required Solvers

Total size of solvers is 635 MB.

References ofiq/expression_neutrality/grimmer/hse_1_2_C_adaboost.yml, ofiq/expression_neutrality/hsemotion/enet_b0_8_bes ofiq/expression_neutrality/hsemotion/enet_b2_8_embed_zeroed.onnx, ofiq/face_detection/ssd_facedetect.caffemodel, ofiq/face_detection/ssd_facedetect.prototxt.txt, ofiq/face_landmark_estimation/ADNet.onnx, ofiq/face_occlusion_segmentation/face_occlusion_segmentation_ort.onnx, ofiq/face_parsing/bisenet_400.onnx, ofiq/head_pose_estimation/mb1_120x120.onnx, ofiq/no_compression_artifacts/ssim_248_model.onnx, ofiq/sharpness/face_sharpness_rtree.xml, ofiq/unified_quality_score/magface_iresnet50_norm.onnx

Parameters

capture	FaceCapture
executor	FaceExecutor used to execute calculation locally or on server

Returns

OFIQFace instance with calculated qualities

Exceptions

NullArgumentException	when capture is null
NullArgumentException	when executor is null
EnrollmentException	when OFIQ calculation fails

4.163.2.3 GetBackgroundUniformity()

```
uint8_t GetBackgroundUniformity () const  [inline], [nodiscard]
```

Measures how uniform the background is behind the face. [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 40.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.4 GetBoundingBox()

```
Rectangle GetBoundingBox () const  [inline], [nodiscard]
```

Returns

Bounding box of detected face

4.163.2.5 GetCompressionArtifacts()

```
uint8_t GetCompressionArtifacts () const  [inline], [nodiscard]
```

Measures degradation from lossy compression (e.g. JPEG). [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.6 GetDynamicRange()

```
uint8_t GetDynamicRange () const  [inline], [nodiscard]
```

Reflects whether the image has sufficient range between darkest and brightest areas.

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.7 GetExpressionNeutrality()

```
uint8_t GetExpressionNeutrality () const  [inline], [nodiscard]
```

Indicates that the face has a neutral expression (e.g. mouth closed, no smile). [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 40.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.8 GetEyesOpen()

```
uint8_t GetEyesOpen () const [inline], [nodiscard]
```

Indicates that both eyes are open (no closed or drooping eyelids). [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.9 GetEyesVisible()

```
uint8_t GetEyesVisible () const [inline], [nodiscard]
```

Indicates that both eyes are visible and not occluded.

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.10 GetFaceOcclusionPrevention()

```
uint8_t GetFaceOcclusionPrevention () const [inline], [nodiscard]
```

Indicates that the face is not occluded by objects, hair, or other obstructions.

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.11 GetHeadPosePitch()

```
uint8_t GetHeadPosePitch () const [inline], [nodiscard]
```

Indicates vertical orientation of the head. [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.12 GetHeadPoseRoll()

```
uint8_t GetHeadPoseRoll () const [inline], [nodiscard]
```

Indicates whether the head is upright (no sideways tilt). [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.13 GetHeadPoseYaw()

```
uint8_t GetHeadPoseYaw () const [inline], [nodiscard]
```

Indicates how frontal the face is in the horizontal direction. [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.14 GetHeadSize()

```
uint8_t GetHeadSize () const  [inline], [nodiscard]
```

Indicates whether the head size (e.g. face size) is within the expected range. [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 40.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.15 GetIlluminationUniformity()

```
uint8_t GetIlluminationUniformity () const  [inline], [nodiscard]
```

Measures uniformity of lighting on the face (e.g. absence of strong shadows or hotspots). [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 40.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.16 GetInterEyeDistance()

```
uint8_t GetInterEyeDistance () const  [inline], [nodiscard]
```

Indicates whether the distance between the eyes is within the expected range for the image. [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 40.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.17 GetLeftwardCropOfTheFaceImage()

```
uint8_t GetLeftwardCropOfTheFaceImage () const  [inline], [nodiscard]
```

Indicates sufficient margin to the left of the face in the frame.

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.18 GetLuminanceMean()

```
uint8_t GetLuminanceMean () const  [inline], [nodiscard]
```

Indicates whether the face is neither under- nor over-exposed on average.

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 40.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.19 GetLuminanceVariance()

```
uint8_t GetLuminanceVariance () const  [inline], [nodiscard]
```

Reflects variation in brightness across the face.

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 40.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.20 GetMarginAboveOfTheFaceImage()

```
uint8_t GetMarginAboveOfTheFaceImage () const [inline], [nodiscard]
```

Indicates sufficient margin above the head in the frame. [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 40.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.21 GetMarginBelowOfTheFaceImage()

```
uint8_t GetMarginBelowOfTheFaceImage () const [inline], [nodiscard]
```

Indicates sufficient margin below the chin in the frame. [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 40.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.22 GetMouthClosed()

```
uint8_t GetMouthClosed () const [inline], [nodiscard]
```

Indicates that the mouth is closed (neutral expression). [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.23 GetMouthOcclusionPrevention()

```
uint8_t GetMouthOcclusionPrevention () const  [inline], [nodiscard]
```

Indicates that the mouth region is not occluded (e.g. by hands or objects).

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.24 GetNaturalColour()

```
uint8_t GetNaturalColour () const  [inline], [nodiscard]
```

Indicates that skin and scene colours appear natural (no strong colour cast or enhancement). [8]

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 40.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.25 GetNoHeadCoverings()

```
uint8_t GetNoHeadCoverings () const  [inline], [nodiscard]
```

Indicates that the face is not obscured by hats, caps, scarves, or similar coverings.

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 40.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.26 GetOverExposurePrevention()

```
uint8_t GetOverExposurePrevention () const  [inline], [nodiscard]
```

Indicates that the image is not too bright (over-exposed or washed out).

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.27 GetRightwardCropOfTheFaceImage()

```
uint8_t GetRightwardCropOfTheFaceImage () const  [inline], [nodiscard]
```

Indicates sufficient margin to the right of the face in the frame.

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.28 GetSharpness()

```
uint8_t GetSharpness () const  [inline], [nodiscard]
```

Sharpness is the clarity of fine details in an image. It will be improved with precise focus, good imaging resolution, and absence of biometric capture subject motion relative to the camera. [8]

This quality component can be used to efficiently choose the better focused face portrait among several face samples of the same biometric capture subject. [8]

Returns

Sharpness score of the face in range [0, 100] where 100 is the best sharpness. Recommended decision threshold is around 40.

Exceptions

EnrollmentException	when sharpness calculation fails

4.163.2.29 GetSingleFacePresent()

```
uint8_t GetSingleFacePresent () const  [inline], [nodiscard]
```

Indicates that exactly one face is present and suitable for assessment.

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.30 GetUnderExposurePrevention()

```
uint8_t GetUnderExposurePrevention () const  [inline], [nodiscard]
```

Indicates that the image is not too dark (under-exposed).

Returns

Score in range [0, 100], where 100 is best. Recommended decision threshold is around 95.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.31 GetUnifiedQualityScore()

```
uint8_t GetUnifiedQualityScore () const  [inline], [nodiscard]
```

Predicts expected face recognition performance when compared to a canonical ISO/ICAO-compliant portrait.

Returns

Score in range [0, 100], where 100 indicates highest conformance. Recommended decision threshold is around 40.

Exceptions

EnrollmentException	when the calculation fails

4.163.2.32 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to serialize OFIQFace

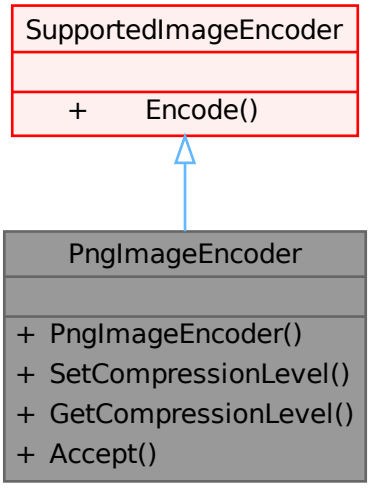
Implements Serializable.

4.164 PngImageEncoder Class Reference

Converts image type ImageType::PNG.

```
#include <ImageEncoder.h>
```

Collaboration diagram for PngImageEncoder:



Public Member Functions

- **PngImageEncoder** ()

  Construct a new Png Image Encoder object The default compression level is 9.
- void SetCompressionLevel (int level)

  Set compression level.
- int GetCompressionLevel () const

  Provides compression level.
- void Accept (ImageEncoderVisitor &visitor) const final

  Allow visitor to visit concrete type of encoder.

Public Member Functions inherited from SupportedImageEncoder

- std::vector< uint8_t > Encode (const RawImage &raw) const final

  Saves given RawImage as formatted image.

Additional Inherited Members

Static Public Member Functions inherited from ImageEncoder

- static std::shared_ptr< ImageEncoder > Create (ImageType type)

  Create concrete default encoder for given type.

4.164.1 Member Function Documentation

4.164.1.1 Accept()

```
void Accept (
            ImageEncoderVisitor & visitor) const [inline], [final], [virtual]
```

Parameters

visitor	ImageEncoderVisitor to call

Implements ImageEncoder.

4.164.1.2 GetCompressionLevel()

```
int GetCompressionLevel () const [inline]
```

Returns

compression level

4.164.1.3 SetCompressionLevel()

```
void SetCompressionLevel (
            int level) [inline]
```

Parameters

level	compression level from 0 to 9. A higher value means a smaller size and longer compression time.

Exceptions

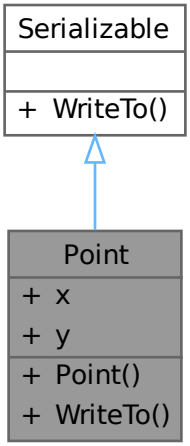
EnrollmentInvalidArgumentException	when level is not in range

4.165 Point Class Reference

It holds X and Y coordinates of point.

```
#include <ImageAttribute.h>
```

Collaboration diagram for Point:



Public Member Functions

- Point (X xCoordinate, Y yCoordinate)

  Constructor store given dimensions.
- void WriteTo (Writer &writer) const final

  Write itself into writer.

Public Attributes

- X **x** = 0

  X Coordinate.
- Y **y** = 0

  Y Coordinate.

4.165.1 Constructor & Destructor Documentation

4.165.1.1 Point()

```
Point (
            X xCoordinate,
            Y yCoordinate)  [inline]
```

Parameters

xCoordinate	x Coordinate
yCoordinate	y Coordinate

4.165.2 Member Function Documentation

4.165.2.1 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

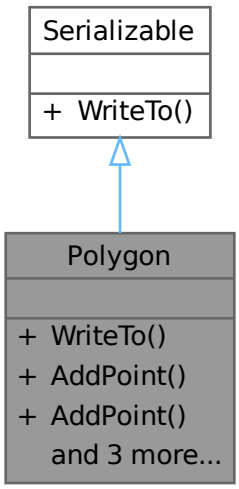
Implements Serializable.

4.166 Polygon Class Reference

It holds polygon points.

`#include <ImageAttribute.h>`

Collaboration diagram for Polygon:



Public Member Functions

- void WriteTo (Writer &writer) const final

  Write itself into writer.
- void AddPoint (const Point &point)

  Adds a point to the polygon.
- void AddPoint (Point &&point)

  Adds a point to the polygon using move semantics.
- const std::vector< Point > & GetPoints () const

  Retrieves the points describing the polygon.
- Polygon (std::vector< Point > &&points)

  Constructs a polygon with a given set of points using move semantics.
- Polygon (const std::vector< Point > &points)

  Constructs a polygon with a given set of points.

4.166.1 Constructor & Destructor Documentation

4.166.1.1 Polygon() [1/2]

```
Polygon (
            std::vector< Point > && points)  [inline], [explicit]
```

Parameters

points	Point objects to initialize the polygon with.

4.166.1.2 Polygon() [2/2]

```
Polygon (
            const std::vector< Point > & points)  [inline], [explicit]
```

Parameters

points	Point objects to initialize the polygon with.

4.166.2 Member Function Documentation

4.166.2.1 AddPoint() [1/2]

```
void AddPoint (
            const Point & point)  [inline]
```

Parameters

point	The point to add to the polygon.

4.166.2.2 AddPoint() [2/2]

```
void AddPoint (
            Point && point)  [inline]
```

Parameters

point	The point to be moved to the polygon.

4.166.2.3 GetPoints()

```
const std::vector< Point > & GetPoints () const  [inline], [nodiscard]
```

Returns

A const reference to points describing the polygon.

4.166.2.4 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

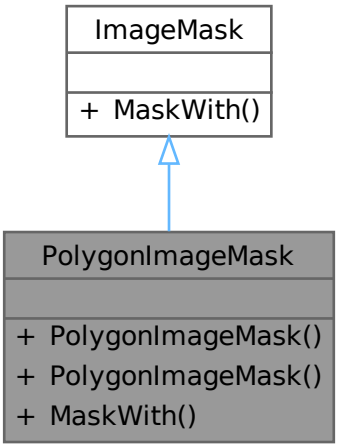
Implements Serializable.

4.167 PolygonImageMask Class Reference

Represents an image mask defined by a polygonal shape.

`#include <ImageMask.h>`

Collaboration diagram for PolygonImageMask:



Public Member Functions

- PolygonImageMask (Polygon polygon, ImageMaskConfiguration config)

  Constructs a PolygonImageMask with a specified polygon and configuration.
- PolygonImageMask (Polygon polygon, ImageMaskConfiguration config, ImageBlurType blur)

  Constructs a PolygonImageMask with a specified polygon and configuration.
- void MaskWith (ImageMasker &masker) const override

  Applies the polygonal mask to an image using the provided ImageMasker.

4.167.1 Detailed Description

It is a convenient class for masking and blurring polygon.

4.167.2 Constructor & Destructor Documentation

4.167.2.1 PolygonImageMask() [1/2]

```
PolygonImageMask (
            Polygon polygon,
            ImageMaskConfiguration config)  [inline]
```

The polygon defines the shape and bounds of the mask, enabling masking of complex and irregular shapes. The configuration specifies additional parameters such as mask color and mask type, enhancing the flexibility and applicability of the mask.

Parameters

polygon	The Polygon object that defines the shape and bounds of the mask.
config	The ImageMaskConfiguration object that contains settings for how the mask should be applied, including blur and mask types.

4.167.2.2 PolygonImageMask() [2/2]

```
PolygonImageMask (
            Polygon polygon,
            ImageMaskConfiguration config,
            ImageBlurType blur)  [inline]
```

The polygon defines the shape and bounds of the mask, enabling masking of complex and irregular shapes. The configuration specifies additional parameters such as mask color and mask type, enhancing the flexibility and applicability of the mask.

Parameters

polygon	The Polygon object that defines the shape and bounds of the mask.
blur	Type of blurring to by applied on mask-to-image transition.

config	The ImageMaskConfiguration object that contains settings for how the mask should be applied, including blur and mask types.

4.167.3 Member Function Documentation

4.167.3.1 MaskWith()

```
void MaskWith (
            ImageMasker & masker) const  [inline], [override], [virtual]
```

Parameters

masker	The ImageMasker instance used to apply the mask to an image.

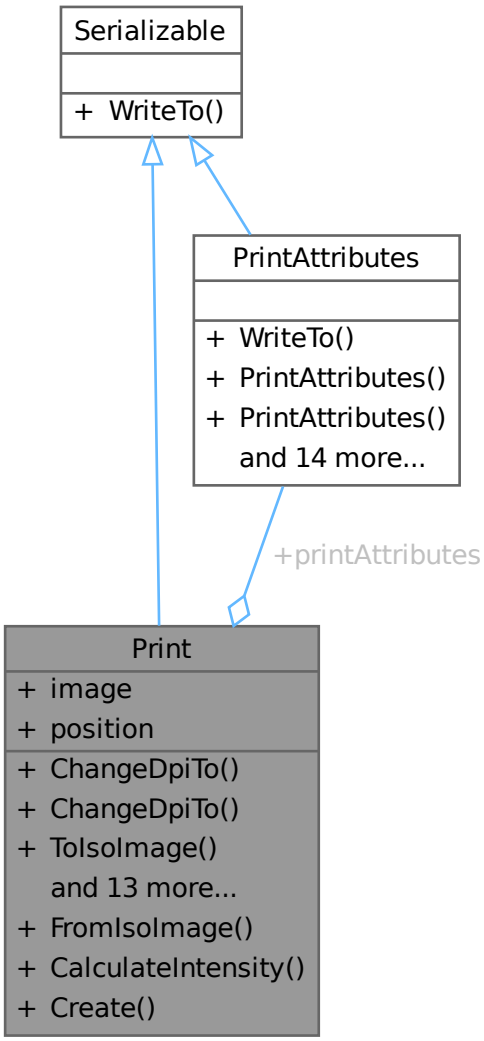
Implements ImageMask.

4.168 Print Class Reference

It holds all functions and data related to print manipulation.

```
#include <Print.h>
```

Collaboration diagram for Print:



Public Member Functions

- std::shared_ptr< Print > ChangeDpiTo (ImageDpi targetDpi) const

  Create new Print with given DPI.
- std::shared_ptr< Print > ChangeDpiTo (const Image::ImageResampler &resampler) const

  Create new Print using provided DPI-change configuration.
- std::vector< uint8_t > ToIsoImage (ImageEncoder &encoder) const

  Function creates ISO image based on ISO/IEC 19794-4 (Finger image data).
- std::vector< uint8_t > ToIsoRawImage () const

  Function creates ISO image based on ISO/IEC 19794-4 (Finger image data) use RawImage.
- uint8_t CalculatePresence ()

  Function calculates fingerprint presence on the fingerprint scanner surface.
- uint8_t CalculateNFIQ1 ()

Function calculates NFIQ score (quality score defined by NIST) of the fingerprint image.

- uint8_t CalculateNFIQ2 ()

  Function calculates NFIQ2 score (quality score defined by NIST) of the fingerprint image.
- uint8_t CalculateImageQuality ()

  Function calculates Image quality of the input fingerprint image.
- uint8_t CalculatePlacement ()

  Function calculates the finger placement quality of the input fingerprint image.
- uint8_t CalculateIntensity ()

  Function calculates the image intensity.
- std::shared_ptr< Image > DrawMinutiae (const std::vector< std::shared_ptr< Minutia > > &minutiae, const ImageShapeAttributes &attr) const

  Draws minutiae on image.
- std::shared_ptr< Image > DrawCriticalPoints (const std::vector< std::shared_ptr< CriticalPoint > > &points, const ImageShapeAttributes &attr) const

  Draws critical points on skeleton image.
- void WriteTo (Writer &writer) const final

  Function serializes print via provided Writer.
- bool IsSameAs (const Print &print) const

  Returns `true` when argument is same object.
- bool operator== (const Print &f) const

  Compares if Print is equal with other one.
- bool operator!= (const Print &f) const

  Compares if Print is not equal with other one.

Static Public Member Functions

- static std::shared_ptr< Print > FromIsoImage (const std::vector< uint8_t > &isoImage)

  Function creates shared pointer of the Print object from ISO image based on ISO/IEC 19794-4 (Finger image data).
- static uint8_t CalculateIntensity (const std::shared_ptr< inno::Image > &image, std::shared_ptr< PrintExecutor > &executor)

  Function calculates the image intensity.
- static std::shared_ptr< Print > Create (std::shared_ptr< Image > img, const PrintPosition &position, const PrintAttributes &pa=PrintAttributes(), std::shared_ptr< PrintExecutor > executor=GlobalPrintExecutor::Get())

  Constructor creates Print object.

Public Attributes

- const std::shared_ptr< Image > **image**

  Provides Print Image.
- const PrintPosition **position**

  Provides Print position.
- const PrintAttributes **printAttributes**

  Provides PrintCapture Image.

4.168.1 Member Function Documentation

4.168.1.1 CalculateImageQuality()

```
uint8_t CalculateImageQuality () [inline]
```

Image quality number is calculated in accordance with the general guidelines contained in Section 7.2.5 of ANSI/↩ INCITS 381 standard (Section 8.3.7.3 Quality score of ISO/IEC 19794-4 international standard). the value is available in serialization.

Returns

the output range is from 0 (lowest quality) to 100 (highest quality)

Exceptions

EnrollmentException	when fails to calculate image quality.

4.168.1.2 CalculateIntensity() [1/2]

```
uint8_t CalculateIntensity () [inline]
```

The intensity value is low for dry fingerprints (and for low pressure prints) and high for wet fingerprints (or for high pressure prints).

Returns

image intensity in range 0 (lowest intensity, dry finger, low pressure) to 100 (highest intensity, wet finger, high pressure)

Exceptions

EnrollmentException	when fails to calculate intensity

4.168.1.3 CalculateIntensity() [2/2]

```
uint8_t CalculateIntensity (
            const std::shared_ptr< inno::Image > & image,
            std::shared_ptr< PrintExecutor > & executor) [inline], [static]
```

The intensity value is low for dry fingerprints (and for low pressure prints) and high for wet fingerprints (or for high pressure prints).

Returns

image intensity in range 0 (lowest intensity, dry finger, low pressure) to 100 (highest intensity, wet finger, high pressure)

Exceptions

EnrollmentException	when fails to calculate intensity

Parameters

image	image to analyze
executor	print executor

4.168.1.4 CalculateNFIQ1()

```
uint8_t CalculateNFIQ1 () [inline]
```

This image quality number is calculated in accordance with the general guidelines contained in Section 7.2.5 of ANSI/INCITS 381 standard (Section 8.3.7.3 Quality score of ISO/IEC 19794-4 international standard). the value is available in serialization.

Returns

the output range is from 1 (highest quality) to 5 (lowest quality)

Exceptions

EnrollmentException	when fails to calculate NFIQ1

4.168.1.5 CalculateNFIQ2()

```
uint8_t CalculateNFIQ2 () [inline]
```

This image quality number is calculated in accordance with the general guidelines contained in Section 7.2.5 of ANSI/INCITS 381 standard (Section 8.3.7.3 Quality score of ISO/IEC 19794-4 international standard). the value is available in serialization. The NFIQ2 version is `2.3.0`.

Warning

NFIQ 2 is developed for images captured at 500 dpi and as such it shall not be used for images of different resolution, e.g. 1000 dpi. NFIQ 2 is developed for plain impression captured using optical sensors or scanned from inked card. Therefore, it shall not be used for images captured using other capture technologies, e.g. capacitive. [1]

Returns

the output range is from 0 (lowest quality) to 100 (highest quality)

Exceptions

EnrollmentException	when fails to calculate NFIQ2
InvalidImageArgumentException	when print image has different DPI as 500.

4.168.1.6 CalculatePlacement()

```
uint8_t CalculatePlacement () [inline]
```

The placement score number ranges from 0 to 100 and represents the fitness of the finger placement, evaluated with respect to the detected fingerprint classification (arch, loop, whorl), the value is available in serialization.

Returns

the output range is from 0 (lowest quality) to 100 (highest quality)

Exceptions

EnrollmentException	when fails to calculate placement

4.168.1.7 CalculatePresence()

```
uint8_t CalculatePresence () [inline]
```

Image quality number is calculated as an area where presence of fingerprint is detected. This function does not measure quality of the presented fingerprint pattern.Presence value is calculated from the fingerprint image, the value is available in serialization.

Returns

the output range is from 0 (lowest quality) to 100 (highest quality)

Exceptions

EnrollmentException	when fails to calculate presence

4.168.1.8 ChangeDpiTo() [1/2]

```
std::shared_ptr< Print > ChangeDpiTo (
            const Image::ImageResampler & resampler) const  [inline], [nodiscard]
```

It creates new print with changed image dpi by function Image::ChangeDpiTo() and with same position and attributes using a DPI-change configuration.

Parameters

resampler	image resampler

Returns

print with given DPI

Exceptions

ImageException	when there is an processing error
EnrollmentInvalidArgumentException	when resize algorithm is NIST and print image is not single-channel grayscale

4.168.1.9 ChangeDpiTo() [2/2]

```
std::shared_ptr< Print > ChangeDpiTo (
            ImageDpi targetDpi) const  [inline], [nodiscard]
```

It creates new print with changed image dpi by function Image::ChangeDpiTo() and with same position and attributes. The function uses Standard resizing algorithm. See BicubicImageResampler.

Parameters

targetDpi	DPI of new print.

Returns

print with target DPI

Exceptions

ImageException	when it is not possible to change image DPI
EnrollmentInvalidArgumentException	when targetDpi is 0

4.168.1.10 Create()

```
std::shared_ptr< Print > Create (
            std::shared_ptr< Image > img,
            const PrintPosition & position,
            const PrintAttributes & pa = PrintAttributes(),
            std::shared_ptr< PrintExecutor > executor = GlobalPrintExecutor::Get())
[inline], [static]
```

Parameters

img	print image
position	PrintPosition
pa	PrintAttributes
executor	PrintExecutor to use for Print functionality execution

Exceptions

NullArgumentException	when executor is null
NullArgumentException	when image is null

Returns

created Print object

4.168.1.11 DrawCriticalPoints()

```
std::shared_ptr< Image > DrawCriticalPoints (
            const std::vector< std::shared_ptr< CriticalPoint > > & points,
            const ImageShapeAttributes & attr) const [inline], [nodiscard]
```

Parameters

points	critical points to be drawn
attr	ImageShapeAttributes of minutiae

Returns

Image with drawn critical points

4.168.1.12 DrawMinutiae()

```
std::shared_ptr< Image > DrawMinutiae (
            const std::vector< std::shared_ptr< Minutia > > & minutiae,
            const ImageShapeAttributes & attr) const [inline], [nodiscard]
```

Parameters

minutiae	Minutiae to be drawn
attr	ImageShapeAttributes of minutiae

Returns

Image with drawn minutiae

4.168.1.13 FromIsoImage()

```
std::shared_ptr< Print > FromIsoImage (
            const std::vector< uint8_t > & isoImage)  [inline], [static]
```

[5]

Parameters

isoImage	ISO image from ISO/IEC 19794-4 (Finger image data)

Returns

shared pointer of the Print object

Exceptions

ImageException	when type is not supported or image data are corrupted
EnrollmentInvalidArgumentException	when it is not possible to decode image
EnrollmentInvalidArgumentException	for invalid conversion from ISO to PrintPosition
EnrollmentInvalidArgumentException	for invalid conversion from ISO to PrintImpression
EnrollmentInvalidArgumentException	for invalid conversion from ISO to PrintCaptureDeviceTechnology
NotSupportedException	when horizontal and vertical image dpi is different
NotSupportedException	when bit depth for raw image is not 8

4.168.1.14 IsSameAs()

```
bool IsSameAs (
            const Print & print) const  [inline]
```

Parameters

print	to check if it is reference to same object

Returns

`true` when given print is the same object

`false` when given print is a different object

4.168.1.15 operator"!=()

```
bool operator!= (
            const Print & f) const  [inline]
```

Parameters

f	compared Print

Returns

`true` when data of Print are different

`false` when data of Print are same

4.168.1.16 operator==()

```
bool operator== (
            const Print & f) const  [inline]
```

Parameters

f	compared Print

Returns

`true` when data of Print are same

`false` when data of Print are different

4.168.1.17 ToIsoImage()

```
std::vector< uint8_t > ToIsoImage (
            ImageEncoder & encoder) const  [inline]
```

The ISO standard specifies a data record interchange format for storing, recording, and transmitting the information from one or more finger or palm image areas [5]

Parameters

encoder	specific encoder for encoding Print image. Supported WSQ, JPG (lossy), JPEG 2000 (lossy & lossless), PNG

Returns

ISO image from ISO/IEC 19794-4 (Finger image data)

Exceptions

ImageException	when it is not possible to encode image
EnrollmentInvalidArgumentException	when encoder is not supported
EnrollmentRuntimeException	for invalid conversion size

4.168.1.18 ToIsoRawImage()

```
std::vector< uint8_t > ToIsoRawImage () const  [inline]
```

Returns

ISO raw image from ISO/IEC 19794-4 (Finger image data)

Exceptions

EnrollmentRuntimeException	for invalid conversion size

4.168.1.19 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize Print.

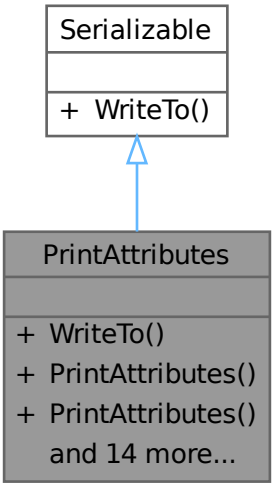
Implements Serializable.

4.169 PrintAttributes Class Reference

Print attributes Holds the print attributes.

```
#include <PrintAttributes.h>
```

Collaboration diagram for PrintAttributes:



Public Member Functions

- void WriteTo (Writer &writer) const final

  Serialize attributes in writer.
- **PrintAttributes** ()=default

  Default constructor of print attributes.
- PrintAttributes (const PrintAttributes &pa)

  Copy constructor of print attributes.
- PrintAttributes (PrintAttributes &&pa) noexcept

  Move constructor of print attributes.
- PrintAttributes & operator= (const PrintAttributes &pa)

  Assign the print attributes.
- PrintAttributes & operator= (PrintAttributes &&pa) noexcept

  Assign the print attributes.
- constexpr bool operator== (const PrintAttributes &pa) const

  Compares the print attributes.
- void SetImpression (PrintImpression imp)

  Sets impression of print.
- PrintImpression GetImpression () const

  Provides impression of print.
- void SetCaptureDeviceTechnology (PrintCaptureDeviceTechnology technology)

  Sets capture device technology.
- PrintCaptureDeviceTechnology GetCaptureDeviceTechnology () const

  Provides capture device technology.
- void SetCaptureDeviceVendorID (PrintCaptureDeviceVendorID vendor)

  Sets capture device vendor ID.
- PrintCaptureDeviceVendorID GetCaptureDeviceVendorID () const

Provides capture device vendor ID.

- void SetCaptureDeviceTypeID (PrintCaptureDeviceTypeID type)

  Sets capture device type ID.

- PrintCaptureDeviceTypeID GetCaptureDeviceTypeID () const

  Provides capture device type ID.

- void SetCertificationFlag (bool deviceCertified)

  Sets the certification flag.

- bool GetCertificationFlag () const

  Provides certification flag Indicates that the equipment was certified to comply with Appendix F (IAFIS Image Quality Specification, January 29, 1999) of CJIS-RS-0010, the Federal Bureau of Investigations's Electronic Fingerprint Transmission Specification.

4.169.1 Detailed Description

Enables to set each print attribute separately. In case, the attributes are not set the default values are used:

- print impression = PrintImpression::UNKNOWN
- capture device technology = PrintCaptureDeviceTechnology::UNKNOWN
- capture device vendor ID = 0
- capture device type ID = 0
- certification flag = false

4.169.2 Constructor & Destructor Documentation

4.169.2.1 PrintAttributes() [1/2]

```
PrintAttributes (
            const PrintAttributes & pa)  [inline]
```

Parameters

pa	Print attributes

4.169.2.2 PrintAttributes() [2/2]

```
PrintAttributes (
            PrintAttributes && pa)  [inline], [noexcept]
```

Parameters

pa	Print attributes

4.169.3 Member Function Documentation

4.169.3.1 GetCaptureDeviceTechnology()

```
PrintCaptureDeviceTechnology GetCaptureDeviceTechnology () const  [inline], [nodiscard]
```

Returns

Capture device technology

4.169.3.2 GetCaptureDeviceTypeID()

```
PrintCaptureDeviceTypeID GetCaptureDeviceTypeID () const  [inline], [nodiscard]
```

Returns

Capture device type ID

4.169.3.3 GetCaptureDeviceVendorID()

```
PrintCaptureDeviceVendorID GetCaptureDeviceVendorID () const  [inline], [nodiscard]
```

Returns

Capture device vendor ID

4.169.3.4 GetCertificationFlag()

```
bool GetCertificationFlag () const  [inline], [nodiscard]
```

[2]

Returns

Certification flag

4.169.3.5 GetImpression()

```
PrintImpression GetImpression () const  [inline], [nodiscard]
```

Returns

Impression of print

4.169.3.6 operator=() [1/2]

```
PrintAttributes & operator= (
            const PrintAttributes & pa)  [inline]
```

Parameters

pa	Print attributes

Returns

this modified object

4.169.3.7 operator=() [2/2]

```
PrintAttributes & operator= (
            PrintAttributes && pa)  [inline], [noexcept]
```

Parameters

pa	Print attributes

Returns

this modified object

4.169.3.8 operator==()

```
bool operator== (
            const PrintAttributes & pa) const  [inline], [constexpr]
```

Parameters

pa	Print attributes

Returns

`true` when given PrintAttributes are same as this one

`false` otherwise

4.169.3.9 SetCaptureDeviceTechnology()

```
void SetCaptureDeviceTechnology (
            PrintCaptureDeviceTechnology technology)  [inline]
```

Parameters

technology	Capture device technology

4.169.3.10 SetCaptureDeviceTypeID()

```
void SetCaptureDeviceTypeID (
            PrintCaptureDeviceTypeID type)  [inline]
```

Parameters

type	Capture device type ID

4.169.3.11 SetCaptureDeviceVendorID()

```
void SetCaptureDeviceVendorID (
            PrintCaptureDeviceVendorID vendor)  [inline]
```

Parameters

vendor	Capture device vendor ID

4.169.3.12 SetCertificationFlag()

```
void SetCertificationFlag (
            bool deviceCertified)  [inline]
```

Parameters

deviceCertified	Indicates that the equipment was certified to comply with Appendix F (IAFIS Image Quality Specification, January 29, 1999) of CJIS-RS-0010, the Federal Bureau of Investigations's Electronic Fingerprint Transmission Specification. [2]

4.169.3.13 SetImpression()

```
void SetImpression (
            PrintImpression imp) [inline]
```

Parameters

imp	Impression of print

4.169.3.14 WriteTo()

```
void WriteTo (
            Writer & writer) const [inline], [final], [virtual]
```

Parameters

writer	Writer for serialization

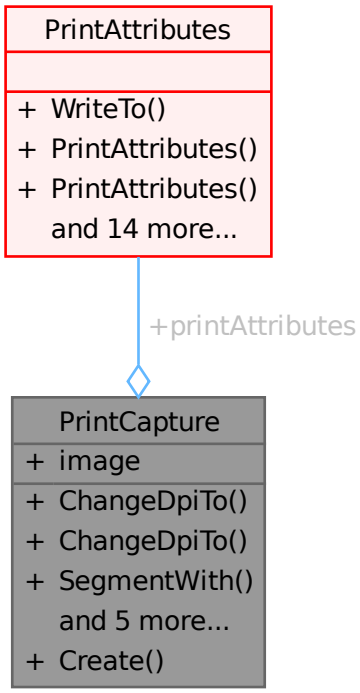
Implements Serializable.

4.170 PrintCapture Class Reference

It holds all functions and data related to print manipulation.

`#include <PrintCapture.h>`

Collaboration diagram for PrintCapture:



Public Member Functions

- std::shared_ptr< PrintCapture > ChangeDpiTo (ImageDpi targetDpi) const

  Create new PrintCapture with given DPI.
- std::shared_ptr< PrintCapture > ChangeDpiTo (const Image::ImageResampler &resampler) const

  Create new PrintCapture using provided DPI-change configuration.
- virtual std::vector< std::shared_ptr< PrintSegment > > SegmentWith (PrintCaptureSegmentation &detector)

  It segments print capture.
- std::shared_ptr< Image > CropPrint (const Rectangle &rectangle, const PrintCropConfiguration &cfg) const final

  It crops image with given rectangle and background color.
- std::shared_ptr< Image > CropPrint (const Rectangle &rectangle) const

  It crops image with given rectangle by default PrintCropConfiguration.
- std::shared_ptr< Image > Draw (const std::vector< std::shared_ptr< PrintSegment > > &segments, const ImageShapeAttributes &attr) const

  It draws rounding boxes into PrintCapture.
- void WriteTo (Writer &writer) const final

  Function serializes print via provided Writer.
- uint8_t CalculateIntensity ()

  Function calculates the image intensity.

Static Public Member Functions

- static std::shared_ptr< PrintCapture > Create (std::shared_ptr< Image > img, const PrintAttributes &pa=PrintAttributes(), std::shared_ptr< PrintExecutor > executor=GlobalPrintExecutor::Get())

  Constructor of PrintCapture.

Public Attributes

- const std::shared_ptr< Image > **image**

  Provides PrintCapture Image.
- const PrintAttributes **printAttributes**

  Provides PrintCapture Image.

4.170.1 Detailed Description

See Description with examples.

4.170.2 Member Function Documentation

4.170.2.1 CalculateIntensity()

```
uint8_t CalculateIntensity () [inline]
```

The intensity value is low for dry fingerprints (and for low pressure prints) and high for wet fingerprints (or for high pressure prints).

Returns

image intensity in range 0 (lowest intensity, dry finger, low pressure) to 100 (highest intensity, wet finger, high pressure)

Exceptions

EnrollmentException	when fails to calculate intensity

4.170.2.2 ChangeDpiTo() [1/2]

```
std::shared_ptr< PrintCapture > ChangeDpiTo (
            const Image::ImageResampler & resampler) const  [inline], [nodiscard]
```

It creates new print with changed image dpi by function Image::ChangeDpiTo() and with same attributes using a DPI-change configuration.

Parameters

resampler	image resampler

Returns

print capture with given DPI

Exceptions

ImageException	when there is an processing error
EnrollmentInvalidArgumentException	when resize algorithm is NIST and print capture image is not single-channel grayscale

4.170.2.3 ChangeDpiTo() [2/2]

```
std::shared_ptr< PrintCapture > ChangeDpiTo (
            ImageDpi targetDpi) const  [inline], [nodiscard]
```

It creates new print with changed image dpi by function Image::ChangeDpiTo() and with same position and attributes. The function uses Standard resizing algorithm. See BicubicImageResampler.

Parameters

targetDpi	DPI of new print.

Returns

print with target DPI

Exceptions

ImageException	when it is not possible to change image DPI
EnrollmentInvalidArgumentException	when targetDpi is 0

4.170.2.4 Create()

```
std::shared_ptr< PrintCapture > Create (
            std::shared_ptr< Image > img,
            const PrintAttributes & pa = PrintAttributes(),
            std::shared_ptr< PrintExecutor > executor = GlobalPrintExecutor::Get())
[inline], [static]
```

Parameters

img	print image, can contain one or more fingerprints to segment
pa	optional PrintAttributes
executor	PrintExecutor to use for PrintCapture functionality execution

Exceptions

NullArgumentException	when image is null

Returns

created PrintCapture object

4.170.2.5 CropPrint() [1/2]

```
std::shared_ptr< Image > CropPrint (
            const Rectangle & rectangle) const  [inline], [nodiscard]
```

Parameters

rectangle	Rectangle to be cropped

Returns

cropped Image

4.170.2.6 CropPrint() [2/2]

```
std::shared_ptr< Image > CropPrint (
            const Rectangle & rectangle,
            const PrintCropConfiguration & cfg) const  [inline], [nodiscard], [final]
```

Parameters

rectangle	Rectangle to be cropped
cfg	configuration of print cropping

Returns

Image cropped image

4.170.2.7 Draw()

```
std::shared_ptr< Image > Draw (
            const std::vector< std::shared_ptr< PrintSegment > > & segments,
            const ImageShapeAttributes & attr) const  [inline]
```

Parameters

segments	detected segments
attr	ImageShapeAttributes of rounding boxes

Returns

Image with drawn rounding boxes

4.170.2.8 SegmentWith()

```
virtual std::vector< std::shared_ptr< PrintSegment > > SegmentWith (
            PrintCaptureSegmentation & detector)  [inline], [virtual]
```

It uses PrintDetector::Segment function to find print segments. It always returns segments without position. The user MUST assign position to segment via function PrintSegment::SetPosition to be possible to transform PrintSegment into Print.

Parameters

detector	PrintDetector to detect PrintSegments in PrintCapture

Returns

detected print segments

Exceptions

EnrollmentException	from PrintDetector

Reimplemented in SlapCapture, and ThumbsCapture.

4.170.2.9 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize PrintCapture.

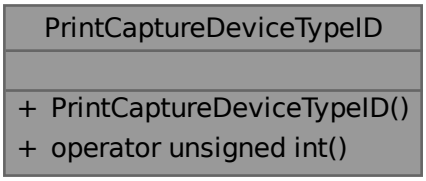
Implements Serializable.

4.171 PrintCaptureDeviceTypeID Class Reference

Capture device type identifier The capture device type identifier shall identify the biometric organization that owns the product that created the BDIR.

```
#include <PrintCaptureDeviceTypeID.h>
```

Collaboration diagram for PrintCaptureDeviceTypeID:



Public Member Functions

- PrintCaptureDeviceTypeID (unsigned int i)

  Construct a new Print Capture Equipment ID.
- **operator unsigned int** () const

  Conversion to capture equipment id number.

4.171.1 Detailed Description

The capture device algorithm type identifier shall be encoded in two bytes carrying a CBEFF biometric organization identifier (registered by IBIA or other approved registration authority). A value of all zeros shall indicate that the capture device type is unreported. [5]

4.171.2 Constructor & Destructor Documentation

4.171.2.1 PrintCaptureDeviceTypeID()

```
PrintCaptureDeviceTypeID (
            unsigned int i)  [inline], [explicit]
```

Parameters

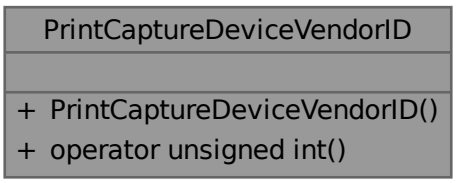
i	equipment ID determined by vendor. Only 16 bits are stored.

4.172 PrintCaptureDeviceVendorID Class Reference

Capture device vendor identifier The capture device vendor identifier shall identify the biometric organization that owns the product that created the BDIR.

```
#include <PrintCaptureDeviceVendorID.h>
```

Collaboration diagram for PrintCaptureDeviceVendorID:



Public Member Functions

- PrintCaptureDeviceVendorID (unsigned int i)

  Construct a new Print Capture Equipment ID.
- **operator unsigned int** () const

  Conversion to capture equipment id number.

4.172.1 Detailed Description

The capture device algorithm vendor identifier shall be encoded in two bytes carrying a CBEFF biometric organization identifier (registered by IBIA or other approved registration authority). A value of all zeros shall indicate that the capture device vendor is unreported. [5]

4.172.2 Constructor & Destructor Documentation

4.172.2.1 PrintCaptureDeviceVendorID()

```
PrintCaptureDeviceVendorID (
            unsigned int i)  [inline], [explicit]
```

Parameters

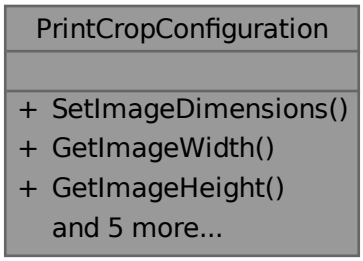
i	equipment ID determined by vendor. Only 16 bits are stored.

4.173 PrintCropConfiguration Class Reference

Holds print cropping configuration.

```
#include <PrintSegment.h>
```

Collaboration diagram for PrintCropConfiguration:



Public Member Functions

- void SetImageDimensions (ImageWidth width, ImageHeight height)

  Function sets output image dimension.
- ImageWidth GetImageWidth () const

  It gets output image width.
- ImageHeight GetImageHeight () const

  It gets output image height.
- void SetBackgroundColor (RGBA color)

  It sets background color rounding box area that is out of image.
- RGBA GetBackgroundColor () const

  It provides background color for rounding box area that is out of image.
- void EnableAlignedCropping (bool aligned)

  Enable Aligned Cropping.
- bool IsAlignedCroppingEnabled () const

  Get Aligned Crop Configuration.
- PrintCropConfiguration ()=default

  Create default PrintCropConfiguration object.

4.173.1 Constructor & Destructor Documentation

4.173.1.1 PrintCropConfiguration()

```
PrintCropConfiguration () [default]
```

Default configuration is

- image width - will be calculated according to cropped rectangle
- image height - will be calculated according to cropped rectangle
- background color - Colors::White
- aligned cropping - true

4.173.2 Member Function Documentation

4.173.2.1 EnableAlignedCropping()

```
void EnableAlignedCropping (
            bool aligned) [inline]
```

This function activates aligned cropping based on the provided argument.

Parameters

aligned	enable aligned crop • When set to true, the cropping aligns the rotated rectangle, positioning its top-left point at [0,0] in the cropped image. • When set to false, the cropping involves the bounding rectangle of the rotated rectangle.

4.173.2.2 GetBackgroundColor()

```
RGBA GetBackgroundColor () const [inline], [nodiscard]
```

Returns

background color for rounding box area that is out of image

4.173.2.3 GetImageHeight()

```
ImageHeight GetImageHeight () const [inline], [nodiscard]
```

Returns

output image height

4.173.2.4 GetImageWidth()

```
ImageWidth GetImageWidth () const  [inline], [nodiscard]
```

Returns

output image width

4.173.2.5 IsAlignedCroppingEnabled()

```
bool IsAlignedCroppingEnabled () const  [inline], [nodiscard]
```

Returns

The current state of the aligned cropping flag.

4.173.2.6 SetBackgroundColor()

```
void SetBackgroundColor (
            RGBA color)  [inline]
```

Parameters

color	background color

4.173.2.7 SetImageDimensions()

```
void SetImageDimensions (
            ImageWidth width,
            ImageHeight height)  [inline]
```

The given rectangle is cropped from image and paste to output image with given dimensions.

Parameters

width	- width of output image
height	- height of output image

Exceptions

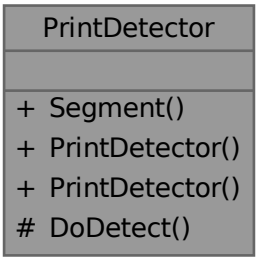
EnrollmentInvalidArgumentException	when width is 0
EnrollmentInvalidArgumentException	when height is 0

4.174 PrintDetector Class Reference

PrintDetector implements segmentation by means of Innovatrics ISegLib library.

```
#include <PrintDetector.h>
```

Collaboration diagram for PrintDetector:



Public Member Functions

- std::vector< std::shared_ptr< PrintSegment > > Segment (const std::shared_ptr< PrintCapture > &capture) final

  Detects segments in given image The Function scans the provided capture to locate fingers.
- **PrintDetector** ()

  Construct a new PrintDetector object GlobalPrintExecutor is used for PrintSegment detection.
- PrintDetector (std::shared_ptr< PrintExecutor > executor)

  Construct a new PrintDetector object.

Protected Member Functions

- virtual inno_PrintDetector_result DoDetect (const std::shared_ptr< PrintExecutor > &executor, const std::↩ shared_ptr< Image > &image, FormattingLogger &logger)=0

  Template method for various segmentation strategies.

4.174.1 Detailed Description

It is class that defines template method PrintDetector::DoDetect for various segmentation strategies.

4.174.2 Constructor & Destructor Documentation

4.174.2.1 PrintDetector()

```
PrintDetector (
            std::shared_ptr< PrintExecutor > executor)  [inline], [explicit]
```

Parameters

executor	PrintExecutor to use for PrintSegment detection

Exceptions

NullArgumentException	when executor is null

4.174.3 Member Function Documentation

4.174.3.1 DoDetect()

```
virtual inno_PrintDetector_result DoDetect (
            const std::shared_ptr< PrintExecutor > & executor,
            const std::shared_ptr< Image > & image,
            FormattingLogger & logger)  [protected], [pure virtual]
```

Parameters

executor	PrintExecutor to use for PrintSegment detection
image	Image to segment
logger	FormattingLogger to log messages

Returns

inno_PrintDetector_result result of segmentation from binary part of library

4.174.3.2 Segment()

```
std::vector< std::shared_ptr< PrintSegment > > Segment (
            const std::shared_ptr< PrintCapture > & capture)  [inline], [final]
```

For each identified finger, it estimates its position on the hand by comparing the x-coordinate of the center of the finger's bounding box rectangle. As a result, the estimation may not be precise for slaps with an angle greater than 45 degrees and is not suitable for PrintCapture types other than Slap or Thumbs.

Parameters

capture	PrintCapture

Returns

slap segments

Exceptions

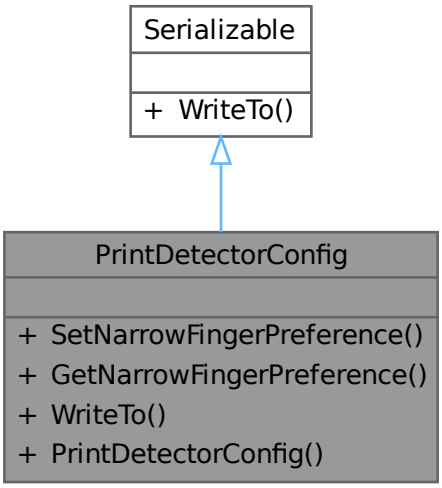
EnrollmentException	when ISegLib returns error

4.175 PrintDetectorConfig Class Reference

General configuration of PrintDetector.

```
#include <PrintDetector.h>
```

Collaboration diagram for PrintDetectorConfig:



Public Member Functions

- void SetNarrowFingerPreference (bool expectNarrow)

  Informs the print detector to anticipate narrow fingers.
- bool GetNarrowFingerPreference () const

  Retrieves information about the expected narrowness of detected fingers.
- void WriteTo (Writer &writer) const override

  Function serializes configuration via provided Writer.
- PrintDetectorConfig ()=default

  Default PrintDetector configuration is.

4.175.1 Constructor & Destructor Documentation

4.175.1.1 PrintDetectorConfig()

```
PrintDetectorConfig () [default]
```

- expectNarrowFinger - true

4.175.2 Member Function Documentation

4.175.2.1 GetNarrowFingerPreference()

```
bool GetNarrowFingerPreference () const [inline], [nodiscard]
```

Returns

true if narrow detected fingers are expected

false if narrow detected fingers are not expected

4.175.2.2 SetNarrowFingerPreference()

```
void SetNarrowFingerPreference (
            bool expectNarrow) [inline]
```

It recommends returning a narrow bounding rectangle even if the detected finger is initially identified as wide. This adjustment is particularly relevant in cases where the finger is rotated, causing a misalignment between the top-left rectangle point and the corresponding point from the finger's perspective. The function then adjusts the coordinates to narrow down the representation of the finger, estimating rectangle points (top-left, top-right, bottom-right, bottom-left) based on the fingerprint's viewpoint. Since there is limited information about the orientation of the fingerprint, this estimation involves some guesswork. It's important to note that the detector provides only the rectangle without indicating the direction of the finger within the bounding rectangle. This means that after alignment, some fingerprints may appear upside down. The recommended adjustment of the top-left point from the finger's perspective plays a crucial role in aligning the cropped finger accurately.

Parameters

expectNarrow	When set to true, the detector will consistently output a narrow bounding rectangle for the detected finger. In contrast, if expectNarrow is false, the detector will refrain from modifying the rectangle and will maintain its original dimensions.

4.175.2.3 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [override], [virtual]
```

Parameters

writer	used to serialize Config.

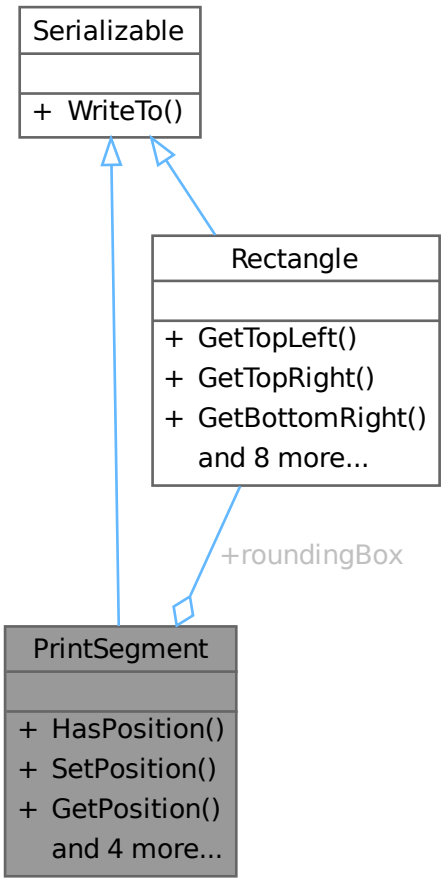
Implements Serializable.

4.176 PrintSegment Class Reference

Segment detected in print capture Usually the segment has also position that is assigned by segmentation algorithm.

```
#include <PrintSegment.h>
```

Collaboration diagram for PrintSegment:



Public Member Functions

- bool HasPosition () const

  It checks if position is set.
- void SetPosition (const PrintPosition &position)

  Set the Position.
- PrintPosition GetPosition ()

  Returns print position.
- std::shared_ptr< PrintSegment > Resize (Rectangle::Width newWidth, Rectangle::Height newHeight) const

  Create new segment with given width and height.
- std::shared_ptr< Print > ToPrint ()

  It transforms segment to Print.
- std::shared_ptr< Print > ToPrint (const PrintCropConfiguration &cfg)

  It transforms segment to Print.
- void WriteTo (Writer &writer) const final

  Function serializes slap segment.

Public Attributes

- const Rectangle **roundingBox**

  Found rounding box of segmented fingertip.

4.176.1 Detailed Description

In rare cases the segmentation algorithm does not detect position. Then it is necessary that user sets correct position to be possible to create Print for further processing.

4.176.2 Member Function Documentation

4.176.2.1 GetPosition()

```
PrintPosition GetPosition () [inline]
```

Returns

print position

Exceptions

EnrollmentRuntimeException	when no print position

4.176.2.2 HasPosition()

```
bool HasPosition () const [inline]
```

Returns

`true` when position is set

`false` when position is not set

4.176.2.3 Resize()

```
std::shared_ptr< PrintSegment > Resize (
            Rectangle::Width newWidth,
            Rectangle::Height newHeight) const [inline]
```

The center and angle of segment will remain same.

Parameters

newWidth	new width of segment
newHeight	new height of segment

Returns

new segment with corresponding width and high

4.176.2.4 SetPosition()

```
void SetPosition (
            const PrintPosition & position) [inline]
```

Parameters

position	PrintPosition of fingertip

4.176.2.5 ToPrint() [1/2]

```
std::shared_ptr< Print > ToPrint () [inline]
```

Usually the segment has also position that is assigned by segmentation algorithm. In rare cases the segmentation algorithm does not detect position. Then it is necessary that user sets correct position to be possible to create Print for further processing. Default PrintCropConfiguration is used for cropping.

Returns

Print

Exceptions

EnrollmentException	when position is not set

4.176.2.6 ToPrint() [2/2]

```
std::shared_ptr< Print > ToPrint (
            const PrintCropConfiguration & cfg)  [inline]
```

Usually the segment has also position that is assigned by segmentation algorithm. In rare cases the segmentation algorithm does not detect position. Then it is necessary that user sets correct position to be possible to create Print for further processing.

Parameters

cfg	crop configuration

Returns

Print

Exceptions

EnrollmentRuntimeException	when position is not set

4.176.2.7 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize PrintSegment.

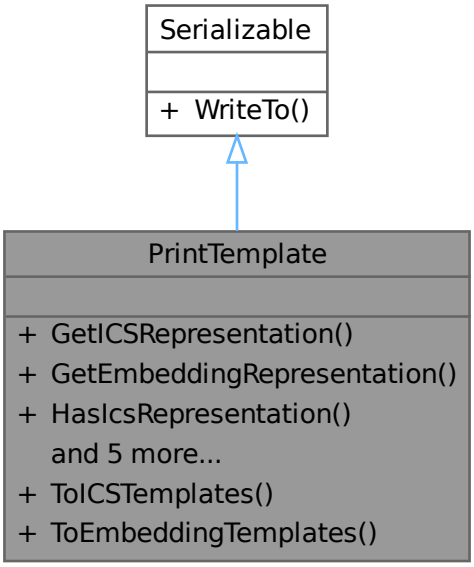
Implements Serializable.

4.177 PrintTemplate Class Reference

Holds various biometric representation types for print.

```
#include <PrintTemplate.h>
```

Collaboration diagram for PrintTemplate:



Public Member Functions

- std::shared_ptr< ICSTemplate > GetICSRepresentation () const

  Provides ICS (Innovatrics proprietary extracted template) representation of print.
- std::shared_ptr< EmbeddingTemplate > GetEmbeddingRepresentation () const

  Retrieves the embedding representation associated with the object.
- bool HasIcsRepresentation () const

  Checks if the object has an ICS representation.
- bool HasEmbeddingRepresentation () const

  Checks if the object has an embedding representation.
- void WriteTo (Writer &writer) const final

  It serializes applicant print via provided Writer.
- PrintTemplate (std::shared_ptr< ICSTemplate > icsTemplate)

  Constructs an PrintTemplate object with the provided ICSTemplate.
- PrintTemplate (std::shared_ptr< ICSTemplate > icsTemplate, std::shared_ptr< EmbeddingTemplate > embedding)

  Constructs an PrintTemplate object with the provided ICSTemplate and embedding.
- PrintTemplate (std::shared_ptr< EmbeddingTemplate > embedding)

  Constructs an PrintTemplate object with the provided embedding.

Static Public Member Functions

- static std::vector< std::shared_ptr< ICSTemplate > > ToICSTemplates (const std::vector< std::shared_ptr< PrintTemplate > > &templates)

  Converts a PrintTemplate objects into a ICSTemplate objects.
- static std::vector< std::shared_ptr< EmbeddingTemplate > > ToEmbeddingTemplates (const std::vector< std::shared_ptr< PrintTemplate > > &templates)

  Converts a PrintTemplate objects into an EmbeddingTemplate objects.

4.177.1 Constructor & Destructor Documentation

4.177.1.1 PrintTemplate() [1/3]

```
PrintTemplate (
            std::shared_ptr< ICSTemplate > icsTemplate)  [inline], [explicit]
```

This constructor initializes an PrintTemplate object with the given ICSTemplate.

Parameters

icsTemplate	A shared pointer to the ICSTemplate object to be associated with the PrintTemplate.

Exceptions

NullArgumentException	Thrown if the provided ICSTemplate pointer is null.

4.177.1.2 PrintTemplate() [2/3]

```
PrintTemplate (
            std::shared_ptr< ICSTemplate > icsTemplate,
            std::shared_ptr< EmbeddingTemplate > embedding)  [inline]
```

This constructor initializes an PrintTemplate object with the given ICSTemplate and EmbeddingTemplate.

Parameters

icsTemplate	A shared pointer to the ICSTemplate object to be associated with the PrintTemplate.
embedding	A shared pointer to the EmbeddingTemplate object to be associated with the PrintTemplate.

Exceptions

NullArgumentException	if the provided ICSTemplate pointer is null.
NullArgumentException	if the provided EmbeddingTemplate pointer is null.
EnrollmentInvalidArgumentException	if ICS template and Embedding positions are different.

4.177.1.3 PrintTemplate() [3/3]

```
PrintTemplate (
            std::shared_ptr< EmbeddingTemplate > embedding)  [inline], [explicit]
```

This constructor initializes an PrintTemplate object with the given EmbeddingTemplate.

Parameters

embedding	A shared pointer to the EmbeddingTemplate object to be associated with the PrintTemplate.

Exceptions

NullArgumentException	if the provided EmbeddingTemplate pointer is null.

4.177.2 Member Function Documentation

4.177.2.1 GetEmbeddingRepresentation()

```
std::shared_ptr< EmbeddingTemplate > GetEmbeddingRepresentation () const [inline],
[nodiscard]
```

This function retrieves the embedding representation associated with the object.

Returns

A EmbeddingTemplate associated with the object.

A nullptr if no embedding representation is associated.

4.177.2.2 GetICSRepresentation()

```
std::shared_ptr< ICSTemplate > GetICSRepresentation () const [inline], [nodiscard]
```

Returns

ICSTemplate

4.177.2.3 HasEmbeddingRepresentation()

```
bool HasEmbeddingRepresentation () const [inline], [nodiscard]
```

This function checks whether the object has an embedding representation associated with it.

Returns

`true` if the object has an embedding representation, `false` otherwise.

4.177.2.4 HasIcsRepresentation()

```
bool HasIcsRepresentation () const [inline], [nodiscard]
```

This function checks whether the object has an Innovatrics' proprietary representation associated with it.

Returns

`true` if the object has an ICS representation, `false` otherwise.

4.177.2.5 ToEmbeddingTemplates()

```
std::vector< std::shared_ptr< EmbeddingTemplate > > ToEmbeddingTemplates (
            const std::vector< std::shared_ptr< PrintTemplate > > & templates)  [inline],
[static]
```

This function takes a PrintTemplate objects and converts each element into an EmbeddingTemplate object, then returns a EmbeddingTemplate objects. When PrintTemplate does not contain EmbeddingTemplate then null is provided.

Parameters

templates	A PrintTemplate objects to be converted.

Returns

A EmbeddingTemplate objects converted from the PrintTemplate objects.

4.177.2.6 ToICSTemplates()

```
std::vector< std::shared_ptr< ICSTemplate > > ToICSTemplates (
            const std::vector< std::shared_ptr< PrintTemplate > > & templates)  [inline],
[static]
```

This function takes a PrintTemplate objects and converts each element into an ICSTemplate object, then returns a ICSTemplate objects.

Parameters

templates	A PrintTemplate objects to be converted. When PrintTemplate does not contain ICSTemplate then null is provided.

Returns

A ICSTemplate objects converted from the PrintTemplate objects.

4.177.2.7 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	used to serialize PrintTemplate.

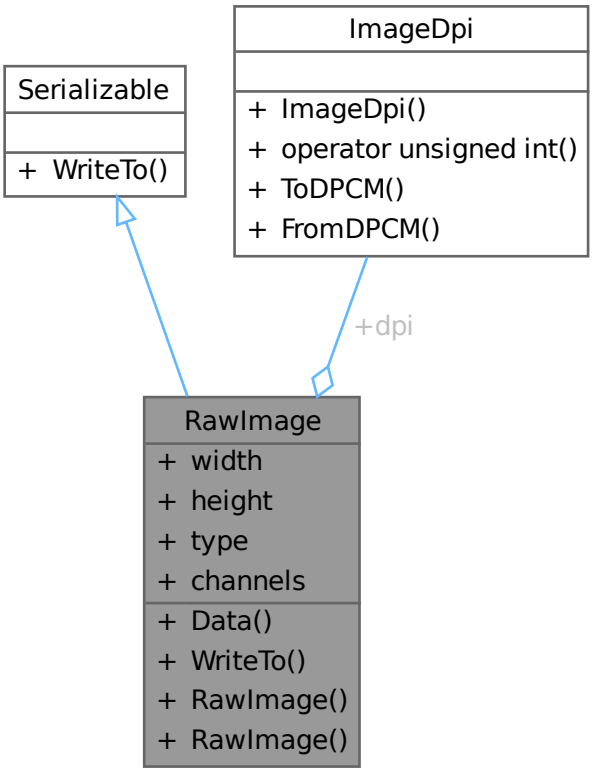
Implements Serializable.

4.178 RawImage Class Reference

Contains plain image data.

```
#include <ImageAttribute.h>
```

Collaboration diagram for RawImage:



Public Types

- enum Type : uint8_t { GRAYSCALE , BGR , BGRA }

  RawImage encoding type.

Public Member Functions

- const std::vector< uint8_t > & Data () const
- void WriteTo (Writer &writer) const final

  Write itself into writer.
- RawImage (ImageWidth w, ImageHeight h, ImageDpi d, Type t, std::vector< uint8_t > &&r)

  Construct a new Raw Image object Data vector is passed by rvalue reference for efficient transfer without unnecessary copying.
- RawImage (ImageWidth w, ImageHeight h, ImageDpi d, Type t, const std::vector< uint8_t > &r)

  Construct a new Raw Image object Data vector is stored as a copy.

Public Attributes

- const ImageDpi **dpi**

  Image DPI.
- const ImageWidth **width**

  Image width in pixels.
- const ImageHeight **height**

  Image height in pixels.
- const Type **type**

  RawImage type.
- const unsigned char **channels**

  Number of RawImage channels.

4.178.1 Member Enumeration Documentation

4.178.1.1 Type

```
enum Type : uint8_t
```

Enumerator

GRAYSCALE	Raw data where image pixels are stored in grayscale format. The data are in format where each pixel has 1 byte. Each byte represent color. Then pixels are stored by columns, it means that data starts with row index 1 and column index 1 pixel and ends with last row and last column pixel.
BGR	Raw data where image pixels are stored in BGR format. The data are in format where each pixel has 3 bytes. Each byte represent color in order Blue, Green, Red. Then pixels are stored by columns, it means that data starts with row index 1 and column index 1 pixel and ends with last row and last column pixel.
BGRA	Raw data where image pixels are stored in BGR with alpha format. The data are in format where each pixel has 4 bytes. Each byte represent color in order Blue, Green, Red and Alpha channel. Then pixels are stored by columns, it means that data starts with row index 1 and column index 1 pixel and ends with last row and last column pixel. RGBA stands for red green blue alpha. While it is sometimes described as a color space, it is actually a three-channel RGB color model supplemented with a fourth alpha channel. Alpha indicates how opaque each pixel is and allows an image to be combined over others using alpha compositing, with transparent areas and anti-aliasing of the edges of opaque regions. [11]

4.178.2 Constructor & Destructor Documentation

4.178.2.1 RawImage() [1/2]

```
RawImage (
            ImageWidth w,
            ImageHeight h,
            ImageDpi d,
            Type t,
            std::vector< uint8_t > && r)  [inline]
```

Parameters

w	Image Width
h	Image Height
d	Image DPI
t	Raw data type
r	raw data

4.178.2.2 RawImage() [2/2]

```
RawImage (
            ImageWidth w,
            ImageHeight h,
            ImageDpi d,
            Type t,
            const std::vector< uint8_t > & r)  [inline]
```

Parameters

w	Image Width
h	Image Height
d	Image DPI
t	Raw data type
r	raw data

4.178.3 Member Function Documentation

4.178.3.1 Data()

```
const std::vector< uint8_t > & Data () const  [inline], [nodiscard]
```

Returns

std::vector<uint8_t>

4.178.3.2 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

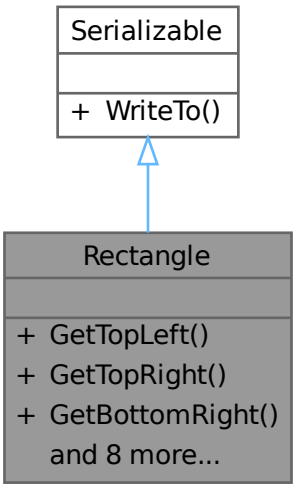
Implements Serializable.

4.179 Rectangle Class Reference

It holds rectangle vertexes, width, height.

```
#include <ImageAttribute.h>
```

Collaboration diagram for Rectangle:



Public Types

- using Dimension = float

  The rectangle dimension.
- using **Height** = Dimension

  Rectangle height.
- using **Width** = Dimension

  Rectangle width.

Public Member Functions

- const Point & GetTopLeft () const

  Provides top-left vertex.
- const Point & GetTopRight () const

  Provides top-right vertex.
- const Point & GetBottomRight () const

  Provides bottom-right vertex.
- const Point & GetBottomLeft () const

  Provides bottom-left vertex.
- Width GetWidth () const

  Provides width of rectangle The width is distance between top-left and top-right vertexes.

- Height GetHeight () const

  Provides height of rectangle The height is distance between bottom-left and top-left vertexes.
- Rectangle Resize (Width w, Height h) const

  Resize rectangle to given width and height.
- Point CalculateCenter () const

  Calculates the center point of the rectangle.
- void WriteTo (Writer &writer) const final

  Write itself into writer.
- Rectangle (Point topLeftPoint, Point topRightPoint, Point bottomRightPoint, Point bottomLeftPoint)

  Constructor store given coordinates.
- Rectangle (Point topLeftPoint, Point topRightPoint, Point bottomRightPoint, Point bottomLeftPoint, std::shared↩ _ptr< ImageExecutor > executor)

  Constructor store given coordinates.

4.179.1 Detailed Description

The vertexes may not be rectangular. When SDK need real rectangle for its functionality it will use minimal bounding rectangle around given vertexes.

4.179.2 Member Typedef Documentation

4.179.2.1 Dimension

```
using Dimension = float
```

It is not integer because rectangle can be rotated.

4.179.3 Constructor & Destructor Documentation

4.179.3.1 Rectangle() [1/2]

```
Rectangle (
            Point topLeftPoint,
            Point topRightPoint,
            Point bottomRightPoint,
            Point bottomLeftPoint)  [inline]
```

The GlobalImageExecutor is used.

Parameters

topLeftPoint	top left point
topRightPoint	top right point
bottomLeftPoint	bottom left point
bottomRightPoint	bottom right point

Exceptions

EnrollmentException	when rectangle dimensions can not be calculated.

4.179.3.2 Rectangle() [2/2]

```
Rectangle (
            Point topLeftPoint,
            Point topRightPoint,
            Point bottomRightPoint,
            Point bottomLeftPoint,
            std::shared_ptr< ImageExecutor > executor)  [inline]
```

Parameters

topLeftPoint	top left point
topRightPoint	top right point
bottomLeftPoint	bottom left point
bottomRightPoint	bottom right point
executor	ImageExecutor to modify rectangle

Exceptions

EnrollmentException	when rectangle dimensions can not be calculated.
NullArgumentException	when executor is null

4.179.4 Member Function Documentation

4.179.4.1 CalculateCenter()

```
Point CalculateCenter () const  [inline], [nodiscard]
```

Returns

Point representing the center of the rectangle.

4.179.4.2 GetBottomLeft()

```
const Point & GetBottomLeft () const  [inline], [nodiscard]
```

Returns

bottom-left vertex

4.179.4.3 GetBottomRight()

```
const Point & GetBottomRight () const  [inline], [nodiscard]
```

Returns

bottom-right vertex

4.179.4.4 GetHeight()

```
Height GetHeight () const  [inline], [nodiscard]
```

When given vertexes are not rectangular then minimal bounding rectangle is calculated and height of the rectangle is returned.

Returns

height

4.179.4.5 GetTopLeft()

```
const Point & GetTopLeft () const  [inline], [nodiscard]
```

Returns

top-left vertex

4.179.4.6 GetTopRight()

```
const Point & GetTopRight () const  [inline], [nodiscard]
```

Returns

top-right vertex

4.179.4.7 GetWidth()

```
Width GetWidth () const  [inline], [nodiscard]
```

When given vertexes are not rectangular then minimal bounding rectangle is calculated and width of the rectangle is returned.

Returns

width

4.179.4.8 Resize()

```
Rectangle Resize (
            Width w,
            Height h) const  [inline], [nodiscard]
```

Parameters

w	width of new rectangle
h	height of new rectangle

Returns

new resized rectangle

Exceptions

EnrollmentInvalidArgumentException	when width is 1 or less
EnrollmentInvalidArgumentException	when height is 1 or less
EnrollmentException	when it is not possible to resize rectangle

4.179.4.9 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

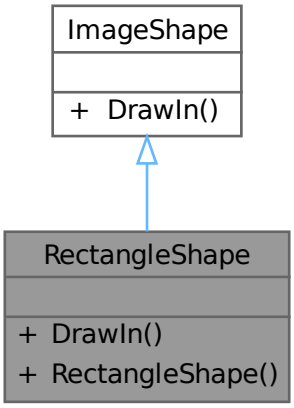
Implements Serializable.

4.180 RectangleShape Class Reference

It holds rectangle and draws it.

```
#include <ImageShape.h>
```

Collaboration diagram for RectangleShape:



Public Member Functions

- void DrawIn (ImageShapeDrawer &drawer) const final

  It draws rectangle in ImageShapeDrawer.
- RectangleShape (Rectangle rectangle, ImageShapeAttributes attr)

  Constructor sets given rectangle for drawing.

4.180.1 Constructor & Destructor Documentation

4.180.1.1 RectangleShape()

```
RectangleShape (
            Rectangle rectangle,
            ImageShapeAttributes attr)  [inline]
```

Parameters

rectangle	Rectangle
attr	ImageShapeAttributes use to draw rectangle

4.180.2 Member Function Documentation

4.180.2.1 DrawIn()

```
void DrawIn (
            ImageShapeDrawer & drawer) const  [inline], [final], [virtual]
```

It draws rectangle as 4 lines shape

Parameters

drawer	the implementation of ImageShapeDrawer

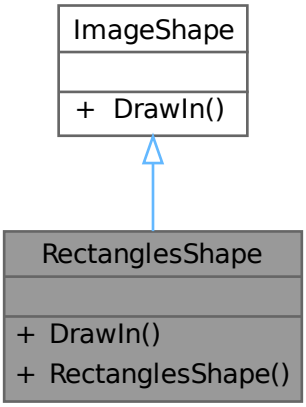
Implements ImageShape.

4.181 RectanglesShape Class Reference

It holds rectangles and draws it as one shape.

```
#include <ImageShape.h>
```

Collaboration diagram for RectanglesShape:



Public Member Functions

- void DrawIn (ImageShapeDrawer &drawer) const final

  It draws all rectangles at once in ImageShapeDrawer.
- RectanglesShape (std::vector< Rectangle > r, ImageShapeAttributes attr)

  Construct a new Rectangles object.

4.181.1 Constructor & Destructor Documentation

4.181.1.1 RectanglesShape()

```
RectanglesShape (
            std::vector< Rectangle > r,
            ImageShapeAttributes attr)  [inline]
```

Parameters

r	rectangles
attr	ImageShapeAttributes use to draw rectangles

4.181.2 Member Function Documentation

4.181.2.1 DrawIn()

```
void DrawIn (
            ImageShapeDrawer & drawer) const  [inline], [final], [virtual]
```

Parameters

drawer	the implementation of ImageShapeDrawer

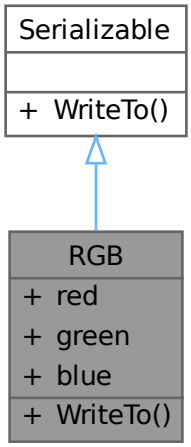
Implements ImageShape.

4.182 RGB Class Reference

It holds RGB colors.

```
#include <ImageAttribute.h>
```

Collaboration diagram for RGB:



Public Member Functions

- void WriteTo (Writer &writer) const final

  Write itself into writer.

Public Attributes

- unsigned char **red** = 0

  Red value of RGB color model. The value is in range <0,255>.
- unsigned char **green** = 0

  Green value of RGB color model. The value is in range <0,255>.
- unsigned char **blue** = 0

  Blue value of RGB color model. The value is in range <0,255>.

4.182.1 Detailed Description

The RGB colour model is an additive color model in which the red, green and blue primary colors of light are added together in various ways to reproduce a broad array of colors. The name of the model comes from the initials of the three additive primary colors, red, green, and blue. The main purpose of the RGB color model is for the sensing, representation, and display of images in electronic systems, such as televisions and computers, though it has also been used in conventional photography. Before the electronic age, the RGB color model already had a solid theory behind it, based in human perception of colors. [17]

4.182.2 Member Function Documentation

4.182.2.1 WriteTo()

```
void WriteTo (
            Writer & writer) const  [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

Implements Serializable.

4.183 RGBA Class Reference

It holds RGB color with transparency.

```
#include <ImageAttribute.h>
```

Collaboration diagram for RGBA:

Public Types

- enum : uint8_t { FULLY_TRANSPARENT = 0 , FULLY_OPAQUE = 255 }

Public Member Functions

- void WriteTo (Writer &writer) const final

  Write itself into writer.

Public Attributes

- RGB **color**

  color definition in RGB model
- unsigned char **transparency** = FULLY_OPAQUE

  Transparency where 0 is fully transparent and 255 is fully opaque.

4.183.1 Member Enumeration Documentation

4.183.1.1 anonymous enum

```
anonymous enum : uint8_t
```

Enumerator

FULLY_TRANSPARENT	Color is fully transparent, means it is not visible.
FULLY_OPAQUE	Color is fully opaque, means it is visible.

4.183.2 Member Function Documentation

4.183.2.1 WriteTo()

```
void WriteTo (
            Writer & writer) const [inline], [final], [virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

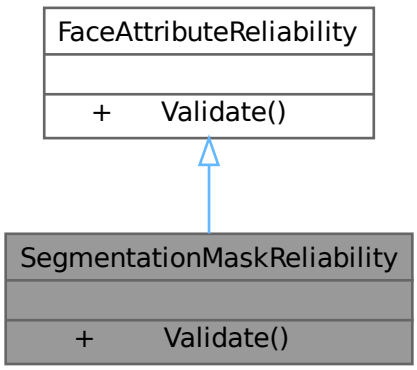
Implements Serializable.

4.184 SegmentationMaskReliability Class Reference

Check if calculation of Segmentation Mask in functions Face::FaceAreaSegment, Face::FaceAreaSegment(RGB), Face::FaceAreaMask will be reliable.

```
#include <FaceValidation.h>
```

Collaboration diagram for SegmentationMaskReliability:



Public Member Functions

- std::vector< FailedFaceAttribute > Validate (const std::shared_ptr< Face > &probe) final

  Check if calculation of Segmentation Mask in functions Face::FaceAreaSegment, Face::FaceAreaSegment(RGB), Face::FaceAreaMask will be reliable.

4.184.1 Member Function Documentation

4.184.1.1 Validate()

```
std::vector< FailedFaceAttribute > Validate (
            const std::shared_ptr< Face > & probe)  [inline], [final], [virtual]
```

Parameters

probe	Face to be checked

Returns

list of attributes that will cause not reliable Segmentation Mask.

Exceptions

NullArgumentException	when probe is null

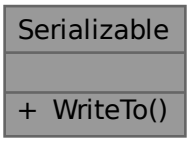
Implements FaceAttributeReliability.

4.185 Serializable Class Reference

Interface that MUST be implemented by all classes those want to be serialized.

```
#include <Writer.h>
```

Collaboration diagram for Serializable:



Public Member Functions

- virtual void WriteTo (Writer &writer) const =0

  Write itself into writer.

4.185.1 Member Function Documentation

4.185.1.1 WriteTo()

```
virtual void WriteTo (
            Writer & writer) const  [pure virtual]
```

Parameters

writer	Writer to use to serialize this Serializable

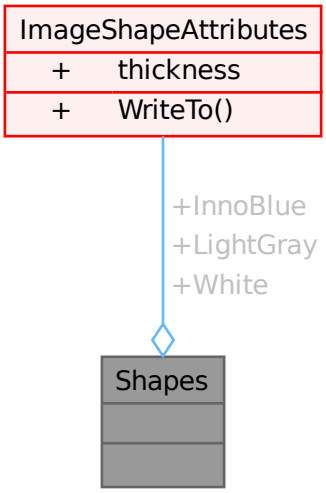
Implemented in AlgorithmicSegmentation::Config, ANSITemplate, Applicant, Circle, CriticalPoint, EmbeddingTemplate, ExtractedFace, ExtractedIris, ExtractedPrint, ExtractedPrintEmbedding, Face::Keypoint, Face::Keypoints, Face, FaceAttributes, FaceCapture, FaceCropConfiguration, FaceDetector::Config, FaceExtractorConfig, FaceTemplate, ICSTemplate, ICSVerifyMatcher::Config, ILOSIDTemplate, ImageShapeAttributes, IndexedModalitiesGalleryConfig, Iris, IrisDetector::Config, IrisExtractorConfig, IrisSegment, IrisTemplate, ISOCCTemplate, ISONCTemplate, ISOTemplate, Line, Minutia, ModalitiesGalleryConfig, ModalitiesICSTemplate, ModalitiesTemplate, ModalitiesVerifyMatcher::Config, ModalityScore, NeuralSegmentation::Config, OFIQFace, Point, Polygon, Print, PrintAttributes, PrintCapture, PrintDetectorConfig, PrintSegment, PrintTemplate, RawImage, Rectangle, RGB, and RGBA.

4.186 Shapes Class Reference

It holds shape attributes literals.

```
#include <ImageShape.h>
```

Collaboration diagram for Shapes:

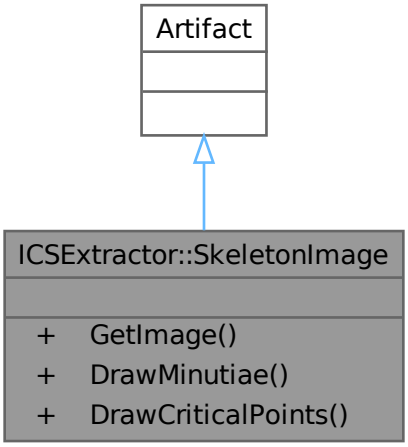


4.187 ICSExtractor::SkeletonImage Class Reference

It enables creation of skeleton image during extraction.

```
#include <ICSExtractor.h>
```

Collaboration diagram for ICSExtractor::SkeletonImage:



Public Member Functions

- std::shared_ptr< Image > GetImage () const

  Get created Image To create Image, extraction must be finished.
- std::shared_ptr< Image > DrawMinutiae (const std::vector< std::shared_ptr< Minutia > > &minutiae, const ImageShapeAttributes &attr) const

  Draws minutiae on skeleton image.
- std::shared_ptr< Image > DrawCriticalPoints (const std::vector< std::shared_ptr< CriticalPoint > > &points, const ImageShapeAttributes &attr) const

  Draws critical points on skeleton image.

4.187.1 Detailed Description

Note

The adjudicator feature MUST be allowed in license to get skeleton image.

Figure 131 Finger for skeleton

Figure 132 Skeleton

4.187.2 Member Function Documentation

4.187.2.1 DrawCriticalPoints()

```
std::shared_ptr< Image > DrawCriticalPoints (
            const std::vector< std::shared_ptr< CriticalPoint > > & points,
            const ImageShapeAttributes & attr) const  [inline], [nodiscard]
```

Parameters

points	critical points to be drawn
attr	ImageShapeAttributes of minutiae

Returns

Image with drawn critical points

4.187.2.2 DrawMinutiae()

```
std::shared_ptr< Image > DrawMinutiae (
            const std::vector< std::shared_ptr< Minutia > > & minutiae,
            const ImageShapeAttributes & attr) const  [inline], [nodiscard]
```

Parameters

minutiae	Minutiae to be drawn
attr	ImageShapeAttributes of minutiae

Returns

Image with drawn minutiae

4.187.2.3 GetImage()

```
std::shared_ptr< Image > GetImage () const [inline], [nodiscard]
```

Exceptions

EnrollmentRuntimeException	when extraction is not finished

Returns

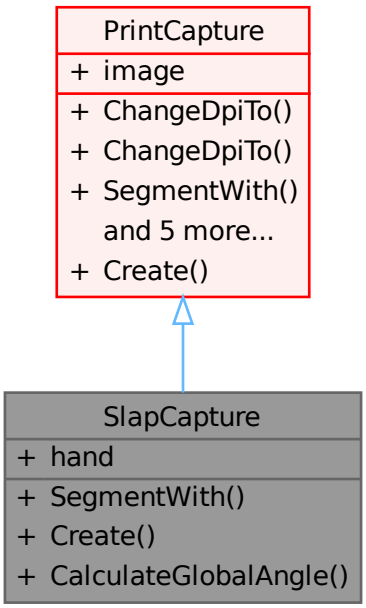
SkeletonImage Image

4.188 SlapCapture Class Reference

It represents PrintCapture with Slap.

```
#include <PrintCapture.h>
```

Collaboration diagram for SlapCapture:



Public Member Functions

- std::vector< std::shared_ptr< PrintSegment > > SegmentWith (PrintCaptureSegmentation &detector) final

  It segments slap capture.

Public Member Functions inherited from PrintCapture

- std::shared_ptr< PrintCapture > ChangeDpiTo (ImageDpi targetDpi) const

  Create new PrintCapture with given DPI.
- std::shared_ptr< PrintCapture > ChangeDpiTo (const Image::ImageResampler &resampler) const

  Create new PrintCapture using provided DPI-change configuration.
- std::shared_ptr< Image > CropPrint (const Rectangle &rectangle, const PrintCropConfiguration &cfg) const final

  It crops image with given rectangle and background color.
- std::shared_ptr< Image > CropPrint (const Rectangle &rectangle) const

  It crops image with given rectangle by default PrintCropConfiguration.
- std::shared_ptr< Image > Draw (const std::vector< std::shared_ptr< PrintSegment > > &segments, const ImageShapeAttributes &attr) const

  It draws rounding boxes into PrintCapture.
- void WriteTo (Writer &writer) const final

  Function serializes print via provided Writer.
- uint8_t CalculateIntensity ()

  Function calculates the image intensity.

Static Public Member Functions

- static std::shared_ptr< SlapCapture > Create (std::shared_ptr< Image > img, const Hand↩ Position &hand, const PrintAttributes &pa=PrintAttributes(), std::shared_ptr< PrintExecutor > executor=GlobalPrintExecutor::Get())

  Constructor of SlapCapture.
- static int CalculateGlobalAngle (const std::vector< std::shared_ptr< inno::PrintSegment > > &segments)

  Calculates the global angle of segmented prints.

Static Public Member Functions inherited from PrintCapture

- static std::shared_ptr< PrintCapture > Create (std::shared_ptr< Image > img, const PrintAttributes &pa=PrintAttributes(), std::shared_ptr< PrintExecutor > executor=GlobalPrintExecutor::Get())

  Constructor of PrintCapture.

Public Attributes

- const HandPosition **hand**

  Provides given hand position.

Public Attributes inherited from PrintCapture

- const std::shared_ptr< Image > **image**

  Provides PrintCapture Image.
- const PrintAttributes **printAttributes**

  Provides PrintCapture Image.

4.188.1 Detailed Description

See Description with examples.

4.188.2 Member Function Documentation

4.188.2.1 CalculateGlobalAngle()

```
int CalculateGlobalAngle (
            const std::vector< std::shared_ptr< inno::PrintSegment > > & segments)  [inline],
[static]
```

Parameters

segments	segmented prints

Exceptions

EnrollmentInvalidArgumentException	if the segments are empty

Returns

global angle calculated from all segmented prints

4.188.2.2 Create()

```
std::shared_ptr< SlapCapture > Create (
            std::shared_ptr< Image > img,
            const HandPosition & hand,
            const PrintAttributes & pa = PrintAttributes(),
            std::shared_ptr< PrintExecutor > executor = GlobalPrintExecutor::Get())
[inline], [static]
```

Parameters

img	Slap image
hand	Slap position
pa	optional PrintAttributes
executor	PrintExecutor to use for PrintCapture functionality execution

Exceptions

NullArgumentException	when image is null

Returns

created SlapCapture object

4.188.2.3 SegmentWith()

```
std::vector< std::shared_ptr< PrintSegment > > SegmentWith (
            PrintCaptureSegmentation & detector) [inline], [final], [virtual]
```

It uses PrintDetector::Segment function to find print segments. It guess print segment position as described in PrintDetector::Segment function.

Parameters

detector	PrintDetector to detect PrintSegments in SlapCapture

Returns

detected print segments

Exceptions

EnrollmentException	from PrintDetector

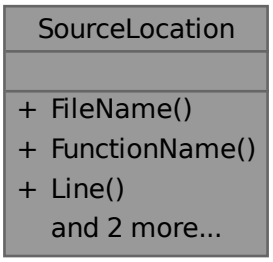
Reimplemented from PrintCapture.

4.189 SourceLocation Class Reference

Class to capture the source location of an error or exception.

```
#include <SourceLocation.h>
```

Collaboration diagram for SourceLocation:



Public Member Functions

- std::string FileName () const

  Get the name of the file where the exception occurred.
- std::string FunctionName () const

  Get the name of the function where the exception occurred.
- constexpr int Line () const

  Get the line number where the exception occurred.
- SourceLocation (const char *file=__builtin_FILE(), const char *function=__builtin_FUNCTION(), int line=__builtin_LINE())

  Constructor to initialize a source location.
- std::string ToString () const

  Creates string representation of SourceLocation.

4.189.1 Detailed Description

This struct serves as a pre-C++20 replacement for `std::source_location`, which is used to capture and store information about the source location where an exception was thrown.

4.189.2 Constructor & Destructor Documentation

4.189.2.1 SourceLocation()

```
SourceLocation (
            const char * file = __builtin_FILE(),
            const char * function = __builtin_FUNCTION(),
            int line = __builtin_LINE())  [inline], [explicit]
```

This constructor initializes a `SourceLocation` object with the file name, function name, and line number. In C++ the values are typically captured using compiler-provided built-in macros like `__builtin_FILE()`, `__builtin_FUNCTION()`, and `__builtin_LINE()`.

Parameters

file	The name of the source file
function	The name of the function
line	The line number within the file

4.189.3 Member Function Documentation

4.189.3.1 FileName()

```
std::string FileName () const  [inline], [nodiscard]
```

This function returns the file name (path) where the exception was thrown.

Returns

file name.

4.189.3.2 FunctionName()

```
std::string FunctionName () const [inline], [nodiscard]
```

This function returns the name of the function in which the exception was thrown.

Returns

function name.

4.189.3.3 Line()

```
int Line () const [inline], [nodiscard], [constexpr]
```

This function returns the line number from where the exception was thrown.

Returns

line number of the exception.

4.189.3.4 ToString()

```
std::string ToString () const [inline], [nodiscard]
```

Format is `file:line` 'function'

Returns

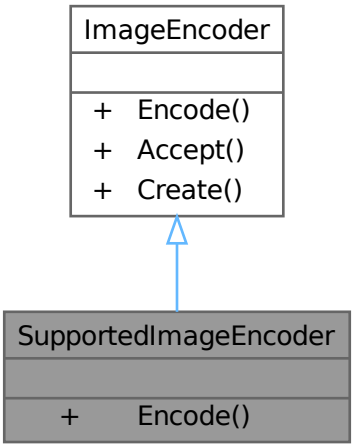
string representation of SourceLocation

4.190 SupportedImageEncoder Class Reference

Encodes raw images into specified supported format, It is used as based class for all supported formats.

```
#include <ImageEncoder.h>
```

Collaboration diagram for SupportedImageEncoder:



Public Member Functions

- std::vector< uint8_t > Encode (const RawImage &raw) const final

  Saves given RawImage as formatted image.

Public Member Functions inherited from ImageEncoder

- virtual void Accept (ImageEncoderVisitor &visitor) const =0

  Allow visitor to visit concrete type of encoder.

Additional Inherited Members

Static Public Member Functions inherited from ImageEncoder

- static std::shared_ptr< ImageEncoder > Create (ImageType type)

  Create concrete default encoder for given type.

4.190.1 Member Function Documentation

4.190.1.1 Encode()

```
std::vector< uint8_t > Encode (
            const RawImage & raw) const  [inline], [final], [virtual]
```

Parameters

raw	RawImage

Returns

encoded image

Exceptions

ImageException	when it is not possible to encode image
InvalidImageArgumentException	when image is empty

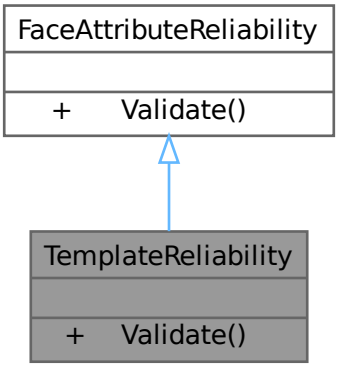
Implements ImageEncoder.

4.191 TemplateReliability Class Reference

Check if template extraction will be reliable.

```
#include <FaceValidation.h>
```

Collaboration diagram for TemplateReliability:



Public Member Functions

- std::vector< FailedFaceAttribute > Validate (const std::shared_ptr< Face > &probe) final
  Check if template extraction will be reliable.

4.191.1 Member Function Documentation

4.191.1.1 Validate()

```
std::vector< FailedFaceAttribute > Validate (
            const std::shared_ptr< Face > & probe)  [inline], [final], [virtual]
```

Parameters

probe	Face to be checked

Returns

list of attributes that will cause not reliable template extraction

Exceptions

NullArgumentException	when probe is null

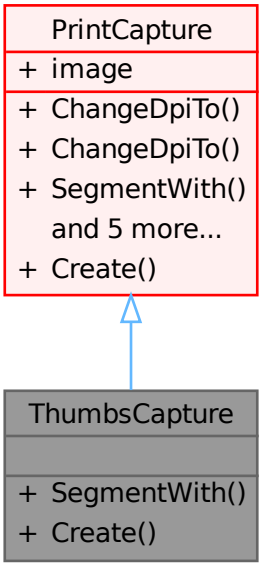
Implements FaceAttributeReliability.

4.192 ThumbsCapture Class Reference

It represents PrintCapture with ThumbsCapture.

```
#include <PrintCapture.h>
```

Collaboration diagram for ThumbsCapture:



Public Member Functions

- std::vector< std::shared_ptr< PrintSegment > > SegmentWith (PrintCaptureSegmentation &detector) final

  It segments thumbs capture.

Public Member Functions inherited from PrintCapture

- std::shared_ptr< PrintCapture > ChangeDpiTo (ImageDpi targetDpi) const

  Create new PrintCapture with given DPI.
- std::shared_ptr< PrintCapture > ChangeDpiTo (const Image::ImageResampler &resampler) const

  Create new PrintCapture using provided DPI-change configuration.
- std::shared_ptr< Image > CropPrint (const Rectangle &rectangle, const PrintCropConfiguration &cfg) const final

  It crops image with given rectangle and background color.
- std::shared_ptr< Image > CropPrint (const Rectangle &rectangle) const

  It crops image with given rectangle by default PrintCropConfiguration.
- std::shared_ptr< Image > Draw (const std::vector< std::shared_ptr< PrintSegment > > &segments, const ImageShapeAttributes &attr) const

  It draws rounding boxes into PrintCapture.
- void WriteTo (Writer &writer) const final

  Function serializes print via provided Writer.
- uint8_t CalculateIntensity ()

  Function calculates the image intensity.

Static Public Member Functions

- static std::shared_ptr< ThumbsCapture > Create (std::shared_ptr< Image > img, const PrintAttributes &pa=PrintAttributes(), std::shared_ptr< PrintExecutor > executor=GlobalPrintExecutor::Get())

  Constructor sets image and attributes of ThumbsCapture.

Static Public Member Functions inherited from PrintCapture

- static std::shared_ptr< PrintCapture > Create (std::shared_ptr< Image > img, const PrintAttributes &pa=PrintAttributes(), std::shared_ptr< PrintExecutor > executor=GlobalPrintExecutor::Get())

  Constructor of PrintCapture.

Additional Inherited Members

Public Attributes inherited from PrintCapture

- const std::shared_ptr< Image > **image**

  Provides PrintCapture Image.
- const PrintAttributes **printAttributes**

  Provides PrintCapture Image.

4.192.1 Detailed Description

See Description with examples.

4.192.2 Member Function Documentation

4.192.2.1 Create()

```
std::shared_ptr< ThumbsCapture > Create (
            std::shared_ptr< Image > img,
            const PrintAttributes & pa = PrintAttributes(),
            std::shared_ptr< PrintExecutor > executor = GlobalPrintExecutor::Get())
[inline], [static]
```

Parameters

img	Thumbs image
pa	optional PrintAttributes
executor	PrintExecutor to use for PrintCapture functionality execution

Exceptions

NullArgumentException	when image is null

Returns

created ThumbsCapture object

4.192.2.2 SegmentWith()

```
std::vector< std::shared_ptr< PrintSegment > > SegmentWith (
            PrintCaptureSegmentation & detector)  [inline], [final], [virtual]
```

It uses PrintDetector::Segment function to find print segments. It guess thumbs position. When 2 segments are detected then first is left and second is right thumb. Otherwise positions are unknown.

Parameters

detector	PrintDetector to detect PrintSegments in ThumbsCapture

Returns

detected print segments

Exceptions

EnrollmentException	from PrintDetector

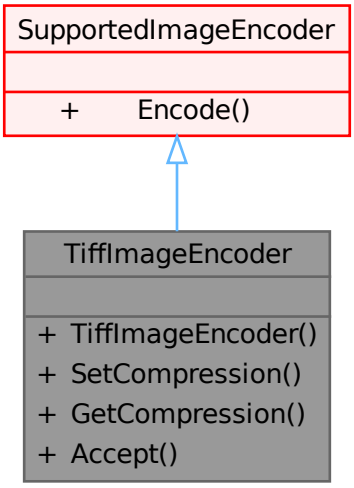
Reimplemented from PrintCapture.

4.193 TiffImageEncoder Class Reference

Converts image type ImageType::TIFF.

```
#include <ImageEncoder.h>
```

Collaboration diagram for TiffImageEncoder:



Public Types

- enum Compression : uint16_t {
  NONE = 1 , CCITTRLE = 2 , CCITTFAX3 = 3 , CCITT_T4 = CCITTFAX3 ,
  CCITTFAX4 = 4 , CCITT_T6 = CCITTFAX4 , LZW = 5 , JPEG = 7 ,
  NEXT = 32766 , CCITTRLEW = 32771 , PACKBITS = 32773 , THUNDERSCAN = 32809 ,
  IT8CTPAD = 32895 , IT8LW = 32896 , IT8MP = 32897 , IT8BL = 32898 ,
  PIXARFILM = 32908 , PIXARLOG = 32909 , DEFLATE = 32946 , ADOBE_DEFLATE = 8 ,
  DCS = 32947 , SGILOG = 34676 , SGILOG24 = 34677 , JP2000 = 34712 }

  Compression supported for tiff images.

Public Member Functions

- TiffImageEncoder ()

  Construct a new Tiff Image Encoder object.
- void SetCompression (Compression compression)

  Set compression level.
- Compression GetCompression () const

  Provides compression.
- void Accept (ImageEncoderVisitor &visitor) const final

  Allow visitor to visit concrete type of encoder.

Public Member Functions inherited from SupportedImageEncoder

- std::vector< uint8_t > Encode (const RawImage &raw) const final

  Saves given RawImage as formatted image.

Additional Inherited Members

Static Public Member Functions inherited from ImageEncoder

- static std::shared_ptr< ImageEncoder > Create (ImageType type)

  Create concrete default encoder for given type.

4.193.1 Member Enumeration Documentation

4.193.1.1 Compression

```
enum Compression : uint16_t
```

See https://www.awaresystems.be/imaging/tiff/tifftags/compression.html.

Enumerator

NONE	No compression.
CCITTRLE	CCITT RLE.
CCITTFAX3	CCITT FAX3.
CCITT_T4	CCITT T4.
CCITTFAX4	CCITT FAX4.
CCITT_T6	CCITT T6.
LZW	LZW.
JPEG	JPEG.
NEXT	NEXT.
CCITTRLEW	CCITT RLE W.
PACKBITS	PACKBITS.
THUNDERSCAN	THUNDERSCAN.
IT8CTPAD	IT8 CTPAD.
IT8LW	IT8 LW.
IT8MP	IT8 MP.
IT8BL	IT8 BL.
PIXARFILM	PIXAR FILM.
PIXARLOG	PIXAR LOG.
DEFLATE	DEFLATE.
ADOBE_DEFLATE	ADOBE DEFLATE.
DCS	DCS.
SGILOG	SGI LOG.

SGILOG24	SGI LOG24.
JP2000	JP2000.

4.193.2 Constructor & Destructor Documentation

4.193.2.1 TiffImageEncoder()

```
TiffImageEncoder () [inline]
```

Default compression is Compression::NONE.

4.193.3 Member Function Documentation

4.193.3.1 Accept()

```
void Accept (
            ImageEncoderVisitor & visitor) const [inline], [final], [virtual]
```

Parameters

visitor	ImageEncoderVisitor to call

Implements ImageEncoder.

4.193.3.2 GetCompression()

```
Compression GetCompression () const [inline]
```

Returns

compression

4.193.3.3 SetCompression()

```
void SetCompression (
            Compression compression) [inline]
```

Parameters

compression	Compression to use in encoding

Exceptions

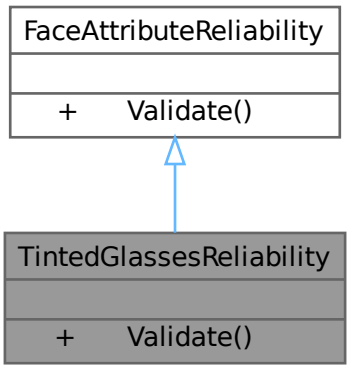
EnrollmentInvalidArgumentException	when compression is unknown

4.194 TintedGlassesReliability Class Reference

Check if calculation of Tinted Glasses attribute will be reliable.

```
#include <FaceValidation.h>
```

Collaboration diagram for TintedGlassesReliability:



Public Member Functions

- std::vector< FailedFaceAttribute > Validate (const std::shared_ptr< Face > &probe) final

  Check if calculation of Tinted Glasses attribute will be reliable.

4.194.1 Member Function Documentation

4.194.1.1 Validate()

```
std::vector< FailedFaceAttribute > Validate (
            const std::shared_ptr< Face > & probe)  [inline], [final], [virtual]
```

Parameters

probe	Face to be checked

Returns

list of attributes that will cause not reliable Tinted Glasses attribute

Exceptions

NullArgumentException	when probe is null

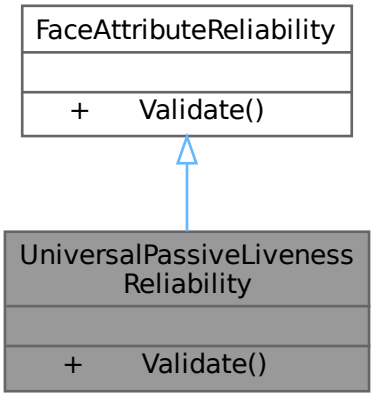
Implements FaceAttributeReliability.

4.195 UniversalPassiveLivenessReliability Class Reference

Check if calculation of Universal passive liveness will be reliable.

```
#include <FaceValidation.h>
```

Collaboration diagram for UniversalPassiveLivenessReliability:



Public Member Functions

- std::vector< FailedFaceAttribute > Validate (const std::shared_ptr< Face > &probe) final

  Check if calculation of Universal passive liveness will be reliable.

4.195.1 Member Function Documentation

4.195.1.1 Validate()

```
std::vector< FailedFaceAttribute > Validate (
            const std::shared_ptr< Face > & probe)  [inline], [final], [virtual]
```

Parameters

probe	Face to be checked

Returns

list of attributes that will cause not reliable Universal passive liveness

Exceptions

NullArgumentException	when probe is null

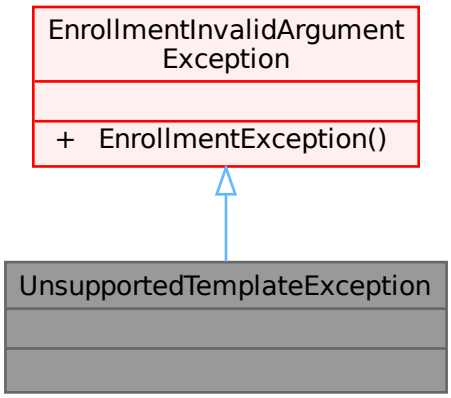
Implements FaceAttributeReliability.

4.196 UnsupportedTemplateException Class Reference

Template version is not supported in this variant of Enrollment SDK.

```
#include <EnrollmentException.h>
```

Inheritance diagram for UnsupportedTemplateException:



Additional Inherited Members

Public Member Functions inherited from EnrollmentInvalidArgumentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.

Public Member Functions inherited from EnrollmentException

- EnrollmentException (std::string reason, SourceLocation location=SourceLocation())

  Construct EnrollmentException object.
- const char * what () const noexcept override

  Provides error message.
- SourceLocation Location () const noexcept

  Provides location of error.
- std::string ToString () const

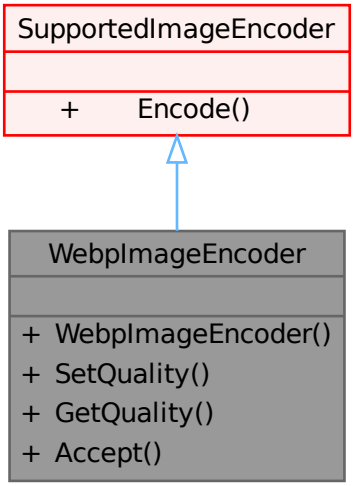
  Provides message with full info about exception.

4.197 WebpImageEncoder Class Reference

Converts image type ImageType::WEBP.

```
#include <ImageEncoder.h>
```

Collaboration diagram for WebpImageEncoder:



Public Member Functions

- **WebpImageEncoder** ()

  Construct a new webp Image Encoder object The default quality is set to 95.
- void SetQuality (int quality)

  Set quality.
- int GetQuality () const

  Provides quality.
- void Accept (ImageEncoderVisitor &visitor) const final

  Allow visitor to visit concrete type of encoder.

Public Member Functions inherited from SupportedImageEncoder

- std::vector< uint8_t > Encode (const RawImage &raw) const final

  Saves given RawImage as formatted image.

Additional Inherited Members

Static Public Member Functions inherited from ImageEncoder

- static std::shared_ptr< ImageEncoder > Create (ImageType type)

  Create concrete default encoder for given type.

4.197.1 Member Function Documentation

4.197.1.1 Accept()

```
void Accept (
            ImageEncoderVisitor & visitor) const [inline], [final], [virtual]
```

Parameters

visitor	ImageEncoderVisitor to call

Implements ImageEncoder.

4.197.1.2 GetQuality()

```
int GetQuality () const [inline]
```

Returns

quality

4.197.1.3 SetQuality()

```
void SetQuality (
            int quality) [inline]
```

Parameters

quality	It it can be a quality from 0 to 100 (the higher is the better). 100 is for lossless compression.

Exceptions

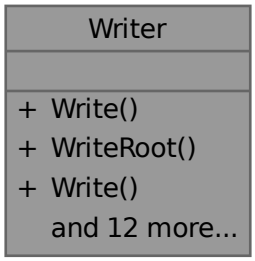
EnrollmentInvalidArgumentException	when quality is not in range

4.198 Writer Class Reference

Provides interface for serialization of all classes.

```
#include <Writer.h>
```

Collaboration diagram for Writer:



Public Member Functions

- virtual void Write (const char ∗name, const Serializable &s)=0

  Serialize serializable.
- virtual void WriteRoot (Serializable &s)=0

  Serialize serializable as root object.
- virtual void Write (const char ∗name, const char ∗value)=0

  Serialize string.
- virtual void Write (const char ∗name, int value)=0

  Serialize int.
- virtual void Write (const char ∗name, unsigned int value)=0

  Serialize unsigned int.
- virtual void Write (const char ∗name, double value)=0

  Serialize double.
- virtual void Write (const char ∗name, int64_t value)=0

  Serialize int64_t integer.
- virtual void Write (const char ∗name, uint64_t value)=0

  Serialize uint64_t integer.
- virtual void Write (const char ∗name, const std::shared_ptr< Image > &value)=0

  Serialize Image.
- virtual void WriteNull (const char ∗name)=0

  Serialize null object.
- void WriteArray (const char ∗name, const std::vector< uint8_t > &value)

  Serialize bytes from vector.

- void SetBytesStrategy (BytesWriterStrategy &s)

  Set bytes serialization strategy .
- template<typename T>
  void WriteArray (const char *name, const T &array)

  Serialize array of elements that are suitable for `range-based for loop`.
- void Write (const char *name, const std::shared_ptr< const Serializable > &s)

  Serialize const serializable given as shared_ptr.
- template<typename T>
  void Write (const char *name, const std::optional< T > &s)

  Serialize serializable given as shared_ptr.

4.198.1 Detailed Description

Interfaces Writer,BytesWriterStrategy are defined for client extension. When client needs own serialization then client implements Writer interface and provide it as Writer for library classes. When client needs to modify only bytes serialization, client implements BytesWriterStrategy interface and sets it into Writer. The library itself provides `JSON` implementation in class JsonWriter. Each class in library that can be serialized must inherit Serializable interface.

Use built in JSON

c++

```
static void UseJson()
{
    auto fp = CreatePrint();
    JsonWriter writer;
    writer.WriteRoot(*fp);
    // print serialized output
    std::cout « "   " « writer.Output() « std::endl;
}
```

java

```
public void useJson() {
    Print fp = createPrint();
    JsonWriter writer = new JsonWriter();
    writer.WriteRoot(fp);
    // print serialized output
    System.out.println("   " + writer.Output());
}
```

csharp

```
public void useJson()
{
    Print fp = createPrint();
    JsonWriter writer = new JsonWriter();
    writer.WriteRoot(fp);
    // print serialized output
    System.Console.WriteLine("   " + writer.Output());
}
```

Use built in JSON with Base64 encoded bytes

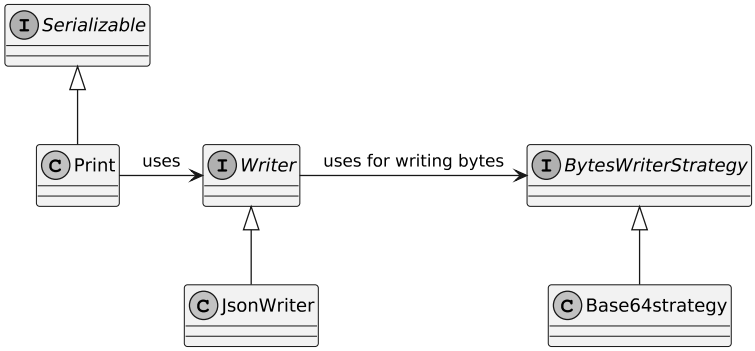


Figure 133 Use built in JSON with Base64 encoded bytes

c++

```
void UseJsonWithBase64()
{
    auto fp = CreatePrint();
    JsonWriter writer;
    Base64Strategy base64;
    writer.SetBytesStrategy(base64);
    writer.WriteRoot(*fp);
    // print serialized output
    std::cout « "    " « writer.Output() « std::endl;
}
```

java

```
public void useJsonWithBase64() {
    Print fp = createPrint();
    JsonWriter writer = new JsonWriter();
    BytesWriterStrategy base64 = new Base64Strategy();
    writer.SetBytesStrategy(base64);
    writer.WriteRoot(fp);
    // print serialized output
    System.out.println("    " + writer.Output());
}
```

csharp

```
public void useJsonWithBase64()
{
    Print fp = createPrint();
    JsonWriter writer = new JsonWriter();
    BytesWriterStrategy base64 = new Base64Strategy();
    writer.SetBytesStrategy(base64);
    writer.WriteRoot(fp);
    // print serialized output
    System.Console.WriteLine("    " + writer.Output());
}
```

Use client specific serialization

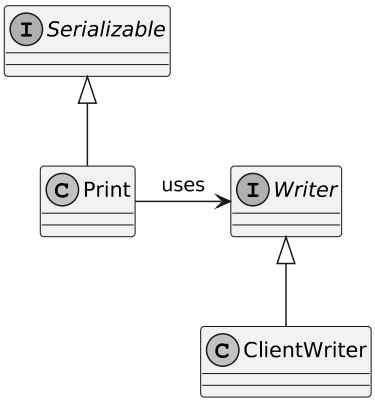


Figure 134 Use client specific serialization

C++

```
class ClientWriter : public Writer
{
public:
    std::string text;

    void Write(const char* name, const Serializable& s) final
    {
        text += name;
        text += "(";
        s.WriteTo(*this);
        text += ")";
    }
    void WriteRoot(Serializable& s) final
    {
        s.WriteTo(*this);
    }
    void WriteNull(const char* name) final
    {
        text += name;
        text += "(";
        text += "null";
        text += ")";
    }
    void Write(const char* name, const char* value) final
    {
        text += name;
        text += "(";
        text += value;
        text += ")";
    }

    void Write(const char* name, double value) final
    {
        Write(name, std::to_string(value).c_str());
    }
    void Write(const char* name, int64_t value) final
    {
        Write(name, std::to_string(value).c_str());
    }
    void Write(const char* name, uint64_t value) final
    {
        Write(name, std::to_string(value).c_str());
    }
    void Write(const char* name, int value) final
    {
        Write(name, std::to_string(value).c_str());
    }
    void Write(const char* name, unsigned int value) final
    {
        Write(name, std::to_string(value).c_str());
    }
    void Write(const char* name, const std::shared_ptr<Image>& value) final
    {
        auto w = std::to_string(value->Width());
        auto h = std::to_string(value->Height());
```

```
        const std::string desc = "w:" + w + ",h:" + h;
        Write(name, desc.c_str());
    }

private:
    void StartArray(const char* /*name*/) final
    {
        // for simplicity we ignore arrays in example
    }
    void EndArray() final
    {
        // for simplicity we ignore arrays in example
    }
};

static void UseClientSpecificWriter()
{
    auto fp = CreatePrint();
    ClientWriter writer;
    writer.WriteRoot(*fp);
    // print serialized output
    std::cout « "   " « writer.text « std::endl;
}
```

java

```
class ClientWriter extends com.innovatrics.enrollment.Writer {
    public String text;

    public ClientWriter() {
        text = "";
    }

    public void Write(String name, com.innovatrics.enrollment.Serializable s) {
        text += name + "(";
        s.WriteTo(this);
        text += ")";
    }

    public void WriteRoot(com.innovatrics.enrollment.Serializable s) {
        s.WriteTo(this);
    }

    public void Write(String name, String value) {
        text += name + "(" + value + ")";
    }

    public void Write(String name, int value) {
        text += name + "(" + value + ")";
    }

    public void Write(String name, long value) {
        text += name + "(" + value + ")";
    }

    public void Write(String name, Image value) {
        Write(name, "w:" + value.Width() + ",h:" + value.Height());
    }

    protected void StartArray(String name) {
        // for simplicity we ignore arrays in example
    }

    protected void EndArray() {
        // for simplicity we ignore arrays in example
    }
}

public void useClientSpecificWriter() {
    Print fp = createPrint();
    ClientWriter writer = new ClientWriter();
    writer.WriteRoot(fp);
    // print serialized output
    System.out.println("   " + writer.text);
}
```

csharp

```
class ClientWriter : Writer
{
    public string text;

    public ClientWriter()
    {
    }

    public override void Write(string name, Serializable s)
```

```
    {
        text += name + "(";
        s.WriteTo(this);
        text += ")";
    }
    public override void WriteRoot(Serializable s)
    {
        s.WriteTo(this);
    }

    public override void Write(string name, string value)
    {
        text += name + "(" + value + ")";
    }

    public override void Write(string name, int value)
    {
        text += name + "(" + value + ")";
    }

    public override void Write(string name, uint value)
    {
        text += name + "(" + value + ")";
    }

    public override void Write(string name, long value)
    {
        text += name + "(" + value + ")";
    }

    public override void Write(string name, ulong value)
    {
        text += name + "(" + value + ")";
    }

    public override void Write(string name, Image value)
    {
        Write(name, "w:" + value.Width() + ",h:" + value.Height());
    }

    protected override void StartArray(string name)
    {
        // for simplicity we ignore arrays in example
    }
    protected override void EndArray()
    {
        // for simplicity we ignore arrays in example
    }
}

public void useClientSpecificWriter()
{
    Print fp = createPrint();
    ClientWriter writer = new ClientWriter();
    writer.WriteRoot(fp);
    // print serialized output
    System.Console.WriteLine("   " + writer.text);
}
```

Use client specific bytes strategy

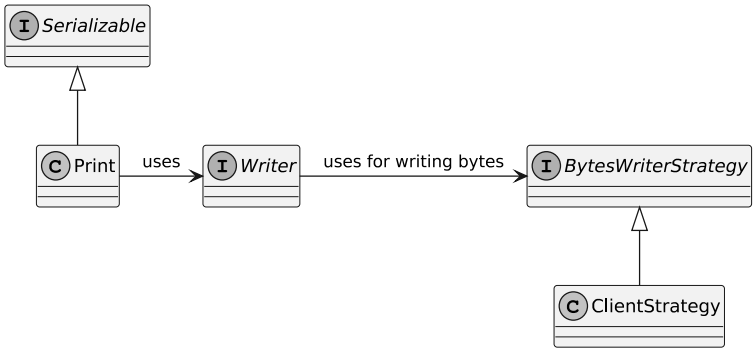


Figure 135 Use client specific bytes strategy

C++

```
class ClientStrategy : public BytesWriterStrategy
{
public:
    void WriteBytes(Writer& writer,
                    const char* name,
                    const std::vector<uint8_t>& value) final
    {
        auto size = std::to_string(value.size());
        auto v0 = std::to_string(value[0]);
        auto v1 = std::to_string(value[1]);
        const std::string text = "Reduced 2/" + size + " " + v0 + "," + v1 + ", ...";
        writer.Write(name, text.c_str());
    }
};

void UseClientSpecificByteStrategy()
{
    auto fp = CreatePrint();
    JsonWriter writer;
    ClientStrategy bytesStrategy;
    writer.SetBytesStrategy(bytesStrategy);
    writer.WriteRoot(*fp);
    // print serialized output
    std::cout « "   " « writer.Output() « std::endl;
}
```

java

```
class ClientStrategy extends BytesWriterStrategy {
    public void WriteBytes(
            com.innovatrics.enrollment.Writer writer, String name, Bytes value) {
        int s = value.size();
        int v0 = value.get(0);
        int v1 = value.get(1);
        writer.Write(name, "Reduced 2/" + s + " " + v0 + "," + v1 + ", ...");
    }
}

public void useClientSpecificByteStrategy() {
    Print fp = createPrint();
    JsonWriter writer = new JsonWriter();
    ClientStrategy bytesStrategy = new ClientStrategy();
    writer.SetBytesStrategy(bytesStrategy);
    writer.WriteRoot(fp);
    // print serialized output
    System.out.println("   " + writer.Output());
}
```

csharp

```
class ClientStrategy : BytesWriterStrategy
{
    public override void WriteBytes(Writer writer, string name, Bytes value)
    {
        int s = value.Count;
        int v0 = value[0];
        int v1 = value[1];
        writer.Write(name, "Reduced 2/" + s + " " + v0 + "," + v1 + ", ...");
    }
}

public void useClientSpecificByteStrategy()
{
    Print fp = createPrint();
    JsonWriter writer = new JsonWriter();
    ClientStrategy bytesStrategy = new ClientStrategy();
    writer.SetBytesStrategy(bytesStrategy);
    writer.WriteRoot(fp);
    // print serialized output
    System.Console.WriteLine("   " + writer.Output());
}
```

4.198.2 Member Function Documentation

4.198.2.1 SetBytesStrategy()

```
void SetBytesStrategy (
            BytesWriterStrategy & s)  [inline]
```

Parameters

s	strategy to serialize bytes

4.198.2.2 Write() [1/10]

```
virtual void Write (
            const char * name,
            const char * value)  [pure virtual]
```

Parameters

name	name of value (key)
value	string value

Implemented in JsonWriter.

4.198.2.3 Write() [2/10]

```
virtual void Write (
            const char * name,
            const Serializable & s)  [pure virtual]
```

Parameters

name	name of serializable
s	class that implements Serializable interface

Implemented in JsonWriter.

4.198.2.4 Write() [3/10]

```
template<typename T>
void Write (
            const char * name,
            const std::optional< T > & s)  [inline]
```

Parameters

name	name of serializable
s	class that implements Serializable interface

4.198.2.5 Write() [4/10]

```
void Write (
            const char * name,
            const std::shared_ptr< const Serializable > & s)  [inline]
```

Parameters

name	name of serializable
s	class that implements Serializable interface

4.198.2.6 Write() [5/10]

```
virtual void Write (
            const char * name,
            const std::shared_ptr< Image > & value)  [pure virtual]
```

Once the image serialization needs to be modified e.g. all images are serialized in specific format, then strategy can be used similar as BytesWriterStrategy.

Parameters

name	name of value (key)
value	Image

Implemented in JsonWriter.

4.198.2.7 Write() [6/10]

```
virtual void Write (
            const char * name,
            double value)  [pure virtual]
```

Parameters

name	name of value (key)
value	double value

Implemented in JsonWriter.

4.198.2.8 Write() [7/10]

```
virtual void Write (
            const char * name,
            int value)  [pure virtual]
```

Parameters

name	name of value (key)
value	integer value

Implemented in JsonWriter.

4.198.2.9 Write() [8/10]

```
virtual void Write (
            const char * name,
            int64_t value)  [pure virtual]
```

Parameters

name	name of value (key)
value	unsigned integer value

Implemented in JsonWriter.

4.198.2.10 Write() [9/10]

```
virtual void Write (
            const char * name,
            uint64_t value)  [pure virtual]
```

Parameters

name	name of value (key)
value	unsigned integer value

Implemented in JsonWriter.

4.198.2.11 Write() [10/10]

```
virtual void Write (
            const char * name,
            unsigned int value)  [pure virtual]
```

Parameters

name	name of value (key)
value	unsigned integer value

Implemented in JsonWriter.

4.198.2.12 WriteArray() [1/2]

```
void WriteArray (
            const char * name,
            const std::vector< uint8_t > & value)  [inline]
```

It uses BytesWriterStrategy to serialize bytes. The strategy can be set by SetBytesStrategy function.

Parameters

name	name of value (key)
value	vector of bytes

4.198.2.13 WriteArray() [2/2]

```
template<typename T>
void WriteArray (
            const char * name,
            const T & array)
```

Parameters

name	name of value (key)
array	array suitable for `range-based for loop`.

Template Parameters

T	type suitable for `range-based for loop`.

4.198.2.14 WriteNull()

```
virtual void WriteNull (
            const char * name)  [pure virtual]
```

Parameters

name	name of value (key)

Implemented in JsonWriter.

4.198.2.15 WriteRoot()

```
virtual void WriteRoot (
            Serializable & s)  [pure virtual]
```

Parameters

s	class that implements Serializable interface

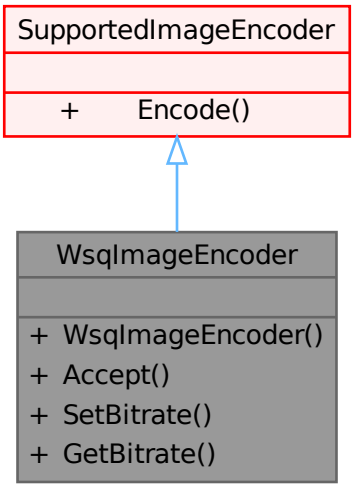
Implemented in JsonWriter.

4.199 WsqImageEncoder Class Reference

Converts image type ImageType::WSQ.

```
#include <ImageEncoder.h>
```

Collaboration diagram for WsqImageEncoder:



Public Member Functions

- **WsqImageEncoder** ()

  Construct a new WSQ Image Encoder object.
- void Accept (ImageEncoderVisitor &visitor) const final

  Allow visitor to visit concrete type of encoder.
- void SetBitrate (float bitrate)

  Set the bitrate for WSQ compression.
- float GetBitrate () const

  Provides WSQ image bitrate.

Public Member Functions inherited from SupportedImageEncoder

- std::vector< uint8_t > Encode (const RawImage &raw) const final

  Saves given RawImage as formatted image.

Additional Inherited Members

Static Public Member Functions inherited from ImageEncoder

- static std::shared_ptr< ImageEncoder > Create (ImageType type)

  Create concrete default encoder for given type.

4.199.1 Member Function Documentation

4.199.1.1 Accept()

```
void Accept (
            ImageEncoderVisitor & visitor) const [inline], [final], [virtual]
```

Parameters

visitor	ImageEncoderVisitor to call

Implements ImageEncoder.

4.199.1.2 GetBitrate()

```
float GetBitrate () const [inline]
```

Returns

average bits per pixel in the WSQ image

4.199.1.3 SetBitrate()

```
void SetBitrate (
            float bitrate) [inline]
```

Defines the bitrate for the WSQ compression algorithm. Note that there is no universal correlation between bitrate and compression ratio as it depends on the image content. The bitrate is defined as the average bits per pixel in the compressed image.

Parameters

bitrate	Valid range is <0.1,5.0>. The default value is 1.25, which corresponds approximately to a 1:10 compression ratio for a standard fingerprint image.

Exceptions

EnrollmentInvalidArgumentException	if the bitrate is not in the valid range <0.1,5.0>.

References

[1] Elham Tabassi, Martin Olsen, Oliver Bausinger, Christoph Busch, Andrew Figlarz, Gregory Fiumara, Olaf Henniger, Johannes Merkle, Timo Ruhland, Christopher Schiel, Michael Schwaiger. Nist fingerprint image quality, July 2021. URL https://nvlpubs.nist.gov/nistpubs/ir/2021/NIST.IR.8382.pdf. 659

[2] Federal Bureau of Investigation, Criminal Justice Information Services Division. Electronic fingerprint transmission specification, January 1999. URL https://fbibiospecs.fbi.gov/file-repository/eftsv7-0.pdf. 668, 670

[3] International Labour Organization. The standard for the biometric template required by the convention, 2006. URL https://www.ilo.org/sites/default/files/wcmsp5/groups/public/@ed_norm/@normes/documents/genericdocument/wcms_144265.pdf. 428

[4] ISO/IEC. Iso/iec 19794-1:2011 - information technology — biometric data interchange formats — part 1: Framework, July 2011. URL https://www.iso.org/standard/50862.html. Accessed: 2026-02-25. 300

[5] ISO/IEC. Iso/iec 19794-4:2011 - information technology — biometric data interchange formats — part 4: Finger image data, December 2011. URL https://www.iso.org/standard/50866.html. Accessed: 2026-02-25. 199, 228, 663, 664, 677

[6] ISO/IEC. Iso/iec 19794-5:2011 - information technology — biometric data interchange formats — part 5: Face image data, December 2011. URL https://www.iso.org/standard/50867.html. Accessed: 2026-02-25. 128, 150, 153, 154, 361, 372, 374, 375, 530

[7] ISO/IEC. Iso/iec 19794-6:2011 - information technology — biometric data interchange formats — part 6: Iris image data, December 2011. URL https://www.iso.org/standard/50868.html. Accessed: 2026-02-25. 180, 184, 190, 191, 192, 498, 506, 507, 508

[8] ISO/IEC. Iso/iec 29794-5:2025 - information technology — biometric sample quality — part 5: Face image data, April 2025. URL https://www.iso.org/standard/81005.html. Accessed: 2026-02-25. 141, 144, 145, 638, 639, 640, 641, 642, 644, 645, 646

[9] Michael D. Garris, R. Michael McCabe, NIST. Fingerprint minutiae from latent and matching tenprint images, June 2000. URL https://www.nist.gov/system/files/documents/srd/nistsd27.pdf. 299, 300, 580, 581

[10] Paul Griffin, Ph.D. Understanding the face image format standards, April 2005. URL https://www.nist.gov/system/files/documents/2021/02/25/ansi-nist_2007_griffin-face-std-m1.pdf. 153

[11] Wikipedia contributors. Rgba color model — Wikipedia, the free encyclopedia, 2021. URL https://en.wikipedia.org/w/index.php?title=RGBA_color_model&oldid=1057086643. [Online; accessed 2-August-2022]. 694

[12] Wikipedia contributors. Jpeg 2000 — Wikipedia, the free encyclopedia, 2022. URL https://en.wikipedia.org/w/index.php?title=JPEG_2000&oldid=1109158702. [Online; accessed 14-September-2022]. 167, 474

[13] Wikipedia contributors. Jpeg — Wikipedia, the free encyclopedia, 2022. URL https://en.wikipedia.org/w/index.php?title=JPEG&oldid=1109544252. [Online; accessed 13-September-2022]. 167, 474

[14] Wikipedia contributors. Portable network graphics — Wikipedia, the free encyclopedia, 2022. URL https://en.wikipedia.org/w/index.php?title=Portable_Network_Graphics&oldid=1109770037. [Online; accessed 14-September-2022]. 166, 474

[15] Wikipedia contributors. Tiff — Wikipedia, the free encyclopedia, 2022. URL https://en.wikipedia.org/w/index.php?title=TIFF&oldid=1105777129. [Online; accessed 14-September-2022]. 167, 474

[16] Wikipedia contributors. Webp — Wikipedia, the free encyclopedia, 2022. URL https://en.wikipedia.org/w/index.php?title=WebP&oldid=1109484490. [Online; accessed 14-September-2022]. 167, 474

[17] Wikipedia contributors. Rgb color model — Wikipedia, the free encyclopedia, 2023. URL https://en.wikipedia.org/w/index.php?title=RGB_color_model&oldid=1182865390. [Online; accessed 10-November-2023]. 704

[18] Wikipedia contributors. Bmp file format — Wikipedia, the free encyclopedia, 2025. URL https://en.wikipedia.org/w/index.php?title=BMP_file_format&oldid=1269954732. [Online; accessed 21-February-2025]. 166, 474

Index

Accept
- BmpImageEncoder, 268
- ImageEncoder, 456
- IRawImageEncoder, 495
- Jp2ImageEncoder, 545
- JpgImageEncoder, 548
- PngImageEncoder, 649
- TiffImageEncoder, 724
- WebpImageEncoder, 729
- WsqImageEncoder, 742

ACCURATE
- Face, 341, 345, 346
- FaceDetector::Config, 283
- FaceExtractorConfig, 383
- IrisDetector::Config, 290
- IrisExtractorConfig, 512

ACCURATE_SERVER
- FaceDetector::Config, 283

Adapt
- ANSITemplateAdapter, 244–246
- ILOSIDTemplateAdapter, 431–433
- ISOCCTemplateAdapter, 526–529
- ISONCTemplateAdapter, 534–536
- ISOTemplateAdapter, 541–543

Add
- ICSExtractor::Artifacts, 260
- ICSVerifyMatcher::Artifacts, 261
- IndexedModalitiesGallery, 480
- ModalitiesGallery, 594

AddFace
- Applicant, 249

AddIris
- Applicant, 249

AddPoint
- Polygon, 652

AddPrint
- Applicant, 249

AddSlap
- Applicant, 249

AddThumbs
- Applicant, 250

AddToBlur
- ImageMasker, 464, 465

AddUniquePrint
- Applicant, 250

AddValidFace
- Applicant, 251

AddValidPrint
- Applicant, 251

ADOBE_DEFLATE
- TiffImageEncoder, 723

AGE
- Face, 343

Age
- Face, 346

AgeEstimationMode
- Face, 341

AgeReliability, 238
- Validate, 238

AlgorithmicSegmentation, 239
- AlgorithmicSegmentation, 240

AlgorithmicSegmentation::Config, 279
- Config, 280
- Default, 280
- GetMaxRotation, 280
- IsNoiseReduction, 281
- SetMaxRotation, 281
- WriteTo, 281

Anonymize
- ICSTemplate, 419

ANSITemplate, 241
- ANSITemplate, 242
- GetVersion, 242
- WriteTo, 242

ANSITemplateAdapter, 243
- Adapt, 244–246
- ANSITemplateAdapter, 244

Applicant, 247
- AddFace, 249
- AddIris, 249
- AddPrint, 249
- AddSlap, 249
- AddThumbs, 250
- AddUniquePrint, 250
- AddValidFace, 251
- AddValidPrint, 251
- GetEnrolledFaces, 252
- GetEnrolledIrises, 252
- GetEnrolledPrints, 252
- SimilarWith, 252
- ToICSTemplate, 253
- ToTemplate, 253
- WriteTo, 253

ApplicantFaceValidation, 254
- Validate, 254

ApplicantMatcher, 255
- Match, 255

ApplicantPrintValidation, 256
- Validate, 256

ApplyBlur
- ImageMasker, 465

AreaImageResampler
- Image::AreaImageResampler, 258

AsBGR
- Image, 438

AsBGRA
- Image, 438

AsRawBGR
- Image, 438

AsRawBGRA

Image, 438
AsRawGrayscale
Image, 439
AsRawImage
Image, 439
AttributeID
Face, 341

BACKGROUND_UNIFORMITY
Face, 343
BackgroundUniformity
Face, 347
BALANCED
Face, 341, 345
FaceDetector::Config, 283
FaceExtractorConfig, 383
Base64Strategy, 262
WriteBytes, 262
BASIC
Face, 153
BGR
RawImage, 694
BGRA
RawImage, 694
BICUBIC
Image::NISTImageResampler, 630
BicubicImageResampler
Image::BicubicImageResampler, 264
BIFURCATION
Minutia, 580
BILINEAR
Image::NISTImageResampler, 630
BinaryFileAccess, 266
ReadAll, 267
WriteAll, 267
Binding, 119
BlurCircle
ImageMasker, 465
BlurPolygon
ImageMasker, 465
BMP
ImageType, 474
BmpImageEncoder, 267
Accept, 268
BoundingRectangle
Circle, 274
BRIGHTNESS
Face, 342
Brightness
Face, 348
Build
InitializerConfigBuilder, 487
BytesWriterStrategy, 269
WriteBytes, 269

C#, 123
CalculateCenter
Rectangle, 698
CalculateEyelid
Iris, 497
CalculateGlobalAngle
SlapCapture, 713
CalculateImageQuality
Print, 658
CalculateIntensity
Print, 658
PrintCapture, 672
CalculateNFIQ1
Print, 659
CalculateNFIQ2
Print, 659
CalculatePlacement
Print, 660
CalculatePresence
Print, 660
CalculateQuality
ICSTemplate, 419
IrisTemplate, 520
CallbackLogger, 270
CallbackLogger, 271
GetCallbacks, 271
LogDebug, 271
LogError, 271
LogInfo, 272
capture
Face, 362
CCITT_T4
TiffImageEncoder, 723
CCITT_T6
TiffImageEncoder, 723
CCITTFAX3
TiffImageEncoder, 723
CCITTFAX4
TiffImageEncoder, 723
CCITTRLE
TiffImageEncoder, 723
CCITTRLEW
TiffImageEncoder, 723
ChangeDpiTo
Image, 439, 440
Print, 660, 661
PrintCapture, 673
Channels
Image, 440
ChinTip
Face::Keypoints, 558
Circle, 273
BoundingRectangle, 274
Circle, 274
CreateInscribedCircle, 274
WriteTo, 275
CircleImageMask, 275
CircleImageMask, 276
MaskWith, 277
CircleIntersection
Image, 441
CircleShape, 277

CircleShape, 278
DrawIn, 278
CMOS_CCD
Iris, 190
Colors, 278
Compose
FormattingComposer, 398
LogMessage, 576
Compression
TiffImageEncoder, 723
CONFIDENCE
Face, 342
confidence
Face, 362
Iris, 500
Config
AlgorithmicSegmentation::Config, 280
FaceDetector::Config, 284
ICSVerifyMatcher::Config, 287
IrisDetector::Config, 290
ModalitiesVerifyMatcher::Config, 294
NeuralSegmentation::Config, 296
Consolidate
ModalitiesGallery, 594
CONTRAST
Face, 342
Contrast
Face, 348
CORE
CriticalPoint, 300
Create
FaceCapture, 370
ImageEncoder, 456
IndexedModalitiesGallery, 480
IrisCapture, 505
ModalitiesGallery, 595, 596
ModalitiesTemplate, 615
OFIQFace, 637
Print, 661
PrintCapture, 674
SlapCapture, 713
ThumbsCapture, 721
CreateInscribedCircle
Circle, 274
CreateModalitiesICSTemplate
ModalitiesICSTemplateDefaultFactory, 609
ModalitiesICSTemplateFactory, 610
CriticalPoint, 298
CORE, 300
CriticalPoint, 300
DELTA, 300
Type, 300
WriteTo, 300
CriticalPointShape, 301
CriticalPointShape, 302
DrawIn, 302
CriticalPointsShape, 303
CriticalPointsShape, 303, 304
DrawIn, 304
Crop
Face, 349
CropAlignedRectangle
Image, 441, 442
CropCircle
Image, 443
CropFace
FaceCapture, 370
CROPPED
Iris, 190
CROPPED_AND_MASKED
Iris, 190
CroppedFace, 304
FaceAreaMask, 305
FaceAreaSegment, 306
GetCropRectangle, 307
GetFaceCapture, 307
IsCropRegionInsideImage, 307
CropPrint
PrintCapture, 674
CropRectangle
Image, 443, 444

Data
RawImage, 695
DCS
TiffImageEncoder, 723
Decode
Image, 445, 446
DECODE_IMAGE
ImageException, 459
DecodeRawImage
Image, 447
DeduceType
Image, 447
Default
AlgorithmicSegmentation::Config, 280
FaceDetector::Config, 284
FaceExtractorConfig, 383
ICSVerifyMatcher::Config, 287
IrisDetector::Config, 291
IrisExtractorConfig, 513
ModalitiesVerifyMatcher::Config, 294
NeuralSegmentation::Config, 297
DefaultModelsDirectory
Manual initialisation, 181
DEFLATE
TiffImageEncoder, 723
DELTA
CriticalPoint, 300
Detect
FaceDetector, 381
IrisDetector, 510
DetectionRectangle
Face, 349
DetectWith
FaceCapture, 371
IrisCapture, 505

Dimension
Rectangle, 697
DoDetect
PrintDetector, 682
Downcast
ModalitiesGalleryIndexedTemplate, 602
Dpi
Image, 448
Draw
FaceCapture, 371, 372
PrintCapture, 675
DrawCircle
ImageShapeDrawer, 472
DrawCriticalPoints
ICSExtractor::SkeletonImage, 710
Print, 662
DrawIn
CircleShape, 278
CriticalPointShape, 302
CriticalPointsShape, 304
ImageShape, 470
LineShape, 573
MinutiaeShape, 582
MinutiaShape, 584
RectangleShape, 701
RectanglesShape, 703
DrawLine
ImageShapeDrawer, 472
DrawMinutiae
ICSExtractor::SkeletonImage, 710
Print, 662
DrawRectangle
ImageShapeDrawer, 472
DrawShape
Image, 448

ELECTRO_LUMINESCENT
Print, 226
EmbeddingExtractor, 307
Extract, 308
EmbeddingMatcher, 308
Match, 309
EmbeddingTemplate, 309
EmbeddingTemplate, 311–313
GetEmbeddingTemplateBiometricPosition, 314
GetVersion, 314
IsEmbeddingTemplateBiometricPositionKnown, 314
SimilarWith, 315
WriteTo, 315
EmbeddingVerifyMatcher, 316
EmbeddingVerifyMatcher, 316
Match, 317
EnableAlignedCropping
PrintCropConfiguration, 679
Encode
ImageEncoder, 456
IRawImageEncoder, 495
SupportedImageEncoder, 717
ENCODE_IMAGE
ImageException, 459
EncodeIn
Image, 448
EnhanceWithEmbedding
ExtractedPrint, 331
Enrollment, 26
EnrollmentException, 318
EnrollmentException, 318
EnrollmentInvalidArgumentException, 320
EnrollmentRuntimeException, 321
Location, 319
ToString, 319
what, 319
EnrollmentInvalidArgumentException, 319
EnrollmentException, 320
EnrollmentRuntimeException, 320
EnrollmentException, 321
Exception, 125
ExpiredLicenseException, 322
Extending SDK, 118
Extract
EmbeddingExtractor, 308
Face, 349, 350
ICSExtractor, 413
IdentificationEmbeddingExtractor, 427
Iris, 497
ExtractedFace, 323
ToTemplates, 324
WriteTo, 324
ExtractedIris, 325
ToTemplates, 327
WriteTo, 327
ExtractedPrint, 328
EnhanceWithEmbedding, 331
ExtractedPrint, 330
GetEmbeddingRepresentation, 331
HasEmbeddingRepresentation, 332
ToANSI, 332
ToICSTemplates, 332
ToISO, 332
ToTemplates, 333
WriteTo, 333
ExtractedPrintEmbedding, 334
ExtractedPrintEmbedding, 335–337
WriteTo, 337
EYE_DISTANCE
Face, 344
EYE_GAZE
Face, 343
EyeDistance
Face, 350
EyeGaze
Face, 350

Face, 126, 338
ACCURATE, 341, 345, 346
AGE, 343
Age, 346

AgeEstimationMode, 341
AttributeID, 341
BACKGROUND_UNIFORMITY, 343
BackgroundUniformity, 347
BALANCED, 341, 345
BASIC, 153
BRIGHTNESS, 342
Brightness, 348
capture, 362
CONFIDENCE, 342
confidence, 362
CONTRAST, 342
Contrast, 348
Crop, 349
DetectionRectangle, 349
Extract, 349, 350
EYE_DISTANCE, 344
EYE_GAZE, 343
EyeDistance, 350
EyeGaze, 350
FACE_RELATIVE_AREA, 344
FACE_RELATIVE_AREA_IN_IMAGE, 345
FACE_SIZE, 344
FaceCaptureDeviceTechnology, 152
FaceCrop, 152
FaceImageType, 153
FaceRelativeArea, 351
FaceRelativeAreaInImage, 351
FaceSimilarityScore, 151
FaceSize, 352
FaceToIsoCaptureDeviceTechnology, 154
FaceToIsoImageType, 154
FAST, 346
FULL, 153
FULL_FRONTAL, 153
FULL_NOT_ALIGNED, 153
GENDER, 344
Gender, 352
GenderEstimationMode, 345
GetKeypoints, 353
GLASS_STATUS, 343
GlassStatus, 353
HEAVY_FRAME, 343
HeavyFrame, 353
IsFaceCropMethodValid, 155
IsoToFaceCaptureDeviceTechnology, 155
IsoToFaceImageType, 155
IsSameAs, 354
LEFT_EYE_STATUS, 343
LEFT_RED_EYE, 344
LeftEyeStatus, 354
LeftRedEye, 355
MOUTH_STATUS, 343
MouthStatus, 355
NOSE_SHADOW, 342
NoseShadow, 355
operator!=, 356
operator==, 356
PassiveLiveness, 356, 357
PassiveLivenessMode, 345
PITCH_ANGLE, 344
PitchAngle, 357
POST_PROCESSED_FRONTAL, 154
RIGHT_EYE_STATUS, 343
RIGHT_RED_EYE, 343
RightEyeStatus, 358
RightRedEye, 358
ROLL_ANGLE, 344
RollAngle, 359
SHADOW, 342
Shadow, 359
SHARPNESS, 342
Sharpness, 359
SPECULARITY, 342
Specularity, 360
STATIC_PHOTOGRAPH_FROM_DIGITAL_CAMERA, 152
STATIC_PHOTOGRAPH_FROM_SCANNER, 152
STATIC_PHOTOGRAPH_FROM_UNKNOWN_SOURCE, 152
TINTED_GLASSES, 345
TintedGlasses, 360
ToIsoImage, 360
TOKEN, 153
TOKEN_FRONTAL, 154
TOKEN_NOT_FRONTAL, 153
UNIQUE_INTENSITY_LEVELS, 342
UniqueIntensityLevels, 361
UNKNOWN, 152
VIDEO_FRAME_FROM_ANALOGUE_CAMERA, 152
VIDEO_FRAME_FROM_DIGITAL_CAMERA, 152
VIDEO_FRAME_FROM_UNKNOWN_SOURCE, 152
WIDTH_HEIGHT_RATIO, 345
WidthHeightRatio, 361
WriteTo, 361
YAW_ANGLE, 344
YawAngle, 362
Face::Keypoint, 554
GetPosition, 555
GetScore, 555
WriteTo, 556
Face::Keypoints, 556
ChinTip, 558
GetImage, 558
LeftEdge, 558
LeftEyebrowInnerEnd, 559
LeftEyebrowOuterEnd, 559
LeftEyeCentre, 559
LeftEyeInnerCorner, 560
LeftEyeOuterCorner, 560
MouthCenter, 560
MouthLeftCorner, 561
MouthLowerEdge, 561
MouthRightCorner, 561

MouthUpperEdge, 562
NoseBottom, 562
NoseLeftBottom, 562
NoseRightBottom, 563
NoseRoot, 563
NoseTip, 563
RightEdge, 564
RightEyebrowInnerEnd, 564
RightEyebrowOuterEnd, 564
RightEyeCentre, 565
RightEyeInnerCorner, 565
RightEyeOuterCorner, 565
WriteTo, 566
FACE_RELATIVE_AREA
Face, 344
FACE_RELATIVE_AREA_IN_IMAGE
Face, 345
FACE_SIZE
Face, 344
FaceAreaMask
CroppedFace, 305
FaceAreaSegment
CroppedFace, 306
FaceAttributeReliability, 363
Validate, 363
FaceAttributes, 364
FaceAttributes, 365
GetCaptureDeviceTechnology, 365
GetCaptureDeviceTypeID, 365
GetCaptureDeviceVendorID, 366
GetFaceImageType, 366
operator=, 366
operator==, 367
SetCaptureDeviceTechnology, 367
SetCaptureDeviceTypeID, 367
SetCaptureDeviceVendorID, 367
SetFaceImageType, 368
WriteTo, 368
FaceCapture, 368
Create, 370
CropFace, 370
DetectWith, 371
Draw, 371, 372
FromIsoImage, 372
IsWholeCroppedRectangleInside, 372
MaxImageDimension, 373
ResizeToMaxDimension, 373
WriteTo, 373
FaceCaptureDeviceTechnology
Face, 152
FaceCaptureDeviceTypeID, 374
FaceCaptureDeviceTypeID, 375
FaceCaptureDeviceVendorID, 375
FaceCaptureDeviceVendorID, 376
FaceCrop
Face, 152
FaceCropConfiguration, 376
FaceCropConfiguration, 377
GetCropBackground, 377
GetCropMethod, 377
GetCropScale, 377
SetCropBackground, 377
SetCropMethod, 378
SetCropScale, 378
WriteTo, 379
FaceDetector, 379
Detect, 381
FaceDetector, 380
FaceDetector::Config, 282
ACCURATE, 283
ACCURATE_SERVER, 283
BALANCED, 283
Config, 284
Default, 284
FAST, 283
GetDetector, 284
GetMaxDetectionSize, 284
GetMinDetectionSize, 284
SetDetector, 285
SetMaxDetectionSize, 285
SetMinDetectionSize, 285
Type, 283
WriteTo, 285
FaceExtractorConfig, 381
ACCURATE, 383
BALANCED, 383
Default, 383
FaceExtractorConfig, 383
FAST, 383
GetExtractor, 383
SetExtractor, 384
Type, 383
WriteTo, 384
FaceImageType
Face, 153
FaceRelativeArea
Face, 351
FaceRelativeAreaInImage
Face, 351
FaceSimilarityScore
Face, 151
FaceSize
Face, 352
FaceSizeReliability, 384
Validate, 385
FaceTemplate, 385
FaceTemplate, 386, 387
GetVersion, 388
SimilarWith, 388
WriteTo, 388
FaceToIsoCaptureDeviceTechnology
Face, 154
FaceToIsoImageType
Face, 154
FailedFaceAttribute, 389
FAST

Face, 346
FaceDetector::Config, 283
FaceExtractorConfig, 383
IrisDetector::Config, 290
IrisExtractorConfig, 512
File, 162
FileException, 389
FileName
SourceLocation, 715
FilesystemAccess, 390
ReadAll, 391
WriteAll, 391
FilterType
Image::NISTImageResampler, 629
FindLicenseFile
Manual initialisation, 181
FlatteningValidation, 393
FlatteningValidation, 395
GetInvalidatingPrint, 395
Validate, 395
FormattingComposer, 396
Compose, 398
FormattingComposer, 398
FormattingLogger, 398
FormattingLogger, 399
GetLogger, 399
LogDebug, 399
LogError, 400
LogInfo, 400
FromDPCM
ImageDpi, 455
FromFileExtension
ImageType, 475
FromIsoImage
FaceCapture, 372
IrisCapture, 506
Print, 663
FULL
Face, 153
FULL_FRONTAL
Face, 153
FULL_NOT_ALIGNED
Face, 153
FULLY_OPAQUE
RGBA, 706
FULLY_TRANSPARENT
RGBA, 706
FunctionName
SourceLocation, 715

GAUSSIAN
Image::NISTImageResampler, 629
GENDER
Face, 344
Gender
Face, 352
GenderEstimationMode
Face, 345
Get
GlobalFaceExecutor, 402
GlobalFileAccess, 404
GlobalImageExecutor, 405
GlobalIrisExecutor, 406
GlobalLogger, 407
GlobalMultiModalityExecutor, 408
GlobalPrintExecutor, 409
GetBackgroundColor
PrintCropConfiguration, 679
GetBackgroundUniformity
OFIQFace, 638
GetBitrate
WsqImageEncoder, 742
GetBottomLeft
Rectangle, 698
GetBottomRight
Rectangle, 698
GetBoundingBox
OFIQFace, 638
GetCallbacks
CallbackLogger, 271
GetCandidate
ModalitiesGallery::Candidate, 273
GetCaptureDeviceTechnology
FaceAttributes, 365
IrisAttributes, 502
PrintAttributes, 668
GetCaptureDeviceTypeID
FaceAttributes, 365
IrisAttributes, 502
PrintAttributes, 668
GetCaptureDeviceVendorID
FaceAttributes, 366
IrisAttributes, 502
PrintAttributes, 668
GetCertificationFlag
PrintAttributes, 668
GetCompression
TiffImageEncoder, 724
GetCompressionArtifacts
OFIQFace, 638
GetCompressionLevel
PngImageEncoder, 649
GetCompressionRatio
Jp2ImageEncoder, 545
GetConfidenceThreshold
NeuralSegmentation::Config, 297
GetCriticalPoints
ICSTemplate, 419
GetCropBackground
FaceCropConfiguration, 377
GetCropMethod
FaceCropConfiguration, 377
GetCropRectangle
CroppedFace, 307
GetCropScale
FaceCropConfiguration, 377
GetData

ILOSIDTemplate, 429
ISOCCTemplate, 523
ISONCTemplate, 532
GetDetector
FaceDetector::Config, 284
IrisDetector::Config, 291
GetDpi
Image::ImageResampler, 468
GetDynamicRange
OFIQFace, 639
GetEmbeddingRepresentation
ExtractedPrint, 331
PrintTemplate, 691
GetEmbeddingTemplateBiometricPosition
EmbeddingTemplate, 314
GetEnrolledFaces
Applicant, 252
GetEnrolledIrises
Applicant, 252
GetEnrolledPrints
Applicant, 252
GetExpressionNeutrality
OFIQFace, 639
GetExtractor
FaceExtractorConfig, 383
IrisExtractorConfig, 513
GetEyesOpen
OFIQFace, 639
GetEyesVisible
OFIQFace, 640
GetFaceCandidates
ModalitiesGallery::ModalitiesCandidates, 585
GetFaceCapture
CroppedFace, 307
GetFaceIdentifiers
IndexedModalitiesGallery::ModalitiesIdentifiers, 611
GetFaceImageType
FaceAttributes, 366
GetFaceIrisOpencvThreads
InitializerConfig, 485
GetFaceIrisOrtInterThreads
InitializerConfig, 485
GetFaceIrisOrtIntraThreads
InitializerConfig, 485
GetFaceOcclusionPrevention
OFIQFace, 640
GetFaceScore
ModalityScore, 624
GetFaceTemplates
ModalitiesICSTemplate, 606
ModalitiesTemplate, 615
GetGalleryMinutiae
ICSVerifyMatcher::MatchingMinutiae, 578
GetGalleryType
IndexedModalitiesGalleryConfig, 483
ModalitiesGalleryConfig, 599
GetHandPosition
GuessedHandPosition, 410
GetHeadPosePitch
OFIQFace, 640
GetHeadPoseRoll
OFIQFace, 641
GetHeadPoseYaw
OFIQFace, 641
GetHeadSize
OFIQFace, 641
GetHeight
Rectangle, 699
GetICSRepresentation
PrintTemplate, 691
GetICSTemplateBiometricPosition
ICSTemplate, 419
GetICSTemplates
ModalitiesICSTemplate, 606
GetIlluminationUniformity
OFIQFace, 642
GetImage
Face::Keypoints, 558
ICSExtractor::BinarizedImage, 266
ICSExtractor::FilteredImage, 393
ICSExtractor::SkeletonImage, 710
GetImageHeight
PrintCropConfiguration, 679
GetImageType
IrisAttributes, 502
GetImageWidth
PrintCropConfiguration, 679
GetImpression
PrintAttributes, 668
GetIndex
ModalitiesGalleryIndexedTemplate, 602
ModalitiesGalleryReference, 603
GetInterEyeDistance
OFIQFace, 642
GetInvalidatingPrint
FlatteningValidation, 395
GetIrisCandidates
ModalitiesGallery::ModalitiesCandidates, 585
GetIrisIdentifiers
IndexedModalitiesGallery::ModalitiesIdentifiers, 611
GetIrisScore
ModalityScore, 624
GetIrisTemplateBiometricPosition
IrisTemplate, 520
GetIrisTemplates
ModalitiesICSTemplate, 606
ModalitiesTemplate, 615
GetISO19794IrisImage
Iris, 498
GetKeypoints
Face, 353
GetLeftwardCropOfTheFaceImage
OFIQFace, 642
GetLicenseContent

InitializerConfig, 486
GetLogger
FormattingLogger, 399
GetLuminanceMean
OFIQFace, 643
GetLuminanceVariance
OFIQFace, 643
GetMarginAboveOfTheFaceImage
OFIQFace, 643
GetMarginBelowOfTheFaceImage
OFIQFace, 644
GetMaskColor
ImageMaskConfiguration, 463
GetMaskType
ImageMaskConfiguration, 463
GetMaxDetectionSize
FaceDetector::Config, 284
IrisDetector::Config, 291
GetMaxRotation
AlgorithmicSegmentation::Config, 280
ICSVerifyMatcher::Config, 287
ModalitiesVerifyMatcher::Config, 294
NeuralSegmentation::Config, 297
GetMinDetectionSize
FaceDetector::Config, 284
IrisDetector::Config, 291
GetMinutiae
ICSTemplate, 420
GetModalitiesICSTemplateData
ModalitiesICSTemplate, 606
GetModalitiesTemplateData
ModalitiesTemplate, 616
GetModalitiesTemplates
ModalitiesGalleryIndexedTemplate, 602
ModalitiesGalleryReference, 603
GetModelsPath
InitializerConfig, 486
GetMouthClosed
OFIQFace, 644
GetMouthOcclusionPrevention
OFIQFace, 644
GetNarrowFingerPreference
PrintDetectorConfig, 684
GetNaturalColour
OFIQFace, 645
GetNoHeadCoverings
OFIQFace, 645
GetOptimize
JpgImageEncoder, 548
GetOverExposurePrevention
OFIQFace, 645
GetPoints
Polygon, 653
GetPosition
Face::Keypoint, 555
PrintSegment, 686
GetPrintCandidates
ModalitiesGallery::ModalitiesCandidates, 586
GetPrintIdentifiers
IndexedModalitiesGallery::ModalitiesIdentifiers, 611
GetPrintScore
ModalityScore, 624
GetPrintTemplates
ModalitiesTemplate, 616
GetProbe
ModalitiesGallery::ModalitiesCandidates, 586
GetProbeMinutiae
ICSVerifyMatcher::MatchingMinutiae, 578
GetProgressive
JpgImageEncoder, 548
GetQuality
JpgImageEncoder, 548
WebpImageEncoder, 729
GetReason
ImageException, 460
GetRightwardCropOfTheFaceImage
OFIQFace, 646
GetScore
Face::Keypoint, 555
ModalitiesGallery::Candidate, 273
GetSharpness
OFIQFace, 646
GetSingleFacePresent
OFIQFace, 646
GetThreadCount
IndexedModalitiesGalleryConfig, 483
ModalitiesGalleryConfig, 599
GetTopLeft
Rectangle, 699
GetTopRight
Rectangle, 699
GetUnderExposurePrevention
OFIQFace, 647
GetUnifiedQualityScore
OFIQFace, 647
GetVersion
ANSITemplate, 242
EmbeddingTemplate, 314
FaceTemplate, 388
ICSTemplate, 420
IrisTemplate, 520
ISOTemplate, 538
GetWidth
Rectangle, 699
GLASS_FIBER
Print, 226
GLASS_STATUS
Face, 343
GlassStatus
Face, 353
GlassStatusReliability, 400
Validate, 401
GlobalFaceExecutor, 402
Get, 402
Set, 402

GlobalFileAccess, 403
Get, 404
Set, 404
GlobalImageExecutor, 404
Get, 405
Set, 405
GlobalIrisExecutor, 405
Get, 406
Set, 406
GlobalLogger, 406
Get, 407
Set, 407
GlobalMultiModalityExecutor, 407
Get, 408
Set, 408
GlobalPrintExecutor, 408
Get, 409
Set, 409
GRAYSCALE
RawImage, 694
GuessedHandPosition, 409
GetHandPosition, 410
IsGuessSuccessful, 410

HasEmbeddingRepresentation
ExtractedPrint, 332
PrintTemplate, 691
HasIcsRepresentation
PrintTemplate, 691
HasPosition
PrintSegment, 686
HEAVY_FRAME
Face, 343
HeavyFrame
Face, 353
Height
Image, 449

ICAOReliability, 410
Validate, 411
ICSExtractor, 411
Extract, 413
ICSExtractor, 412
ICSExtractor::Artifact, 259
ICSExtractor::Artifacts, 260
Add, 260
ICSExtractor::BinarizedImage, 265
GetImage, 266
ICSExtractor::FilteredImage, 392
GetImage, 393
ICSExtractor::SkeletonImage, 709
DrawCriticalPoints, 710
DrawMinutiae, 710
GetImage, 710
ICSMatcher, 414
Match, 415
ICSTemplate, 415
Anonymize, 419
CalculateQuality, 419
GetCriticalPoints, 419
GetICSTemplateBiometricPosition, 419
GetMinutiae, 420
GetVersion, 420
ICSTemplate, 417, 418
IsICSTemplateBiometricPositionKnown, 420
SimilarWith, 420
ToANSI, 421
ToISO, 421
WriteTo, 422
ICSVerifyMatcher, 422
ICSVerifyMatcher, 423, 424
Match, 424
ICSVerifyMatcher::Artifact, 259
ICSVerifyMatcher::Artifacts, 261
Add, 261
ICSVerifyMatcher::Config, 286
Config, 287
Default, 287
GetMaxRotation, 287
SetMaxRotation, 287
WriteTo, 288
ICSVerifyMatcher::MatchingMinutiae, 577
GetGalleryMinutiae, 578
GetProbeMinutiae, 578
IDEAL
Image::NISTImageResampler, 629
IdentificationEmbeddingExtractor, 425
Extract, 427
IdentificationEmbeddingExtractor, 426
Identify
ModalitiesGallery, 596
IdentifyModalities
IndexedModalitiesGallery, 481
ModalitiesGallery, 597
ILOSIDTemplate, 427
GetData, 429
ILOSIDTemplate, 428, 429
WriteTo, 429
ILOSIDTemplateAdapter, 430
Adapt, 431–433
ILOSIDTemplateAdapter, 431
Image, 164, 434
AsBGR, 438
AsBGRA, 438
AsRawBGR, 438
AsRawBGRA, 438
AsRawGrayscale, 439
AsRawImage, 439
ChangeDpiTo, 439, 440
Channels, 440
CircleIntersection, 441
CropAlignedRectangle, 441, 442
CropCircle, 443
CropRectangle, 443, 444
Decode, 445, 446
DecodeRawImage, 447
DeduceType, 447

Dpi, 448
DrawShape, 448
EncodeIn, 448
Height, 449
ImageBlurType, 179
ImageMaskType, 180
INSIDE, 180, 437
Intersection, 436
ISO_19794_BORDER_BLURRING, 180
MaskImageBy, 449
NO, 437
NO_BORDER_BLURRING, 179
operator==, 449
OriginalEncoder, 450
OUTSIDE, 180, 437
PARTIAL, 437
PointIntersection, 450
RectangleIntersection, 450
Resize, 451
ResizeToMaxDimension, 451
SAME, 437
SaveAs, 452
SaveIn, 452
Width, 452
Image::AreaImageResampler, 257
AreaImageResampler, 258
Resample, 258
Image::BicubicImageResampler, 263
BicubicImageResampler, 264
Resample, 264
Image::ImageResampler, 466
GetDpi, 468
ImageResampler, 467
Resample, 468
Image::NISTImageResampler, 628
BICUBIC, 630
BILINEAR, 630
FilterType, 629
GAUSSIAN, 629
IDEAL, 629
InterpolationMethod, 629
NISTImageResampler, 630
Resample, 631
ImageBlurType
Image, 179
ImageDimensions, 453
ImageDimensions, 454
ImageDpi, 454
FromDPCM, 455
ImageDpi, 455
ToDPCM, 455
ImageEncoder, 455
Accept, 456
Create, 456
Encode, 456
ImageEncoderVisitor, 457
ImageException, 457
DECODE_IMAGE, 459
ENCODE_IMAGE, 459
GetReason, 460
ImageException, 459
PROCESS_IMAGE, 459
Reason, 459
UNKNOWN_IMAGE, 459
ImageFileSaver, 460
ImageFileSaver, 461
Save, 461
ImageMask, 461
MaskWith, 462
ImageMaskConfiguration, 462
GetMaskColor, 463
GetMaskType, 463
ImageMaskConfiguration, 463
ImageMasker, 464
AddToBlur, 464, 465
ApplyBlur, 465
BlurCircle, 465
BlurPolygon, 465
MaskCircle, 466
MaskPolygon, 466
ImageMaskType
Image, 180
ImageResampler
Image::ImageResampler, 467
ImageSaver, 468
Save, 469
ImageShape, 469
DrawIn, 470
ImageShapeAttributes, 470
WriteTo, 471
ImageShapeDrawer, 471
DrawCircle, 472
DrawLine, 472
DrawRectangle, 472
ImageType, 472
BMP, 474
FromFileExtension, 475
ImageType, 474
IRAW, 474
JPG, 474
JPG2000, 474
PNG, 474
TIFF, 474
Type, 474
WEBP, 474
WSQ, 474
IncompatibleTemplateException, 475
IndexedModalitiesGallery, 476
Add, 480
Create, 480
IdentifyModalities, 481
Remove, 481
IndexedModalitiesGallery::ModalitiesIdentifiers, 610
GetFaceIdentifiers, 611
GetIrisIdentifiers, 611
GetPrintIdentifiers, 611

IndexedModalitiesGalleryConfig, 482
GetGalleryType, 483
GetThreadCount, 483
IndexedModalitiesGalleryConfig, 482
SetGalleryType, 483
SetThreadsCount, 483
WriteTo, 484
Initialize
Manual initialisation, 182
InitializerConfig, 484
GetFaceIrisOpencvThreads, 485
GetFaceIrisOrtInterThreads, 485
GetFaceIrisOrtIntraThreads, 485
GetLicenseContent, 486
GetModelsPath, 486
InitializerConfigBuilder, 486
Build, 487
SetFaceIrisOpencvThreads, 487
SetFaceIrisOrtInterThreads, 488
SetFaceIrisOrtIntraThreads, 488
SetLicenseContent, 488
SetLicensePath, 489
SetModelsPath, 489
UseAllAvailableCores, 489
UseSingleCore, 489
INSIDE
Image, 180, 437
InterpolationMethod
Image::NISTImageResampler, 629
Intersection
Image, 436
InvalidHWIDException, 490
InvalidImageArgumentException, 491
InvalidLicenseException, 492
InvalidTemplateException, 493
IRAW
ImageType, 474
IRawImageEncoder, 494
Accept, 495
Encode, 495
Iris, 183, 495
CalculateEyelid, 497
CMOS_CCD, 190
confidence, 500
CROPPED, 190
CROPPED_AND_MASKED, 190
Extract, 497
GetISO19794IrisImage, 498
IrisCaptureDeviceTechnology, 189
IrisImageType, 190
IrisImageTypeToIso, 190
IrisPosition, 190
IrisPositionToIsoPosition, 191
IrisSimilarityScore, 189
IrisToIsoCaptureDeviceTechnology, 191
IsoToIrisCaptureDeviceTechnology, 191
IsoToIrisImageType, 192
IsoToIrisPosition, 192
LEFT_EYE, 190
RIGHT_EYE, 190
ToIsoImage, 498
ToIsoRawImage, 499
UNCROPPED, 190
UNKNOWN, 190
VGA, 190
WriteTo, 499
IrisAttributes, 500
GetCaptureDeviceTechnology, 502
GetCaptureDeviceTypeID, 502
GetCaptureDeviceVendorID, 502
GetImageType, 502
IrisAttributes, 501
operator=, 502
operator==, 503
SetCaptureDeviceTechnology, 503
SetCaptureDeviceTypeID, 503
SetCaptureDeviceVendorID, 503
SetImageType, 504
IrisCapture, 504
Create, 505
DetectWith, 505
FromIsoImage, 506
IrisCaptureDeviceTechnology
Iris, 189
IrisCaptureDeviceTypeID, 506
IrisCaptureDeviceTypeID, 507
IrisCaptureDeviceVendorID, 507
IrisCaptureDeviceVendorID, 508
IrisDetector, 508
Detect, 510
IrisDetector, 509, 510
IrisDetector::Config, 288
ACCURATE, 290
Config, 290
Default, 291
FAST, 290
GetDetector, 291
GetMaxDetectionSize, 291
GetMinDetectionSize, 291
SetDetector, 291
SetMaxDetectionSize, 292
SetMinDetectionSize, 292
Type, 290
WriteTo, 292
IrisExtractorConfig, 511
ACCURATE, 512
Default, 513
FAST, 512
GetExtractor, 513
IrisExtractorConfig, 513
SetExtractor, 513
Type, 512
WriteTo, 513
IrisImageType
Iris, 190
IrisImageTypeToIso

Iris, 190
IrisPosition
Iris, 190
IrisPositionToIsoPosition
Iris, 191
IrisSegment, 514
WriteTo, 515
IrisSimilarityScore
Iris, 189
IrisTemplate, 515
CalculateQuality, 520
GetIrisTemplateBiometricPosition, 520
GetVersion, 520
IrisTemplate, 517–519
IsIrisTemplateBiometricPositionKnown, 521
SimilarWith, 521
WriteTo, 521
IrisToIsoCaptureDeviceTechnology
Iris, 191
IsAlignedCroppingEnabled
PrintCropConfiguration, 680
IsCropRegionInsideImage
CroppedFace, 307
IsEmbeddingTemplateBiometricPositionKnown
EmbeddingTemplate, 314
IsFaceCropMethodValid
Face, 155
IsGuessSuccessful
GuessedHandPosition, 410
IsICSTemplateBiometricPositionKnown
ICSTemplate, 420
IsIrisTemplateBiometricPositionKnown
IrisTemplate, 521
IsLossless
Jp2ImageEncoder, 545
IsLossy
Jp2ImageEncoder, 545
IsNoiseReduction
AlgorithmicSegmentation::Config, 281
ISO_19794_BORDER_BLURRING
Image, 180
ISOCCTemplate, 522
GetData, 523
ISOCCTemplate, 522, 523
WriteTo, 523
ISOCCTemplateAdapter, 524
Adapt, 526–529
ISOCCTemplateAdapter, 525, 526
MinutiaSortOrder, 525
NONE, 525
X_ASC, 525
X_DESC, 525
Y_ASC, 525
Y_DESC, 525
ISOFullFrontalImageValidator, 530
Validate, 530
ISONCTemplate, 531
GetData, 532
ISONCTemplate, 532
WriteTo, 532
ISONCTemplateAdapter, 533
Adapt, 534–536
ISONCTemplateAdapter, 534
ISOTemplate, 537
GetVersion, 538
ISOTemplate, 538
WriteTo, 538
ISOTemplateAdapter, 539
Adapt, 541–543
ISOTemplateAdapter, 540
Version, 540
VERSION_2, 540
VERSION_3, 540
IsoToFaceCaptureDeviceTechnology
Face, 155
IsoToFaceImageType
Face, 155
IsoToIrisCaptureDeviceTechnology
Iris, 191
IsoToIrisImageType
Iris, 192
IsoToIrisPosition
Iris, 192
IsoToPrintCaptureDeviceTechnology
Print, 230
IsoToPrintImpression
Print, 230
IsoToPrintPosition
Print, 230
IsSameAs
Face, 354
Print, 663
IsWholeCroppedRectangleInside
FaceCapture, 372
IT8BL
TiffImageEncoder, 723
IT8CTPAD
TiffImageEncoder, 723
IT8LW
TiffImageEncoder, 723
IT8MP
TiffImageEncoder, 723

Java, 120
JP2000
TiffImageEncoder, 724
Jp2ImageEncoder, 544
Accept, 545
GetCompressionRatio, 545
IsLossless, 545
IsLossy, 545
SetCompressionRatio, 546
JPEG
TiffImageEncoder, 723
JPG
ImageType, 474
JPG2000

ImageType, 474
JpgImageEncoder, 546
Accept, 548
GetOptimize, 548
GetProgressive, 548
GetQuality, 548
JpgImageEncoder, 548
SetOptimize, 549
SetProgressive, 549
SetQuality, 549
JsonWriter, 550
Output, 551
Write, 551–553
WriteNull, 554
WriteRoot, 554

LATENT_IMPRESSION
Print, 227
LATENT_LIFT
Print, 227
LATENT_PALM_IMPRESSION
Print, 227
LATENT_PALM_LIFT
Print, 227
LATENT_PALM_PHOTO
Print, 227
LATENT_PALM_TRACING
Print, 227
LATENT_PHOTO
Print, 227
LATENT_TRACING
Print, 227
LEFT_EYE
Iris, 190
LEFT_EYE_STATUS
Face, 343
LEFT_FULL_PALM
Print, 229
LEFT_HYPOTHENAR
Print, 229
LEFT_INDEX
Print, 228
LEFT_INTERDIGITAL
Print, 229
LEFT_LITTLE
Print, 229
LEFT_LOWER_PALM
Print, 229
LEFT_MIDDLE
Print, 228
LEFT_RED_EYE
Face, 344
LEFT_RING
Print, 229
LEFT_THENAR
Print, 229
LEFT_THUMB
Print, 228
LEFT_UPPER_PALM
Print, 229
LEFT_WRITERS_PALM
Print, 229
LeftEdge
Face::Keypoints, 558
LeftEyebrowInnerEnd
Face::Keypoints, 559
LeftEyebrowOuterEnd
Face::Keypoints, 559
LeftEyeCentre
Face::Keypoints, 559
LeftEyeInnerCorner
Face::Keypoints, 560
LeftEyeOuterCorner
Face::Keypoints, 560
LeftEyeStatus
Face, 354
LeftRedEye
Face, 355
LicenseException, 566
LicenseFeatureNotAllowedException, 567
LicenseLimitReachedException, 568
LicenseNotFoundException, 569
Line, 570
Line, 571
SourceLocation, 716
WriteTo, 571
LineShape, 572
DrawIn, 573
LineShape, 572
LIVE_SCAN_OPTICAL_CONCTACTLESS_PLAIN
Print, 227
LIVE_SCAN_PALM
Print, 227
LIVE_SCAN_PLAIN
Print, 227
LIVE_SCAN_ROLLED
Print, 227
LIVE_SCAN_SWIPE
Print, 227
LIVE_SCAN_VERTICAL_ROLL
Print, 227
Location
EnrollmentException, 319
log
LoggerCallBacks, 575
LogDebug
CallbackLogger, 271
FormattingLogger, 399
Logger, 574
LogError
CallbackLogger, 271
FormattingLogger, 400
Logger, 574
Logger, 193, 573
LogDebug, 574
LogError, 574
LoggerCallback, 196

LogInfo, 574
LogMessageCallback, 196
LoggerCallback
Logger, 196
LoggerCallBacks, 574
log, 575
LogInfo
CallbackLogger, 272
FormattingLogger, 400
Logger, 574
LogMessage, 576
Compose, 576
LogMessageCallback
Logger, 196
LZW
TiffImageEncoder, 723

Manual initialisation, 180
DefaultModelsDirectory, 181
FindLicenseFile, 181
Initialize, 182
MaskCircle
ImageMasker, 466
MaskImageBy
Image, 449
MaskPolygon
ImageMasker, 466
MaskWith
CircleImageMask, 277
ImageMask, 462
PolygonImageMask, 655
Match
ApplicantMatcher, 255
EmbeddingMatcher, 309
EmbeddingVerifyMatcher, 317
ICSMatcher, 415
ICSVerifyMatcher, 424
ModalitiesMatcher, 612, 613
ModalitiesVerifyMatcher, 620, 621
MatchFaces
ModalitiesVerifyMatcher, 621
MatchIrises
ModalitiesVerifyMatcher, 622
MatchPrints
ModalitiesEmbeddingVerifyMatcher, 589
ModalitiesVerifyMatcher, 622
MaxImageDimension
FaceCapture, 373
MECHANICAL
Print, 226
Minutia, 578
BIFURCATION, 580
Minutia, 580
quality, 581
RIDGE_ENDING, 580
Type, 580
WriteTo, 581
MinutiaeShape, 581
DrawIn, 582
MinutiaeShape, 582
MinutiaShape, 583
DrawIn, 584
MinutiaShape, 584
MinutiaSortOrder
ISOCCTemplateAdapter, 525
ModalitiesEmbeddingVerifyMatcher, 586
MatchPrints, 589
ModalitiesEmbeddingVerifyMatcher, 588
ModalitiesGallery, 590
Add, 594
Consolidate, 594
Create, 595, 596
Identify, 596
IdentifyModalities, 597
Remove, 597
ModalitiesGallery::Candidate, 272
GetCandidate, 273
GetScore, 273
ModalitiesGallery::ModalitiesCandidates, 585
GetFaceCandidates, 585
GetIrisCandidates, 585
GetPrintCandidates, 586
GetProbe, 586
ModalitiesGalleryConfig, 598
GetGalleryType, 599
GetThreadCount, 599
ModalitiesGalleryConfig, 599
SetGalleryType, 599
SetThreadsCount, 599
ToIndexedModalitiesGalleryConfig, 600
WriteTo, 600
ModalitiesGalleryIndexedTemplate, 600
Downcast, 602
GetIndex, 602
GetModalitiesTemplates, 602
ModalitiesGalleryIndexedTemplate, 601
ModalitiesGalleryReference, 603
GetIndex, 603
GetModalitiesTemplates, 603
ModalitiesICSTemplate, 604
GetFaceTemplates, 606
GetICSTemplates, 606
GetIrisTemplates, 606
GetModalitiesICSTemplateData, 606
ModalitiesICSTemplate, 605, 606
WriteTo, 607
ModalitiesICSTemplateDefaultFactory, 607
CreateModalitiesICSTemplate, 609
ModalitiesICSTemplateDefaultFactory, 608
ModalitiesICSTemplateFactory, 609
CreateModalitiesICSTemplate, 610
ModalitiesMatcher, 612
Match, 612, 613
ModalitiesTemplate, 613
Create, 615
GetFaceTemplates, 615
GetIrisTemplates, 615

GetModalitiesTemplateData, 616
GetPrintTemplates, 616
SimilarWith, 616
ToModalitiesICSTemplate, 616, 617
WriteTo, 617
ModalitiesVerifyMatcher, 618
Match, 620, 621
MatchFaces, 621
MatchIrises, 622
MatchPrints, 622
ModalitiesVerifyMatcher, 619, 620
ModalitiesVerifyMatcher::Config, 293
Config, 294
Default, 294
GetMaxRotation, 294
SetMaxRotation, 294
WriteTo, 295
ModalityScore, 623
GetFaceScore, 624
GetIrisScore, 624
GetPrintScore, 624
ModalityScore, 624
WriteTo, 624
MONOCHROMATIC_VISIBLE_IR_OPTICAL
Print, 226
MONOCHROMATIC_VISIBLE_IR_OPTICAL_TIR
Print, 226
MONOCHROMATIC_VISIBLE_IR_OPTICALTOUCHLESS
Print, 226
MONOCHROMATIC_VISIBLE_OPTICAL_DIRECT
Print, 226
MONOCHROMATIC_VISIBLE_OPTICAL_TIR
Print, 226
MONOCHROMATIC_VISIBLE_OPTICAL_TOUCHLESS
Print, 226
MOUTH_STATUS
Face, 343
MouthCenter
Face::Keypoints, 560
MouthLeftCorner
Face::Keypoints, 561
MouthLowerEdge
Face::Keypoints, 561
MouthOpennessReliability, 625
Validate, 626
MouthRightCorner
Face::Keypoints, 561
MouthStatus
Face, 355
MouthUpperEdge
Face::Keypoints, 562
MULTISPECTRAL_OPTICAL_DIRECT
Print, 226
MULTISPECTRAL_OPTICAL_TIR
Print, 226
MULTISPECTRAL_OPTICAL_TOUCHLESS
Print, 226

NeuralSegmentation, 626
NeuralSegmentation, 627, 628
NeuralSegmentation::Config, 295
Config, 296
Default, 297
GetConfidenceThreshold, 297
GetMaxRotation, 297
SetConfidenceThreshold, 297
SetMaxRotation, 298
WriteTo, 298
NEXT
TiffImageEncoder, 723
NISTImageResampler
Image::NISTImageResampler, 630
NO
Image, 437
NO_BORDER_BLURRING
Image, 179
NONE
ISOCCTemplateAdapter, 525
TiffImageEncoder, 723
NONLIVE_SCAN_PALM
Print, 227
NONLIVE_SCAN_PLAIN
Print, 227
NONLIVE_SCAN_ROLLED
Print, 227
NOSE_SHADOW
Face, 342
NoseBottom
Face::Keypoints, 562
NoseLeftBottom
Face::Keypoints, 562
NoseRightBottom
Face::Keypoints, 563
NoseRoot
Face::Keypoints, 563
NoseShadow
Face, 355
NoseTip
Face::Keypoints, 563
NotSupportedException, 632
NullArgumentException, 633
NullLogger, 634

OFIQFace, 634
Create, 637
GetBackgroundUniformity, 638
GetBoundingBox, 638
GetCompressionArtifacts, 638
GetDynamicRange, 639
GetExpressionNeutrality, 639
GetEyesOpen, 639
GetEyesVisible, 640
GetFaceOcclusionPrevention, 640
GetHeadPosePitch, 640
GetHeadPoseRoll, 641
GetHeadPoseYaw, 641
GetHeadSize, 641
GetIlluminationUniformity, 642

GetInterEyeDistance, 642
GetLeftwardCropOfTheFaceImage, 642
GetLuminanceMean, 643
GetLuminanceVariance, 643
GetMarginAboveOfTheFaceImage, 643
GetMarginBelowOfTheFaceImage, 644
GetMouthClosed, 644
GetMouthOcclusionPrevention, 644
GetNaturalColour, 645
GetNoHeadCoverings, 645
GetOverExposurePrevention, 645
GetRightwardCropOfTheFaceImage, 646
GetSharpness, 646
GetSingleFacePresent, 646
GetUnderExposurePrevention, 647
GetUnifiedQualityScore, 647
WriteTo, 647
operator!=
Face, 356
Print, 663
operator=
FaceAttributes, 366
IrisAttributes, 502
PrintAttributes, 668, 669
operator==
Face, 356
FaceAttributes, 367
Image, 449
IrisAttributes, 503
Print, 664
PrintAttributes, 669
OriginalEncoder
Image, 450
OTHER
Print, 227
Output
JsonWriter, 551
OUTSIDE
Image, 180, 437

PACKBITS
TiffImageEncoder, 723
PARTIAL
Image, 437
PassiveLiveness
Face, 356, 357
PassiveLivenessMode
Face, 345
PITCH_ANGLE
Face, 344
PitchAngle
Face, 357
PIXARFILM
TiffImageEncoder, 723
PIXARLOG
TiffImageEncoder, 723
PNG
ImageType, 474
PngImageEncoder, 648
Accept, 649
GetCompressionLevel, 649
SetCompressionLevel, 649
Point, 650
Point, 651
WriteTo, 651
PointIntersection
Image, 450
Polygon, 651
AddPoint, 652
GetPoints, 653
Polygon, 652
WriteTo, 653
PolygonImageMask, 653
MaskWith, 655
PolygonImageMask, 654
POST_PROCESSED_FRONTAL
Face, 154
PRESSURE_SENSITIVE
Print, 226
Print, 196, 655
CalculateImageQuality, 658
CalculateIntensity, 658
CalculateNFIQ1, 659
CalculateNFIQ2, 659
CalculatePlacement, 660
CalculatePresence, 660
ChangeDpiTo, 660, 661
Create, 661
DrawCriticalPoints, 662
DrawMinutiae, 662
ELECTRO_LUMINESCENT, 226
FromIsoImage, 663
GLASS_FIBER, 226
IsoToPrintCaptureDeviceTechnology, 230
IsoToPrintImpression, 230
IsoToPrintPosition, 230
IsSameAs, 663
LATENT_IMPRESSION, 227
LATENT_LIFT, 227
LATENT_PALM_IMPRESSION, 227
LATENT_PALM_LIFT, 227
LATENT_PALM_PHOTO, 227
LATENT_PALM_TRACING, 227
LATENT_PHOTO, 227
LATENT_TRACING, 227
LEFT_FULL_PALM, 229
LEFT_HYPOTHENAR, 229
LEFT_INDEX, 228
LEFT_INTERDIGITAL, 229
LEFT_LITTLE, 229
LEFT_LOWER_PALM, 229
LEFT_MIDDLE, 228
LEFT_RING, 229
LEFT_THENAR, 229
LEFT_THUMB, 228
LEFT_UPPER_PALM, 229
LEFT_WRITERS_PALM, 229

LIVE_SCAN_OPTICAL_CONCTACTLESS_PLAIN, 227
LIVE_SCAN_PALM, 227
LIVE_SCAN_PLAIN, 227
LIVE_SCAN_ROLLED, 227
LIVE_SCAN_SWIPE, 227
LIVE_SCAN_VERTICAL_ROLL, 227
MECHANICAL, 226
MONOCHROMATIC_VISIBLE_IR_OPTICAL, 226
MONOCHROMATIC_VISIBLE_IR_OPTICAL_TIR, 226
MONOCHROMATIC_VISIBLE_IR_OPTICALTOUCHLESS, 226
MONOCHROMATIC_VISIBLE_OPTICAL_DIRECT, 226
MONOCHROMATIC_VISIBLE_OPTICAL_TIR, 226
MONOCHROMATIC_VISIBLE_OPTICAL_TOUCHLESS, 226
MULTISPECTRAL_OPTICAL_DIRECT, 226
MULTISPECTRAL_OPTICAL_TIR, 226
MULTISPECTRAL_OPTICAL_TOUCHLESS, 226
NONLIVE_SCAN_PALM, 227
NONLIVE_SCAN_PLAIN, 227
NONLIVE_SCAN_ROLLED, 227
operator!=, 663
operator==, 664
OTHER, 227
PRESSURE_SENSITIVE, 226
PrintCaptureDeviceTechnology, 226
PrintImpression, 226
PrintPosition, 227
PrintSimilarityScore, 225
PrintToIsoCaptureDeviceTechnology, 231
PrintToIsoImpression, 231
PrintToIsoPosition, 231
RIGHT_FULL_PALM, 229
RIGHT_HYPOTHENAR, 229
RIGHT_INDEX, 228
RIGHT_INTERDIGITAL, 229
RIGHT_LITTLE, 228
RIGHT_LOWER_PALM, 229
RIGHT_MIDDLE, 228
RIGHT_RING, 228
RIGHT_THENAR, 229
RIGHT_THUMB, 228
RIGHT_UPPER_PALM, 229
RIGHT_WRITERS_PALM, 229
SEMICONDUCTOR_CAPACITIVE, 226
SEMICONDUCTOR_RF, 226
SEMICONDUCTOR_THERMAL, 226
ToIsoImage, 664
ToIsoRawImage, 665
ULTRASOUND, 226
UNKNOWN, 226, 227
WHITE_LIGHT_OPTICAL_DIRECT, 226
WHITE_LIGHT_OPTICAL_TIR, 226
WHITE_LIGHT_OPTICAL_TOUCHLESS, 226
WriteTo, 665
PrintAttributes, 665
GetCaptureDeviceTechnology, 668
GetCaptureDeviceTypeID, 668
GetCaptureDeviceVendorID, 668
GetCertificationFlag, 668
GetImpression, 668
operator=, 668, 669
operator==, 669
PrintAttributes, 667
SetCaptureDeviceTechnology, 669
SetCaptureDeviceTypeID, 670
SetCaptureDeviceVendorID, 670
SetCertificationFlag, 670
SetImpression, 670
WriteTo, 671
PrintCapture, 671
CalculateIntensity, 672
ChangeDpiTo, 673
Create, 674
CropPrint, 674
Draw, 675
SegmentWith, 675
WriteTo, 676
PrintCaptureDeviceTechnology
Print, 226
PrintCaptureDeviceTypeID, 676
PrintCaptureDeviceTypeID, 677
PrintCaptureDeviceVendorID, 677
PrintCaptureDeviceVendorID, 678
PrintCropConfiguration, 678
EnableAlignedCropping, 679
GetBackgroundColor, 679
GetImageHeight, 679
GetImageWidth, 679
IsAlignedCroppingEnabled, 680
PrintCropConfiguration, 679
SetBackgroundColor, 680
SetImageDimensions, 680
PrintDetector, 681
DoDetect, 682
PrintDetector, 682
Segment, 682
PrintDetectorConfig, 683
GetNarrowFingerPreference, 684
PrintDetectorConfig, 684
SetNarrowFingerPreference, 684
WriteTo, 684
PrintImpression
Print, 226
PrintPosition
Print, 227
PrintSegment, 685
GetPosition, 686
HasPosition, 686
Resize, 686
SetPosition, 687
ToPrint, 687

WriteTo, 688
PrintSimilarityScore
Print, 225
PrintTemplate, 688
GetEmbeddingRepresentation, 691
GetICSRepresentation, 691
HasEmbeddingRepresentation, 691
HasIcsRepresentation, 691
PrintTemplate, 690
ToEmbeddingTemplates, 691
ToICSTemplates, 692
WriteTo, 692
PrintToIsoCaptureDeviceTechnology
Print, 231
PrintToIsoImpression
Print, 231
PrintToIsoPosition
Print, 231
PROCESS_IMAGE
ImageException, 459

quality
Minutia, 581

RawImage, 693
BGR, 694
BGRA, 694
Data, 695
GRAYSCALE, 694
RawImage, 694, 695
Type, 694
WriteTo, 695
ReadAll
BinaryFileAccess, 267
FilesystemAccess, 391
Reason
ImageException, 459
Rectangle, 696
CalculateCenter, 698
Dimension, 697
GetBottomLeft, 698
GetBottomRight, 698
GetHeight, 699
GetTopLeft, 699
GetTopRight, 699
GetWidth, 699
Rectangle, 697, 698
Resize, 699
WriteTo, 700
RectangleIntersection
Image, 450
RectangleShape, 700
DrawIn, 701
RectangleShape, 701
RectanglesShape, 702
DrawIn, 703
RectanglesShape, 702
Remove
IndexedModalitiesGallery, 481
ModalitiesGallery, 597
Resample
Image::AreaImageResampler, 258
Image::BicubicImageResampler, 264
Image::ImageResampler, 468
Image::NISTImageResampler, 631
Resize
Image, 451
PrintSegment, 686
Rectangle, 699
ResizeToMaxDimension
FaceCapture, 373
Image, 451
RGB, 703
WriteTo, 704
RGBA, 704
FULLY_OPAQUE, 706
FULLY_TRANSPARENT, 706
WriteTo, 706
RIDGE_ENDING
Minutia, 580
RIGHT_EYE
Iris, 190
RIGHT_EYE_STATUS
Face, 343
RIGHT_FULL_PALM
Print, 229
RIGHT_HYPOTHENAR
Print, 229
RIGHT_INDEX
Print, 228
RIGHT_INTERDIGITAL
Print, 229
RIGHT_LITTLE
Print, 228
RIGHT_LOWER_PALM
Print, 229
RIGHT_MIDDLE
Print, 228
RIGHT_RED_EYE
Face, 343
RIGHT_RING
Print, 228
RIGHT_THENAR
Print, 229
RIGHT_THUMB
Print, 228
RIGHT_UPPER_PALM
Print, 229
RIGHT_WRITERS_PALM
Print, 229
RightEdge
Face::Keypoints, 564
RightEyebrowInnerEnd
Face::Keypoints, 564
RightEyebrowOuterEnd
Face::Keypoints, 564
RightEyeCentre

Face::Keypoints, 565
RightEyeInnerCorner
Face::Keypoints, 565
RightEyeOuterCorner
Face::Keypoints, 565
RightEyeStatus
Face, 358
RightRedEye
Face, 358
ROLL_ANGLE
Face, 344
RollAngle
Face, 359

SAME
Image, 437
Save
ImageFileSaver, 461
ImageSaver, 469
SaveAs
Image, 452
SaveIn
Image, 452
Segment
PrintDetector, 682
SegmentationMaskReliability, 706
Validate, 707
SegmentWith
PrintCapture, 675
SlapCapture, 713
ThumbsCapture, 721
SEMICONDUCTOR_CAPACITIVE
Print, 226
SEMICONDUCTOR_RF
Print, 226
SEMICONDUCTOR_THERMAL
Print, 226
Serializable, 707
WriteTo, 708
Serialization, 232
Set
GlobalFaceExecutor, 402
GlobalFileAccess, 404
GlobalImageExecutor, 405
GlobalIrisExecutor, 406
GlobalLogger, 407
GlobalMultiModalityExecutor, 408
GlobalPrintExecutor, 409
SetBackgroundColor
PrintCropConfiguration, 680
SetBitrate
WsqImageEncoder, 742
SetBytesStrategy
Writer, 736
SetCaptureDeviceTechnology
FaceAttributes, 367
IrisAttributes, 503
PrintAttributes, 669
SetCaptureDeviceTypeID
FaceAttributes, 367
IrisAttributes, 503
PrintAttributes, 670
SetCaptureDeviceVendorID
FaceAttributes, 367
IrisAttributes, 503
PrintAttributes, 670
SetCertificationFlag
PrintAttributes, 670
SetCompression
TiffImageEncoder, 724
SetCompressionLevel
PngImageEncoder, 649
SetCompressionRatio
Jp2ImageEncoder, 546
SetConfidenceThreshold
NeuralSegmentation::Config, 297
SetCropBackground
FaceCropConfiguration, 377
SetCropMethod
FaceCropConfiguration, 378
SetCropScale
FaceCropConfiguration, 378
SetDetector
FaceDetector::Config, 285
IrisDetector::Config, 291
SetExtractor
FaceExtractorConfig, 384
IrisExtractorConfig, 513
SetFaceImageType
FaceAttributes, 368
SetFaceIrisOpencvThreads
InitializerConfigBuilder, 487
SetFaceIrisOrtInterThreads
InitializerConfigBuilder, 488
SetFaceIrisOrtIntraThreads
InitializerConfigBuilder, 488
SetGalleryType
IndexedModalitiesGalleryConfig, 483
ModalitiesGalleryConfig, 599
SetImageDimensions
PrintCropConfiguration, 680
SetImageType
IrisAttributes, 504
SetImpression
PrintAttributes, 670
SetLicenseContent
InitializerConfigBuilder, 488
SetLicensePath
InitializerConfigBuilder, 489
SetMaxDetectionSize
FaceDetector::Config, 285
IrisDetector::Config, 292
SetMaxRotation
AlgorithmicSegmentation::Config, 281
ICSVerifyMatcher::Config, 287
ModalitiesVerifyMatcher::Config, 294
NeuralSegmentation::Config, 298

SetMinDetectionSize
FaceDetector::Config, 285
IrisDetector::Config, 292
SetModelsPath
InitializerConfigBuilder, 489
SetNarrowFingerPreference
PrintDetectorConfig, 684
SetOptimize
JpgImageEncoder, 549
SetPosition
PrintSegment, 687
SetProgressive
JpgImageEncoder, 549
SetQuality
JpgImageEncoder, 549
WebpImageEncoder, 729
SetThreadsCount
IndexedModalitiesGalleryConfig, 483
ModalitiesGalleryConfig, 599
SGILOG
TiffImageEncoder, 723
SGILOG24
TiffImageEncoder, 724
SHADOW
Face, 342
Shadow
Face, 359
Shapes, 708
SHARPNESS
Face, 342
Sharpness
Face, 359
SimilarWith
Applicant, 252
EmbeddingTemplate, 315
FaceTemplate, 388
ICSTemplate, 420
IrisTemplate, 521
ModalitiesTemplate, 616
SlapCapture, 711
CalculateGlobalAngle, 713
Create, 713
SegmentWith, 713
Solvers, 156
SourceLocation, 714
FileName, 715
FunctionName, 715
Line, 716
SourceLocation, 715
ToString, 716
SPECULARITY
Face, 342
Specularity
Face, 360
STATIC_PHOTOGRAPH_FROM_DIGITAL_CAMERA
Face, 152
STATIC_PHOTOGRAPH_FROM_SCANNER
Face, 152
STATIC_PHOTOGRAPH_FROM_UNKNOWN_SOURCE
Face, 152
SupportedImageEncoder, 716
Encode, 717

TemplateReliability, 718
Validate, 718
ThumbsCapture, 719
Create, 721
SegmentWith, 721
THUNDERSCAN
TiffImageEncoder, 723
TIFF
ImageType, 474
TiffImageEncoder, 722
Accept, 724
ADOBE_DEFLATE, 723
CCITT_T4, 723
CCITT_T6, 723
CCITTFAX3, 723
CCITTFAX4, 723
CCITTRLE, 723
CCITTRLEW, 723
Compression, 723
DCS, 723
DEFLATE, 723
GetCompression, 724
IT8BL, 723
IT8CTPAD, 723
IT8LW, 723
IT8MP, 723
JP2000, 724
JPEG, 723
LZW, 723
NEXT, 723
NONE, 723
PACKBITS, 723
PIXARFILM, 723
PIXARLOG, 723
SetCompression, 724
SGILOG, 723
SGILOG24, 724
THUNDERSCAN, 723
TiffImageEncoder, 724
TINTED_GLASSES
Face, 345
TintedGlasses
Face, 360
TintedGlassesReliability, 725
Validate, 725
ToANSI
ExtractedPrint, 332
ICSTemplate, 421
ToDPCM
ImageDpi, 455
ToEmbeddingTemplates
PrintTemplate, 691
ToICSTemplate
Applicant, 253

ToICSTemplates
ExtractedPrint, 332
PrintTemplate, 692
ToIndexedModalitiesGalleryConfig
ModalitiesGalleryConfig, 600
ToISO
ExtractedPrint, 332
ICSTemplate, 421
ToIsoImage
Face, 360
Iris, 498
Print, 664
ToIsoRawImage
Iris, 499
Print, 665
TOKEN
Face, 153
TOKEN_FRONTAL
Face, 154
TOKEN_NOT_FRONTAL
Face, 153
ToModalitiesICSTemplate
ModalitiesTemplate, 616, 617
ToPrint
PrintSegment, 687
ToString
EnrollmentException, 319
SourceLocation, 716
ToTemplate
Applicant, 253
ToTemplates
ExtractedFace, 324
ExtractedIris, 327
ExtractedPrint, 333
Type
CriticalPoint, 300
FaceDetector::Config, 283
FaceExtractorConfig, 383
ImageType, 474
IrisDetector::Config, 290
IrisExtractorConfig, 512
Minutia, 580
RawImage, 694

ULTRASOUND
Print, 226
UNCROPPED
Iris, 190
UNIQUE_INTENSITY_LEVELS
Face, 342
UniqueIntensityLevels
Face, 361
UniversalPassiveLivenessReliability, 726
Validate, 726
UNKNOWN
Face, 152
Iris, 190
Print, 226, 227
UNKNOWN_IMAGE
ImageException, 459
UnsupportedTemplateException, 727
UseAllAvailableCores
InitializerConfigBuilder, 489
UseSingleCore
InitializerConfigBuilder, 489

Validate
AgeReliability, 238
ApplicantFaceValidation, 254
ApplicantPrintValidation, 256
FaceAttributeReliability, 363
FaceSizeReliability, 385
FlatteningValidation, 395
GlassStatusReliability, 401
ICAOReliability, 411
ISOFullFrontalImageValidator, 530
MouthOpennessReliability, 626
SegmentationMaskReliability, 707
TemplateReliability, 718
TintedGlassesReliability, 725
UniversalPassiveLivenessReliability, 726
Version
ISOTemplateAdapter, 540
Version History and Legal Notices, 53
VERSION_2
ISOTemplateAdapter, 540
VERSION_3
ISOTemplateAdapter, 540
VGA
Iris, 190
VIDEO_FRAME_FROM_ANALOGUE_CAMERA
Face, 152
VIDEO_FRAME_FROM_DIGITAL_CAMERA
Face, 152
VIDEO_FRAME_FROM_UNKNOWN_SOURCE
Face, 152

WEBP
ImageType, 474
WebpImageEncoder, 728
Accept, 729
GetQuality, 729
SetQuality, 729
what
EnrollmentException, 319
WHITE_LIGHT_OPTICAL_DIRECT
Print, 226
WHITE_LIGHT_OPTICAL_TIR
Print, 226
WHITE_LIGHT_OPTICAL_TOUCHLESS
Print, 226
Width
Image, 452
WIDTH_HEIGHT_RATIO
Face, 345
WidthHeightRatio
Face, 361
Write

JsonWriter, 551–553
Writer, 737–739
WriteAll
BinaryFileAccess, 267
FilesystemAccess, 391
WriteArray
Writer, 740
WriteBytes
Base64Strategy, 262
BytesWriterStrategy, 269
WriteNull
JsonWriter, 554
Writer, 740
Writer, 730
SetBytesStrategy, 736
Write, 737–739
WriteArray, 740
WriteNull, 740
WriteRoot, 740
WriteRoot
JsonWriter, 554
Writer, 740
WriteTo
AlgorithmicSegmentation::Config, 281
ANSITemplate, 242
Applicant, 253
Circle, 275
CriticalPoint, 300
EmbeddingTemplate, 315
ExtractedFace, 324
ExtractedIris, 327
ExtractedPrint, 333
ExtractedPrintEmbedding, 337
Face, 361
Face::Keypoint, 556
Face::Keypoints, 566
FaceAttributes, 368
FaceCapture, 373
FaceCropConfiguration, 379
FaceDetector::Config, 285
FaceExtractorConfig, 384
FaceTemplate, 388
ICSTemplate, 422
ICSVerifyMatcher::Config, 288
ILOSIDTemplate, 429
ImageShapeAttributes, 471
IndexedModalitiesGalleryConfig, 484
Iris, 499
IrisDetector::Config, 292
IrisExtractorConfig, 513
IrisSegment, 515
IrisTemplate, 521
ISOCCTemplate, 523
ISONCTemplate, 532
ISOTemplate, 538
Line, 571
Minutia, 581
ModalitiesGalleryConfig, 600
ModalitiesICSTemplate, 607
ModalitiesTemplate, 617
ModalitiesVerifyMatcher::Config, 295
ModalityScore, 624
NeuralSegmentation::Config, 298
OFIQFace, 647
Point, 651
Polygon, 653
Print, 665
PrintAttributes, 671
PrintCapture, 676
PrintDetectorConfig, 684
PrintSegment, 688
PrintTemplate, 692
RawImage, 695
Rectangle, 700
RGB, 704
RGBA, 706
Serializable, 708
WSQ
ImageType, 474
WsqImageEncoder, 741
Accept, 742
GetBitrate, 742
SetBitrate, 742

X_ASC
ISOCCTemplateAdapter, 525
X_DESC
ISOCCTemplateAdapter, 525

Y_ASC
ISOCCTemplateAdapter, 525
Y_DESC
ISOCCTemplateAdapter, 525
YAW_ANGLE
Face, 344
YawAngle
Face, 362