Evaluation of Latent Fingerprint Technology

Application Programming Interface

0.1 Main Page	1
0.1.1 Overview	1
0.1.2 Implementation	1
0.1.3 Contact	1
0.1.4 License	1
0.2 Namespace Documentation	1
0.2.1 ELFT Namespace Reference	1
0.2.1.1 Enumeration Type Documentation	4
0.2.1.2 Variable Documentation	8
0.3 Class Documentation	9
0.3.1 ELFT::Candidate Struct Reference	9
0.3.1.1 Detailed Description	10
0.3.1.2 Constructor & Destructor Documentation	10
0.3.1.3 Member Function Documentation	10
0.3.1.4 Member Data Documentation	10
0.3.2 ELFT::ProductIdentifier::CBEFFIdentifier Struct Reference	1
0.3.2.1 Detailed Description	1
	12
0.3.3 ELFT::Coordinate Struct Reference	12
0.3.3.1 Detailed Description	12
0.3.3.2 Constructor & Destructor Documentation	12
0.3.3.3 Member Function Documentation	13
0.3.3.4 Member Data Documentation	13
0.3.4 ELFT::Correspondence Struct Reference	14
0.3.4.1 Detailed Description	14
0.3.4.2 Constructor & Destructor Documentation	14
0.3.4.3 Member Data Documentation	15
0.3.5 ELFT::CreateTemplateResult Struct Reference	15
0.3.5.1 Detailed Description	16
0.3.5.2 Member Data Documentation	16
0.3.6 ELFT::EFS Struct Reference	16
0.3.6.1 Detailed Description	17
0.3.6.2 Member Data Documentation	18
0.3.7 ELFT::ExtractionInterface Class Reference	21
0.3.7.1 Detailed Description	22
0.3.7.2 Constructor & Destructor Documentation	22
0.3.7.3 Member Function Documentation	22
0.3.8 ELFT::Image Struct Reference	26
0.3.8.1 Detailed Description	26
0.3.8.2 Constructor & Destructor Documentation	27
0.3.8.3 Member Data Documentation	27
0.3.9 ELFT::Minutia Struct Reference	29
0.3.9.1 Detailed Description	29

	0.3.9.2 Constructor & Destructor Documentation	29
	0.3.9.3 Member Data Documentation	30
	0.3.10 ELFT::ProductIdentifier Struct Reference	30
	0.3.10.1 Detailed Description	31
	0.3.10.2 Member Data Documentation	31
	0.3.11 ELFT::ReturnStatus Struct Reference	31
	0.3.11.1 Detailed Description	32
	0.3.11.2 Member Enumeration Documentation	32
	0.3.11.3 Member Function Documentation	32
	0.3.11.4 Member Data Documentation	32
	0.3.12 ELFT::RidgeQualityRegion Struct Reference	33
	0.3.12.1 Detailed Description	33
	0.3.12.2 Member Data Documentation	33
	0.3.13 ELFT::SearchInterface Class Reference	34
	0.3.13.1 Detailed Description	35
	0.3.13.2 Constructor & Destructor Documentation	35
	0.3.13.3 Member Function Documentation	35
	0.3.14 ELFT::SearchResult Struct Reference	39
	0.3.14.1 Detailed Description	39
	0.3.14.2 Member Data Documentation	39
	0.3.15 ELFT::ExtractionInterface::SubmissionIdentification Struct Reference	40
	0.3.15.1 Detailed Description	40
	0.3.15.2 Constructor & Destructor Documentation	40
	0.3.15.3 Member Data Documentation	41
	0.3.16 ELFT::TemplateData Struct Reference	42
	0.3.16.1 Detailed Description	42
	0.3.16.2 Member Data Documentation	42
0.4 F	Tile Documentation	43
	0.4.1 elft.h File Reference	43
	0.4.2 libelft.cpp File Reference	46
Index		47

0.1 Main Page 1

0.1 Main Page

0.1.1 Overview

This is the API that must be implemented to participate in the National Institute of Standards and Technology (NIST)'s Evaluation of Latent Friction Ridge Technology (ELFT).

0.1.2 Implementation

Two pure-virtual (abstract) classes called ELFT::ExtractionInterface and ELFT::SearchInterface have been defined. Participants must implement all methods of both classes in subclasses and submit the implementations in a shared library. The name of the library must follow the requirements outlined in the test plan and be identical to the required information returned from ELFT::ExtractionInterface::getIdentification(). NIST's testing application will link against the submitted library and instantiate instances of the implementations with their respective getImplementation() functions (ELFT::ExtractionInterface::getImplementation()).

0.1.3 Contact

Additional information regarding ELFT can be received by emailing questions to the test liaisons at elft@nist.gov.

0.1.4 License

This software was developed at NIST by employees of the Federal Government in the course of their official duties. Pursuant to title 17 Section 105 of the United States Code, this software is not subject to copyright protection and is in the public domain. NIST assumes no responsibility whatsoever for its use by other parties, and makes no guarantees, expressed or implied, about its quality, reliability, or any other characteristic.

0.2 Namespace Documentation

0.2.1 ELFT Namespace Reference

Classes

struct Candidate

Elements of a candidate list.

struct Coordinate

Pixel location in an image.

struct Correspondence

Location of identical features from two images.

struct CreateTemplateResult

Output from extracting features into a template .

• struct EFS

Collection of ANSI/NIST-ITL 1-2011 (Update: 2015) Extended Feature Set fields understood by ELFT.

class ExtractionInterface

Interface for feature extraction implemented by participant.

struct Image

Data and metadata for an image.

• struct Minutia

Friction ridge feature details.

struct ProductIdentifier

Identifying details about algorithm components for documentation.

struct ReturnStatus

Information about the result of calling an ELFT API function.

• struct RidgeQualityRegion

Region defined in a map of ridge quality/confidence.

class SearchInterface

Interface for database search implemented by participant.

struct SearchResult

The results of a searching a database.

• struct TemplateData

Information possibly stored in a template.

Enumerations

enum Impression {
 Impression::PlainContact = 0, Impression::RolledContact = 1, Impression::Latent = 4,
 Impression::LiveScanSwipe = 8,
 Impression::PlainContactlessStationary = 24, Impression::RolledContactlessStationary = 25,
 Impression::Other = 28, Impression::Unknown = 29,
 Impression::RolledContactlessMoving = 41, Impression::PlainContactlessMoving = 42 }
 Friction ridge impression types from ANSI/NIST-ITL 1-2011 (2015).

enum FrictionRidgeCaptureTechnology {

FrictionRidgeCaptureTechnology::Unknown = 0, FrictionRidgeCaptureTechnology::ScannedInkOnPaper = 2, FrictionRidgeCaptureTechnology::OpticalTIRBright = 3, FrictionRidgeCaptureTechnology::OpticalDirect = 5,

FrictionRidgeCaptureTechnology::Capacitive = 9, FrictionRidgeCaptureTechnology::Electroluminescent = 11, FrictionRidgeCaptureTechnology::LatentImpression = 18, FrictionRidgeCaptureTechnology::LatentLift = 22 }

Capture device codes from ANSI/NIST-ITL 1-2011 (2015).

• enum FrictionRidgeGeneralizedPosition {

FrictionRidgeGeneralizedPosition:: UnknownFinger = 0, FrictionRidgeGeneralizedPosition:: RightThumb = 1, FrictionRidgeGeneralizedPosition:: RightIndex = 2, FrictionRidgeGeneralizedPosition:: RightMiddle = 3, FrictionRidgeGen

FrictionRidgeGeneralizedPosition::RightRing = 4, FrictionRidgeGeneralizedPosition::RightLittle = 5, FrictionRidgeGeneralizedPosition::LeftIndex = 7,

FrictionRidgeGeneralizedPosition::LeftMiddle = 8, FrictionRidgeGeneralizedPosition::LeftRing = 9, FrictionRidgeGeneralizedPosition::LeftLittle = 10, FrictionRidgeGeneralizedPosition::RightExtraDigit = 16

Friction Ridge Generalized Position:: Left Extra Digit = 17, Friction Ridge Generalized Position:: Right Four = 13, Friction Ridge Generalized Position:: Left Four = 14, Friction Ridge Generalized Position:: Right And Left Thumbs = 15.

FrictionRidgeGeneralizedPosition:: UnknownPalm = 20, FrictionRidgeGeneralizedPosition:: RightFullPalm = 21, FrictionRidgeGeneralizedPosition:: RightWritersPalm = 22, FrictionRidgeGeneralizedPosition:: LeftFullPalm = 23, FrictionRidgeGeneralizedPosition:: LeftFullPalm = 23, FrictionRidgeGeneralizedPosition:: LeftFullPalm = 23, FrictionRidgeGeneralizedPosition:: LeftFullPalm = 24, FrictionRidgeGeneralizedPosition:: LeftFullPalm = 25, FrictionRidgeGeneralizedPosition:: LeftFullPalm = 26, FrictionRidgeGeneralizedPosition:: LeftFul

FrictionRidgeGeneralizedPosition:: LeftWritersPalm = 24, FrictionRidgeGeneralizedPosition:: RightLowerPalm = 25, FrictionRidgeGeneralizedPosition:: LeftLowerPalm = 26, Fricti

```
= 27,
      FrictionRidgeGeneralizedPosition::LeftUpperPalm = 28, FrictionRidgeGeneralizedPosition::RightPalmOther
      = 29, FrictionRidgeGeneralizedPosition::LeftPalmOther = 30, FrictionRidgeGeneralizedPosition::RightInterdigital
      Friction Ridge Generalized Position:: Right Thenar=32, Friction Ridge Generalized Position:: Right Hypothenar=32, Friction Ridge Generalized Position:: Ridge Generalized Posit
      = 33, FrictionRidgeGeneralizedPosition::LeftInterdigital = 34, FrictionRidgeGeneralizedPosition::LeftThenar
      Friction Ridge Generalized Position:: Left Hypothenar=36, Friction Ridge Generalized Position:: Right Grasp Ridge Generalized Position: Ridge 
      = 37, FrictionRidgeGeneralizedPosition::LeftGrasp = 38, FrictionRidgeGeneralizedPosition::RightCarpalDeltaArea
      = 81,
      Friction Ridge Generalized Position:: Left Carpal Delta Area = 82, Friction Ridge Generalized Position:: Right Full Palm Anna Palman Ridge Generalized Position: Right Full Palm Anna Palman Ridge Generalized Position: Right Full Palman Ridge Generalized Position: Ridge Gener
      = 83, FrictionRidgeGeneralizedPosition::LeftFullPalmAndWritersPalm = 84, FrictionRidgeGeneralizedPosition::Rig
      FrictionRidgeGeneralizedPosition::LeftWristBracelet = 86, FrictionRidgeGeneralizedPosition::UnknownFrictionRid
      = 18, FrictionRidgeGeneralizedPosition::EJIOrTip = 19}
               Friction positions codes from ANSI/NIST-ITL 1-2011 (2015).
enum ProcessingMethod {
      ProcessingMethod::Indanedione, ProcessingMethod::BlackPowder, ProcessingMethod::Other,
      ProcessingMethod::Cyanoacrylate,
                                                                                           ProcessingMethod::RUVIS, ProcessingMethod::StickysidePowder,
      ProcessingMethod::Laser,
      ProcessingMethod::Visual,
      ProcessingMethod::WhitePowder }
                EFS processing method codes from ANSI/NIST-ITL 1-2011 (2015).
     enum PatternClassification {
      PatternClassification::Arch,
                                                                                                     PatternClassification::Whorl,
                                                                                                                                                                                                        PatternClassification::RightLoop,
      PatternClassification::LeftLoop,
      PatternClassification::Amputation, PatternClassification::UnableToPrint, PatternClassification::Unclassifiable,
      PatternClassification::Scar,
      PatternClassification::DissociatedRidges }
                Classification of friction ridge structure.
• enum ValueAssessment { ValueAssessment::Value, ValueAssessment::Limited, ValueAssessment::NoValue
                EFS value assessment codes from ANSI/NIST-ITL 1-2011 (2015).
• enum Substrate {
      Substrate::Paper, Substrate::PorousOther, Substrate::Plastic, Substrate::Glass,
      Substrate::MetalPainted, Substrate::MetalUnpainted, Substrate::TapeAdhesiveSide, Substrate::NonporousOther,
      Substrate::PaperGlossy, Substrate::SemiporousOther, Substrate::Other, Substrate::Unknown }
               EFS substrate codes from ANSI/NIST-ITL 1-2011 (2015).

    enum MinutiaType { MinutiaType::RidgeEnding, MinutiaType::Bifurcation, MinutiaType::Other,

      MinutiaType::Unknown }
                Types of minutiae.
enum RidgeQuality {
      RidgeQuality::Background = 0, RidgeQuality::DebatableRidgeFlow = 1, RidgeQuality::DebatableMinutiae
      = 2, RidgeQuality::DefinitiveMinutiae = 3,
      RidgeQuality::DefinitiveRidgeEdges = 4, RidgeQuality::DefinitivePores = 5}
                Local ridge quality codes from ANSI/NIST-ITL 1-2011 (2015).

    enum TemplateType { TemplateType::Probe, TemplateType::Reference }
```

Variables

uint16_t API_MAJOR_VERSION {0}

Types of templates created by this interface.

- API major version number.
- uint16_t API_MINOR_VERSION {0}

API minor version number.

• uint16_t API_PATCH_VERSION {1}

API patch version number.

0.2.1.1 Enumeration Type Documentation

$\textbf{0.2.1.1.1} \quad Impression \quad \texttt{enum ELFT::Impression} \quad \texttt{[strong]}$

Friction ridge impression types from ANSI/NIST-ITL 1-2011 (2015).

Enumerator

PlainContact	
RolledContact	
Latent	
LiveScanSwipe	
PlainContactlessStationary	
RolledContactlessStationary	
Other	
Unknown	
RolledContactlessMoving	
PlainContactlessMoving	

Definition at line 48 of file elft.h.

$\textbf{0.2.1.1.2} \quad FrictionRidgeCaptureTechnology} \quad \texttt{enum} \quad \texttt{ELFT::FrictionRidgeCaptureTechnology} \quad \texttt{[strong]}$

Capture device codes from ANSI/NIST-ITL 1-2011 (2015).

Enumerator

Unknown	
ScannedInkOnPaper	
OpticalTIRBright	
OpticalDirect	
Capacitive	
Electroluminescent	
LatentImpression	
LatentLift	

Definition at line 63 of file elft.h.

$\textbf{0.2.1.1.3} \quad \textbf{FrictionRidgeGeneralizedPosition} \quad \textbf{enum ELFT::} \textbf{FrictionRidgeGeneralizedPosition} \quad \texttt{[strong]}$

Friction positions codes from ANSI/NIST-ITL 1-2011 (2015).

Enumerator

Entimerator	_
UnknownFinger	
RightThumb	
RightIndex	
RightMiddle	
RightRing	
RightLittle	
LeftThumb	
LeftIndex	
LeftMiddle	
LeftRing	
LeftLittle	
RightExtraDigit	
LeftExtraDigit	
RightFour	
LeftFour	
RightAndLeftThumbs	
UnknownPalm	
RightFullPalm	
RightWritersPalm	
LeftFullPalm	
LeftWritersPalm	
RightLowerPalm	
RightUpperPalm	
LeftLowerPalm	
LeftUpperPalm	
RightPalmOther	
LeftPalmOther	
RightInterdigital	
RightThenar	
RightHypothenar	
LeftInterdigital	
LeftThenar	
LeftHypothenar	
RightGrasp	
LeftGrasp	П
RightCarpalDeltaArea	
LeftCarpalDeltaArea	П
RightFullPalmAndWritersPalm	П
LeftFullPalmAndWritersPalm	Н
RightWristBracelet	П
LeftWristBracelet	Н
<u> </u>	

Enumerator

UnknownFrictionRidge	
EJIOrTip	

Definition at line 77 of file elft.h.

$\textbf{0.2.1.1.4} \quad Processing Method \quad \texttt{enum ELFT::ProcessingMethod} \quad \texttt{[strong]}$

EFS processing method codes from ANSI/NIST-ITL 1-2011 (2015).

Enumerator

Indanedione	
BlackPowder	
Other	
Cyanoacrylate	
Laser	
RUVIS	
StickysidePowder	
Visual	
WhitePowder	

Definition at line 128 of file elft.h.

0.2.1.1.5 PatternClassification enum ELFT::PatternClassification [strong]

Classification of friction ridge structure.

Note

These enumerations map to ANSI/NIST-ITL 1-2011 Update: 2015's PCT "General Class" codes from Table 44.

Enumerator

Arch	
Whorl	
RightLoop	
LeftLoop	
Amputation	
UnableToPrint	
Unclassifiable	
Scar	
DissociatedRidges	

Definition at line 148 of file elft.h.

0.2.1.1.6 ValueAssessment enum ELFT::ValueAssessment [strong]

EFS value assessment codes from ANSI/NIST-ITL 1-2011 (2015).

Enumerator

Value	
Limited	
NoValue	

Definition at line 162 of file elft.h.

0.2.1.1.7 Substrate enum **ELFT**::Substrate [strong]

EFS substrate codes from ANSI/NIST-ITL 1-2011 (2015).

Enumerator

Paper	
PorousOther	
Plastic	
Glass	
MetalPainted	
MetalUnpainted	
TapeAdhesiveSide	
NonporousOther	
PaperGlossy	
SemiporousOther	
Other	
Unknown	

Definition at line 170 of file elft.h.

0.2.1.1.8 MinutiaType enum ELFT::MinutiaType [strong]

Types of minutiae.

Enumerator

RidgeEnding

Enumerator

Bifurcation	
Other	
Unknown	

Definition at line 336 of file elft.h.

0.2.1.1.9 RidgeQuality enum ELFT::RidgeQuality [strong]

Local ridge quality codes from ANSI/NIST-ITL 1-2011 (2015).

Enumerator

Background	No ridge information.
DebatableRidgeFlow	Continuity of ridge flow is uncertain.
DebatableMinutiae	Continuity of ridge flow is certain; minutiae are debatable.
DefinitiveMinutiae	Minutiae and ridge flow are obvious and unambiguous; ridge edges are debatable.
DefinitiveRidgeEdges	Ridge edges, minutiae, and ridge flow are obvious and unambiguous; pores are either debatable or not present.
DefinitivePores	Pores and ridge edges are obvious and unambiguous.

Definition at line 411 of file elft.h.

0.2.1.1.10 TemplateType enum ELFT::TemplateType [strong]

Types of templates created by this interface.

Enumerator

Probe	Template to be used as probe in a search.
Reference	Template to be added to a reference database.

Definition at line 632 of file elft.h.

0.2.1.2 Variable Documentation

$0.2.1.2.1 \quad API_MAJOR_VERSION \quad \text{uint16_t ELFT::API_MAJOR_VERSION \{0\}}$

API major version number.

Definition at line 1225 of file elft.h.

0.2.1.2.2 API_MINOR_VERSION uint16_t ELFT::API_MINOR_VERSION {0}

API minor version number.

Definition at line 1227 of file elft.h.

0.2.1.2.3 API_PATCH_VERSION uint16_t ELFT::API_PATCH_VERSION {1}

API patch version number.

Definition at line 1229 of file elft.h.

0.3 Class Documentation

0.3.1 ELFT::Candidate Struct Reference

Elements of a candidate list.

#include <elft.h>

Public Member Functions

• Candidate (const std::string &identifier={}, const FrictionRidgeGeneralizedPosition frgp={}, const double similarity={})

Candidate constructor.

- bool operator== (const Candidate &rhs) const
- bool operator< (const Candidate &rhs) const

Public Attributes

• std::string identifier {}

Identifier of the sample in the reference database.

• FrictionRidgeGeneralizedPosition frgp {}

Most localized position in the identifier.

• double similarity {}

 $Quantification\ of\ probe's\ similarity\ to\ reference\ sample.$

0.3.1.1 Detailed Description

Elements of a candidate list.

Definition at line 579 of file elft.h.

0.3.1.2 Constructor & Destructor Documentation

Candidate constructor.

Parameters

identifier	Identifier of the sample in the reference database.
frgp	Most localized position in the identifier.
similarity	Quantification of probe's similarity to reference sample.

Definition at line 62 of file libelft.cpp.

0.3.1.3 Member Function Documentation

Definition at line 74 of file libelft.cpp.

Definition at line 83 of file libelft.cpp.

0.3.1.4 Member Data Documentation

```
0.3.1.4.1 identifier std::string ELFT::Candidate::identifier {}
```

Identifier of the sample in the reference database.

Definition at line 582 of file elft.h.

```
\textbf{0.3.1.4.2} \quad frgp \quad \textbf{FrictionRidgeGeneralizedPosition} \quad \textbf{ELFT::Candidate::frgp} \ \{\}
```

Most localized position in the identifier.

Definition at line 584 of file elft.h.

```
0.3.1.4.3 similarity double ELFT::Candidate::similarity {}
```

Quantification of probe's similarity to reference sample.

Definition at line 586 of file elft.h.

The documentation for this struct was generated from the following files:

- elft.h
- libelft.cpp

0.3.2 ELFT::ProductIdentifier::CBEFFIdentifier Struct Reference

CBEFF information registered with and assigned by IBIA.

```
#include <elft.h>
```

Public Attributes

```
• uint16_t owner {}
```

CBEFF Product Owner of the product.

• std::optional < uint16_t > algorithm {}

CBEFF Algorithm Identifier of the product.

0.3.2.1 Detailed Description

CBEFF information registered with and assigned by IBIA.

Definition at line 644 of file elft.h.

0.3.2.2 Member Data Documentation

```
0.3.2.2.1 \quad owner \quad \texttt{uint16\_t} \; \; \texttt{ELFT::ProductIdentifier::CBEFFIdentifier::owner} \; \; \{\}
```

CBEFF Product Owner of the product.

Definition at line 647 of file elft.h.

```
0.3.2.2.2 algorithm std::optional<uint16_t> ELFT::ProductIdentifier::CBEFFIdentifier::algorithm {}
```

CBEFF Algorithm Identifier of the product.

Definition at line 649 of file elft.h.

The documentation for this struct was generated from the following file:

• elft.h

0.3.3 ELFT::Coordinate Struct Reference

Pixel location in an image.

```
#include <elft.h>
```

Public Member Functions

- Coordinate (const uint32_t x={}, const uint32_t y={}) Coordinate constructor.
- bool operator== (const Coordinate &rhs) const
- bool operator< (const Coordinate &rhs) const

Public Attributes

• uint32_t x {}

X coordinate in pixels.

• uint32_t y {}

Y coordinate in pixels.

0.3.3.1 Detailed Description

Pixel location in an image.

Definition at line 304 of file elft.h.

0.3.3.2 Constructor & Destructor Documentation

Coordinate constructor.

Parameters

x	X coordinate in pixels.
y	Y coordinate in pixels.

Definition at line 91 of file libelft.cpp.

0.3.3.3 Member Function Documentation

Definition at line 101 of file libelft.cpp.

Definition at line 108 of file libelft.cpp.

0.3.3.4 Member Data Documentation

```
\textbf{0.3.3.4.1} \quad x \quad \texttt{uint32\_t ELFT::Coordinate::x \{} \\
```

X coordinate in pixels.

Definition at line 307 of file elft.h.

```
\textbf{0.3.3.4.2} \quad y \quad \text{uint32\_t ELFT::Coordinate::y } \{\}
```

Y coordinate in pixels.

Definition at line 309 of file elft.h.

The documentation for this struct was generated from the following files:

- elft.h
- libelft.cpp

0.3.4 ELFT::Correspondence Struct Reference

Location of identical features from two images.

```
#include <elft.h>
```

Public Member Functions

• Correspondence (const uint8_t referenceInputIdentifier={}, const Minutia &referenceMinutia={}, const uint8_t probeInputIdentifier={}, const Minutia &probeMinutia={})

Correspondence constructor.

Public Attributes

• uint8_t referenceInputIdentifier {}

Link to Image::identifier and/or EFS::identifier for reference.

• Minutia referenceMinutia {}

Location in the reference image of a probe image feature.

• uint8_t probeInputIdentifier {}

Link to Image::identifier and/or EFS::identifier for probe.

• Minutia probeMinutia {}

Location in the probe image of a reference image feature.

0.3.4.1 Detailed Description

Location of identical features from two images.

Definition at line 376 of file elft.h.

0.3.4.2 Constructor & Destructor Documentation

Correspondence constructor.

Parameters

referenceInputIdentifier	Link to Image::identifier and/or EFS::identifier for reference.
referenceMinutia	Location in the reference image of a probe image feature.
probeInputIdentifier	Link to Image::identifier and/or EFS::identifier for probe.
probeMinutia	Location in the probe image of a reference image feature.

Definition at line 130 of file libelft.cpp.

0.3.4.3 Member Data Documentation

$\textbf{0.3.4.3.1} \quad \textbf{referenceInputIdentifier} \quad \textbf{uint8_t} \; \; \texttt{ELFT::Correspondence::referenceInputIdentifier} \; \; \textbf{\{}\}$

Link to Image::identifier and/or EFS::identifier for reference.

Definition at line 382 of file elft.h.

0.3.4.3.2 referenceMinutia Minutia ELFT::Correspondence::referenceMinutia {}

Location in the reference image of a probe image feature.

Definition at line 384 of file elft.h.

0.3.4.3.3 probeInputIdentifier uint8_t ELFT::Correspondence::probeInputIdentifier {}

Link to Image::identifier and/or EFS::identifier for probe.

Definition at line 386 of file elft.h.

0.3.4.3.4 probeMinutia Minutia ELFT::Correspondence::probeMinutia {}

Location in the probe image of a reference image feature.

Definition at line 388 of file elft.h.

The documentation for this struct was generated from the following files:

- elft.h
- libelft.cpp

0.3.5 ELFT::CreateTemplateResult Struct Reference

Output from extracting features into a template .

#include <elft.h>

Public Attributes

• ReturnStatus status {}

Result of extracting features and creating a template.

• std::vector< std::byte > data {}

Contents of the template.

0.3.5.1 Detailed Description

Output from extracting features into a template .

Definition at line 295 of file elft.h.

0.3.5.2 Member Data Documentation

```
0.3.5.2.1 status ReturnStatus ELFT::CreateTemplateResult::status {}
```

Result of extracting features and creating a template.

Definition at line 298 of file elft.h.

```
0.3.5.2.2 data std::vector<std::byte> ELFT::CreateTemplateResult::data {}
```

Contents of the template.

Definition at line 300 of file elft.h.

The documentation for this struct was generated from the following file:

• elft.h

0.3.6 ELFT::EFS Struct Reference

Collection of ANSI/NIST-ITL 1-2011 (Update: 2015) Extended Feature Set fields understood by ELFT.

#include <elft.h>

Public Attributes

• uint8_t identifier {}

An identifier for this set of data.

• uint16_t ppi {}

Resolution of the image used to derive these features in pixels per inch.

• Impression imp {Impression::Unknown}

Impression type of the depicted region.

• FrictionRidgeCaptureTechnology frct

Capture technology that created this image.

• FrictionRidgeGeneralizedPosition frgp

Description of the depicted region.

• std::optional < int16_t > orientation {}

Degrees to rotate image upright.

• std::optional< std::vector< ProcessingMethod > > lpm {}

Methods used process the print.

std::optional < ValueAssessment > valueAssessment {}

Examiner/algorithmic value assessment for identification.

• std::optional < Substrate > lsb {}

Substrate from which the print was developed.

• std::optional < PatternClassification > pat {}

Observed pattern classification.

• std::optional < bool > plr {}

Image is known to be or may possibly be laterally reversed.

• std::optional < bool > trv {}

Part or all of image is known to be or may possibly be tonally reversed.

• std::optional< std::vector< Coordinate > > cores {}

Core locations.

• std::optional < std::vector < Coordinate > > deltas {}

Delta locations.

• std::optional< std::vector< Minutia > > minutiae {}

Locations of minutiae.

std::optional < std::vector < Coordinate > > roi {}

Closed convex polygon forming region of interest.

• std::optional < std::vector < RidgeQualityRegion > > rqm {}

Assessment of ridge quality within local areas of an image.

0.3.6.1 Detailed Description

Collection of ANSI/NIST-ITL 1-2011 (Update: 2015) Extended Feature Set fields understood by ELFT.

Note

All measurements and locations within the image SHALL be expressed in pixels, *not* units of 10 micrometers.

Definition at line 460 of file elft.h.

0.3.6.2 Member Data Documentation

```
0.3.6.2.1 identifier uint8_t ELFT::EFS::identifier {}
```

An identifier for this set of data.

Used to link EFS to Image, TemplateData, and Correspondence.

Definition at line 466 of file elft.h.

```
0.3.6.2.2 ppi uint16_t ELFT::EFS::ppi {}
```

Resolution of the image used to derive these features in pixels per inch.

Definition at line 472 of file elft.h.

```
0.3.6.2.3 imp Impression ELFT::EFS::imp {Impression::Unknown}
```

Impression type of the depicted region.

Definition at line 475 of file elft.h.

```
\textbf{0.3.6.2.4} \quad \textbf{frct} \quad \textbf{FrictionRidgeCaptureTechnology} \; \textbf{ELFT::EFS::frct}
```

```
Initial value:
```

FrictionRidgeCaptureTechnology::Unknown}

Capture technology that created this image.

Definition at line 477 of file elft.h.

```
0.3.6.2.5 frgp FrictionRidgeGeneralizedPosition ELFT::EFS::frgp
```

Initial value:

FrictionRidgeGeneralizedPosition::UnknownFrictionRidge}

Description of the depicted region.

Definition at line 480 of file elft.h.

```
0.3.6.2.6 orientation std::optional<int16_t> ELFT::EFS::orientation {}
```

Degrees to rotate image upright.

Uncertainty is assumed to be +/- 15 degrees.

Definition at line 487 of file elft.h.

```
0.3.6.2.7 \quad lpm \quad \texttt{std::optional} < \texttt{std::vector} < \texttt{ProcessingMethod} > \\ \texttt{ELFT::EFS::lpm} \ \{\}
```

Methods used process the print.

Definition at line 489 of file elft.h.

```
\textbf{0.3.6.2.8} \quad value Assessment \quad \texttt{std::optional} < \texttt{ValueAssessment} > \texttt{ELFT::EFS::valueAssessment} \ \ \{\}
```

Examiner/algorithmic value assessment for identification.

Definition at line 491 of file elft.h.

```
\textbf{0.3.6.2.9} \quad lsb \quad \textit{std::optional} < \textit{Substrate} > \; \textit{ELFT::EFS::lsb} \; \{\}
```

Substrate from which the print was developed.

Definition at line 493 of file elft.h.

```
0.3.6.2.10 pat std::optional<PatternClassification> ELFT::EFS::pat {}
```

Observed pattern classification.

Definition at line 495 of file elft.h.

```
\textbf{0.3.6.2.11} \quad plr \quad \texttt{std::optional} < \texttt{bool} > \; \texttt{ELFT::EFS::plr} \; \{\}
```

Image is known to be or may possibly be laterally reversed.

Definition at line 500 of file elft.h.

```
\textbf{0.3.6.2.12} \quad trv \quad \texttt{std::optional} < \texttt{bool} > \ \texttt{ELFT::EFS::trv} \ \{\}
```

Part or all of image is known to be or may possibly be tonally reversed.

Definition at line 505 of file elft.h.

```
0.3.6.2.13 cores std::optional<std::vector<Coordinate> > ELFT::EFS::cores {}
```

Core locations.

Coordinate are relative to the bounding rectangle created by roi, if supplied. Otherwise, they are relative to the source image.

Definition at line 515 of file elft.h.

```
0.3.6.2.14 deltas std::optional<std::vector<Coordinate> > ELFT::EFS::deltas {}
```

Delta locations.

Coordinate are relative to the bounding rectangle created by roi, if supplied. Otherwise, they are relative to the source image.

Definition at line 524 of file elft.h.

```
0.3.6.2.15 minutiae std::vector<Minutia> > ELFT::EFS::minutiae {}
```

Locations of minutiae.

Coordinate are relative to the bounding rectangle created by roi, if supplied. Otherwise, they are relative to the source image.

Note

NIST **strongly** discourages more than one Minutia at equivalent Coordinate. This can result in ambiguous Correspondence.

Definition at line 538 of file elft.h.

```
\textbf{0.3.6.2.16} \quad \textbf{roi} \quad \texttt{std::optional} < \texttt{std::vector} < \texttt{Coordinate} > \\ \texttt{ELFT::EFS::roi} \ \{\}
```

Closed convex polygon forming region of interest.

Definition at line 540 of file elft.h.

```
0.3.6.2.17 rqm std::optional<std::vector<RidgeQualityRegion> > ELFT::EFS::rqm {}
```

Assessment of ridge quality within local areas of an image.

Note

If populated, regions not explicitly defined will default to RidgeQuality::Background.

Definition at line 549 of file elft.h.

The documentation for this struct was generated from the following file:

• elft.h

0.3.7 ELFT::ExtractionInterface Class Reference

Interface for feature extraction implemented by participant.

```
#include <elft.h>
```

Classes

struct SubmissionIdentification

Identifying information about this submission that will be included in reports.

Public Member Functions

- virtual SubmissionIdentification getIdentification () const =0
 - Obtain identification and version information for the extraction portion of this submission.
- virtual CreateTemplateResult createTemplate (const TemplateType templateType, const std::string &identifier, const std::vector< std::tuple< std::optional< Image >, std::optional< EFS >>> &samples) const =0

Extract features from one or more images and encode them into a template.

• virtual std::optional< std::vector< TemplateData > > extractTemplateData (const TemplateType templateType, const CreateTemplateResult &templateResult) const =0

Extract information contained within a template.

virtual CreateTemplateResult mergeTemplates (const std::vector < std::vector < std::byte >> &templates) const =0

Merge multiple templates into a single template.

virtual ReturnStatus createReferenceDatabase (const std::vector< std::vector< std::byte >> &referenceTemplates, const std::filesystem::path &databaseDirectory, const uint64_t maxSize)
 const =0

Create a reference database on the filesystem.

- ExtractionInterface ()
- virtual ~ExtractionInterface ()

Static Public Member Functions

• static std::shared_ptr< ExtractionInterface > getImplementation (const std::filesystem::path &configurationDirectory)

Obtain a managed pointer to an object implementing ExtractionInterface.

0.3.7.1 Detailed Description

Interface for feature extraction implemented by participant.

Definition at line 662 of file elft.h.

0.3.7.2 Constructor & Destructor Documentation

```
0.3.7.2.1 ExtractionInterface() ELFT::ExtractionInterface::ExtractionInterface() [default]
```

```
0.3.7.2.2 ~ExtractionInterface() ELFT::ExtractionInterface::~ExtractionInterface() [virtual], [default]
```

0.3.7.3 Member Function Documentation

```
\textbf{0.3.7.3.1} \quad \textbf{getIdentification()} \quad \textbf{virtual SubmissionIdentification ELFT::} \textbf{ExtractionInterface::} \textbf{getIdentification()} \\ \textbf{o.sonst [pure virtual]}
```

Obtain identification and version information for the extraction portion of this submission.

Returns

SubmissionIdentification populated with information used to identify the feature extraction algorithms in reports.

Note

This method shall return instantly.

Extract features from one or more images and encode them into a template.

Parameters

templateType	Where this template will be used in the future.
identifier	Unique identifier used to identify the returned template in future <i>search</i> operations (e.g., Candidate::identifier).
samples	One or more biometric samples to be considered and encoded into a template.

Returns

A single CreateTemplateResult, which contains information about the result of the operation and a single template.

Note

This method must return in $\leq N * M$ seconds for each element of samples, on average, as measured on a fixed subset of data, where

- N
- 20.0 for latent images
- 5.0 for exemplar images
- 2.5 for feature sets
- M
- 1 for single fingers
- 2 for two-finger simultaneous captures
- 4 for four-finger simultaneous captures
- 8 for upper palm, lower palm, and all other palm/joint regions *except* full palm
- 16 for full palm

If samples contained RightThumb, LeftFour, and EJIOrTip, the time requirement would be $\leq ((5 * 1) + (5 * 4) + (5 * 8))$ seconds.

The value of the returned CreateTemplateResult::data will only be recorded if CreateTemplateResult's ReturnStatus::result is ReturnStatus::Result::Success. On ReturnStatus::Result:: Failure, subsequent searches will automatically increase false negative identification rate.

Extract information contained within a template.

Parameters

templateType templateType passed to createTemplate().		templateType passed to createTemplate().
	templateResult	Object returned from createTemplate() or mergeTemplates().

Returns

One or more TemplateData describing the contents of CreateTemplateResult::data from template Result. If CreateTemplateResult::data contains information separated by position (e.g., when provided a multi-position image) or multiple views of the same image (e.g., a compact and verbose template), there can be multiple TemplateData returned.

Note

You must implement this method to compile, but providing the requested information is optional. If provided, information may help in debugging as well as inform future NIST analysis.

You should not return information that was provided in createTemplate(). For instance, if Minutia was provided, EFS::minutiae should be left std::nullopt. However, if you discovered different Minutia, they should be returned.

The ReturnStatus member of CreateTemplateResult is not guaranteed to be populated with ReturnStatus::message and should not be consulted.

```
0.3.7.3.4 mergeTemplates() virtual CreateTemplateResult ELFT::ExtractionInterface::mergeTemplates ( const std::vector< std::byte >> & templates ) const [pure virtual]
```

Merge multiple templates into a single template.

This method is necessary because more than one set of samples for a given identifier may have been provided to createTemplate().

Parameters

templates	One or more template returned from createTemplate() that should be merged into a single
·	template.

Returns

A single CreateTemplateResult that can accurately represent identifier in future operations.

Note

The merged template does not need to include all information. For instance, if two reference templates are provided, each derived from an identification flat capture, and an internal heuristic deems one set of samples to be of significantly lower quality, a reasonable result would be for this method to return the template from the higher quality set of samples without modification.

The contents of templates may be the result of previous calls to mergeTemplates().

Identifiers and template types should be stored within the template data itself and so are not provided as input.

This method shall return in $\leq 10 * templates.size()$ milliseconds.

The value of the returned CreateTemplateResult::data will only be recorded if CreateTemplateResult's ReturnStatus::Result::Success. On ReturnStatus::Result:: Failure, subsequent searches will automatically increase false negative identification rate.

See also

SearchInterface::insert.

Create a reference database on the filesystem.

Parameters

referenceTemplates	One or more templates returned from createTemplate() or mergeTemplates() with a templateType of TemplateType::Reference.
databaseDirectory	Entry to a read/write directory where the reference database shall be written.
maxSize	The maximum number of bytes of storage available to write.

Returns

Information about the result of executing the method.

Note

This method may use more than one thread.

This method must return in $\leq 10 * referenceTemplates.size()$ milliseconds.

Obtain a managed pointer to an object implementing ExtractionInterface.

Parameters

configurationDirectory	Read-only directory populated with configuration files provided in valida-
	tion.

Returns

Shared pointer to an instance of ExtractionInterface containing the participant's code to perform extraction operations.

Note

A possible implementation might be: return (std::make_shared<ExtractionImplementation>(configurationDirectory));

This method shall return in <= 5 seconds.

The documentation for this class was generated from the following files:

- elft.h
- libelft.cpp

0.3.8 ELFT::Image Struct Reference

Data and metadata for an image.

```
#include <elft.h>
```

Public Member Functions

- Image ()
- Image (const uint8_t identifier, const uint16_t width, const uint16_t height, const uint16_t ppi, const uint8_t bpc, const uint8_t bpp, const std::vector < std::byte > &pixels)

Image constructor.

Public Attributes

```
• uint8_t identifier {}
```

An identifier for this image.

• uint16_t width {}

Width of the image.

• uint16_t height {}

Height of the image.

• uint16_t ppi {}

 $Resolution\ of\ the\ image\ in\ pixels\ per\ inch.$

• uint8_t bpc {}

Number of bits used by each color component (8 or 16).

• uint8_t bpp {}

Number of bits comprising a single pixel.

• std::vector< std::byte > pixels {}

Raw pixel data of image.

0.3.8.1 Detailed Description

Data and metadata for an image.

Definition at line 219 of file elft.h.

0.3.8.2 Constructor & Destructor Documentation

Image constructor.

Parameters

identifier	An identifier for this image. Used to link Image to TemplateData and Correspondence.
width	Width of the image in pixels.
height	Height of the image in pixels.
ppi	Resolution of the image in pixels per inch.
bpc	Number of bits used by each color component (8 or 16).
bpp	Number of bits comprising a single pixel.
pixels	width * height * (bpp / bpc) bytes of image data, with pixels.front() representing the first byte of the top-left pixel, and pixels.back() representing the last byte of bottom-right pixel. It is decompressed little endian image data, canonically coded as defined in ISO/IEC 19794-4:2005, section 6.2.

Note

Number of color components is bpp / bpc and shall be either 1 (grayscale) or 3 (RGB).

Definition at line 35 of file libelft.cpp.

0.3.8.3 Member Data Documentation

```
0.3.8.3.1 identifier uint8_t ELFT::Image::identifier {}
```

An identifier for this image.

Used to link Image to EFS, TemplateData, and Correspondence.

Definition at line 265 of file elft.h.

```
0.3.8.3.2 width uint16_t ELFT::Image::width {}
```

Width of the image.

Definition at line 267 of file elft.h.

```
0.3.8.3.3 height uint16_t ELFT::Image::height {}
```

Height of the image.

Definition at line 269 of file elft.h.

```
0.3.8.3.4 ppi uint16_t ELFT::Image::ppi {}
```

Resolution of the image in pixels per inch.

Definition at line 271 of file elft.h.

```
0.3.8.3.5 bpc uint8_t ELFT::Image::bpc {}
```

Number of bits used by each color component (8 or 16).

Definition at line 273 of file elft.h.

```
0.3.8.3.6 bpp uint8_t ELFT::Image::bpp {}
```

Number of bits comprising a single pixel.

Definition at line 275 of file elft.h.

```
0.3.8.3.7 pixels std::vector<std::byte> ELFT::Image::pixels {}
```

Raw pixel data of image.

width * height * (bpp / bpc) bytes of image data, with pixels.front() representing the first byte of the top-left pixel, and pixels.back() representing the last byte of bottom-right pixel. It is decompressed little endian image data, canonically coded as defined in ISO/IEC 19794-4:2005,

Note

To pass pixels to a C-style array, invoke pixel's data() method (pixels.data()).

Definition at line 291 of file elft.h.

The documentation for this struct was generated from the following files:

- elft.h
- libelft.cpp

0.3.9 ELFT::Minutia Struct Reference

Friction ridge feature details.

```
#include <elft.h>
```

Public Member Functions

• Minutia (const Coordinate &coordinate={}, const uint16_t theta={}, const MinutiaType type=MinutiaType::Unknown)

Minutia constructor.

Public Attributes

• Coordinate coordinate {}

Location of the feature.

• uint16_t theta {}

Ridge direction of the feature, in degrees [0,359], following conventions from ANSI/NIST-ITL 1-2011 (2015) Field 9.331.

• MinutiaType type {MinutiaType::Unknown}

Type of feature.

0.3.9.1 Detailed Description

Friction ridge feature details.

Definition at line 345 of file elft.h.

0.3.9.2 Constructor & Destructor Documentation

Minutia constructor.

Parameters

coordinate	Location of the feature.
theta	Ridge direction of the feature, in degrees [0,359], following conventions from ANSI/N ← IST-ITL 1-2011 (2015) Field 9.331.
type	Type of feature.

Definition at line 143 of file libelft.cpp.

0.3.9.3 Member Data Documentation

```
0.3.9.3.1 coordinate Coordinate ELFT::Minutia::coordinate {}
```

Location of the feature.

Definition at line 348 of file elft.h.

```
\textbf{0.3.9.3.2} \quad theta \quad \texttt{uint16\_t ELFT::Minutia::theta \{}\}
```

Ridge direction of the feature, in degrees [0,359], following conventions from ANSI/NIST-ITL 1-2011 (2015) Field 9.331.

Definition at line 353 of file elft.h.

```
0.3.9.3.3 type MinutiaType ELFT::Minutia::type {MinutiaType::Unknown}
```

Type of feature.

Definition at line 355 of file elft.h.

The documentation for this struct was generated from the following files:

- elft.h
- libelft.cpp

0.3.10 ELFT::ProductIdentifier Struct Reference

Identifying details about algorithm components for documentation.

```
#include <elft.h>
```

Classes

• struct CBEFFIdentifier

CBEFF information registered with and assigned by IBIA.

Public Attributes

- std::optional < std::string > marketing {}

 Non-infringing marketing name of the product.
- std::optional < CBEFFIdentifier > cbeff {} CBEFF information about the product.

0.3.10.1 Detailed Description

Identifying details about algorithm components for documentation.

Definition at line 641 of file elft.h.

0.3.10.2 Member Data Documentation

```
0.3.10.2.1 marketing std::optional<std::string> ELFT::ProductIdentifier::marketing {}
```

Non-infringing marketing name of the product.

Case sensitive. Must match the regular expression [[:graph:]]*.

Definition at line 656 of file elft.h.

```
\textbf{0.3.10.2.2} \quad cbeff \quad \texttt{std::optional} < \texttt{CBEFFIdentifier} > \texttt{ELFT::ProductIdentifier::cbeff} \ \{\}
```

CBEFF information about the product.

Definition at line 658 of file elft.h.

The documentation for this struct was generated from the following file:

• elft.h

0.3.11 ELFT::ReturnStatus Struct Reference

Information about the result of calling an ELFT API function.

```
#include <elft.h>
```

Public Types

• enum Result { Result::Success = 0, Result::Failure } Possible outcomes when performing operations.

Public Member Functions

• operator bool () const noexcept

Public Attributes

• Result result {}

The result of the operation.

• std::optional < std::string > message {}

Information about the result.

0.3.11.1 Detailed Description

Information about the result of calling an ELFT API function.

Definition at line 190 of file elft.h.

0.3.11.2 Member Enumeration Documentation

$\textbf{0.3.11.2.1} \quad Result \quad \texttt{enum ELFT::ReturnStatus::Result} \quad \texttt{[strong]}$

Possible outcomes when performing operations.

Enumerator

Success	Successfully performed operation.
Failure	Failed to perform operation.

Definition at line 193 of file elft.h.

0.3.11.3 Member Function Documentation

```
0.3.11.3.1 operator bool() ELFT::ReturnStatus::operator bool ( ) const [explicit], [noexcept]
```

Returns

true if result is Result::Success, false otherwise.

Definition at line 54 of file libelft.cpp.

0.3.11.4 Member Data Documentation

```
0.3.11.4.1 result Result ELFT::ReturnStatus::result {}
```

The result of the operation.

Definition at line 202 of file elft.h.

```
0.3.11.4.2 message std::optional<std::string> ELFT::ReturnStatus::message {}
```

Information about the result.

Must match the regular expression [[:graph:]]*.

Definition at line 207 of file elft.h.

The documentation for this struct was generated from the following files:

- elft.h
- libelft.cpp

0.3.12 ELFT::RidgeQualityRegion Struct Reference

Region defined in a map of ridge quality/confidence.

```
#include <elft.h>
```

Public Attributes

- std::vector < Coordinate > region {}
 - Closed convex polygon whose contents is quality.
- RidgeQuality quality {RidgeQuality::Background}

Clarity of ridge features enclosed within region.

0.3.12.1 Detailed Description

Region defined in a map of ridge quality/confidence.

Definition at line 437 of file elft.h.

0.3.12.2 Member Data Documentation

```
0.3.12.2.1 region std::vector<Coordinate> ELFT::RidgeQualityRegion::region {}
```

Closed convex polygon whose contents is quality.

Coordinate are relative to the bounding rectangle created by EFS::roi, if supplied. Otherwise, they are relative to the the source image.

Definition at line 447 of file elft.h.

```
0.3.12.2.2 quality RidgeQuality ELFT::RidgeQualityRegion::quality {RidgeQuality::Background}
```

Clarity of ridge features enclosed within region.

Definition at line 449 of file elft.h.

The documentation for this struct was generated from the following file:

elft.h

0.3.13 ELFT::SearchInterface Class Reference

Interface for database search implemented by participant.

#include <elft.h>

Public Member Functions

- virtual std::optional < ProductIdentifier > getIdentification () const =0
 Obtain identification and version information for the search portion of this submission.
- virtual std::tuple < ReturnStatus, bool > exists (const std::string &identifier) const =0

 Determine if an identifier is in the reference database.
- virtual ReturnStatus insert (const std::string &identifier, const std::vector < std::byte > &reference ← Template)=0

Insert or update an identifier in a loaded reference database.

- virtual ReturnStatus remove (const std::string &identifier)=0
 - Remove an identifier from a loaded reference database.
- virtual SearchResult search (const std::vector< std::byte > &probeTemplate, const uint16_t max← Candidates) const =0

 ${\it Search \ the \ reference \ database \ for \ the \ samples \ represented \ in \ {\tt probeTemplate}.}$

- virtual std::optional< std::vector< std::vector< Correspondence > > extractCorrespondence (const std::vector< std::byte > &probeTemplate, const SearchResult &searchResult) const =0
 - Extract pairs of corresponding minutiae between probe template and reference template.
- SearchInterface ()
- virtual ~SearchInterface ()

Static Public Member Functions

• static std::shared_ptr< SearchInterface > getImplementation (const std::filesystem::path &configurationDirectory, const std::filesystem::path &databaseDirectory)

Obtain a managed pointer to an object implementing SearchInterface.

0.3.13.1 Detailed Description

Interface for database search implemented by participant.

Definition at line 972 of file elft.h.

0.3.13.2 Constructor & Destructor Documentation

```
0.3.13.2.1 SearchInterface() ELFT::SearchInterface::SearchInterface() [default]
```

```
0.3.13.2.2 ~SearchInterface() ELFT::SearchInterface::~SearchInterface ( ) [virtual], [default]
```

0.3.13.3 Member Function Documentation

```
\textbf{0.3.13.3.1} \quad \textbf{getIdentification()} \quad \textbf{virtual} \qquad \textbf{std::optional < ProductIdentifier>} \qquad \textbf{ELFT::SearchInterface::get} \\ \textbf{Identification ( ) const} \quad \textbf{[pure virtual]}
```

Obtain identification and version information for the search portion of this submission.

Returns

ProductIdentifier populated with information used to identify the search algorithm in reports.

Note

This method shall return instantly.

```
0.3.13.3.2 exists() virtual std::tuple<ReturnStatus, bool> ELFT::SearchInterface::exists ( const std::string & identifier ) const [pure virtual]
```

Determine if an identifier is in the reference database.

Parameters

identifier	Identifier to check.

Returns

A tuple whose first member is the result of executing the operation, and whose second member is true if identifier is represented in the reference database, and false otherwise.

Note

This method must return in ≤ 5 seconds.

This method need not be threadsafe. It may use more than one thread.

Insert or update an identifier in a loaded reference database.

This method should limit the amount of I/O and processing necessary, as indicated by the runtime limitation noted below.

Parameters

identifier	Identifier to add or update.
	A template returned from ExtractionInterface::createTemplate() with a template ← Type of TemplateType::Reference.

Returns

Information about the result of executing the method.

Note

If identifier already exists in the enrollment database, this method should "merge" data that already exists in the database with referenceTemplate before replacing the entry in the database.

This method must return in ≤ 5 seconds.

This method need not be threadsafe. It may use more than one thread.

Remove an identifier from a loaded reference database.

This method should limit the amount of I/O and processing necessary, as indicated by the runtime limitation noted below.

Parameters

identifier	Identifier to remove.
------------	-----------------------

Returns

Information about the result of executing the method.

Note

This method must return in <= 5 seconds.

This method need not be threadsafe. It may use more than one thread.

Search the reference database for the samples represented in probeTemplate.

Parameters

probeTemplate	Object returned from createTemplate() or mergeTemplates() with templateType of TemplateType::Probe.
maxCandidates	The maximum number of Candidate to return.

Returns

A SearchResult object containing information on if this task was able to be completed and a list of less than or equal to maxCandidates Candidate.

Note

SearchResult.candidateList will be sorted by descending similarity upon return from this method using std::stable_sort().

If provided a probe template that contains comes from multiple regions, Candidate.frgp will be ignored.

Candidate.frgp shall be the most localized region where the match was made to be considered as correct as possible. See the test plan for more information.

This method must return in $\leq 10 *$ number of database identifiers milliseconds, on average, as measured on a fixed subset of data.

Extract pairs of corresponding minutiae between probe template and reference template.

Parameters

probe Template	Probe template sent to searchReferences().
searchResult	Object returned from searchReferences().

Returns

A vector the length of searchResult.candidateList.size(), where each entry is the collection of corresponding minutiae points between probeTemplate and the reference template of the Candidate at the same position as searchResult's SearchResult.candidateList.

Note

ELFT::Minutia must align with minutiae returned from ExtractionInterface::extractTemplateData() for the given identifier + position pair.

You must implement this method to compile, but providing the requested information is optional. If provided, information may help in debugging, as well as informing future NIST analysis.

searchResult is **not guaranteed** to be the identical object returned from search(). Specifically, ordering of searchResult.candidateList may have changed (e.g., sorted by descending similarity) and the ReturnStatus member is not guaranteed to populated with ReturnStatus::message.

Obtain a managed pointer to an object implementing SearchInterface.

Parameters

configurationDirectory	Read-only directory populated with configuration files provided in validation.						
databaseDirectory	Read-only ExtractionIn	directory terface::create	populated ReferenceDatab	with pase().	files	written	in

Returns

Shared pointer to an instance of SearchInterface containing the participant's code to perform search operations.

Note

A possible implementation might be: return (std::make_shared<SearchImplementation>(configurationDirectory, databaseDirectory);

This method shall return in <= 5 seconds.

The documentation for this class was generated from the following files:

- elft.h
- libelft.cpp

0.3.14 ELFT::SearchResult Struct Reference

The results of a searching a database.

```
#include <elft.h>
```

Public Attributes

• ReturnStatus status {}

Status of searching reference database and assembling candidate list.

• bool decision {}

Best guess on if candidateList contains an identification.

• std::vector < Candidate > candidateList {}

List of Candidate most similar to the probe.

0.3.14.1 Detailed Description

The results of a searching a database.

Definition at line 616 of file elft.h.

0.3.14.2 Member Data Documentation

```
\textbf{0.3.14.2.1} \quad \textbf{status} \quad \textbf{ReturnStatus} \; \; \textbf{ELFT::SearchResult::status} \; \{\}
```

Status of searching reference database and assembling candidate list.

Definition at line 622 of file elft.h.

```
0.3.14.2.2 decision bool ELFT::SearchResult::decision {}
```

Best guess on if candidateList contains an identification.

Definition at line 626 of file elft.h.

```
0.3.14.2.3 candidateList std::vector<Candidate> ELFT::SearchResult::candidateList {}
```

List of Candidate most similar to the probe.

Definition at line 628 of file elft.h.

The documentation for this struct was generated from the following file:

• elft.h

0.3.15 ELFT::ExtractionInterface::SubmissionIdentification Struct Reference

Identifying information about this submission that will be included in reports.

```
#include <elft.h>
```

Public Member Functions

- SubmissionIdentification ()
- SubmissionIdentification (const uint16_t versionNumber, const std::string &libraryIdentifier, const std::optional < ProductIdentifier > &exemplarAlgorithmIdentifier={}, const std::optional < ProductIdentifier > &latentAlgorithmIdentifier={})

SubmissionIdentification constructor.

Public Attributes

• uint16_t versionNumber {}

Version number of this submission.

• std::string libraryIdentifier {}

Non-infringing identifier of this submission.

- std::optional < ProductIdentifier > exemplarAlgorithmIdentifier {}
 - Information about the exemplar feature extraction algorithm in this submission.
- std::optional < ProductIdentifier > latentAlgorithmIdentifier {}

Information about the latent feature extraction algorithm in this submission.

0.3.15.1 Detailed Description

Identifying information about this submission that will be included in reports.

Definition at line 669 of file elft.h.

0.3.15.2 Constructor & Destructor Documentation

```
0.3.15.2.1 SubmissionIdentification() [1/2] ELFT::ExtractionInterface::SubmissionIdentification::Submission← Identification ( ) [default]
```

SubmissionIdentification constructor.

Parameters

versionNumber	Version number of this submission. Required to be unique for each new submission.
libraryIdentifier	Non-infringing identifier of this submission. Should be the same for all submissions. Case sensitive. Must match the regular expression [:alnum:]+.
exemplarAlgorithmIdentifier	Information about the exemplar feature extraction algorithm in this submission.
latentAlgorithmIdentifier	Information about the latent feature extraction algorithm in this submission.

Note

The name of the core library submitted for evaluation shall be "libelft_libraryIdentifier>_← <versionNumber (capital hex)>.so". Refer to the test plan for more information.

Definition at line 18 of file libelft.cpp.

0.3.15.3 Member Data Documentation

 $\textbf{0.3.15.3.1} \quad version Number \quad \textit{uint16_t} \; \; \textit{ELFT::ExtractionInterface::SubmissionIdentification::versionNumber} \; \; \{\} \\$

Version number of this submission.

Required to be unique for each new submission.

Definition at line 710 of file elft.h.

Non-infringing identifier of this submission.

Should be the same for all submissions from an organization. Case sensitive. Must match the regular expression [:alnum:]+.

Definition at line 717 of file elft.h.

0.3.15.3.3 exemplarAlgorithmIdentifier std::optional<ProductIdentifier> ELFT::ExtractionInterface::← SubmissionIdentification::exemplarAlgorithmIdentifier {}

Information about the exemplar feature extraction algorithm in this submission.

Definition at line 724 of file elft.h.

```
0.3.15.3.4 latentAlgorithmIdentifier std::optional<ProductIdentifier> ELFT::ExtractionInterface::←
SubmissionIdentification::latentAlgorithmIdentifier {}
```

Information about the latent feature extraction algorithm in this submission.

Definition at line 730 of file elft.h.

The documentation for this struct was generated from the following files:

- elft.h
- libelft.cpp

0.3.16 ELFT::TemplateData Struct Reference

Information possibly stored in a template.

```
#include <elft.h>
```

Public Attributes

• std::string candidateIdentifier {}

Candidate identifier provided in ExtractionInterface::createTemplate().

• uint8_t inputIdentifier {}

Link to Image::identifier and/or EFS::identifier.

• std::optional < EFS > efs {}

Extended feature set data.

• std::optional < uint8_t > imageQuality {}

Quality of the image, [0-100].

0.3.16.1 Detailed Description

Information possibly stored in a template.

Note

If provided a multi-position image and applicable to the feature extraction algorithm, roi should be populated with segmentation coordinates and frgp should be set for each position.

Definition at line 560 of file elft.h.

0.3.16.2 Member Data Documentation

0.4 File Documentation 43

```
\textbf{0.3.16.2.1} \quad candidate I dentifier \quad \texttt{std::string} \ \ \texttt{ELFT::TemplateData::candidateIdentifier} \quad \texttt{\{}\}
```

Candidate identifier provided in ExtractionInterface::createTemplate().

Definition at line 566 of file elft.h.

```
0.3.16.2.2 inputIdentifier uint8_t ELFT::TemplateData::inputIdentifier {}
```

Link to Image::identifier and/or EFS::identifier.

Definition at line 569 of file elft.h.

```
0.3.16.2.3 \quad efs \quad \texttt{std::optional} < \texttt{EFS} > \texttt{ELFT::TemplateData::efs} \ \{\}
```

Extended feature set data.

Definition at line 572 of file elft.h.

```
0.3.16.2.4 imageQuality std::optional<uint8_t> ELFT::TemplateData::imageQuality {}
```

Quality of the image, [0-100].

Definition at line 575 of file elft.h.

The documentation for this struct was generated from the following file:

• elft.h

0.4 File Documentation

0.4.1 elft.h File Reference

```
#include <cstddef>
#include <cstdint>
#include <filesystem>
#include <memory>
#include <optional>
#include <string>
#include <tuple>
#include <vector>
```

Classes

• struct ELFT::ReturnStatus

Information about the result of calling an ELFT API function.

• struct ELFT::Image

Data and metadata for an image.

• struct ELFT::CreateTemplateResult

Output from extracting features into a template .

• struct ELFT::Coordinate

Pixel location in an image.

• struct ELFT::Minutia

Friction ridge feature details.

• struct ELFT::Correspondence

Location of identical features from two images.

struct ELFT::RidgeQualityRegion

Region defined in a map of ridge quality/confidence.

struct ELFT::EFS

Collection of ANSI/NIST-ITL 1-2011 (Update: 2015) Extended Feature Set fields understood by ELFT.

• struct ELFT::TemplateData

Information possibly stored in a template.

• struct ELFT::Candidate

Elements of a candidate list.

struct ELFT::SearchResult

The results of a searching a database.

struct ELFT::ProductIdentifier

Identifying details about algorithm components for documentation.

• struct ELFT::ProductIdentifier::CBEFFIdentifier

CBEFF information registered with and assigned by IBIA.

• class ELFT::ExtractionInterface

Interface for feature extraction implemented by participant.

• struct ELFT::ExtractionInterface::SubmissionIdentification

Identifying information about this submission that will be included in reports.

• class ELFT::SearchInterface

Interface for database search implemented by participant.

Namespaces

ELFT

Enumerations

enum ELFT::Impression {
 ELFT::Impression::PlainContact = 0, ELFT::Impression::RolledContact = 1, ELFT::Impression::Latent
 = 4, ELFT::Impression::LiveScanSwipe = 8,
 ELFT::Impression::PlainContactlessStationary = 24, ELFT::Impression::RolledContactlessStationary
 = 25, ELFT::Impression::Other = 28, ELFT::Impression::Unknown = 29,

ELFT::Impression::RolledContactlessMoving = 41, ELFT::Impression::PlainContactlessMoving = 42}

Friction ridge impression types from ANSI/NIST-ITL 1-2011 (2015).

0.4 File Documentation 45

```
    enum ELFT::FrictionRidgeCaptureTechnology {

           ELFT::FrictionRidgeCaptureTechnology::Unknown=0, ELFT::FrictionRidgeCaptureTechnology::ScannedInkOnParticles.
           = 2, ELFT::FrictionRidgeCaptureTechnology::OpticalTIRBright = 3, ELFT::FrictionRidgeCaptureTechnology::OpticalTIRBright
           ELFT::FrictionRidgeCaptureTechnology::Capacitive = 9, ELFT::FrictionRidgeCaptureTechnology::Electroluminesce
           = 11, ELFT::FrictionRidgeCaptureTechnology::LatentImpression = 18, ELFT::FrictionRidgeCaptureT
                             Capture device codes from ANSI/NIST-ITL 1-2011 (2015).

    enum ELFT::FrictionRidgeGeneralizedPosition {

            ELFT:: Friction Ridge Generalized Position:: Unknown Finger = 0, ELFT:: Friction Ridge Generalized Position:: Right Thursday Friction Ridge Generalized Position:: Ridge Friction Ridge Generalized Position Ridge Friction Ridge Fric
           = 1, ELFT::FrictionRidgeGeneralizedPosition::RightIndex = 2, ELFT::FrictionRidgeGeneralizedPosition::RightMiddl
           = 3,
           ELFT:: FrictionRidgeGeneralizedPosition:: RightRing = 4, ELFT:: FrictionRidgeGeneralizedPosition:: RightLittle (Control of the Control of t
           = 5, ELFT::FrictionRidgeGeneralizedPosition::LeftIndex
           ELFT::FrictionRidgeGeneralizedPosition::LeftMiddle = 8, ELFT::FrictionRidgeGeneralizedPosition::LeftRing
           = 9, ELFT::FrictionRidgeGeneralizedPosition::LeftLittle = 10, ELFT::FrictionRidgeGeneralizedPosition::RightExtraD
           ELFT:: FrictionRidgeGeneralizedPosition:: LeftExtraDigit = 17, ELFT:: FrictionRidgeGeneralizedPosition:: RightFour Properties of the pro
           = 13, ELFT::FrictionRidgeGeneralizedPosition::LeftFour = 14, ELFT::FrictionRidgeGeneralizedPosition::RightAndLo
           ELFT::FrictionRidgeGeneralizedPosition::UnknownPalm = 20, ELFT::FrictionRidgeGeneralizedPosition::RightFullF
           = 21, ELFT::FrictionRidgeGeneralizedPosition::RightWritersPalm = 22, ELFT::FrictionRidgeGeneralizedPosition::Le
           ELFT::FrictionRidgeGeneralizedPosition::LeftWritersPalm = 24, ELFT::FrictionRidgeGeneralizedPosition::RightLov
           = 25, ELFT::FrictionRidgeGeneralizedPosition::RightUpperPalm = 26, ELFT::FrictionRidgeGeneralizedPosition::Lef
           ELFT:: FrictionRidgeGeneralizedPosition:: LeftUpperPalm = 28, ELFT:: FrictionRidgeGeneralizedPosition:: RightPalm = 28, ELFT:: RightPal
           = 29, ELFT::FrictionRidgeGeneralizedPosition::LeftPalmOther = 30, ELFT::FrictionRidgeGeneralizedPosition::Right
           ELFT::FrictionRidgeGeneralizedPosition::RightThenar=32, ELFT::FrictionRidgeGeneralizedPosition::RightHypothenar=32, ELFT::Fr
           = 33, ELFT::FrictionRidgeGeneralizedPosition::LeftInterdigital = 34, ELFT::LeftInterdigital = 34, ELFT::LeftInterd
           ELFT::FrictionRidgeGeneralizedPosition::LeftHypothenar = 36, ELFT::FrictionRidgeGeneralizedPosition::RightGra
           = 37, ELFT::FrictionRidgeGeneralizedPosition::LeftGrasp = 38, ELFT::FrictionRidgeGeneralizedPosition::RightCarp
           ELFT::FrictionRidgeGeneralizedPosition::LeftCarpalDeltaArea = 82, ELFT::FrictionRidgeGeneralizedPosition::RightCarpalDeltaArea = 82, ELFT::RightCarpalDeltaArea = 82, ELFT::Right
           = 83, ELFT::FrictionRidgeGeneralizedPosition::LeftFullPalmAndWritersPalm = 84, ELFT::FrictionRidgeGeneralizedPosition:
           ELFT:: FrictionRidgeGeneralizedPosition:: LeftWristBracelet = 86, ELFT:: FrictionRidgeGeneralizedPosition:: Unknown and Unkn
           = 18, ELFT::FrictionRidgeGeneralizedPosition::EJIOrTip = 19}
                            Friction positions codes from ANSI/NIST-ITL 1-2011 (2015).
         enum ELFT::ProcessingMethod {
           ELFT::ProcessingMethod::Indanedione, ELFT::ProcessingMethod::BlackPowder, ELFT::ProcessingMethod::Other,
           ELFT::ProcessingMethod::Cyanoacrylate,
           ELFT::ProcessingMethod::Laser, ELFT::ProcessingMethod::RUVIS, ELFT::ProcessingMethod::StickysidePowder,
           ELFT::ProcessingMethod::Visual,
           ELFT::ProcessingMethod::WhitePowder }
                             EFS processing method codes from ANSI/NIST-ITL 1-2011 (2015).
         enum ELFT::PatternClassification {
           ELFT::PatternClassification::Arch, ELFT::PatternClassification::Whorl, ELFT::PatternClassification::RightLoop,
           ELFT::PatternClassification::LeftLoop,
           ELFT::PatternClassification::UnableToPrint, ELFT::PatternClassification::UnableToPrint, ELFT::PatternClassification::
           ELFT::PatternClassification::Scar,
           ELFT::PatternClassification::DissociatedRidges }
                             Classification of friction ridge structure.
```

enum ELFT::ValueAssessment { ELFT::ValueAssessment::Value, ELFT::ValueAssessment::Limited,

ELFT::ValueAssessment::NoValue

EFS value assessment codes from ANSI/NIST-ITL 1-2011 (2015).

enum ELFT::Substrate {

ELFT::Substrate::Paper, ELFT::Substrate::Paper, ELFT::Substrate::Plastic, ELFT::Substrate::Glass, ELFT::Substrate::Paper, ELFT::Substrate::Glass, ELFT::Substrate::Paper, ELFT::Substrate::Glass, ELFT::Glass, ELFT::G

ELFT::Substrate::MetalPainted, ELFT::Substrate::MetalUnpainted, ELFT::Substrate::TapeAdhesiveSide,

ELFT::Substrate::NonporousOther,

 $ELFT:: Substrate:: Semiporous Other, \qquad ELFT:: Substrate:: Other, \\$

ELFT::Substrate::Unknown }

EFS substrate codes from ANSI/NIST-ITL 1-2011 (2015).

• enum ELFT::MinutiaType { ELFT::MinutiaType::RidgeEnding, ELFT::MinutiaType::Bifurcation, ELFT::MinutiaType::Other, ELFT::MinutiaType::Unknown }

Types of minutiae.

• enum ELFT::RidgeQuality {

ELFT::RidgeQuality::Background = 0, ELFT::RidgeQuality::DebatableRidgeFlow = 1, ELFT::RidgeQuality::DebatableR

= 2, ELFT::RidgeQuality::DefinitiveMinutiae = 3,

 $ELFT::RidgeQuality::DefinitiveRidgeEdges = 4, ELFT::RidgeQuality::DefinitivePores = 5 \ \}$

Local ridge quality codes from ANSI/NIST-ITL 1-2011 (2015).

• enum ELFT::TemplateType { ELFT::TemplateType::Probe, ELFT::TemplateType::Reference }

Types of templates created by this interface.

Variables

• uint16_t ELFT::API_MAJOR_VERSION {0}

API major version number.

• uint16_t ELFT::API_MINOR_VERSION {0}

API minor version number.

• uint16_t ELFT::API_PATCH_VERSION {1}

API patch version number.

0.4.2 libelft.cpp File Reference

#include <elft.h>

Index

~ExtractionIntertace	ELFT::ExtractionInterface, 22
ELFT::ExtractionInterface, 22	Cyanoacrylate
~SearchInterface	ELFT, 6
ELFT::SearchInterface, 35	
	data
algorithm	ELFT::CreateTemplateResult, 16
ELFT::ProductIdentifier::CBEFFIdentifier, 12	DebatableMinutiae
Amputation	ELFT, 8
ELFT, 6	DebatableRidgeFlow
API_MAJOR_VERSION	ELFT, 8
ELFT, 8	decision
API_MINOR_VERSION	ELFT::SearchResult, 39
ELFT, 9	DefinitiveMinutiae
API_PATCH_VERSION	ELFT, 8
ELFT, 9	DefinitivePores
Arch	ELFT, 8
ELFT, 6	DefinitiveRidgeEdges
EDI 1, 0	ELFT, 8
Background	deltas
ELFT, 8	ELFT::EFS, 20
Bifurcation	DissociatedRidges
ELFT, 8	ELFT, 6
BlackPowder	ELF1,0
ELFT, 6	efs
	ELFT::TemplateData, 43
bpc FI FT. Image 28	-
ELFT::Image, 28	EJIOrTip
bpp	ELFT, 6
ELFT::Image, 28	Electroluminescent
C - 1:1-1-	ELFT, 4
Candidate	ELFT, 1
ELFT::Candidate, 10	Amputation, 6
candidateIdentifier	API_MAJOR_VERSION, 8
ELFT::TemplateData, 42	API_MINOR_VERSION, 9
candidateList	API_PATCH_VERSION, 9
ELFT::SearchResult, 39	Arch, 6
Capacitive	Background, 8
ELFT, 4	Bifurcation, 8
cbeff	BlackPowder, 6
ELFT::ProductIdentifier, 31	Capacitive, 4
Coordinate	Cyanoacrylate, 6
ELFT::Coordinate, 12	DebatableMinutiae, 8
coordinate	DebatableRidgeFlow, 8
ELFT::Minutia, 30	DefinitiveMinutiae, 8
cores	DefinitivePores, 8
ELFT::EFS, 20	DefinitiveRidgeEdges, 8
Correspondence	DissociatedRidges, 6
ELFT::Correspondence, 14	EJIOrTip, 6
createReferenceDatabase	Electroluminescent, 4
ELFT::ExtractionInterface, 25	FrictionRidgeCaptureTechnology
createTemplate	FrictionRidgeCaptureTechnology FrictionRidgeGeneralizedPosition
VIVALVE IVIIII/IIIIV.	THEORIGAN CONTRACTOR OSTROL

Glass, 7	RightIndex, 5
Impression, 4	RightInterdigital, 5
Indanedione, 6	RightLittle, 5
Laser, 6	RightLoop, 6
Latent, 4	RightLowerPalm, 5
LatentImpression, 4	RightMiddle, 5
LatentLift, 4	RightPalmOther, 5
LeftCarpalDeltaArea, 5	RightRing, 5
LeftExtraDigit, 5	RightThenar, 5
LeftFour, 5	RightThumb, 5
LeftFullPalm, 5	RightUpperPalm, 5
LeftFullPalmAndWritersPalm, 5	RightWristBracelet, 5
LeftGrasp, 5	RightWritersPalm, 5
LeftHypothenar, 5	RolledContact, 4
• •	
LeftIndex, 5	RolledContactlessMoving, 4
LeftInterdigital, 5	RolledContactlessStationary, 4
LeftLittle, 5	RUVIS, 6
LeftLoop, 6	ScannedInkOnPaper, 4
LeftLowerPalm, 5	Scar, 6
LeftMiddle, 5	SemiporousOther, 7
LeftPalmOther, 5	StickysidePowder, 6
LeftRing, 5	Substrate, 7
LeftThenar, 5	TapeAdhesiveSide, 7
LeftThumb, 5	TemplateType, 8
LeftUpperPalm, 5	UnableToPrint, 6
LeftWristBracelet, 5	Unclassifiable, 6
LeftWritersPalm, 5	Unknown, 4, 7, 8
Limited, 7	UnknownFinger, 5
LiveScanSwipe, 4	UnknownFrictionRidge, 6
MetalPainted, 7	UnknownPalm, 5
MetalUnpainted, 7	Value, 7
Minutia Type, 7	ValueAssessment, 7
NonporousOther, 7	Visual, 6
NoValue, 7	WhitePowder, 6
OpticalDirect, 4	Whorl, 6
OpticalTIRBright, 4	elft.h, 43
Other, 4, 6–8	ELFT::Candidate, 9
Paper, 7	Candidate, 10
PaperGlossy, 7	frgp, 11
PatternClassification, 6	identifier, 10
PlainContact, 4	operator = 10
PlainContactlessMoving, 4	operator==, 10
PlainContactlessStationary, 4	similarity, 11
Plastic, 7	ELFT::Coordinate, 12
PorousOther, 7	Coordinate, 12
Probe, 8	operator<, 13
ProcessingMethod, 6	operator==, 13
Reference, 8	x, 13
RidgeEnding, 7	y, 13
RidgeQuality, 8	ELFT::Correspondence, 14
RightAndLeftThumbs, 5	Correspondence, 14
RightCarpalDeltaArea, 5	probeInputIdentifier, 15
RightExtraDigit, 5	probeMinutia, 15
RightFour, 5	referenceInputIdentifier, 15
RightFullPalm, 5	referenceMinutia, 15
RightFullPalmAndWritersPalm, 5	ELFT::CreateTemplateResult, 15
RightGrasp, 5	data, 16
RightHypothenar, 5	status, 16
- · · -	

ELFT::EFS, 16	Result, 32
cores, 20	result, 32
deltas, 20	Success, 32
frct, 18	ELFT::RidgeQualityRegion, 33
frgp, 18	quality, 34
identifier, 18	region, 33
imp, 18	ELFT::SearchInterface, 34
lpm, 19	~SearchInterface, 35
lsb, 19	exists, 35
·	extractCorrespondence, 37
minutiae, 20	getIdentification, 35
orientation, 18	getImplementation, 38
pat, 19	insert, 36
plr, 19	
ppi, 18	remove, 36
roi, 20	search, 37
rqm, 20	SearchInterface, 35
trv, 19	ELFT::SearchResult, 39
valueAssessment, 19	candidateList, 39
ELFT::ExtractionInterface, 21	decision, 39
~ExtractionInterface, 22	status, 39
createReferenceDatabase, 25	ELFT::TemplateData, 42
createTemplate, 22	candidateIdentifier, 42
ExtractionInterface, 22	efs, 43
extractTemplateData, 23	imageQuality, 43
getIdentification, 22	inputIdentifier, 43
getImplementation, 25	exemplarAlgorithmIdentifier
mergeTemplates, 24	ELFT::ExtractionInterface::SubmissionIdentification
ELFT::ExtractionInterface::SubmissionIdentification	n, 41
40	exists
exemplarAlgorithmIdentifier, 41	ELFT::SearchInterface, 35
latentAlgorithmIdentifier, 41	extractCorrespondence
libraryIdentifier, 41	ELFT::SearchInterface, 37
SubmissionIdentification, 40	ExtractionInterface
versionNumber, 41	ELFT::ExtractionInterface, 22
ELFT::Image, 26	extractTemplateData
bpc, 28	ELFT::ExtractionInterface, 23
bpp, 28	п 1
height, 28	Failure
identifier, 27	ELFT::ReturnStatus, 32
Image, 27	frct
pixels, 28	ELFT::EFS, 18
ppi, 28	frgp
width, 27	ELFT::Candidate, 11
ELFT::Minutia, 29	ELFT::EFS, 18
coordinate, 30	FrictionRidgeCaptureTechnology
Minutia, 29	ELFT, 4
theta, 30	FrictionRidgeGeneralizedPosition
type, 30	ELFT, 4
ELFT::ProductIdentifier, 30	Q.1
cbeff, 31	getIdentification
marketing, 31	ELFT::ExtractionInterface, 22
ELFT::ProductIdentifier::CBEFFIdentifier, 11	ELFT::SearchInterface, 35
algorithm, 12	getImplementation
· ·	ELFT::ExtractionInterface, 25
owner, 12	ELFT::SearchInterface, 38
ELFT::ReturnStatus, 31	Glass
Failure, 32	ELFT, 7
message, 33	hoight
operator bool, 32	height

ELFT::Image, 28	ELFT, 5
. 1	LeftRing
identifier	ELFT, 5
ELFT::Candidate, 10	LeftThenar
ELFT::EFS, 18	ELFT, 5
ELFT::Image, 27	LeftThumb
Image	ELFT, 5
ELFT::Image, 27	LeftUpperPalm
imageQuality	ELFT, 5
ELFT::TemplateData, 43	LeftWristBracelet
imp	ELFT, 5
ELFT::EFS, 18	LeftWritersPalm
Impression	ELFT, 5
ELFT, 4	
Indanedione	libelft.cpp, 46
ELFT, 6	libraryIdentifier
	ELFT::ExtractionInterface::SubmissionIdentification,
inputIdentifier	41
ELFT::TemplateData, 43	Limited
insert	ELFT, 7
ELFT::SearchInterface, 36	LiveScanSwipe
	ELFT, 4
Laser	lpm
ELFT, 6	ELFT::EFS, 19
Latent	lsb
ELFT, 4	ELFT::EFS, 19
latentAlgorithmIdentifier	EEF 1EF 5, 17
ELFT::ExtractionInterface::SubmissionIdentific	Cation keting
41	ELFT::ProductIdentifier, 31
LatentImpression	
ELFT, 4	mergeTemplates
LatentLift	ELFT::ExtractionInterface, 24
	message
ELFT, 4	ELFT::ReturnStatus, 33
LeftCarpalDeltaArea	MetalPainted
ELFT, 5	ELFT, 7
LeftExtraDigit	MetalUnpainted
ELFT, 5	ELFT, 7
LeftFour	Minutia
ELFT, 5	ELFT::Minutia, 29
LeftFullPalm	minutiae
ELFT, 5	ELFT::EFS, 20
LeftFullPalmAndWritersPalm	
ELFT, 5	MinutiaType
LeftGrasp	ELFT, 7
ELFT, 5	Name are constituted as
	NonporousOther
LeftHypothenar	ELFT, 7
ELFT, 5	NoValue
LeftIndex	ELFT, 7
ELFT, 5	
LeftInterdigital	operator bool
ELFT, 5	ELFT::ReturnStatus, 32
LeftLittle	operator<
ELFT, 5	ELFT::Candidate, 10
LeftLoop	ELFT::Coordinate, 13
ELFT, 6	operator==
LeftLowerPalm	ELFT::Candidate, 10
ELFT, 5	ELFT::Coordinate, 13
LeftMiddle	OpticalDirect
ELFT, 5	ELFT, 4
LeftPalmOther	
Letu aiiiiOtilei	OpticalTIRBright

DI DE 4	D: 1
ELFT, 4	RidgeEnding
orientation	ELFT, 7
ELFT::EFS, 18	RidgeQuality
Other	ELFT, 8
ELFT, 4, 6–8	
	RightAndLeftThumbs
owner	ELFT, 5
ELFT::ProductIdentifier::CBEFFIdentifier, 12	RightCarpalDeltaArea
	ELFT, 5
Paper	
ELFT, 7	RightExtraDigit
	ELFT, 5
PaperGlossy	RightFour
ELFT, 7	ELFT, 5
pat	RightFullPalm
ELFT::EFS, 19	ELFT, 5
PatternClassification	
	RightFullPalmAndWritersPalm
ELFT, 6	ELFT, 5
pixels	RightGrasp
ELFT::Image, 28	ELFT, 5
PlainContact	
ELFT, 4	RightHypothenar
	ELFT, 5
PlainContactlessMoving	RightIndex
ELFT, 4	ELFT, 5
PlainContactlessStationary	RightInterdigital
ELFT, 4	-
Plastic	ELFT, 5
	RightLittle
ELFT, 7	ELFT, 5
plr	RightLoop
ELFT::EFS, 19	-
PorousOther	ELFT, 6
ELFT, 7	RightLowerPalm
	ELFT, 5
ppi	RightMiddle
ELFT::EFS, 18	ELFT, 5
ELFT::Image, 28	
Probe	RightPalmOther
ELFT, 8	ELFT, 5
	RightRing
probeInputIdentifier	ELFT, 5
ELFT::Correspondence, 15	RightThenar
probeMinutia	
ELFT::Correspondence, 15	ELFT, 5
ProcessingMethod	RightThumb
ELFT, 6	ELFT, 5
ELF1, 0	RightUpperPalm
11.	ELFT, 5
quality	
ELFT::RidgeQualityRegion, 34	RightWristBracelet
	ELFT, 5
Reference	RightWritersPalm
ELFT, 8	ELFT, 5
	roi
referenceInputIdentifier	
ELFT::Correspondence, 15	ELFT::EFS, 20
referenceMinutia	RolledContact
ELFT::Correspondence, 15	ELFT, 4
	RolledContactlessMoving
region	<u> </u>
ELFT::RidgeQualityRegion, 33	ELFT, 4
remove	RolledContactlessStationary
ELFT::SearchInterface, 36	ELFT, 4
Result	rqm
ELFT::ReturnStatus, 32	ELFT::EFS, 20
result	RUVIS
ELFT::ReturnStatus, 32	ELFT, 6

```
ScannedInkOnPaper
                                                       ELFT, 6
    ELFT, 4
                                                  WhitePowder
Scar
                                                       ELFT, 6
    ELFT, 6
                                                   Whorl
search
                                                       ELFT, 6
    ELFT::SearchInterface, 37
                                                  width
SearchInterface
                                                       ELFT::Image, 27
    ELFT::SearchInterface, 35
SemiporousOther
                                                  \mathbf{x}
    ELFT, 7
                                                       ELFT::Coordinate, 13
similarity
    ELFT::Candidate, 11
                                                  y
                                                       ELFT::Coordinate, 13
    ELFT::CreateTemplateResult, 16
    ELFT::SearchResult, 39
StickysidePowder
    ELFT, 6
SubmissionIdentification
    ELFT::ExtractionInterface::SubmissionIdentification,
Substrate
    ELFT, 7
Success
    ELFT::ReturnStatus, 32
TapeAdhesiveSide
    ELFT, 7
TemplateType
    ELFT, 8
theta
    ELFT::Minutia, 30
trv
    ELFT::EFS, 19
type
    ELFT::Minutia, 30
UnableToPrint
    ELFT, 6
Unclassifiable
    ELFT, 6
Unknown
    ELFT, 4, 7, 8
UnknownFinger
    ELFT, 5
UnknownFrictionRidge
    ELFT, 6
UnknownPalm
    ELFT, 5
Value
    ELFT, 7
ValueAssessment
    ELFT, 7
valueAssessment
    ELFT::EFS, 19
versionNumber
    ELFT::ExtractionInterface::SubmissionIdentification,
         41
```

Visual