



---

## Hierarchy For All Packages

### Class Hierarchy

- class java.lang.Object
  - ◆ class org.javacardforum.javacard.biometry.**BioBuilder**
  - ◆ class java.lang.Throwable
    - ◇ class java.lang.Exception
      - class java.lang.RuntimeException
        - class javacard.framework.CardRuntimeException
          - ◆ class org.javacardforum.javacard.biometry.**BioException**

### Interface Hierarchy

- interface org.javacardforum.javacard.biometry.**BioTemplate**
    - ◆ interface org.javacardforum.javacard.biometry.**OwnerBioTemplate**
    - ◆ interface org.javacardforum.javacard.biometry.**SharedBioTemplate** (also extends javacard.framework.Shareable)
  - interface javacard.framework.Shareable
    - ◆ interface org.javacardforum.javacard.biometry.**SharedBioTemplate** (also extends org.javacardforum.javacard.biometry.**BioTemplate**)
- 
-



---

## org.javacardforum.javacard.biometry Class BioBuilder

```
java.lang.Object
|
+--org.javacardforum.javacard.biometry.BioBuilder
```

---

*public final class **BioBuilder***  
*extends java.lang.Object*

Builds an empty/blank reference template.

**See Also:**

[OwnerBioTemplate](#)

---

Field Summary	
static byte	<b>BODY_ODOR</b> Body Odor.
static byte	<b>DNA_SCAN</b> Pattern is a DNA sample for matching.
static byte	<b>EAR_GEOMETRY</b> Ear geometry ID is based on overall geometry/shape of the ear.
static byte	<b>FACIAL_FEATURE</b> Facial feature recognition (visage).
static byte	<b>FINGER_GEOMETRY</b> Finger geometry ID is based on overall geometry/shape of a finger.
static byte	<b>FINGERPRINT</b> Fingerprint identification (any finger).
static byte	<b>GAIT_STYLE</b> Gait (behavioral).
static byte	<b>HAND_GEOMETRY</b> Hand geometry ID is based on overall geometry/shape of the hand.
static byte	<b>IRIS_SCAN</b> Pattern is a scan of the eye's iris.



static byte	<b>KEYSTROKES</b> Keystrokes dynamics (behavioral).
static byte	<b>LIP_MOVEMENT</b> Lip movement (behavioral).
static byte	<b>PALM_GEOMETRY</b> Palm geometry ID is based on overall geometry/shape of a palm.
static byte	<b>PASSWORD</b> General password (a PIN is a special case of the password).
static byte	<b>RETINA_SCAN</b> Pattern is an infrared scan of the blood vessels of the retina of the eye.
static byte	<b>SIGNATURE</b> Written signature dynamics ID (behavioral).
static byte	<b>THERMAL_FACE</b> Thermal Face Image.
static byte	<b>THERMAL_HAND</b> Thermal Hand Image.
static byte	<b>VEIN_PATTERN</b> Pattern is an infrared scan of the vein pattern in a face, wrist, or, hand.
static byte	<b>VOICE_PRINT</b> Pattern is a voice sample (specific or unspecified speech).

## Method Summary

static <code>OwnerBioTemplate</code>	<b><code>buildBioTemplate</code></b> (byte bioType, byte tryLimit) Creates an empty/blank biometric reference template.
--------------------------------------	--

## Methods inherited from class `java.lang.Object`

`equals`

## Field Detail



## **FACIAL\_FEATURE**

```
public static final byte FACIAL_FEATURE
```

Facial feature recognition (visage). Value = (byte)1;

---

## **VOICE\_PRINT**

```
public static final byte VOICE_PRINT
```

Pattern is a voice sample (specific or unspecified speech). Value = (byte)2;

---

## **FINGERPRINT**

```
public static final byte FINGERPRINT
```

Fingerprint identification (any finger). Value = (byte)3;

---

## **IRIS\_SCAN**

```
public static final byte IRIS_SCAN
```

Pattern is a scan of the eye's iris. Value = (byte)4;

---

## **RETINA\_SCAN**

```
public static final byte RETINA_SCAN
```

Pattern is an infrared scan of the blood vessels of the retina of the eye. Value = (byte)5;

---

## **HAND\_GEOMETRY**

```
public static final byte HAND_GEOMETRY
```

Hand geometry ID is based on overall geometry/shape of the hand. Value = (byte)6;

---



## **SIGNATURE**

```
public static final byte SIGNATURE
```

Written signature dynamics ID (behavioral). Value = (byte)7;

---

## **KEYSTROKES**

```
public static final byte KEYSTROKES
```

Keystrokes dynamics (behavioral). Value = (byte)8;

---

## **LIP\_MOVEMENT**

```
public static final byte LIP_MOVEMENT
```

Lip movement (behavioral). Value = (byte)9;

---

## **THERMAL\_FACE**

```
public static final byte THERMAL_FACE
```

Thermal Face Image. Value = (byte)10;

---

## **THERMAL\_HAND**

```
public static final byte THERMAL_HAND
```

Thermal Hand Image. Value = (byte)11;

---

## **GAIT\_STYLE**

```
public static final byte GAIT_STYLE
```

Gait (behavioral). Value = (byte)12;

---



## **BODY\_ODOR**

```
public static final byte BODY_ODOR
```

Body Odor. Value = (byte)13;

---

## **DNA\_SCAN**

```
public static final byte DNA_SCAN
```

Pattern is a DNA sample for matching. Value = (byte)14;

---

## **EAR\_GEOMETRY**

```
public static final byte EAR_GEOMETRY
```

Ear geometry ID is based on overall geometry/shape of the ear. Value = (byte)15;

---

## **FINGER\_GEOMETRY**

```
public static final byte FINGER_GEOMETRY
```

Finger geometry ID is based on overall geometry/shape of a finger. Value = (byte)16;

---

## **PALM\_GEOMETRY**

```
public static final byte PALM_GEOMETRY
```

Palm geometry ID is based on overall geometry/shape of a palm. Value = (byte)17;

---

## **VEIN\_PATTERN**

```
public static final byte VEIN_PATTERN
```

Pattern is an infrared scan of the vein pattern in a face, wrist, or, hand. Value = (byte)18;

---



## PASSWORD

```
public static final byte PASSWORD
```

General password (a PIN is a special case of the password). Note that this is not a biometric, but is nevertheless a pattern that must be matched for security purposes, and since it is frequently combined with biometrics for security, we provide a code here to assist with that combination. Value = (byte)31;

### Method Detail

#### buildBioTemplate

```
public static OwnerBioTemplate buildBioTemplate(byte bioType,  
                                               byte tryLimit)  
    throws BioException
```

Creates an empty/blank biometric reference template.

**Parameters:**

`bioType` – the type of the template to be generated. Valid codes are listed in the biometric pattern type constants.

`tryLimit` – maximum unsuccessful matches before template is blocked. `tryLimit` must be  $\geq 1$ .

**Returns:**

the `OwnerBioTemplate` object instance of the requested type and `tryLimit` access.

**Throws:**

`BioException.ILLEGAL_VALUE` – if `tryLimit` parameter is less than 1.

`BioException.NO_SUCH_BIO_TEMPLATE` – if the requested template associated with the specified `bioType` is not supported.

---

---



## org.javacardforum.javacard.biometry

### Class BioException

```

java.lang.Object
|
+--java.lang.Throwable
    |
    +--java.lang.Exception
        |
        +--java.lang.RuntimeException
            |
            +--javacard.framework.CardRuntimeException
                |
                +--org.javacardforum.javacard.biometry.BioException
  
```

*public class **BioException***  
*extends javacard.framework.CardRuntimeException*

The BioException class encapsulates specific exceptions which can be thrown by the methods of the biometry package in case of error.

*See Also:*

[BioBuilder](#), [BioTemplate](#), [OwnerBioTemplate](#)

Field Summary	
static short	<b>ILLEGAL_USE</b> This reason code is used to indicate that the method should not be invoked based on the current state of the card.
static short	<b>ILLEGAL_VALUE</b> This reason code is used to indicate that one or more input parameters is out of allowed bounds.
static short	<b>INVALID_DATA</b> This reason code is used to indicate that the data the system encountered is illegible.
static short	<b>NO_SUCH_BIO_TEMPLATE</b> This reason code is used to indicate that the provided bio template type is not supported by the template builder.
static short	<b>NO_TEMPLATES_ENROLLED</b> This reason code is used to indicate that no reference template is available for



matching, or that the reference template is uninitialized.
--

## Constructor Summary

<code>BioException</code> (short reason) Construct a new biometric exception using a provided reason code.
---

## Method Summary

static void	<code>throwIt</code> (short reason) Throws the JCRE owned instance of <code>BioException</code> with the specified reason.
-------------	---

### Methods inherited from class `javacard.framework.CardRuntimeException`

`getReason`, `setReason`

### Methods inherited from class `java.lang.Object`

`equals`

## Field Detail

### ILLEGAL\_VALUE

```
public static final short ILLEGAL_VALUE
```

This reason code is used to indicate that one or more input parameters is out of allowed bounds. Value = (short)1;

---

### INVALID\_DATA

```
public static final short INVALID_DATA
```

This reason code is used to indicate that the data the system encountered is illegible. Value = (short)2;

---



## NO\_SUCH\_BIO\_TEMPLATE

```
public static final short NO_SUCH_BIO_TEMPLATE
```

This reason code is used to indicate that the provided bio template type is not supported by the template builder. Value = (short)3;

---

## NO\_TEMPLATES\_ENROLLED

```
public static final short NO_TEMPLATES_ENROLLED
```

This reason code is used to indicate that no reference template is available for matching, or that the reference template is uninitialized. Value = (short)4;

---

## ILLEGAL\_USE

```
public static final short ILLEGAL_USE
```

This reason code is used to indicate that the method should not be invoked based on the current state of the card. Value = (short)5;

## Constructor Detail

### BioException

```
public BioException(short reason)
```

Construct a new biometric exception using a provided reason code. To conserve on resources use `throwIt()` to use the JCRE instance of this class.

**Parameters:**

reason – the reason code for this exception.

## Method Detail

### throwIt

```
public static void throwIt(short reason)
    throws BioException
```



Throws the JCRE owned instance of [BioException](#) with the specified reason. JCRE owned instances of exception classes are temporary JCRE Entry Point Objects and can be accessed from any applet context. References to these objects cannot be stored in class variables or instance variables or array components. See "Java Card Runtime Environment (JCRE) 2.2 Specification" for details.

**Parameters:**

reason – the reason for the exception.

**Throws:**

[BioException](#) – always.

---

---



---

## org.javacardforum.javacard.biometry

# Interface BioTemplate

*All Known Subinterfaces:*

[OwnerBioTemplate](#), [SharedBioTemplate](#)

---

*public interface* **BioTemplate**

The BioTemplate interface is the base interface for all biometric templates. It provides to the user the means for accessing biometric functionality.

*See Also:*

[OwnerBioTemplate](#)

---

### Field Summary

static short	<a href="#">MATCH_NEEDS_MORE_DATA</a> This score value indicates that more data are needed to continue the matching session.
static short	<a href="#">MINIMUM_SUCCESSFUL_MATCH_SCORE</a> The minimum successful matching score.

### Method Summary

byte	<a href="#">getBioType()</a> Get the biometric type.
short	<a href="#">getPublicTemplateData</a> (short publicOffset, byte[] dest, short destOffset, short length) Get public part of the reference template.
byte	<a href="#">getTriesRemaining()</a> Returns the number of times remaining that an incorrect candidate template can be presented before the reference template is blocked.
short	<a href="#">getVersion</a> (byte[] dest, short offset) Get the matching algorithm version and ID.
short	<a href="#">initMatch</a> (byte[] candidate, short offset, short length) Initialize or re-initialize a biometric matching session.



boolean	<b>isInitialized()</b> Returns true if the reference template is completely loaded and ready for matching functions.
boolean	<b>isValidated()</b> Returns true if the template has been successfully checked since the last card reset or last call to <code>reset()</code> .
short	<b>match</b> (byte[] candidate, short offset, short length) Continues a biometric matching session.
void	<b>reset()</b> Resets the reference validated flag.

## Field Detail

### MINIMUM\_SUCCESSFUL\_MATCH\_SCORE

```
public static final short MINIMUM_SUCCESSFUL_MATCH_SCORE
```

The minimum successful matching score. Value =(short)16384;

---

### MATCH\_NEEDS\_MORE\_DATA

```
public static final short MATCH_NEEDS_MORE_DATA
```

This score value indicates that more data are needed to continue the matching session. Value =(short)-1;

## Method Detail

### isInitialized

```
public boolean isInitialized()
```

Returns true if the reference template is completely loaded and ready for matching functions. This is independent of whether or not the match process has been initialized (see `initMatch`).

**Returns:**

true if initialized, false otherwise.

---



## isValidated

```
public boolean isValidated()
```

Returns `true` if the template has been successfully checked since the last card reset or last call to `reset()`.

**Returns:**

`true` if validated, `false` otherwise.

---

## reset

```
public void reset()
```

Resets the reference validated flag. This could be appropriate as a last action after an access is completed.

---

## getTriesRemaining

```
public byte getTriesRemaining()
```

Returns the number of times remaining that an incorrect candidate template can be presented before the reference template is blocked.

**Returns:**

the number of tries remaining.

---

## getBioType

```
public byte getBioType()
```

Get the biometric type. Valid type are described in [BioBuilder](#).

**Returns:**

biometric general type.

---

## getVersion

```
public short getVersion(byte[] dest,  
                        short offset)
```

Get the matching algorithm version and ID.

**Parameters:**

`dest` – : destination byte array.

`offset` – : starting offset within the destination byte array.

**Returns:**

number of bytes written in the destination byte array.

---

## getPublicTemplateData

```
public short getPublicTemplateData(short publicOffset,  
                                   byte[] dest,  
                                   short destOffset,  
                                   short length)  
    throws BioException
```

Get public part of the reference template. It copies all or a piece of the reference public data to the destination array.

**Parameters:**

publicOffset – starting offset within the public data.

dest – destination byte array.

destOffset – starting offset within the destination byte array.

length – maximum bytelength of the requested data.

**Returns:**

- number of bytes really written in the destination byte array.
- 0 if public data are not available.

**Throws:**

**BioException.NO\_TEMPLATES\_ENROLLED** – if the reference template is uninitialized.

---

## initMatch

```
public short initMatch(byte[] candidate,  
                       short offset,  
                       short length)  
    throws BioException
```

Initialize or re-initialize a biometric matching session. The exact return score value is implementation dependant and can be used, for example, to code a confidence rate. If the reference is not blocked, a matching session starts and, before any other processing, the validated flag is reset, the try counter is decremented if it has reached zero, the reference is blocked.

This method makes the matching session :

- ◆ end with success state if the templates match: the validated flag is set, the try counter is reset to its maximum and if the reference has been blocked at the beginning, it is unblocked ; or
- ◆ end with failed state if the templates don't match ; or
- ◆ continue if the matching needs more data : `match` method has to be called to continue the matching session.

Notes:



- ◆ A correct matching sequence is : `initMatch-[match]`. Calling `initMatch` is mandatory, calling `match` is optional.
- ◆ If a matching session is in progress (case needs more data), a call to `initMatch` makes the current session to fail and starts a new matching session.
- ◆ Even if a transaction is in progress, internal state such as the try counter, the validated flag and the blocking state must not be conditionally updated.

**Parameters:**

`candidate` – the data or part of the data of the candidate template.

`offset` – starting offset into the candidate array where the candidate data is to be found.

`length` – number of bytes to be taken from the candidate array.

**Returns:**

the matching score with the following meaning :

- $\geq$  `MINIMUM_SUCCESSFUL_MATCH_SCORE` : the matching session is successful
- $\geq 0$  and  $<$  `MINIMUM_SUCCESSFUL_MATCH_SCORE` : the matching session has failed
- `MATCH_NEEDS_MORE_DATA` : the matching session needs more data
- other negative values are reserved for future use

**Throws:**

`IOException.INVALID_DATA` – if the submitted candidate template data does not have the required format.

`IOException.NO_TEMPLATES_ENROLLED` – if the reference template is uninitialized.

**See Also:**

[match\(byte\[\], short, short\)](#), [MATCH\\_NEEDS\\_MORE\\_DATA](#), [MINIMUM\\_SUCCESSFUL\\_MATCH\\_SCORE](#)

---

## match

```
public short match(byte[] candidate,  
                   short offset,  
                   short length)  
    throws IOException
```

Continues a biometric matching session. The exact return score value is implementation dependant and can be used, for example, to code a confidence rate.

If a matching session is in progress, this method makes the session :

- ◆ end with success state if the templates match: the validated flag is set, the try counter is reset to its maximum and if the reference has been blocked at the beginning of the session, it is unblocked ; or
- ◆ end with failed state if the templates don't match ; or
- ◆ continue if the matching needs more data : a new `match` method has to be called to continue the matching session.

Note:



- ◆ A correct matching sequence is : `initMatch-[match]`. Calling `initMatch` is mandatory, calling `match` is optional.
- ◆ Even if a transaction is in progress, internal state such as the try counter, the validated flag and the blocking state must not be conditionally updated.

**Parameters:**

`candidate` – the data or part of the data of the candidate template.  
`offset` – starting offset into the candidate array where the candidate data is to be found.  
`length` – number of bytes to be taken from the candidate array.

**Returns:**

the matching score with the following meaning :

- `>= MINIMUM_SUCCESSFUL_MATCH_SCORE` : the matching session is successful
- `>= 0` and `< MINIMUM_SUCCESSFUL_MATCH_SCORE` : the matching session has failed
- `= MATCH_NEEDS_MORE_DATA` : the matching session needs more data
- other negative values are reserved for future use

**Throws:**

`IOException.ILLEGAL_USE` – if used outside a matching session.  
`IOException.INVALID_DATA` – if the submitted candidate template data does not have the required format.  
`IOException.NO_TEMPLATES_ENROLLED` – if the reference template is uninitialized.

**See Also:**

`initMatch(byte[], short, short)`, `MATCH_NEEDS_MORE_DATA`,  
`MINIMUM_SUCCESSFUL_MATCH_SCORE`

---

---



---

## org.javacardforum.javacard.biometry Interface OwnerBioTemplate

*All Superinterfaces:*  
[BioTemplate](#)

---

*public interface **OwnerBioTemplate***  
*extends [BioTemplate](#)*

The OwnerBioTemplate interface provides the ability to enrol a reference template and thus owner functionality.

*See Also:*  
[OwnerBioTemplate](#), [BioBuilder](#)

---

### Fields inherited from interface org.javacardforum.javacard.biometry.[BioTemplate](#)

[MATCH\\_NEEDS\\_MORE\\_DATA](#), [MINIMUM\\_SUCCESSFUL\\_MATCH\\_SCORE](#)

### Method Summary

void	<a href="#">doFinal</a> () Finalizes the enrolment of a reference template.
void	<a href="#">init</a> (byte[] bArray, short offset, short length) Initializes the enrolment of a reference template.
void	<a href="#">resetUnblockAndSetTryLimit</a> (byte newTryLimit) Resets the validated flag, unblocks the reference, updates the try limit value and resets the try counter to the try limit value.
void	<a href="#">update</a> (byte[] bArray, short offset, short length) Continues the enrolment of a reference template.

### Methods inherited from interface org.javacardforum.javacard.biometry.[BioTemplate](#)

[getBioType](#), [getPublicTemplateData](#), [getTriesRemaining](#), [getVersion](#),  
[initMatch](#), [isInitialized](#), [isValidated](#), [match](#), [reset](#)

## Method Detail

### init

```
public void init(byte[] bArray,
                 short offset,
                 short length)
    throws BioException
```

Initializes the enrolment of a reference template. It is also used to update a reference template. It resets the validated flag and, in the update case, uninitialized the previous reference.

Note:

- ◆ A correct enrolment sequence is : `init`–[`update`]–`doFinal`. Calling `init` and `doFinal` is mandatory, calling `update` is optional.

**Parameters:**

`bArray` – byte array containing the data of the template  
`offset` – starting offset in the `bArray`  
`length` – byte length of the template data in the `bArray`

**Throws:**

`BioException.INVALID_DATA` – if the submitted template data does not have the required format.

**See Also:**

[update\(byte\[\], short, short\), doFinal\(\)](#)

---

### update

```
public void update(byte[] bArray,
                   short offset,
                   short length)
    throws BioException
```

Continues the enrolment of a reference template. This method should only be used if all the input data required for the `init` is not available in one byte array. It can be called several times.

Note:

- ◆ A correct enrolment sequence is : `init`–[`update`]–`doFinal`. Calling `init` and `doFinal` is mandatory, calling `update` is optional.

**Parameters:**

`bArray` – byte array containing the data of the template  
`offset` – starting offset in the `bArray`  
`length` – byte length of the template data in the `bArray`

**Throws:**

`BioException.ILLEGAL_USE` – if the reference is already initialized or the current



enrolment state doesn't expect this method.

`BioException.INVALID_DATA` – if the submitted template data does not have the required format.

*See Also:*

`init(byte[], short, short), doFinal()`

---

## doFinal

```
public void doFinal()  
    throws BioException
```

Finalizes the enrolment of a reference template. Final action of enrollment is to designate a reference template as being complete and ready for use (marks the reference as initialized, resets the try counter and unblocks the reference). This routine may also include some error checking prior to the validation of reference template as ready for use.

Note:

- ◆ A correct enrolment sequence is : `init`–`update`–`doFinal`. Calling `init` and `doFinal` is mandatory, calling `update` is optional.

*Throws:*

`BioException.ILLEGAL_USE` – if the reference is already initialized or the current enrolment state doesn't expect this method.

`BioException.INVALID_DATA` – if the submitted template data does not have the required format.

*See Also:*

`init(byte[], short, short), update(byte[], short, short)`

---

## resetUnblockAndSetTryLimit

```
public void resetUnblockAndSetTryLimit(byte newTryLimit)  
    throws BioException
```

Resets the validated flag, unblocks the reference, updates the try limit value and resets the try counter to the try limit value.

*Parameters:*

`newTryLimit` – the number of tries allowed before the reference is blocked.  
`newTryLimit` must be  $\geq 1$ .

*Throws:*

`BioException.ILLEGAL_VALUE` – if `newTryLimit` parameter is less than 1.

---

---



---

## org.javacardforum.javacard.biometry Interface SharedBioTemplate

*All Superinterfaces:*

[BioTemplate](#), [javacard.framework.Shareable](#)

---

*public interface **SharedBioTemplate***

*extends [BioTemplate](#), [javacard.framework.Shareable](#)*

The **SharedBioTemplate** interface provides the means for accessing unrestricted biometric functionality, e.g., the biometric matching functions. A biometric manager/server can implement this interface with a proxy to the public matching functions; thus giving a biometric client access to matching functions but not to the enrolment functions. Without this interface, the client could potentially cast a biometric reference to gain access to enrolment functionality and thereby circumvent security measures.

*See Also:*

[OwnerBioTemplate](#)

---

### Fields inherited from interface org.javacardforum.javacard.biometry.[BioTemplate](#)

[MATCH\\_NEEDS\\_MORE\\_DATA](#), [MINIMUM\\_SUCCESSFUL\\_MATCH\\_SCORE](#)

### Methods inherited from interface org.javacardforum.javacard.biometry.[BioTemplate](#)

[getBioType](#), [getPublicTemplateData](#), [getTriesRemaining](#), [getVersion](#),  
[initMatch](#), [isInitialized](#), [isValidated](#), [match](#), [reset](#)

---

---