

# *eNBSP SDK for JAVA*

## *Programmer's Manual*

*SDK Ver 5.x*

**© Copyright 2009 – 2012 NITGEN&COMPANY Co., Ltd.**

**ALL RIGHTS RESERVED**

**Specifications subject to change without notice.**

"NITGEN", the NITGEN logo, "eNBSP", "NBioBSP", "NBioAPI",

"NITGEN Fingkey Mouse", and "eNDeSS" are trademarks of

NITGEN&COMPANY Co., Ltd. All other brands or products may be trademarks

or service marks of their respective owners.

# Contents

<b>CHAPTER 1 INTRODUCTION .....</b>	<b>1</b>
<b>CHAPTER 2 FILE LIST.....</b>	<b>2</b>
<b>CHAPTER 3 JAVA CLASS LIBRARY.....</b>	<b>3</b>
■ NBioBSPJNI Class Initialize and Terminate.....	3
■ Device Control .....	3
■ Fingerprint Enrollment .....	4
■ Verification .....	5
■ Client / Server Environment.....	6
<b>CHAPTER 4 NBIOBSPJNI CLASS .....</b>	<b>7</b>
■ NBioBSPJNI.ERROR .....	7

■ NBioBSPJNI.FIR_SECURITY_LEVEL.....	9
■ NBioBSPJNI.FINGER_ID.....	9
■ NBioBSPJNI.DEVICE_ID .....	10
■ NBioBSPJNI.DEVICE_NAME.....	10
■ NBioBSPJNI.FIR_FORMAT .....	11
■ NBioBSPJNI.FIR_DATA_TYPE .....	11
■ NBioBSPJNI.FIR_PURPOSE .....	12
■ NBioBSPJNI.FIR_FORM .....	12
■ NBioBSPJNI.WINDOW_STYLE.....	13
■ NBioBSPJNI.INIT_INFO_0 .....	14
■ NBioBSPJNI.DEVICE_INFO .....	15
■ NBioBSPJNI.DEVICE_INFO_EX.....	15
■ NBioBSPJNI.DEVICE_ENUM_INFO.....	16
■ NBioBSPJNI.MATCH_OPTION .....	16

■ NBioBSPJNI.FIR_HANDLE .....	16
■ NBioBSPJNI.FIR_HEADER .....	17
■ NBioBSPJNI.FIR .....	17
■ NBioBSPJNI.FIR_TEXTENCODING .....	17
■ NBioBSPJNI.INPUT_FIR .....	18
■ NBioBSPJNI.FIR_PAYLOAD .....	19
■ NBioBSPJNI.WINDOW_OPTION .....	20
■ NBioBSPJNI .....	21
int GetErrorCode() .....	21
Boolean IsErrorOccured() .....	21
void dispose() .....	21
int SetSkinResource(String szResPath) .....	21
String GetVersion() .....	22
int GetInitInfo(INIT_INFO_0 initInfo0) .....	22
int SetInitInfo(INIT_INFO_0 initInfo0) .....	22
int EnumerateDevice(DEVICE_ENUM_INFO deviceInfo) .....	22
int OpenDevice() .....	23
int CloseDevice() .....	23

int GetDeviceInfo(DEVICE_INFO deviceInfo) .....	24
int SetDeviceInfo(DEVICE_INFO deviceInfo) .....	24
int AdjustDevice() .....	25
short GetOpenedDeviceID() .....	25
int GetFIRFromHandle(FIR_HANDLE hFIR, FIR fullFIR) .....	25
int GetTextFIRFromHandle(FIR_HANDLE hFIR, FIR_TEXTENCODING textFIR) .....	26
int Enroll(FIR_HANDLE hEnrolledFIR, FIR_PAYLOAD payload) .....	26
FIR_HANDLE hAudit, WINDOW_OPTION winOption) .....	26
int Capture(FIR_HANDLE hFIR) .....	27
FIR_HANDLE hAudit, WINDOW_OPTION winOption) .....	27
int RollCapture(FIR_HANDLE hFIR) .....	28
FIR_HANDLE hAudit, WINDOW_OPTION winOption) .....	28
int Process(INPUT_FIR capturedFIR, FIR_HANDLE hProcessedFIR) .....	29
int CreateTemplate(INPUT_FIR capturedFIR, INPUT_FIR storedFIR, .....	29
int Verify(INPUT_FIR storedFIR, Boolean bResult, FIR_PAYLOAD payload) .....	30
int nTimeout, FIR_HANDLE hAudit, WINDOW_OPTION winOption) .....	30
int VerifyMatch(INPUT_FIR capturedFIR, INPUT_FIR storedFIR, Boolean bResult, FIR_PAYLOAD payload) .....	31
FIR_PAYLOAD payload, MATCH_OPTION matchOption) .....	31
int CheckFinger(Boolean bFingerExist) .....	31
int GetNFIQInfoFromHandle(FIR_HANDLE auditData, NFIQINFO qualityInfo) .....	32
static int GetNFIQFromNBioBSP(Export.AUDIT AuditData, NFIQINFO Info) .....	32
static int GetNFIQFromRaw(byte[] RawImage, int ImgW, int ImgH, Integer NFIQ) .....	32
static int ExportRawToISOV1(NBioBSPJNI.Export.AUDIT AuditData, ISOBUFFER ISOBuf,	

boolean blsRollDevice, byte CompressMode) .....	32
static int ExportRawToISOV2(byte FingerID, short ImgW, short ImgH, byte[] RawBuf, ISOBUFFER ISOBuf, boolean blsRollDevice, byte CompressMode).....	33
static int ImportISOToRaw(ISOBUFFER ISOBuf, NIMPORTRAWSET RawSet).....	33
<b>■ NBioBSPJNI.EXPORT_MINCONV_TYPE .....</b>	<b>34</b>
<b>■ NBioBSPJNI.Export.TEMPLATE_DATA .....</b>	<b>36</b>
<b>■ NBioBSPJNI.Export.FINGER_DATA .....</b>	<b>36</b>
<b>■ NBioBSPJNI.Export.DATA .....</b>	<b>36</b>
<b>■ NBioBSPJNI.Export.AUDIT .....</b>	<b>37</b>
<b>■ NBioBSPJNI.Export.....</b>	<b>38</b>
int ExportFIR(INPUT_FIR inputFIR, Export.DATA exportData, int exportType).....	38
int ImportFIR(byte[] Template, int nLen, int type, FIR_HANDLE hFIR) .....	38
int ExportAudit(INPUT_FIR inputFIR, Export.AUDIT exportAudit) .....	39
int ImportAudit(Export.AUDIT exportAudit, FIR_HANDLE hFIR).....	39
int ImportBioAPIOpaqueToFIR(byte[] bioAPIOpaqueData, FIR_HANDLE hFIR).....	39
int ConvertRawToWsqr(byte[] rawImage, int nWidth, int nHeight, Export.TEMPLATE_DATA wsqrImage, float fQuality) .....	40
int ConvertWsqrToRaw(byte[] wsqrImage, int nWsqrLen, Export.AUDIT exportAudit) ..	40
<b>■ NBioBSPJNI.IndexSearch.....</b>	<b>42</b>

void dispose().....	42
int GetInitInfo(IndexSearch.INIT_INFO initInfo).....	42
int SetInitInfo(IndexSearch.INIT_INFO initInfo).....	42
int AddFIR(INPUT_FIR inputFIR, int userID, IndexSearch.SAMPLE_INFO sampleInfo) .....	43
int Identify(INPUT_FIR inputFIR, int secuLevel, IndexSearch.FP_INFO fpInfo) .....	43
int RemoveData(IndexSearch.FP_INFO fpInfo) .....	44
int RemoveUser(int userID).....	44
int ClearDB() .....	44
int SaveDB(String szFilePath) .....	44
int LoadDB(String szFilePath).....	45
int GetDBCount(Integer nCount).....	45
int CheckDataExist(IndexSearch.FP_INFO fpInfo, Boolean bExist) .....	45
<b>■ NBioBSPJNI.NFIQINFO .....</b>	<b>46</b>
<b>■ NBioBSPJNI.COMPRESS_MODE .....</b>	<b>46</b>
<b>■ NBioBSPJNI.NIMPORTRAW .....</b>	<b>46</b>
<b>■ NBioBSPJNI.NIMPORTRAWSET .....</b>	<b>47</b>
<b>■ NBioBSPJNI.ISOBUFFER.....</b>	<b>47</b>

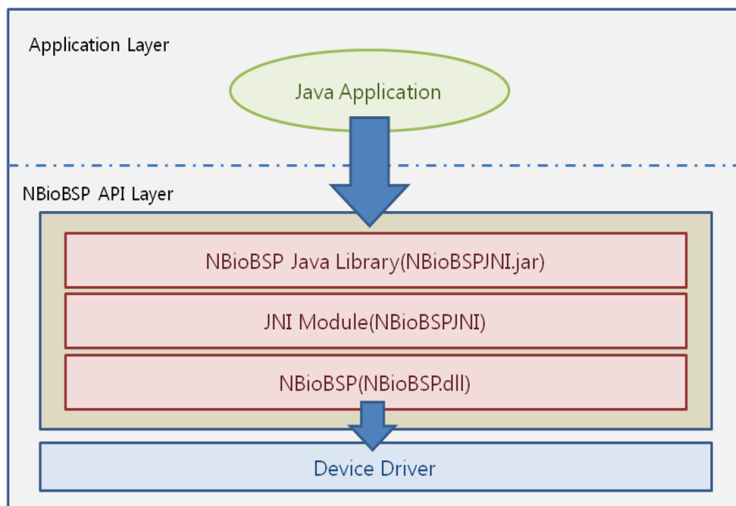


# Chapter 1 Introduction

The NBioBSP Java Library (NBioBSPJNI.jar) is designed to support developers using JAVA environment.

NBioBSP Java Library based Java Native Interface Technology that facilitates easily integration of NBioBSP by developers using Java development.

The NBioBSP Java Library also uses NBioBSP.dll and provides higher level of interfaces. NBioBSP Java Library supports almost all NBioBSP functions.



Only valid for NBioBSP.dll version 4.781 or higher.

Only valid for JDK version 1.5.0\_20 or higher.

## Chapter 2 File list

### ■ NBioBSPJNI.jar

This is the java library of the eNBSP SDK.

### ■ NBioBSPJNI.dll

It implements all java native interfaces. It use with the windows platform.

### ■ NBioBSPJNI.so

It implements all java native interfaces. It use with the Linux platform.

### ■ Sample files

NBioAPI\_JavaDemo.java : Basic function demo application for java.

NBioAPI\_JavaUITest.java : User interface demo application for java.

NBioAPI\_JavaExport.java : Template data Export / Import demo application for java.

NBioAPI\_JavaIndexSearch.java : IndexSearch(1:N) demo application for java.

NBioAPI\_JavaRollDemo.java : Roll demo application for java

NBioAPI\_ANSIMatching.java : ANSI 378 Matching demo application for java.

NBioAPI\_WSQDemo.java : WSQ Image demo application for java.

### ■ Make files

Provides batch files that can be build on Windows.

Provides make files that can be build on Linux.

## Chapter 3 Java Class Library

### ■ NBioBSPJNI Class Initialize and Terminate

```
import com.nitgen.SDK.BSP.NBioBSPJNI; // Import NBioBSP Java class library
...
NBioBSPJNI bsp;                        // Declare NBioBSPJNI Class Object

bsp = new NBioBSPJNI();                // Create NBioBSPJNI Class Object
...

bsp.dispose();                        // Terminate NBioBSPJNI Class Object
bsp = null;
```

### ■ Device Control

The functions that control the fingerprint device can only be used after the device is first opened.

The following describes how to get a list of devices and how to open / close the device.

```
// Enumerate device
NBioBSPJNI.DEVICE_ENUM_INFO deviceEnumInfo;

deviceEnumInfo = bsp.new DEVICE_ENUM_INFO();
bsp.EnumerateDevice(deviceEnumInfo);

// Device Open
bsp.OpenDevice(deviceEnumInfo.DeviceInfo[0].NameID,
               deviceEnumInfo.DeviceInfo[0].Instance);

// Device Close
bsp.CloseDevice(deviceEnumInfo.DeviceInfo[0].NameID,
                deviceEnumInfo.DeviceInfo[0].Instance);
```

## ■ Fingerprint Enrollment

The Enroll or Capture method is used to enroll(register) fingerprint data.

**(Enroll method support only windows platform.)**

The Fingerprint Identification Record(FIR) data returned by calling enroll or capture method.

The application can't know the FIR structure, and refers to it only as a Handle.

The following describes how to get FIR handle and how to get FIR data.

```
NBioBSPJNI.FIR_HANDLE hSavedFIR;

hSavedFIR = bsp.new FIR_HANDLE();

bsp.Enroll(hSavedFIR, null);           // Enroll
bsp.Capture(hSavedFIR);               // Capture

// Get Text FIR
if (bsp.IsErrorOccured() == false) {
    NBioBSPJNI.FIR_TEXTENCODING textSavedFIR;

    textSavedFIR = bsp.new FIR_TEXTENCODING();
    bsp.GetTextFIRFromHandle(hSavedFIR, textSavedFIR);
}

// Get Binary FIR
if (bsp.IsErrorOccured() == false) {
    NBioBSPJNI.FIR fullSavedFIR;

    fullSavedFIR = bsp.new FIR();
    bsp.GetFIRFromHandle(hSavedFIR, fullSavedFIR);
}
```

The textFIR or BinaryFIR can be used in a file or a database.

## ■ Verification

The Verify method captures the fingerprint data from a device and compares that sample against the previously enrolled FIR.

If the FIR data contains payload data, Verify method can be retrieved after successful fingerprint verification.

```
NBioBSPJNI.INPUT_FIR inputFIR = bsp.new INPUT_FIR();  
Boolean bResult = new Boolean(false);  
NBioBSPJNI.FIR_PAYLOAD payload = bsp.new FIR_PAYLOAD();  
  
// Set stored textFIR data.  
inputFIR.SetTextFIR(textSavedFIR);  
  
bsp.Verify(inputFIR, bResult, payload);  
  
if (bsp.IsErrorOccured() == false) {  
    if (bResult)  
        labelStatus.setText("Verify OK - Payload: " + payload.GetText());  
    else  
        labelStatus.setText("Verify Failed");  
}
```

## ■ Client / Server Environment

In a client/Server environment, fingerprint capture is executed on the client system and the matching and storing of the fingerprint data takes place on server system.

For this reason, Verify method can't be used. Instead, VerifyMatch will be used.

```
NBioBSPJNI.INPUT_FIR inputFIR = bsp.new INPUT_FIR();
NBioBSPJNI.INPUT_FIR inputFIR2 = bsp.new INPUT_FIR();
Boolean bResult = new Boolean(false);
NBioBSPJNI.FIR_PAYLOAD payload = bsp.new FIR_PAYLOAD();

inputFIR.SetTextFIR(textSavedFIR);
inputFIR2. SetTextFIR(textCapturedFIR);

bsp.VerifyMatch(inputFIR, inputFIR2, bResult, payload);

if (bsp.IsErrorOccured() == false) {
    if (bResult)
        labelStatus.setText("Verify OK - Payload: " + payload.GetText());
    else
        labelStatus.setText("Verify Failed");
}
```

## Chapter 4 NBioBSPJNI Class

### ■ NBioBSPJNI.ERROR

This NBioBSPJNI.ERROR class provides eNBSP's error code.

Literal	Value	Mean
NBioAPIERROR_BASE_DEVICE	0	No Error
NBioAPIERROR_INVALID_HANDLE	0x01	Invalid handle
NBioAPIERROR_INVALID_POINTER	0x02	Invalid pointer
NBioAPIERROR_INVALID_TYPE	0x03	Invalid type
NBioAPIERROR_FUNCTION_FAIL	0x04	Function failed
NBioAPIERROR_STRUCTTYPE_NOT_MATCHED	0x05	Input structure type not support
NBioAPIERROR_ALREADY_PROCESSED	0x06	The FIR data processed already
NBioAPIERROR_EXTRACTION_OPEN_FAIL	0x07	Extraction engine open fail
NBioAPIERROR_VERIFICATION_OPEN_FAIL	0x08	Verification engine open fail
NBioAPIERROR_DATA_PROCESS_FAIL	0x09	Extraction fail
NBioAPIERROR_MUST_BE_PROCESSED_DATA	0x0a	The FIR data must be process
NBioAPIERROR_INTERNAL_CHECKSUM_FAIL	0x0b	Invalid FIR data
NBioAPIERROR_ENCRYPTED_DATA_ERROR	0x0c	FIR data encryption / decryption fail
NBioAPIERROR_UNKNOWN_FORMAT	0x0d	Unknown FIR data format
NBioAPIERROR_UNKNOWN_VERSION	0x0e	Unknown BSP version
NBioAPIERROR_VALIDITY_FAIL	0x0f	BSP validity fail
NBioAPIERROR_INIT_MAXFINGER	0x10	BSP Maxfinger option set fail
NBioAPIERROR_INIT_SAMPLESPERFINGER	0x11	BSP Samplesperfinger option set fail
NBioAPIERROR_INIT_ENROLLQUALITY	0x12	BSP EnrollQuality option set fail
NBioAPIERROR_INIT_VERIFYQUALITY	0x13	BSP VerifyQuality option set fail
NBioAPIERROR_INIT_IDENTIFYQUALITY	0x14	BSP IdentifyQuality option set fail
NBioAPIERROR_INIT_SECURITYLEVEL	0x15	BSP SecurityLevel option set fail
NBioAPIERROR_INVALID_MINSIZE	0x16	Template data size error
NBioAPIERROR_INVALID_TEMPLATE	0x17	Invalid template data
NBioAPIERROR_EXPIRED_VERSION	0x18	Expired BSP
NBioAPIERROR_INVALID_SAMPLESPERFINGER	0x19	Invalid Samplesperfinger option value
NBioAPIERROR_UNKNOWN_INPUTFORMAT	0x1a	Unknown INPUT_FIR type
NBioAPIERROR_INIT_ENROLLSECURITYLEVEL	0x1b	Invalid EnrollSecurityLevel option value
NBioAPIERROR_INIT_NECESSARYENROLLNUM	0x1c	Invalid NecessaryEnrollNum option value
NBioAPIERROR_OUT_OF_MEMORY	0x24	Out of memory

Literal	Value	Mean
NBioAPIERROR_DEVICE_OPEN_FAIL	0x101	Device open fail
NBioAPIERROR_INVALID_DEVICE_ID	0x102	Invalid device ID
NBioAPIERROR_WRONG_DEVICE_ID	0x103	Wrong device ID
NBioAPIERROR_DEVICE_ALREADY_OPENED	0x104	Already open device
NBioAPIERROR_DEVICE_NOT_OPENED	0x105	Device not opened
NBioAPIERROR_DEVICE_BRIGHTNESS	0x106	Invalid Brightness option value
NBioAPIERROR_DEVICE_CONTRAST	0x107	Invalid Contrast option value
NBioAPIERROR_DEVICE_GAIN	0x108	Invalid Gain option value
NBioAPIERROR_LOWVERSION_DRIVER	0x109	Low version driver
NBioAPIERROR_DEVICE_INIT_FAIL	0x10a	Device initialize fail
NBioAPIERROR_DEVICE_LOST_DEVICE	0x10b	Device disconnected.
NBioAPIERROR_DEVICE_DLL_LOAD_FAIL	0x10c	Device module load fail.
NBioAPIERROR_DEVICE_MAKE_INSTANCE_FAIL	0x10d	Device Instance creation fail.
NBioAPIERROR_DEVICE_DLL_GET_PROC_FAIL	0x10e	Device function load fail.
NBioAPIERROR_DEVICE_IO_CONTROL_FAIL	0x10f	Device IO fail.

Literal	Value	Mean
NBioAPIERROR_USER_CANCEL	0x201	Operation cancel from user
NBioAPIERROR_USER_BACK	0x202	Operation back from user
NBioAPIERROR_CAPTURE_TIMEOUT	0x203	Capture time out
NBioAPIERROR_CAPTURE_FAKE_SUSPICIOUS	0x204	Fake input occurred
NBioAPIERROR_ENROLL_EVENT_PLACE	0x205	Enroll method event
NBioAPIERROR_ENROLL_EVENT_HOLD	0x206	Enroll method event
NBioAPIERROR_ENROLL_EVENT_REMOVE	0x207	Enroll method event
NBioAPIERROR_ENROLL_EVENT_PLACE_AGAIN	0x208	Enroll method event
NBioAPIERROR_ENROLL_EVENT_EXTRACT	0x209	Enroll method event
NBioAPIERROR_ENROLL_EVENT_MATCH_FAILED	0x20a	Enroll method event

Literal	Value	Mean
NBioAPIERROR_INIT_PRESEARCHRATE	0x501	Invalid PreSearchRate option value
NBioAPIERROR_INDEXSEARCH_INIT_FAIL	0x502	IndexSearch engine initialize failed
NBioAPIERROR_INDEXSEARCH_SAVE_DB	0x503	IndexSearch engine save db failed
NBioAPIERROR_INDEXSEARCH_LOAD_DB	0x504	IndexSearch engine load db failed
NBioAPIERROR_INDEXSEARCH_UNKNOWN_VER	0x505	Unknown IndexSearch engine version
NBioAPIERROR_INDEXSEARCH_IDENTIFY_FAIL	0x506	IndexSearch engine identify failed
NBioAPIERROR_INDEXSEARCH_DUPLICATED_ID	0x507	IndexSearch engine ID duplicated
NBioAPIERROR_INDEXSEARCH_IDENTIFY_STOP	0x508	IndexSearch engine identify stop from user



## ■ NBioBSPJNI.FIR\_SECURITY\_LEVEL

This NBioBSPJNI.FIR\_SECURITY\_LEVEL class provides eNBSP's security level value.

Literal	Value	Mean
LOWEST	1	Security level value
LOWER	2	Security level value
LOW	3	Security level value
BELOW_NORMAL	4	Security level value
NORMAL	5	Security level value
ABOVE_NORMAL	6	Security level value
HIGH	7	Security level value
HIGHER	8	Security level value
HIGHEST	9	Security level value

## ■ NBioBSPJNI.FINGER\_ID

This NBioBSPJNI.FIR\_FINGER\_ID class provides eNBSP's finger id value.

Literal	Value	Mean
UNKNOWN	0	Unknown finger
RIGHT_THUMB	1	Right thumb finger
RIGHT_INDEX	2	Right index finger
RIGHT_MIDDLE	3	Right middle finger
RIGHT_RING	4	Right ring finger
RIGHT_LITTLE	5	Right little finger
LEFT_THUMB	6	Left thumb finger
LEFT_INDEX	7	Left index finger
LEFT_MIDDLE	8	Left middle finger
LEFT_RING	9	Left ring finger
LEFT_LITTLE	10	Left little finger

## ■ NBioBSPJNI.DEVICE\_ID

This NBioBSPJNI.DEVICE\_ID class provides eNBSP's device id value.

Literal	Value	Mean
NONE	0x0000	Unknown device ID value
AUTO	0x00ff	Auto detect value

## ■ NBioBSPJNI.DEVICE\_NAME

This NBioBSPJNI.DEVICE\_NAME class provides eNBSP's name id value.

Literal	Value	Mean
FDP02	0x01	Parallel type device
FDU01	0x02	USB type device
OSU02	0x03	Not used
FDU11	0x04	USB type device
FSC01	0x05	Not used
FDU03	0x06	USB mouse type device
FDU05	0x07	Flat / Roll type device
FDU08	0x08	eNBioScan-C1
NND_URU4KB	0xA1	UareU4000B device
NND_FPC6410	0xA2	FPC6410 device

## ■ NBioBSPJNI.FIR\_FORMAT

This NBioBSPJNI.FIR\_FORMAT class provides eNBSP's FIR format value.

Literal	Value	Mean
STANDARD	1	BSP standard format(SEED encryption)
NBAS	2	Not used
EXTENSION	3	BSP extension format(SEED encryption)
STANDARD_AES	4	BSP standard format(AES 128 encryption)
STANDARD_3DES	5	BSP standard format(3DES encryption)
STANDARD_256AES	6	BSP standard format(AES 256 encryption)

## ■ NBioBSPJNI.FIR\_DATA\_TYPE

This NBioBSPJNI.FIR\_DATA\_TYPE class provides eNBSP's FIR data type value.

Literal	Value	Mean
RAW	0x00	Raw Image data type
INTERMEDIATE	0x01	BSP internal data type
PROCESSED	0x02	Processed data type
ENCRYPTED	0x10	Encrypted data type
LINEPATTERN	0x20	Line pattern data type

## ■ NBioBSPJNI.FIR\_PURPOSE

This NBioBSPJNI.FIR\_PURPOSE class provides eNBSP's FIR purpose value.

Literal	Value	Mean
VERIFY	0x01	For Verification
IDENTIFY	0x02	For identify (currently not used)
ENROLL	0x03	For registration
ENROLL_FOR_VERIFICATION_ONLY	0x04	For verification (only)
ENROLL_FOR_IDENTIFICATION_ONLY	0x05	For identification (only)
AUDIT	0x06	For audit (currently not use)
UPDATE	0x07	For update (currently not use)

## ■ NBioBSPJNI.FIR\_FORM

This NBioBSPJNI.FIR\_FORM class provides eNBSP's FIR form value.

Literal	Value	Mean
HANDLE	0x02	FIR data is handle value
FULLFIR	0x03	FIR data is binary value
TEXTENCODE	0x04	FIR data is text encoded value

## ■ NBioBSPJNI.WINDOW\_STYLE

This NBioBSPJNI.WINDOW\_STYLE class provides eNBSP's UI option value.

Literal	Value	Mean
POPUP	0	Popup style on the parent window
INVISIBLE	1	This value can be used for Capture method. When calling the Capture method with this option, no dialog prompts.
CONTINUOS	2	This value can be used for Enroll method. This option can be used to make a customized enrollment wizard.
NO_FPIMG	65536	This value can be set when fingerprint image must not be displayed on eNBSP SDK UI. (Capture method only)
TOPMOST	131072	This value can be set when UI is needed to be the top most window
NO_WELCOME	262144	This value is to hide the welcome page on the Enroll method
NO_TOPMOST	524288	This value can be set when UI is not needed to be the top most window

## ■ NBioBSPJNI.INIT\_INFO\_0

This class contains information about the initial settings of the eNBS SDK

Type	Name	Description
int	MaxFingersForEnroll	A value indicating the number of fingers that can be enrolled Default value: 10
int	SamplesPerFinger	This read-only member specifies the number of samples used per each finger Default value: 2
int	DefaultTimeout	This value, in millisecond, specifies the waiting time to capture fingerprint images from the device Default value: 10000ms(10sec)
int	EnrollImageQuality	This value is used to set the threshold of image quality on enrollment Default value: 50
int	VerifyImageQuality	This value is used to set the threshold of image quality on verification Default value: 30
int	IdentifyImageQuality	This value is used to set the threshold of image quality on identification Default value: 50
int	SecurityLevel	The security level for fingerprint verification and identification Default value: <a href="#">FIR_SECURITY_LEVEL.NORMAL</a>

## ■ NBioBSPJNI.DEVICE\_INFO

This class contains information about a device

Type	Name	Description
int	ImageWidth	A value indicating the image width in pixel
int	ImageHeight	A value indicating the image height in pixel
int	Brightness	A value indicating the brightness of the device
int	Contrast	A value indicating the contrast of the device
int	Gain	A value indicating the gain of the device

## ■ NBioBSPJNI.DEVICE\_INFO\_EX

This class contains detail information about a device

Type	Name	Description
short	DeviceID	A value indicating the device ID( <a href="#">NBioBSPJNI.DEVICE_NAME</a> value)
short	NameID	Same the DeviceID
short	Instance	A value indicating the device instance
String	Name	A value indication the device name string
String	Description	A value indication the device description string
String	Dll	A value indication the dll file name of device driver
String	Sys	A value indication the sys file name of device driver
int	AutoOn	A value indication the auto on supported device. If the device can support the auto on function, this value set to 1
int	Brightness	A value indicating the image brightness of the device
int	Contrast	A value indicating the image contrast of the device
int	Gain	A value indicating the image gain of the device

## ■ NBioBSPJNI.DEVICE\_ENUM\_INFO

This class contains information about device attached to the system

Type	Name	Description
int	DeviceCount	A value indicating the number of device attached to the system
<a href="#">DEVICE_INFO_EX[]</a>	DeviceInfo	A value indicating the information of the device

## ■ NBioBSPJNI.MATCH\_OPTION

This class contains information about matching configuration

Type	Name	Description
byte[]	NoMatchFinger	This value indicates a set of exclusion of target to be matched 0: Match 1 = Sample1 not match 2 = Sample2 not match 3 = Not match
Int[]	Reserved	Reserved for future use

## ■ NBioBSPJNI.FIR\_HANDLE

This class contains handle about FIR data

Type	Name	Description
int	Handle	A value indicating the handle of the FIR data



## ■ NBioBSPJNI.FIR\_HEADER

This class contains information about the FIR data

Type	Name	Description
short	Version	A value indicating the version of FIR data
short	DataType	A value indicating the FIR data type
short	Purpose	A value indicating the purpose of using the FIR data
short	Quality	A value indicating the quality of a fingerprint image
int	Reserved	Reserved for future use

## ■ NBioBSPJNI.FIR

This class contains information about the FIR

Type	Name	Description
int	Format	A value indicating the FIR data format
<a href="#">FIR_HEADER</a>	Header	Specifies a <a href="#">FIR_HEADER</a> class that contain information about the FIR header
byte[]	Data	Data to a byte array that specifies actual FIR data

## ■ NBioBSPJNI.FIR\_TEXTENCODE

This class contains information about the text encoded FIR data

Type	Name	Description
String	TextFIR	Data to a String that text encoded FIR data

## ■ NBioBSPJNI.INPUT\_FIR

This class used to input a FIR to the NBioBSPJNI methods

```
void SetFIRHandle(FIR_HANDLE handle)
```

Parameters

[FIR\\_HANDLE](#) handle  
[in] The handle of the FIR data

---

Return Value

---

This method changes the FIR data of the INPUT\_FIR class

---

```
void SetFullFIR(FIR fullFIR)
```

Parameters

[FIR](#) fullFIR  
[in] The binary type of the FIR data

---

Return Value

---

This method changes the FIR data of the INPUT\_FIR class

---

```
void SetTextFIR(FIR_TEXTENCODING textFIR)
```

Parameters

[FIR\\_TEXTENCODING](#) textFIR  
[in] The string type of the FIR data

---

Return Value

---

This method changes the FIR data of the INPUT\_FIR class

---

## ■ NBioBSPJNI.FIR\_PAYLOAD

This class contains information about payload data

```
void SetData(byte[] data)
```

Parameters

byte[] data  
[in] payload data

Return Value

This method changes the payload data of the FIR\_PAYLOAD class

```
void SetText(String text)
```

Parameters

String text  
[in] payload data

Return Value

This method changes the payload data of the FIR\_PAYLOAD class

```
byte[] GetData()
```

Parameters

Return Value

byte[]  
The byte array type of payload data

This method gets the payload data of the FIR\_PAYLOAD class

```
String GetText()
```

Parameters

Return Value

String  
The string type of payload data

This method gets the payload data of the FIR\_PAYLOAD class

## NBIOBSPJNI.WINDOW\_OPTION

This class contains UI information about eNBSP SDK

Type	Name	Description
int	WindowStyle	This value must be one of the <a href="#">WINDOW_STYLE</a> values
Component	ParentWnd	A component to the parent Before using this, set the JrePath member of WINDOW_OPTION class
Component	FingerWnd	Specifies where the fingerprint image to be shown Before using this, set the JrePath member of WINDOW_OPTION class This component must be Canvas object of the Java.AWT
String	JrePath	The path to the JRE directory
String	CaptionMsg	Message text displayed on the caption of the message box that prompts when canceling the enrollment dialog
String	CancelMsg	Message text displayed on the message box that prompts when canceling the enrollment dialog
int	FPForeColorR	Red color for the fingerprint images
int	FPForeColorG	Green color for the fingerprint images
int	FPForeColorB	Blue color for the fingerprint images
int	FPBackColorR	Red color for the fingerprint background
int	FPBackColorG	Green color for the fingerprint background
int	FPBackColorB	Blue color for the fingerprint background
int	DisableFingerForEnroll0	Disable enrollment for the right thumb fingerprint
int	DisableFingerForEnroll1	Disable enrollment for the right index fingerprint
int	DisableFingerForEnroll2	Disable enrollment for the right middle fingerprint
int	DisableFingerForEnroll3	Disable enrollment for the right ring fingerprint
int	DisableFingerForEnroll4	Disable enrollment for the right little fingerprint
int	DisableFingerForEnroll5	Disable enrollment for the left thumb fingerprint
int	DisableFingerForEnroll6	Disable enrollment for the left index fingerprint
int	DisableFingerForEnroll7	Disable enrollment for the left middle fingerprint
int	DisableFingerForEnroll8	Disable enrollment for the left ring fingerprint
int	DisableFingerForEnroll9	Disable enrollment for the left little fingerprint



# NBioBSPJNI

This class contains basic method about eNBSP SDK

int GetErrorCode()

Parameters

Return Value

int

The value is the NBioBSPJNI's last-error code

Retrieves the NBioBSPJNI's last-error code value

Refer [NBioBSPJNI.ERROR](#)

Boolean IsErrorOccured()

Parameters

Return Value

Boolean

Returns a Boolean value indicating whether a NBioBSPJNI is an error

Returns a Boolean value indicating whether a NBioBSPJNI is an error

void dispose()

Parameters

Return Value

Frees the loaded system resource and memory

int SetSkinResource(String szResPath)

Parameters

String szResPath

[in] The path to the Skin Module

Return Value

int

The value is the NBioBSPJNI's last-error code

Use this method to set the resource module that determines where the default resources of the eNBSP SDK are loaded

## String GetVersion()

### Parameters

#### Return Value

String

The return value includes the major and minor version string of eNBSP SDK

Retrieves the version string of eNBSP SDK

## int GetInitInfo(INIT\_INFO\_0 initInfo0)

### Parameters

[INIT\\_INFO\\_0](#) initInfo0

[out] Receives the eNBSP SDK initialization information

#### Return Value

int

The value is the NBioBSPJNI's last-error code

Retrieves initialization information about the eNBSP SDK

## int SetInitInfo(INIT\_INFO\_0 initInfo0)

### Parameters

[INIT\\_INFO\\_0](#) initInfo0

[in] eNBSP SDK Initialization information

#### Return Value

int

The value is the NBioBSPJNI's last-error code

Sets a new initialization information for eNBSP SDK

## int EnumerateDevice(DEVICE\_ENUM\_INFO deviceInfo)

### Parameters

[DEVICE\\_ENUM\\_INFO](#) deviceInfo

[in] Device list information

#### Return Value

int

The value is the NBioBSPJNI's last-error code

This method enumerates all devices on the system

```
int OpenDevice()  
int OpenDevice(short nDeviceName, short nInstance)
```

Parameters

short nDeviceName  
    [in] Device Name (Refer [DEVICE\\_INFO\\_EX](#))  
short nInstance  
    [in] Device Instance ID (Refer [DEVICE\\_INFO\\_EX](#))

---

Return Value

int  
    The value is the NBioBSPJNI's last-error code

---

This method is to initialize the last opened device

---

If the nDeviceName and nInstance parameter specifies, this method is to initialize it

---

```
int CloseDevice()  
int CloseDevice(short nOpenedDeviceID)  
int CloseDevice(short nDeviceName, short nInstance)
```

Parameters

short nOpenedDeviceID  
    [in] Opened device ID  
short nDeviceName  
    [in] Device Name (Refer [DEVICE\\_INFO\\_EX](#))  
short nInstance  
    [in] Device Instance ID (Refer [DEVICE\\_INFO\\_EX](#))

---

Return Value

int  
    The value is the NBioBSPJNI's last-error code

---

This method is to close the last opened device

---

If the nOpenedDeviceID parameter specifies, this method is to close it

---

If the nDeviceName and nInstance parameter specifies, this method is to close it

---

```
int GetDeviceInfo(DEVICE_INFO deviceInfo)
int GetDeviceInfo(short nDeviceName, short nInstance, DEVICE_INFO deviceInfo)
```

Parameters

short nDeviceName  
[in] Device Name (Refer [DEVICE\\_INFO\\_EX](#))

short nInstance  
[in] Device Instance ID (Refer [DEVICE\\_INFO\\_EX](#))

[DEVICE\\_INFO](#) deviceInfo  
[in] Receives the device information

---

Return Value

int  
The value is the NBioBSPJNI's last-error code

---

This method retrieves information about the opened device

---

If the nDeviceName and nInstance parameter specifies, retrieves information about it

---

```
int SetDeviceInfo(DEVICE_INFO deviceInfo)
int SetDeviceInfo(short nDeviceName, short nInstance, DEVICE_INFO deviceInfo)
```

Parameters

short nDeviceName  
[in] Device Name (Refer [DEVICE\\_INFO\\_EX](#))

short nInstance  
[in] Device Instance ID (Refer [DEVICE\\_INFO\\_EX](#))

[DEVICE\\_INFO](#) deviceInfo  
[in] [DEVICE\\_INFO](#) class value

---

Return Value

int  
The value is the NBioBSPJNI's last-error code

---

This method sets information about the opened device

---

If the nDeviceName and nInstance parameter specifies, sets information about it

---



```
int AdjustDevice()  
int AdjustDevice(WINDOW_OPTION winOption)
```

Parameters

[WINDOW\\_OPTION](#) winOption

[in, optional] UI Option Class value, this parameter can be null

---

Return Value

int

The value is the NBioBSPJNI's last-error code

---

This method is used to configure the brightness of the device

**This method support windows platform only**

---

```
short GetOpenedDeviceID()
```

Parameters

---

Return Value

short

Returns opened device ID

---

Retrieves opened device ID

---

```
int GetFIRFromHandle(FIR_HANDLE hFIR, FIR fullFIR)  
int GetFIRFromHandle(FIR_HANDLE hFIR, FIR fullFIR, int Format)
```

Parameters

[FIR\\_HANDLE](#) hFIR

[in] FIR data handle class value

[FIR](#) fullFIR

[out] FIR data class value

int Format

[in] FIR data [format](#) value

---

Return Value

int

The value is the NBioBSPJNI's last-error code

---

Retrieves FIR data(Binary data) from a relative handle

If the Format parameter specifies, this method extracts FIR data in specified format

---

```
int GetTextFIRFromHandle(FIR_HANDLE hFIR, FIR_TEXTENCODING textFIR)
int GetTextFIRFromHandle(FIR_HANDLE hFIR, FIR_TEXTENCODING textFIR, int Format)
```

#### Parameters

[FIR\\_HANDLE](#) hFIR  
[in] FIR data handle class value

[FIR\\_TEXTENCODING](#) textFIR  
[out] Text FIR data class value

int Format  
[in] FIR data [format](#) value

#### Return Value

int  
The value is the NBioBSPJNI's last-error code

Retrieves FIR data(Text encoded data) from a relative handle

If the Format parameter specifies, this method extracts FIR data in specified format

```
int Enroll(FIR_HANDLE hEnrolledFIR, FIR_PAYLOAD payload)
int Enroll(INPUT_FIR storedFIR, FIR_HANDLE hEnrolledFIR, FIR_PAYLOAD payload, int nTimeout,
           FIR_HANDLE hAudit, WINDOW_OPTION winOption)
```

#### Parameters

[INPUT\\_FIR](#) storedFIR  
[in, optional] [INPUT\\_FIR](#) class value, this parameter can be null

[FIR\\_HANDLE](#) hEnrolledFIR  
[out] FIR data handle class value

[FIR\\_PAYLOAD](#) payload  
[in, optional] payload data class value, this parameter can be null

int nTimeout  
[in] capture time out value

[FIR\\_HANDLE](#) hAudit  
[out, optional] FIR data handle class value(Image data), this parameter can be null

[WINDOW\\_OPTION](#) winOption  
[in, optional] UI option class value, this parameter can be null

#### Return Value

int  
The value is the NBioBSPJNI's last-error code

This method captures fingerprint data from the attached device to create a hEnrolledFIR for the purpose of enrollment

If the storedFIR parameter specifies, the FIR to adapted

If the payload parameter specifies, it be wrapped inside the newly created template

If the hAudit parameter specifies, returns handle of fingerprint image

If the winOption parameter specifies, this method runs in specified option

**This method support windows platform only**

```
int Capture(FIR_HANDLE hFIR)
int Capture(int purpose, FIR_HANDLE hFIR, int nTimeout,
            FIR_HANDLE hAudit, WINDOW_OPTION winOption)
```

---

Parameters

int [purpose](#)

[in] The purpose of using the FIR data

[FIR\\_HANDLE](#) hFIR

[out] FIR data handle class value

int nTimeout

[in] capture time out value

[FIR\\_HANDLE](#) hAudit

[out, optional] FIR data handle class value(Image data), this parameter can be null

[WINDOW\\_OPTION](#) winOption

[in, optional] UI option class value, this parameter can be null

---

Return Value

int

The value is the NBioBSPJNI's last-error code

---

This method captures samples for the purpose specified from the attached device

If the hAudit parameter specifies, returns handle of fingerprint image

If the winOption parameter specifies, this method runs in specified option

**This method not support UI for Linux platform**

---

```
int RollCapture(FIR_HANDLE hFIR)
int RollCapture(int purpose, FIR_HANDLE hFIR, int nTimeout,
                FIR_HANDLE hAudit, WINDOW_OPTION winOption)
```

---

#### Parameters

int [purpose](#)

[in] The purpose of using the FIR data

[FIR\\_HANDLE](#) hFIR

[out] FIR data handle class value

int nTimeout

[in] capture time out value

[FIR\\_HANDLE](#) hAudit

[out, optional] FIR data handle class value(Image data), this parameter can be null

[WINDOW\\_OPTION](#) winOption

[in, optional] UI option class value, this parameter can be null

---

#### Return Value

int

The value is the NBioBSPJNI's last-error code

---

This method captures samples for the purpose specified from the attached roll device

If the hAudit parameter specifies, returns handle of fingerprint image

If the winOption parameter specifies, this method runs in specified option

**This method not support UI for Linux platform**

---

```
int Process(INPUT_FIR capturedFIR, FIR_HANDLE hProcessedFIR)
```

#### Parameters

[INPUT\\_FIR](#) capturedFIR  
[in] [INPUT\\_FIR](#) class value  
[FIR\\_HANDLE](#) hProcessedFIR  
[out] FIR data handle class value

---

#### Return Value

int  
The value is the NBioBSPJNI's last-error code

---

This method processes the intermediate data captured via a call to Capture for the purpose of either verification of identification.

A call to this method is not necessary because the Capture and Enroll methods perform processing operation. It can be used when processing the Finger Image data included form audit data

---

```
int CreateTemplate(INPUT_FIR capturedFIR, INPUT_FIR storedFIR,  
                  FIR_HANDLE hNewFIR, FIR_PAYLOAD payload)
```

#### Parameters

[INPUT\\_FIR](#) capturedFIR  
[in] [INPUT\\_FIR](#) class value  
[INPUT\\_FIR](#) storedFIR  
[in, optional] [INPUT\\_FIR](#) class value, this parameter can be null  
[FIR\\_HANDLE](#) hNewFIR  
[out] FIR data handle class value  
[FIR\\_PAYLOAD](#) payload  
[in, optional] payload data class value, this parameter can be null

---

#### Return Value

int  
The value is the NBioBSPJNI's last-error code

---

This method takes a FIR containing raw fingerprint data for the purpose of creating a new enrollment template

A new FIR is constructed from the CapturedFIR, and (optionally) it may perform an adaptation based on an existing StoredTFIR

If the StoredFIR contains a payload, the payload is not copied into the hNewFIR. If the hNewFIR needs a payload, then that Payload must be presented as an argument to the method

---

```
int Verify(INPUT_FIR storedFIR, Boolean bResult, FIR_PAYLOAD payload)
int Verify(INPUT_FIR storedFIR, Boolean bResult, FIR_PAYLOAD payload,
           int nTimeout, FIR_HANDLE hAudit, WINDOW_OPTION winOption)
```

#### Parameters

[INPUT\\_FIR](#) storedFIR

[in] [INPUT\\_FIR](#) class value

Boolean bResult

[out] Matching result

[FIR\\_PAYLOAD](#) payload

[in, optional] payload data class value, this parameter can be null

int nTimeout

[in] capture time out value

[FIR\\_HANDLE](#) hAudit

[out, optional] FIR data handle class value(Image data), this parameter can be null

[WINDOW\\_OPTION](#) winOption

[in, optional] UI option class value, this parameter can be null

---

#### Return Value

int

The value is the NBioBSPJNI's last-error code

---

This method captures fingerprint data from the attached device, and compares it against the storedFIR

If the storedFIR contains a payload, it is returned

If the hAudit parameter specifies, returns handle of fingerprint image

If the winOption parameter specifies, this method runs in specified option

This method support windows platform only

---

```
int VerifyMatch(INPUT_FIR capturedFIR, INPUT_FIR storedFIR, Boolean bResult, FIR_PAYLOAD payload)
int VerifyMatch(INPUT_FIR capturedFIR, INPUT_FIR storedFIR, Boolean bResult,
                FIR_PAYLOAD payload, MATCH_OPTION matchOption)
```

---

Parameters

[INPUT\\_FIR](#) capturedFIR

[in] [INPUT\\_FIR](#) class value

[INPUT\\_FIR](#) storedFIR

[in] [INPUT\\_FIR](#) class value

Boolean bResult

[out] Matching result

[FIR\\_PAYLOAD](#) payload

[in, optional] payload data class value, this parameter can be null

[MATCH\\_OPTION](#) matchOption

[in, optional] Matching option data class value, this parameter can be null

---

Return Value

int

The value is the NBioBSPJNI's last-error code

---

This method performs a verification (1-to-1) match between FIRs, the capturedFIR and the storedFIR

If the storedFIR contains a payload, it is returned

---

```
int CheckFinger(Boolean bFingerExist)
```

---

Parameters

Boolean bFingerExist

[out] A finger is placed on the fingerprint sensor or not

---

Return Value

int

The value is the NBioBSPJNI's last-error code

---

This function is to check if a finger is placed on the fingerprint sensor

Only valid for USB fingerprint devices and device driver version 4.1.0.1 or higher

**This method support windows platform only**

---

```
int GetNFIQInfoFromHandle(FIR_HANDLE auditData, NFIQINFO qualityInfo)
static int GetNFIQFromNBioBSP(Export.AUDIT AuditData, NFIQINFO Info)
static int GetNFIQFromRaw(byte[] RawImage, int ImgW, int ImgH, Integer NFIQ)
```

#### Parameters

[FIR\\_HANDLE](#) handle  
 [in] Assigns [FIR\\_HANDLE](#) value  
[Export.AUDIT](#) AuditData  
 [in] Image information  
[NFIQINFO](#) qualityInfo  
 [out] NFIQINFO result information  
 Integer NFIQ  
 [out] NFIQ result value

#### Return Value

int  
 Returns Error of NBioBSPJNI

Gets NFIQ value via image information.

Prints NFIQ value on a scale of 1(lowest quality) to 5(highest quality).

```
static int ExportRawToISOV1(NBioBSPJNI.Export.AUDIT AuditData, ISOBUFFER ISOBuf, boolean
blsRollDevice, byte CompressMode)
```

#### Parameters

[Export.AUDIT](#) AuditData  
 [in] Image information  
[ISOBUFFER](#) ISOBuf  
 [out] Class to store ISO data  
 Boolean blsRollDevice  
 [in] Specifies whether input is Roll Image (true: Roll Image, false: Flat Image)  
 byte CompressMode  
 [in] Selects compression mode (Refer to [COMPRESS\\_MODE](#))

#### Return Value

int  
 returns Error of NBioBSPJNI

Converts from image information to ISO 19794-4 data.



```
static int ExportRawToISOV2(byte FingerID, short ImgW, short ImgH, byte[] RawBuf, ISOBUFFER ISOBuf,
boolean blsRollDevice, byte CompressMode)
```

#### Parameters

byte FingerID  
[in] Finger identification number (Refer to [FINGER\\_ID](#))

short ImgW  
[in] Width of Raw image

short ImgH  
[in] Height of Raw image

byte[] RawBuf  
[in] Byte array which Raw image data is stored in.

[ISOBUFFER](#) ISOBuf  
[out] Class to store ISO data

Boolean blsRollDevice  
[in] Specifies whether input is Roll Image (true: Roll Image, false: Flat Image)

byte CompressMode  
[in] Selects compression mode (Refer to [COMPRESS\\_MODE](#))

#### Return Value

int  
returns Error of NBioBSPJNI

Converts from image information to ISO 19794-4 data.

```
static int ImportISOToRaw(ISOBUFFER ISOBuf, NIMPORTRAWSET RawSet)
```

#### Parameters

[ISOBUFFER](#) ISOBuf  
[in] Class which ISO data is stored in.

[NIMPORTRAWSET](#) RawSet  
[out] Raw Images stored in ISO data are returned.

#### Return Value

int  
returns Error of NBioBSPJNI

Extracts Raw Image information from ISO 19794-4 data.

## ■ NBioBSPJNI.EXPORT\_MINCONV\_TYPE

This NBioBSPJNI.EXPORT\_MINCONV\_TYPE class provides eNBSP's Template type value.

Literal	Value	Mean
FDP	0	FDP device template format
FDU	1	FDU device template format
FDA	2	FDA device template format
OLD_FDA	3	FDA device template format
FDAC	4	Access control device template format
FIM10_HV	5	FIM10-HV device template format
FIM10_LV	6	FIM10-LV device template format
FIM01_HV	7	FIM01-HV device template format
FIM01_HD	8	FIM01-HD device template format
FELICA	9	FELICA device template format
EXTENSION	10	Extension type template format(1024bytes)
TEMPLATESIZE_32	11	32 size template format
TEMPLATESIZE_48	12	48 size template format
TEMPLATESIZE_64	13	64 size template format
TEMPLATESIZE_80	14	80 size template format
TEMPLATESIZE_96	15	96 size template format
TEMPLATESIZE_112	1	112 size template format
TEMPLATESIZE_128	17	128 size template format
TEMPLATESIZE_144	18	144 size template format
TEMPLATESIZE_160	19	160 size template format
TEMPLATESIZE_176	20	176 size template format
TEMPLATESIZE_192	21	192 size template format
TEMPLATESIZE_208	22	208 size template format

Literal	Value	Mean
TEMPLATESIZE_224	23	224 size template format
TEMPLATESIZE_240	24	240 size template format
TEMPLATESIZE_256	25	256 size template format
TEMPLATESIZE_272	26	272 size template format
TEMPLATESIZE_288	27	288 size template format
TEMPLATESIZE_304	28	304 size template format
TEMPLATESIZE_320	29	320 size template format
TEMPLATESIZE_336	30	336 size template format
TEMPLATESIZE_352	31	352 size template format
TEMPLATESIZE_368	32	368 size template format
TEMPLATESIZE_384	33	384 size template format
TEMPLATESIZE_400	34	400 size template format
ANSI	35	ANSI template format
ISO	36	ISO template format

## ■ NBioBSPJNI.Export.TEMPLATE\_DATA

This class contains Template data

Type	Name	Description
byte[]	Data	A value indicating the template data

## ■ NBioBSPJNI.Export.FINGER\_DATA

This class contains information about one fingerprint

Type	Name	Description
byte	FingerID	<a href="#">Finger ID</a> value
<a href="#">Export.TEMPLATE_DATA[]</a>	Template	A value indicating the template data array

## ■ NBioBSPJNI.Export.DATA

This class contains information about fingerprint

Type	Name	Description
byte	EncryptType	A value indicating the encryption type
byte	FingerNum	A value indicating the count of finger
byte	DefaultFingerID	A value indicating the default finger ID
byte	SamplesPerFinger	A value indicating the sample count
<a href="#">Export.FINGER_DATA[]</a>	FingerData	A value indicating the fingerprint Data array

## ■ NBioBSPJNI.Export.AUDIT

This class contains information about fingerprint image

Type	Name	Description
byte	FingerNum	A value indicating the count of fingerprint image
byte	SamplesPerFinger	A value indicating the sample count
int	ImageWidth	A value indicating the image width
int	ImageHeight	A value indicating the image height
<a href="#">Export.FINGER_DATA[]</a>	FingerData	A value indicating the fingerprint Data array

## ■ NBioBSPJNI.Export

This class contains export method about eNBSP SDK

```
int ExportFIR(INPUT_FIR inputFIR, Export.DATA exportData, int exportType)
```

### Parameters

[INPUT\\_FIR](#) inputFIR  
[in] The FIR data to be converted  
[Export.DATA](#) exportData  
[out] Receives the data converted from the inputFIR  
int exportType  
[in] A value indicating the [type](#) of exportation

---

### Return Value

int  
The value is the NBioBSPJNI's last-error code

---

This method is to convert the FIR data to other data format

In case of re-use the exportData, call System.gc() after set the null(Refer to Sample)

---

```
int ImportFIR(byte[] Template, int nLen, int type, FIR_HANDLE hFIR)
int ImportFIR(byte[] Template, int nLen, int type, int purpose, FIR_HANDLE hFIR)
int ImportFIR(Export.DATA exportData, FIR_HANDLE hFIR)
int ImportFIR(Export.DATA exportData, int purpose, int dataType, FIR_HANDLE hFIR)
```

### Parameters

byte[] Template  
[in] The exportation data to be converted  
int nLen  
[in] The length of Template  
int type  
[in] A value indicating the [type](#) of exportation  
int purpose  
[in] The [purpose](#) of using the FIR data  
Export.DATA exportData  
[in] The exportation data to be converted  
int dataType  
[in] Must set FIR\_DATA\_TYPE.PROCESSED  
FIR\_HANDLE  
[out] Receives the handle converted from the Template or exportData

---

### Return Value

int  
The value is the NBioBSPJNI's last-error code

---

This method is to convert the other type data to the FIR format in a handle of FIR

---

```
int ExportAudit(INPUT_FIR inputFIR, Export.AUDIT exportAudit)
```

Parameters

[INPUT\\_FIR](#) inputFIR

[in] Specifying the FIR to be converted to image data

[Export.AUDIT](#) exportAudit

[out] receives the image data converted from the FIR

---

Return Value

int

The value is the NBioBSPJNI's last-error code

---

This method is to convert the FIR data to an image data(RAW format)

In case of re-use the exportAudit, call System.gc() after set the null(Refer to Sample)

---

```
int ImportAudit(Export.AUDIT exportAudit, FIR_HANDLE hFIR)
```

Parameters

[Export.AUDIT](#) exportAudit

[in] Data to be converted

[FIR\\_HANDLE](#) hFIR

[out] Receives the FIR handle

---

Return Value

int

The value is the NBioBSPJNI's last-error code

---

This method is to convert raw image data into a FIR format

---

```
int ImportBioAPIOpaqueToFIR(byte[] bioAPIOpaqueData, FIR_HANDLE hFIR)
```

Parameters

byte[] bioAPIOpaqueData

[in] BioAPI Data array

[FIR\\_HANDLE](#) hFIR

[out] Receives the FIR handle

---

Return Value

int

The value is the NBioBSPJNI's last-error code

---

This method is to convert BioAPI data into a FIR format

---

```
int ConvertRawToWsq(byte[] rawImage, int nWidth, int nHeight, Export.TEMPLATE_DATA wsqImage, float fQuality)
```

Parameters

byte[] rawImage  
    [in] Fingerprint image array  
int nWidth  
    [in] Fingerprint image width  
int nHeight  
    [in] Fingerprint image height  
[Export.TEMPLATE\\_DATA](#) wsqImage  
    [out] WSQ image  
Float fQuality  
    [in] WSQ rate

---

Return Value

int  
    The value is the NBioBSPJNI's last-error code

---

This method is to convert fingerprint image into a WSQ Image.

---

```
int ConvertWsqToRaw(byte[] wsqImage, int nWsqLen, Export.AUDIT exportAudit)
```

Parameters

byte[] wsqImage  
    [in] WSQ image  
int nWsqLen  
    [in] WSQ image length  
[Export.AUDIT](#) exportAudit  
    [out] fingerprint image

---

Return Value

int  
    The value is the NBioBSPJNI's last-error code

---

This method is to convert WSQ image into a fingerprint Image.

---



## ■ NBioBSPJNI.IndexSearch.INIT\_INFO

This class contains initialization information about NBioBSPJNI.IndexSearch

Type	Name	Description
int	PresearchRate	Reserved for future use

## ■ NBioBSPJNI.IndexSearch.SAMPLE\_INFO

This class contains Sample information about NBioBSPJNI.IndexSearch

Type	Name	Description
int	ID	A value indicating the user ID
byte[]	SampleCount	A value indicating the sample count

## ■ NBioBSPJNI.IndexSearch.FP\_INFO

This class contains Fingerprint information about NBioBSPJNI.IndexSearch

Type	Name	Description
int	ID	A value indicating the user ID
byte	FingerID	A value indicating the finger ID
byte	SampleNumber	A value indicating the sample count

# ■ NBioBSPJNI.IndexSearch

This class contains IndesSearch method about eNBSP SDK

void dispose()

Parameters

Return Value

Frees the loaded system resource and memory

int GetInitInfo(IndexSearch.INIT\_INFO initInfo)

Parameters

[IndexSearch.INIT\\_INFO](#) initInfo

[out] Receives the IndexSearch engine initialization information

Return Value

int

The value is the NBioBSPJNI's last-error code

Retrieves initialization information about the IndexSearch engine

int SetInitInfo(IndexSearch.INIT\_INFO initInfo)

Parameters

[IndexSearch.INIT\\_INFO](#) initInfo

[in] IndexSearch engine Initialization information

Return Value

int

The value is the NBioBSPJNI's last-error code

Sets a new initialization information for IndexSearch engine

```
int AddFIR(INPUT_FIR inputFIR, int userID, IndexSearch.SAMPLE_INFO sampleInfo)
```

#### Parameters

[INPUT\\_FIR](#) inputFIR

[in] The data to be registered

int userID

[in] A user ID number to be registered

[IndexSearch.SAMPLE\\_INFO](#) sampleInfo

[out] Receives some information of a registered template

---

#### Return Value

int

The value is the NBioBSPJNI's last-error code

---

This method is to register a fingerprint template data, along with a user ID, into the fingerprint DB on memory

---

```
int Identify(INPUT_FIR inputFIR, int secuLevel, IndexSearch.FP_INFO fpInfo)
```

```
int Identify(INPUT_FIR inputFIR, int secuLevel, IndexSearch.FP_INFO fpInfo, int nTimeOut)
```

#### Parameters

[INPUT\\_FIR](#) inputFIR

[in] [INPUT\\_FIR](#) class value

int secuLevel

[in] Indicates the security level set for fingerprint recognition

[IndexSearch.FP\\_INFO](#) fpInfo

[out] Receives template information

Int nTimeOut

[in] Identify time out value

---

#### Return Value

int

The value is the NBioBSPJNI's last-error code

---

This method is to perform identification and determine if the same fingerprint exists within the finger DB

---

```
int RemoveData(IndexSearch.FP_INFO fpInfo)
```

Parameters

[IndexSearch.FP\\_INFO](#) fpInfo

[in] [IndexSearch.FP\\_INFO](#) class value

Return Value

int

The value is the NBioBSPJNI's last-error code

This method is to remove a template data from the fingerprint DB on memory

```
int RemoveUser(int userID)
```

Parameters

int userID

[in] User ID value

Return Value

int

The value is the NBioBSPJNI's last-error code

This method is to remove all template data of a user from the fingerprint DB on memory

```
int ClearDB()
```

Parameters

Return Value

int

The value is the NBioBSPJNI's last-error code

This method is to delete all template data from the fingerprint DB on memory

```
int SaveDB(String szFilePath)
```

Parameters

String szFilePath

[in] The path to save file

Return Value

int

The value is the NBioBSPJNI's last-error code

This method is to backup the fingerprint DB, in memory, into a file

**int LoadDB(String szFilePath)**

Parameters

String szFilePath

[in] The path to load file

Return Value

int

The value is the NBioBSPJNI's last-error code

This method is to load the fingerprint DB file onto memory

**int GetDBCount(Integer nCount)**

Parameters

Integer nCount

[out] Receives the count of template data stored in the fingerprint DB

Return Value

int

The value is the NBioBSPJNI's last-error code

This method is to retrieve the count of template data in the fingerprint DB

**int CheckDataExist(IndexSearch.FP\_INFO fpInfo, Boolean bExist)**

Parameters

[IndexSearch.FP\\_INFO](#) fpInfo

[in] [IndexSearch.FP\\_INFO](#) class value

Boolean bExist

[out] receives the flag of existence

Return Value

int

The value is the NBioBSPJNI's last-error code

This method is to check if a specific template data exists in the fingerprint DB

## ■ NBioBSPJNI.NFIQINFO

Class to store NIST Fingerprint Image Quality value

Type	Name	Description
byte[][]	Quality	This is two-dimensional array of 11 rows and 2 columns to store NFIQ value. The first one specifies Finger ID and the second one means Sample Num.

## ■ NBioBSPJNI.COMPRESS\_MODE

Definition of compression mode

Type	Name	Description
byte	NONE	Fixed value 0. No compression.
byte	WSQ	Fixed value 1. Compression with WSQ format performed.

## ■ NBioBSPJNI.NIMPORTRAW

Definition of Raw Image data

Type	Name	Description
byte	FingerID	Specifies finger identification number(Refer to <a href="#">FINGER_ID</a> )
short	ImgWidth	Width of Image data
short	ImgHeight	Height of Image data
byte[]	Data	Byte array which Raw image data is stored in.

## ■ NBioBSPJNI.NIMPORTRAWSET

Definition of Raw Image data set

Type	Name	Description
byte	Count	The number of Raw Image data
<a href="#">NIMPORTRAW[]</a>	RawData	An array which Raw Image data is stored in.

## ■ NBioBSPJNI.ISOBUFFER

Definition of ISO data

Type	Name	Description
byte[]	Data	A byte array which ISO data is stored in.