eNBSP - *NBioBSP*

NITGEN&COMPANY Biometric Service Provider SDK

Programmer's Manual VB

SDK version 4.8x

Serial Number:

Specifications subject to change without notice.

목 차

제 1 장 개요

eNBSP (이하 NBioBSP 로 표시)는 니트젠 지문 인식기를 이용하여 어플리케이션을 개발할 수 있도록 만든 High-Level SDK 로써 니트젠에서 설계한 지문 인식을 위한 API 인 NBioAPI 를 바탕으로 제작되었다.

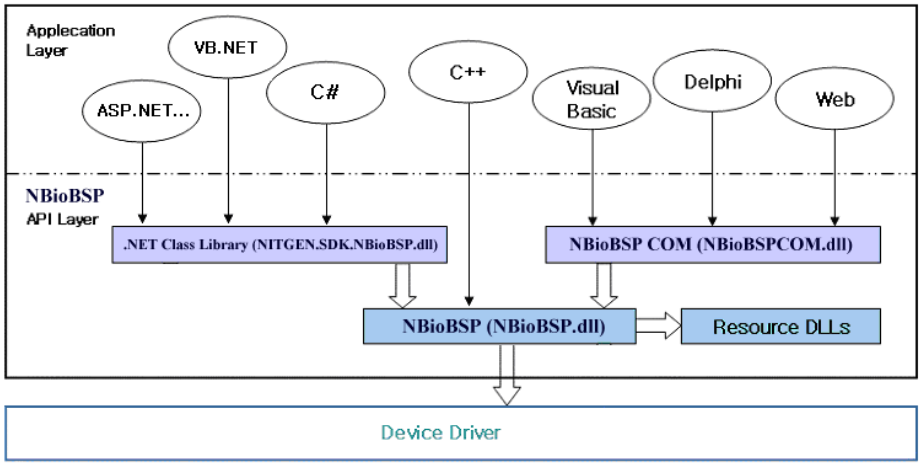
NBioBSP 는 기본적으로 지문 인증과 관련된 모든 API 들과 함께 사용자 등록 및 인증을 위한 사용자 인터페이스 등을 위저드 방식으로 모두 제공하기 때문에 개발자들은 최소한의 노력으로 개발코자 하는 어플리케이션에 니트젠의 지문 인식 기능을 접목시킬 수 있다.

1.1 제공 모듈

- NBioBSP.dll
  NBioBSP.dll 은 사용자 등록, 인증 등 지문 인증과 관련된 모든 기능을 구현하고 있는 NBioBSP 의 핵심 모듈이다.

- NBioBSPCOM.dll
  NBioBSPCOM.dll 은 COM 을 사용하는 모든 언어(Visual Basic, Delphi, ASP ..)에서 사용할 수 있도록 개발되었다.
  NBioBSPCOM 은 NBioBSP.dll 을 이용해서 기능을 제공하도록 개발되었으며, NBioBSP.dll 의 거의 모든 기능을 제공하고 있다.

- NBioBSPCOMLib.dll
  64bit SDK 에서만 제공되며, .NET 언어에서 COM 을 사용할 때 필요한 모듈이다.



[NBioBSP SDK 개발 모델]

1.2 Sample 프로그램

1.2.1 제공되는 Sample (32bit SDK)

- Visual Basic

  BSPDemoVB – NBioBSP 의 기본적인 기능에 대한 예제

  DataExportDemoVB – NBioBSP 의 데이터 Export / Import 기능에 대한 예제

  UITestVB – NBioBSP 의 UI 옵션을 조정하는 기능에 대한 예제

  IndexSerchVB – NBioBSP 의 IndexSearch 를 사용하는 방법에 대한 예제

  BSPRollDemoVB – NBioBSP 의 회전 지문 취득에 대한 예제

- ASP

  ASP 를 이용하여 등록 / 인증을 하는 방법에 대한 예제

- C#

  BSPDemoCSharp – NBioBSP 의 기본적인 기능에 대한 예제

  UITestCSharp – NBioBSP 의 UI 옵션을 조정하는 기능에 대한 예제

- Delphi

  BSPDemoDP - NBioBSP 의 기본적인 기능에 대한 예제

  IndexSearchDP - NBioBSP 의 IndexSearch 를 사용하는 방법에 대한 예제

  UITestDP - NBioBSP 의 UI 옵션을 조정하는 기능에 대한 예제

1.2.2 제공되는 Sample (64bit SDK)

- C#

  BSPDemoCSharp – NBioBSP 의 기본적인 기능에 대한 예제

  UITestCSharp – NBioBSP 의 UI 옵션을 조정하는 기능에 대한 예제

제 2 장 Visual Basic 프로그래밍

이 장에서는 NBioBSP COM 모듈을 이용한 Visual Basic 프로그래밍에 대해 설명한다.
NBioBSP COM 모듈(NBioBSPCOM.dll)은 COM 을 이용하는 언어를 지원하기 위한 목적으로 개발되었다.

NBioBSP COM 모듈은 NBioBSP 에서 제공하는 모든 기능을 제공하지는 않는다. 지문 데이터는 웹 프로그래밍에서 사용할 수 있도록 텍스트 형태로 제공하며 Visual Basic 프로그래밍이나 Delphi 프로그래밍에서는 바이너리 형태의 지문 데이터 와 텍스트 형태의 지문 데이터 포맷을 모두 지원한다.

이 장에서는 NBioBSP COM 모듈 사용법을 Visual Basic 을 예로 설명한다.

2.1 모듈 초기화 및 종료

2.1.1 모듈 초기화

NBioBSP COM 모듈을 초기화 하는 방법은 새로운 Object 로 선언해 주는 방법과 CreateObject() 함수를 이용하여 초기화 하는 방법이 있다. 이 두 가지 방법은 동일한 결과를 나타낸다.

- Method 1

```
Dim objNBioBSP As NBioBSPCOMLib.NBioBSP   'Declaration NBioBSPCOM Object
...
Set objNBioBSP = New NBioBSPCOMLib.NBioBSP
```

이 방법을 사용하려면 NBioBSPCOM 객체를 사용하기 전에 Visual Basic "프로젝트" 메뉴의 "참조" 에서 사용 가능한 참조 항목에 "NITGEN's NBioBSP SDK Type Library"를 포함시키도록 한다.

- Method 2

```
Dim objNBioBSP As NBioBSPCOMLib.NBioBSP   ' Declaration NBioBSPCOM Object
...
Set objNBioBSP = CreateObject("NBioBSPCOM.NBioBSP")
```

2.1.2 Child Object 의 선언

NBioBSP COM 모듈에는 7 가지 하위 Object 가 있다. 각각의 Object 는 디바이스, 추출, 매칭, 지문 데이터 처리, 지문이미지, 서치 등의 7 가지 기능을 수행한다.

- Device Object 의 선언
  디바이스의 옵션 설정 및 오픈, 클로즈 등의 기능을 수행한다.

```
Dim objDevice As IDevice ' Declaration Device Object
...
Set objDevice = objNBioBSP.Device
```

- Extraction Object 의 선언
  지문의 캡쳐, 등록 등의 지문 데이터 추출 기능을 수행한다.

```
Dim objExtraction As IExtraction  ' Declaration Extraction Object
...
Set objExtraction = objNBioBSP.Extraction
```

- Matching Object 의 선언
  지문 데이터의 매칭을 수행한다.

```
Dim objMatching As IMatching      ' Declaration Matching Object
...
Set objMatching = objNBioBSP.Matching
```

- FPData Object 의 선언
  지문 데이터의 변환 및 새로운 지문 데이터로 합치는 기능을 수행한다.

```
Dim objFPData As IFPData ' Declaration FPData Object
...
Set objFPData = objNBioBSP.FPData
```

- FPImage Object 의 선언
  지문 이미지의 추출 및 저장 기능을 수행한다.

```
Dim objFPImage As IFPImage        ' Declaration FPImage Object
...
Set objFPImage = objNBioBSP.FPImage
```

- IndexSearch Object 의 선언
  IndexSearch Engine 의 기능을 수행하는 모듈이다.

```
Dim objIndexSearch As IIndexSearch        ' Declaration IndexSearch Object
...
Set objIndexSearch = objNBioBSP.IndexSearch
```

- NSearch Object 의 선언
  NSearch Engine 의 기능을 수행하는 모듈이다.

```
Dim objNSearch As INSearch        ' Declaration NSearch Object
...
Set objNSearch = objNBioBSP.NSearch
```

2.1.3 모듈 사용 종료

어플리케이션을 종료할 때는 선언했던 Object 를 Free 해 주는 것이 좋다. 비주얼 베이직 같은 경우 어플리케이션이 종료되면서 자동적으로 수행되므로 반드시 해제할 필요는 없다.

```
Set objNBioBSP = nothing ' Free NBioBSPCOM object
```

2.2 디바이스 관련 프로그래밍

특정 디바이스를 사용하기 위해서는 반드시 디바이스를 오픈하는 과정을 거쳐야 한다. 먼저 시스템에 어떤 디바이스들이 연결되어 있는지에 대한 정보를 얻기 위해서는 Device 에 대한 object 를 선언한 다음 이 object 의 Enumerate 메소드를 이용하면 된다.

2.2.1 디바이스 열거하기

디바이스를 오픈하기 하기 전에 Enumerate 메소드를 이용하여 사용자의 PC 에 연결되어 있는 디바이스의 개수 및 종류를 알 수 있다.

Enumerate 메소드를 호출하면 EnumCount 속성에 현재 시스템에 부착되어 있는 디바이스의 개수의 값이 들어가며, EnumDeviceID 속성에는 각 디바이스에 대한 ID 값이 들어가 있다. EnumDeviceID 는 LONG 값을 갖는 배열이다.

EnumDeviceID 는 내부적으로 디바이스 이름과 인스턴스 번호로 구성되어 있다. NBioBSP 에서는 디폴트로 FDU01 지문인식기에 대해서는 0x02 를, FDU11 지문인식기 타입에 대해서는 0x04 를 할당해 놓고 있다.

```
DeviceID = Instance Number + Device Name
```

만일 현재 사용하고 있는 시스템에서 각 타입의 디바이스가 하나씩 만 존재하는 경우 인스턴스 번호가 0 이므로 디바이스 이름과 DeviceID 는 같은 값을 갖게 된다. 더 자세한 것은 C / C++ 프로그래밍을 참조하기 바란다.
아래는 Enumerate 메소드를 사용하는 예제를 보여주고 있다.

```
Set objDevice = objNBioBSP.Device ' Set Device object
...
Call objDevice.Enumerate          ' Enumerate devices
...
comboDevice.AddItem "Auto_Detect"
For DeviceNumber = 0 To objDevice.EnumCount
   comboDevice.AddItem objDevice.EnumDeviceName(DeviceNumber)
Next i
...
```

EnumDeviceID(DeviceNumber) 속성에서 DeviceNumber 부분에 디바이스 번호를 입력하면 해당 디바이스의 ID 를 알려준다. 예를 들어 첫번째 디바이스의 DeviceID 를 알고 싶은 경우엔 EnumDeviceID(0) 라고 써주면 된다.

2.2.2 디바이스 초기화

NBioBSP COM 에서 사용할 디바이스를 선택하기 위해서는 Open 메소드를 호출하면 된다. Enroll, Verify, Capture 등, 디바이스와 관련된 작업을 수행하기 위해서는 반드시 Open 메소드를 이용하여 디바이스 초기화를 먼저 수행하여야 한다.

설치되어 있는 디바이스의 ID 를 모르는 경우에는 Enumerate 메소드를 이용하여 먼저 설치되어 있는 디바이스 ID 리스트를 얻는다.

```
DeviceID = NBioBSP_DEVICE_ID_FDU01_0      ' First instance of FDU01 device
Call objDevice.Open(DeviceID)

If objDevice.ErrorCode = NBioBSPERROR_NONE Then
   ' Open device success ...
Else
   ' Open device failed ...
End If
```

만일 자동으로 사용할 디바이스를 지정하고 싶으면 DeviceID 로 NBioBSP_DEVICE_ID_AUTO_DETECT 를 사용하면 된다.

```
objDevice.Open(NBioBSP_DEVICE_ID_AUTO_DETECT)
```

NBioBSP_DEVICE_ID_AUTO_DETECT 를 사용하면 디바이스가 여러 개 있는 경우 가장최근에 사용한 디바이스를 먼저 검색하게 된다.

2.2.3 디바이스 사용 끝내기

사용중인 디바이스를 더 이상 사용할 필요가 없을 경우 Close 메소드를 이용하여 디바이스의 사용을 해제할 수 있다. Close 메소드를 호출할 때는 반드시 Open 메소드를 호출할 때 사용했던 DeviceID 를 사용하여야 한다.

```
DeviceID = NBioBSP_DEVICE_ID_FDU01_0
Call objDevice.Open(DeviceID)
...
Call objDevice.Close(DeviceID)

If objDevice.ErrorCode = NBioBSPERROR_NONE Then
   ' Close device success ...
Else
   ' Close device failed ...
End If
```

다른 디바이스를 오픈 하는 경우에도 반드시 이전에 사용하고 있던 디바이스의 사용을 먼저 해제해주어야 한다.

2.3 지문 등록

지문을 등록하기 위해서는 Extraction object 를 선언한 다음 Enroll 메소드를 사용한다. NBioBSP COM 모듈에서는 모든 지문 데이터를 바이너리 혹은 텍스트로 인코딩된 두 가지 형태를 사용할 수 있다. Enroll 이 성공하면 FIR 혹은 TextEncodeFIR 속성에 Payload 가 포함된 지문 데이터가 담겨진다. FIR 은 바이트 형식이고 TextEncodeFIR 는 String 형식을 갖는다.

```
Dim szTextEncodeFIR As String
Dim szPayload As String
...
Set objExtraction = objNBioBSP.Extraction
Call objExtraction.Enroll(szPayload, null)

If objExtraction.ErrorCode = NBioBSPERROR_NONE Then
   ' Enroll success ...
   szTextEncodeFIR = objExtraction.TextEncodeFIR
   ' Write FIR data to file or DB
Else
   ' Enroll failed ...
End If
```

위 예는 텍스트로 인코딩된 지문 데이터를 얻는 방법을 설명한 것으로 이를 저장하려면 szTextEncodeFIR 를 파일 또는 DB 에 저장하면 된다.

바이너리 형태로 인코딩된 지문 데이터를 얻을 경우는

```
szTextEncodeFIR = objExtraction.TextEncodeFIR
```

이 부분을 아래와 같이 바꿔주면 된다. 필요한 경우에는 동시에 사용도 가능하다. 이 경우 biFIR 에 바이너리 형태로 인코딩된 지문 데이터가 저장된다.

```
Dim biFIR() As Byte
...
ReDim biFIR(objExtraction.FIRLength) As Byte
biFIR = Space(objExtraction.FIRLength)
biFIR = objExtraction.FIR
```

2.4 지문 인증

인증을 수행할 때는 Matching object 를 선언한 다음 Verify 메소드를 사용한다. Verify 메소드는 파라미터로 이전에 등록되어 있던 지문 데이터를 취한다. Verify 메소드는 현재 입력 받은 지문 데이터와 등록되어 있던 바이너리 형태의 지문 데이터 혹은 텍스트로 인코딩된 데이터를 비교하며 이 두 가지의 등록된 데이터 중 하나를 Verify 에 인수로 넘겨준다. 그 결과값은 MatchingResult 속성에 저장된다. MatchingResult 에는 인증이 성공하면 1 이, 실패하면 0 이 들어간다. Verify 메소드는 Payload 를 되돌려주게 되는데 인증이 성공 했을 경우 ExistPayload 를 확인해 1 이면 Payload 가 존재하는 것이고 0 이면 존재하지 않는 것이다. Payload 가 존재할 경우 TextEncodePayload 에 포함된다.

```
Dim storedFIRTextData As String
Dim szPayload As String
...
' Read stored FIR Data from File or DB.
...
Set objMatching = objNBioBSP.Matching
Call objMatching.Verify(storedFIRTextData)        ' TextEncodedFIR

If objMatching.MatchingResult = NBioAPI_TRUE then
   ' Verify success
   If objMatching.ExistPayload = NBioAPI_TRUE Then
      ' Exist
      szPayload = objMatching.TextEncodePayload
   Else
      ...
   End If
Else
   ' Verify failed
End if
```

2.5 Client / Server 환경에서의 프로그래밍

독립 실행형 환경과는 달리 클라이언트/서버 환경에서는 지문을 입력 받는 곳과 매칭하는 곳이 다르다. 즉 클라이언트에서는 보통 지문을 캡쳐하고 서버에서 매칭이 이루어진다. 등록을 위한 지문을 입력 받을 때는 Enroll 메소드를 사용하고 인증을 위한 지문을 입력 받을 때는 Capture 메소드를 사용한다.

서버에서 매칭을 하는 경우에는 VerifyMatch 메소드를 사용한다. VerifyMatch 메소드는 파라미터로 등록되어 있던 지문 데이터와 클라이언트로부터 입력 받은 지문을 취한다.

2.5.1 지문 등록하기

클라이언트에서 등록용 지문 데이터를 입력 받기 위해서는 Enroll 메소드를 사용한다.

```
Dim szTextEncodeFIR As String
Dim szPayload As String
...
Set objExtraction = objNBioBSP.Extraction
Call objExtraction.Enroll(szPayload, null)

If objExtraction.ErrorCode = NBioBSPERROR_NONE Then
   ' Enroll success ...
   szTextEncodeFIR = objExtraction.TextEncodeFIR
   ' Write FIR data to file or DB
Else
   ' Enroll failed ...
End If
```

2.5.2 지문 인증하기

먼저 인증에 사용할 지문 데이터를 클라이언트로부터 얻어온다. 이때 사용할 메소드는 Capture 이다.
Enroll 메소드와 Capture 메소드를 이용하여 가져오는 지문 데이터의 차이점은 Enroll 메소드를 이용하면 어떤 손가락을 등록했는지에 대한 정보를 가지고 있어 여러 개의 지문 정보를 하나의 지문 데이터로 전송이 가능한 반면 Capture 메소드를 이용하면 단순히 하나의 지문 데이터만 입력 받는다. Capture 메소드를 사용하기 위해서는 Extraction object 를 선언한 다음 사용하게 되며 Capture 의 용도를 인자로 넘겨 주어야 한다. 현재 인자로 사용하는 값이 여러 개 있으나 현재는 NBioAPI_FIR_PURPOSE_VERIFY 만을 지원한다.

```
Dim szTextEncodeFIR As String
...
Set objExtraction = objNBioBSP.Extraction
Call objExtraction.Capture(NBioAPI_FIR_PURPOSE_VERIFY)

If objExtraction.ErrorCode = NBioBSPERROR_NONE Then
    ' Capture success ...
    szTextEncodeFIR = objExtraction.TextEncodeFIR
    ' Write FIR data to file or DB
Else
   ' Capture failed ...
End If
```

서버쪽에서의 저장된 지문 데이터 간의 비교는 VerifyMatch 메소드를 이용하면 된다. VerifyMatch 메소드에는 클라이언트에서 넘겨 받은 FIR 과 서버에 저장된 FIR 이렇게 두 개의 인자를 넣어 주어야 한다. 비교 결과는 MatchingResult 속성에서 확인할 수 있다. MatchingResult 에는 인증이 성공하면 1 이, 실패하면 0 이 들어간다. VerifyMatch 메소드는 Payload 를 되돌려주게 되는데 인증이 성공했을 경우 ExistPayload 를 확인해 1 이면 Payload 가 존재하는 것이고 0 이면 존재하지 않는 것이다. Payload 가 존재할 경우 TextEncodePayload 에 포함된다. 이때 TextEncodePayload 에 포함된 Payload 는 두 번째 인수인 storedFIRTextData 가 가지고 있던 Payload 로 첫번째 processedFIRTextData 의 Payload 에는 영향을 받지 않는다.

```
Dim storedFIRTextData As String
Dim processedFIRTextData As String
Dim szPayload As String
...
' Get processed FIR Data from Client and Read stored FIR Data from File or DB.
...
Set objMatching = objNBioBSP.Matching
Call objMatching.VerifyMatch(processedFIRTextData, storedFIRTextData)

If objMatching.MatchingResult = NBioAPI_TRUE then
   ' Matching success
   If objMatching.ExistPayload = NBioAPI_TRUE Then
      ' Exist
      szPayload = objMatching.TextEncodePayload
   Else
      ...
   End If
Else
   ' Matching failed
End if
```

2.6 Payload 사용

지문 데이터 속에 사용자가 원하는 데이터를 포함 시킬 수 있는데 이때 지문 데이터 속에 포함되는 사용자 데이터를 Payload 라고 한다. NBioBSP COM 모듈에서는 바이너리 혹은 텍스트 인코딩된 데이터를 이용 할 수 있다.

2.6.1 지문 데이터에 Payload 삽입

Payload 를 지문 데이터에 삽입하는 방법은 Enroll 메소드를 이용하여 지문 데이터를 작성할 때 삽입하는 방법과 이미 작성된 지문 데이터에 CreateTemplate 메소드를 이용하여 삽입하는 방법이 있다.

Enroll 메소드를 이용하는 방법은 삽입을 원하는 Payload 데이터를 Enroll 메소드를 호출할 때 파라미터로 넘겨 주어 Payload 가 삽입된 지문 데이터를 얻을 수 있다.

```
Dim szTextEncodeFIR As String
Dim szPayload As String
...
szPayload = "Your Payload Data"
...
Set objExtraction = objNBioBSP.Extraction
Call objExtraction.Enroll(szPayload, null)

If objExtraction.ErrorCode = NBioBSPERROR_NONE Then
   ' Enroll success ...
   szTextEncodeFIR = objExtraction.TextEncodeFIR
   ' Write FIR data to file or DB
Else
   ' Enroll failed ...
End If
```

이미 등록된 지문 데이터에 Payload 데이터를 삽입하는 방법은 CreateTemplate 메소드를 이용하면 된다. CreateTemplate 메소드는 이외에도 기존의 지문 데이터와 신규 지문 데이터를 합치는 역할도 수행할 수 있다. CreateTemplate 메소드를 사용하기 위해서는 FPData object 를 선언해야 한다.

이때 새로 생성된 지문 데이터는 Enroll 에서 했던 것과 마찬가지로 TextEncodeFIR 속성에 담겨진다.

```
Dim storedFIRTextData As String
Dim newFIRTextData As String
Dim szPayload As String
...
szPayload = "Your Payload Data"
...
Set objFPData = objNBioBSP.FPData
Call objFPData.CreateTemplate(storedFIRTextData, null, szPayload)

If objFPData.ErrorCode = NBioBSPERROR_NONE Then
   ' CreateTemplate success ...
   newFIRTextData = objFPData.TextEncodeFIR
   ' Write FIR data to file or DB
Else
   ' CreateTemplate failed ...
End If
```

2.6.2 지문 데이터로부터 Payload 추출

지문 템플릿(등록용 데이터) 에 저장되어 있는 Payload 데이터는 Verify 또는 VerifyMatch 메소드를 이용하여 매칭한 결과가 참일 때만 꺼내올 수 있다.

매칭 후 지문 데이터 속에 Payload 가 있는지 여부는 ExistPayload 속성에 들어가 있다. 만일 ExistPayload 가 참이면 TextEncodePayload 속성에 Payload 데이터가 담겨진다.

```
Dim storedFIRTextData As String
Dim szPayload As String
...
' Read FIRText Data from File or DB.
...
Set objMatching = objNBioBSP.Matching
objMatching.Verify(storedFIRTextData)

If objMatching.MatchingResult = NBioAPI_TRUE Then
   ' Verify success
   If objNBioBSP.ExistPayload = NBioAPI_TRUE Then
      ' Exist payload
      szPayload = objMatching.TextEncodePayload
   End If
Else
   ' Verify failed
End if
```

VerifyMatch 메소드를 이용하여 Payload 데이터를 추출하는 방법도 Verify 메소드를 이용하는 방법과 동일하다. VerifyMatch 를 호출할 때 첫번째 파라미터로 비교데이터(client)를, 두 번째 파라미터로 등록된 데이터(server)를 입력해야 한다.

Payload 데이터를 포함하고 있는 지문 데이터를 두 번째 파라미터로 사용하여야만 매칭이 성공한 경우 Payload 데이터를 추출할 수 있다. 만약 첫번째 파라미터에도 Payload 데이터가 포함되어 있으면 이는 무시하고 추출되지 않는다. 그러므로 지문 등록 시 작성된 지문 데이터를 두 번째 파라미터로 사용하여야 된다.

```
...
Set objMatching = objNBioBSP.Matching
Call objMatching.VerifyMatch(capturedFIRTextData, enrolledFIRTextData)

if objMatching.MatchingResult = NBioAPI_TRUE then
   ' Verify success
   if objMatching.ExistPayload = NBioAPI_TRUE then
      ' Get payload
      szPayload = objMatching.TextEncodePayload
    End if
End if
...
```

2.7 UI 변경

NBioBSP 에서 기본적으로 제공되는 UI 가 아닌 Customize 된 UI 를 사용하고 싶은 경우 NBioBSP COM 모듈에서는 Customized 된 UI 가 들어가 있는 리소스 파일을 읽어올 수 있는 방법을 제공한다.

NBioBSP 는 기본적으로 영문 UI 가 제공되므로 영문 UI 가 아닌 다른 언어의 UI 가 들어있는 리소스 파일을 로드하고 싶은 경우 SetSkinResource 메소드를 사용하면 된다.

```
Dim szSkinPath
...
CommonDialog.ShowOpen

If CommonDialog.CancelError = False then
     szSkinPath = CommonDialog.FileName
     objNBioBSP.SetSkinResource(szSkinPath)
End if
```

이 때 리소스 파일 경로는 전체 경로를 넘겨주어야 한다. Customized 된 UI 를 작성하는 방법은 별도의 문서로 제공된다.

부록 A. COM Reference

A.1 NBioBSP Object

NBioBSP COM 을 사용하기 위한 Child Object 를 선언하기 위한 프로퍼티와 기타 NBioBSP 의 버전 정보 및 스킨 설정 등의 기능을 사용하기 위한 오브젝트입니다. 이 오브젝트는 반드시 선언되어야 합니다.

A.1.1 Property

long ErrorCode

실행한 메소드 및 프로퍼티 설정 중에 발생한 에러에 대한 코드값을 가져올 때 사용합니다.

0 인 경우 성공 이외의 값은 실패를 나타낸다.

BSTR ErrorDescription

ErrorCode 를 통하여 받은 에러 코드값을 문자형식으로 출력하고자 할 때 사용합니다.

VARIANT Device

디바이스에 관한 명령 집합을 가지고 있는 오브젝트입니다.

디바이스 사용 시작 및 종료 디바이스 종류 선택, 옵션 값의 변경 및 설정 등의 기능을 수행합니다.

VARIANT Extraction

지문 이미지를 입력 받아 특징점을 추출하는 기능을 제공합니다.

UI 의 옵션을 변경하거나 지문 이미지를 저장할 수 있습니다.

VARIANT Matching

지문 데이터를 이용하여 본인 여부를 확인할 수 있는 기능을 제공합니다.

UI 의 옵션을 변경하는 기능 및 저장된 지문 데이터를 이용하거나 즉석해서 입력 받은 지문과 비교하여 결과를 알려주는 기능을 수행합니다.

VARIANT FPData

두개의 지문 데이터를 받아서 하나의 지문 데이터로 합치거나 지문 데이터에 페이로드 값을 넣을 때 사용합니다.

VARIANT FPImagae

지문을 캡쳐 할 때 이 지문 이미지를 추출하기 위해 사용하는 오브젝트입니다.

VARIANT NSearch

다수의 지문 데이터를 메모리 데이터베이스에 저장한 후 특정 지문을 검색하여 찾아주는 기능을 수행하는 오브젝트입니다.

NSearch Engine 오브젝트입니다.

VARIANT IndexSearch

다수의 지문 데이터를 메모리 데이터베이스에 저장한 후 특정 지문을 검색하여 찾아주는 기능을 수행하는 오브젝트입니다.

IndexSearch Engine 오브젝트입니다.

BOOL CheckValidityModule

NBioBSP 모듈의 유효성을 검사하는 프로퍼티입니다. TRUE 면 안전한 NBioBSP 모듈이고 FALSE 면 모듈이 변조된 것입니다.

BSTR MajorVersion

NBioBSP 의 메이저 버전을 표시합니다.

BSTR MinorVersion

NBioBSP 의 마이너 버전을 표시합니다. 2 자리 입니다.

BSTR BuildNumber

NBioBSP 의 빌드 번호를 표시합니다.

A.1.2 Method

SetSkinResource (BSTR bszSkinPath)

Description

NBioBSP 의 리소스 파일을 변경하고자 할 때 사용하는 메소드 입니다.

Parameter

bszSkinPath : 변경할 리소스 DLL 파일의 전체 경로를 입력한다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

A.2 Device Object

디바이스에 관한 명령 집합을 가지고 있는 오브젝트입니다. 디바이스 사용시작 및 종료 디바이스 종류 선택, 옵션 값의 변경 및 설정 등의 기능을 수행합니다.

A.2.1 Property

long ErrorCode

실행한 메소드 및 프로퍼티 설정 중에 발생한 에러에 대한 코드값을 가져올 때 사용합니다. 0 인 경우 성공 이외의 값은 실패를 나타냅니다.

BSTR ErrorDescription

ErrorCode 를 통하여 받은 에러 코드값을 문자형식으로 출력하고자 할 때 사용합니다.

long EnumCount

PC 에 연결되어 있는 디바이스의 개수를 가져오는 프로퍼티 입니다.

Enumerate 메소드를 통해 값이 할당됩니다.

long EnumDeviceID(long nIndex)

PC 에 연결되어있는 디바이스 아이디를 가져옵니다.

Enumerate 메소드를 통해 값이 할당됩니다.

long EnumDeviceNameID(long nIndex)

PC 에 연결되어있는 디바이스 이름의 아이디를 가져옵니다.

Enumerate 메소드를 통해 값이 할당됩니다.

long EnumDeviceInstance(long nIndex)

PC 에 연결되어있는 디바이스의 Instance 를 가져옵니다.

Enumerate 메소드를 통해 값이 할당됩니다.

BSTR EnumDeviceName(long nIndex)

PC 에 연결되어있는 디바이스의 이름을 문자열로 가져옵니다.

Enumerate 메소드를 통해 값이 할당됩니다.

BSTR EnumDeviceDescription(long nIndex)

PC 에 연결되어있는 디바이스의 설명을 문자열로 가져옵니다.

Enumerate 메소드를 통해 값이 할당됩니다.

BSTR EnumDeviceDll(long nIndex)

PC 에 연결되어있는 디바이스의 드라이버 DLL 파일명을 문자열로 가져옵니다.

Enumerate 메소드를 통해 값이 할당됩니다.

BSTR EnumDeviceSys(long nIndex)

PC 에 연결되어있는 디바이스의 드라이버 Sys 파일명을 문자열로 가져옵니다.

Enumerate 메소드를 통해 값이 할당됩니다.

long EnumDeviceAutoOn(long nIndex)

PC 에 연결되어있는 디바이스가 AutoOn 을 지원하는지 가져옵니다. AutoOn 을 지원하는 디바이스의 경우 1 의 값이, 그렇지 않은경우 0 의 값을 가져옵니다.

Enumerate 메소드를 통해 값이 할당됩니다.

long EnumDeviceBrightness(long nIndex)

PC 에 연결되어있는 디바이스의 밝기값을 가져옵니다.

Enumerate 메소드를 통해 값이 할당됩니다.

long EnumDeviceContrast(long nIndex)

PC 에 연결되어있는 디바이스의 고대비값을 가져옵니다.

Enumerate 메소드를 통해 값이 할당됩니다.

long EnumDeviceGain(long nIndex)

PC 에 연결되어있는 디바이스의 Gain 값을 가져옵니다.

Enumerate 메소드를 통해 값이 할당됩니다.

long OpenedDeviceID

Open 메소드를 이용해서 마지막으로 Open 한 디바이스 아이디를 가져옵니다.

long GetDeviceName(long nDeviceID)

해당 디바이스 아이디를 파라메터로 입력하면 디바이스의 이름을 가져옵니다.

long GetDeviceNumber(long nDeviceID)

해당 디바이스 아이디를 파라메터로 입력하면 디바이스의 번호를 가져옵니다.

디바이스 번호는 USB 형식에서만 의미가 있으며 디바이스를 PC 에 연결한 순서대로 0 부터 값이 설정됩니다.

long MakeDeviceID(long nDeviceName, long nDeviceNumber)

디바이스 이름과 디바이스 번호를 이용해서 디바이스 아이디를 생성합니다.

이렇게 생성된 디바이스 아이디를 이용해서 디바이스를 Open 또는 Close 하게 됩니다.

long ImageWidth(long nDeviceID)

파라메터로 입력된 디바이스 아이디를 갖는 디바이스에서 캡쳐되는 이미지의 넓이를 가져옵니다.

long ImageHeight(long nDeviceID)

파라메터로 입력된 디바이스 아이디를 갖는 디바이스에서 캡쳐되는 이미지의 높이를 가져옵니다.

long Brightness(long nDeviceID)

파라메터로 입력된 디바이스 아이디를 갖는 디바이스에서 캡쳐되는 이미지의 밝기를 설정하거나 가져옵니다.

long Contrast(long nDeviceID)

파라메터로 입력된 디바이스 아이디를 갖는 디바이스에서 캡쳐되는 이미지의 명도를 설정하거나 가져옵니다.

long Gain(long nDeviceID)

파라메터로 입력된 디바이스 아이디를 갖는 디바이스에서 캡쳐되는 이미지의 게인값을 설정하거나 가져옵니다.

long WindowStyle

화면에 보여지는 NBioBSP 의 UI 의 형식을 선택합니다.

윈도우의 형식에는 POPUP, INVISIBLE, CONTINUOUS 등 3 가지가 있습니다.

COM 에서는 CONTINUOUS 형식은 사용하지 않고 있습니다.

BOOL WindowOption(long nOption)

화면에 보여지는 NBioBSP 의 UI 의 형식을 선택합니다.

윈도우 옵션은 지문이미지 출력금지, 윈도우를 최상위 윈도우로 만들기, Enroll 메소드에서 Welcome 메시지 출력 금지 등이 있습니다.

long ParentWnd

현재 사용하지 않는 프로퍼티 입니다.

long FingerWnd

윈도우 스타일을 지정할 때 윈도우를 보이지 않도록 설정한 경우 지문 이미지가 출력되는 컨트롤을 지정할 수 있습니다. 지문 이미지가 출력되길 원하는 컨트롤의 핸들을 입력하면 됩니다.

BSTR CaptionMsg

Enroll 메소드를 이용할 때 취소를 선택한 경우 화면에 나타나는 메시지 박스의 문구를 지정하는 프로퍼티 입니다.

BSTR CancelMsg

Enroll 메소드를 이용할 때 취소를 선택한 경우 화면에 나타나는 메시지 박스의 캡션을 지정하는 프로퍼티 입니다.

BSTR FPForeColor

지문 윈도우가 화면에 보이지 않도록 하는 INVISIBLE 옵션과 사용자가 지정한 컨트롤에 지문이 출력되도록 한 경우 해당 컨트롤에 출력되는 지문 이미지의 색상을 변경할 때 사용합니다.

BSTR FPBackColor

지문 윈도우가 화면에 보이지 않도록 하는 INVISIBLE 옵션과 사용자가 지정한 컨트롤에 지문이 출력되도록 한 경우 해당 컨트롤의 배경 색상을 변경할 때 사용합니다.

BOOL DisableFingerForEnroll(long nFingerID)

Enroll 메소드를 이용하여 지문등록을 수행하기 전에 이 프로퍼티의 파라메터로 손가락 ID 를 입력한 후 TRUE 를 입력하면 해당 손가락을 선택하지 못하도록 설정할 수 있습니다. 이 옵션을 해제하려면 FALSE 를 입력하면 됩니다.

BOOL CheckFinger

현재 Open 된 디바이스의 Sensor 에 손가락을 올려놓았는지를 검사한다. 이 함수는 현재 USB 디바이스에서만 지원하며 디바이스 드라이버의 파일버전이 4.1.0.1 이상이어야 지원된다.
지문이 Sensor 위에 있을 경우 True 를 리턴한다.

A.2.2 Method

Open(long nDeviceID)

Description
지문을 입력 받기 위해서 디바이스의 동작을 개시할 때 사용하는 메소드 입니다.

Parameter
nDeviceID : 사용하고자 하는 디바이스의 아이디. PC 에 연결된 디바이스 개수가 1 개 이상인 경우 MakeDeviceID 프로퍼티를 통해 생성할 수 있습니다.

Relation Property
ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력.

Close(long nDeviceID)

Description
디바이스 사용을 종료할 때 사용하는 메소드 입니다.

Parameter
nDeviceID : 종료하고자 하는 디바이스의 아이디. 디바이스를 Open 한 후 종료하고자 할 때는 이 메소드를 호출하여야 합니다. Open 한 디바이스 아이디는 OpenedDeviceID 프로퍼티를 이용하여 가져올 수 있습니다.

Relation Property
ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력.

Enumerate()

Description
현재 PC 에 연결되어 있는 디바이스의 종류를 가져올 때 사용합니다.

Relation Property
ErrorCode : 실행 결과를 출력합니다. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력합니다.
EnumCount : 연결된 디바이스의 개수
EumDeviceID(long nIndex) : 검색된 디바이스 아이디

Adjust()

Description
디바이스 밝기 조정 윈도우를 출력한다. 이 함수를 통해 지문의 밝기를 조정하여 좀더 고품질의 지문 데이터를 얻을 수 있도록 할 수 있습니다.

Relation Property
ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력.

A.3 Extraction Object

지문 이미지를 입력 받아 특징점을 추출하는 기능을 제공합니다. UI 의 옵션을 변경하거나 지문 이미지를 저장할 수 있습니다.

A.3.1 Property

long ErrorCode

실행한 메소드 및 프로퍼티 설정 중에 발생한 에러에 대한 코드값을 가져올 때 사용합니다. 0 인 경우 성공 이외의 값은 실패를 나타냅니다.

BSTR ErrorDescription

ErrorCode 를 통하여 받은 에러 코드값을 문자형식으로 출력하고자 할 때 사용합니다.

BSTR TextEncodeFIR

텍스트로 인코딩 된 지문 데이터입니다.

long FIRLength

바이너리 형태의 지문 데이터의 길이입니다.

VARIANT FIR

바이너리 형태의 지문 데이터입니다. 이 지문 데이터를 받아 오기 위해서는 FIRLength 만큼의 공간이 미리 정의되어 있어야 합니다.

long FIRFormat

FIR data 의 Format 값

long MaxFingerForEnroll

Enroll 메소드에서 등록할 수 있는 최대 손가락 개수를 설정합니다.

long SamplesPerFinger

Enroll 메소드에서 한 개의 손가락 당 등록되는 지문의 수를 설정합니다. (현재 2 로 고정)

long DefaultTimeout

Caputer, Verify 등의 작업을 수행할 때 지문 입력을 기다리는 시간을 설정합니다.
ms 단위로 입력하여야 합니다.

long EnrollImageQuality

Enroll 작업 수행 시 지문의 이미지 품질을 결정합니다.
여기서 설정한 값 이상이 되어야 지문입력이 됩니다. (30 ~ 100 값 입력)

long VerifyImageQuality

Verify 작업 수행 시 지문의 이미지 품질을 결정합니다.
여기서 설정한 값 이상이 되어야 지문입력이 됩니다. (0 ~ 100 값 입력)

long IdentifyImageQuality

현재 사용하지 않습니다.

long SecurityLevel

지문 데이터의 매칭 작업을 수행하는 경우 매칭 레벨을 결정합니다. (1 ~ 9 값 입력)

long WindowStyle

화면에 보여지는 NBioBSP 의 UI 의 형식을 선택합니다.

윈도우의 형식에는 POPUP, INVISIBLE, CONTINUOUS 등 3 가지가 있습니다.

COM 에서는 CONTINUOUS 형식은 사용하지 않고 있습니다.

BOOL WindowOption(long nOption)

화면에 보여지는 NBioBSP 의 UI 의 형식을 선택합니다.

윈도우 옵션은 지문이미지 출력금지, 윈도우를 최상위 윈도우로 만들기, Enroll 메소드에서 Welcome 메시지 출력 금지 등이 있습니다.

long ParentWnd

현재 사용하지 않는 프로퍼티 입니다.

long FingerWnd

윈도우 스타일을 지정할 때 윈도우를 보이지 않도록 설정한 경우 지문 이미지가 출력되는 컨트롤을 지정할 수 있습니다. 지문 이미지가 출력되길 원하는 컨트롤의 핸들을 입력하면 됩니다.

BSTR CaptionMsg

Enroll 메소드를 이용할 때 취소를 선택한 경우 화면에 나타나는 메시지 박스의 문구를 지정하는 프로퍼티 입니다.

BSTR CancelMsg

Enroll 메소드를 이용할 때 취소를 선택한 경우 화면에 나타나는 메시지 박스의 캡션을 지정하는 프로퍼티 입니다.

BSTR FPForeColor

지문 윈도우가 화면에 보이지 않도록 하는 INVISIBLE 옵션과 사용자가 지정한 컨트롤에 지문이 출력되도록 한 경우 해당 컨트롤에 출력되는 지문 이미지의 색상을 변경할 때 사용합니다.

BSTR FPBackColor

지문 윈도우가 화면에 보이지 않도록 하는 INVISIBLE 옵션과 사용자가 지정한 컨트롤에 지문이 출력되도록 한 경우 해당 컨트롤의 배경 색상을 변경할 때 사용합니다.

BOOL DisableFingerForEnroll(long nFingerID)

Enroll 메소드를 이용하여 지문등록을 수행하기 전에 이 프로퍼티의 파라메터로 손가락 ID 를 입력한 후 TRUE 를 입력하면 해당 손가락을 선택하지 못하도록 설정할 수 있습니다. 이 옵션을 해제하려면 FALSE 를 입력하면 됩니다.

A.3.1 Method

Capture (/*[in, optional]*/ long nPurpose)

Description

한 개의 지문을 받기 위한 메소드 입니다. 간단히 한 개의 지문을 입력하는 창이 나타나 지문입력을 받습니다.

Parameter

nPurpose : 지문 입력 윈도우의 목적을 선택합니다.

이 값은 선택사항 이므로 입력하지 않아도 상관없습니다. (현재는 1 번만 사용)

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

TextEncodeFIR : 텍스트로 인코딩 된 지문 데이터

FIRLength : 지문 데이터의 길이

FIR : 입력 받은 지문 데이터

RollCapture (/*[in, optional]*/ long nPurpose)

Description

회전 지문을 받기 위한 메소드 입니다.

Parameter

nPurpose : 지문 입력 윈도우의 목적을 선택합니다.

이 값은 선택사항 이므로 입력하지 않아도 상관없습니다. (현재는 1 번만 사용)

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

TextEncodeFIR : 텍스트로 인코딩 된 지문 데이터

FIRLength : 지문 데이터의 길이

FIR : 입력 받은 지문 데이터

Enroll (VARIANT payload, /*[in, optional]*/ VARIANT storedFIR)

Description

한 개의 지문을 받기 위한 메소드 입니다. 간단히 한 개의 지문을 입력하는 창이 나타납니다

Parameter

payload : 지문 데이터를 생성할 때 지문 데이터에 임의의 데이터를 넣을 수 있습니다. 이 데이터는 Verify, VerifyMatch 메소드를 통하여 본인의 지문으로 확인이 된 경우에 다시 읽어올 수 있습니다.
데이터 형식은 바이너리 또는 문자열을 사용할 수 있습니다.

storedFIR : 기존의 저장된 지문 데이터를 입력하면 해당 지문 데이터를 수정 또는 임의의 손가락만을 삭제할 수 있습니다. 이 파라메터는 옵션이므로 데이터가 없는 경우 입력하지 않아도 상관없습니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

TextEncodeFIR : 텍스트로 인코딩 된 지문 데이터

FIRLength : 지문 데이터의 길이

FIR : 입력 받은 지문 데이터

A.4 Matching Object

지문 데이터를 이용하여 본인 여부를 확인할 수 있는 기능을 제공합니다. UI 의 옵션을 변경하는 기능 및 저장된 지문 데이터를 이용하거나 즉석해서 입력 받은 지문과 비교하여 결과를 알려주는 기능을 수행합니다.

A.4.1 Property

long ErrorCode

실행한 메소드 및 프로퍼티 설정 중에 발생한 에러에 대한 코드값을 가져올 때 사용합니다. 0 인 경우 성공, 이외의 값은 실패를 나타냅니다.

BSTR ErrorDescription

ErrorCode 를 통하여 받은 에러 코드값을 문자형식으로 출력하고자 할 때 사용합니다.

BOOL MatchingResult

지문 데이터를 매칭한 경우의 결과를 알려줍니다.

BOOL ExistPayload

지문 데이터 매칭이 성공한 경우 Payload 값이 지문 데이터에 포함되어 있는지 여부를 알려줍니다.

BSTR TextEncodePayload

Payload 데이터가 문자열인 경우 이 프로퍼티로 가져올 수 있습니다.

long PayloadLength

바이너리 형태의 Payload 데이터의 길이입니다.

VARIANT Payload

바이너리 형태의 Payload 데이터 입니다. 이 Payload 데이터를 받아 오기 위해서는 PayloadLength 만큼의 공간이 미리 할당되어 있어야 합니다.

long HitNum

CompareTwo Method 수행 결과 중 HitNum 를 알려줍니다.

Long MatchScore

CompareTwo Method 수행 결과 중 MatchScore 를 알려줍니다.

long MaxFingerForEnroll

Enroll 메소드에서 등록할 수 있는 최대 손가락 개수를 설정합니다.

long SamplesPerFinger

Enroll 메소드에서 한 개의 손가락 당 등록되는 지문의 수를 설정합니다. (현재 2 로 고정)

long DefaultTimeout

Caputer, Verify 등의 작업을 수행할 때 지문 입력을 기다리는 시간을 설정합니다.
ms 단위로 입력하여야 합니다.

long EnrollImageQuality

Enroll 작업 수행 시 지문의 이미지 품질을 결정합니다.
여기서 설정한 값 이상이 되어야 지문입력이 됩니다. (30 ~ 100 값 입력)

long VerifyImageQuality

Verify 작업 수행 시 지문의 이미지 품질을 결정합니다.

여기서 설정한 값 이상이 되어야 지문입력이 됩니다. (0 ~ 100 값 입력)

long IdentifyImageQuality

현재 사용하지 않습니다.

long SecurityLevel

지문 데이터의 매칭 작업을 수행하는 경우 매칭 레벨을 결정합니다. (1 ~ 9 값 입력)

long WindowStyle

화면에 보여지는 NBioBSP 의 UI 의 형식을 선택합니다.

윈도우의 형식에는 POPUP, INVISIBLE, CONTINUOUS 등 3 가지가 있습니다.

COM 에서는 CONTINUOUS 형식은 사용하지 않고 있습니다.

BOOL WindowOption(long nOption)

화면에 보여지는 NBioBSP 의 UI 의 형식을 선택합니다.

윈도우 옵션은 지문이미지 출력금지, 윈도우를 최상위 윈도우로 만들기,

Enroll 메소드에서 Welcome 메시지 출력 금지 등이 있습니다.

long ParentWnd

현재 사용하지 않는 프로퍼티 입니다.

long FingerWnd

윈도우 스타일을 지정할 때 윈도우를 보이지 않도록 설정한 경우 지문 이미지가 출력되는 컨트롤을 지정할 수 있습니다.

지문 이미지가 출력되길 원하는 컨트롤의 핸들을 입력하면 됩니다.

BSTR CaptionMsg

Enroll 메소드를 이용할 때 취소를 선택한 경우 화면에 나타나는 메시지 박스의 문구를 지정하는 프로퍼티 입니다.

BSTR CancelMsg

Enroll 메소드를 이용할 때 취소를 선택한 경우 화면에 나타나는 메시지 박스의 캡션을 지정하는 프로퍼티 입니다.

BSTR FPForeColor

지문 윈도우가 화면에 보이지 않도록 하는 INVISIBLE 옵션과 사용자가 지정한 컨트롤에 지문이 출력되도록 한 경우 해당 컨트롤에 출력되는 지문 이미지의 색상을 변경할 때 사용합니다.

BSTR FPBackColor

지문 윈도우가 화면에 보이지 않도록 하는 INVISIBLE 옵션과 사용자가 지정한 컨트롤에 지문이 출력되도록 한 경우 해당 컨트롤의 배경 색상을 변경할 때 사용합니다.

BOOL DisableFingerForEnroll(long nFingerID)

Enroll 메소드를 이용하여 지문등록을 수행하기 전에 이 프로퍼티의 파라메터로 손가락 ID 를 입력한 후 TRUE 를 입력하면 해당 손가락을 선택하지 못하도록 설정할 수 있습니다. 이 옵션을 해제하려면 FALSE 를 입력하면 됩니다.

A.4.2 Method

Verify(VARIANT storedFIR)

Description

저장된 지문 데이터와 즉석에서 받은 지문 데이터를 비교하여 매칭 여부를 알려줍니다.

Parameter

storedFIR : 저장된 지문 데이터를 바이너리 또는 문자열로 입력합니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

MatchingResult : 지문 매칭 결과

ExistPayload : Payload 존재 여부

TextEncodePayload : 텍스트로 인코딩 된 Payload 데이터

PayloadLength : Payload 데이터의 길이

Payload : 입력 받은 Payload 데이터

VerifyMatch (VARIANT processedFIR , VARIANT storedFIR)

Description

저장된 두개의 지문 데이터를 매칭하는 함수입니다.

Parameter

processedFIR : 저장된 지문 데이터를 바이너리 또는 문자열로 입력합니다.

storedFIR : 저장된 지문 데이터를 바이너리 또는 문자열로 입력합니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

MatchingResult : 지문 매칭 결과

ExistPayload : Payload 존재 여부

TextEncodePayload : 텍스트로 인코딩 된 Payload 데이터

PayloadLength : Payload 데이터의 길이

Payload : 입력 받은 Payload 데이터

CompareTwo(VARIANT processedFIR, VARIANT storedFIR)

Description

저장된 두개의 지문 데이터를 매칭하는 함수입니다.

Parameter

processedFIR : 저장된 지문 데이터를 바이너리 또는 문자열로 입력합니다.

storedFIR : 저장된 지문 데이터를 바이너리 또는 문자열로 입력합니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

HitNum : 매칭된 특징점의 개수를 알려줍니다.

MatchScore : 매칭 점수를 알려줍니다.

A.5 FPData Object

두개의 지문 데이터를 받아서 하나의 지문 데이터로 합치거나 지문 데이터에 Payload 값을 넣을 때 사용합니다.

A.5.1 Property

long ErrorCode

실행한 메소드 및 프로퍼티 설정 중에 발생한 에러에 대한 코드값을 가져올 때 사용합니다. 0 인 경우 성공 이외의 값은 실패를 나타냅니다.

BSTR ErrorDescription

ErrorCode 를 통하여 받은 에러 코드값을 문자형식으로 출력하고자 할 때 사용합니다.

long TotalFingerCount

NBioBSP 형식의 지문 데이터로부터 다른 형식의 지문 데이터를 추출하는 경우 추출된 지문 데이터의 개수를 나타냅니다.

long FingerID(long nIndex)

추출된 지문 데이터의 손가락 아이디를 알려줍니다.

파라메터 nIndex 에 0 부터 TotalFingerCount - 1 까지의 숫자를 넣어주면 해당 순서의 손가락 아이디를 알려줍니다.

long SampleNumber

추출된 지문 한 개당 등록된 지문 데이터의 개수를 알려줍니다.

long FPDataSize(long nFingerID)

추출된 지문 데이터의 크기를 알려줍니다.

파라메터 nFingerID 에 손가락 아이디를 넣어주면 해당 손가락의 바이너리 데이터 크기를 알려줍니다.

long FPSampleDataSize(long nFingerID, long nSampleNum)

추출된 지문 데이터의 크기를 알려줍니다.

파라메터 nFingerID 에 손가락 아이디를 nSampleNum 에 sample 번호를 넣어주면 해당 데이터의 바이너리 크기를 알려줍니다.

long FPData(long nFingerID, long nSampleNum)

추출된 지문 데이터입니다.

파라메터 nFingerID 에는 손가락 아이디를 넣어주고 nSampleNum 에 Sample 번호를 넣어주면 해당 손가락의 바이너리 데이터를 가져올 수 있습니다.

BSTR TextEncodeFIR

텍스트로 인코딩 된 지문 데이터입니다.

long FIRLength

바이너리 형태의 지문 데이터의 길이입니다.

VARIANT FIR

바이너리 형태의 지문 데이터 입니다. 이 지문 데이터를 받아 오기 위해서는 FIRLength 만큼의 공간이 미리 정의되어 있어야 합니다.

long QuailtyInfo(long nFingerID, long nSampleNum)

지문의 품질값을 알려줍니다. 파라메터 nFingerID 에는 손가락 아이디를 넣어주고 nSampleNum 에 Sample 번호를 넣어주면 해당 손가락의 지문 품질값을 가져올 수 있습니다.

A.5.2 Method

Export(VARIANT storedFIR, long nDesFPDataType)

Description

NBioBSP 형식의 지문 데이터를 이용하여 다른 형식의 지문 데이터로 변경하고자 할 경우 사용하는 메소드 입니다.

Parameter

storedFIR : 변경하고자 하는 NBioBSP 형식의 지문 데이터입니다.

nDesFPDataType : 변경하여 얻고자 하는 지문 데이터의 형식입니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

FingerID : 추출된 지문 손가락 아이디

SampleNumber : 손가락 당 지문 개수

FPDataSize : 추출된 지문 데이터 크기

FPData : 추출된 지문 데이터

Import(BOOL bInitialize, long nFingerID, long nPurpose, long nSrcFPDataType,
long nFPDataSize, VARIANT FPData1, /*[in, optional]*/ VARIANT FPData2)

Description

다른 형식의 지문 데이터를 NBioBSP 형식의 지문 데이터로 변환 하고자 할 때 사용하는 메소드 입니다.

Parameter

bInitialize : 새로운 NBioBSP 형식의 지문 데이터를 생성할 지 결정합니다. TRUE 를 입력하면 새로운 지문 데이터를 생성하고, FALSE 를 입력하면 기존의 데이터에 첨부하게 됩니다.

nFingerID : 입력한 지문 데이터의 손가락 아이디를 입력합니다.

nSrcFPDataType : 원본 지문 데이터의 형식을 알려주기 위한 파라메터입니다.

nFPDataSize : 입력하고자 하는 원본 지문 데이터의 크기입니다.

FPData1 : 변경하고자 하는 원본 지문 데이터입니다.

FPData2 : 변경하고자 하는 두 번째 원본 지문 데이터입니다. 만약 SampleNumber 가 2 이면 이 데이터를 입력하고 그렇지 않으면 이 데이터는 의미가 없습니다. 이 파라메터는 옵션이므로 반드시 입력하지 않아도 됩니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

TextEncodeFIR : 텍스트로 인코딩 된 지문 데이터

FIRLength : 지문 데이터의 길이

FIR : 입력 받은 지문 데이터

CreateTemplate(VARIANT capturedFIR, VARIANT storedFIR, VARIANT payload)

Description

입력된 capturedFIR, storedFIR 을 이용하여 새로운 FIR 를 생성하는 메소드입니다.

Parameter

capturedFIR : 저장된 지문 데이터를 바이너리 또는 문자열로 입력합니다.

storedFIR : 저장된 지문 데이터를 바이너리 또는 문자열로 입력합니다.

Payload : 새롭게 생성되는 FIR 에 저장할 payload 값을 입력합니다.

이 파라메터는 옵션이므로 데이터가 없는 경우 입력하지 않아도 상관없습니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

TextEncodeFIR : 텍스트로 인코딩 된 지문 데이터

FIRLength : 지문 데이터의 길이

FIR : 입력 받은 지문 데이터

ImportImage(long ImageWidth, long ImageHeight, VARIANT RawImage)

Description

입력된 Raw 이미지에서 FIR 데이터를 생성하는 메소드입니다.

Parameter

ImageWidth : RawImage 의 Widht 값을 입력합니다.

ImageHeight : RawImage 의 Height 값을 입력합니다.

RawImage : RAW 이미지 바이너리 데이터를 입력합니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

TextEncodeFIR : 텍스트로 인코딩 된 지문 데이터

FIRLength : 지문 데이터의 길이

FIR : 입력 받은 지문 데이터

CheckQuality(VARIANT storedFIR, VARIANT auditFIR)

Description

지문의 품질을 검사하고자 할 경우 사용하는 메소드 입니다.

Parameter

storedFIR : 검사하고자 하는 지문의 FIR 데이터 입니다.

auditFIR : 검사하고자 하는 지문의 Audit 데이터 입니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

QualityInfo : 처리된 지문의 품질값

A.6 FPImage Object

지문을 캡쳐할 때 이 지문 이미지를 추출하기 위해 사용하는 오브젝트입니다.

A.6.1 Property

long ErrorCode

실행한 메소드 및 프로퍼티 설정 중에 발생한 에러에 대한 코드값을 가져올 때 사용합니다. 0 인 경우 성공 이외의 값은 실패를 나타냅니다.

BSTR ErrorDescription

ErrorCode 를 통하여 받은 에러 코드값을 문자형식으로 출력하고자 할 때 사용합니다.

long TotalFingerCount

추출된 지문의 개수를 알려줍니다.

long FingerID(long nIndex)

추출된 지문 순서에 따른 손가락 아이디를 알려줍니다. 파라메터 nIndex 는 0 부터 TotalFingerCount – 1 까지의 값을 가집니다.

long ImageWidth

추출된 지문 이미지의 넓이를 알려줍니다. (단위 픽셀)

long ImageHeight

추출된 지문 이미지의 높이를 알려줍니다. (단위 픽셀)

VARIANT RawData(long nFingerID, /*[in, optional]*/ long nSampleNumber)

추출된 지문 이미지 데이터를 바이너리 형태로 넘겨 줍니다.

파라메터 nFingerID 는 추출하고자 하는 지문의 손가락 아이디를 입력하고 nSampleNumber 는 0 을 입력합니다.

nSampleNumber 는 옵션이므로 입력하지 않아도 상관 없습니다. (현재 nSampleNumber 는 의미가 없습니다.)

long ConvertImageWidth

ConvertWsqToRaw 메소드 수행 후 생성된 지문 이미지 데이터의 Width 값을 가집니다.

long ConvertImageHeight

ConvertWsqToRaw 메소드 수행 후 생성된 지문 이미지 데이터의 Height 값을 가집니다.

BSTR TextEncodeAuditData

추출된 지문 이미지 데이터를 Text Encode 형태로 가져올 때 사용합니다.

Long AuditdataLength

추출된 지문 데이터의 바이너리 크기를 알려줍니다.

VARIANT Auditdata

추출된 지문 이미지 데이터를 바이너리 형태로 가져올 때 사용합니다.

A.6.2 Method

Export()

Description

Enroll, Capture 등을 이용하여 획득한 지문의 이미지 데이터를 가져오고자 할 때 사용합니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

TotalFingerCount : 추출된 지문의 개수

FingerID : 추출된 지문 순서에 따른 손가락 아이디

ImageWidth : 추출된 지문의 넓이

ImageHeight : 추출된 지문의 높이

RawData : Raw 형식의 지문 이미지 데이터 출력

TextEncodeAuditData : Text Encode 형식의 지문 이미지 데이터 출력

AuditdataLength : Raw 형식의 전체 지문 이미지 데이터의 길이

Auditdata : Raw 형식의 전체 지문 이미지 데이터 출력

ExportEx(VARIANT auditData)

Description

입력된 auditData 의 지문 이미지 데이터를 가져오고자 할 때 사용합니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

TotalFingerCount : 추출된 지문의 개수

FingerID : 추출된 지문 순서에 따른 손가락 아이디

ImageWidth : 추출된 지문의 넓이

ImageHeight : 추출된 지문의 높이

RawData : Raw 형식의 지문 데이터 출력

TextEncodeAuditData : Text Encode 형식의 지문 이미지 데이터 출력

AuditdataLength : Raw 형식의 전체 지문 이미지 데이터의 길이

Auditdata : Raw 형식의 전체 지문 이미지 데이터 출력

Save(BSTR bszImgFilePath, long nImageType, long nFingerID, /*[in, optional]*/ long nSampleNumber)

Description

Enroll, Capture 등을 이용하여 획득한 지문의 이미지 데이터를 가져오고자 할 때 사용합니다.

Parameter

bszImgFilePath : 지문 이미지를 저장할 파일명을 입력합니다. (Full Path)

nImageType : 지문 이미지의 형식을 지정합니다.

nFingerID : 저장하고자 하는 손가락 아이디를 입력합니다.

nSampleNumber : 저장하고자 하는 지문의 SampleNumber 를 입력합니다. 이 값은 현재 사용하지 않습니다. 이 파라메터는 옵션이므로 입력하지 않아도 된다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

ConvertWsqToRaw(long WsqLen, VARIANT WsqImage)

Description

Wsq 데이터를 Raw 데이터 형식으로 변환합니다.

Parameter

WsqLen : Wsq 데이터의 길이를 입력합니다.

WsqImage : Wsq 데이터를 입력합니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

Return Value

VARIANT RawImage

메소드의 수행 결과로 Raw 데이터 형식으로 변환된 데이터가 반환됩니다.

ConvertRawToWsq(long ImageWidth, long ImageHeight, VARIANT RawImage, float fQuality)

Description

Raw 데이터를 Wsq 데이터로 변환합니다.

Parameter

ImageWidth : Raw 데이터의 Width 를 입력합니다.

ImageHeight : Raw 데이터의 Height 를 입력합니다.

RawImage : Raw 데이터를 입력합니다.

fQuality : Wsq Quality 를 지정합니다. (0.1 ~ 7.0)

0.75 값을 권장하며 이는 15:1 을 의미 합니다. 이 값이 0.75 이하로 설정 할 경우 이미지의 왜곡이 심해집니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

Return Value

VARIANT WsqImage

메소드의 수행 결과로 Raw 데이터 형식으로 변환된 데이터가 반환됩니다.

A.7 IndexSearch Object

다수의 지문 데이터를 메모리 데이터베이스에 저장한 후 특정 지문을 검색하여 찾아주는 기능을 수행하는 오브젝트입니다.

A.7.1 Property

long ErrorCode

실행한 메소드 및 프로퍼티 설정 중에 발생한 에러에 대한 코드값을 가져올 때 사용합니다. 0 인 경우 성공 이외의 값은 실패를 나타냅니다.

BSTR ErrorDescription

ErrorCode 를 통하여 받은 에러 코드값을 문자형식으로 출력하고자 할 때 사용합니다.

long Count

검색된 지문의 개수를 알려줍니다.

long MaxCandidatenumber

검색되어지는 지문 후보의 최대수를 설정합니다.

long GetDataCountFromDB

현재 메모리 DB 에 저장되어 있는 데이터의 개수를 알려줍니다.

BOOL CheckDataExistFromDB(long nUserID, long nFingerID, long nSampleNumber)

특정 데이터가 현재 메모리 DB 에 등록되어 있는지 확인 할 수 있습니다.

long UserID

Identify 를 수행하였을 경우 검색된 사용자 아이디를 알려줍니다.

long MaxSearchTime

Identify 를 수행하였을 경우 검색할 최대 시간을 설정합니다.

A.7.2 Property(CandidateList Object)

검색 결과 및 지문 등록 결과를 출력하기 위해 사용되는 오브젝트입니다. 이 오브젝트는 컬렉션으로 구현되어 별도의 선언 없이 사용할 수 있습니다.

long UserID

사용자 아이디를 출력합니다. 정수형 숫자만 사용됩니다.

long FingerID

손가락 아이디를 출력합니다.

long SampleNumber

지문 SampleNumber 를 알려줍니다. 0 또는 1 의 값을 가집니다.

A.7.3 Method

AddFIR (VARIANT FIR, long nUserID)

Description
지문 데이터를 추가하고자 할 때 사용합니다.

Parameter
FIR : Search 데이터베이스에 등록할 지문 데이터입니다.
nUserID : 지문의 소유자 아이디를 입력합니다. 정수형 숫자만 입력 가능합니다.

Relation Property
ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력.

RemoveData(long nUserID, long nFingerID, long nSampleNumber)

Description
특정 지문 데이터를 삭제하고자 할 때 사용합니다. 이 메소드는 한명의 사용자가 여러 개의 지문을 가지고 있는 경우 등록된 지문 중 한 개만을 삭제하고자 할 때 사용합니다.

Parameter
nUserID : 삭제하고자 하는 사용자의 아이디를 입력합니다. 정수형 숫자만 입력 가능합니다.
nFingerID : 삭제하고자 하는 손가락 아이디를 입력합니다.
nSampleNumber : 삭제하고자 하는 지문의 SampleNumber 를 입력합니다. 0 또는 1 의 값을 가질 수 있습니다.

Relation Property
ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력.

RemoveUser(long nUserID)

Description
해당 사용자가 등록한 지문을 모두 삭제하고자 할 때 사용합니다.

Parameter
nUserID : 삭제하고자 하는 사용자의 아이디를 입력합니다. 정수형 숫자만 입력 가능합니다.

Relation Property
ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력.

IdentifyUser(VARIANT storedFIR, long nSecuLevel)

Description
해당 지문을 가지고 있는 사용자를 검색하는 메소드 입니다.

Parameter
storedFIR : 검색을 원하는 지문 데이터입니다.
nSecuLevel : 검색을 할 때 보안 수준을 결정합니다. 숫자가 높을수록 결과가 정확해 지지만 본인 거부율이 높아질 수 있습니다.

Relation Property
ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력.
UserID : 검색된 사용자 아이디

ClearDB()

Description
Search 데이터베이스를 삭제할 때 사용합니다.

Relation Property
ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력.

SaveDBToFile(BSTR bszFilePath)

Description
메모리에 저장된 데이터베이스는 서버가 종료되면 데이터가 손실됩니다. 이 때문에 이 메소드를 이용하여 메모리 데이터베이스를 파일로 저장할 수 있습니다.

Parameter
bszFilePath : 메모리 데이터베이스를 저장할 파일명을 입력합니다 (Full Path)

Relation Property
ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력.

LoadDBFromFile(BSTR bszFilePath)

Description
SaveDBToFile 메소드를 이용하여 저장된 메모리 데이터베이스를 다시 메모리로 읽어오기 위한 기능을 수행합니다.

Parameter
bszFilePath : 저장된 메모리 데이터베이스의 파일명을 입력합니다 (Full Path)

Relation Property
ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력.

A.8 NSearch Object

다수의 지문 데이터를 메모리 데이터베이스에 저장한 후 특정 지문을 검색하여 찾아주는 기능을 수행하는 오브젝트입니다. 자세한 설명은 NSearch 매뉴얼을 참고하세요.

A.8.1 property

long ErrorCode

실행한 메소드 및 프로퍼티 설정 중에 발생한 에러에 대한 코드값을 가져올 때 사용합니다. 0 인 경우 성공 이외의 값은 실패를 나타냅니다.

BSTR ErrorDescription

ErrorCode 를 통하여 받은 에러 코드값을 문자형식으로 출력하고자 할 때 사용합니다.

long Count

검색된 지문의 개수를 알려줍니다.

long MaxCandidatenumber

검색되어지는 지문 후보의 최대수를 설정합니다.

long GetDataCountFromDB

현재 메모리 DB 에 저장되어 있는 데이터의 개수를 알려줍니다.

BOOL CheckDataExistFromDB(long nUserID, long nFingerID, long nSampleNumber)

특정 데이터가 현재 메모리 DB 에 등록되어 있는지 확인 할 수 있습니다.

long UserID

Identify 를 수행하였을 경우 검색된 사용자 아이디를 알려줍니다.

A.8.2 property (CandidateList Object)

검색 결과 및 지문 등록 결과를 출력하기 위해 사용되는 오브젝트입니다. 이 오브젝트는 컬렉션으로 구현되어 별도의 선언 없이 사용할 수 있습니다.

long UserID

사용자 아이디를 출력합니다. 정수형 숫자만 사용됩니다.

long FingerID

손가락 아이디를 출력합니다.

long SampleNumber

지문 SampleNumber 를 알려줍니다. 0 또는 1 의 값을 가집니다.

long ConfidenceLevel

검색을 수행한 경우 검색된 지문의 매칭 점수를 알려줍니다.

1 – 9 까지의 값을 가진다. 숫자가 높을수록 지문이 유사한 것입니다.

A.8.3 Method

AddFIR (VARIANT FIR, long nUserID)

Description

지문 데이터를 추가하고자 할 때 사용합니다.

Parameter

FIR : Search 데이터베이스에 등록할 지문 데이터입니다.

nUserID : 지문의 소유자 아이디를 입력합니다. 정수형 숫자만 입력 가능합니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

CandidateList Object : 입력된 결과값을 출력해 줍니다. 지문 등록 결과 출력 시에는 ConfidenceLevel 값은 출력되지 않습니다.

RemoveData(long nUserID, long nFingerID, long nSampleNumber)

Description

특정 지문 데이터를 삭제하고자 할 때 사용합니다. 이 메소드는 한명의 사용자가 여러 개의 지문을 가지고 있는 경우 등록된 지문 중 한 개만을 삭제하고자 할 때 사용합니다.

Parameter

nUserID : 삭제하고자 하는 사용자의 아이디를 입력합니다. 정수형 숫자만 입력 가능합니다.

nFingerID : 삭제하고자 하는 손가락 아이디를 입력합니다.

nSampleNumber : 삭제하고자 하는 지문의 SampleNumber 를 입력합니다. 0 또는 1 의 값을 가질 수 있습니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

RemoveUser(long nUserID)

Description

해당 사용자가 등록한 지문을 모두 삭제하고자 할 때 사용합니다.

Parameter

nUserID : 삭제하고자 하는 사용자의 아이디를 입력합니다. 정수형 숫자만 입력 가능합니다.

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

SearchData(VARIANT storedFIR)

Description
해당 지문과 유사한 지문을 모두 검색하는 메소드 입니다.

Parameter
storedFIR : 검색을 원하는 지문 데이터입니다.

Relation Property
ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력.
CandidateList Object : 검색된 결과값을 출력해 줍니다.

IdentifyUser(VARIANT storedFIR, long nSecuLevel)

Description
해당 지문을 가지고 있는 사용자를 검색하는 메소드 입니다.

Parameter
storedFIR : 검색을 원하는 지문 데이터입니다.
nSecuLevel : 검색을 할 때 보안 수준을 결정합니다. 숫자가 높을수록 결과가 정확해 지지만 본인 거부율이 높아질 수 있습니다.

Relation Property
ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력.
UserID : 검색된 사용자 아이디

ClearDB()

Description
Search 데이터베이스를 삭제할 때 사용합니다.

Relation Property
ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0
ErrorDescript : ErrorCode 의 값을 문자열로 출력.

SaveDBToFile(BSTR bszFilePath)

Description

메모리에 저장된 데이터베이스는 서버가 종료되면 데이터가 손실됩니다. 이 때문에 이 메소드를 이용하여 메모리 데이터베이스를 파일로 저장할 수 있습니다.

Parameter

bszFilePath : 메모리 데이터베이스를 저장할 파일명을 입력합니다 (Full Path)

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

LoadDBFromFile(BSTR bszFilePath)

Description

SaveDBToFile 메소드를 이용하여 저장된 메모리 데이터베이스를 다시 메모리로 읽어오기 위한 기능을 수행합니다.

Parameter

bszFilePath : 저장된 메모리 데이터베이스의 파일명을 입력합니다 (Full Path)

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.

ImportIndexSearchDB(BSTR bszFilePath)

Description

IndexSearch DB 의 SaveDBToFile 메소드를 이용하여 저장된 메모리 데이터베이스를 메모리로 읽어오기 위한 기능을 수행합니다.

Parameter

bszFilePath : 저장된 메모리 데이터베이스의 파일명을 입력합니다 (Full Path)

Relation Property

ErrorCode : 실행 결과를 출력. 성공인 경우 : NBioAPIERROR_NONE=0

ErrorDescript : ErrorCode 의 값을 문자열로 출력.