CROSSMATCH

U.are.U® SDK v3

Platform Guide for Windows

Contents

Introduction

THIS CHAPTER PROVIDES AN OVERVIEW OF THE CHAPTERS IN THE U.ARE.U SDK PLATFORM GUIDE FOR WINDOWS.

This manual describes how to use the U.are.U SDK to develop applications for devices based on Microsoft Windows. The U.are.U SDK is available for multiple platforms and this document describes issues specific to developing applications for devices based on Microsoft Windows.

Introduction (this chapter) describes how to get the latest version of this documentation.

Installation provides instructions for installing on your development system and on the target machine.

Developing Applications with C/C++ lists system requirements for developing and running applications in C/C++ and describes the sample application.

Developing Applications with .NET lists system requirements for developing and running applications with .NET and describes the .NET sample applications for VB.NET and C#.

Developing Applications with ActiveX / .NET lists system requirements for developing and running applications using ActiveX, and other ActiveX information.

Developing Applications with Java lists system requirements for developing and running applications using Java, provides additional installation instructions and describes the Java sample application.

Developing Applications with JavaScript lists system requirements for developing and running applications using Java, provides additional installation instructions and describes the JavaScript sample application.

Developing Applications with JavaPOS provides information on using the JavaPOS-compliant API built on the U.are.U framework.

Developing Applications with OPOS provides information on using the JavaPOS-compliant API built on the U.are.U framework.

Redistribution describes the merge modules that are provided to help you redistribute applications built using the U.are.U SDK.

For a detailed description of the SDK, consult the U.are.U SDK Developer Guide.

Getting Updated Documentation

If you are viewing this guide from the download package for the SDK, you may want to check online at our website for an updated version of this document at

http://www.crossmatch.com/Support/Reference-Material/SDK-Reference-Material/

Except as noted in the language-specific chapters, the installation process is the same for development on all Windows systems.

Installing on the Development and Target Systems

There are two steps to the installation:

1   Installing on the development system

2   Installing on the Windows device (the target hardware)

These steps are described below. Note that the same distribution file is used for installing on both development and test/target systems -- during installation, different files are copied to the product folder depending on how you install.

Step 1: Installing on the Development System (SDK Installation)

To install the SDK on your development system:

1   Unzip the distribution file into a folder.

2   For 32-bit systems, run `SDK\x86\setup.exe`
    For 64-bit systems, run `SDK\x64\setup.exe`

The installer copies all necessary files to the selected folder (by default, the product folder is `Program Files\ DigitalPersona\U.are.U SDK`). The files installed on the developer's machine are located in the following folders within the main product folder:

Folder	Contents
Bin	SDK components, DLLs and Multiple User Interface (mui) files for supported languages.
Include	Header files
Windows\Docs	End user license agreement (EULA) plus documentation:
	U.are.U SDK Developer Guide - Documentation for developers, describing use of all included APIs.
	Platform Guide for Windows - Windows-specific details for developers, including screenshots and instructions on features illustrated in the included sample applications.
	C_API - Doxygen for the C/C++ API
	Java_API - Javadoc for the Java API
	.NET_ActiveX_API - Doxygen for the .NET and ActiveX APIs
Windows\Lib	Runtime files:
	.NET - libraries and controls for .NET and ActiveX
	Java - Java and JavaPOS JAR files
	Win32 - libraries, OPOS libraries
	x64 - libraries for 64-bit processes

Folder	Contents
Windows\Samples\	Compiled sample applications:
	Bin
	.NET - .NET sample
	Java - Java sample
	JavaPOS - JavaPOS sample
	OPOS - OPOS sample
	WEB - Javascript sample
	Win32 - C/C++ sample
	x64 - C/C++ sample for 64-bit processes
	Source files for sample applications:
	Include - WTL80 files for C/C++ sample
	UareUSampleCpp - C/C++ sample
	UareUSampleCSharp - .NET /C# sample
	UareUSampleCSharp_CaptureOnly - .NET/C# sample that demonstrates only capture
	UareUSampleJava - Java sample
	UareUSampleJavaPOS - JavaPOS sample
	UareUSampleOPOS - OPOS sample
	UareUSampleVBNET - .NET /VBNET sample
	UareUSampleVBNET_CaptureOnly - .NET/VBNet sample that demonstrates only capture
	UareUSampleWEB - JavaScript sample

Step 2: Installing on the Target Hardware (RTE Installation)

To install the run-time environment on the target hardware platform:

1 Unzip the distribution file into a folder on the target machine.

2 For 32-bit systems, run `RTE\x86\setup.exe`
   For 64-bit systems, run `RTE\x64\setup.exe`

The installer copies all necessary files to the selected folder (by default, the product folder is `Program Files\ DigitalPersona\U.are.U RTE`). The files installed on the target machine are located in the following folder within the main product folder:

Folder	Contents
`Windows\Lib`	Runtime files for:
	▪ `.NET` - libraries and controls for .NET and ActiveX
	▪ `x64` - libraries for 64-bit processes
	▪ `Win32` - libraries, OPOS libraries
	▪ `Java` - Java and JavaPOS JAR files

DigitalPersona Authentication Service

The DigitalPersona Authentication Service is an optional feature that is installed by default during installation of the SDK, which allows multiple applications to work with the same fingerprint reader, sending captured fingerprint data to whichever application currently has the focus.

In most cases, the service is not required for authentication per se, since the FingerJet fingerprint matching engine is separate and not part of the authentication service, and the service only needs to be run when a fingerprint reader must serve more than one application at a time.

You can choose **not** to install the DigitalPersona Authentication Service, by deselecting the feature from the Custom Setup page of the installation wizard.
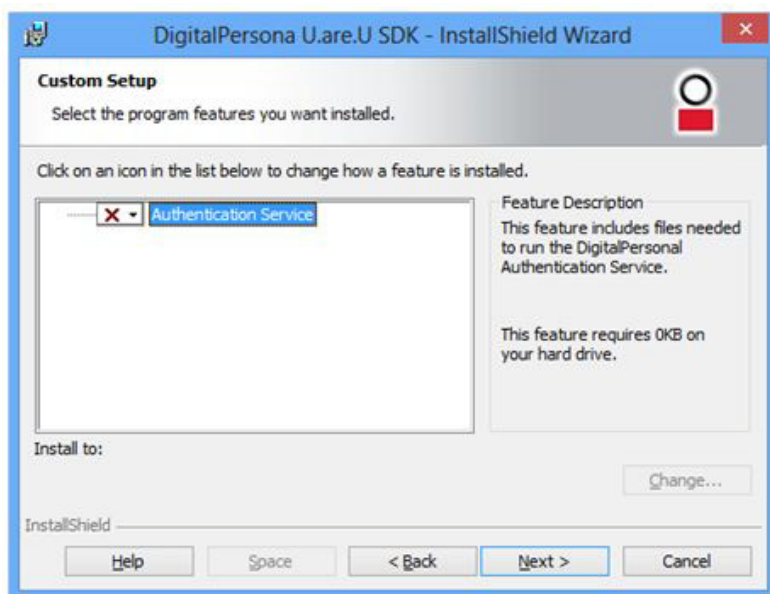
The DigitalPersona Authentication Service can be managed in the usual way via the Services Control Applet in the Microsoft Management Console by running services.msc as Administrator.

Uninstalling the SDK or RTE

If you need to uninstall the SDK or RTE, use the installation applet in the Control Panel.

Developing Applications with C/C++

THIS CHAPTER PROVIDES INFORMATION NECESSARY FOR DEVELOPING WINDOWS APPLICATIONS IN C/C++ WITH THE U.ARE.U SDK.

Pre-Requisites

This chapter assumes that you have a working knowledge of C/C++ and that you know how to develop for Microsoft Windows machines.

System Requirements

Development System

- Microsoft Windows XP Professional or higher, 32-bit or 64-bit
- Microsoft Visual Studio 2008 or later

Target Runtime Hardware (Windows machine)

The Windows-based machine that will run the application must be one of the following hardware platforms:

- Intel x86 architecture with CPU from 600MHz and at least 16MB of available RAM
- Intel x64 (x86-64) architecture with CPU from 600MHz and at least 16MB of available RAM

The file sizes are:

Function	x86	ARMv41
Capture runtime (drivers + SDK layer) - includes service	17 MB	16 MB
Fingerprint recognition runtime	0.8 MB	1 MB

In addition, the machine must also have:

- a USB port
- 16 Mb free memory

The SDK works on a variety of hardware and is intended to have a small footprint so that it can run even on minimal hardware. Less capable hardware will work, but response time may not be optimal.

The C/C++ Sample Application

The U.are.U SDK includes a sample application to demonstrate the features of the SDK. The sample application is located in the `Samples` folder. The compiled file, `UareUSample.exe` can be downloaded to your machine for testing. Depending on your version of Visual Studio, you can use one of the following:

- `UareUSample2008.vcproj.`
- `UareUSample2010.vcproj`
- `UareUSample2013.vcproj.`
- `UareUSample2017.vcproj.`

The application demonstrates the features of the SDK. When you launch the application, you see the main screen as shown below.
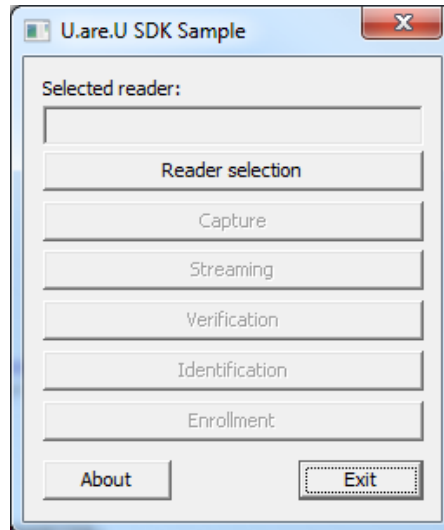
Click on **Reader Selection** to open a reader. All available readers will be displayed, as shown on the screen below.
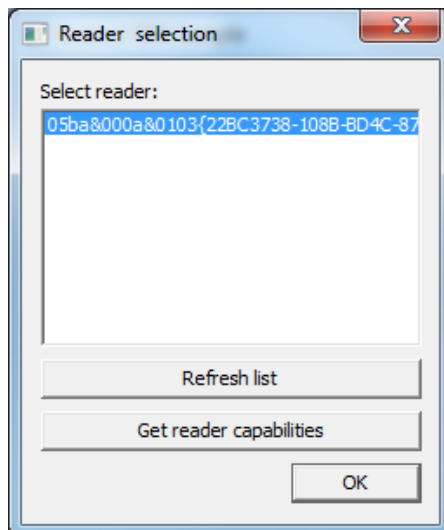
Clicking on the **Get reader capabilities** button will display additional information about the selected reader, as shown below.
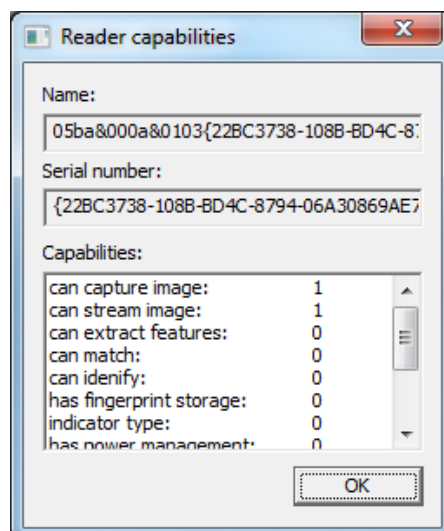
Click **OK** to return to the previous screen. Click **OK** to select the reader. At this point, you are returned to the main screen and all of the buttons are enabled.

Click on the **Capture** button to select capture mode. Place your finger on the reader to capture a fingerprint and display it on the screen as shown below.
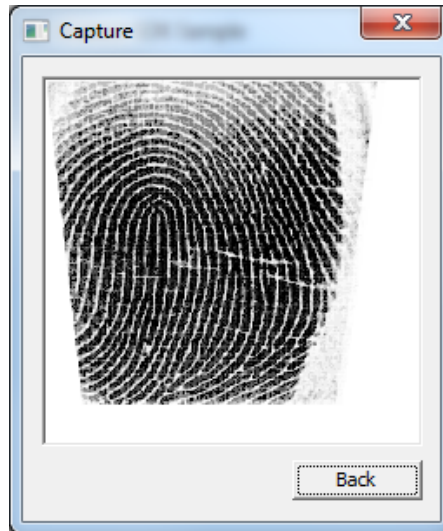
Click on the **Back** button to return to the main screen.

To see a demonstration of the streaming feature, click on the **Streaming** button to select streaming mode. Place your finger on the reader to capture a fingerprint and display it on the screen as shown below.
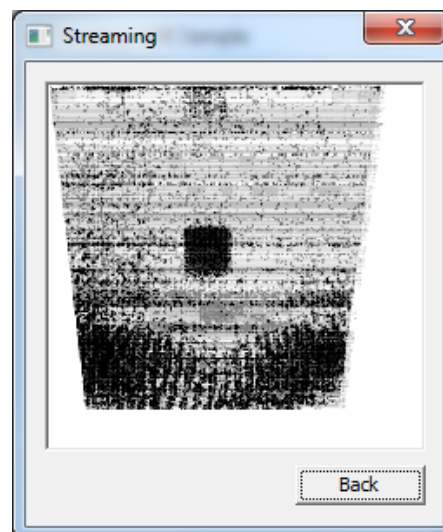
After you click on **Back**, you can click on the **Verification** button next. You will be prompted to place your finger on the reader. Then you can place a second finger on the reader. If you use the same finger, you will see a message that the fingerprints matched, as shown below.

```
┌─────────────────────────────────┐
│ ▪▫ Verification            ✕     │
├─────────────────────────────────┤
│  ┌──────────────────────────┐▲  │
│  │  fingerprint captured,    │   │
│  │  features extracted.      │   │
│  │                           │   │
│  │  Fingerprints matched.    │   │
│  │                           │   │
│  │                           │   │
│  │  dissimilarity score: 0x0.│   │
│  │  false match rate: 0.000000e+000.│ │
│  │                           │   │
│  │                           │   │
│  │  Verification started     │   │
│  │                           │   │
│  │  Put any finger on the reader...│ │
│  │                          ▼│   │
│  └──────────────────────────┘   │
│                                  │
│                  ┌───────────┐   │
│                  │   Back    │   │
│                  └───────────┘   │
└─────────────────────────────────┘
```

When you click on **Back** you will return to the main screen.

Click on **Identification** to test the next component of the sample program. You will be prompted to place a thumb, index finger, etc. on the reader. Then you will be prompted to provide another finger and you will receive a message indicating if there was a match and which finger was detected, as shown in the image below.

```
┌─────────────────────────────────┐
│ ▪▫ Identification          ✕     │
├─────────────────────────────────┤
│  ┌──────────────────────────┐▲  │
│  │  Put any finger on the reader...│ │
│  │     fingerprint captured, │   │
│  │     features extracted.   │   │
│  │                           │   │
│  │  Fingerprint identified, your index finger│ │
│  │  dissimilarity score: 0x0.│   │
│  │  false match rate: 0.000000e+000.│ │
│  │                           │   │
│  │                           │   │
│  │  Identification started   │   │
│  │                           │   │
│  │  Put your thumb on the reader...│ │
│  │                          ▼│   │
│  └──────────────────────────┘   │
│                                  │
│                  ┌───────────┐   │
│                  │   Back    │   │
│                  └───────────┘   │
└─────────────────────────────────┘
```
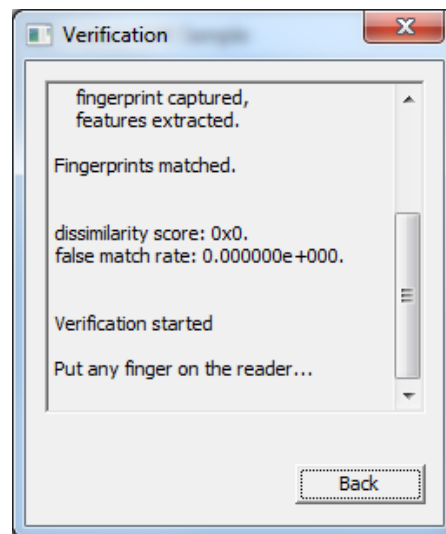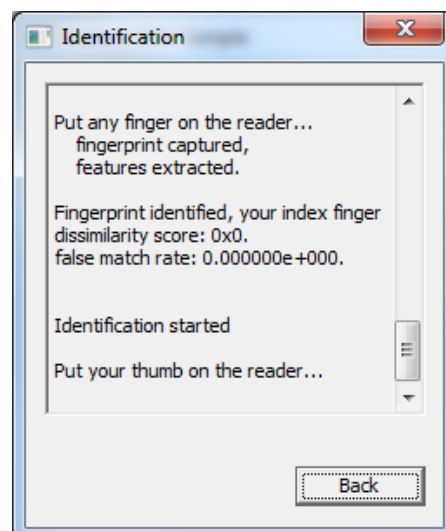
Next, click on the **Enrollment** button from the main screen.
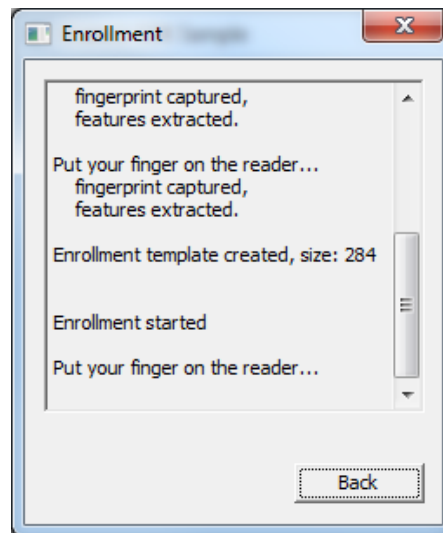
This feature simply captures a fingerprint, creates a FMD, and displays a message on the screen to confirm that it was successful.

Note that if you unplug the reader, you will receive an error message and the associated error code.

Developing Applications with .NET

THIS CHAPTER PROVIDES INFORMATION NECESSARY FOR DEVELOPING WINDOWS APPLICATIONS IN .NET WITH THE U.ARE.U SDK.

Pre-Requisites

This chapter assumes that you have a working knowledge of .NET and that you know how to develop for Microsoft Windows machines. You must also have tools and knowledge for your target language, typically C# or Visual Basic (VB.NET).

System Requirements

Development System

- Microsoft Windows XP Professional or higher, 32-bit or 64-bit
- Microsoft Visual Studio 2008 or later
- .NET Framework 2.0

Target Runtime Hardware (Windows machine)

The Windows-based machine that will run the application must be one of the following hardware platforms:

- Intel x86 architecture with CPU from 600MHz and at least 96MB of available RAM
- Intel x64 (x86-64) architecture with CPU from 600MHz and at least 96MB of available RAM

The file sizes (wrapper only, not including the C/C++ API) are:

- Capture runtime (drivers + SDK layer) with fingerprint recognition: 54 KB
- Enrollment and identification controls: 203 KB

In addition, the machine must also have:

- a USB port

The SDK works on a variety of hardware and is intended to have a small footprint so that it can run even on minimal hardware. Less capable hardware will work, but response time may not be optimal.

Static libraries and DLLs

The SDK installation installs

- DPCtlUruNet.dll - .Net GUI controls
- DPUruNet.dll - .Net API Library

The .NET Sample Application

The U.are.U SDK includes two .NET sample applications that demonstrate the features of the SDK.

- The C# sample application is located in the `Samples/UareUSampleCSharp` folder. The compiled file, `UareUSampleCSharp.exe` can be downloaded to your device for testing or you can use `UareUSampleCSharp.csproj` in Visual Studio.
- The VB.NET sample application is located in the `Samples/UareUSampleVBNET` folder. The compiled file, `UareUSampleVBNET.exe` can be downloaded to your device for testing or you can use `UareUSampleVBNET.vbproj` in Visual Studio.

The interfaces for the VB.NET and C# sample applications are identical, except for the text on the title bar of the opening screen.

The sample application demonstrates the features of the SDK. When you launch the application, you see the main screen as shown below.
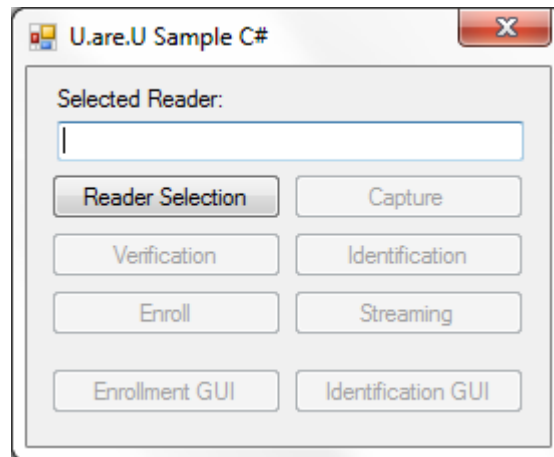
The sample program demonstrates:

- How to capture fingerprints both in scan mode and in streaming mode
- How to enroll a subject finger
- How to identify a fingerprint
- How to verify a fingerprint
- The built-in control for enrollment
- The built-in control for identification

Selecting a Reader

Click on **Reader Selection** to open a device. All available devices will be displayed in the pull-down list, as shown on the screen below.
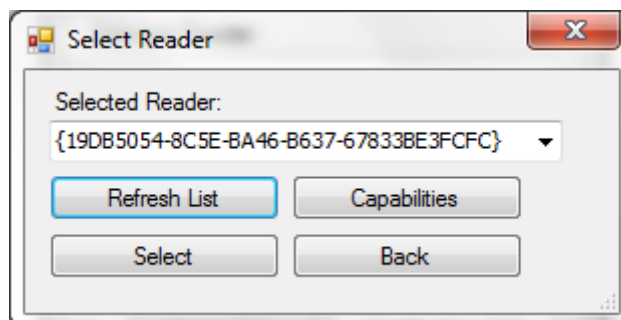
If you choose a reader from the list and click on the **Capabilities** button the application will display additional information about the selected reader, as shown below.
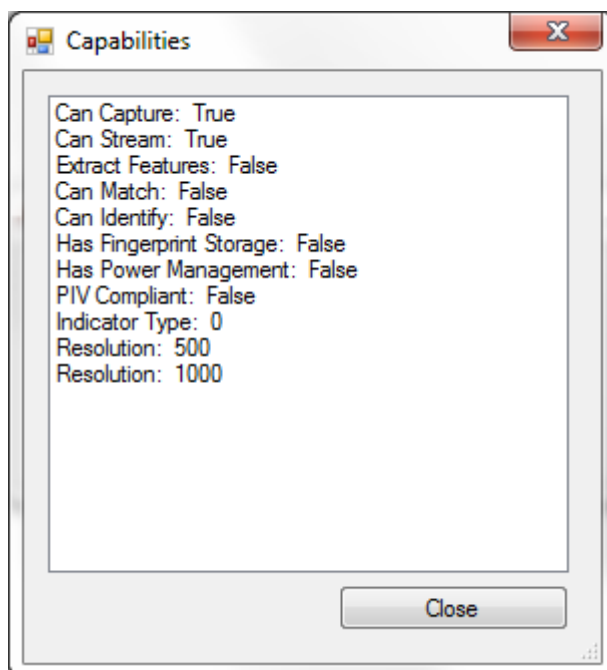
Click **Close** to return to the previous screen. Click **Select** to select the device. At the point, you are returned to the main screen and all of the buttons are enabled.

Capturing a Fingerprint

Click on the **Capture** button to select capture mode. Place your finger on the reader to capture a fingerprint and display it on the screen as shown below.
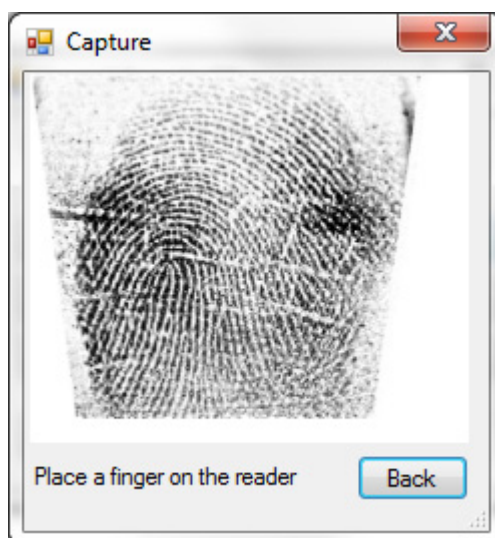
While the reader is in capture mode, you can capture repeatedly by placing a finger onto the reader. Click on the **Back** button to return to the main screen.

Testing Streaming Mode

Click on the **Streaming** button from the main dialog to select streaming capture mode. Place your finger on the reader to capture a fingerprint and display it on the screen as shown below.
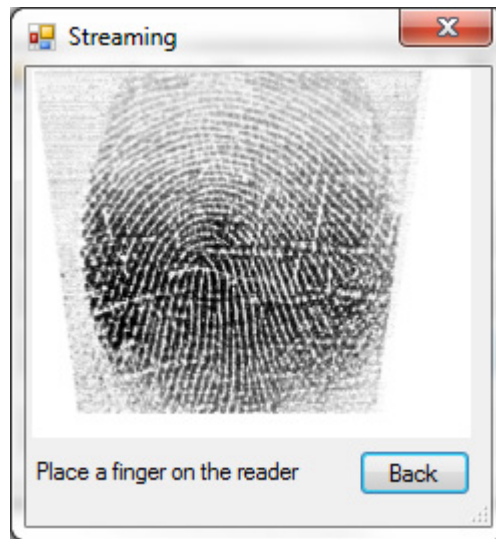
While the reader is in streaming mode, you can capture repeatedly by placing a finger on the reader. Click on the **Back** button to return to the main screen.

Enrolling a Finger

Click on **Enrollment** to begin enrolling the first test subject.

You will be prompted to scan the first finger for enrollment, as shown below.
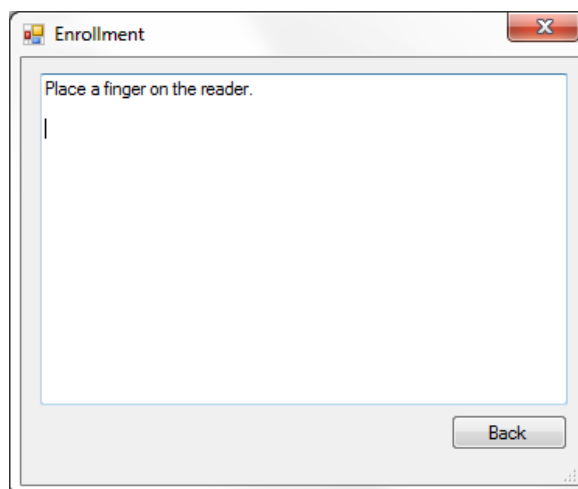
After that finger is successfully scanned, you will be prompted to scan a second finger. The sample application will prompt you to scan additional fingers until a sufficient number of high quality scans are complete. (The number of fingers requested will vary depending on the image scans - the enrollment functions will continue to request scans until an acceptable enrollment record has been created.)

Once the enrollment is complete, you will see confirmation that the enrollment process is finished, as shown in the screen below. In this case, four fingerprint scans were sufficient.
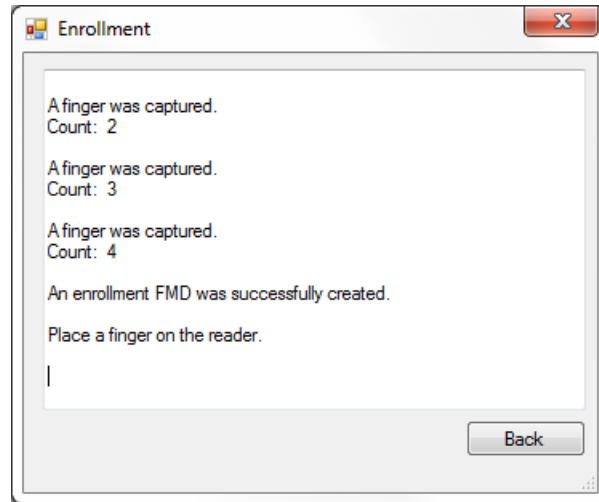
Note that the enrollment FMD that is created is stored in memory only and will be deleted when you click the **Back** button.

Identifying a Fingerprint

To test the identification feature, click on the **Identification** button. Recall that identification is a 1-to-many comparison where the application would normally search through all of the enrolled fingers to find a match. For this sample, we don't have any enrolled fingers, so you will be prompted to provide a finger. Then you will be prompted to provide another finger and you will receive a message indicating if there was a match, as shown in the image below
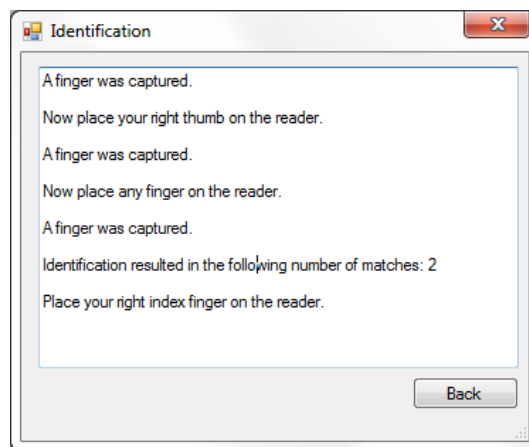
To exit identification mode, click on the **Back** button.

Verifying a Fingerprint

To test the verification feature, click on the **Verification** button. Recall that verification is a 1-to-1 comparison where the application matches against a specified fingerprint. When you click the **Verification** button, you will be prompted to place your finger on the reader. In the screen below, we have tried to verify a finger.
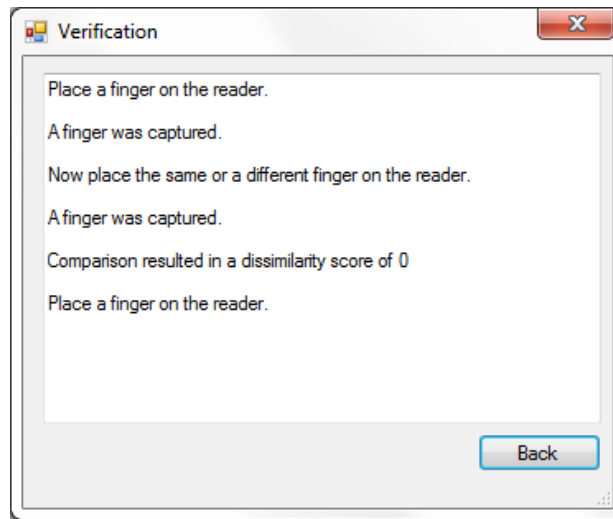
To exit identification mode, click on the **Back** button.

Testing the Enrollment UI Control

If you look at the sample code, you will see that enrollment (as described above) calls functions in the SDK. An alternate way to use the .NET SDK is to use the pre-built control for enrollment. To try out the pre-built control, click on the **Enrollment GUI** button. This will launch the control. In our sample (shown below), we have the control at the left and demo/debug info at the right side of the window.

If you click on a finger, for example the index finger of the right hand, you will be prompted to scan your finger.

As you scan your finger, you can see the events and status information on the right, as shown below.
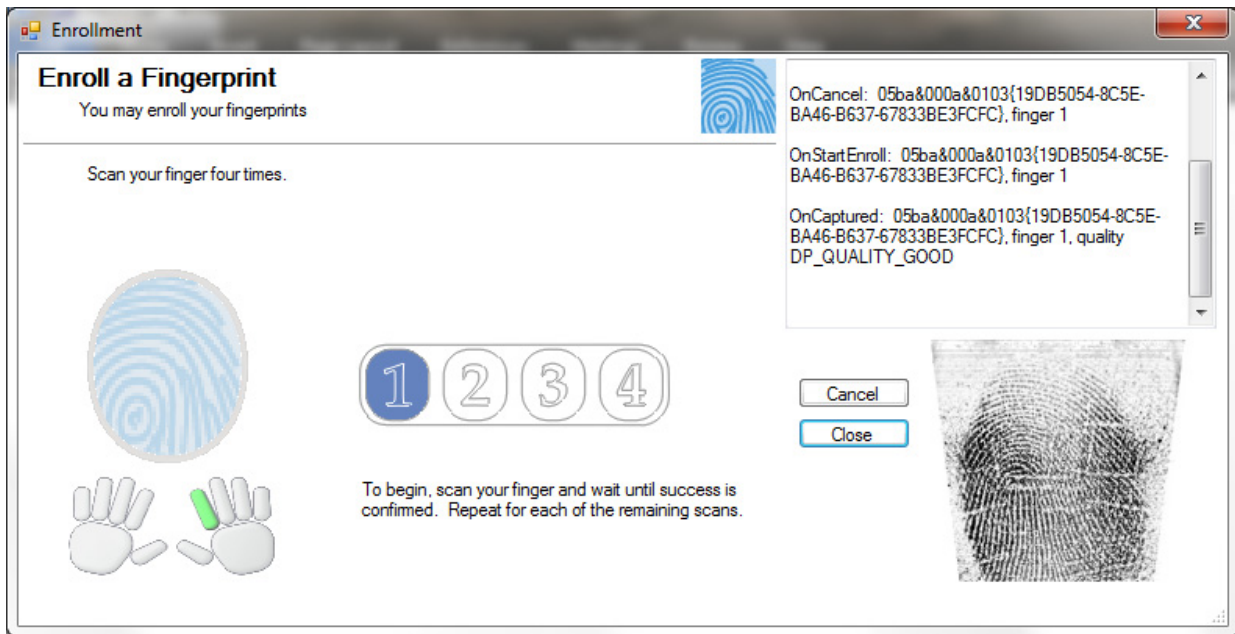
If you click on the window's **Cancel** button, it will cancel the enrollment of the current finger.

Once the enrollment process is complete, you will be returned to the opening screen of the enrollment process. Note that the finger you enrolled now shows in green and you can click on another finger to enroll another fingerprint.

To delete an enrolled fingerprint, click on an enrolled finger in this dialog and you will be prompted to confirm that you wish to delete the fingerprint for the finger that you clicked on.

The enrollment record created by the control is stored in memory until you exit from the sample application.

To return to the sample application, click **Close**.

Testing the Identification UI Control

If you look at the sample code, you will see that identification (as described above) calls functions in the SDK. An alternate way to use the .NET SDK is to use the pre-built control for identification. To try out the pre-built control, click on the **Identification GUI** button. This will launch the control. In our sample (shown below), we have the control at the left and demo/debug information at the right side of the window.
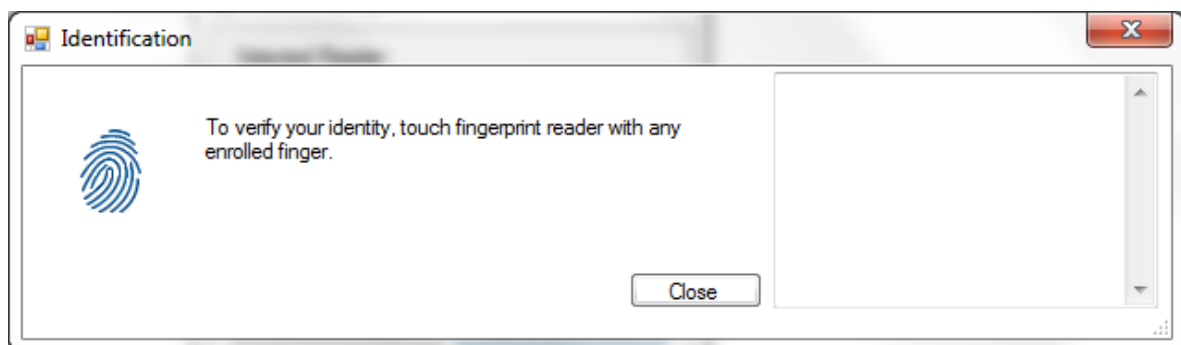
(Note that the identification is performed against the fingerprints enrolled through the **Enrollment GUI** feature previously. When you exit from the sample application, all enrollment records are deleted.)

If the identification succeeds, you will see the details in the status box at the right. The example below shows the result of a successful identification.
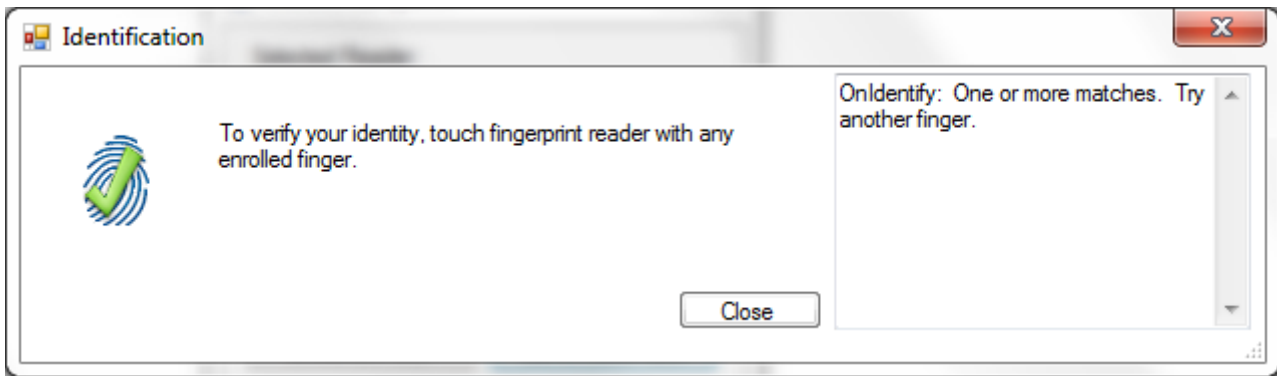
Note that if you unplug the device, you will receive an error message and the associated error code.

To exit the sample application, click on the **Close** button.

Developing Applications with ActiveX/.NET

This chapter provides information necessary for developing windows applications in .NET with ActiveX through the U.are.U SDK.

Pre-Requisites

This chapter assumes that you have a working knowledge of .NET and ActiveX and that you know how to develop for Microsoft Windows machines. You must also have tools and knowledge for your target language (typically C# or Visual Basic).

Overview

The ActiveX option has the same requirements and installation as the .NET components. The file sizes are approximately 15K larger than the .NET files.

Note that ActiveX does not work with Mozilla Firefox and Google Chrome browsers.

Static libraries and DLLs

The following DLLs are registered upon installation and may be imported into a Visual Basic 6.0 or Delphi project:

- DPXUru.dll– ActiveX
- DPCtlXUru.dll – ActiveX GUI controls

ActiveX Control Unique Identifiers

Use the following unique identifiers to access the U.are.U ActiveX controls. ActiveX control are run in a variety of different environments, such as on an HTML page, through a Visual Basic 6.0 application, or a Delphi application.

[Guid("977AA4C5-6737-4E79-BBAD-657A94362D56")] - EnrollmentXControl

[Guid("DB3C2981-2434-403B-B3DE-71A34741D1AB")] - IdentificationXControl

[Guid("EF84894C-1C02-4ECD-8602-E64D85E97557")] - XFmd

[Guid("36C6859B-8543-4DBF-9C37-24E30CB6CAFA")] - XFmv

[Guid("9D324B94-0931-483C-90DA-2A25AF2D5848")] - XFiv

[Guid("803FCBB9-D4BA-48F1-BB36-C6040783B3D1")] - XImporter

[Guid("733A2D1B-9F3D-423D-8700-4F2C8E88EAF9")] - XFeatureExtraction

[Guid("A1589E23-FE6E-43D8-9EDF-93142671C47A")] - XEnrollment

[Guid("C864A916-E288-439B-8054-C695C9677D84")] - XComparison

[Guid("C4287526-1485-48CB-99BB-6CC4A3552B81")] - XReader

[Guid("CAC5592F-EBA5-487C-AF8A-F35A70FAA33B")] - XReaderCollection

Developing Applications with Java

THIS CHAPTER PROVIDES INFORMATION NECESSARY FOR DEVELOPING WINDOWS APPLICATIONS IN JAVA WITH THE U.ARE.U SDK.

Pre-Requisites

This chapter assumes that you have a working knowledge of Java and that you know how to develop for Microsoft Windows machines.

System Requirements

Development System

- Microsoft Windows XP Professional or higher, 32-bit or 64-bit
- Microsoft Visual Studio 2008 or later
- Java SE 7 (JDK 7) or newer

Target Runtime Hardware (Windows machine)

The Windows-based machine that will run the application must be one of the following hardware platforms:

- Intel x86 architecture with CPU from 600MHz and at least 96MB of available RAM
- Intel x64 (x86-64) architecture with CPU from 600MHz and at least 96MB of available RAM

Function	x86	x64
Capture runtime (drivers + SDK layer) with fingerprint recognition	17 MB	16 MB

In addition, the machine must also have:

- a USB port

The SDK works on a variety of hardware and is intended to have a small footprint so that it can run even on minimal hardware. Less capable hardware will work, but response time may not be optimal.

Extra Installation Steps

After installing as described in Installing on the Development and Target Systems on page 5, you must do the following additional steps on both the development and target machines:

1 Copy the files in these two folders: `U.are.U SDK\Windows\Lib\Java` and `U.are.U SDK\Windows\Lib\ <x86 or x64>` to the location of your choice.

2 Make sure that `dpuareu.jar` is in the classpath and `dpuareu_jni.dll` is accessible by JVM. For example:

```
java.exe -classpath ".;C:\Program Files\DigitalPersona\U.areU SDK\Windows\Lib\Java\
dpuareu.jar" -Djava.library.path="C:\Program Files\DigitalPersona\U.areU SDK\Windows\
Lib\win32" UareUSampleJava
```

The Java Sample Application

The U.are.U SDK includes a sample application to demonstrate the features of the SDK when using the Java API. The sample application is located in the `Samples` folder. The compiled file, `UareUSampleJava.exe` can be downloaded to your machine for testing or you can compile it for yourself using the source files provided.

The application demonstrates the features of the SDK. When you launch the application, you see the main screen as shown below.
The sample program demonstrates:

- How to enroll a finger
- How to identify a fingerprint
- How to verify a fingerprint
- The built-in control for enrollment
- The built-in control for identification
- How to use the streaming feature to display live fingerprint data on the screen

Selecting a Reader

To choose the reader, click on the **Select new reader** button. You will see a list of available readers and you can choose the desired device, as shown below:

Simply clicking on a reader selects it.

To see the reader capabilities, click on the **Get reader capabilities** button.

The capabilities will be displayed, as shown in the image to the right.

Click on the **Back** button to continue.

Click on the **Back** button from the previous screen to return to the main screen.
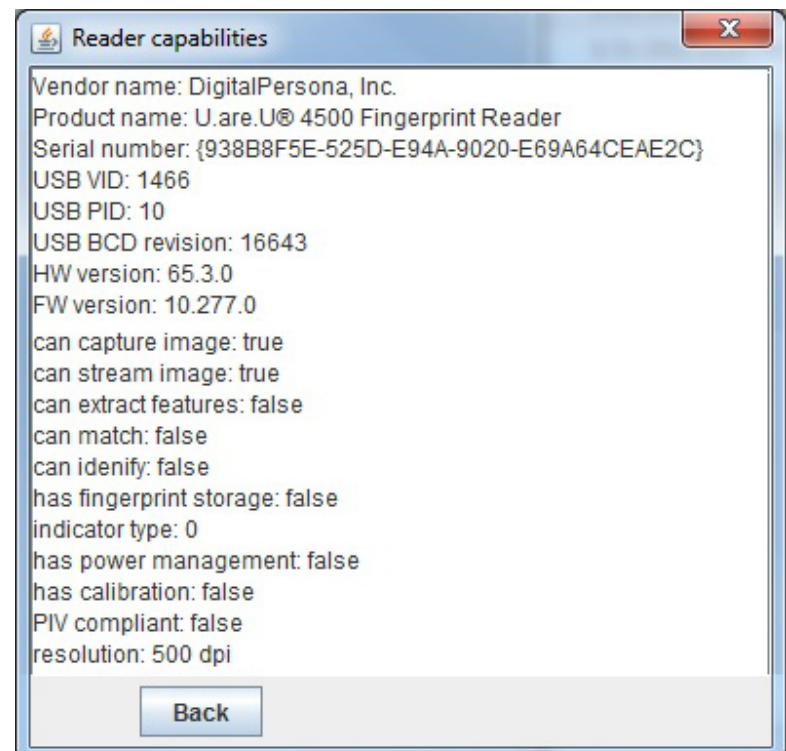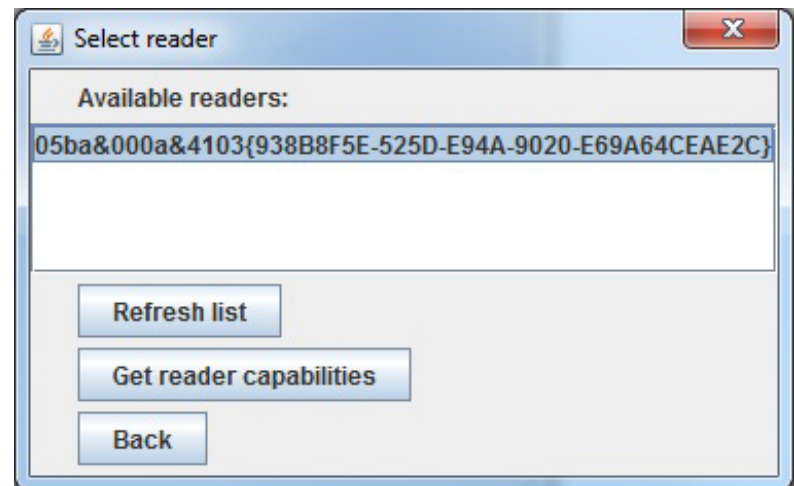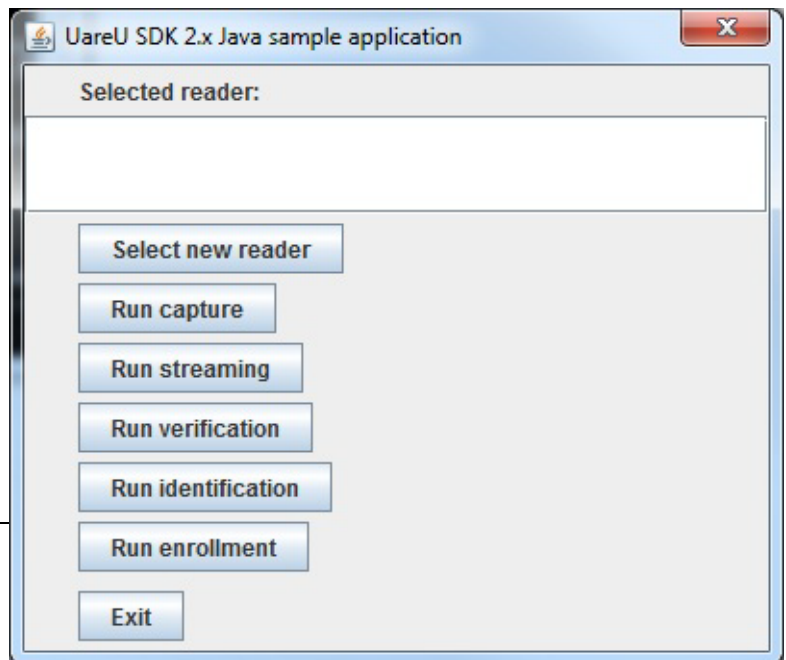
Enrolling a Finger

Click on **Run enrollment** to begin enrolling a test subject.

You will see a series of prompts to scan fingers for enrollment, as shown below.
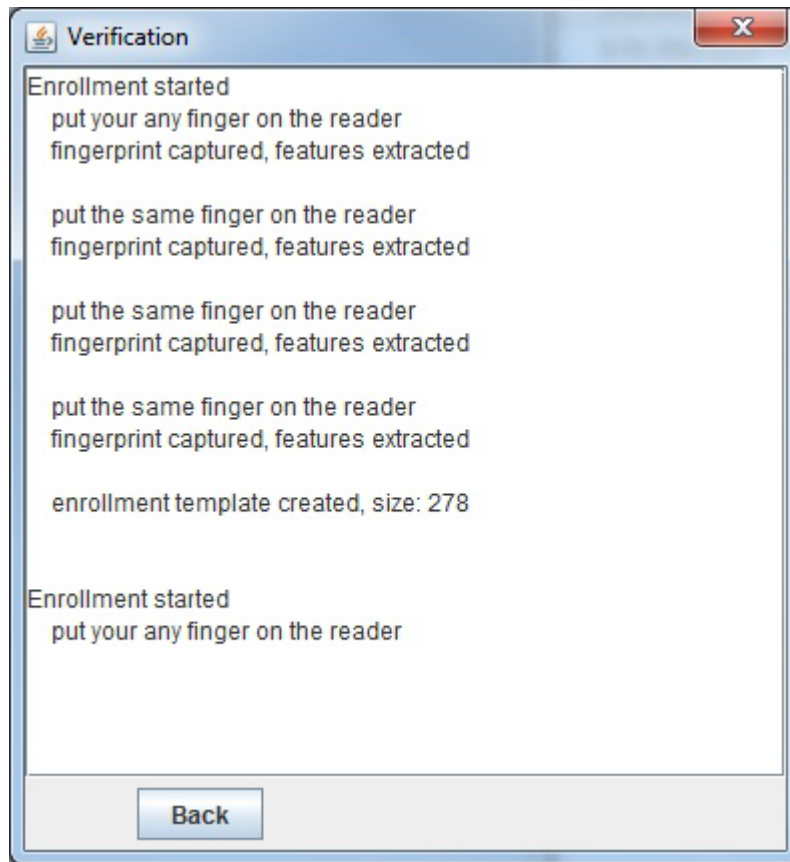
After the first finger is successfully scanned, you will be prompted to scan additional fingers until a sufficient number of high quality scans are complete. The number of fingers requested will vary depending on the image scans - the enrollment functions will continue to request scans until an acceptable enrollment record has been created.

When enrollment is complete, click **Back** to return to the main screen. (Note that enrollment FMDs that are created are not stored.)

Identifying a Fingerprint

To test the identification feature, click on the **Run identification** button. Recall that identification is a 1-to-many comparison where the application searches through all of the enrolled fingers to find a match. For this example, we do not have a stored database, so the sample application first prompts you to place a few fingers on the reader so that the application has some fingerprints to check against.

After the application scans four fingers, you will be prompted to place any finger on the reader to identify against the fingers that were just scanned. If you place a finger that was previously scanned on the reader, you will see that a match was found. In the screen image below, we successfully identified a user.
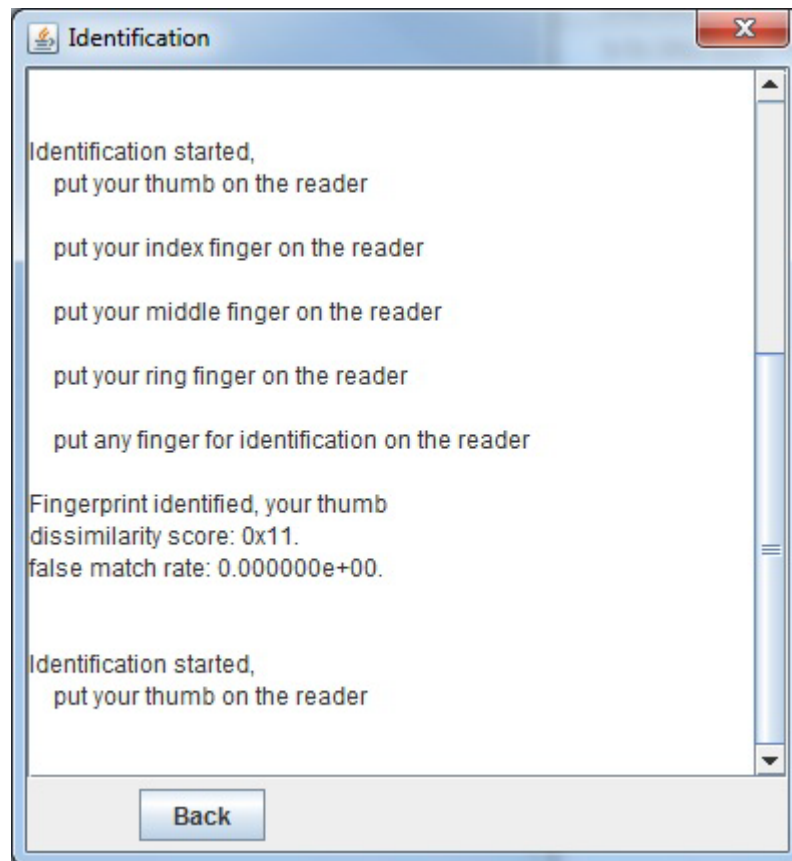
To exit identification mode, click on the **Back** button.

Verifying a Fingerprint

To test the verification feature, click on the **Run verification** button. Recall that verification is a 1-to-1 comparison where the application matches against a specified fingerprint. When you click the **Run verification** button, you will be prompted to place your finger on the reader. Then you will be prompted to place the same finger or another finger, to verify against the first finger. In the screen below, we have successfully verified a user.
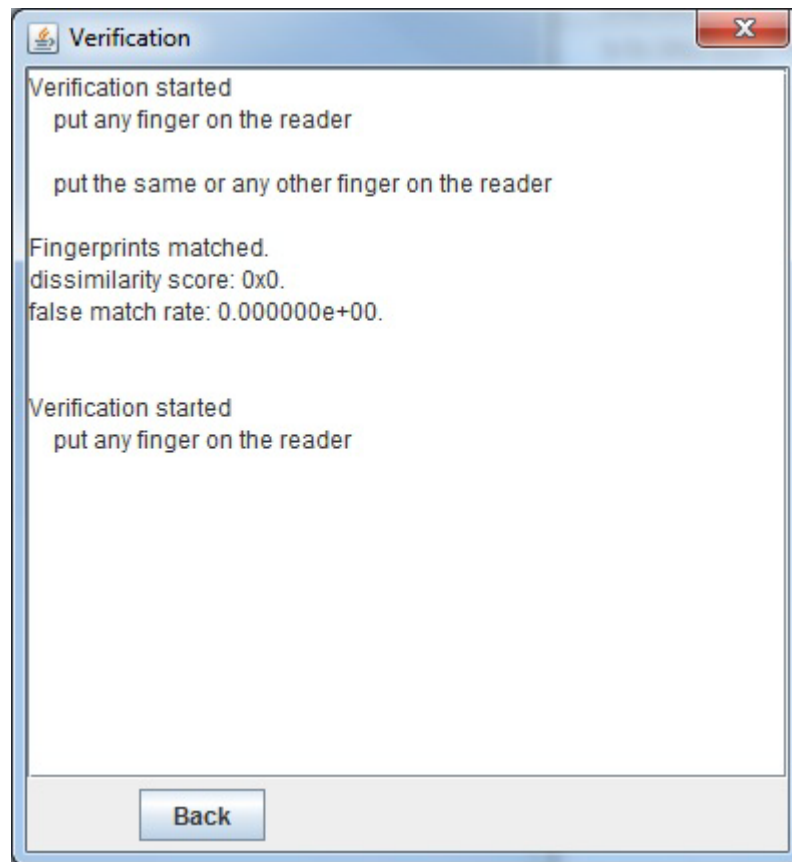
To exit identification mode, click on the **Back** button.

Using the Capture and Streaming Feature

The sample application also demonstrates the streaming feature (on fingerprint readers that support that feature). To test capturing or streaming, from the main window, click on the **Run capture** or **Run streaming** button.

This places the reader into capture/streaming mode and immediately the results are displayed in the window. For streaming mode, the window then becomes like a live window on the reader as it streams results. Placing a finger on the reader displays the streamed fingerprint, as shown below.
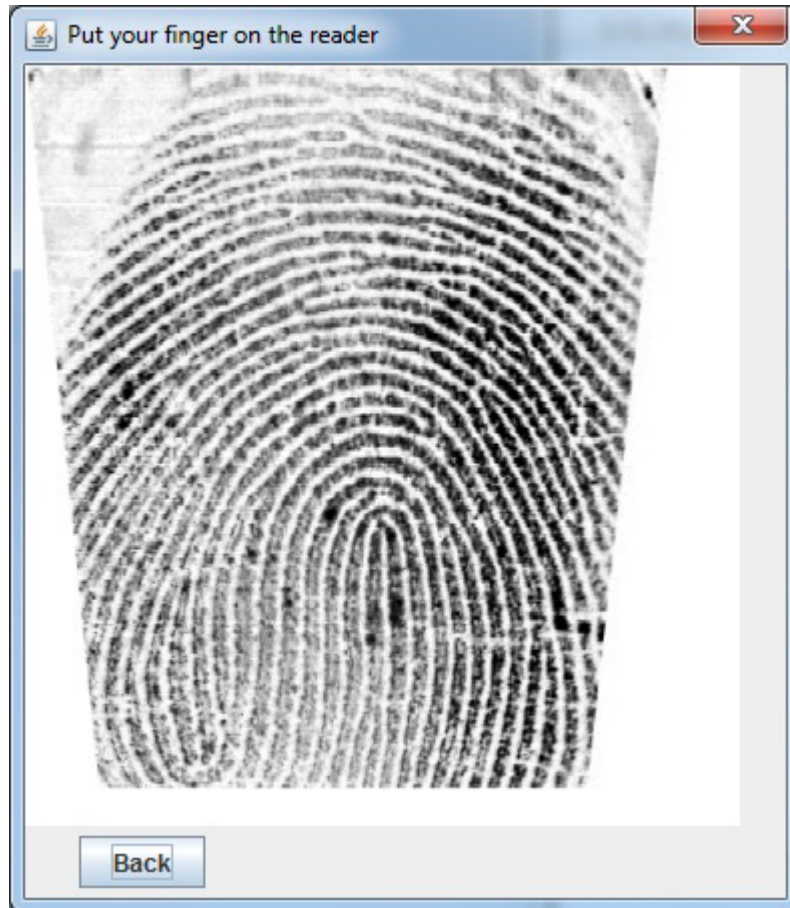
For streaming, removing the finger shows a blank stream.

To exit capture / streaming mode, click on **Back**.

Developing Applications with JavaScript

THIS CHAPTER PROVIDES INFORMATION NECESSARY FOR DEVELOPING WEB APPLICATIONS USING JAVASCRIPT AND THE U.ARE.U SDK.

Pre-Requisites

This chapter assumes that you have a working knowledge of JavaScript and that you know how to develop for modern internet browsers.

System Requirements

Development System

- Microsoft Windows XP Professional or higher, 32-bit or 64-bit
- The U.are.U. SDK for Windows, version 3.0 or above
- An ECMAScript 6 compatible shim such as the open source es6-shim (https://github.com/paulmillr/es6-shim)
- One of the following web browsers
  - Google Chrome
  - Mozilla Firefox - If installed after the SDK, reboot before running the JavaScript sample application or any applications you develop using JavaScript and this SDK
  - Internet Explorer or Microsoft Edge - The included JavaScript sample application and any other applications you develop using JavaScript and this SDK must be hosted on either an Apache TomCat web server or Microsoft Web Server (IIS) in order to work correctly with the Internet Explorer or Microsoft Edge browsers.
- Your developed application must allow calls to the U.are.U SDK by allowing *https://localhost:* wss://localhost:** in the connect-src policy directive of its Content Security Policy (CSP).

For specific supported browser versions, see the readme.txt file included with this SDK.

Target System

The included JavaScript sample application, and any web application created using JavaScript with the U.are.U. SDK for Windows, should run successfully on any system meeting the same minimum requirements stated above for the development system. Note that a USB port is also required in order to connect a fingerprint reader for capturing fingerprints.

Recommended minimum hardware for Windows machines is as follows.

- Intel x86 architecture with CPU from 600MHz and at least 96MB of available RAM
- Intel x64 (x86-64) architecture with CPU from 600MHz and at least 96MB of available RAM

Approximate file sizes are:

Function	x86	x64
Capture runtime (drivers + SDK layer) with fingerprint recognition	18 MB	17 MB

The SDK works on a variety of hardware and is intended to have a small footprint so that it can run even on minimal hardware. Less capable hardware may work, but response time may be less than optimal.

Installation

Install the U.are.U SDK as described in the chapter *Installing on the Development and Target Systems* on page 5. The DigitalPersona Authentication Service is part of the Typical installation, and is required for JavaScript development.

1f using Internet Explorer or Microsoft Edge for development or testing, you will need access to either an Apache TomCat web server or a Microsoft Web Server (IIS).

Folder structure

The JavaScript sample application folder structure for development is as follows, located under the U.are.U SDK\ Windows\Samples\UareUSampleWEB folder.

Name	Date modified	Type	Size
css	3/31/2016 12:29 AM	File folder	
images	3/31/2016 12:34 AM	File folder	
lib	3/31/2016 12:29 AM	File folder	
scripts	3/31/2016 12:29 AM	File folder	
app.css	3/30/2016 11:20 PM	Cascading Style S...	2 KB
app.js	3/31/2016 10:16 AM	JScript Script File	10 KB
index.html	3/30/2016 11:20 PM	Chrome HTML Do...	6 KB

The sample application can also be run from the index.html file located in the U.are.U SDK\Windows\Samples\Bin\ WEB folder.

Starting the Javascript sample application

Chrome or Firefox - To start the sample application, simply double-click the provided *index.html* file, or drag and drop the file onto the browser icon.

Internet Explorer or *Microsoft Edge* - To start the sample application, upload the `Samples/UareUJavaScript` folder to an Apache TomCat web server or Microsoft Web Server (IIS).

Uploading to a Tomcat server

1 Copy the `content of the` U.are.U SDK\Windows\Samples\UareUSampleWEB folder to the Tomcat `webapps` folder.

2 Restart the Tomcat server.

3 Open Internet Explorer or Microsoft Edge and enter the following location in the address bar.

   `http://localhost:8080/sampleapplication/`

4 Note that 8080 is the default port for Tomcat, but your installation may vary.

Uploading to a Windows Web Server (IIS)

1 Open IIS Manager.

2 In the Connections pane, expand the Sites node.

3 Right-click the *Default Web Site* or other site where you want to create the application.

4 Select *Add Application*.

5 Provide any necessary configuration information and the *Physical Path* to the U.are.U SDK\Windows\Samples\ UareUSampleWEB folder.

6 Restart the Web Server (IIS).

The JavaScript Sample Application

The U.are.U SDK includes a sample application to demonstrate the features of the SDK when using the JavaScript API. The sample application is located in the `Samples` folder.

The application demonstrates the following features of the SDK.

- Enumerate fingerprint readers
- Select a fingerprint reader to be used with fingerprint capture
- Get the characteristics of a fingerprint reader
- Start fingerprint capture using a selected fingerprint reader
- Stop fingerprint capture
- Receive captured fingerprints in the following formats: PNG image, WSQ, Intermediate and Raw.
- Receive activity notifications from the fingerprint reader
- Receive an indication of the quality of the fingerprint capture
- Monitor device connection and disconnection

Selecting a reader

When you launch the application, the appearance of the main screen will vary depending on whether a single fingerprint reader is connected to the computer, or whether multiple fingerprint readers are connected.

With no fingerprint reader connected
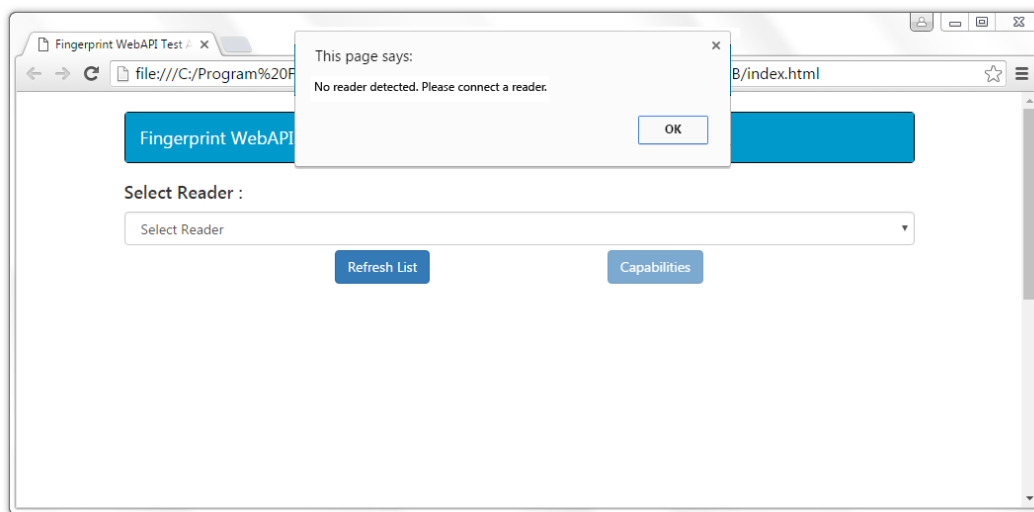
With a single fingerprint reader connected - The reader is automatically selected and the *Capture* tab is displayed.
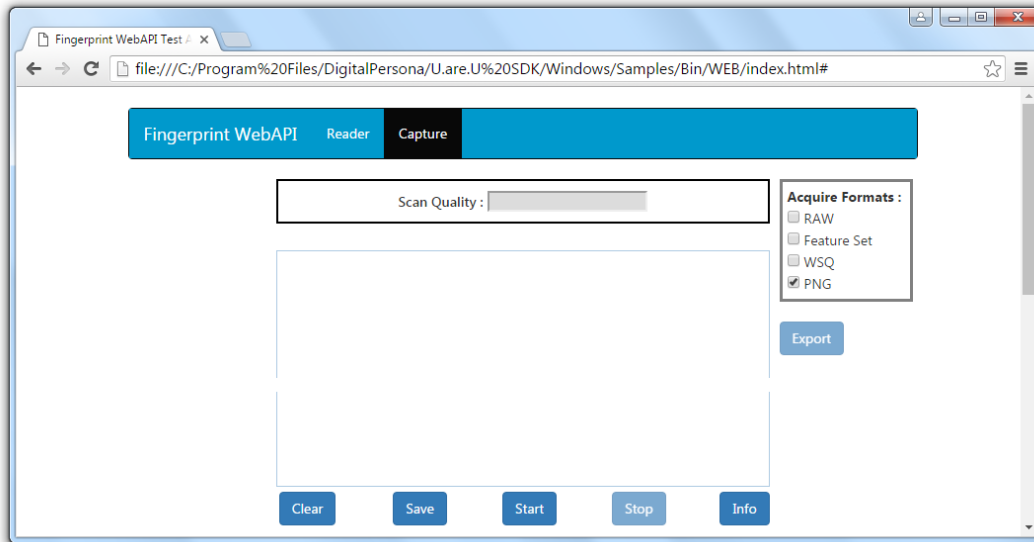
With multiple fingerprint readers connected - Select a fingerprint reader from the dropdown menu.
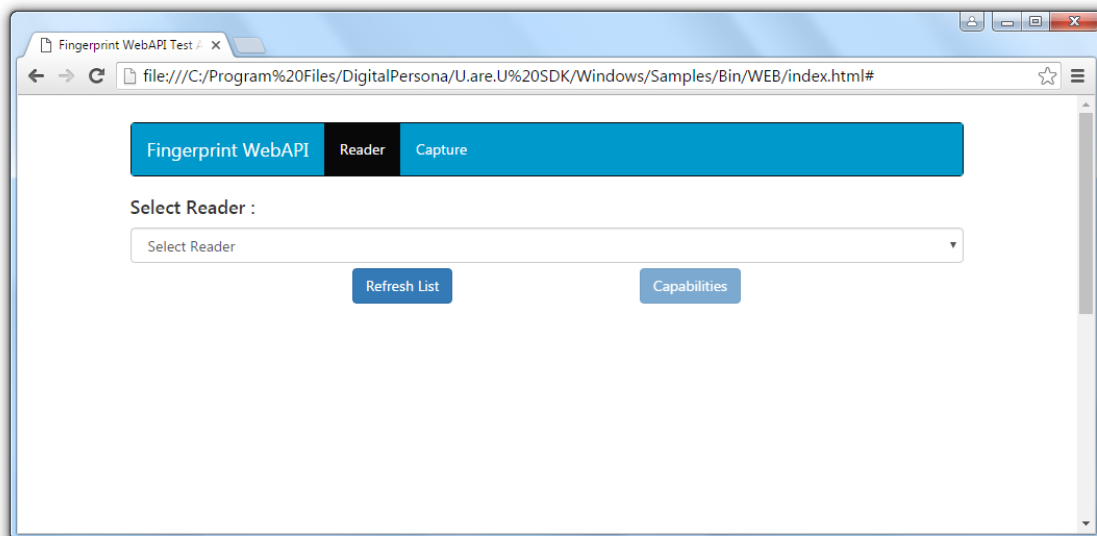
To see the reader capabilities, click the *Capabilities* button. The capabilities of the reader will be displayed, as shown in the image below. Click the *Close* button to continue.
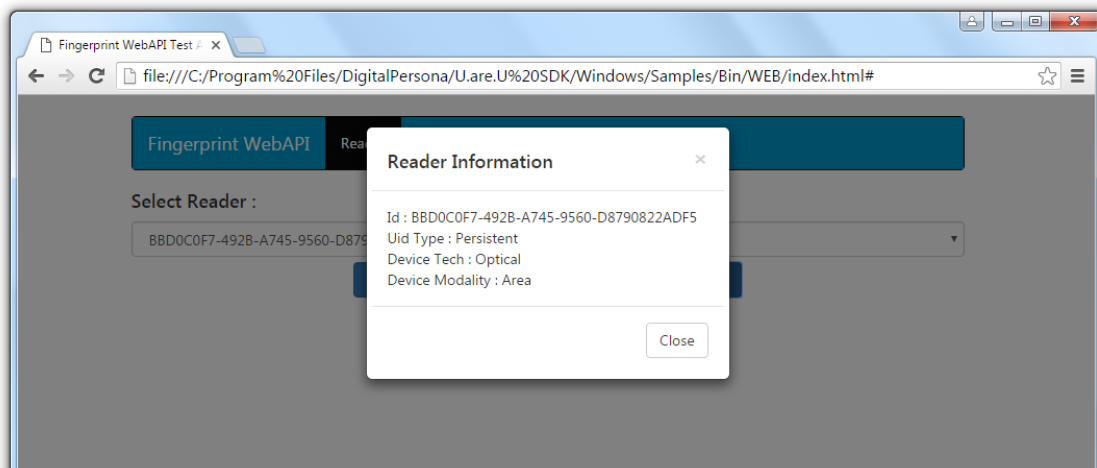
Capturing a fingerprint

On the *Capture* tab, click *Start* to begin fingerprint capture. An image of the captured fingerprint will be displayed in the middle of the web page. The *Start* button will be dimmed and the *Stop* button will become active.
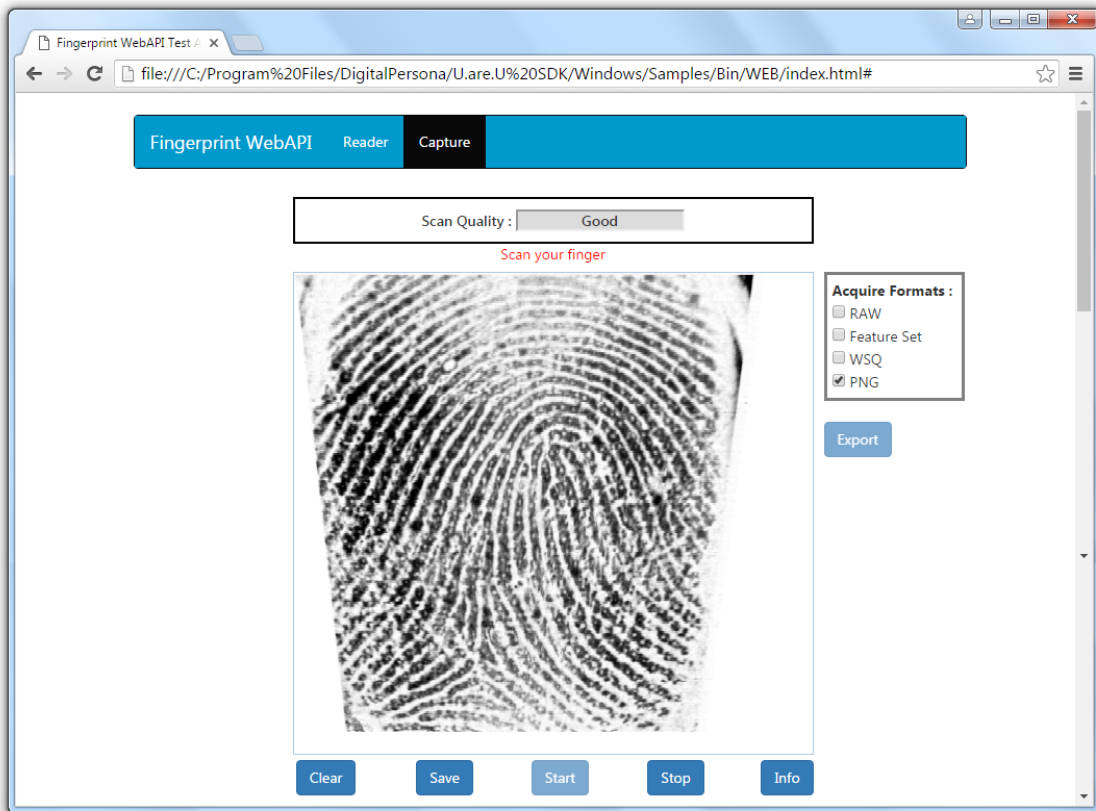
Place a finger on the reader to capture a fingerprint. Fingerprints can be captured until the *Stop* button is clicked. Indications of the Scan Quality will be displayed in the field below the Fingerprint WebAPI heading. Additional instructions or error messages will be displayed just below the Scan Quality area.

Saving a fingerprint thumbprint

After the first fingerprint is successfully captured, you can save a thumbnail of the fingerprint by clicking the *Save* button. Saved thumbnails are stored in the browser cache and displayed as smaller images below the row of buttons. This area will display up to five thumbnails at a time, and will then clear the area when the sixth image is saved.

Exporting fingerprint data

After a fingerprint is successfully captured, you can export the fingerprint data to your browser's default download folder, which in most cases will be the standard *Downloads* folder.

You can save the fingerprint data in one of the following file formats by selecting the format and clicking the *Export button*. Note that the *Export* button is not active until a fingerprint has been captured and is being displayed in the main fingerprint display area.

- **RAW** - Data format is a raw (unprocessed) biometric sample, also referred to as a Fingerprint Image in most biometric documentation.
- **Feature Set** - Data format is a partially processed (aka *Intermediate*) biometric sample, also referred to as a Feature Set in most biometric documentation.
- **WSQ** - Data format is a fully processed and compressed biometric sample, also referred to as a Fingerprint Template in most biometric documentation. To view this format, open the exported .wsq file in a WSQ viewer such as the free one available from Cognaxon.com.
- **PNG** - Data format is a .PNG image file. This is the default format if no other format has been selected.

Clearing a fingerprint image

To remove the fingerprint image from the image area, click the *Clear* button.

Displaying fingerprint reader information

To display information about the selected fingerprint reader, click the *Info* button. Any connected fingerprint readers will be listed below the thumbprint image area.
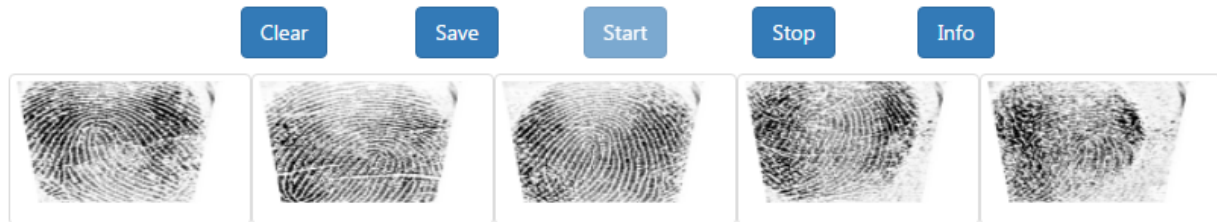
Click the down-arrow to reveal the reader information for a specific reader.

Developing Applications with JavaPOS

THIS CHAPTER PROVIDES INFORMATION NECESSARY FOR DEVELOPING WINDOWS APPLICATIONS IN JAVAPOS WITH THE U.ARE.U SDK.

Pre-Requisites

This chapter assumes that you have a working knowledge of JavaPOS and that you know how to develop for Microsoft Windows machines.

System Requirements

Development System

- Microsoft Windows XP Professional or higher, 32-bit or 64-bit
- Microsoft Visual Studio 2008 or later
- Java SE 7 (JDK 7) or newer

Target Runtime Hardware (Windows machine)

The Windows-based machine that will run the application must be one of the following hardware platforms:

- Intel x86 architecture with CPU from 600MHz and at least 96MB of available RAM
- Intel x64 (x86-64) architecture with CPU from 600MHz and at least 96MB of available RAM

Function	x86	x64
Capture runtime (drivers + SDK layer) with fingerprint recognition (wrapper only -- not including the base Java and C/C++ APIs)	17 MB	16 MB

Extra Installation Steps

After installing as described in Installing on the Development and Target Systems on page 5, you must do the following additional steps on target machines:

1   In your config path, find the `jpos.properties` folder and update the file's last line to contain the location of your `JPOSUareU.xml` file.

2   Copy the files in these two folders: `U.are.U SDK\Windows\Lib\Java` and `U.are.U SDK\Windows\Lib\ <x86 or x64>` to the location of your choice.

3   Make sure that `dpuareu.jar` is in the classpath and `dpuareu_jni.dll` is accessible by JVM. For example:

```
java.exe -classpath ".;C:\Program Files\DigitalPersona\U.areU SDK\Windows\Lib\Java\
dpuareu.jar" -Djava.library.path="C:\Program Files\DigitalPersona\U.areU SDK\Windows\
Lib\win32" UareUSampleJava
```

Registering your Device after Installation

To enable U.are.U support in your JavaPOS environment, you may need to register the DigitalPersona U.are.U Device Service.

To register the Device Service

1   Modify the `JAVA_POS_CONFIG_PATH` variable in the `register.bat` file in the `<Destination folder>\Windows\Lib\Java` folder. The variable should point to the JavaPOS `config` folder.

2   Run **register.bat**

To unregister the Device Service

- Run **register.bat -u**.

Upgrading from Previous Versions of the JavaPOS API

To upgrade your existing applications, be sure to do the following steps:

1   Add a reference to `<install directory>/U. are. U SDK/Windows/Lib/Java/dpuareu.jar` in your classpath. This change is often done in a build or run script.

2   Replace the old `dpjavapos.jar` with the newest one, located in `<install directory>/U. are. U SDK/Windows/Lib/Java`.

The JavaPOS Sample Application

This section describes the functionality of the sample application, which is located in the `<Install Directory>\U.are.U SDK\Windows\Samples` directory. For more information about the sample application and the sample code, particularly button functionality, refer to the `readme.txt` file located in the same directory.

IMPORTANT: To run the sample application, Java runtime environment® (JRE) 1.5 or higher must be installed on your machine.

To start the application

1   Open the `<Install Directory>\U.are.U SDK\Windows\Samples\Bin\JavaPOS` folder.

2   Run `run.bat`

The **sample application** window appears as shown below.
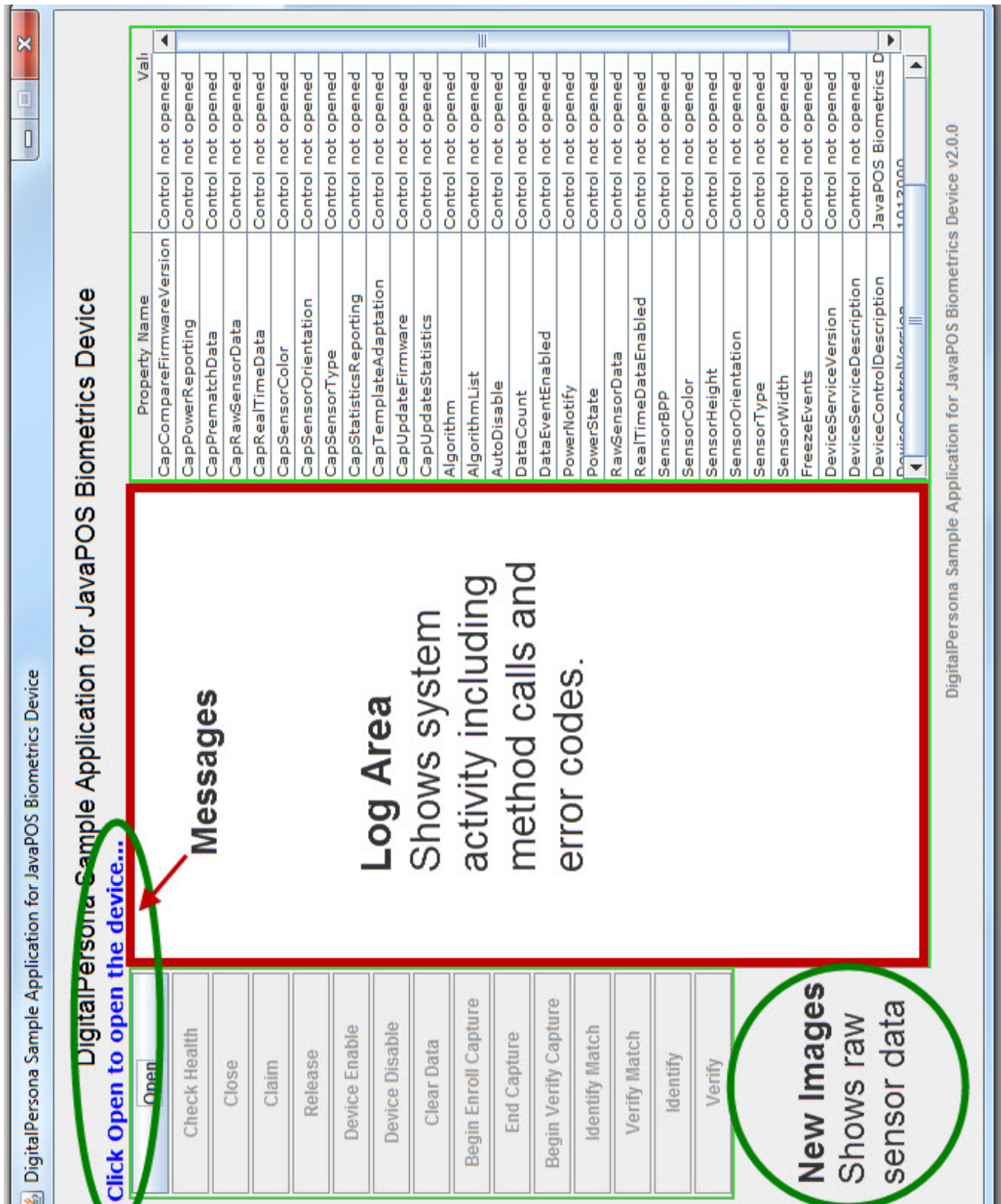
The DigitalPersona U.are.U SDK window is made up of the following distinct areas:

- Buttons area

  This area is located at the left of the window and contains buttons that initiate calls to various methods for interacting with the fingerprint reader and for performing fingerprint enrollment, verification, and identification operations.

- Messages area

  This area is located above the Buttons area and displays messages that inform the developer of system activity, invite them to perform actions such as touching the fingerprint reader, or advise them of system errors. The message that appears when you start the application is "Click to open the device...".

- New Image area

  This area is located at the bottom left and displays raw sensor data when a StatusUpdate event is returned signaling raw data is available.

- Log area

  This area is located in the middle of the window and displays a log of system activity, including method calls and error codes.

- Properties area

  This area is located at the right of the window and displays a list of properties, both common and specific (in the **Property name** column), and their current values (in the **Value** column).

To open the connection with the fingerprint reader

- Click **Open**.

  The **open** method of the Device Control object is called.

  If the call succeeds, the connection with the fingerprint reader is opened and various properties (common and specific) are set to their default values. These properties and values are displayed in the Properties area, and "Device opened..." appears in the Messages area, as shown in the screen shot below.

  NOTE:As each method is called, any properties that change are displayed in the Properties area.

  If the method call fails, a failure message appears in the Messages area and error codes are displayed in the Log area.
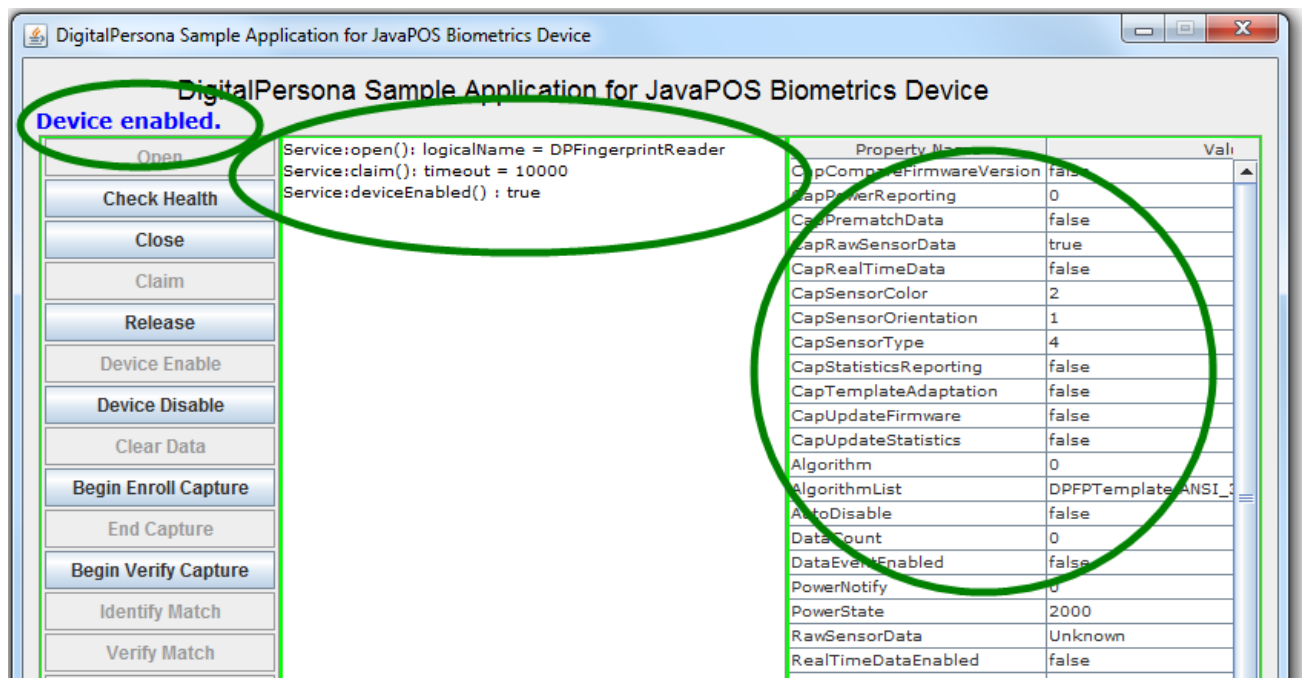
Once the connection with the fingerprint reader has been opened, it must be claimed.

To claim the fingerprint reader

- Click **Claim**.

  The **claim** method of the Device Control is called, the **claimed** property is set to true, and "Exclusive accessed" appears in the Messages area.

Once the connection with the fingerprint reader has been claimed, it must be enabled.

To enable the fingerprint reader

- Click **Device Enable**.

  The **deviceEnabled** property is set to true and "Physical Device Operational" appears in the Messages area.

  Enrolling a fingerprint consists of capturing four fingerprint images, converting them into fingerprint pre-enrollment templates, and then creating an enrollment template from these templates.

To perform fingerprint enrollment

1 Click **Begin Enroll Capture**.

   The **beginEnrollCapture** method of the Device Control is called and "Touch the sensor four times" appears in the Messages area.

2 Touch the fingerprint reader four times. Follow the instructions that appear in the Messages area to guide you.

   If the method call succeeds, an enrollment template is created and "Total enrollment completed: N" appears in the Messages area, where N is the number of total enrollments.

   If the method call fails, a failure message appears in the box in the Messages area. If an error occurs, appropriate messages appear in the Messages area, and error codes are displayed in the Log area.

To perform fingerprint verification

1 Click **Begin Verify Capture**.

   The **beginVerifyCapture** method of the Device Control is called and "Touch the sensor to capture sample data" appears in the Messages area.

   If the method call fails, a failure message appears in the Messages area. If an error occurs, appropriate messages appear in the Messages area, and error codes are displayed in the Log area.

2 Touch the fingerprint reader.

   If the method call succeeds, a verification template is created and "Sample Data Captured" appears in the Messages area.

   If the method call fails, a failure message appears in the Messages area. If an error occurs, appropriate messages appear in the Messages area, and error codes are displayed in the Log area.

3 Click **Verify Match**.

   The **verifyMatch** method of the Device Control is called.

   If the method call succeeds, a match is performed using the latest enrollment template available and the verification template that was created in step 2, and "Verification success!" or "Verification failed!" appears in the Messages area.

   If the method call fails, a failure message appears in the Messages area. If an error occurs, appropriate messages appear the Messages area, and error codes are displayed in the Log area.

To perform fingerprint identification

1 Click **Begin Verify Capture**.

   If the method call fails, a failure message appears in the Messages area. If an error occurs, appropriate messages appear in the Messages area, and error codes are displayed in the Log area.

2 Touch the fingerprint reader.

   If the method call succeeds, a verification template is created and "Sample Data Captured" appears in the Messages area.

If the method call fails, a failure message appears in the Messages area. If an error occurs, appropriate messages appear in the Messages area, and error codes are displayed in the Log area.

3   Click **Identify Match**.

The **identifyMatch** method of the Device Control is called.

If the method call succeeds, a match is performed using all of the enrollment templates available and the verification template that was created in step 2. A candidate ranking is generated by listing only the indices of the enrollment templates that match, and "Identification success!" or "Identification Failed!" appears in the Messages area.

If the method call fails, a failure message appears in the Messages area. If an error occurs, appropriate messages appear in the Messages area, and error codes are displayed in the Log area.

To perform fingerprint verification using a verification template created on-the-fly

1   Click **Verify**.

The **verify** method of the Device Control is called, and "Please touch the sensor for verification" appears in the Messages area.

2   Touch the fingerprint reader.

If the method call succeeds, a verification template is created on-the-fly. Then a match is performed using the latest enrollment template available and the verification template, and "Verification success!" or "Verification failed!" appears in the Messages area.

If the method call fails, a failure message appears in the Messages area. If an error occurs, appropriate messages appear in the Messages area, and error codes are displayed in the Log area.

To perform fingerprint identification using a verification template created on-the-fly

1   Click **Identify**.

The **identify** method of the Device Control is called and "Please touch the sensor for Identification" appears in the Messages area.

2   Touch the fingerprint reader.

If the method call succeeds, a verification template is created on-the-fly. Then a match is performed using all of the enrollment templates available and the verification template. A candidate ranking is generated by listing only the indices of the enrollment templates that match, and "Identification success!" or "Identification Failed!" appears in the Messages area.

If the method call fails, a failure message appears in the Messages area. If an error occurs, appropriate messages appear in the Messages area, and error codes are displayed in the Log area.

To clear the enrollment template array set and the verification template

• Click **Clear Data**.

The **clearInplace** method of the Device Control is called and "Clear data to start enrolling again" appears in the Messages area.

If the method call succeeds, the enrollment template array set and the verification template are cleared. A new verification template and a set of enrollment templates can now be created.

If the method call fails, a failure message appears in the Messages area, and error codes are displayed in the Log area.

Here is a demonstration of a sample sequence, showing the log area and messages.

In the screenshot below, the sample application window shows the following sequence of actions:

•   Open

- Claim,
- Device enable,
- Begin enroll capture,
- Captured 4 fingers,
- Begin verify capture,
- Captured 1 finger,
- Verify match,
- Returns a success notification (final message at top left)

To close the connection with the fingerprint reader

- Click **Close**.

    The **Close** method of the Device Control is called.

    If the method call succeeds, the connection with the fingerprint reader is closed, all of the controls other than the **Open** button are disabled, and the properties are reset, or cleared.
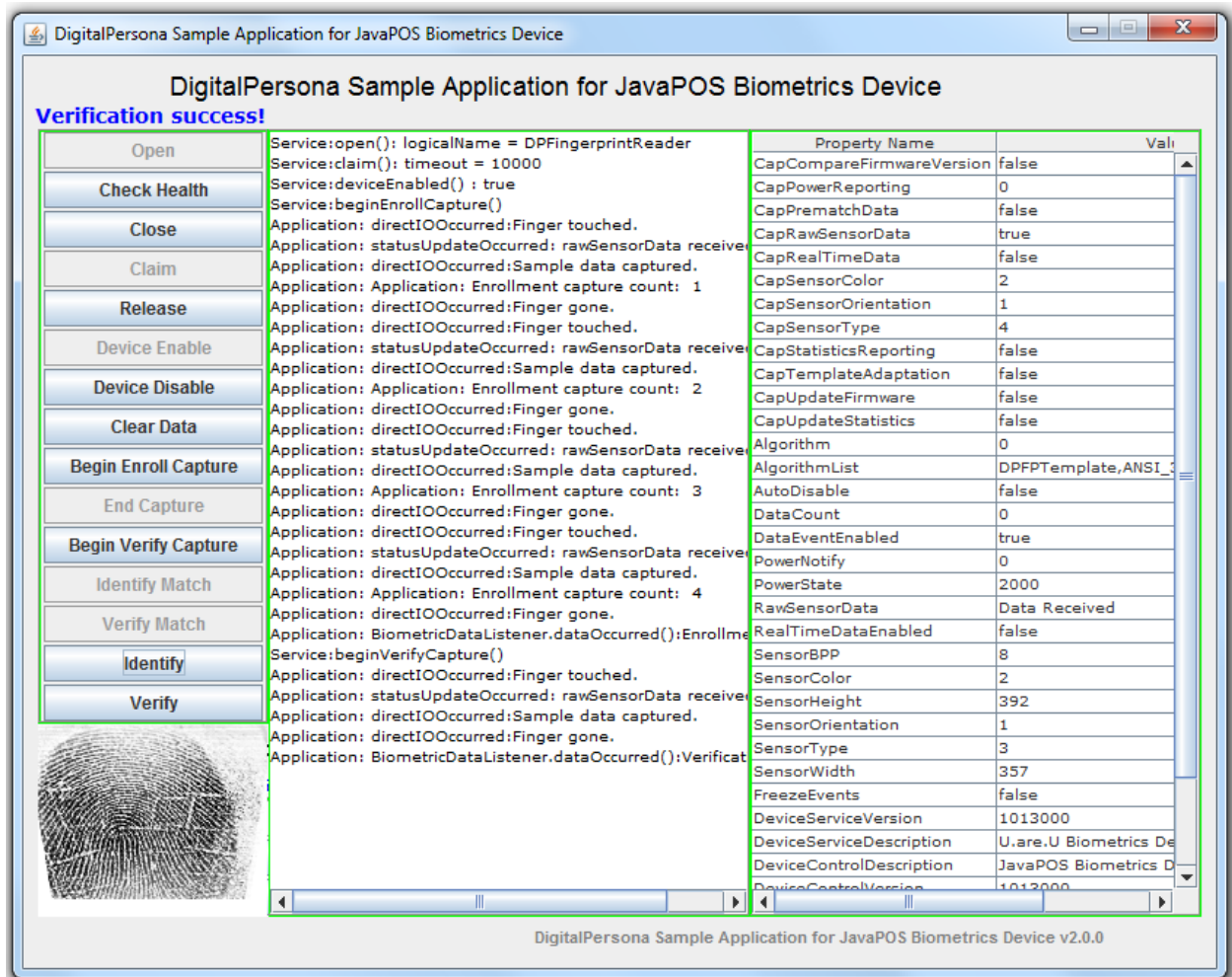
    If the method call fails, a failure message appears in the Messages area, and error codes are displayed in the Log area.

To close the application

Click the **Close** button at the top right of the window.

Developing Applications with OPOS

THIS CHAPTER PROVIDES INFORMATION NECESSARY FOR DEVELOPING WINDOWS APPLICATIONS IN OPOS WITH THE U.ARE.U SDK.

Pre-Requisites

This chapter assumes that you have a working knowledge of OPOS and that you know how to develop for Microsoft Windows machines.

System Requirements

Development System

- Microsoft Windows XP Professional or higher, 32-bit or 64-bit
- Microsoft Visual Studio 2008 or 2010 OR Visual Basic 6

Target Runtime Hardware (Windows machine)

The Windows-based machine that will run the application must be one of the following hardware platforms:

- Intel x86 architecture with CPU from 600MHz and at least 16MB of available RAM
- Intel x64 (x86-64) architecture with CPU from 600MHz and at least 16MB of available RAM

Function	x86	x64
Capture runtime (drivers + SDK layer) with fingerprint recognition (wrapper only -- not including the base C/C++ API)	18 MB	17 MB

Upgrading from Previous Versions of the OPOS API

To upgrade your existing applications, be sure to do the following steps:

1 Replace your previous `dpServiceObject.dll` and `OPOSBiometrics.ocs` with the new versions supplied in the product directory.

2 Run `install.bat` in the same directory (default is `Windows/Lib/Win32`).

Using the Sample Application

This section describes the functionality of the sample application, which is located in the `<Install Directory>\ U.are.U SDK\Windows\Samples` directory. For more information about the sample application and the sample code, particularly button functionality, refer to the `readme.txt` file located in the same directory.

To start the application

- Open the DPOPOSDemo.exe file.

   The **DigitalPersona U.are.U UPOS for OPOS** window appears as shown in the screen shot below.
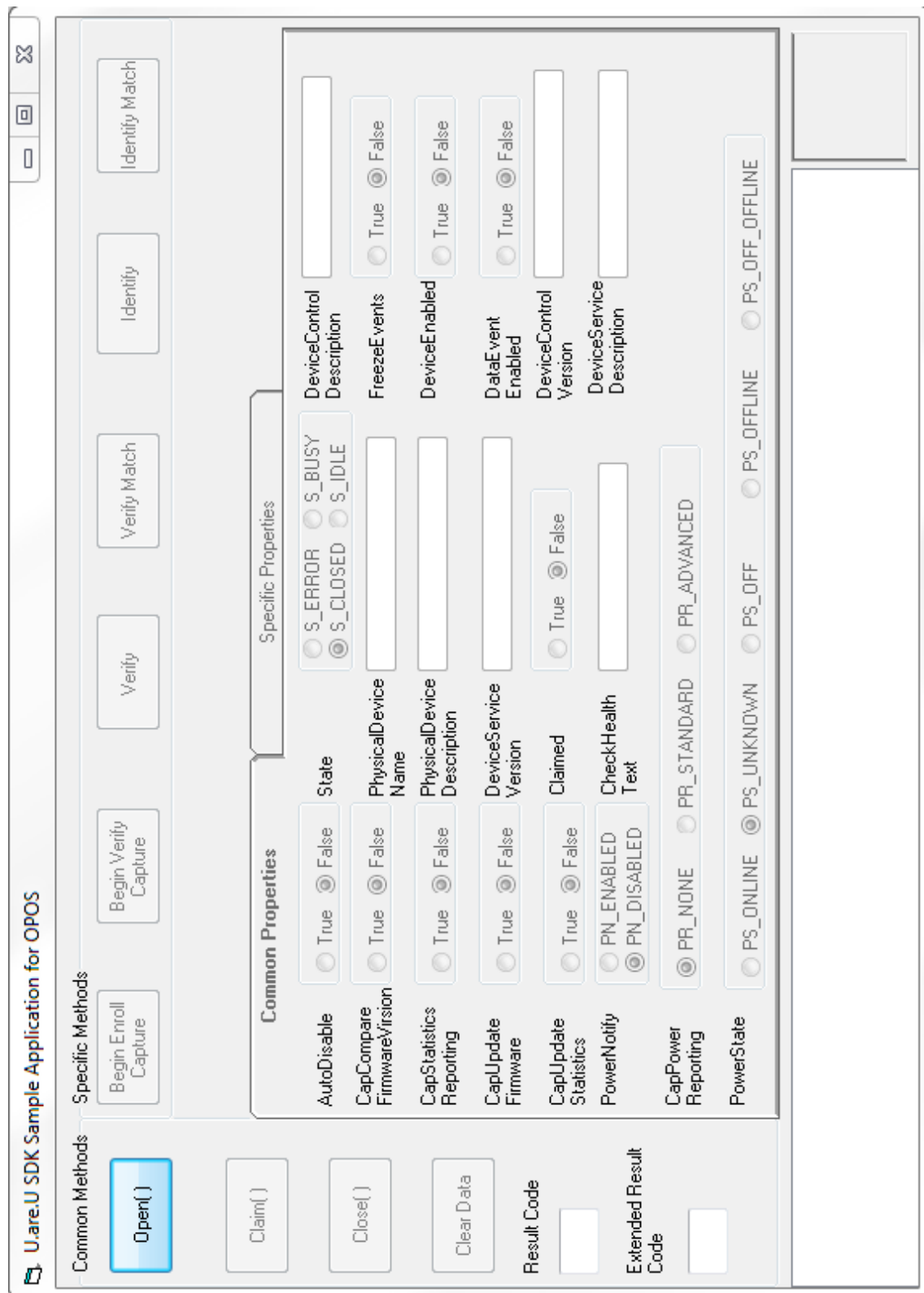
To open the connection with the fingerprint reader

- Click **Open()**.

  The **Open** method of the Control Object (CO) is called.

  If the call succeeds, the connection with the fingerprint reader is opened and various properties (common and specific) are set to their default values, which are displayed in the **Common Properties** and **Specific Properties** tabs. Also, "Device Opened" appears in the area under the **Specific Methods** control box.

NOTE:As each method is called, any properties that change are displayed in the **Common Properties** and **Specific Properties** tabs.

If the method call fails, a failure message appears in the box at the bottom of the window, and error codes are displayed in the **Result Code** and **Extended Result Code** boxes.

Once the connection with the fingerprint reader has been opened, it must be claimed.

To claim the fingerprint reader

- Click **Claim()**.

The **Claim** method of the CO is called, and the **Claimed** property is set to true. Then the **DeviceEnabled** and **DataEventEnabled** properties are set to true, and "Device Claimed" appears in the area under the **Specific Methods** control box.

If the method call fails, a failure message appears in the box at the bottom of the window, and error codes are displayed in the **Result Code** and **Extended Result Code** boxes.

Once you have opened and claimed the device, the application will look like the screenshot below.

Enrolling a fingerprint consists of capturing four fingerprint images, converting them into fingerprint pre-enrollment templates, and then creating an enrollment template from these templates.

To perform fingerprint enrollment

1 Click **Begin Enroll Capture**.

The **beginEnrollCapture** method of the CO is called, and "Waiting for fingerprint scan" appears in the area under the **Specific Methods** control box.

2 Touch the fingerprint reader four times. Follow the instructions that appear in the area under the **Specific Methods** control box to guide you.

If the method call succeeds, an enrollment template is created and "Fingerprint Image Scanned" appears in the area under the **Specific Methods** control box.

If the method call fails, a failure message appears in the area under the **Specific Methods** control box. If an error occurs, appropriate messages appear in the box at the bottom of the window, and error codes are displayed in the **Result Code** and **Extended Result Code** boxes.

To perform fingerprint verification

1 Click **Begin Verify Capture**.

The **beginVerifyCapture** method of the CO is called, and "Waiting for fingerprint scan" appears in the area under the **Specific Methods** control box.

2 Touch the fingerprint reader.

If the method call succeeds, a verification template is created and "Fingerprint Image Scanned" appears in the area under the **Specific Methods** control box.

If the method call fails, a failure message appears in the area under the **Specific Methods** control box. If an error occurs, appropriate messages appear in the box at the bottom of the window, and error codes are displayed in the **Result Code** and **Extended Result Code** boxes.

3 Click **Verify Match**.

The **verifyMatch** method of the CO is called.

If the method call succeeds, a match is performed using the latest enrollment template available and the verification template that was created in step 2. The result appears in the area under the **Specific Methods** control box: "Fingerprint matches" or "Fingerprint does not match."

If the method call fails, a failure message appears in the area under the **Specific Methods** control box. If an error occurs, appropriate messages appear in the box at the bottom of the window, and error codes are displayed in the **Result Code** and **Extended Result Code** boxes.

To perform fingerprint identification

1. Click **Begin Verify Capture**.

The **beginVerifyCapture** method of the CO is called, and "Waiting for fingerprint scan" appears in the area under the **Specific Methods** control box.

2. Touch the fingerprint reader.

If the method call succeeds, a verification template is created and the **Fingerprint Image Scanned** message appears in the area under the **Specific Methods** control box.

If the method call fails, a failure message appears in the area under the **Specific Methods** control box. If an error occurs, appropriate messages appear in the box at the bottom of the window, and error codes are displayed in the **Result Code** and **Extended Result Code** boxes.

3. Click **Identify Match**.

The **identifyMatch** method of the CO is called.

If the method call succeeds, a match is performed using all of the enrollment templates available and the verification template that was created in step 2. A candidate ranking is generated by listing only the indices of the enrollment templates that match. The result appears in the area under the **Specific Methods** control box, for example, "Candidate array: 0 2," or, if none of the templates matches, "Candidate ranking array is empty."

If the method call fails, a failure message appears in the area under the **Specific Methods** control box. If an error occurs, appropriate messages appear in the box at the bottom of the window, and error codes are displayed in the **Result Code** and **Extended Result Code** boxes.

On completion of the verify match and identification match, the screen will look similar to the image below.

To perform fingerprint verification using a verification template created on-the-fly

1   Click **Verify**.

The **verify** method of the CO is called, and "Waiting for fingerprint scan" appears in the area under the **Specific Methods** control box.

2   Touch the fingerprint reader.

If the method call succeeds, a verification template is created on-the-fly. Then a match is performed using the latest enrollment template available and the verification template. The result appears in the area under the **Specific Methods** control box: "Fingerprint matches" or "Fingerprint does not match."

If you do not place your finger on the fingerprint reader within the stipulated time (10 seconds in this sample), the operation times out and "Timeout error…" appears in the area under the **Specific Methods** control box.

If the method call fails, a failure message appears in the area under the **Specific Methods** control box. If an error occurs, appropriate messages appear in the box at the bottom of the window, and error codes are displayed in the **Result Code** and **Extended Result Code** boxes.
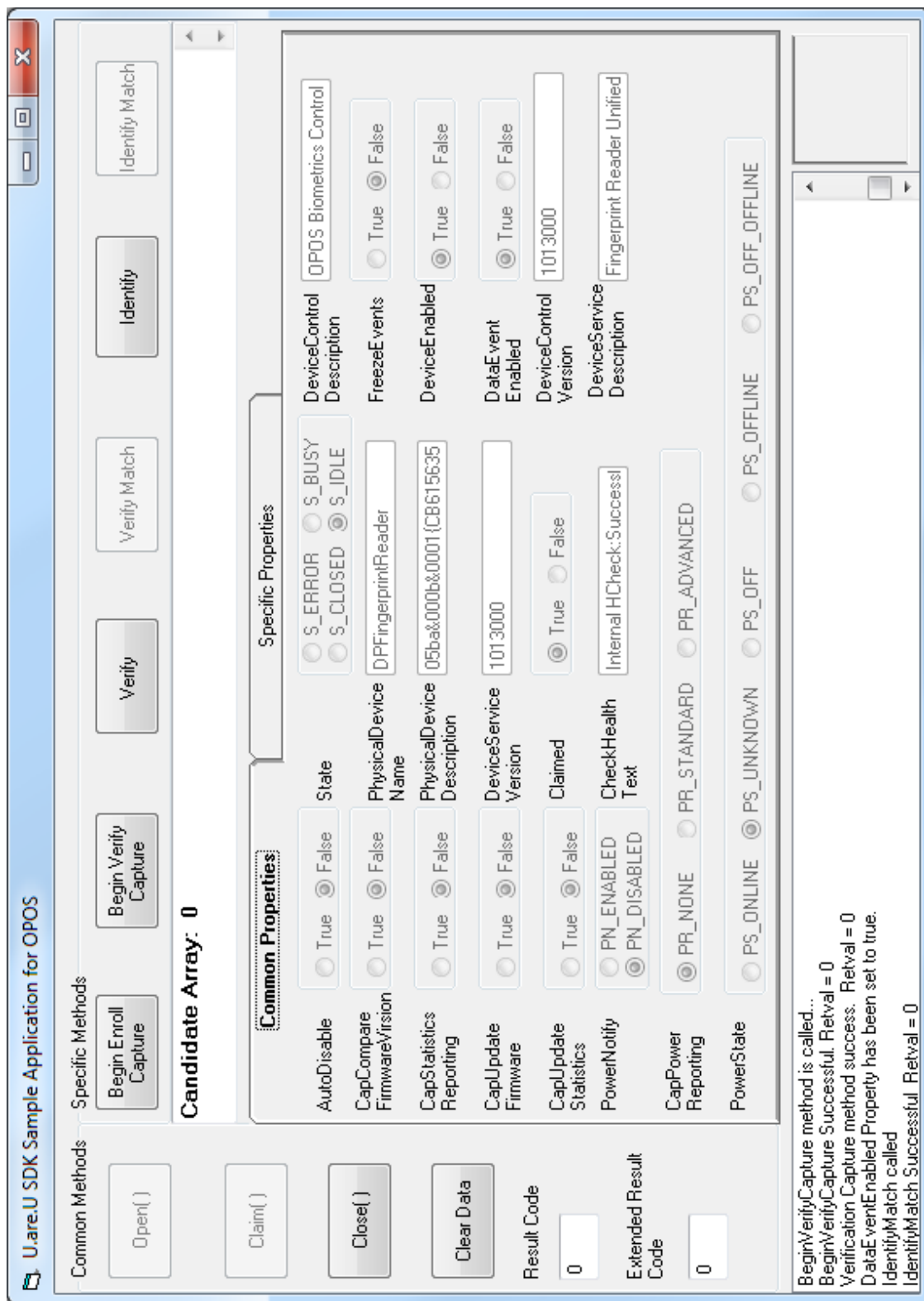
To perform fingerprint identification using a verification template created on-the-fly

1   Click **Identify**.

The **identify** method of the CO is called, and "Waiting for fingerprint scan" appears in the area under the **Specific Methods** control box.

2   Touch the fingerprint reader.

If the method call succeeds, a verification template is created on-the-fly. Then a match is performed using all of the enrollment templates available and the verification template. A candidate ranking is generated by listing only the indices of the enrollment templates that match. The result appears in the area under the **Specific Methods** control box, for example, "Candidate array: 0 2," or, if none of the templates matches, "Candidate ranking array is empty."

If you do not place your finger on the fingerprint reader within the stipulated time (10 seconds in this sample), the operation times out and "Timeout error…" message appears in the area under the **Specific Methods** control box.

If the method call fails, a failure message appears in the area under the **Specific Methods** control box. If an error occurs, appropriate messages appear in the box at the bottom of the window, and error codes are displayed in the **Result Code** and **Extended Result Code** boxes.

To close the connection with the fingerprint reader

• Click **Close()**.

The **Close** method of the CO is called.

If the method call succeeds, the connection with the fingerprint reader is closed, all of the controls other than the **Open()** button are disabled, and the properties are reset, or cleared.

If the method call fails, a failure message appears in the box at the bottom of the window, and error codes are displayed in the **Result Code** and **Extended Result Code** boxes.

To clear the enrollment template array set and the verification template

• Click **Clear Data**.

The **clearInplace** method of the CO is called.

If the method call succeeds, the enrollment template array set and the verification template are cleared. A new verification template and a set of enrollment templates can now be created.

If the method call fails, a failure message appears in the box at the bottom of the window, and error codes are displayed in the **Result Code** and **Extended Result Code** boxes.

To close the application

Click the **Close** button.

PAD: Presentation Attack Detection

This chapter describes how to use the Presentation Attack Detection (PAD) feature of the U.are.U SDK.

Pre-Requisites

When installing without the DigitalPersona Authentication Service, the U.are.U Windows SDK (version 3.2.0 and above) includes a Presentation Attack Detection (PAD) option against 3-dimensional (molded) spoofs which can be used with U.are.U 5300 readers and modules only.

By default, the PAD algorithm is turned off, but it can be turned on by calling dpfpd_set_parameter() with the parm_id equal to DPFPDD_PARMID_PAD_ENABLE with the first byte of the buffer set to 1.

Regardless of whether or not the 3-dimensional PAD algorithm is enabled, the DigitalPersona 2-dimensional (printed) spoof detection is always enable for U.are.U 5300 readers and modules.

Enable/disable functions

```
//enable PAD

unsigned char bEnable = 1;

dpfpdd_set_parameter(hReader, DPFPDD_PARMID_PAD_ENABLE, 1, &bEnable);


//disable PAD

unsigned char bEnable = 0;

dpfpdd_set_parameter(hReader, DPFPDD_PARMID_PAD_ENABLE, 1, &bEnable);


//check if PAD is enabled

unsigned char bEnabled = 0;

dpfpdd_get_parameter(hReader, DPFPDD_PARMID_PAD_ENABLE, 1, &bEnabled);

if(bEnabled) {

    //PAD enabled

}

else {

    //PAD disabled

}
```

False Accept/Reject functions

The default False Accept Rate is 3%. The target False Accept Rate for PAD can be changed by calling dpfpdd_set_parameter() with parm_id equal to DPFPDD_PAD_CONFIDENCE with the first byte of the buffer set to 1.

```
//set FAR at 5%

unsigned char bFAR = 5;
```

dpfpdd_set_parameter(hReader, DPFPDD_PAD_CONFIDENCE, 1, &bFAR);

//get current FAR

unsigned char bFAR = 0;

dpfpdd_get_parameter(hReader, DPFPDD_PAD_CONFIDENCE, 1, &bFAR);

When a spoofed finger is detected, the quality field of the DPFPDD_CAPTURE_RESULT structure returned from dpfpdd_capture() will have a value of DPFPDD_QUALITY_FAKE_FINGER.

FAR/FRR relationships

Setting the False Accept Rate determines the corresponding False Reject Rate, as shown in the table below.

False Accept Rate (FAR) %	False Reject Rate (FRR) %
49	0.087
47	0.087
46	0.092
43	0.114
41	0.119
39	0.126
37	0.143
34	0.153
32	0.165
31	0.180
28	0.211
27	0.233
25	0.243
24	0.265
22	0.311
21	0.325
18	0.403
17	0.454
16	0.478
15	0.544
14	0.609
13	0.638
12	0.704
11	0.804
10	0.961
9	1.073
8	1.258
7	1.430
6	1.629
5	1.937
4	2.304
3	2.957
2	4.208
1	8.049

Redistribution

THIS CHAPTER PROVIDES DETAILS ABOUT THE FILES INCLUDED IN THE U.ARE.U SDK THAT MUST BE DISTRIBUTED WITH YOUR APPLICATION.

Locating the Redistributable Installation Files

When you unzip the distribution file, the unzipped collection of files includes:

- `Redist` - folder containing materials for redistributing the product
- `RTE\Install` - folder containing installation files for installing applications on target hardware (user workstations or hardware devices)

When you develop a product based on the U.are.U SDK, you need to distribute U.are.U files to your end users. You may redistribute the files in the `Redist` folder to your end users pursuant to the terms of the end user license agreement (EULA), attendant to the software and located in the `Windows\Docs` folder in the installed product folder. These files are designed and licensed for use with your application.

You may integrate U.are.U files in three ways:

1. Add the supplied merge modules to your installer.
2. Have users run the U.are.U installer as a prerequisite to installing your application.
3. Call the U.are.U installer from your installer.

Per the terms of the EULA, Crossmatch grants you a non-transferable, non-exclusive, worldwide license to redistribute, either directly or via the respective merge modules, the files contained in the `RTE\Install` and `Redist` folders in the U.are.U SDK software package to your end users and to incorporate these files into derivative works for sale and distribution.

Merge Modules

The table below shows the merge modules in the `Redist` folder that are required for each platform.

Merge Module File	Description	C/C++		.NET		ActiveX		Java		JavaPOS		OPOS	
		x86	x64	x86	x64	x86	x64	x86	x64	x86	x64	x86	x64
CMFPDrivers7X	Device Components	x		x		x		x		x		x	
CMFPDrivers7X64			x		x		x		x		x		x
DPDrivers		x		x		x		x		x		x	
DPDrivers64			x		x		x		x		x		x
DPFPDrivers5X		x		x		x		x		x		x	
DPFPDrivers5X64			x		x		x		x		x		x
DPFPCapture		x		x		x		x		x		x	
DPFPCapture64			x		x		x		x		x		x
DPEngine	Fingerprint-Matching Engine	x		x		x		x		x			
DPEngine64			x		x		x		x		x		x
DPFingerJet		x		x		x		x		x		x	
DPFingerJet64			x		x		x		x		x		x

Merge Module File	Description	C/C++ x86	C/C++ x64	.NET x86	.NET x64	ActiveX x86	ActiveX x64	Java x86	Java x64	JavaPOS x86	JavaPOS x64	OPOS x86	OPOS x64
DPFPApi	Authentication Service Components	x		x		x		x		x			
DPFPApi64			x		x		x		x		x		
DPUareUDevices			x		x		x		x		x		
DPUareUDevices64			x		x		x		x		x		
DPUareUHostService			x		x		x		x		x		
DPUareUHostService64			x		x		x		x		x		
DPUareUPolicies			x		x		x		x		x		
DPUareUPolicies64			x		x		x		x		x		
DPUareUSecurityApps			x		x		x		x		x		
DPUareUSecurityApps64			x		x		x		x		x		
DPUareUAgent			x		x		x		x		x		
DPUareUJava	Java Components							x	x	x	x		
DPUareUJni								x		x			
DPUareUJni64									x		x		
DPJavaPOS	JavaPOS components									x	x		
DPUareUNET	.NET libraries			x	x								
DPUareUX	ActiveX libraries					x	x						
DPOpos	OPOS libraries											x	x

Redistributable documentation

You may redistribute the end-user documentation included in the `Redist` directory to your users, according to the terms of the product EULA. It is located in the following directory after installation.

`[Installation Directory]\DigitalPersona\U.are.U SDK\Windows\Docs`

The documentation in the `Redist` directory includes Regulatory and Warranty information as well as Usage and Care Guides for DigitalPersona and TouchChip readers that should be provided to anyone purchasing readers or modules from you.

CROSSMATCH