

DigitalPersona, Inc.

U.are.U SDK

Version 2

Platform Guide for Windows



DigitalPersona, Inc.

© 2012 DigitalPersona, Inc. All Rights Reserved.

All intellectual property rights in the DigitalPersona software, firmware, hardware and documentation included with or described in this guide are owned by DigitalPersona or its suppliers and are protected by United States copyright laws, other applicable copyright laws, and international treaty provisions. DigitalPersona and its suppliers retain all rights not expressly granted.

U.are.U® and DigitalPersona® are trademarks of DigitalPersona, Inc. registered in the United States and other countries. Windows, Windows Server 2003/2008, Windows Vista, Windows 7 and Windows XP are registered trademarks of Microsoft Corporation. All other trademarks are the property of their respective owners.

This DigitalPersona U.are.U SDK Platform Guide for Windows and the software it describes are furnished under license as set forth in the "License Agreement".

Except as permitted by such license, no part of this document may be reproduced, stored, transmitted and translated, in any form and by any means, without the prior written consent of DigitalPersona. The contents of this manual are furnished for informational use only and are subject to change without notice.

Any mention of third-party companies and products is for demonstration purposes only and constitutes neither an endorsement nor a recommendation. DigitalPersona assumes no responsibility with regard to the performance or use of these third-party products.

DigitalPersona makes every effort to ensure the accuracy of its documentation and assumes no responsibility or liability for any errors or inaccuracies that may appear in it.

Technical Support

To access DigitalPersona technical support resources, go to <http://www.digitalpersona.com/support>.

Phone support is available at (877) 378-2740 (US) or +1 650-474-4000 (outside the US).

Feedback

Although the information in this guide has been thoroughly reviewed and tested, we welcome your feedback on any errors, omissions, or suggestions for future improvements. Please contact us at

TechPubs@digitalpersona.com

or

DigitalPersona, Inc.
720 Bay Road, Suite 100
Redwood City, California 94063
USA
(650) 474-4000
(650) 298-8313 Fax

Table of Contents

1	Introduction	5
	Getting Updated Documentation	5
2	Installation	6
	Installing on the Development and Target Systems	6
	Step 1: Installing on the Development System (SDK Installation)	6
	Step 2: Installing on the Target Hardware (RTE Installation)	7
	Authentication Service	8
	Uninstalling	8
3	Developing Applications with C/C++	9
	Pre-Requisites	9
	System Requirements	9
	The C/C++ Sample Application	9
4	Developing Applications with .NET	14
	Pre-Requisites	14
	System Requirements	14
	Static libraries and DLLs	14
	The .NET Sample Application	15
	Selecting a Reader	16
	Capturing a Fingerprint	17
	Testing Streaming Mode	17
	Enrolling a Finger	18
	Identifying a Fingerprint	19
	Verifying a Fingerprint	19
	Testing the Enrollment UI Control	20
	Testing the Identification UI Control	22
5	Developing Applications with ActiveX / .NET	23
	Pre-Requisites	23
	Overview	23
	Static libraries and DLLs	23
	ActiveX Control Unique Identifiers	23
	The ActiveX Samples	24
6	Developing Applications with Java	25
	Pre-Requisites	25
	System Requirements	25

Extra Installation Steps	25
The Java Sample Application	26
Selecting a Reader	27
Enrolling a Finger	29
Identifying a Fingerprint	30
Verifying a Fingerprint	31
Using the Capture and Streaming Feature	32
7 Developing Applications with JavaPOS	33
Pre-Requisites	33
System Requirements	33
Extra Installation Steps	33
Registering your Device after Installation	34
Upgrading from Previous Versions of the JavaPOS API	34
The JavaPOS Sample Application	34
8 Developing Applications with OPOS	43
Pre-Requisites	43
System Requirements	43
Upgrading from Previous Versions of the OPOS API	43
Using the Sample Application	43
9 Redistribution	53
Locating the Redistributable Installation Files	53
Merge Modules	53
Fingerprint Reader Documentation	55
Hardware Warnings and Regulatory Information	55
Fingerprint Reader Use and Maintenance Guide	55

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.

Chapter 1, *Introduction* (this chapter) describes how to get the latest version of this documentation.

Chapter 2, *Installation* provides instructions for installing on your development system and on the target (Windows) reader.

Chapter 3, *Developing Applications with C/C++* lists system requirements for developing and running applications in C/C++ and describes the sample application.

Chapter 4, *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#.

Chapter 5, *Developing Applications with ActiveX / .NET* lists system requirements for developing and running applications using Active and other ActiveX notes.

Chapter 6, *Developing Applications with Java* lists system requirements for developing and running applications using Java, provides additional installation instructions and describes the Java sample application.

Chapter 7, *Developing Applications with JavaPOS* provides information on using the JavaPOS-compliant API built on the U.are.U framework.

Chapter 8, *Developing Applications with OPOS* provides information on using the JavaPOS-compliant API built on the U.are.U framework.

Chapter 9, *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 U.are.U SDK, you may want to check online at our website for an updated version of this document at

<http://www.digitalpersona.com/Support/Reference-Material/DigitalPersona-SDK-Reference-Material/>

Except as noted in the platform/language-specific chapters, the installation process is the same for development on all Windows-based fingerprint capture devices.

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.msi`
For 64-bit systems, run `SDK\x64\setup.msi`

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
Include	Header files for C/C++ API.
Windows\Docs	End user license agreement (EULA) plus documentation: <ul style="list-style-type: none"> ■ SDK Developer Guide - describes all APIs ■ Platform Guide for Windows - Windows-specific details ■ C_API - Doxygen for C/C++ API ■ Java_API - Javadoc for Java API ■ .NET_ActiveX_API - Doxygen for .NET and ActiveX APIs

Folder	Contents
Windows\Lib	Runtime files: <ul style="list-style-type: none"> ■ .NET - libraries and controls for .NET and ActiveX ■ x64 - libraries for 64-bit processes ■ Win32 - libraries, OPOS libraries ■ Java - Java and JavaPOS JAR files
Windows\Samples\	Compiled sample applications: <ul style="list-style-type: none"> ■ Bin <ul style="list-style-type: none"> ■ Java - Java sample ■ JavaPOS - JavaPOS sample ■ OPOS - OPOS sample ■ .NET - .NET sample ■ x64 - C/C++ sample for 64-bit processes ■ Win32 - C/C++ sample Source files for sample applications: <ul style="list-style-type: none"> ■ Include - WTL80 files for C/C++ sample ■ UareUSample - C/C++ sample ■ UareUSampleJava - Java sample ■ UareUSampleJavaPOS - JavaPOS sample ■ UareUSampleOPOS - OPOS sample ■ UareUSampleVBNET - .NET /VBNET sample ■ UareUSampleVBNET_CaptureOnly - .NET/VBNet sample that demonstrates only capture ■ UareUSampleCSharp - .NET /C# sample ■ UareUSampleCSharp_CaptureOnly - .NET/C# sample that demonstrates only capture ■ UareUSampleActiveX - ActiveX 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.msi`
For 64-bit systems, run `RTE\x64\setup.msi`

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: <ul style="list-style-type: none">■ .NET - libraries and controls for .NET and ActiveX■ x64 - libraries for 64-bit processes■ Win32 - libraries, OPOS libraries■ Java - Java and JavaPOS JAR files

Authentication Service

The installation process installs and registers a service named **Authentication Service** on both the target and development systems. The service can be managed in the regular way via the Services Control Applet in the Microsoft Management Console by running `services.msc` as Administrator. This service provides fingerprint capture. If your application only uses FingerJet Engine, than it's not necessary to run the service.

Uninstalling

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

Pre-Requisites

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

System Requirements

Development System

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

Target Runtime Hardware (Windows Reader)

The Windows-based reader 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:

	x86	x64
Capture runtime (drivers + SDK layer) - includes service	5.0 MB	5.5 MB
Fingerprint recognition runtime	160 KB	220 KB

In addition, the reader must also have:

- a USB port
- 16 Mb free memory

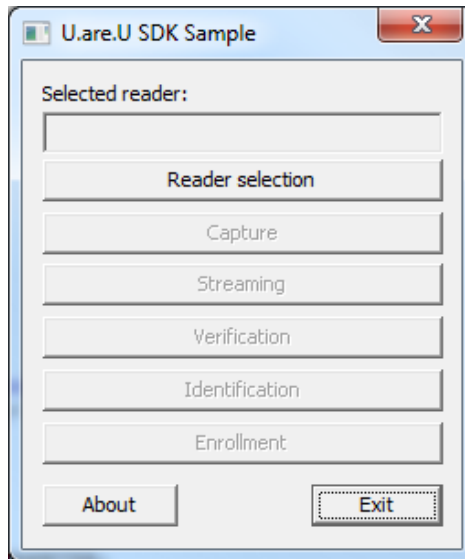
The SDK works on a variety of hardware and is intended to have a small footprint so that it can run even on minimal hardware. Less capable hardware will work, but response time may not be optimal.

The C/C++ Sample Application

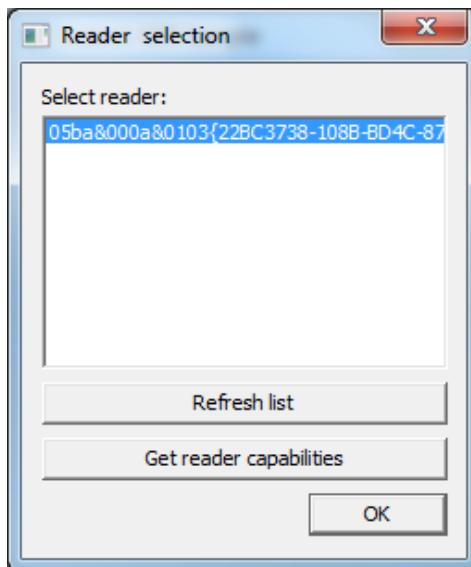
U.are.U SDK includes a sample application to demonstrate the features of the SDK. The sample application is located in the `Samples` folder. The compiled file, `UareUSample.exe` can be downloaded to your reader for

testing. Depending on your version of Visual Studio, you can use `UareUSample2010.vcproj` or `UareUSample2008.vcproj`.

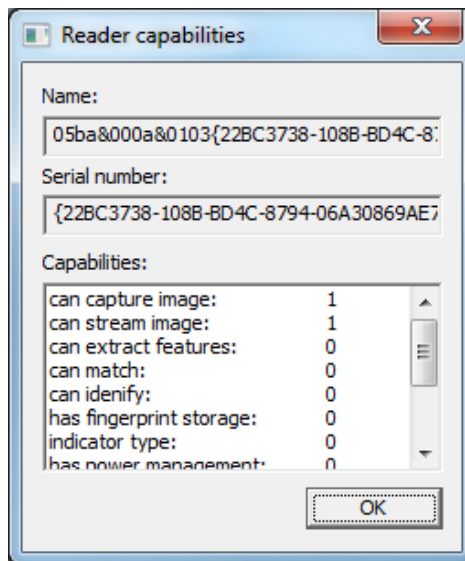
The application demonstrates the features of the SDK. When you launch the application, you see the main screen as shown below.



Click on **Reader Selection** to open a reader. All available readers will be displayed, as shown on the screen below.

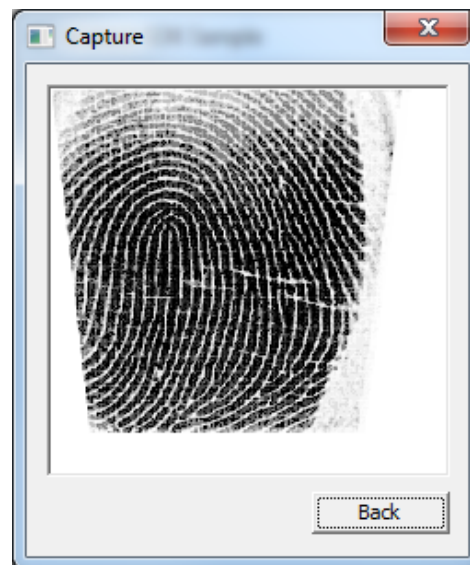


Clicking on the **Get reader capabilities** button will display additional information about the selected reader, as shown below.



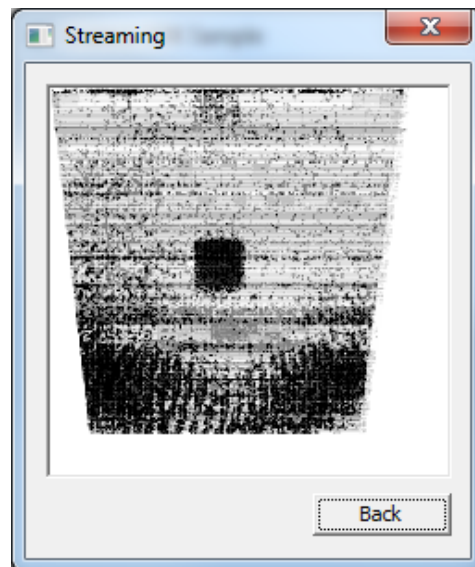
Click **OK** to return to the previous screen. Click **OK** to select the reader. At the point, you are returned to the main screen and all of the buttons are enabled.

Click on the **Capture** button to put the reader into capture mode and you can press your finger onto the reader to capture a fingerprint and display it on the screen as shown below.

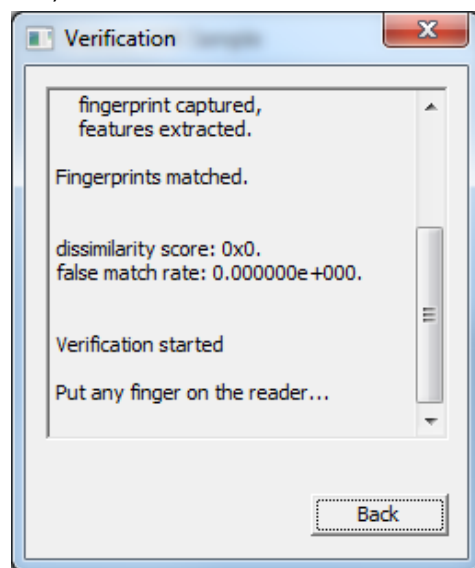


Click on the **Back** button to return to the main screen.

To see a demonstration of the streaming feature, click on the **Streaming** button to put the reader into streaming mode and you can press your finger onto the reader to capture a fingerprint and display it on the screen as shown below.

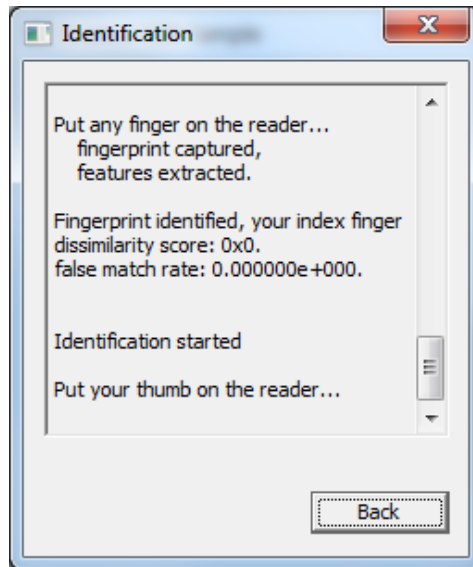


After you click on **Back**, you can click on the **Verification** button next. You will be prompted to put your finger onto the reader. Then you can put a second finger on the reader. If you use the same finger, you will see a message that the fingerprints matched, as shown below.

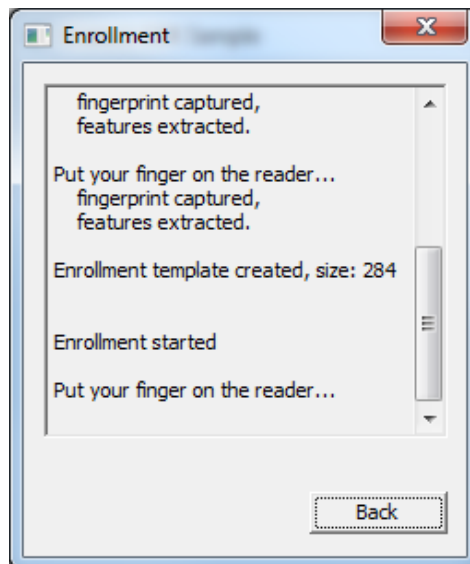


When you click on **Back** you will return to the main screen.

Click on **Identification** to test the next component of the sample program. You will be prompted to provide a thumbprint, index finger, etc. 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.



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.

Pre-Requisites

This chapter assumes that you have a working knowledge of .NET and that you know how to develop for Windows readers. 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 2010
- .NET Framework 2.0

Target Runtime Hardware (Windows Reader)

The Windows-based reader 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 reader 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

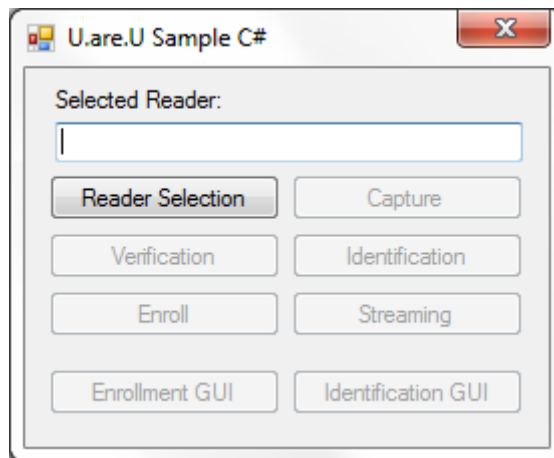
The .NET Sample Application

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.

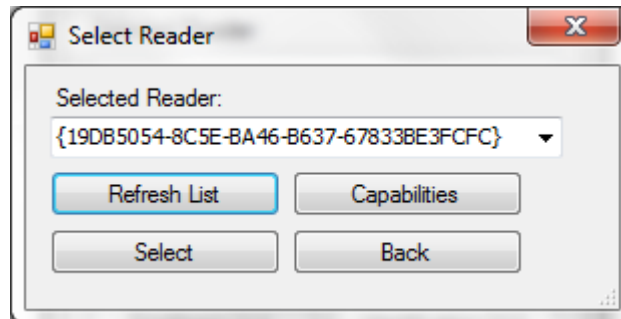


The sample program demonstrates:

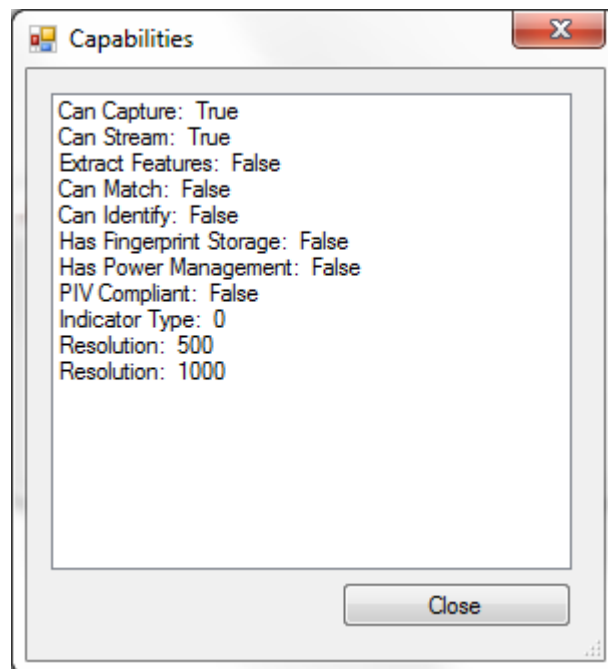
- How to capture fingerprints both in scan mode and in streaming mode
- How to enroll a subject finger
- How to identify a fingerprint
- How to verify a fingerprint
- The built-in control for enrollment
- The built-in control for identification

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 put the device into capture mode and you can press your finger onto 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 pressing a finger to the scanner plate. Click on the **Back** button to return to the main screen.

Testing Streaming Mode

Click on the **Streaming** button from the main dialog to put the device into streaming capture mode and you can press your finger onto the reader to capture a fingerprint and display it on the screen as shown below.

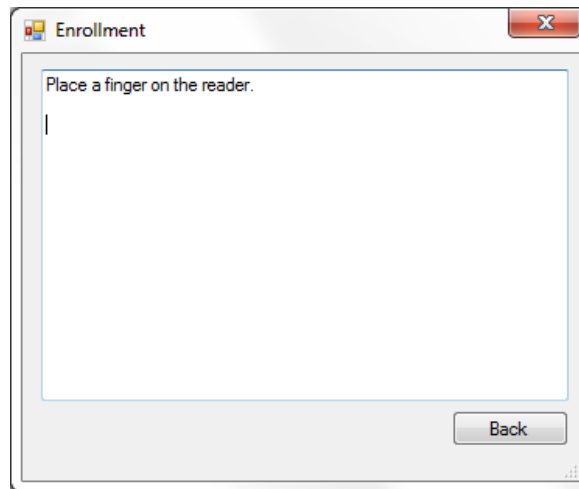


While the reader is in streaming mode, you can capture repeatedly by pressing a finger to the scanner plate. Click on the **Back** button to return to the main screen.

Enrolling a Finger

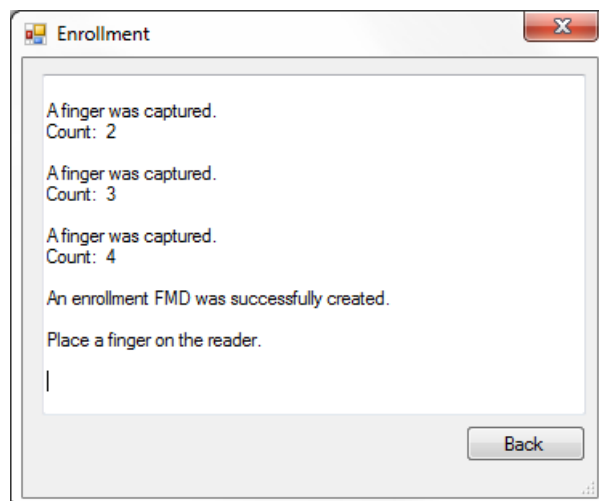
Click on **Enrollment** to begin enrolling the first test subject.

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.)

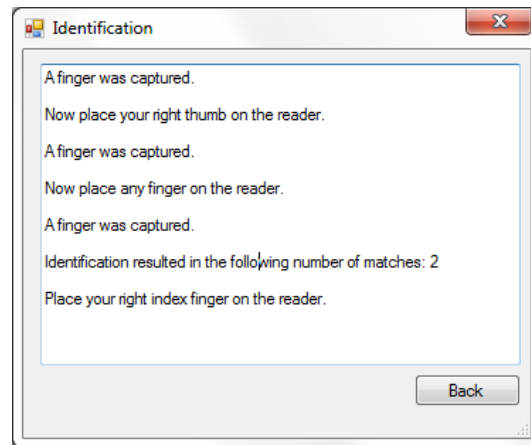
Once the enrollment is complete, you will see confirmation that the enrollment process is finished, as shown in the screen below. In this case, four fingerprint scans were sufficient.



Note that the enrollment FMD this is created is stored in memory only and will be deleted when you click the **Back** button.

Identifying a Fingerprint

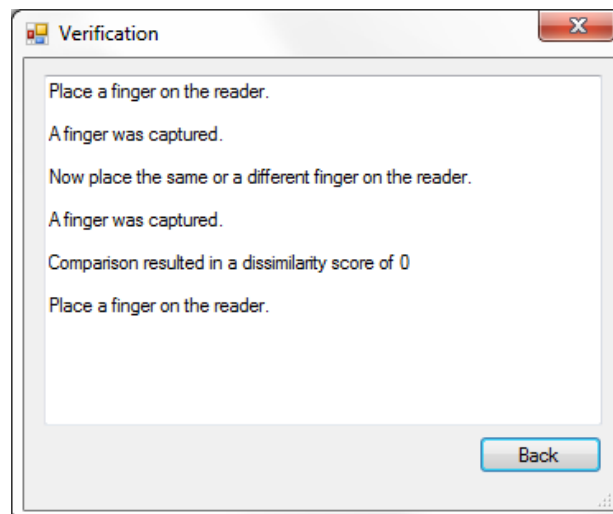
To test the identification feature, click on the **Identification** button. Recall that identification is a 1-to-many comparison where the application would normally search through all of the enrolled fingers to find a match. For this sample, we don't have any enrolled fingers, so you will be prompted 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

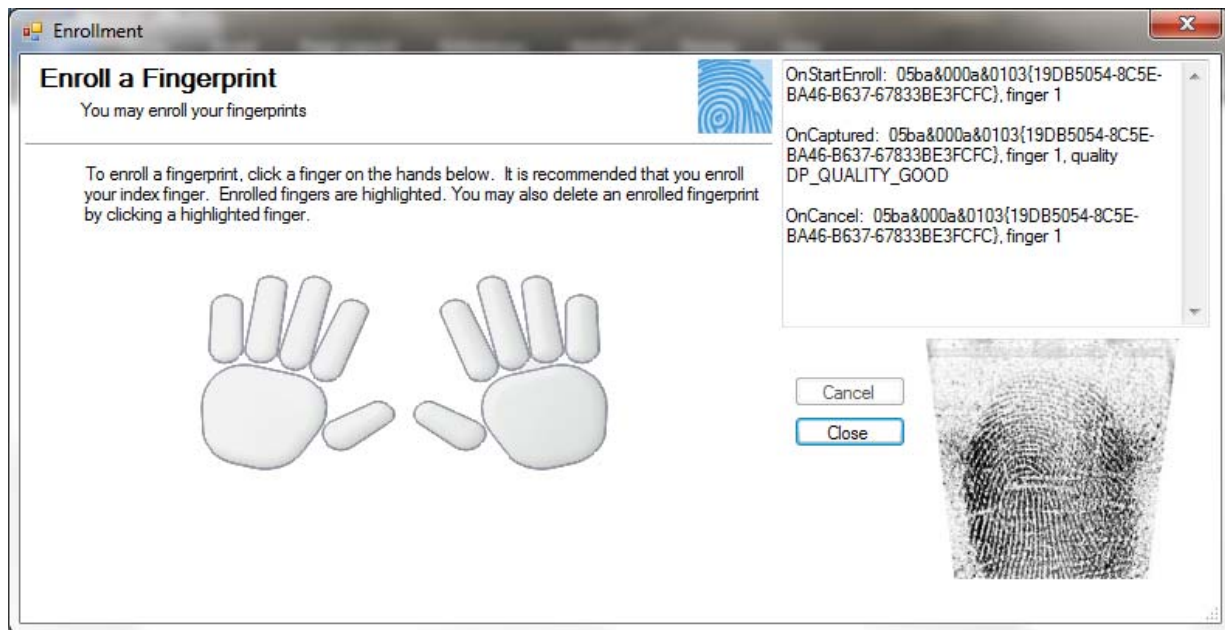
To test the verification feature, click on the **Verification** button. Recall that verification is a 1-to-1 comparison where the application matches against a specified fingerprint. When you click the **Verification** button, you will be prompted to place your finger on the reader. In the screen below, 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 **Cancel** button on this window, 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.

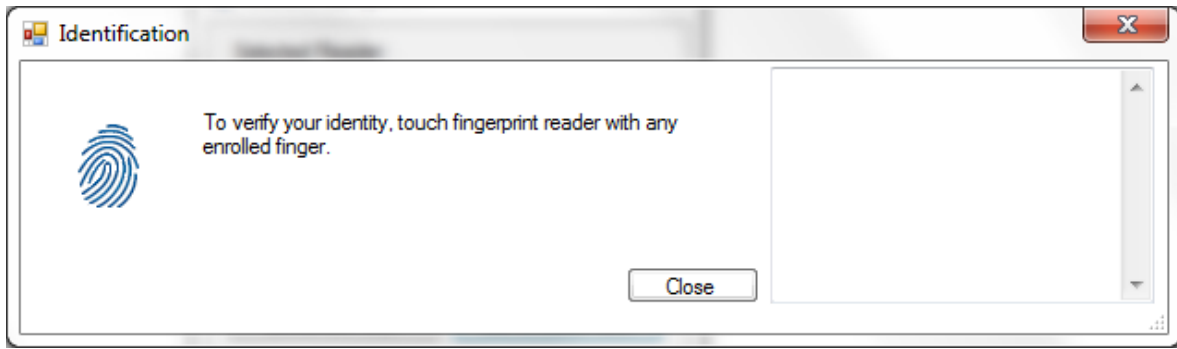
To delete an enrolled fingerprint, click on an enrolled finger in this dialog and you will be prompted to confirm that you wish to delete the fingerprint for the finger that you clicked on.

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**.

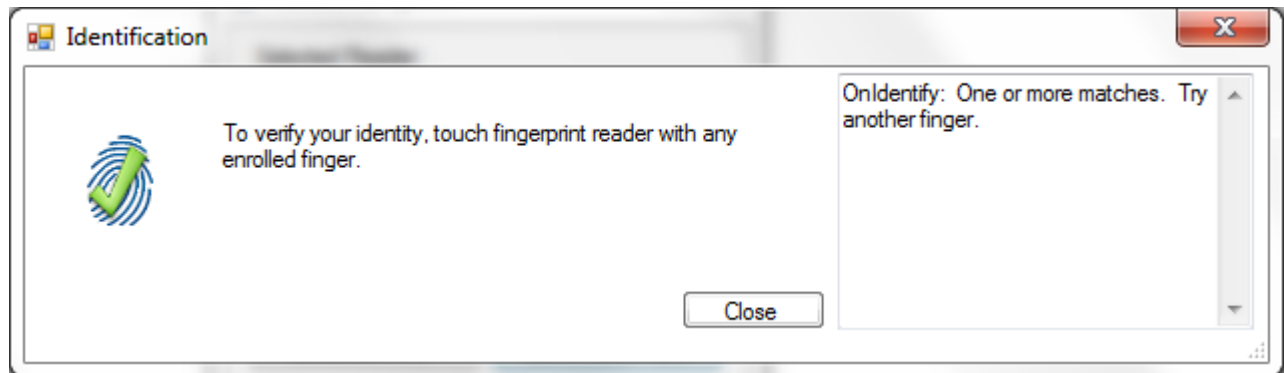
Testing the Identification UI Control

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 info 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.

Pre-Requisites

This chapter assumes that you have a working knowledge of .NET and ActiveX and that you know how to develop for Windows readers. 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
- DPctIXUru.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 environment, 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

The ActiveX Samples

Sample HTML pages are stored in the `UareUSampleActiveX` folder to demonstrate ActiveX usage. Since ActiveX has been implemented as a wrapper to the .NET components of the SDK, the ActiveX samples demonstrate the same features as the .NET samples, documented in *The .NET Sample Application* on page 15.

Pre-Requisites

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

System Requirements

Development System

- Microsoft Windows XP Professional or higher, 32-bit or 64-bit
- Microsoft Visual Studio 2008 or 2010
- Java SE 6 (JDK 6) or newer

Target Runtime Hardware (Windows Reader)

The Windows-based reader 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 are (in Kb):

	x86	x64
Capture runtime (drivers + SDK layer) with fingerprint recognition	100	120

In addition, the reader 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 6, 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.

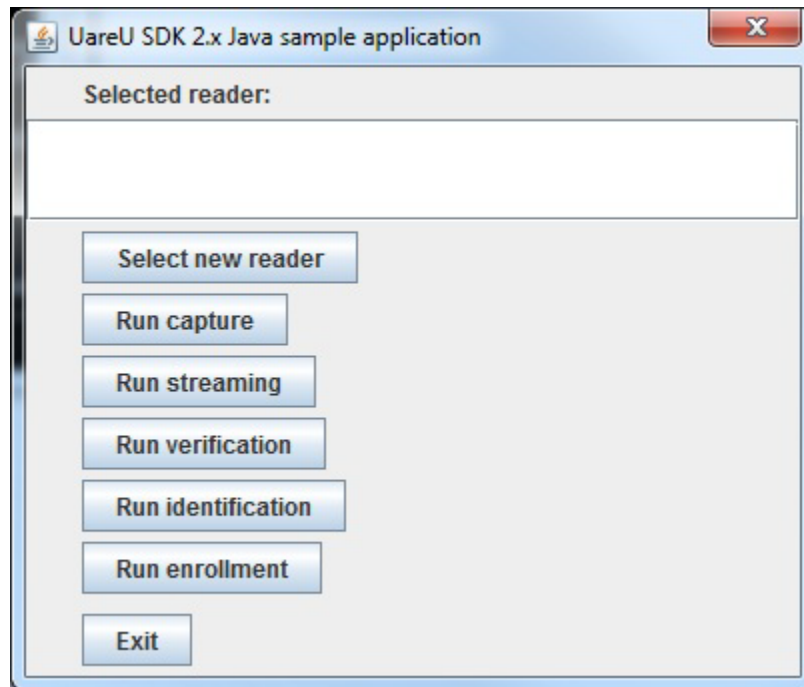
2. Make sure that `dpuareu.jar` is in the classpath and `dpuareu_jni.dll` is accessible by JVM. For example:

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

The Java Sample Application

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 reader 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.



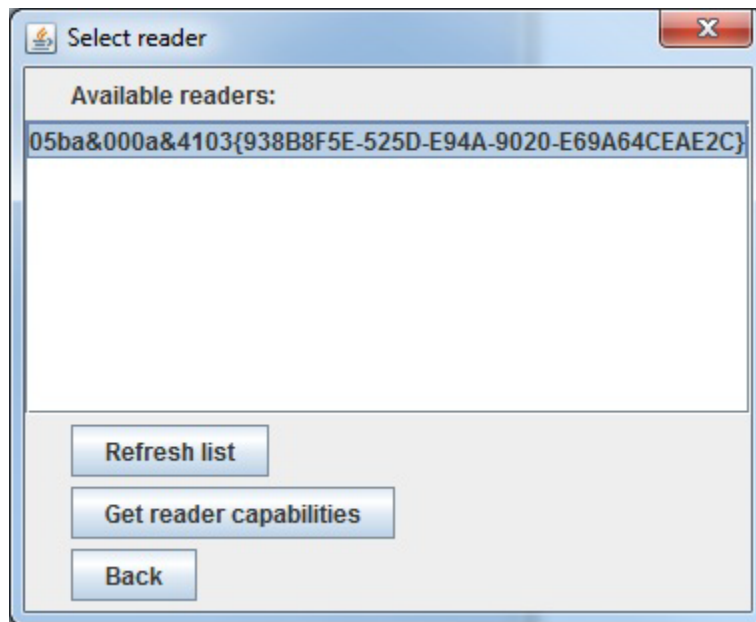
The sample program demonstrates:

- How to enroll a subject 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

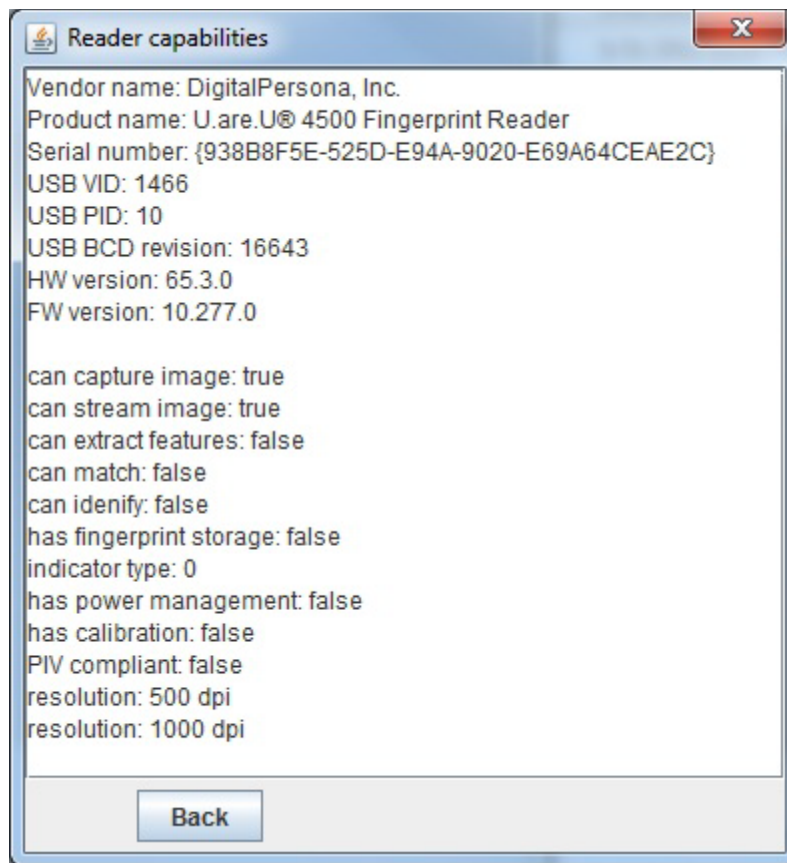
Selecting a Reader

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 below.



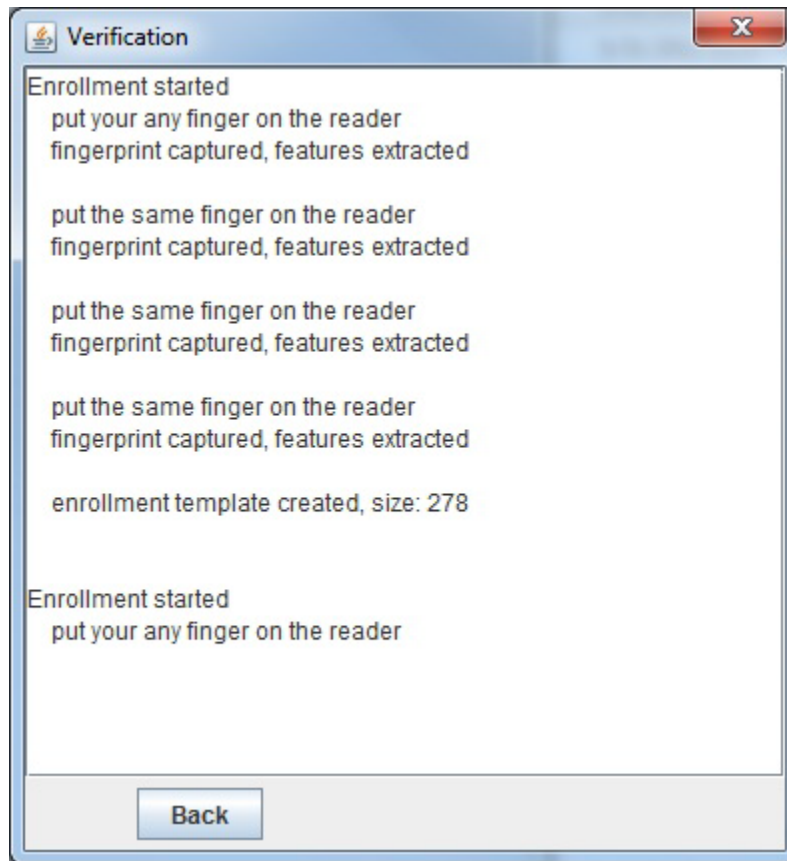
Click on the **Back** button to continue.

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.



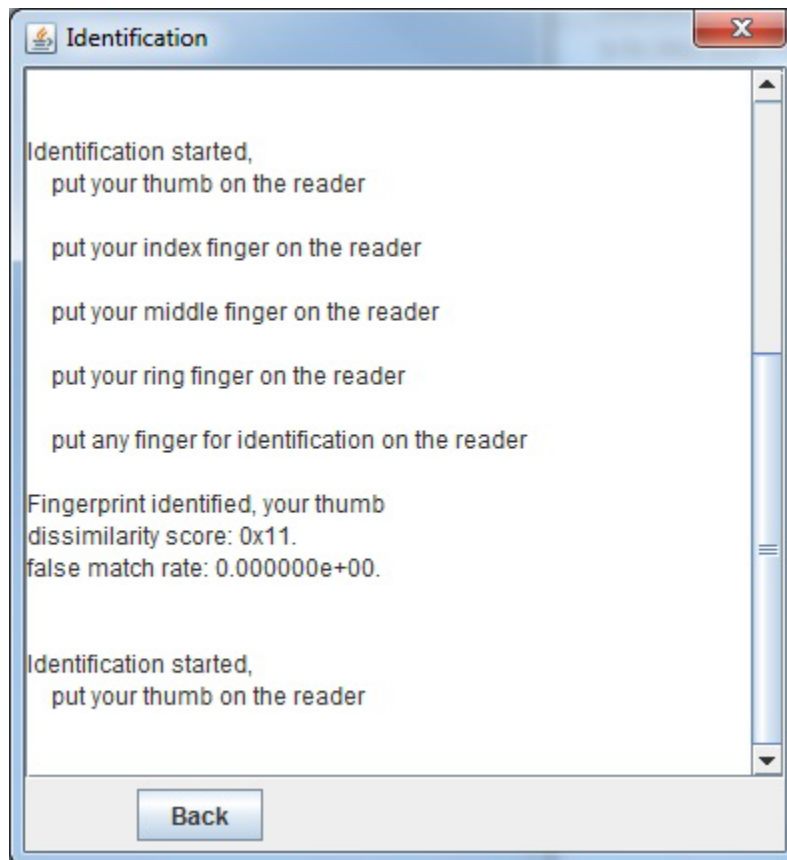
After the first finger is successfully scanned, you will be prompted to scan additional fingers until a sufficient number of high quality scans are complete. The number of fingers requested will vary depending on the image scans - the enrollment functions will continue to request scans until an acceptable 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 put fingers on the reader so that the application has some fingerprints to check against.

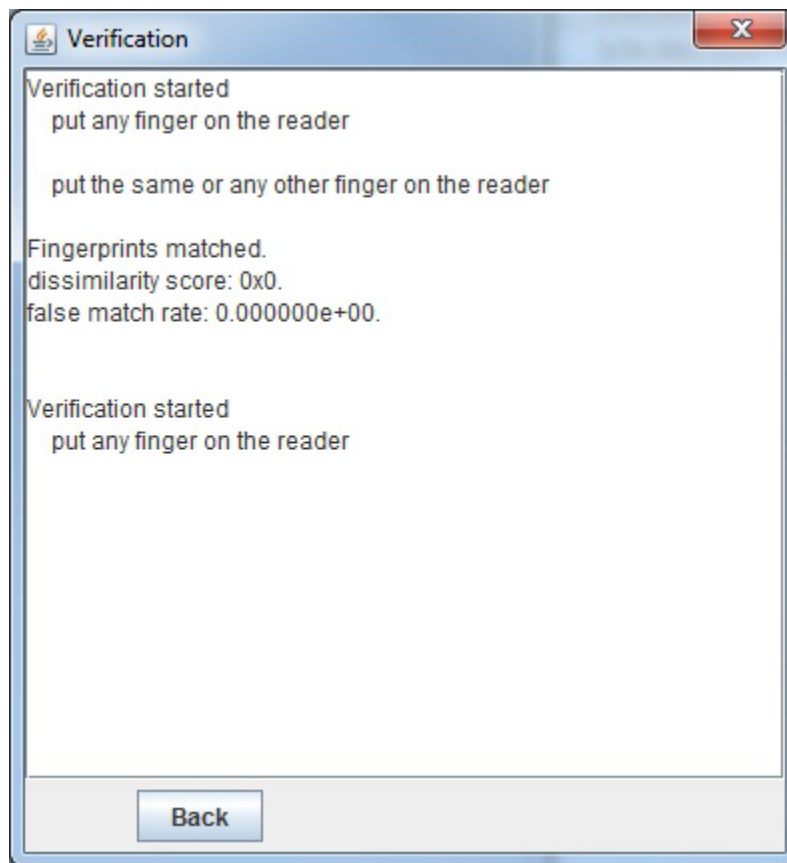
After the application scans four fingers, you will be prompted to put any finger on the reader to identify against the fingers that were just scanned. If you press a finger that was previously scanned on the reader, you will see that a match was found. In the screen image below, we successfully identified 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 put the same finger or another finger, to verify against the first finger. In the screen below, we have successfully verified a user.

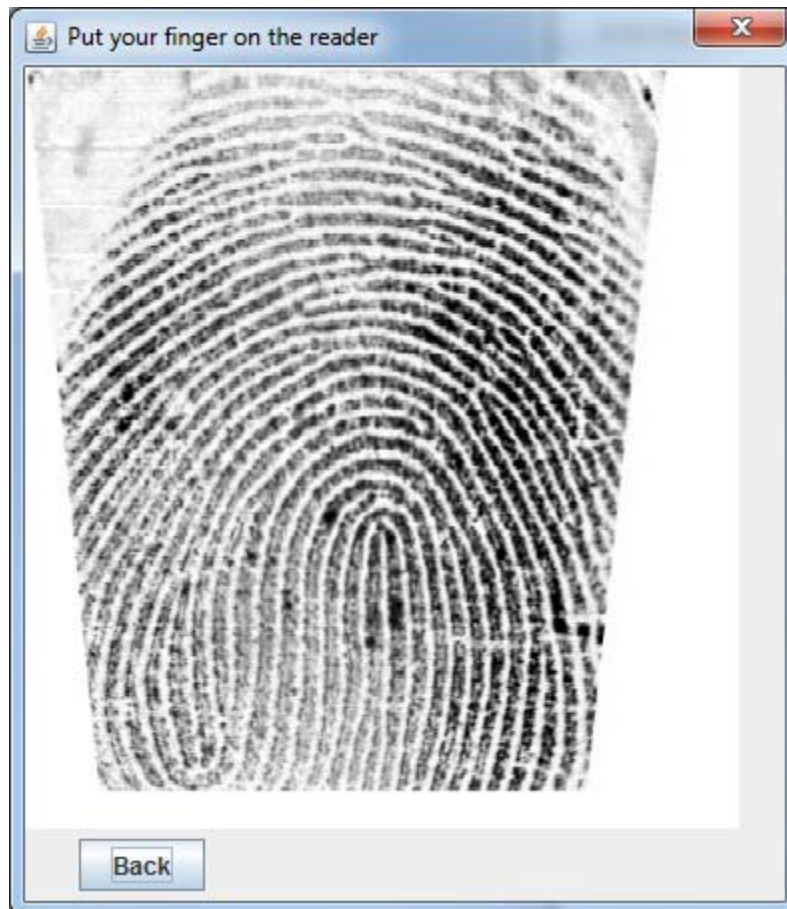


To exit identification mode, click on the **Back** button.

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 puts 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**.

Pre-Requisites

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

System Requirements

Development System

- Microsoft Windows XP Professional or higher, 32-bit or 64-bit
- Microsoft Visual Studio 2008 or 2010
- Java SE 6 (JDK 6) or newer

Target Runtime Hardware (Windows Reader)

The Windows-based reader 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 are (in KB):

	x86	x64
Capture runtime (drivers + SDK layer) with fingerprint recognition (wrapper only -- not including the base Java and C/C++ APIs)	1929	1948

Extra Installation Steps

After installing as described in *Installing on the Development and Target Systems* on page 6, 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.are.U
```

```
SDK\Windows\Lib\Java\dpwareu.jar" -Djava.library.path="C:\Program
Files\DigitalPersona\U.are.U SDK\Windows\Lib\win32" UareUSampleJava
```

Registering your Device after Installation

To enable DigitalPersona 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\dpwareu.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

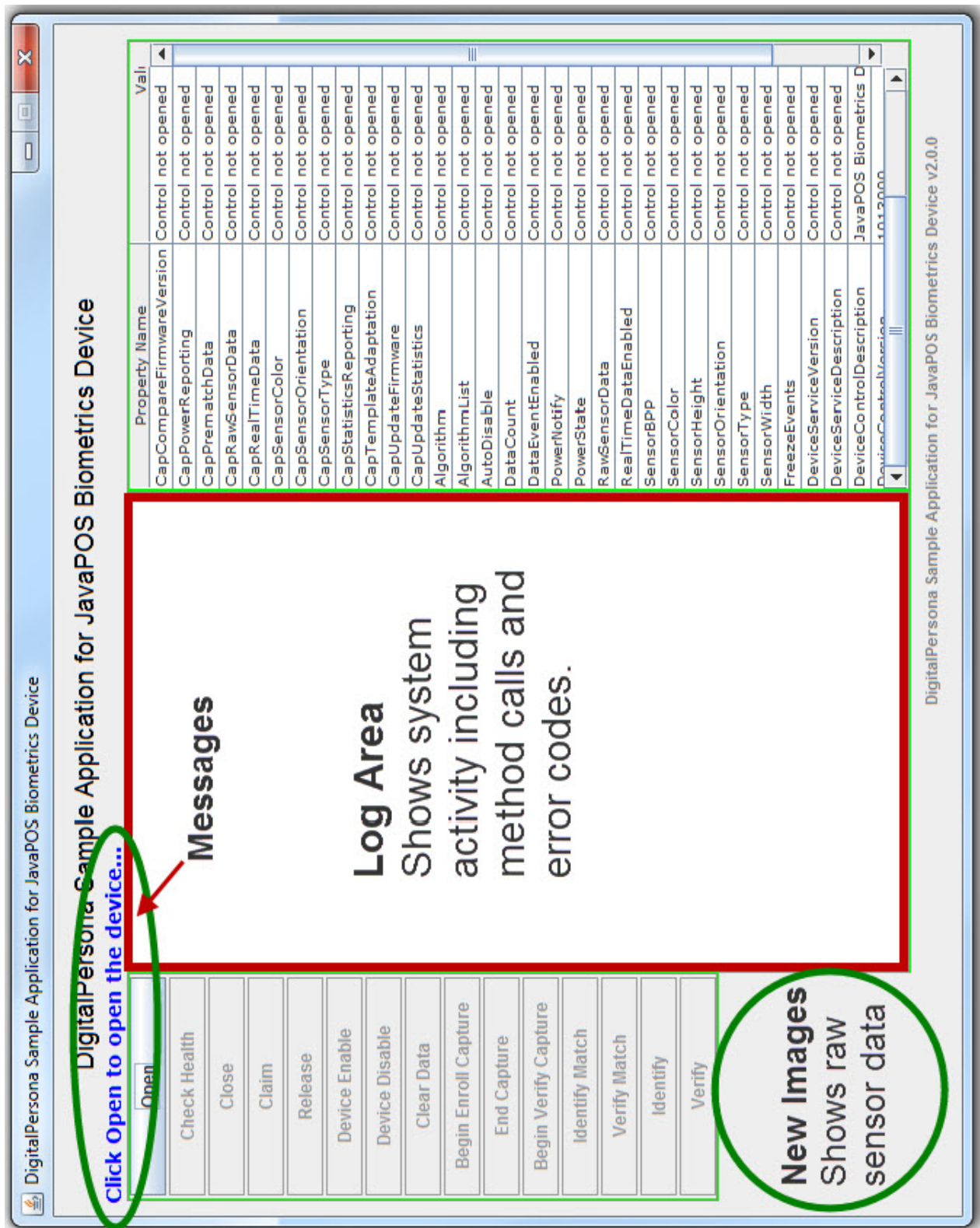
This section describes the functionality of the sample application, which is located in the <Install Directory>\U.are.U SDK\Windows\Samples directory. For more information about the sample application and the sample code, particularly button functionality, refer to the `readme.txt` file located in the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

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

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 U.are.U SDK window is made up of the following four 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 user of system activity, invite the user to perform actions such as touching the fingerprint reader, or advise the user 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

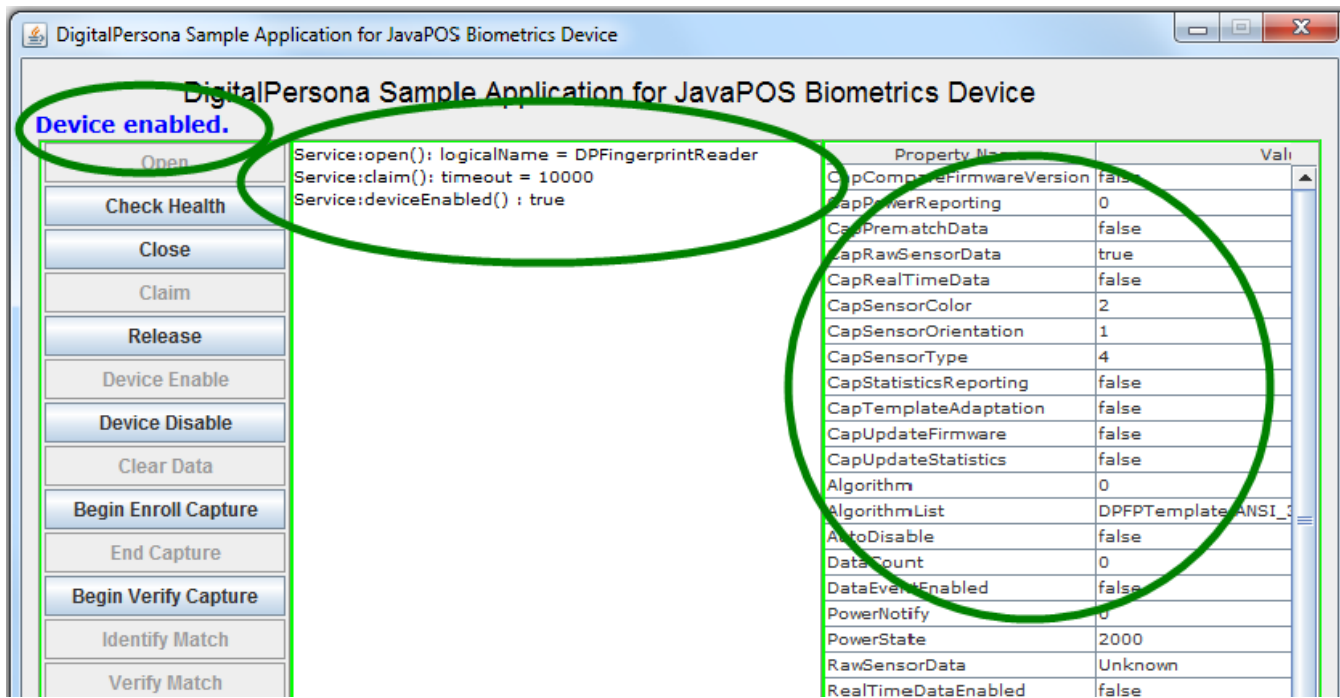
- Click **Open**.

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

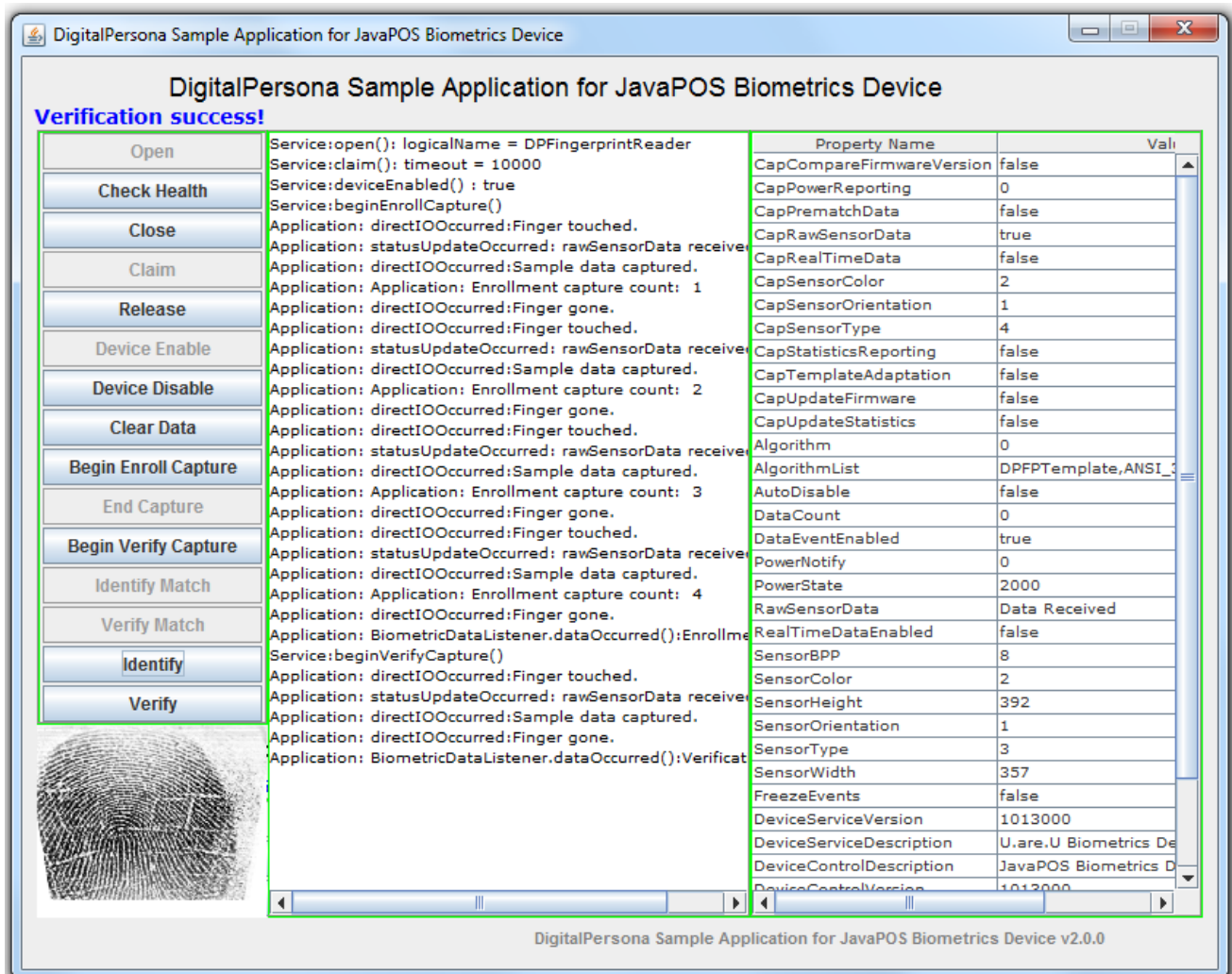
If the method call succeeds, the enrollment template array set and the verification template are cleared. A new verification template and a set of enrollment templates can now be created.

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.

Pre-Requisites

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

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 Reader)

The Windows-based reader 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 (in KB):

	x86	x64
Capture runtime (drivers + SDK layer) with fingerprint recognition (wrapper only -- not including the base C/C++ API)	857	857

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.

U.are.U SDK Sample Application for OPOS

Common Methods

Open()

Claim()

Close()

Clear Data

Result Code

Extended Result Code

Specific Methods

Begin Enroll Capture

Begin Verify Capture

Verify

Verify Match

Identify

Identify Match

Common Properties

AutoDisable

☐ True ☒ False

CapCompare FirmwareVersion

☐ True ☒ False

CapStatistics Reporting

☐ True ☒ False

CapUpdate Firmware

☐ True ☒ False

CapUpdate Statistics

☐ True ☒ False

PowerNotify

☐ PR_NONE ☐ PR_STANDARD ☒ PR_ADVANCED

CapPower Reporting

☐ PS_ONLINE ☒ PS_UNKNOWNWN ☐ PS_OFF ☐ PS_OFFLINE ☐ PS_OFF_OFFLINE

PowerState

Specific Properties

S_ERROR ☐ S_BUSY' ☐ S_CLOSED ☒ S_IDLE

PhysicalDevice Name

PhysicalDevice Description

DeviceService Version

Claimed

☐ True ☒ False

CheckHealth Text

DeviceControl Description

☐ True ☒ False

FreezeEvents

☐ True ☒ False

DeviceEnabled

☐ True ☒ False

DataEvent Enabled

☐ True ☒ False

DeviceControl Version

DeviceService Description

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, the 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 as shown in the screenshot below.

U.are.U SDK Sample Application for OPOS

Common Methods

Open() Claim() Close() Clear Data

Specific Methods

Begin Enroll Capture Begin Verify Capture Verify Verify Match Identify Identify Match

Device Claimed

Common Properties

AutoDisable: ☐ True ☒ False State: ☐ S_ERROR ☐ S_BUSY' ☐ S_CLOSED ☒ S_IDLE OPOS Biometrics Control

CapCompare FirmwareVersion: ☐ True ☒ False PhysicalDevice Name: DPFingerprintReader FreezeEvents: ☐ True ☒ False

CapStatistics Reporting: ☐ True ☒ False PhysicalDevice Description: 05ba&000b&0001(CB615635) DeviceEnabled: ☒ True ☐ False

CapUpdate Firmware: ☐ True ☒ False DeviceService Version: 1013000 DataEvent Enabled: ☒ True ☐ False

CapUpdate Statistics: ☐ True ☒ False Claimed: ☒ True ☐ False DeviceControl Version: 1013000

PowerNotify: ☐ PN_ENABLED ☒ PN_DISABLED CheckHealth Text: Internal HCheck:Success DeviceService Description: Fingerprint Reader Unified

CapPower Reporting: ☒ PR_NONE ☐ PR_STANDARD ☐ PR_ADVANCED

PowerState: ☐ PS_ONLINE ☒ PS_UNKNOWN/N ☐ PS_OFF ☐ PS_OFFLINE ☐ PS_OFF_OFFLINE

Result Code: 0

Extended Result Code: 0

Open method is called... RetVal = 0
 Open method success. RetVal = 0
 ClaimDevice method is called... RetVal = 0
 ClaimDevice Success. RetVal = 0
 DeviceEnabled Property has been set to true.
 DataEventEnabled Property has been set to true.

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 might look like the screen shot shown below.

U.are.U SDK Sample Application for OPOS

Common Methods

Open() Claim() Close() Clear Data

Specific Methods

Begin Enroll Capture Begin Verify Capture Verify Match Identity Match

Candidate Array: 0

Common Properties

AutoDisable ☐ True ☒ False State ☐ S_ERROR ☐ S_BUSY ☐ S_CLOSED ☒ S_IDLE

CapCompare FirmwareVersion ☐ True ☒ False PhysicalDevice Name DPFingerprintReader FreezeEvents ☐ True ☒ False

CapStatistics Reporting ☐ True ☒ False PhysicalDevice Description 05ba&000b&0001 (CB615635) DeviceEnabled ☐ True ☒ False

CapUpdate Firmware ☐ True ☒ False DeviceService Version 1013000 DataEvent Enabled ☐ True ☒ False

CapUpdate Statistics ☐ True ☒ False Claimed ☐ True ☒ False DeviceControl Version 1013000

PowerNotify ☐ PN_ENABLED ☒ PN_DISABLED Internal HCheck: Success! DeviceService Description Fingerprint Reader Unified

CapPower Reporting ☒ PR_NONE ☐ PR_STANDARD ☐ PR_ADVANCED

PowerState ☐ PS_ONLINE ☒ PS_UNKNOWN/N ☐ PS_OFF ☐ PS_OFFLINE ☐ PS_OFF_OFFLINE

Specific Properties

DeviceControl Description OPOS Biometrics Control

DeviceService Description Fingerprint Reader Unified

Result Code 0

Extended Result Code 0

BeginVerifyCapture method is called...
 BeginVerifyCapture Successful. Retval = 0
 Verification Capture method success. Retval = 0
 DataEventEnabled Property has been set to true.
 IdentityMatch called
 IdentityMatch Successful. Retval = 0

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 **clearInput** 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.

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, DigitalPersona 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.

		C/C++		.NET		ActiveX		Java		JavaPOS		OPOS	
Merge Module File	Description	x86	x64	x86	x64	x86	x64	x86	x64	x86	x64	x86	x64
DPDevices	Device components	x		x		x		x		x			
DPDevices64			x		x		x		x		x		
DpDrivers		x	x	x	x	x	x	x	x	x	x		
DPFingerJet	PIV-certified FingerJet Engine	x		x		x		x		x			
DPFingerJet64			x		x		x		x		x		
DPFPApi	Device APIs	x		x		x		x		x			
DPFPApi64			x		x		x		x		x		
DPFPCapture	Fingerprint capture	x		x		x		x		x			
DPFPCapture64			x		x		x		x		x		
DPHostServiceSDK	Host Service SDK	x		x		x		x		x			
DPHostServiceSDK64			x		x		x		x		x		
DPWorkstationPro	DigitalPersona Workstation Pro components	x		x		x		x		x			
DPWorkstationPro64			x		x		x		x		x		
DPPProUtils		x		x		x		x		x			
DPPProUtils64			x		x		x		x		x		
DPPIVDrivers	PIV drivers	x		x		x		x		x			
DPPIVDrivers64			x		x		x		x		x		
DPJavaPOS	JavaPOS libraries									x	x		
DPUareUJava	Java libraries							x	x	x	x		
DPUareUJni	Native libraries and JNI wrapper							x		x			
DPUareUJni64									x		x		
DPUareUX	ActiveX libraries					x	x						
DPUareUNET	.NET libraries			x	x								
DPOpos	OPOS libraries											x	x

Fingerprint Reader Documentation

You may redistribute the documentation included in the `Redist` directory to your end users pursuant to the terms of this section and of the EULA, attendant to the software and located in the `Windows\Docs` directory in the installed product directory.

Hardware Warnings and Regulatory Information

If you distribute DigitalPersona U.are.U fingerprint readers to your end users, you are responsible for advising them of the warnings and regulatory information included in the **Warnings and Regulatory Information.pdf** file in the `Redist` directory. You may copy and redistribute the language, including the copyright and trademark notices, set forth in the **Warnings and Regulatory Information.pdf** file.

Fingerprint Reader Use and Maintenance Guide

The DigitalPersona U.are.U Fingerprint Reader Use and Maintenance Guide, **DigitalPersona Reader Maintenance.pdf**, is located in the `Redist` directory. You may copy and redistribute the **DigitalPersona Reader Maintenance.pdf** file, including the copyright and trademark notices, to those who purchase a U.are.U module or fingerprint reader from you.