DigitalPersona, Inc.

U.are.U SDK
Version 2.x

Platform Guide for Windows
# 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 ..................... 7  
      Step 2: Installing on the Target Hardware ......................... 9  
   Authentication Service ...................................................... 10  
   Uninstalling ...................................................................... 10

3 Developing Applications with C/C++ .................................... 11  
   Pre-Requisites .................................................................. 11  
   System Requirements ....................................................... 11  
   The C/C++ Sample Application ........................................... 11

4 Developing Applications with .NET ...................................... 16  
   Pre-Requisites .................................................................. 16  
   System Requirements ....................................................... 16  
   Static libraries and DLLs ................................................... 16  
   The .NET Sample Application ............................................ 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  
      Using the Streaming Feature .......................................... 23

5 Developing Applications with ActiveX / .NET ....................... 24  
   Pre-Requisites .................................................................. 24  
   Overview ......................................................................... 24  
      Static libraries and DLLs ............................................... 24  
      ActiveX Control Unique Identifiers ................................. 24  
   The ActiveX Samples ....................................................... 25

6 Developing Applications with Java ...................................... 26  
   Pre-Requisites .................................................................. 26  
   System Requirements ....................................................... 26  
      Installing on the Target Hardware .................................... 26  
   The Java Sample Application ............................................ 27
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selecting a Reader</td>
<td>28</td>
</tr>
<tr>
<td>Enrolling a Finger</td>
<td>30</td>
</tr>
<tr>
<td>Identifying a Fingerprint</td>
<td>31</td>
</tr>
<tr>
<td>Verifying a Fingerprint</td>
<td>32</td>
</tr>
<tr>
<td>Using the Capture and Streaming Feature</td>
<td>33</td>
</tr>
</tbody>
</table>
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 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 application.

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.

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

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.
Step 1: Installing on the Development System

To install the SDK on your development system:

1. Unzip the distribution file into a folder.
2. Run `SDK\x86\setup.msi` OR `SDK\x64\setup.msi`

Files Installed on 32-bit Development Systems

The files installed on the Developer's machine (SDK installation, 32-bit) are:

GlobalAssemblyCache
- DPCtlUruNet.dll
- DPCtlXUru.dll
- DPUrNet.dll
- DPXUru.dll

Windows\System32
- DpClback.dll
- dpfj.dll
- DPPPApi.dll
- dpfpdd.dll

Program Files\DigitalPersona\Pro Workstation\Bin
- DPAppSyn.dll
- DPCms.dll
- DPCOper2.dll
- DPDevice2.dll
- DPDevTS.dll
- DpFnd2.dll
- DpHostW.exe
- DPJasPer.dll
- DPMux.dll
- DPPTUtils.dll
- DpSvInfo2.dll
- DPTSClnt.dll

Program Files\DigitalPersona\U.are.U SDK
Include
- dpfj.h
- dpfj_compression.h
- dpfpdd.h

Windows
Docs
- U.are.U_SDK_Developer_Guide.pdf

Lib
.NET
- DPCtlUruNet.dll
- DPCtlXUru.dll
- DPUrNet.dll
- DPXUru.dll
Chapter 2: Installation
Step 1: Installing on the Development System

Java
  dpuareu.jar
  win32
dpfj.dll
  dpfj.lib
  dpfpdd.dll
  dpfpdd.lib
  dpuareu_jni.dll
x64
dpfj.dll
  dpfj.lib
  dpfpdd.dll
  dpfpdd.lib
  dpuareu_jni.dll

Samples
  Bin
    .NET
      UareUsampleCSharp.exe
      win32
      UareUSample2010.exe
      x64
      UareUSample2010.exe
    Java
      run_win32.bat
      run_x64.bat
      UareUSampleJava.jar
  Include
    <WTL80, needed to compile C++ sample code>
    UareUSample
    <C++ sample code>
    UareUSampleCSharp
    <C# sample code>
    UareUSampleJava
    <Java sample code>

Additional Files Installed on 64-bit Development Systems

For 64-bit development systems, the SDK installation installs these additional files:

Program Files (x86)\DigitalPersona\Pro Workstation\Bin
  DPAppSyn.dll
  DPCms.dll
  DPDevTS.dll
  DpFnd2.dll
  DPJasPer.dll
  DPPTUtils.dll

Windows\SysWOW64
  DPClback.dll
  dpfj.dll
  DPFPApi.dll
  dpfpdd.dll
Step 2: Installing on the Target Hardware

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

1. Unzip the distribution file into a folder on the target machine.
2. Run RTE\x86\setup.msi OR RTE\x64\setup.msi

Files Installed on 32-bit Target Systems

The files installed when installing the run-time on 32-bit target hardware are:

- GlobalAssemblyCache
  - DPCtlUruNet.dll
  - DPCtlXUru.dll
  - DPUruneNet.dll
  - DPXUru.dll

- Windows\System32
  - DpClbck.dll
  - dpfj.dll
  - DPFPApi.dll
  - dpfpdd.dll

- Program Files\DigitalPersona\Pro Workstation\Bin
  - DPAppSyn.dll
  - DPCms.dll
  - DPCOper2.dll
  - DPDvice2.dll
  - DPDevTS.dll
  - DpFnd2.dll
  - DpHostW.exe
  - DPJasPer.dll
  - DPMux.dll
  - DPPTUtilis.dll
  - DpSvInfo2.dll
  - DPTSCInt.dll

Additional Files Installed on 64-bit Target Systems

When installing the run-time on 64-bit target hardware, the following additional files are installed:

- Windows\SysWOW64
  - DPClbck.dll
  - dpfj.dll
**Chapter 2: Installation**

**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:

<table>
<thead>
<tr>
<th></th>
<th>x86</th>
<th>x64</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capture runtime (drivers + SDK layer)</td>
<td>5.0 MB</td>
<td>5.5 MB</td>
</tr>
<tr>
<td>Fingerprint recognition runtime</td>
<td>160 KB</td>
<td>220 KB</td>
</tr>
</tbody>
</table>

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.

![Reader capabilities window](image)

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.

![Capture window](image)

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.

![Streaming](image)

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.

![Verification](image)

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).

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 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 a .NET sample application to demonstrate the features of the SDK. The sample application is located in the Samples folder. The compiled file, UareUSampleCSharp.exe can be downloaded to your reader for testing or you can use UareUSampleCSharp_.csproj in Visual Studio.

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

![Sample Application Screen](image)

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
Enrolling a Finger

Click on **Enroll** 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, three fingerprint scans were sufficient.
Identifying a Fingerprint

To test the identification feature, click on the **Identify** button. Recall that identification is a 1-to-many comparison where the application searches through all of the enrolled fingers to find a match. When you click the **Identify** button, you will be prompted to place your finger on the reader. If you press an enrolled finger on the reader, you will see that a match was found. In the screen image below, we have tried to identify two fingers -- neither identification succeeded -- the fingers were not enrolled.

![Identify screenshot](image)

Verifying a Fingerprint

To test the verification feature, click on the **Verify** button. Recall that verification is a 1-to-1 comparison where the application matches against a specified fingerprint. When you click the **Verify** button, you will be prompted to place your finger on the reader. As with the identification example above, in the screen below, we have tried to verify two fingers -- the first verification succeeded and the second attempt was a finger that was not enrolled.

![Verify screenshot](image)
Testing the Enrollment UI Control

If you look at the sample code, you will see that the enrollment test (as discussed above) called functions in the SDK. An alternate way to use the SDK is to use the pre-built control for enrollment. To see 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:
As you scan your finger, you can see the events and status information on the right in the window, 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 as shown below. 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. Click **Close** to exit the control.
Testing the Identification UI Control

If you look at the sample code, you will see that the identification test (as discussed above) called functions in the SDK. An alternate way to use the SDK is to use the pre-built control for identification. To see 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.

![Identification Control](image)

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, followed by an unsuccessful attempt at identification.

![Identification Control](image)

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

To exit the control, click on the **Close** button.
Using the Streaming Feature

The sample application also demonstrates the streaming feature (on fingerprint readers that support that feature). To test streaming, from the main window, click on **Click to start streaming** (at the right on the main window).

This puts the reader into streaming mode and immediately the results of the stream are displayed at the right side of the window. For the sample program, 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.

Removing the finger shows a blank stream.

To exit streaming mode, click on one of the other buttons.

Click **Close** to exit the sample application.
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 GUI
- DPCtlXUru.dll – ActiveX control

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("733A2D1B-9F3D-423D-8700-4F2C8E88EAF9")] - XConversion
- [Guid("A1589E23-FE6E-43D8-9EDF-93142671C47A")] - XEnrollment
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 17.
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):

<table>
<thead>
<tr>
<th></th>
<th>x86</th>
<th>x64</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capture runtime (drivers + SDK layer) with fingerprint recognition</td>
<td>100</td>
<td>120</td>
</tr>
</tbody>
</table>

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.

Installing on the Target Hardware

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


2. Copy U.are.U SDK\Windows\Lib\Java\dpuareu.jar and U.are.U SDK\Windows\Lib\<x86 or x64>\dpuareu_jni.dll 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
```

### 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 use UareUSampleJava_.csproj in Visual Studio.

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

![Java Sample Application Screen](image)

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:

![Select reader window]

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.

![Reader capabilities](image)

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.

![Enrollment process screenshot]

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.

![Verification Screen]

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.