FDx SDK for Java, v1.0
Developer's Manual

For applications using SecuGen® fingerprint peripherals

SG1-0029A Rev 001 (10/2005)

SecuGen END USER LICENSE AGREEMENT

SecuGen Corporation ("SecuGen") is granting you (an individual or an entity, either of which is referred to herein as "Licensee") a license to use SecuGen's Software Developer Kit, including computer software, hardware, associated media and printed materials (individually "SDK" and collectively the "Products") only upon the condition that Licensee accepts all of the terms and conditions contained in this User License Agreement (the "Agreement"). If Licensee does not agree to the terms of this Agreement, Licensee should not install or use the Products, but rather, promptly return the Products to the place of purchase.

1. **Grant of License**. This Agreement grants Licensee a personal, limited, non-transferable, non-exclusive right to install and use one copy of the Products on a single computer exclusively for the following purposes:

    a. Licensee may use the SDK solely for developing, designing, and testing SecuGen software applications for use with SecuGen products ("Applications").
    b. Licensee may modify the sample source code located in the Products' "samples" directories ("Sample Code") to design, develop and test SecuGen Applications. Licensee may also reproduce and distribute the Sample Code in object code form along with any modifications Licensee makes to the Sample Code, provided that Licensee complies with the distribution requirements described below. For purposes of this section, "modifications" shall mean enhancements to the functionality of the Sample Code.
    c. Licensee may copy and redistribute the Sample Code provided that: (a) it is distributed as part of an Application prepared by Licensee; (b) Licensee's Application adds significant and primary functionality to the Sample Code; (c) the Sample Code only operates in conjunction with the Products; (d) Licensee does not permit further redistribution of the Sample Code; (e) Licensee does not use SecuGen's name, logo or trademarks to mark Licensee's Application; (f) Licensee includes a valid copyright notice on Licensee's Application; and (g) Licensee agrees to indemnify, hold harmless, and defend SecuGen from and against any claims or lawsuits, including attorney's fees, that arise or result from the use or distribution of Licensee's Application.
    d. No other uses and/or distribution of the Products or Sample Code are permitted without SecuGen's prior written consent. SecuGen reserves all rights not expressly granted to Licensee.

2. **Limitations or Restrictions.** Licensee may not: (i) modify or translate the Products; (ii) reverse engineer, decompile, or disassemble the Products, except and only to the extent this restriction is expressly permitted by applicable law; (iii) separate the Products component parts to transfer to a third party; (iv) rent, lease, sell or lend the Products.

3. **Intellectual Property Rights.** Licensee acknowledges that no title to the intellectual property in the Products and any components thereof are transferred to Licensee. All rights, title and copyrights in and to the Products and any copies are owned by SecuGen. The Products are protected by copyright laws and international copyright conventions and treaties.

4. **Warranty.** SecuGen's Product warranty runs to the Licensee for a period of thirty (30) days. If the date of receipt by Licensee of Products is the same as the date of purchase by Licensee of the Products, the thirty-day warranty period begins on the date of purchase. If the date of receipt by Licensee of Products is later than the date of purchase of the Products by Licensee, then the thirty-day warranty period begins on the date of shipment by a distributor of the Products to Licensee. During the thirty day warranty period, SecuGen warrants that (a) the media on which the Products or documentation covering Products is delivered shall be free from material defects in workmanship and materials, and (b) the Products, when installed, configured, used and maintained in accordance with SecuGen's then-current published installation, configuration, use and maintenance specifications, will, in its unaltered form, conform substantially to SecuGen's then-current published functional specifications for such Products. SecuGen's sole obligation, for a breach of this warranty shall be for SecuGen to replace the media in the case of breach of this Warranty. SecuGen does not warrant that the Products will meet Licensee's requirements, that the Products will operate in combinations selected for use by Licensee, or that use of the Products will be uninterrupted or error-free. Because not all errors in software can or need be corrected, SecuGen does not warrant that the Products are error-free or that all software errors will be corrected.

5. **LIMITATIONS OF WARRANTY.** THE PRODUCTS ARE DEEMED ACCEPTED BY LICENSEE. SECUGEN DISCLAIMS ALL WARRANTIES NOT EXPRESSLY PROVIDED IN THIS AGREEMENT INCLUDING, WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT. ANY AND ALL RISK ARISING OUT OF THE USE OR PERFORMANCE OF THE PRODUCTS REMAINS WITH LICENSEE. IN NO EVENT SHALL SECUGEN OR ITS SUPPLIERS BE LIABLE FOR

ANY CONSEQUENTIAL, INCIDENTAL, DIRECT, INDIRECT, SPECIAL, PUNITIVE, OR OTHER DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS) ARISING OUT OF THIS AGREEMENT OR THE USE OF OR INABILITY TO USE THE PRODUCTS, EVEN IF SECUGEN HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME STATES/JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY FOR CONSEQUENTIAL OR INCIDENTAL DAMAGES, THE ABOVE LIMITATION APPLIES ONLY TO THE EXTENT PERMITTED BY APPLICABLE LAW.

6. **LIMITATION OF LIABILITY.** SECUGEN'S TOTAL LIABILITY AND LICENSEE'S EXCLUSIVE REMEDY UNDER THIS AGREEMENT SHALL NOT EXCEED THE REPLACEMENT COST OF A SINGLE COPY OF THE PRODUCTS.

7. **Indemnification.** Licensee shall hold harmless, indemnify and defend SecuGen, its officers, directors and employees, from and against any claim, suit or proceeding and any losses, damages, fines and expenses (including attorneys' fees and costs) arising out of or relating to any claims that Licensee's use of the Products in conjunction with any Licensee Application infringes the patent, copyright, trademark, trade secret, or other proprietary rights of any third party, or resulting from any breach of this Agreement by Licensee.

8. **Termination.** Without prejudice to any other rights, SecuGen may terminate this Agreement if Licensee fails to comply with any term or condition of this Agreement. Upon termination of this Agreement, Licensee shall immediately discontinue the use of the Products and certify destruction of all full or partial copies of the Products and related materials provided by SecuGen. Licensee may also terminate this Agreement at any time by destroying the Products and all copies thereof.

9. **General**. Licensee acknowledges that he or she has read this Agreement, understands it, and that by using the Products Licensee agrees to be bound by the terms and conditions set forth in this Agreement. Licensee further agrees that the Agreement is the complete and exclusive statement of the understanding between SecuGen and Licensee which supersedes any proposal or prior agreement, oral or written, and any other communication between SecuGen and Licensee relating to the subject matter of this Agreement. This Agreement may not be modified except in writing duly signed by an authorized representative of SecuGen and Licensee. If any provision of this Agreement is held to be unenforceable for any reason, such provision shall be reformed only to the extent necessary to make it enforceable, and such decision shall not affect the enforceability of such provision under other circumstances, or of the remaining provisions hereof under all circumstances.

10. **U.S. Government Restricted Rights.** Use duplication, or disclosure by the Government is subject to restrictions as set forth subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (c)(1) and (2) of the Commercial Computer Software - Restricted Rights regulation found at 48 CFR 52-227-19, as applicable. Manufacturer is SecuGen Corp., 2356 Walsh Avenue, Santa Clara, California 95051, USA. Licensee acknowledges that the laws and regulations of the United States restrict the export and re-export of the Products. Licensee agrees that the Products in any form will not be exported or re-exported without the appropriate United States and foreign government approval.

THIS PUBLICATION COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY MADE TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED INTO NEW EDITIONS OF THIS PUBLICATION. SECUGEN CORPORATION MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS PUBLICATION AT ANY TIME AND WITHOUT NOTICE.

Contents

Before You Begin

Biometrics Overview

Biometrics is an automated method of recognizing a person based on physical or behavioral characteristics. Biometric information that can be used to accurately identify people includes fingerprint, voice, face, iris, handwriting, and hand geometry.

There are two key functions offered by a biometric system. One method is **identification**, a “one-to-many” matching process in which a biometric sample is compared sequentially to a set of stored samples to determine the closest match. The other is **verification**, a “one-to-one” matching process in which the biometric system checks previously enrolled data for a specific user to verify whether that individual is who he or she claims to be. The verification method provides the best combination of speed and security, especially where multiple users are concerned, and requires a user ID or other identifier for direct matching.

With an increasing reliance on online technology and other shared resources, the information age is quickly revolutionizing the way transactions are initiated and completed. Business transactions of all types are increasingly being handled online and remotely. This unprecedented growth in electronic transactions has underlined the need for a faster, more secure, and more convenient method of user verification than passwords can provide.

Using biometric identifiers offers several advantages over traditional and current authentication methods. This is because only biometric authentication is based on the identification of an intrinsic part of a human being. Tokens such as smart cards, magnetic stripe cards, and physical keys, can be lost, stolen, duplicated, or left behind; passwords can be forgotten, shared, hacked or unintentionally observed by a third party. By eliminating all of these potential trouble spots, only biometric technology can provide the security and convenience needed for today’s complex electronic landscape.

Advantages of Using Fingerprints

The advantages of using fingerprints include widespread public acceptance, convenience, and reliability. It takes little time and effort to acquire one’s fingerprint with a fingerprint identification device, and so fingerprint recognition is considered among the least intrusive of all biometric verification techniques. Ancient officials used thumbprints to seal documents thousands of years ago, and law enforcement agencies have been using fingerprint identification since the late 1800s. Fingerprints have been used so extensively and for so long, there is a great accumulation of scientific data supporting the idea that no two fingerprints are alike.

About SecuGen

SecuGen provides biometric solutions for physical and network security employing advanced fingerprint recognition technology. The company’s comprehensive product line includes quality optical fingerprint sensors and peripherals, software, and development kits used for a variety of innovative applications including Internet, enterprise network and desktop security, physical access control, time and attendance management, and financial and medical records control. SecuGen patented products feature the industry’s longest warranty and are renowned for their accuracy, reliability and versatility. Based in Silicon Valley, SecuGen has been serving the biometric community since 1998 and is an active member of the Biometrics Consortium (www.biometrics.org) and the BioAPI Consortium (www.bioapi.org).

Technical Support

Email	techsupport@secugen.com	**Phone**	+1-408-727-7787
Website	www.secugen.com	**Fax**	+1-408-727-7105
Mail	SecuGen Corporation 2356 Walsh Avenue Santa Clara, California 95051 USA		

About SecuGen Products

SecuGen Sensor Qualities

- **Excellent Image Quality:** Clear, distortion-free fingerprint images are generated using advanced, patented optical methods. Quality imaging yields better sampling for minutiae data extraction.
- **Durability:** Mechanical strength tests show resistance to impact, shock and scratches.
- **Powerful Software:** Precise, fast processing algorithm ensures efficiency and reliability.
- **Ruggedness and Versatility:** Solid engineering and superior materials allows for use under extreme conditions.
- **Ergonomic Design:** Compact, modular design for seamless integration into small devices, ease of use, and compatibility make it ideal for a broad range of applications.
- **Low Cost:** Products are developed to deliver high performance, zero maintenance at very affordable prices for general and industrial use.

Advantages of SecuGen Sensors Over Other Optical Sensors

- Unique optical method captures fine details, even from dry skin
- Extremely low image-distortion
- Reinforced materials
- Wear resistance
- Attractively small size
- Ease of integration
- Ready-to-use
- Low cost through longer life and no maintenance requirements

Advantages SecuGen Sensors Over Semiconductor (Capacitive) Sensors

- Non-metal, non-silicon components make it less susceptible to corrosion when exposed to salts, oil and moisture from skin and environment
- Superior surface properties eliminate need for costly coating and processing procedures
- Greater mechanical strength, wear-resistance, and durability
- Broader range of applicability, especially for use in extreme conditions and climates
- Immunity from electrostatic discharge
- Low cost through longer life and no maintenance requirements

Strengths of SecuGen Software and Algorithms

- Unique image processing algorithm extracts fingerprint minutiae very accurately
- High signal-to-noise ratio processing algorithm screens out false features
- Highly efficient matching algorithm
- Fast overall process of extraction, matching and verification
- Encryption function to protect user privacy
- Compatibility with existing desktop, laptop PCs interface computers
- Ease in developing applications for various purposes

Chapter 1. System Settings

This chapter outlines system requirements, SDK set-up, hardware specifications, and hardware testing the provided programs.

1.1. System Requirements

The FDU SDK for Java all classes of SecuGen fingerprint recognition devices. The following platforms are supported:

- Microsoft Windows Operating System (JNI Library)
- Java2 SDK v1.4.2_09
- Java Plug-in v1.4.2_09
- Internet Explorer v6.0
- Netscape Browser v8.0
- Mozilla FireFox Browser v1.0.6

SecuGen fingerprint devices capture a fingerprint image and digitize the image to an 8-bit gray-scale image at 500 DPI resolution. The host system then retrieves the image through its USB or parallel port for subsequent processing.

- IBM-compatible PC Pentium II – 350 (Recommendation)
- 1 USB or parallel port
- 128MB RAM (Recommendation)
- 20MB available hard disk space
- Microsoft Windows XP, Windows 2000

1.2. Development Environment Configuration

Requires: SecuGen FDx SDK for Windows v3.0 or v4.0

1.2.1. Install the FDx SDK for Windows

The SecuGen Java SDK is a Java Native Interface (JNI) wrapper for the SecuGen FDx SDK. The JNIFPLIB runtime library included in this SDK passes calls to the FPLIB runtime library included on the FDx SDK. Refer to the FDx SDK for Windows Developer's Guide for detailed installation instructions.

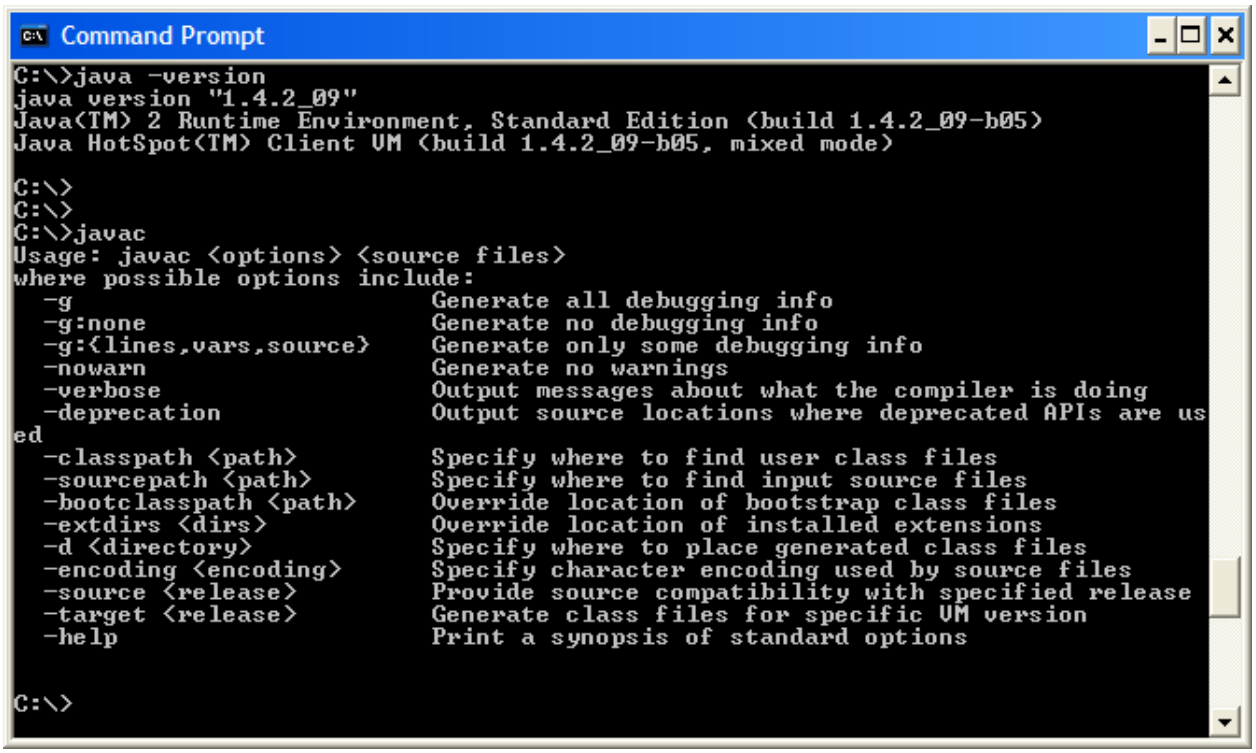
1.2.2. Install the Java 2 SDK v1.4.2_09

The SDK can be downloaded at http://java.sun.com. Refer to the SUN documentation for detailed installation instructions.

After installing the Java SDK, verify that you have installed it correctly by launching a command prompt and running the following commands

- **java – version**
- **javac**

Output should be similar to the following:



1.2.3. Copy the SecuGen FDx SDK for Java Directory to your target location

The SecuGen FDx SDK for Java is distributed as a directory structure containing all required Jar files, the JNI library and various batch files that can be used to compile and run the included sample applications. As long as the Java 2 SDK and SecuGen FDx SDK are correctly installed, the SecuGen FDx SDK for Java can be installed in any convenient location.

1.3. Setting up FDx SDK for Java

Copy the SDK distribution into a new directory on the development machine.

1.3.1. Java SDK Directory Structure

The following files and directories are contained in the Java SDK:

- **applet/** - Directory containing JSGDApplet sample applet
- **doc/** - Directory containing JavaDoc, distribution guide and programmer's guide for the FDx SDK for Java
- **AbsoluteLayout.jar** – executable jar file containing SWING AbsoluteLayout class
- **buildsigndapplet.bat** – Batch file to build signed applet jar file
- **build_samples.bat** – Batch file to build sample applications after extracting them from FDxSDK.jar
- **exportcert.bat** – Batch file to export the certification from the certificate store
- **extract_samples.bat** – Batch file to extract the sample application source code from FDxSDK.jar
- **FDxSDK.jar** – The SecuGen FDx SDK for Java API Jar file

- **genkey.bat** – Batch file to generate private and public keys for the signed applet jar file
- **jnifplib.dll** – SecuGen JNI wrapper for FPLIB fingerprint library
- **JSGDAppletDemo.html** – html to load the JSTDApplet demonstration
- **readme.txt** – latest release information
- **run_JFPLibTest.bat** – Batch file to run JFPLIBTest, a console test utility
- **run_JSGD.bat** – Batch file to run JSGD, a SWING SecuGen Diagnostic Utility.

1.4. Hardware Specifications

1.4.1. About the SecuGen Fingerprint Sensor

The SecuGen fingerprint-capturing consists of an OPP03 fingerprint sensor module embedded in a USB or parallel PC peripheral device. Peripherals may be in the form of a stand-alone device or a mouse The FDU02 module consists of an optical unit and a USB interface board with power supplied through the USB connection. The parallel module is powered through a PS/2 connector.

Optical Module

The optical module consists of a light source (LED), optical assembly (prism and lenses), and a CMOS image sensor. The image sensor captures three-dimensional fingerprints at 500 dpi resolution and sends data to the host computer via the USB port. The image sensor can be controlled by commands provided in the SDK.

1.5. Hardware Testing

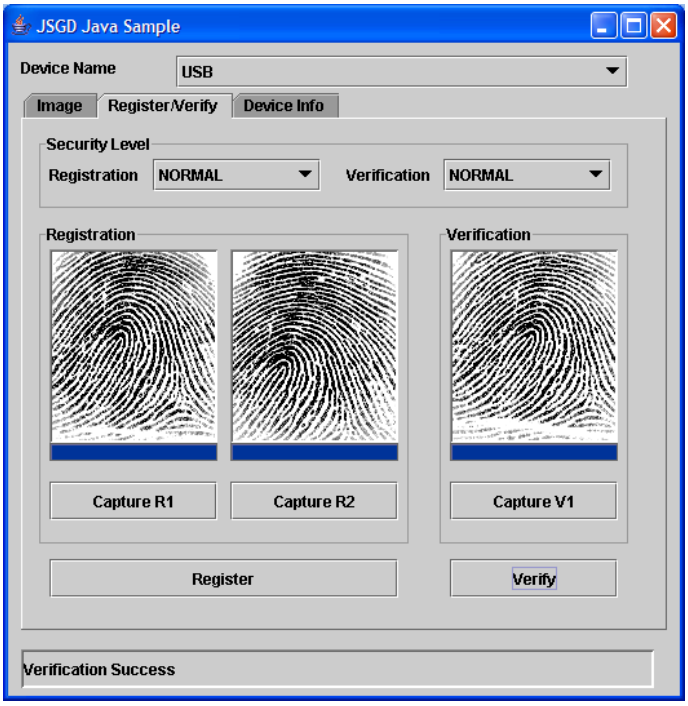
After installing the hardware and software, it is recommended that all components be checked to verify that they are working properly.

1.5.1. JSGD - Hardware Test Program

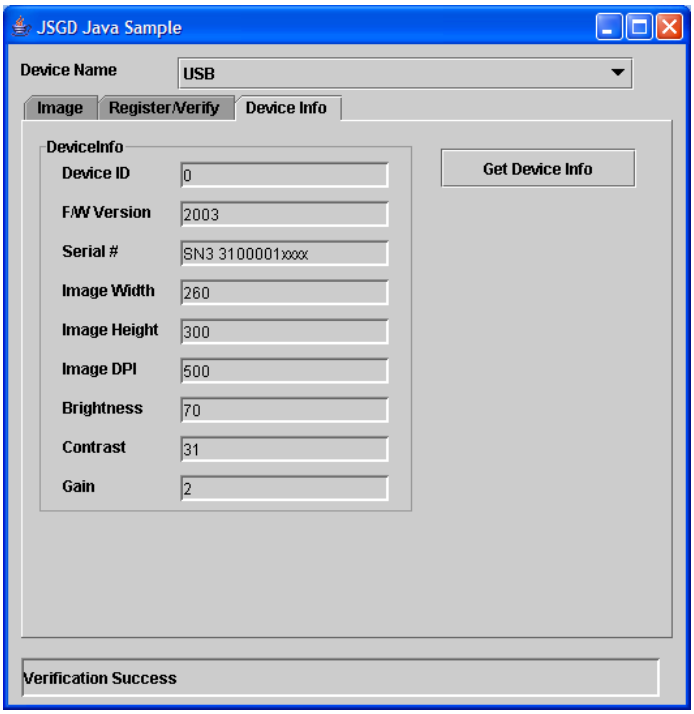
The **SecuGen Diagnostic Hardware Test** program (JSGD.class) is located in the `FDxSDK.jar` archive. This program scans fingerprint images and also performs fingerprint registration and verification. If this program fails to capture a fingerprint image, the system is not configured correctly.

1. Launch a command prompt.
2. cd `<FDx_SDK_FOR_JAVA_INSTALL_DIR>`
3. Type ***run_JSGD.bat*** and then **Enter.**
4. Click **Initialize** to initialize the device. The result of initialization (success or failure) will be displayed in the status bar at the bottom left of the screen. If initialization fails, check the device connection and repeat the above steps.
5. If initialization is successful, place your thumb or finger in position on the fingerprint reader, and click **Capture**. The fingerprint image should be displayed if your device is working properly.

6. To register and verify a finger, click the **Register/Verify** tab and perform the following steps:
   - Place a finger on the device and click **Capture R1**
   - Remove and place the same finger on the device and click **Capture R2**
   - Click the **Register** button to view the registration score
   - Place the same finger on the device and click **Capture V1**
   - Click Verify and the message **Verification Success** will be displayed in the status bar if the fingers match. An error will be displayed in the case of a non-match.



7. To view information about the current device, click the **Device Info** tab:



1.5.2. JFPLib Test Program

The JFPLib Test program demonstrates all of the functionality included in the SecuGen FDx SDK for Java.

1. Launch a command prompt.

2. cd <FDx_SDK_FOR_JAVA_INSTALL_DIR>

3. Type ***run_jfplibtest.bat*** and then **Enter.**

4. Follow the prompts on the screen and place your finger on the sensor when prompted. A sample session is shown below.

```
D:\development\jni\release>
D:\development\jni\release>run_jfplibtest

D:\development\jni\release>del *.min
Could Not Find D:\development\jni\release\*.min

D:\development\jni\release>del *.raw
Could Not Find D:\development\jni\release\*.raw

D:\development\jni\release>java -cp ".;FDxSDK.jar" SecuGen.FDxSDK.samples.JFPLibTest

################################
SecuGen FDx SDK for Java
JFPLib JNI Library Test Program
################################
```

```
Instantiate JFPLib Object
SecuGen JFPLib version: 1.0.0.0
Call jfplib.openDevice(FDxPortAddr.AUTO_DETECT)
openDevice returned : [0]
Press <Enter> to turn fingerprint sensor LEDs on >>
Call jfplib.setLedOn(true)
setLedOn returned : [0]
Fingerprint Sensor LEDS should now be illuminated.
Press <Enter> to turn fingerprint sensor LEDs off >>
Call jfplib.setLedOn(false)
setLedOn returned : [0]
Place a finger on the fingerprint sensor. Press <Enter> to tune device >>
Call jfplib.autoTuning()
Processing ... Please hold finger on sensor until complete.
Done ...
autoTuning returned : [0]
Call jfplib.configure()
configure returned : [0]
Call jfplib.resetMatchingEngine()
resetMatchingEngine returned : [0]
Call jfplib.getDeviceInfo()
GetDeviceInfo returned : [0]
        deviceInfo.DeviceSN:    [[B@1ea2dfe]
        deviceInfo.Brightness:  [20]
        deviceInfo.ComPort:     [0]
        deviceInfo.ComSpeed:    [0]
        deviceInfo.Contrast:    [31]
        deviceInfo.DeviceID:    [0]
        deviceInfo.FWVersion:   [8195]
        deviceInfo.Gain:        [2]
        deviceInfo.ImageDPI:    [500]
        deviceInfo.ImageHeight: [300]
        deviceInfo.ImageWidth:  [260]
Fingerprint Sensor LEDS should now be off.
The next tests will require mutiple captures of the same finger.
Which finger would you like to test with? (e.g. left thumb) >> right index
Call jfplib.setLedOn(true)
setLedOn returned : [0]
Capture 1. Please place [right index] on sensor with LEDs on and press <ENTER>
Call jfplib.getImage()
GetImage returned : [0]
Call jfplib.getError()
getError returned : [0]
Call jfplib.setBrightness()
setBrightness returned : [0]
Call jfplib.setContrast() - 31
setContrast returned : [0]
Call jfplib.setGain()
setGain returned : [0]
Call jfplib.resetExtractionModule()
resetExtractionModule returned : [0]
Call jfplib.getMinutiae()
getMinutiae returned : [0]
Call jfplib.setLedOn(true)
```

```
setLedOn returned : [0]
Capture 2. Remove and replace [right index] on sensor with LEDs on and press <EN
TER>
Call jfplib.getImage()
GetImage returned : [0]
Call jfplib.getMinutiae()
getMinutiae returned : [0]
Call jfplib.setLedOn(true)
setLedOn returned : [0]
Capture 3. Remove and replace [right index] on sensor with LEDs on and press <EN
TER>
Call jfplib.getImage()
GetImage returned : [0]
Call jfplib.getMinutiae()
getMinutiae returned : [0]
Call jfplib.matchForRegister()
matchForRegister returned : [103]
<<NO MATCH>>)
Call jfplib.matchForVerification()
matchForVerification returned : [0]
<<MATCH>>)
Call jfplib.matchForVerificationEx()
matchForVerificationEx returned : [0]
<<MATCH>>)
Call jfplib.closeDevice()
CloseDevice returned : [0]
Call jfplib.close()
close returned : [0]

D:\development\jni\release>
```

Chapter 2. Files Included

2.1. FDx SDK for Java

Library files

FDxSDK.jar - SecuGen Java SDK
jnifplib.dll - SecuGen JNI library
Absolutelayout.jar - NetBeans 4.x Swing layout runtime

Sample Program files

extract_samples.bat - extracts sample source code
build_samples.bat - builds sample applications
run_JSGD.bat - runs the JSGD sample application
run_JFPLibTest.bat - runs the JFPLibTest sample application

Sample Signed Applet files

applet/ - Directory containing signed applet demo
genkey.bat - generate certificate & key pair
buildsignedapplet.bat - build signed jar file
exportcert.bat - export certificate
JSGDAppletDemo.html - html to load signed applet

Programmer's Guide

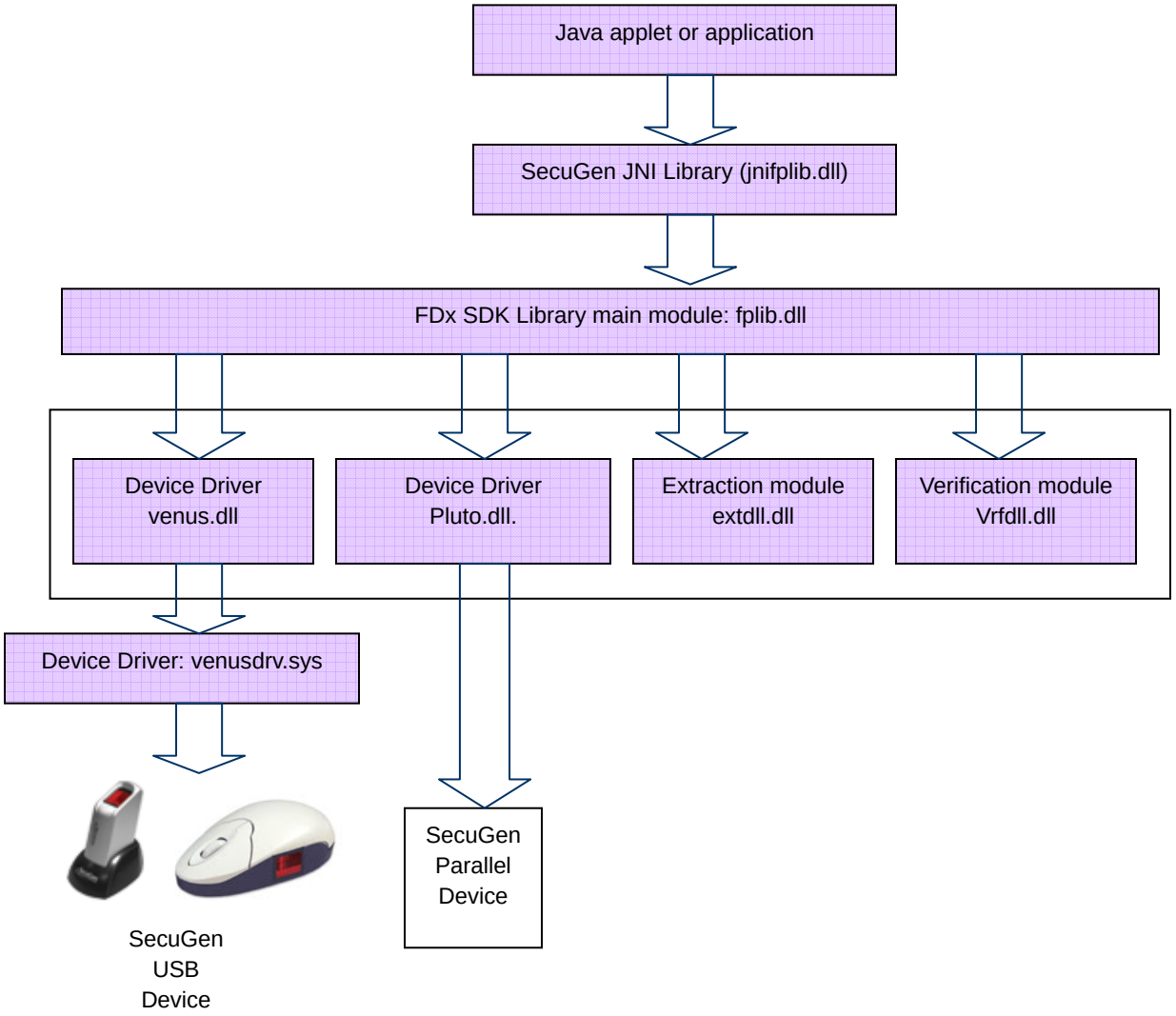
readme.txt - latest release information for the FDx SDK for Java.
doc/ - Directory containing JavaDoc for SecuGen FDxSDK for Java
FDx SDK Manual (Java).pdf – This document.
FDx SDK Distribution Guide (Java).pdf – Description of files to distribute with finished applications.

Chapter 3. Programming with the SDK

Programming with SecuGen's FDU SDK for Java is easy, thanks to the SecuGen APIs that are provided. The inclusion of device facility, extraction, and verification algorithms in the SecuGen SDK enables programmers to build fingerprint recognition applications quickly.

All SDK functions are integrated into the JFPLib class, and are therefore under its direct control. This includes device initialization, fingerprint capture, minutiae extraction, and verification.

3.1. SecuGen FDU SDK for Java Structure



Chapter 4. Programming with JFPLib class

4.1. Importing the FDxSDK Package

In order to use the SecuGen ***JNIFPLib*** library, you must import ***SecuGen.FDxSDK.jni*** package as shown below.

```
import SecuGen.FDxSDK.jni.*;
```

4.2. Instantiating the JFPLib Object

First, instantiate a JFPLib object. This function has one parameter for the device code name. For SecuGen FDU02 devices, the device code name is ***FdxDeviceName.CN_FDU02***. Refer to the ***FDxDeviceName*** class JavaDoc reference for a complete list of device names.

```
JFPLib fplib = new JFPLib(FDxDeviceName.CN_FDU02);
```

4.3. Destroying FPM Object

When exiting a program, you must close the JFPLib object with **closeDevice()** to unload the native library and free resources.

```
err = fplib.closeDevice();
```

4.4. Opening the Fingerprint Device

Call the **openDevice()** method to open the fingerprint sensor. Use ***FDxPortAddr.AUTO_DETECT*** to auto detect the device. To open a specific device, refer to the ***FDxPortAddr*** class JavaDoc reference for a complete list of device names.

```
err = fplib.openDevice(FDxPortAddr.AUTO_DETECT);
```

4.5. Getting Device Information

After successful initialization, you can retrieve device information by calling the **getDeviceInfo()** method. It is called for obtaining required information such as image height and width.

```
FDxDeviceInfoParam deviceInfo = new FDxDeviceInfoParam();
err = fplib.getDeviceInfo(deviceInfo);
System.out.println("GetDeviceInfo returned : [" + err + "]");
System.out.println("\tdeviceInfo.DeviceSN:    [" + deviceInfo.deviceSN() + "]");
System.out.println("\tdeviceInfo.Brightness:  [" + deviceInfo.brightness + "]");
System.out.println("\tdeviceInfo.ComPort:     [" + deviceInfo.comPort + "]");
System.out.println("\tdeviceInfo.ComSpeed:    [" + deviceInfo.comSpeed + "]");
System.out.println("\tdeviceInfo.Contrast:    [" + deviceInfo.contrast + "]");
System.out.println("\tdeviceInfo.DeviceID:    [" + deviceInfo.deviceID + "]");
System.out.println("\tdeviceInfo.FWVersion:   [" + deviceInfo.FWVersion + "]");
System.out.println("\tdeviceInfo.Gain:        [" + deviceInfo.gain + "]");
```

```
System.out.println("\tdeviceInfo.ImageDPI:    [" + deviceInfo.imageDPI + "]");
System.out.println("\tdeviceInfo.ImageHeight: [" + deviceInfo.imageHeight + "]");
System.out.println("\tdeviceInfo.ImageWidth:  [" + deviceInfo.imageWidth + "]");
```

4.6. Image Capturing Operation

After successful initialization, fingerprint images from the device can be captured using the **getImage()** method. The captured fingerprint is a 256 gray-level image. (Image width and height can be retrieved by the **getDeviceInfo()** API.)

```
imageBuffer1 = new byte[deviceInfo.imageHeight*deviceInfo.imageWidth];
byte[] imageBuffer1;
err = fplib.getImage(imageBuffer1);
```

4.7. Exposure Control

Call the **setBrightness()** method to set the exposure value (the default value is 70).

```
err = fplib.setBrightness(deviceInfo.brightness);
```

4.8. Registration

Call the **getMinutiae()** method to extract minutiae data from fingerprint image data. The extracted minutiae data can be saved to a file, database, or buffer for further verification processing. Minutiae data are encrypted for high security. The size of encrypted minutiae data is 400 bytes.

```
err = fplib.getMinutiae(imageBuffer1, minutiaeBuffer1);
```

For the highest degree of accuracy, the registration process requires that two fingerprint images be captured from the same finger. The minutiae data from both packets must be extracted before calling the **matchForRegister()** method to confirm the fingerprints. This confirmation routine is analogous to the password confirmation routine required when entering a new password.

```
err = fplib.matchForRegister(
                        minutiaeBuffer1,
                        minutiaeBuffer2,
                        FDxSecurityLevel.L5_NORMAL);
```

This function has two parameters: two minutiae buffers and a security level. The two minutiae data packets are created by **getImage()** and **getMinutiae(),** and the security level may be adjusted according to the security policy set by users.

4.9. Verification

Call the **matchForVerificationEx()** method to match new packets of minutiae data to the two packets of registered minutiae data. This function has four parameters: two registered minutiae data packets, the minutiae packet to be matched, and a security level.

```
err = fplib.matchForVerificationEx(
                                minutiaeBuffer1,
```

```
                                        minutiaeBuffer2,
                                        minutiaeBuffer3,
                                        FDxSecurityLevel.L5_NORMAL);
```

4.10. Getting Matching Score

For improved quality control during the registration or verification process, you may prefer to use the matching score rather than security level to determine the success of the operation. This will allow you to accept or reject minutiae data that are registered or verified by setting a minimum score threshold instead of a security level. The matching score value is from 0 to 199. The **getMatchingScore()** method requires two minutiae data.

```
byte[] score = new byte[1];
err = fplib.getMatchingScore (minutiaeBuffer1, minutiaeBuffer2, score);
```

Security Level vs. Corresponding Matching Score

Constant	Value	Corresponding Matching Score	Description
LOWEST	1	<=10	
LOWER	2	< = 15	
LOW	3	<= 20	
BELOW_NORMAL	4	<= 25	
NORMAL	5	<= 30	
ABOVE_NORMAL	6	<= 35	
HIGH	7	<= 40	
HIGHER	8	<= 50	
HIGHEST	9	<= 60	

Chapter 5. Building Signed Applets

5.1. Overview

In order to use the Java SDK in a web browser it is necessary to sign the applet Jar file.

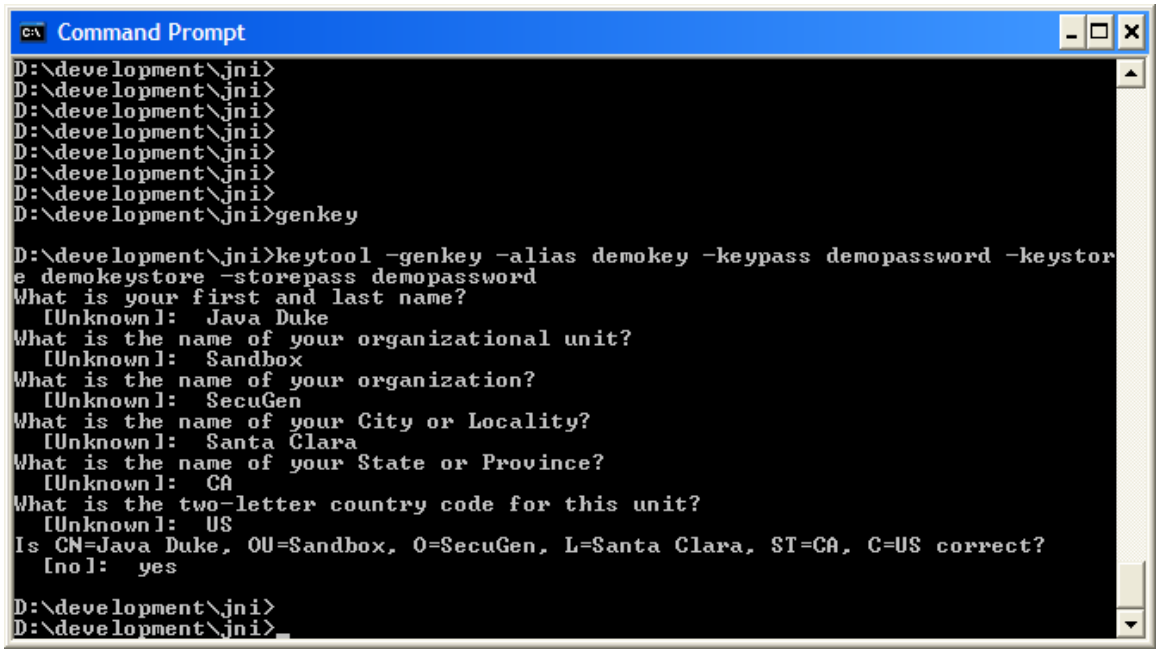
- The applet must be signed
- The JRE 1.4.2_09 must be installed
- The Java Plug-In 1.4.2_09 must be installed.

Java security prevents an unsigned applet that accesses system resources such as file systems or fingerprint sensors from executing unless the applet is signed. This chapter describes the process of signing the JSGDApplet demo applet and deploying it in Internet Explorer, FireFox and Netscape browsers.

5.2. Sign the Applet

1. Generate a digital certificate and key pair (GENKEY.BAT)

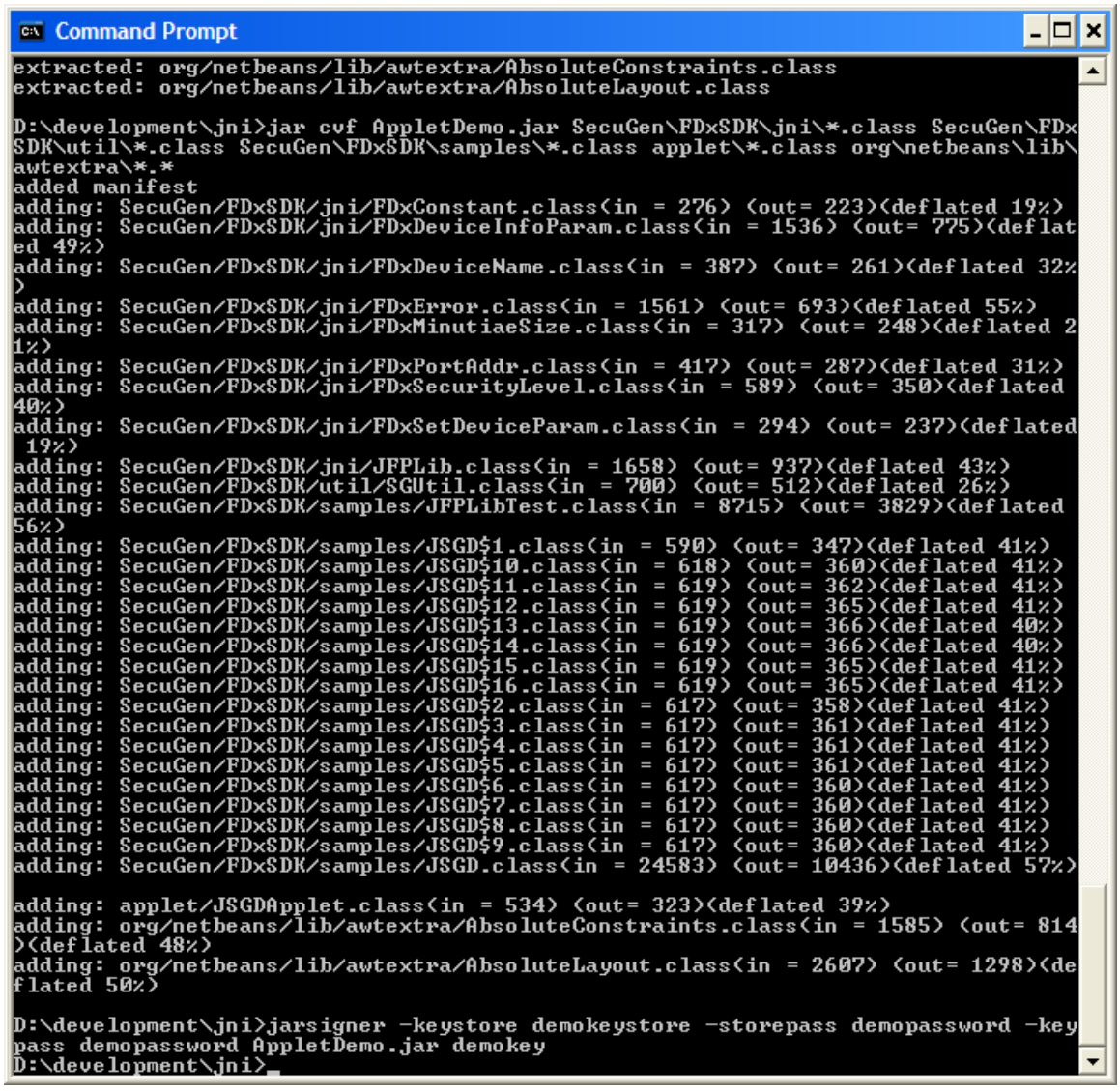
```
keytool -genkey -alias demokey -keypass demopassword -keystore demokeystore -
storepass demopassword
```



2. Build a signed jar file (BUILDSIGNEDAPPLET.BAT)

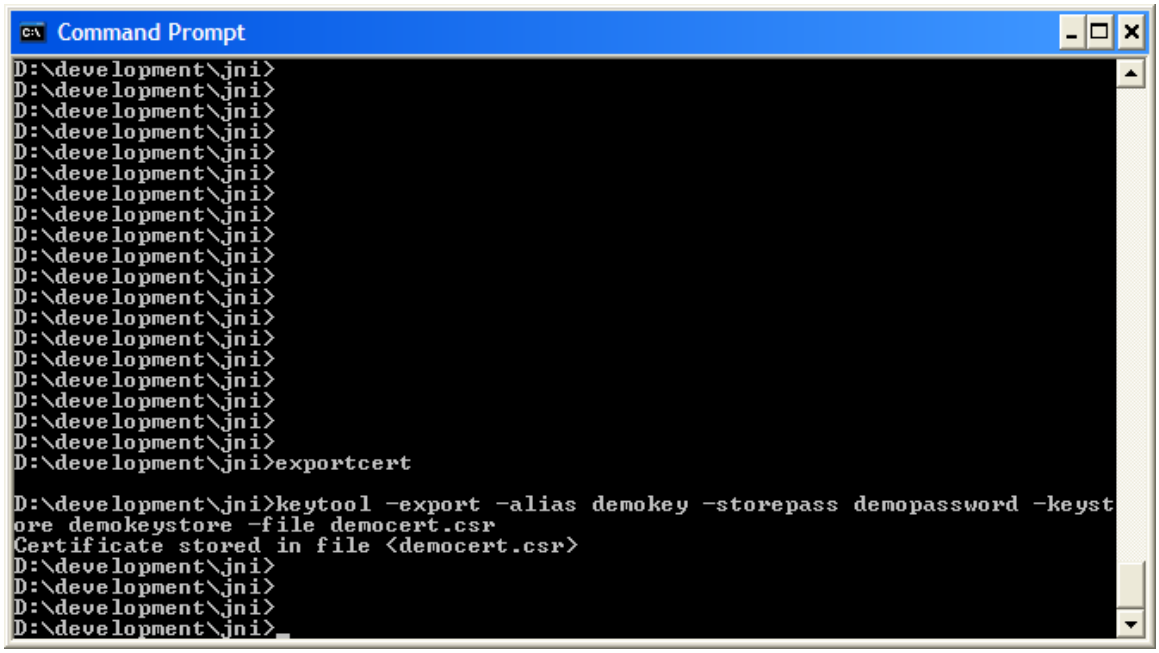
```
javac -deprecation applet\*.java
jar xvf FDxSDK.jar
jar xvf AbsoluteLayout.jar
```

```
jar cvf AppletDemo.jar SecuGen\FDxSDK\jni\*.class SecuGen\FDxSDK\util\*.class
SecuGen\FDxSDK\samples\*.class applet\*.class org\netbeans\lib\awtextra\*.*
jarsigner -keystore demokeystore -storepass demopassword -keypass demopassword
AppletDemo.jar demokey
```

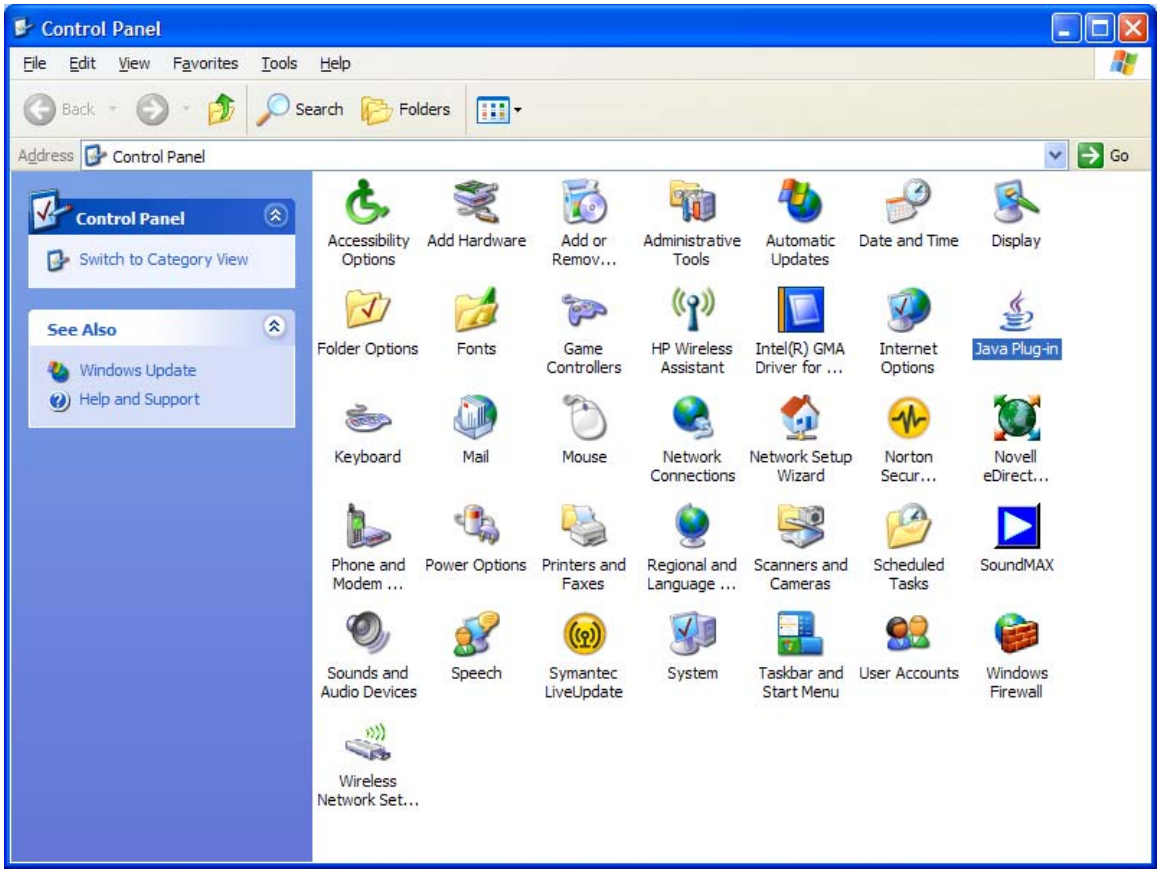


4. Export the certificate (EXPORTCERT.BAT)

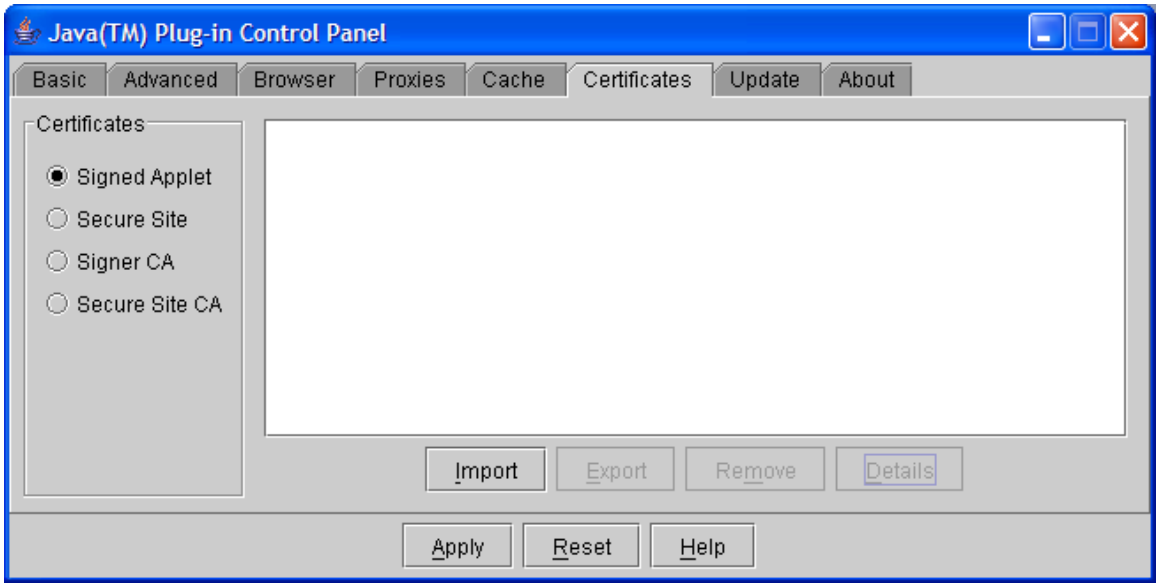
```
keytool -export -alias demokey -storepass demopassword -keystore demokeystore
-file democert.csr
```



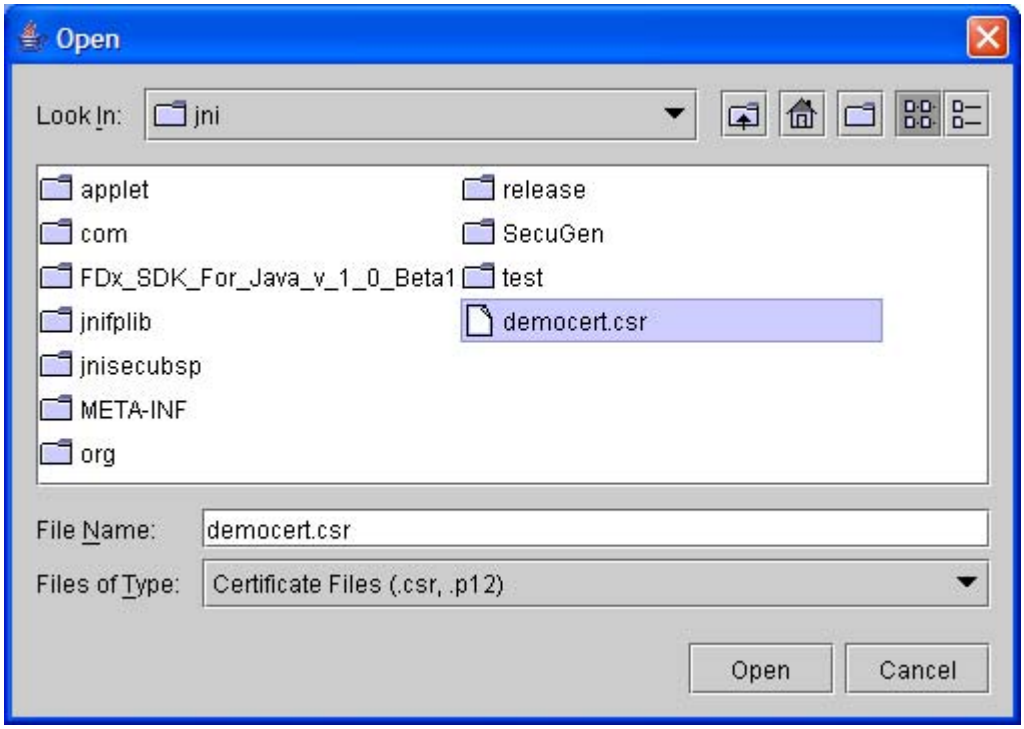
5. Import the certificate into the Java Plug-In. Launch the control panel and select the Java Plug-in.



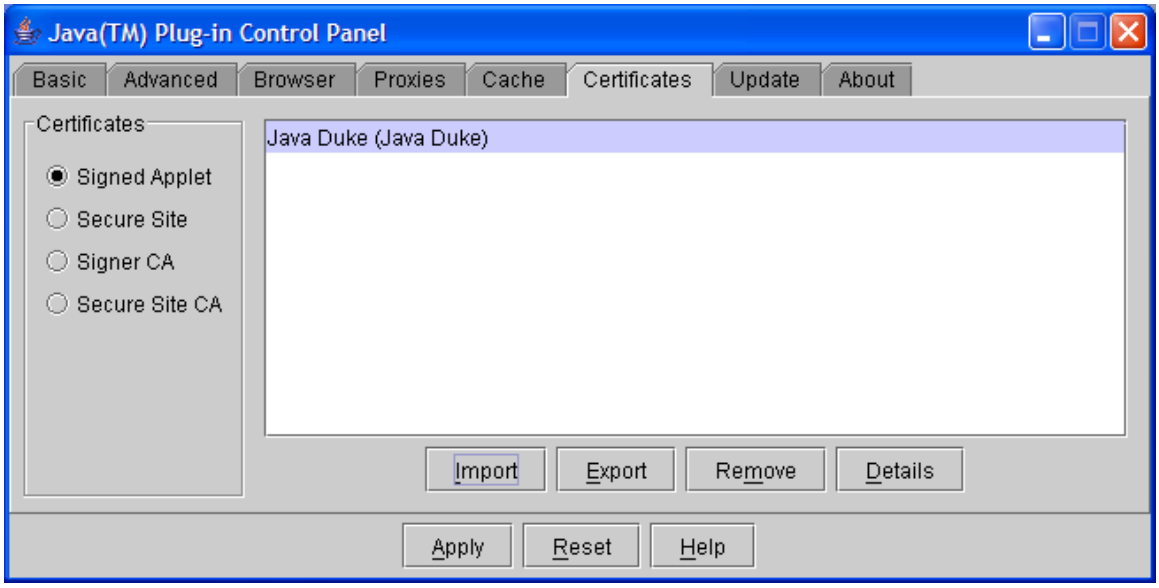
6. Select the ***Certificates*** tab and click the ***Signed Applet*** radio button. Then click ***Import***.



7. Browse to the location of the certificate exported previously. Select the CERT and click ***Open***



8. Verify the details of the CERT after the import is complete and then click ***Apply***.



5.3. HTML File to Load the Signed Applet

The JSGDAppletDemo.html file contains the following HTML code which is used to load the signed applet.

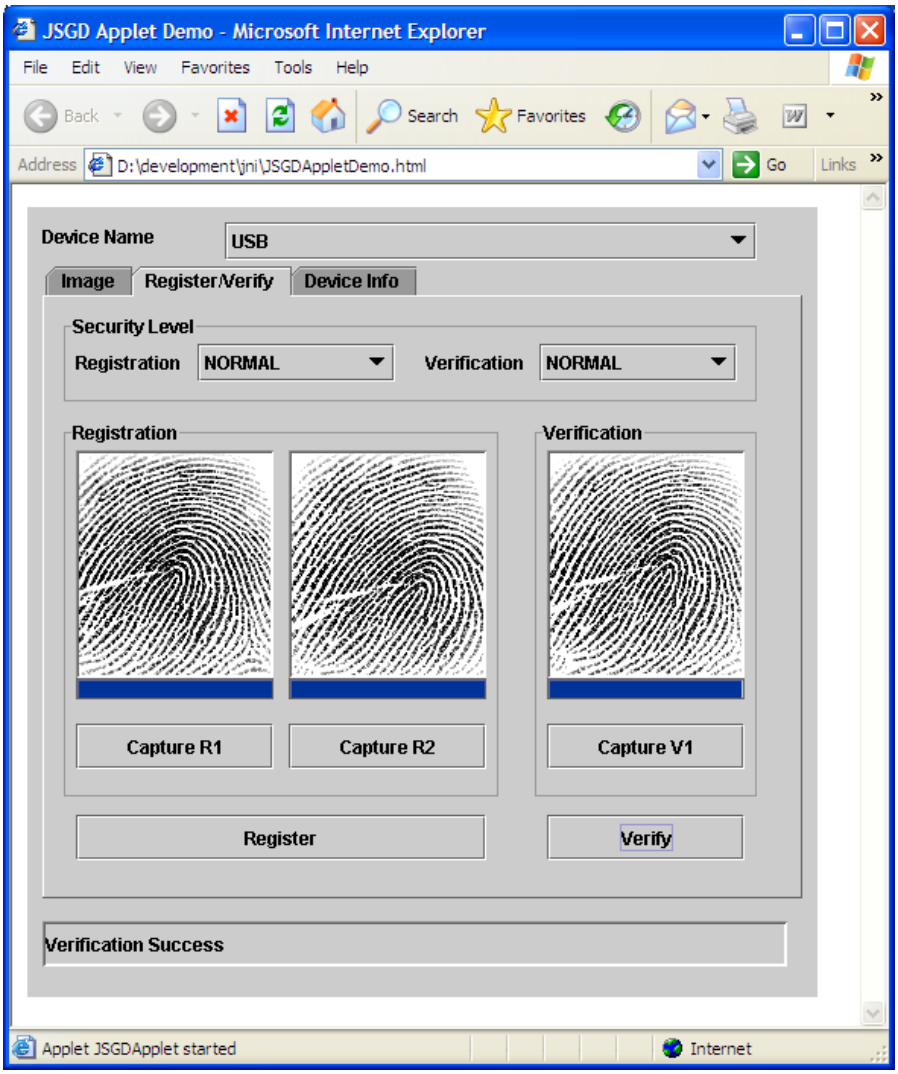
```
<html>
<head>
<title>
JSGD Applet Demo
</title>
</head>
<body>

<applet
  code     = "applet.JSGDApplet.class"
  name     = "JSGDApplet"
  archive  = "AppletDemo.jar"
  width    = "500"
  height   = "300"
  hspace   = "0"
  vspace   = "0"
  align    = "middle"
>
</applet>
</body>
</html>
```

5.4. Internet Explorer 6.0

1. Copy JNIFPLIB.DLL to C:\WINDOWS\SYSTEM32

2. Open JSGDAppletDemo.html

3. The applet should load in the browser as shown below.

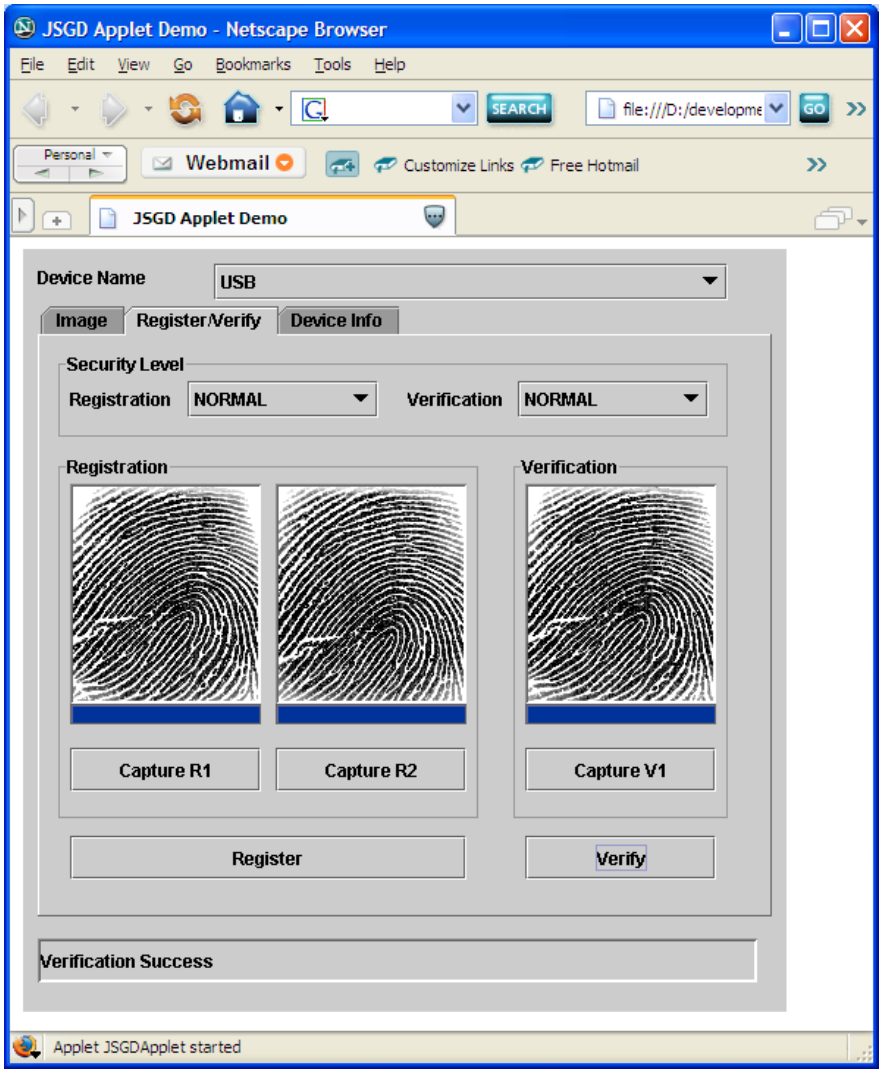


5.5. Netscape Browser v8.0

1. Copy JNIFPLIB.DLL to C:\Program Files\Netscape\Netscape Browser\

2. Open JSGDAppletDemo.html

3. The applet should load in the browser as shown below.



If the JNIFPLIB is not installed in the correct location, the following exception will be thrown:

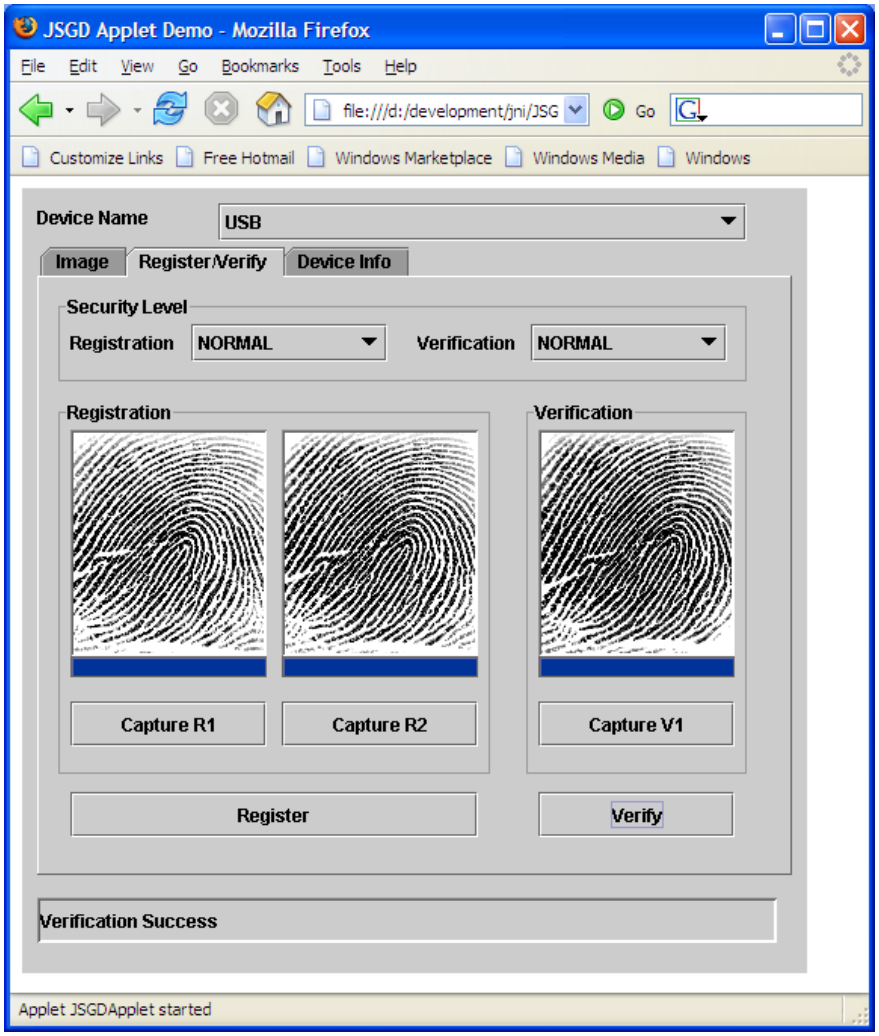
```
java.lang.UnsatisfiedLinkError: C:\Program Files\Netscape\Netscape Browser\jnifplib.dll: Can't
find dependent libraries
```

5.6. Mozilla FireFox Browser v1.0.6

1. Copy JNIFPLIB.DLL to C:\Program Files\Mozilla Firefox\

2. Open JSGDAppletDemo.html

3. The applet should load in the browser as shown below.



If the JNIFPLIB is not installed in the correct location, the following exception will be thrown:

```
java.lang.UnsatisfiedLinkError: C:\Program Files\Mozilla Firefox\jnifplib.dll: Can't find
dependent libraries
```

Chapter 6. JFPLib Class Reference

6.1. JavaDoc

API Documentation for the SecuGen FDx SDK for Java is provided in the standard JavaDoc format. The Javadoc can be found in the ***<doc>*** directory in the FDx SDK for Java folder. All functions included in this SDK are implemented using the JFPLib class. Using the JFPLib class allows direct control of functions such as device initialization, fingerprint capture, minutiae extraction, and verification. Documentation for supporting classes and error codes is also included in the JavaDoc.