

OpenCard Framework 1.2 Developer's Guide

SDK/J Authentication Package Version:1.0



Copyright © 2008-2010 Ricoh Co., Ltd.

Terms of Use and Trademarks

1. The contents of this book may be changed without notice in the future.
2. The copying, reproducing, changing, quoting, reprinting, or distributing a part/all of this book are prohibited.
3. We make no warranty, express or implied, regarding this document and the sample codes described in this document. We will not be held responsible for any of our customer's losses, damages resulting from lost profits, or claims from any third party on using this document and the sample codes described in this document.
4. Trademarks

Ethernet® is a registered trademark of Xerox Corporation.

PostScript® and Acrobat® are registered trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Microsoft® and Windows® are registered trademarks of Microsoft Corporation in the United States and other countries.

UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Ltd.

Red Hat is a registered trademark of Red Hat, Inc. in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States and other countries, or both.

Java, JVM(CVM) and CDC are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Eclipse is a trademark of eclipse.org in the United States, other countries, or both.

OSGi(TM) is a trademark, registered trademark, or service mark of The Open Services Gateway Initiative in the US and other countries.

Apache is a registered trademark of The Apache Software Foundation in the United States, other countries, or both.

Other product names used herein are for identification purposes only and might be trademarks of their respective companies. We disclaim any and all rights in those marks.

Contents

1. Introduction	2
1.1. Target Readers.....	2
1.2. Software Requirements	2
1.3. Restrictions	2
2. About the OCF 1.2	3
3. Optimization of the OCF 1.2 for the SDK/J.....	4
3.1. opencard.core.service.DefaultCHVDialog class.....	4
3.2. com.ibm.opencard.terminal.pcsc10.Pcsc10CardTerminal class.....	4
3.3. com.ibm.opencard.terminal.pcsc10.OCFPCSC1 class.....	4
4. Application Installation	5
4.1. DALP file settings	5
4.2. Installation	5
5. Appendix	6
5.1. How to check the operation of a SmartCard Framework application by using an emulator	6
Change History	8

1. Introduction

This document contains important information on developing a Device SDK Type-J (hereinafter referred to as “SDK/J”) application by using the OpenCard Framework (hereinafter referred to as “the OCF”).

1.1. Target Readers

This document is intended for the application developers who are experienced in SDK/J application development and are familiar with the basics of SmartCard.

1.2. Software Requirements

Use of the OCF requires an environment where:

- PC/SC daemon is enabled.

* About the setting of PC/SC daemon, see “SDK/J Authentication Package Settings Guide”.

1.3. Restrictions

Restrictions on using the OCF:

- The operation will not be guaranteed when the OCF and the SmartCard Framework v1.0 are used together on a same execution environment at the same time.
- The operation will not be guaranteed when more than one card reader device or other USB device are connected to the USB host at the same time.

2. About the OCF 1.2

OCF 1.2 provides Java APIs for working with smart cards. OCF 1.2 for SDK/J is based on OCF 1.2 which had been provided by the OpenCard Consortium.

3. Optimization of the OCF 1.2 for the SDK/J

The SDK/J uses the OCF 1.2 and the pcsc-wrapper-2.0, a JNI to access PC/SC from OCF, that have been optimized for the SDK/J.

As a result of the optimization, the source code of the OCF 1.2 and the pcsc-wrapper-2.0 have been partially modified.

This chapter describes how they were changed and what should be noted regarding the changes.

3.1. `opencard.core.service.DefaultCHVDialog` class

In the standard OCF 1.2, this class is implemented by the `java.awt` package. However, since the SDK/J does not support the package, this class is not implemented in the optimized OCF 1.2. Accordingly, the `getCHV` method of this class will always return null. This method is called when null is specified to the argument of a method that takes `CHVDialog` as its argument. So regarding such a method, a method that takes `CHVDialog` as its argument, be sure to specify an instance of an original class that inherits `CHVDialog` to the argument.

3.2. `com.ibm.opencard.terminal.pcsc10.Pcsc10CardTerminal` class

In the standard `pcsc-wrapper-2.0`, this class uses the exclusive mode when it accesses a card by using the PC/SC. However, this class of the optimized `pcsc-wrapper-2.0` uses the nonexclusive mode so that multiple applications can access a card.

Note that the exclusive mode can not be specified for this class of the optimized `pcsc-wrapper-2.0`.

3.3. `com.ibm.opencard.terminal.pcsc10.OCFPCSC1` class

In the standard `pcsc-wrapper-2.0`, this class downloads the `OCFPCSC1` library and accesses to the PC/SC by using the JNI. However, the name of the library to be downloaded is different in the optimized `pcsc-wrapper-2.0`. This is because the name of the library has been changed in the SDK/J.

Please use the original `pcsc-wrapper-2.0` when running an OCF application on the SDK/J emulator.

4. Application Installation

This chapter describes how to install an SDK/J application that employs the OCF 1.2.

4.1. DALP file settings

About DALP format, see the SDK/J Developer's Guide.

Since the OCF jar files (base-core.jar, base-opt.jar) are deployed in SDK/J SD card, jar file description of the OCF jar files is unnecessary.

The following is a configuration example of the DALP file of an application that uses the OCF.

```

    :
    :
    <resources>
      <dsdk version = "2.0"/>
      <jar href="./123456789.jar" basepath="current" main="true"/>
      <option-jar href="./cardservice.jar" basepath="current" main="false"/> *
      <encode-file>123456789</encode-file>
    </resources>
    :
    :

```

Note: Use an option-jar tag for the installation of the jar file of the CardService interfaces if they are referred to by other application(s). If they are referred to by only one application (the installed application itself), it is not needed to use option-jar tag.

4.2. Installation

The installation procedure is the same as a normal SDK/J application.

For details, see the User's Guide of the SDK/J.

Note: When the option-jar tag is used in the DALP file, it is needed to reboot the target device after the installation.

5. Appendix

5.1. How to check the operation of a OpenCard Framework application by using an emulator

<< Note >>

This chapter is described assuming that the Embedded Software Architecture Emulator 4.13d is used for the operation check. Therefore, if you use other emulators, the operation check may not be performed properly.

Go through the following steps, and you can check the operation of an OCF application by using the Embedded Software Architecture Emulator 4.13d.

STEP-1 Establishing the necessary environment for using the OCF

First, establish the necessary operation environment for the operation check.

1. Installing a Card Reader Driver

Install a card reader driver if there is no such driver available.

Be sure to install a card reader driver supporting PC/SC.

2. Locating the OCFPCSC1.dll to “cdc-dsdk4” of the emulator

Locate the OCFPCSC1.dll, which is the JNI necessary to use PC/SC in the OCF, to

<emulator_installpath>\cdc-dsdk4 directory. The OCFPCSC1.dll can be downloaded from MUSCLE project's web site.

MUSCLE project : <http://www.linuxnet.com/>

3. Adding the OCF jar files to the classpath of the emulator

Add the pcsc-wrapper-2.0.jar, which is the API necessary to use PC/SC in the OCF, and the jar file of the OCF to the classpath of the emulator. Edit -classpath option of the activation batch file of the emulator as follows. The pcsc-wrapper-2.0.jar can be downloaded from the following site.

Gemalto site : <http://www.gemalto.com/techno/opencard/cardterminals/pcsc/download.html>

```

:
:
SET MYCLASSPATH=-classpath
.\resource\sdk\emulator\common\emulatorCommon.jar;.\mnt\sd2\sdk\common\jars\dsdk\dsd
kCommon.jar;.\lib\jh.jar;.\lib\kxml2-2.3.0.jar;.\mnt\sd2\sdk\common\framework.jar;.\
mnt\sd2\sdk\common\jars\smartcard\base-core.jar;.\mnt\sd2\sdk\common\jars\smartcard\
base-opt.jar;.\pcsc-wrapper-2.0.jar
:
:

```

STEP-2 Adding the jar file that is installed by the option-jar to the classpath of the emulator

Add the jar file that is specified by the option-jar to the classpath of the emulator in the same way as in STEP-1.3.

Note: The option-jar tag of the DALP file is not supported by the emulator.

STEP-3 Registering CardService services

When the application does not dynamically register CardService services, they need to be described in the opencard.properties file.

```
⋮  
OpenCard.services = \  
    my.package.MyCardServiceFactory  
⋮
```

The opencard.properties file should be placed under <emulator_installpath>\cdc-dsdk4\cdc-toolkit\CDCTK10\lib directory.

STEP-4 Installing the application

Start the emulator by using <emulator_installpath>\startemulator-jvm.bat.

Install the application in the same way that a normal SDK/J application is installed.

Change History

Ver. 1.0	SDK/J v4 or later version based on SDK/J AP v1.0 option package for SDK/J v2. Update software requirements base-core Implementation-Version:1.2-1.1 base-opt Implementation-Version:1.2-1.0 pcsc-wrapper Implementation-Version:2.0-1.0
Ver. 1.0.4	Update “5.1. How to check the operation of a OpenCard Framework application by using an emulator” base-core Implementation-Version:1.2-1.1 base-opt Implementation-Version:1.2-1.0 pcsc-wrapper Implementation-Version:2.0-1.0