RMI Client Application Programming Interface

Java Card™ Platform, Version 2.2.2

Sun Microsystems, Inc.
www.sun.com

3-31-06

Contents

CHAPTER 1

Client-Side API for RMI on the Java Card Platform

A Java Card™ remote method invocation (Java Card RMI) client application runs on a Card Acceptance Device (CAD) terminal that supports a Java™ Platform, Standard Edition (Java SE) or Java™ Platform, Micro Edition (Java ME). The client application requires a portable and platform independent mechanism to access the Java Card RMI server applet executing on the smart card.

This client-side architecture currently focuses on a Java SE client running on a JDK™ version 1.2 or higher environment. The client interfaces are designed to be independent of the card access framework. The interfaces therefore only provide Java Card RMI-specific access mechanisms and allow the client application to use its preferred card access mechanisms for its basic card interaction needs.

The example in Sun Microsystem's reference implementation uses the APDU I/O library for card access. See the *Development Kit User's Guide, Java Card Platform, Version 2.2.2* provided with this release.

The Java ME environment is more limited, and since most Java ME profiles do not support the Java Card RMI machinery at this time, it is not discussed in detail. The design of the Java Card RMI client-side architecture ensures that it does not preclude the Java ME client.

Basic Features

The Java Card RMI client side architecture caters to the following:

- Provides the client application mechanisms to initialize and initiate an RMI session with a Java Card applet.
- Provides the client application access to the initial remote reference.

- Allows the client application to customize the message packet during communication with a Java Card technology-compliant smart card to introduce transport level security.
- Allows the CAD terminal developer to customize the card access mechanisms for the specifics of the communications hardware.
- Allows the CAD terminal developer to customize the stub generation mechanism for one that is best suited to the platform capabilities.

Client Application Interfaces

These classes provide the basic Java Card RMI client application with the front-end mechanisms to connect to the Java Card applet and obtain an initial reference to invoke remote methods.

`CardAccessor` Interface

The `CardAccessor` interface is used by the Java Card RMI client framework to communicate with the card. It defines the following methods:

- `public byte[] exchangeAPDU( byte[] command )`

  The Java Card RMI client framework uses this method to send a command APDU to the card and receive the response APDU.

- `public void closeCard()`

  This method closes connection with the card and powers down the card.

In the Java Card RMI example provided in this release (see Chapter 3 of the *Development Kit User's Guide, Java Card Platform, Version 2.2.2*), the client application instantiates the `ApduIOCardAccessor`, which implements the `CardAccessor` interface.

`JCRMIConnect` Class

The client application uses the `JCRMIConnect` class to initiate the Java Card RMI session and obtain the initial remote reference from the card.

Constructor:

```
public JCRMIConnect(CardAccessor cA)
```

The client constructs an instance of the `JCRMIConnect` class, initializing it with a `CardAccessor` object.

The methods of `JCRMIConnect` are:

- `public byte[] selectApplet(byte[] appletAID, byte format)`

  The client starts a Java Card RMI session with the Java Card applet on the card using this method. The second parameter specifies the format of the remote references and can be either of two constants defined in this class: REF_WITH_CLASS_NAME or REF_WITH_INTERFACE_NAMES.

- `public Remote getInitialReference()`

  To begin a Java Card RMI-based dialog, the client obtains the first remote object reference via this method and then casts it to the appropriate remote interface type.

`ApduIOCardAccessor` Class

This class implements a `CardAccessor` object using the ApduIO library. It has a default constructor, which takes initialization parameters from the file `jcclient.properties`, which must be located in one of the directories listed in the CLASSPATH.

Format of the `jcclient.properties` File

The `jcclient.properties` file contains a set of parameters for the `ApduIOCardAccessor`. Each line of the file is in the format `parameter=value`. Comments are preceded with the # symbol. All parameter names and values are case-sensitive.

The parameter `connection` is mandatory, with possible values of `TCP`, `SERIAL`, `PCSC`, for example:

```
connection = TCP
```

If the connection is `TCP` or `SERIAL`, the parameter `protocol` is required, with possible values of `T0` and `T1`, for example:

```
protocol = T1
```

If the connection is `TCP`, the parameters `TCP_HOST` and `TCP_PORT` are required, for example:

```
TCP_HOST = localhost
```

```
TCP_PORT = 9025
```

If the connection is `SERIAL`, the parameter `SERIAL_PORT` is required, for example:

```
SERIAL_PORT = COM1
```

Note: If parameter connection is set to `PCSC`, no other parameters are necessary. The `ApduIOCardAccessor` will attempt to use default sections of the card reader's driver.

Securing the Transport Layer

A simple Java Card RMI client application would only depend on the above methods for a remote method based card session with a Java Card applet. The Java Card applet might need to layer additional security services to protect the message transport data for integrity and privacy. In order to do this, the client application (or security service provider) could develop its own version of the `CardAccessor` class with the `exchangeAPDU` method performing data transformation. The Java Card RMI example provided in this release contains the `CustomCardAccessor` class, which shows how to perform such a task within its `exchangeAPDU` method.

Stub/Proxy Generation

The client does not usually interact with the client side stub generation classes. It is used by the `JCRMIConnect` class to instantiate the initial remote stub object and subsequently the stub objects themselves to instantiate other remote stub objects.

The client library includes the class `JCCardObjectFactory` that encapsulates the mechanism used to instantiate stubs on the client. The `JCCardObjectFactory` class uses the SELECT REF_FORMAT_CLASS format, locates the pre-compiled stub class corresponding to the implementation class name returned in the remote reference descriptor from the card, and then instantiates it.

The client library also includes the class `JCCardProxyFactory`, which uses the SELECT REF_FORMAT_INTERFACES format. The dynamic proxy generation mechanism, defined by JDK version 1.3, generates the proxy instances according to the list of remote interfaces implemented by the remote class.

Exceptions

The remote method invoked on the card may throw an exception to signal that an unexpected condition has been detected. The `RMIService` class on the card catches the exception and returns this information to the client.

If the exception thrown on the card is an exception defined in the Java Card API version 2.2.2, the same exception is re-thrown by the stub object back to the client application. The client can access the reason code associated with Java Card platform-specific exceptions using the standard `getReason()` method.

If the exception thrown on the card is a subclass of an exception defined in the Java Card API version 2.2.2, a client subclass of the closest exception defined in the API (along with the reason code, if applicable) is re-thrown to the client by the stub object. The exception object will contain the following error message string: "A subclass was thrown on the card."

Apart from the exceptions thrown by the remote method itself, errors during communication, marshalling, protocol, unmarshalling, stub generation and so forth related to the Java Card RMI method invocation results in a `RemoteException` exception being thrown to the client application with appropriate error message strings which include the following:

- Applet selection failed, `SW = ...`
- Incorrect (too short) response received from the card
- Invalid format requested by the factory
- Invoke operation failed, `SW = ...`
- Method not found
- Object not exported
- Out of param resources
- Out of response resources
- Protocol error reported by the card
- Signature mismatch
- Thrown on the card
- Unexpected exception received while instantiating the stub
- Unsupported APPLICATION tag
- Unsupported error type received from the card
- Unsupported exception type received from the card
- Unsupported FCI tag
- Unsupported Java Card RMI tag
- Unsupported Java Card RMI version
- Unsupported subclass exception type received from the card
- Unsupported tag

CHAPTER 2

API Documentation

A typical Java Card RMI client application would commonly use the following classes from the client-side API. See the RMI-based demos included with the Java Card platform, version 2.2.2 development kit for examples of usage of these classes.

- `package com.sun.javacard.clientlib`
  - `CardAccessor` - as an interface, which must be implemented by any custom card accessor.
  - `ApduIOCardAccessor` - as a base class or as a source code to implement a custom card accessor.
- `package com.sun.javacard.rmiclientlib`
  - `JCRMIConnect` - to select a card applet and to obtain an initial reference.

The remainder of this whitepaper contains a compilation of the API documentation generated by the Javadoc tool for the client-side Java Card RMI application programming interface packages. It includes the classes and interfaces mentioned above, as well as classes that represent various exceptions that may be thrown on the card and re-thrown on the client. Please note the "FRAMES" feature is not supported in this version.

Packages

All Classes

Packages

com.sun.javacard.clientlib	Provides a framework for building client applications capable of exchanging APDUs with Java Cards.
com.sun.javacard.javax.smartcard.rmiclient	Provides framework of classes and interfaces for a Java Card technology-based client.
com.sun.javacard.rmiclientlib	Provides a framework of classes and interfaces for building Java Card RMI-based client applications.
javacard.framework	Provides Java Card API exceptions that may be re-thrown on the client.
javacard.framework.service	Provides Java Card API exceptions that may be re-thrown on the client.
javacard.security	Provides Java Card API exceptions that may be re-thrown on the client.
javacardx.biometry	Provides Java Card API exceptions that may be re-thrown on the client.
javacardx.external	Provides Java Card API exceptions that may be re-thrown on the client.
javacardx.framework.tlv	Provides Java Card API exceptions that may be re-thrown on the client.
javacardx.framework.util	Provides Java Card API exceptions that may be re-thrown on the client.

All Classes

APDUException
APDUExceptionSubclass
ApduIOCardAccessor
BioException
BioExceptionSubclass
CardAccessor
CardException
CardExceptionSubclass
CardObjectFactory
CardRuntimeException
CardRuntimeExceptionSubclass
CryptoException
CryptoExceptionSubclass
ExternalException
ExternalExceptionSubclass
ISOException
ISOExceptionSubclass
JCCardObjectFactory
JCCardProxyFactory
JCRemoteRefImpl
JCRMIConnect
PINException
PINExceptionSubclass
ServiceException
ServiceExceptionSubclass
SystemException
SystemExceptionSubclass
TLVException
TLVExceptionSubclass
TransactionException
TransactionExceptionSubclass
UserException
UserExceptionSubclass
UtilException
UtilExceptionSubclass

javacard.framework
Class APDUException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.framework.APDUException
```

All Implemented Interfaces:

Direct Known Subclasses:
APDUExceptionSubclass

`public class` **`APDUException`**

extends CardRuntimeException

`APDUException` represents an `APDU` related exception.

See Also:

Field Summary	
`static short`	**BAD_LENGTH** This reason code is used by the `APDU.setOutgoingLength()` method to indicate that the length parameter is greater that 256 or if non BLOCK CHAINED data transfer is requested and `len` is greater than (IFSD-2), where IFSD is the Outgoing Block Size.

static short	**BUFFER_BOUNDS** This reason code is used by the `APDU.sendBytes()` method to indicate that the sum of buffer offset parameter and the byte length parameter exceeds the APDU buffer size.
static short	**ILLEGAL_USE** This APDUException reason code indicates that the method should not be invoked based on the current state of the APDU.
static short	**IO_ERROR** This reason code indicates that an unrecoverable error occurred in the I/O transmission layer.
static short	**NO_T0_GETRESPONSE** This reason code indicates that during T=0 protocol, the CAD did not return a GET RESPONSE command in response to a <61xx> response status to send additional data.
static short	**NO_T0_REISSUE** This reason code indicates that during T=0 protocol, the CAD did not reissue the same APDU command with the corrected length in response to a <6Cxx> response status to request command reissue with the specified length.
static short	**T1_IFD_ABORT** This reason code indicates that during T=1 protocol, the CAD returned an ABORT S-Block command and aborted the data transfer.

Constructor Summary

APDUException(short reason)
Constructs an APDUException.

Method Summary

static void	**throwIt**(short reason) Throws an instance of `APDUException` with the specified reason.

Methods inherited from class javacard.framework.CardRuntimeException

getReason, setReason

Methods inherited from class java.lang.Throwable

`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Field Detail

ILLEGAL_USE

```
public static final short ILLEGAL_USE
```

This APDUException reason code indicates that the method should not be invoked based on the current state of the APDU.

See Also:
Constant Field Values

BUFFER_BOUNDS

```
public static final short BUFFER_BOUNDS
```

This reason code is used by the `APDU.sendBytes()` method to indicate that the sum of buffer offset parameter and the byte length parameter exceeds the APDU buffer size.

See Also:
Constant Field Values

BAD_LENGTH

```
public static final short BAD_LENGTH
```

This reason code is used by the `APDU.setOutgoingLength()` method to indicate that the length parameter is greater that 256 or if non BLOCK CHAINED data transfer is requested and `len` is greater than (IFSD-2), where IFSD is the Outgoing Block Size.

See Also:
Constant Field Values

IO_ERROR

`public static final short` **`IO_ERROR`**

This reason code indicates that an unrecoverable error occurred in the I/O transmission layer.

See Also:
Constant Field Values

NO_T0_GETRESPONSE

`public static final short` **`NO_T0_GETRESPONSE`**

This reason code indicates that during T=0 protocol, the CAD did not return a GET RESPONSE command in response to a <61xx> response status to send additional data. The outgoing transfer has been aborted. No more data or status can be sent to the CAD in this `APDU.process()` method.

See Also:
Constant Field Values

T1_IFD_ABORT

`public static final short` **`T1_IFD_ABORT`**

This reason code indicates that during T=1 protocol, the CAD returned an ABORT S-Block command and aborted the data transfer. The incoming or outgoing transfer has been aborted. No more data can be received from the CAD. No more data or status can be sent to the CAD in this `APDU.process()` method.

See Also:
Constant Field Values

NO_T0_REISSUE

`public static final short` **`NO_T0_REISSUE`**

This reason code indicates that during T=0 protocol, the CAD did not reissue the same APDU command with the corrected length in response to a <6Cxx> response status to request command reissue with the specified length. The outgoing transfer has been aborted. No more data or status can be sent to the CAD in this `APDU.process()` method.

See Also:
Constant Field Values

Constructor Detail

APDUException

`public` **`APDUException`**`(short reason)`

Constructs an APDUException.

Parameters:
`reason` - the reason for the exception.

Method Detail

throwIt

`public static void` **`throwIt`**`(short reason)`

Throws an instance of `APDUException` with the specified reason.

Parameters:

`reason` - the reason for the exception.

Throws:

`APDUException` - always.

Packages

com.sun.javacard.clientlib	Provides a framework for building client applications capable of exchanging APDUs with Java Cards.
com.sun.javacard.javax.smartcard.rmiclient	Provides framework of classes and interfaces for a Java Card technology-based client.
com.sun.javacard.rmiclientlib	Provides a framework of classes and interfaces for building Java Card RMI-based client applications.
javacard.framework	Provides Java Card API exceptions that may be re-thrown on the client.
javacard.framework.service	Provides Java Card API exceptions that may be re-thrown on the client.
javacard.security	Provides Java Card platform exceptions that may be re-thrown on the client.
javacardx.biometry	Extension package that contains functionality for implementing a biometric framework on the Java Card platform.
javacardx.external	Extension package that provides mechanisms to access memory subsystems which are not directly addressable by the Java Card runtime environment(Java Card RE) on the Java Card platform.
javacardx.framework.tlv	Extension package that contains functionality, for managing storage for BER TLV formatted data, based on the ASN.1 BER encoding rules of ISO/IEC 8825-1:2002, as well as parsing and editing BER TLV formatted data in I/O buffers.
javacardx.framework.util	Extension package that contains common utility functions for manipulating arrays of primitive components - `byte`, `short` or `int`.

Hierarchy For All Packages

Package Hierarchies:

com.sun.javacard.clientlib, com.sun.javacard.javax.smartcard.rmiclient, com.sun.javacard.rmiclientlib, javacard.framework, javacard.framework.service, javacard.security, javacardx.biometry, javacardx.external, javacardx.framework.tlv, javacardx.framework.util

Class Hierarchy

- java.lang.Object
  - com.sun.javacard.clientlib.**ApduIOCardAccessor** (implements com.sun.javacard.clientlib.CardAccessor)
  - com.sun.javacard.javax.smartcard.rmiclient.**CardObjectFactory**
    - com.sun.javacard.rmiclientlib.**JCCardObjectFactory**
    - com.sun.javacard.rmiclientlib.**JCCardProxyFactory**
  - com.sun.javacard.rmiclientlib.**JCRemoteRefImpl** (implements java.lang.reflect.InvocationHandler, java.rmi.server.RemoteRef)
  - com.sun.javacard.rmiclientlib.**JCRMIConnect**
  - java.lang.Throwable
    - java.lang.Exception
      - javacard.framework.**CardException**
        - com.sun.javacard.javax.smartcard.rmiclient.**CardExceptionSubclass**
        - javacard.framework.**UserException**
          - com.sun.javacard.javax.smartcard.rmiclient.**UserExceptionSubclass**
      - java.lang.RuntimeException
        - javacard.framework.**CardRuntimeException**
          - javacard.framework.**APDUException**
            - com.sun.javacard.javax.smartcard.rmiclient.**APDUExceptionSubclass**
          - javacardx.biometry.**BioException**
            - com.sun.javacard.javax.smartcard.rmiclient.

            - **BioExceptionSubclass**
        - com.sun.javacard.javax.smartcard.rmiclient.**CardRuntimeExceptionSubclass**
        - javacard.security.**CryptoException**
            - com.sun.javacard.javax.smartcard.rmiclient.**CryptoExceptionSubclass**
        - javacardx.external.**ExternalException**
            - com.sun.javacard.javax.smartcard.rmiclient.**ExternalExceptionSubclass**
        - javacard.framework.**ISOException**
            - com.sun.javacard.javax.smartcard.rmiclient.**ISOExceptionSubclass**
        - javacard.framework.**PINException**
            - com.sun.javacard.javax.smartcard.rmiclient.**PINExceptionSubclass**
        - javacard.framework.service.**ServiceException**
            - com.sun.javacard.javax.smartcard.rmiclient.**ServiceExceptionSubclass**
        - javacard.framework.**SystemException**
            - com.sun.javacard.javax.smartcard.rmiclient.**SystemExceptionSubclass**
        - javacardx.framework.tlv.**TLVException**
            - com.sun.javacard.javax.smartcard.rmiclient.**TLVExceptionSubclass**
        - javacard.framework.**TransactionException**
            - com.sun.javacard.javax.smartcard.rmiclient.**TransactionExceptionSubclass**
        - javacardx.framework.util.**UtilException**
            - com.sun.javacard.javax.smartcard.rmiclient.**UtilExceptionSubclass**

Interface Hierarchy

- com.sun.javacard.clientlib.**CardAccessor**

Deprecated API

Contents

- Deprecated Methods

Deprecated Methods
com.sun.javacard.rmiclientlib.JCRemoteRefImpl.done(RemoteCall)
com.sun.javacard.rmiclientlib.JCRemoteRefImpl.invoke(RemoteCall)
com.sun.javacard.rmiclientlib.JCRemoteRefImpl.newCall(RemoteObject, Operation[], int, long)

throwIt(short) - Static method in exception javacard.framework.TransactionException

Throws an instance of `TransactionException` with the specified reason.

throwIt(short) - Static method in exception javacard.framework.UserException

Throws an instance of `UserException` with the specified reason.

throwIt(short) - Static method in exception javacard.security.CryptoException

Throws an instance of `CryptoException` with the specified reason.

throwIt(short) - Static method in exception javacardx.biometry.BioException

Throws the Java Card runtime environment owned instance of BioException with the specified reason.

throwIt(short) - Static method in exception javacardx.external.ExternalException

Throws the Java Card runtime environment-owned instance of `ExternalException` with the specified reason.

throwIt(short) - Static method in exception javacardx.framework.tlv.TLVException

Throws the Java Card runtime environment-owned instance of `TLVException` with the specified reason.

throwIt(short) - Static method in exception javacardx.framework.util.UtilException

Throws the Java Card runtime environment-owned instance of `UtilException` with the specified reason.

TLV_LENGTH_GREATER_THAN_32767 - Static variable in exception javacardx.framework.tlv.TLVException

This reason code is used to indicate that the Length component value in the TLV is greater than 32767

TLV_SIZE_GREATER_THAN_32767 - Static variable in exception javacardx.framework.tlv.TLVException

This reason code is used to indicate that the TLV requires more that 32767 bytes to represent

TLVException - Exception in javacardx.framework.tlv

`TLVException` represents a TLV-related exception.

TLVException(short) - Constructor for exception javacardx.framework.tlv.TLVException

Constructs a `TLVException` with the specified reason.

TLVExceptionSubclass - Exception in com.sun.javacard.javax.smartcard.rmiclient

`TLVException` represents a TLV-related exception.

TLVExceptionSubclass(String, short) - Constructor for exception com.sun.javacard.javax.smartcard.rmiclient.TLVExceptionSubclass

TransactionException - Exception in javacard.framework

`TransactionException` represents an exception in the transaction subsystem.

TransactionException(short) - Constructor for exception javacard.framework.TransactionException

Constructs a TransactionException with the specified reason.

TransactionExceptionSubclass - Exception in com.sun.javacard.javax.smartcard.rmiclient

This exception class represents a card subclass of `TransactionException`.

Unsupported operation.

How This API Document Is Organized

This API (Application Programming Interface) document has pages corresponding to the items in the navigation bar, described as follows.

Overview

The Overview page is the front page of this API document and provides a list of all packages with a summary for each. This page can also contain an overall description of the set of packages.

Package

Each package has a page that contains a list of its classes and interfaces, with a summary for each. This page can contain four categories:

- Interfaces (italic)
- Classes
- Enums
- Exceptions
- Errors
- Annotation Types

Class/Interface

Each class, interface, nested class and nested interface has its own separate page. Each of these pages has three sections consisting of a class/interface description, summary tables, and detailed member descriptions:

- Class inheritance diagram
- Direct Subclasses
- All Known Subinterfaces
- All Known Implementing Classes
- Class/interface declaration
- Class/interface description

- Nested Class Summary
- Field Summary
- Constructor Summary
- Method Summary

- Field Detail
- Constructor Detail
- Method Detail

Each summary entry contains the first sentence from the detailed description for that item. The summary entries are alphabetical, while the detailed descriptions are in the order they appear in the source code. This preserves the logical groupings established by the programmer.

Annotation Type

Each annotation type has its own separate page with the following sections:

- Annotation Type declaration
- Annotation Type description
- Required Element Summary
- Optional Element Summary
- Element Detail

Enum

Each enum has its own separate page with the following sections:

- Enum declaration
- Enum description
- Enum Constant Summary
- Enum Constant Detail

Tree (Class Hierarchy)

There is a Class Hierarchy page for all packages, plus a hierarchy for each package. Each hierarchy page contains a list of classes and a list of interfaces. The classes are organized by inheritance structure starting with `java.lang.Object`. The interfaces do not inherit from `java.lang.Object`.

- When viewing the Overview page, clicking on "Tree" displays the hierarchy for

all packages.
- When viewing a particular package, class or interface page, clicking "Tree" displays the hierarchy for only that package.

Deprecated API

The Deprecated API page lists all of the API that have been deprecated. A deprecated API is not recommended for use, generally due to improvements, and a replacement API is usually given. Deprecated APIs may be removed in future implementations.

Index

The Index contains an alphabetic list of all classes, interfaces, constructors, methods, and fields.

Prev/Next

These links take you to the next or previous class, interface, package, or related page.

Frames/No Frames

These links show and hide the HTML frames. All pages are available with or without frames.

Constant Field Values

The Constant Field Values page lists the static final fields and their values.

This help file applies to API documentation generated using the standard doclet.

All Classes

Package javacard.framework

Provides Java Card API exceptions that may be re-thrown on the client.

See:
Description

Exception Summary	
APDUException	`APDUException` represents an `APDU` related exception.
CardException	The `CardException` class defines a field `reason` and two accessor methods `getReason()` and `setReason()`.
CardRuntimeException	The `CardRuntimeException` class defines a field `reason` and two accessor methods `getReason()` and `setReason()`.
ISOException	`ISOException` class encapsulates an ISO 7816-4 response status word as its `reason` code.
PINException	`PINException` represents a `OwnerPIN` class access-related exception.
SystemException	`SystemException` represents a `JCSystem` class related exception.
TransactionException	`TransactionException` represents an exception in the transaction subsystem.
UserException	`UserException` represents a User exception.

Package javacard.framework Description

Provides Java Card API exceptions that may be re-thrown on the client.

Hierarchy For Package javacard.framework

Package Hierarchies:

All Packages

Class Hierarchy

- java.lang.Object
  - java.lang.Throwable
    - java.lang.Exception
      - javacard.framework.**CardException**
        - javacard.framework.**UserException**
      - java.lang.RuntimeException
        - javacard.framework.**CardRuntimeException**
          - javacard.framework.**APDUException**
          - javacard.framework.**ISOException**
          - javacard.framework.**PINException**
          - javacard.framework.**SystemException**
          - javacard.framework.**TransactionException**

Hierarchy For Package com.sun.javacard.rmiclientlib

Package Hierarchies:

All Packages

Class Hierarchy

- java.lang.Object
  - com.sun.javacard.javax.smartcard.rmiclient.**CardObjectFactory**
    - com.sun.javacard.rmiclientlib.**JCCardObjectFactory**
    - com.sun.javacard.rmiclientlib.**JCCardProxyFactory**
  - com.sun.javacard.rmiclientlib.**JCRemoteRefImpl** (implements java.lang.reflect.InvocationHandler, java.rmi.server.RemoteRef)
  - com.sun.javacard.rmiclientlib.**JCRMIConnect**

Package com.sun.javacard.rmiclientlib

Provides a framework of classes and interfaces for building Java Card RMI-based client applications.

See:
Description

Class Summary	
JCCardObjectFactory	Processes the data returned from the card in the format defined for Java Card RMI.
JCCardProxyFactory	Processes the data returned from the card in the format defined for Java Card RMI.
JCRemoteRefImpl	Represents a reference to a card object.
JCRMIConnect	The main class of the Java Card RMI client API.

Package com.sun.javacard.rmiclientlib Description

Provides a framework of classes and interfaces for building Java Card RMI-based client applications.

Package com.sun.javacard.javax.smartcard.rmiclient

Provides framework of classes and interfaces for a Java Card technology-based client.

See:
Description

Class Summary	
CardObjectFactory	This `CardObjectFactory` abstract class represents the base class for Java Card RMI version 2.2.2 stub generation implementations.

Exception Summary	
APDUExceptionSubclass	This exception class represents a card subclass of `APDUException`.
BioExceptionSubclass	The `BioException` class encapsulates specific exceptions which can be thrown by the methods of the `javacardx.biometry` package in case of error.
CardExceptionSubclass	This exception class represents a subclass of `CardException` on the card.
CardRuntimeExceptionSubclass	This exception class represents a subclass of `CardRuntimeException` on the card.
CryptoExceptionSubclass	This exception class represents a card subclass of `CryptoException`.
ExternalExceptionSubclass	`ExternalException` represents an external subsystem related exception.
ISOExceptionSubclass	This exception class represents a card subclass of `ISOException`.
PINExceptionSubclass	This exception class represents a card subclass of `PINException`.

ServiceExceptionSubclass	This exception class represents a card subclass of `ServiceException`.
SystemExceptionSubclass	This exception class represents a card subclass of `SystemException`.
TLVExceptionSubclass	`TLVException` represents a TLV-related exception.
TransactionExceptionSubclass	This exception class represents a card subclass of `TransactionException`.
UserExceptionSubclass	This exception class represents a card subclass of `UserException`.
UtilExceptionSubclass	`UtilException` represents a util related exception.

Package com.sun.javacard.javax.smartcard.rmiclient Description

Provides framework of classes and interfaces for a Java Card technology-based client.

Hierarchy For Package com.sun.javacard.javax.smartcard.rmiclient

Package Hierarchies:

All Packages

Class Hierarchy

- java.lang.Object
    - com.sun.javacard.javax.smartcard.rmiclient.**CardObjectFactory**
    - java.lang.Throwable
        - java.lang.Exception
            - javacard.framework.**CardException**
                - com.sun.javacard.javax.smartcard.rmiclient.**CardExceptionSubclass**
                - javacard.framework.**UserException**
                    - com.sun.javacard.javax.smartcard.rmiclient.**UserExceptionSubclass**
            - java.lang.RuntimeException
                - javacard.framework.**CardRuntimeException**
                    - javacard.framework.**APDUException**
                        - com.sun.javacard.javax.smartcard.rmiclient.**APDUExceptionSubclass**
                    - javacardx.biometry.**BioException**
                        - com.sun.javacard.javax.smartcard.rmiclient.**BioExceptionSubclass**
                    - com.sun.javacard.javax.smartcard.rmiclient.**CardRuntimeExceptionSubclass**
                    - javacard.security.**CryptoException**
                        - com.sun.javacard.javax.smartcard.rmiclient.**CryptoExceptionSubclass**
                    - javacardx.external.**ExternalException**
                        - com.sun.javacard.javax.smartcard.rmiclient.

        - **ExternalExceptionSubclass**
    - javacard.framework.**ISOException**
        - com.sun.javacard.javax.smartcard.rmiclient.**ISOExceptionSubclass**
    - javacard.framework.**PINException**
        - com.sun.javacard.javax.smartcard.rmiclient.**PINExceptionSubclass**
    - javacard.framework.service.**ServiceException**
        - com.sun.javacard.javax.smartcard.rmiclient.**ServiceExceptionSubclass**
    - javacard.framework.**SystemException**
        - com.sun.javacard.javax.smartcard.rmiclient.**SystemExceptionSubclass**
    - javacardx.framework.tlv.**TLVException**
        - com.sun.javacard.javax.smartcard.rmiclient.**TLVExceptionSubclass**
    - javacard.framework.**TransactionException**
        - com.sun.javacard.javax.smartcard.rmiclient.**TransactionExceptionSubclass**
    - javacardx.framework.util.**UtilException**
        - com.sun.javacard.javax.smartcard.rmiclient.**UtilExceptionSubclass**

Hierarchy For Package com.sun.javacard.clientlib

Package Hierarchies:
All Packages

Class Hierarchy

- java.lang.Object
  - com.sun.javacard.clientlib.**ApduIOCardAccessor** (implements com.sun.javacard.clientlib.CardAccessor)

Interface Hierarchy

- com.sun.javacard.clientlib.**CardAccessor**

Package com.sun.javacard.clientlib

Provides a framework for building client applications capable of exchanging APDUs with Java Cards.

See:
Description

Interface Summary	
CardAccessor	The `CardAccessor` interface represents a generic smartcard communication API.

Class Summary	
ApduIOCardAccessor	Implementation of CardAccessor using ApduIO library

Package com.sun.javacard.clientlib Description

Provides a framework for building client applications capable of exchanging APDUs with Java Cards.

com.sun.javacard.javax.smartcard.rmiclient

Class APDUExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.framework.APDUException
                      └com.sun.javacard.javax.smartcard.rmiclient.
APDUExceptionSubclass
```

All Implemented Interfaces:

`public class` **`APDUExceptionSubclass`**

extends APDUException

This exception class represents a card subclass of `APDUException`. `APDUException` represents an `APDU` related exception on the card.

See Also:

Field Summary

Fields inherited from class javacard.framework.APDUException
`BAD_LENGTH`, `BUFFER_BOUNDS`, `ILLEGAL_USE`, `IO_ERROR`, `NO_T0_GETRESPONSE`, `NO_T0_REISSUE`, `T1_IFD_ABORT`

Constructor Summary

APDUExceptionSubclass(java.lang.String msg, short reason)   Constructs an APDUExceptionSubclass with the specified reason and specified error message.

Method Summary

java.lang.String	**getMessage**() Returns the error message string of this throwable object.

Methods inherited from class javacard.framework.APDUException

`throwIt`

Methods inherited from class javacard.framework.CardRuntimeException

`getReason, setReason`

Methods inherited from class java.lang.Throwable

`fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object

`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Detail

APDUExceptionSubclass

```
public APDUExceptionSubclass(java.lang.String msg,
                             short reason)
```

Constructs an APDUExceptionSubclass with the specified reason and specified error message.

Parameters:

`msg` - the associated message string
`reason` - the reason for the exception.

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Returns the error message string of this throwable object.

Overrides:

`getMessage` in class `java.lang.Throwable`

Returns:

the error message string of this Throwable object if it was created with an error message string; or null if it was created with no error message.

---

com.sun.javacard.javax.smartcard.rmiclient

Class BioExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacardx.biometry.BioException
                      └com.sun.javacard.javax.smartcard.rmiclient.
BioExceptionSubclass
```

All Implemented Interfaces:

`public class` **`BioExceptionSubclass`**

extends BioException

The `BioException` class encapsulates specific exceptions which can be thrown by the methods of the `javacardx.biometry` package in case of error.

See Also:

Field Summary

Fields inherited from class javacardx.biometry.BioException
`ILLEGAL_USE`, `ILLEGAL_VALUE`, `INVALID_DATA`, `NO_SUCH_BIO_TEMPLATE`, `NO_TEMPLATES_ENROLLED`

Constructor Summary

BioExceptionSubclass(java.lang.String msg, short reason)

Method Summary

java.lang.String	**getMessage**()

Methods inherited from class javacardx.biometry.BioException
throwIt

Methods inherited from class javacard.framework.CardRuntimeException
getReason, setReason

Methods inherited from class java.lang.Throwable
fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

BioExceptionSubclass

```
public BioExceptionSubclass(java.lang.String msg,
                            short reason)
```

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Overrides:
`getMessage` in class `java.lang.Throwable`

com.sun.javacard.javax.smartcard.rmiclient

Class CardExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └javacard.framework.CardException
              └com.sun.javacard.javax.smartcard.rmiclient.
CardExceptionSubclass
```

All Implemented Interfaces:

`public class` **`CardExceptionSubclass`**

extends CardException

This exception class represents a subclass of `CardException` on the card. The `CardException` class on the card defines a field `reason` and two accessor methods `getReason()` and `setReason()`. The `reason` field encapsulates the exception cause identifier in the Java Card API All Java Card API checked Exception classes on the card should extend `CardException`.

See Also:

Constructor Summary

`CardExceptionSubclass``(java.lang.String msg, short reason)` Construct a CardExceptionSubclass instance with the specified reason and the specified error message.

Method Summary

`java.lang.String`	**`getMessage`**`()` Returns the error message string of this throwable object.

Methods inherited from class javacard.framework.CardException
`getReason, setReason, throwIt`

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Detail

CardExceptionSubclass

```
public CardExceptionSubclass(java.lang.String msg,
                             short reason)
```

Construct a CardExceptionSubclass instance with the specified reason and the specified error message.

Parameters:

`msg` - the associated message string
`reason` - the reason for the exception

Method Detail

getMessage

`public java.lang.String` **`getMessage`**`()`

Returns the error message string of this throwable object.

Overrides:
`getMessage` in class `java.lang.Throwable`

Returns:
the error message string of this Throwable object if it was created with an error message string; or null if it was created with no error message.

com.sun.javacard.javax.smartcard.rmiclient

Class CardObjectFactory

```
java.lang.Object
  └com.sun.javacard.javax.smartcard.rmiclient.CardObjectFactory
```

Direct Known Subclasses:

JCCardObjectFactory, JCCardProxyFactory

`public abstract class` **`CardObjectFactory`**

extends java.lang.Object

This `CardObjectFactory` abstract class represents the base class for Java Card RMI version 2.2.2 stub generation implementations. An instance of this class is associated with one Java Card applet selection session.

Field Summary

`static byte`	**`REF_FORMAT_CLASS`** This value (=1) is used to signify that the CardObjectFactory implementation suppports the Java Card RMI remote reference format using the name of the card implementation remote class.
`static byte`	**`REF_FORMAT_INTERFACES`** This value (=2) is used to signify that the CardObjectFactory implementation suppports the Java Card RMI remote reference format using the names of the remote interfaces implemented by the card implementation remote class.
`static byte`	**`REF_FORMAT_NONE`** This value (=0) is used to signify that the CardObjectFactory implementation does not suppport any Java Card RMI remote reference descriptor formats.

Constructor Summary

`CardObjectFactory``()` Creates a new CardObjectFactory for this RMI session

Method Summary

`byte`	**`getINSByte`**`()` returns the configured ISO 7816-4 command INS byte to be used in the Java Card platform remote method invocation command
`java.lang.Object`	**`getObject`**`(byte[] buffer, int tagOffset, java.lang.Class type, CardAccessor cardAccessor)` This abstract method returns the instance of the card object corresponding to the value returned from the card.
`protected abstract java.rmi.Remote`	**`getRemoteObject`**`(byte[] buffer, int tagOffset, CardAccessor cardAccessor)` This abstract method instantiates a stub or proxy object corresponding to the remote reference returned from the card
`abstract byte`	**`getRemoteRefFormat`**`()` returns the format of Java Card RMI remote object reference descriptor supported.
`void`	**`setINSByte`**`(byte ins)` Sets the ISO 7816-4 header INS byte to use for Java Card RMI method invocation commands.

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Field Detail

REF_FORMAT_NONE

`public static final byte` **`REF_FORMAT_NONE`**

This value (=0) is used to signify that the CardObjectFactory implementation does not suppport any Java Card RMI remote reference descriptor formats.

See Also:
Constant Field Values

REF_FORMAT_CLASS

```
public static final byte REF_FORMAT_CLASS
```

This value (=1) is used to signify that the CardObjectFactory implementation suppports the Java Card RMI remote reference format using the name of the card implementation remote class.

See Also:
Constant Field Values

REF_FORMAT_INTERFACES

```
public static final byte REF_FORMAT_INTERFACES
```

This value (=2) is used to signify that the CardObjectFactory implementation suppports the Java Card RMI remote reference format using the names of the remote interfaces implemented by the card implementation remote class.

See Also:
Constant Field Values

Constructor Detail

CardObjectFactory

```
public CardObjectFactory()
```

Creates a new CardObjectFactory for this RMI session

Method Detail

getObject

```
public java.lang.Object getObject(byte[] buffer,
                                  int tagOffset,
                                  java.lang.Class type,
                                  CardAccessor cardAccessor)
                           throws java.rmi.RemoteException,
                                  java.rmi.StubNotFoundException,
                                  java.lang.Exception
```

This abstract method returns the instance of the card object corresponding to the value returned from the card. The value returned by the card may be any of the valid return types in Java Card RMI version 2.2

protocol format. In particular the return type may be a primitive, an array type or a remote reference descriptor or null. The method delegates processing to getRemoteObject(...) when the response tag is normal and expected type is a remote reference.

Parameters:

`cardAccessor` - used to instantiate stubs or proxies of remote objects
`buffer` - the byte array containing the Java Card RMI version 2.2 response data
`tagOffset` - the offset within the array where the response tag is located
`type` - the expected return type in the response data

Returns:

the object associated with the response data. Primitive return value is encapsulated in a wrapper object.

Throws:

`java.rmi.StubNotFoundException` - if an appropriate object could not be instantiated
`java.rmi.RemoteException` - for any RMI-specific exceptions
`java.lang.Exception` - for exceptions thrown on the card

---

getRemoteObject

```
protected abstract java.rmi.Remote getRemoteObject(byte[] buffer,
                                                   int tagOffset,
                                                   CardAccessor cardAccessor)
                                            throws java.lang.Exception
```

This abstract method instantiates a stub or proxy object corresponding to the remote reference returned from the card

Parameters:

`buffer` - the byte array containing the JC 2.2 RMI response data
`tagOffset` - the offset within the array where the response tag is located

Returns:

Stub or proxy

Throws:

`java.lang.Exception` - In case of error during processing

---

setINSByte

```
public void setINSByte(byte ins)
```

Sets the ISO 7816-4 header INS byte to use for Java Card RMI method invocation commands.

Parameters:

`ins` - the ISO 7816-4 INS command header byte to use for Java Card RMI invocation commands

---

getINSByte

```
public byte getINSByte()
```

returns the configured ISO 7816-4 command INS byte to be used in the Java Card platform remote method invocation command

Returns:
the ISO-7816-4 INS byte

Since:
Java Card 2.2

---

getRemoteRefFormat

```
public abstract byte getRemoteRefFormat()
```

returns the format of Java Card RMI remote object reference descriptor supported. It returns one of : REF_FORMAT_NONE, REF_FORMAT_CLASS, REF_FORMAT_INTERFACES

Returns:
one of the REF_FORMAT.. values defined above

---

com.sun.javacard.javax.smartcard.rmiclient

Class CardRuntimeExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └com.sun.javacard.javax.smartcard.rmiclient.
CardRuntimeExceptionSubclass
```

All Implemented Interfaces:

`public class` **`CardRuntimeExceptionSubclass`**

extends CardRuntimeException

This exception class represents a subclass of `CardRuntimeException` on the card. The `CardRuntimeException` class on the card defines a field `reason` and two accessor methods `getReason()` and `setReason()`. The `reason` field encapsulates exception cause identifier in the Java Card API. All Java Card API unchecked Exception classes on the card should extend `CardRuntimeException`.

See Also:

Constructor Summary
`CardRuntimeExceptionSubclass``(java.lang.String msg, short reason)` Construct a CardRuntimeExceptionSubclass instance with the specified reason and the specified error message.

Method Summary

`java.lang.String`	`getMessage()` Returns the error message string of this throwable object.

Methods inherited from class javacard.framework.CardRuntimeException
`getReason, setReason, throwIt`

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Detail

CardRuntimeExceptionSubclass

```
public CardRuntimeExceptionSubclass(java.lang.String msg,
                                    short reason)
```

Construct a CardRuntimeExceptionSubclass instance with the specified reason and the specified error message.

Parameters:

`msg` - the associated message string
`reason` - the reason for the exception

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Returns the error message string of this throwable object.

Overrides:
`getMessage` in class `java.lang.Throwable`

Returns:
the error message string of this Throwable object if it was created with an error message string; or null if it was created with no error message.

com.sun.javacard.javax.smartcard.rmiclient

Class CryptoExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.security.CryptoException
                      └com.sun.javacard.javax.smartcard.rmiclient.
CryptoExceptionSubclass
```

All Implemented Interfaces:

`public class` **`CryptoExceptionSubclass`**

extends CryptoException

This exception class represents a card subclass of `CryptoException`. `CryptoException` represents a cryptography-related exception.

See Also:

Field Summary

Fields inherited from class javacard.security.CryptoException
`ILLEGAL_USE`, `ILLEGAL_VALUE`, `INVALID_INIT`, `NO_SUCH_ALGORITHM`, `UNINITIALIZED_KEY`

Constructor Summary

CryptoExceptionSubclass(java.lang.String msg, short reason) Constructs an CryptoExceptionSubclass with the specified reason and specified error message.

Method Summary

java.lang.String	**getMessage**() Returns the error message string of this throwable object.

Methods inherited from class javacard.security.CryptoException
throwIt

Methods inherited from class javacard.framework.CardRuntimeException
getReason, setReason

Methods inherited from class java.lang.Throwable
fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

CryptoExceptionSubclass

```
public CryptoExceptionSubclass(java.lang.String msg,
                               short reason)
```

Constructs an CryptoExceptionSubclass with the specified reason and specified error message.

Parameters:

`msg` - the associated message string
`reason` - the reason for the exception.

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Returns the error message string of this throwable object.

Overrides:

`getMessage` in class `java.lang.Throwable`

Returns:

the error message string of this Throwable object if it was created with an error message string; or null if it was created with no error message.

com.sun.javacard.javax.smartcard.rmiclient

Class ExternalExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacardx.external.ExternalException
                      └com.sun.javacard.javax.smartcard.rmiclient.
ExternalExceptionSubclass
```

All Implemented Interfaces:

`public class` **`ExternalExceptionSubclass`**

extends ExternalException

`ExternalException` represents an external subsystem related exception.

The API classes throw Java Card runtime environment-owned instances of `ExternalException`.

Java Card runtime environment-owned instances of exception classes are temporary Java Card runtime environment Entry Point Objects and can be accessed from any applet context. References to these temporary objects cannot be stored in class variables or instance variables or array components.

Since:
2.2.2

See Also:

Field Summary

Fields inherited from class javacardx.external.ExternalException
`INTERNAL_ERROR`, `INVALID_PARAM`, `NO_SUCH_SUBSYSTEM`

Constructor Summary

`ExternalExceptionSubclass``(java.lang.String msg, short reason)`

Method Summary

`java.lang.String`	**`getMessage`**`()`

Methods inherited from class javacardx.external.ExternalException
`throwIt`

Methods inherited from class javacard.framework.CardRuntimeException
`getReason`, `setReason`

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Detail

ExternalExceptionSubclass

```
public ExternalExceptionSubclass(java.lang.String msg,
                                 short reason)
```

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Overrides:
`getMessage` in class `java.lang.Throwable`

---

com.sun.javacard.javax.smartcard.rmiclient

Class ISOExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.framework.ISOException
                      └com.sun.javacard.javax.smartcard.rmiclient.
ISOExceptionSubclass
```

All Implemented Interfaces:

`public class` **`ISOExceptionSubclass`**

extends ISOException

This exception class represents a card subclass of `ISOException`. `ISOException` class encapsulates an ISO 7816-4 response status word as its `reason` code.

See Also:

Constructor Summary

`ISOExceptionSubclass``(java.lang.String msg, short reason)` Constructs an ISOExceptionSubclass with the specified reason and specified error message.

Method Summary

`java.lang.String`	**`getMessage`**`()` Returns the error message string of this throwable object.

Methods inherited from class javacard.framework.ISOException
`throwIt`

Methods inherited from class javacard.framework.CardRuntimeException
`getReason, setReason`

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Detail

ISOExceptionSubclass

```
public ISOExceptionSubclass(java.lang.String msg,
                            short reason)
```

Constructs an ISOExceptionSubclass with the specified reason and specified error message.

Parameters:

`msg` - the associated message string
`reason` - the ISO 7816-4 defined status word

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Returns the error message string of this throwable object.

Overrides:
`getMessage` in class `java.lang.Throwable`

Returns:
the error message string of this Throwable object if it was created with an error message string; or null if it was created with no error message.

com.sun.javacard.javax.smartcard.rmiclient

Class PINExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.framework.PINException
                      └com.sun.javacard.javax.smartcard.rmiclient.
PINExceptionSubclass
```

All Implemented Interfaces:

`public class` **`PINExceptionSubclass`**

extends PINException

This exception class represents a card subclass of `PINException`. `PINException` represents a `OwnerPIN` class access-related exception.

See Also:

Field Summary

Fields inherited from class javacard.framework.PINException
`ILLEGAL_VALUE`

Constructor Summary

PINExceptionSubclass(java.lang.String msg, short reason) Constructs an PINExceptionSubclass with the specified reason and specified error message.

Method Summary

java.lang.String	**getMessage**() Returns the error message string of this throwable object.

Methods inherited from class javacard.framework.PINException

throwIt

Methods inherited from class javacard.framework.CardRuntimeException

getReason, setReason

Methods inherited from class java.lang.Throwable

fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

PINExceptionSubclass

```
public PINExceptionSubclass(java.lang.String msg,
                            short reason)
```

Constructs an PINExceptionSubclass with the specified reason and specified error message.

Parameters:

`msg` - the associated message string
`reason` - the reason for the exception.

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Returns the error message string of this throwable object.

Overrides:

`getMessage` in class `java.lang.Throwable`

Returns:

the error message string of this Throwable object if it was created with an error message string; or null if it was created with no error message.

com.sun.javacard.javax.smartcard.rmiclient

Class ServiceExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.framework.service.ServiceException
                      └com.sun.javacard.javax.smartcard.rmiclient.
ServiceExceptionSubclass
```

All Implemented Interfaces:

`public class` **`ServiceExceptionSubclass`**

extends ServiceException

This exception class represents a card subclass of `ServiceException`. `ServiceException` represents a service framework related exception.

See Also:

Field Summary

Fields inherited from class javacard.framework.service.ServiceException

CANNOT_ACCESS_IN_COMMAND, CANNOT_ACCESS_OUT_COMMAND, COMMAND_DATA_TOO_LONG, COMMAND_IS_FINISHED, DISPATCH_TABLE_FULL, ILLEGAL_PARAM, REMOTE_OBJECT_NOT_EXPORTED

Constructor Summary
ServiceExceptionSubclass(java.lang.String msg, short reason) Constructs an ServiceExceptionSubclass with the specified reason and specified error message.

Method Summary	
java.lang.String	**getMessage**() Returns the error message string of this throwable object.

Methods inherited from class javacard.framework.service.ServiceException
throwIt

Methods inherited from class javacard.framework.CardRuntimeException
getReason, setReason

Methods inherited from class java.lang.Throwable
fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

ServiceExceptionSubclass

```
public ServiceExceptionSubclass(java.lang.String msg,
                                short reason)
```

Constructs an ServiceExceptionSubclass with the specified reason and specified error message.

Parameters:

`msg` - the associated message string
`reason` - the reason for the exception.

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Returns the error message string of this throwable object.

Overrides:

`getMessage` in class `java.lang.Throwable`

Returns:

the error message string of this Throwable object if it was created with an error message string; or null if it was created with no error message.

com.sun.javacard.javax.smartcard.rmiclient

Class SystemExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.framework.SystemException
                      └com.sun.javacard.javax.smartcard.rmiclient.SystemExceptionSubclass
```

All Implemented Interfaces:

`public class` **`SystemExceptionSubclass`**

extends SystemException

This exception class represents a card subclass of `SystemException`. `SystemException` represents a `JCSystem` class related exception.

See Also:

Field Summary

Fields inherited from class javacard.framework.SystemException
`ILLEGAL_AID`, `ILLEGAL_TRANSIENT`, `ILLEGAL_USE`, `ILLEGAL_VALUE`, `NO_RESOURCE`, `NO_TRANSIENT_SPACE`

Constructor Summary

SystemExceptionSubclass(java.lang.String msg, short reason) Constructs an SystemExceptionSubclass with the specified reason and specified error message.

Method Summary

java.lang.String	**getMessage**() Returns the error message string of this throwable object.

Methods inherited from class javacard.framework.SystemException
throwIt

Methods inherited from class javacard.framework.CardRuntimeException
getReason, setReason

Methods inherited from class java.lang.Throwable
fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

SystemExceptionSubclass

```
public SystemExceptionSubclass(java.lang.String msg,
                               short reason)
```

Constructs an SystemExceptionSubclass with the specified reason and specified error message.

Parameters:

`msg` - the associated message string
`reason` - the reason for the exception.

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Returns the error message string of this throwable object.

Overrides:
`getMessage` in class `java.lang.Throwable`

Returns:
the error message string of this Throwable object if it was created with an error message string; or null if it was created with no error message.

com.sun.javacard.javax.smartcard.rmiclient

Class TLVExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacardx.framework.tlv.TLVException
                      └com.sun.javacard.javax.smartcard.rmiclient.
TLVExceptionSubclass
```

All Implemented Interfaces:

`public class` **`TLVExceptionSubclass`**

extends TLVException

`TLVException` represents a TLV-related exception.

The API classes throw Java Card runtime environment-owned instances of `TLVException`.

Java Card runtime environment-owned instances of exception classes are temporary Java Card runtime environment Entry Point Objects and can be accessed from any applet context. References to these temporary objects cannot be stored in class variables, instance variables, or array components.

See Also:

Field Summary

Fields inherited from class javacardx.framework.tlv.TLVException
EMPTY_TAG, EMPTY_TLV, ILLEGAL_SIZE, INSUFFICIENT_STORAGE, INVALID_PARAM, MALFORMED_TAG, MALFORMED_TLV, TAG_NUMBER_GREATER_THAN_32767, TAG_SIZE_GREATER_THAN_127, TLV_LENGTH_GREATER_THAN_32767, TLV_SIZE_GREATER_THAN_32767

Constructor Summary
TLVExceptionSubclass(java.lang.String msg, short reason)

Method Summary	
java.lang.String	**getMessage**()

Methods inherited from class javacardx.framework.tlv.TLVException
throwIt

Methods inherited from class javacard.framework.CardRuntimeException
getReason, setReason

Methods inherited from class java.lang.Throwable
fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

TLVExceptionSubclass

```
public TLVExceptionSubclass(java.lang.String msg,
                            short reason)
```

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Overrides:

`getMessage` in class `java.lang.Throwable`

---

com.sun.javacard.javax.smartcard.rmiclient

Class TransactionExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.framework.TransactionException
                      └com.sun.javacard.javax.smartcard.rmiclient.
TransactionExceptionSubclass
```

All Implemented Interfaces:

`public class` **`TransactionExceptionSubclass`**

extends TransactionException

This exception class represents a card subclass of `TransactionException`. `TransactionException` represents an exception in the transaction subsystem.

See Also:

Field Summary

Fields inherited from class javacard.framework.TransactionException
`BUFFER_FULL, IN_PROGRESS, INTERNAL_FAILURE, NOT_IN_PROGRESS`

Constructor Summary

TransactionExceptionSubclass(java.lang.String msg, short reason) Constructs an TransactionExceptionSubclass with the specified reason and specified error message.

Method Summary

java.lang.String	**getMessage**() Returns the error message string of this throwable object.

Methods inherited from class javacard.framework.TransactionException
throwIt

Methods inherited from class javacard.framework.CardRuntimeException
getReason, setReason

Methods inherited from class java.lang.Throwable
fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

TransactionExceptionSubclass

```
public TransactionExceptionSubclass(java.lang.String msg,
                                    short reason)
```

Constructs an TransactionExceptionSubclass with the specified reason and specified error message.

Parameters:
`msg` - the associated message string
`reason` - the reason for the exception.

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Returns the error message string of this throwable object.

Overrides:
`getMessage` in class `java.lang.Throwable`

Returns:
the error message string of this Throwable object if it was created with an error message string; or null if it was created with no error message.

com.sun.javacard.javax.smartcard.rmiclient

Class UserExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └javacard.framework.CardException
              └javacard.framework.UserException
                  └com.sun.javacard.javax.smartcard.rmiclient.
UserExceptionSubclass
```

All Implemented Interfaces:

`public class` **`UserExceptionSubclass`**

extends UserException

This exception class represents a card subclass of `UserException`. `UserException` represents a User exception.

See Also:

Constructor Summary

`UserExceptionSubclass``(java.lang.String msg, short reason)` Constructs an UserExceptionSubclass with the specified reason and specified error message.

Method Summary

`java.lang.String`	**`getMessage`**`()` Returns the error message string of this throwable object.

Methods inherited from class javacard.framework.UserException
`throwIt`

Methods inherited from class javacard.framework.CardException
`getReason, setReason`

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Detail

UserExceptionSubclass

```
public UserExceptionSubclass(java.lang.String msg,
                             short reason)
```

Constructs an UserExceptionSubclass with the specified reason and specified error message.

Parameters:

`msg` - the associated message string
`reason` - the reason for the exception.

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Returns the error message string of this throwable object.

Overrides:

`getMessage` in class `java.lang.Throwable`

Returns:

the error message string of this Throwable object if it was created with an error message string; or null if it was created with no error message.

com.sun.javacard.javax.smartcard.rmiclient

Class UtilExceptionSubclass

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacardx.framework.util.UtilException
                      └com.sun.javacard.javax.smartcard.rmiclient.
UtilExceptionSubclass
```

All Implemented Interfaces:

`public class` **`UtilExceptionSubclass`**

extends UtilException

`UtilException` represents a util related exception.

The API classes throw Java Card runtime environment-owned instances of `UtilException`.

Java Card runtime environment-owned instances of exception classes are temporary Java Card runtime environment Entry Point Objects and can be accessed from any applet context. References to these temporary objects cannot be stored in class variables, instance variables, or array components.

See Also:

Field Summary

Fields inherited from class javacardx.framework.util.UtilException
`ILLEGAL_VALUE, TYPE_MISMATCHED`

Constructor Summary
`UtilExceptionSubclass(java.lang.String msg, short reason)`

Method Summary	
`java.lang.String`	`getMessage()`

Methods inherited from class javacardx.framework.util.UtilException
`throwIt`

Methods inherited from class javacard.framework.CardRuntimeException
`getReason, setReason`

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Detail

UtilExceptionSubclass

```
public UtilExceptionSubclass(java.lang.String msg,
                             short reason)
```

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Overrides:
`getMessage` in class `java.lang.Throwable`

javacard.framework

Class CardRuntimeException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
```

All Implemented Interfaces:

Direct Known Subclasses:

APDUException, BioException, CardRuntimeExceptionSubclass, CryptoException, ExternalException, ISOException, PINException, ServiceException, SystemException, TLVException, TransactionException, UtilException

---

`public class` **`CardRuntimeException`**

extends java.lang.RuntimeException

The `CardRuntimeException` class defines a field `reason` and two accessor methods `getReason()` and `setReason()`. The `reason` field encapsulates exception cause identifier in the Java Card API. All Java Card API unchecked Exception classes should extend `CardRuntimeException`.

See Also:

Constructor Summary

`CardRuntimeException(short reason)` Construct a CardRuntimeException instance with the specified reason.

Method Summary	
`short`	`getReason()` Get reason code
`void`	`setReason(short reason)` Set reason code.
`static void`	`throwIt(short reason)` Throw an instance of the `CardRuntimeException` class with the specified reason.

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Detail

CardRuntimeException

```
public CardRuntimeException(short reason)
```

Construct a CardRuntimeException instance with the specified reason.

Parameters:

`reason` - the reason for the exception

Method Detail

getReason

```
public short getReason()
```

Get reason code

Returns:
the reason for the exception

---

setReason

```
public void setReason(short reason)
```

Set reason code.

Parameters:
`reason` - the reason for the exception

---

throwIt

```
public static void throwIt(short reason)
                    throws CardRuntimeException
```

Throw an instance of the `CardRuntimeException` class with the specified reason.

Parameters:
`reason` - the reason for the exception

Throws:
`CardRuntimeException` - always.

---

javacard.framework

Class CardException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └javacard.framework.CardException
```

All Implemented Interfaces:

Direct Known Subclasses:

CardExceptionSubclass, UserException

`public class` **`CardException`**

extends java.lang.Exception

The `CardException` class defines a field `reason` and two accessor methods `getReason()` and `setReason()`. The `reason` field encapsulates exception cause identifier in the Java Card API. All Java Card API checked Exception classes should extend `CardException`.

See Also:

Constructor Summary
`CardException``(short reason)` Construct a CardException instance with the specified reason.

Method Summary

`short`	`getReason()` Get reason code
`void`	`setReason(short reason)` Set reason code.
`static void`	`throwIt(short reason)` Throw an instance of `CardException` class with the specified reason.

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Detail

CardException

`public` **`CardException`**`(short reason)`

Construct a CardException instance with the specified reason.

Parameters:

`reason` - the reason for the exception

Method Detail

getReason

`public short` **`getReason`**`()`

Get reason code

Returns:
the reason for the exception

---

setReason

```
public void setReason(short reason)
```

Set reason code.

Parameters:
`reason` - the reason for the exception

---

throwIt

```
public static void throwIt(short reason)
                    throws CardException
```

Throw an instance of `CardException` class with the specified reason.

Parameters:
`reason` - the reason for the exception
Throws:
`CardException` - always.

---

javacard.framework

Class UserException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └javacard.framework.CardException
              └javacard.framework.UserException
```

All Implemented Interfaces:

Direct Known Subclasses:

UserExceptionSubclass

`public class` **`UserException`**

extends CardException

`UserException` represents a User exception.

See Also:

Constructor Summary
`UserException``()` Constructs a `UserException` with reason = 0.
`UserException``(short reason)` Constructs a `UserException` with the specified reason.

Method Summary

`static void`	**`throwIt`**`(short reason)` Throws an instance of `UserException` with the specified reason.

Methods inherited from class javacard.framework.CardException
`getReason, setReason`

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Detail

UserException

`public` **`UserException`**`()`

Constructs a `UserException` with reason = 0.

UserException

`public` **`UserException`**`(short reason)`

Constructs a `UserException` with the specified reason.

Parameters:

`reason` - the reason for the exception.

Method Detail

throwIt

```
public static void throwIt(short reason)
                    throws UserException
```

Throws an instance of `UserException` with the specified reason.

Parameters:
`reason` - the reason for the exception.
Throws:
`UserException` - always.

javacard.framework

Class TransactionException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.framework.TransactionException
```

All Implemented Interfaces:

Direct Known Subclasses:

TransactionExceptionSubclass

`public class` **`TransactionException`**

extends CardRuntimeException

`TransactionException` represents an exception in the transaction subsystem.

See Also:

Field Summary	
`static short`	**`BUFFER_FULL`** This reason code is used during a transaction to indicate that the commit buffer is full.

`static short`	**IN_PROGRESS** This reason code is used by the `beginTransaction` method to indicate a transaction is already in progress.
`static short`	**INTERNAL_FAILURE** This reason code is used during a transaction to indicate an internal Java Card runtime environment problem (fatal error).
`static short`	**NOT_IN_PROGRESS** This reason code is used by the `abortTransaction` and `commitTransaction` methods when a transaction is not in progress.

Constructor Summary

TransactionException`(short reason)` Constructs a TransactionException with the specified reason.

Method Summary

`static void`	**throwIt**`(short reason)` Throws an instance of `TransactionException` with the specified reason.

Methods inherited from class javacard.framework.CardRuntimeException

`getReason, setReason`

Methods inherited from class java.lang.Throwable

`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object

`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Field Detail

IN_PROGRESS

`public static final short` **`IN_PROGRESS`**

This reason code is used by the `beginTransaction` method to indicate a transaction is already in progress.

See Also:
Constant Field Values

NOT_IN_PROGRESS

`public static final short` **`NOT_IN_PROGRESS`**

This reason code is used by the `abortTransaction` and `commitTransaction` methods when a transaction is not in progress.

See Also:
Constant Field Values

BUFFER_FULL

`public static final short` **`BUFFER_FULL`**

This reason code is used during a transaction to indicate that the commit buffer is full.

See Also:
Constant Field Values

INTERNAL_FAILURE

`public static final short` **`INTERNAL_FAILURE`**

This reason code is used during a transaction to indicate an internal Java Card runtime environment problem (fatal error).

See Also:

Constant Field Values

Constructor Detail

TransactionException

```
public TransactionException(short reason)
```

Constructs a TransactionException with the specified reason.

Method Detail

throwIt

```
public static void throwIt(short reason)
```

Throws an instance of `TransactionException` with the specified reason.

Throws:

`TransactionException` - always.

javacard.framework

Class SystemException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.framework.SystemException
```

All Implemented Interfaces:

Direct Known Subclasses:
SystemExceptionSubclass

---

public class **SystemException**

extends CardRuntimeException

`SystemException` represents a `JCSystem` class related exception.

See Also:

Field Summary

`static short`	**ILLEGAL_AID** This reason code is used by the `javacard.framework.Applet.register()` method to indicate that the input AID parameter is not a legal AID value.

`static short`	**ILLEGAL_TRANSIENT** This reason code is used to indicate that the request to create a transient object is not allowed in the current applet context.
`static short`	**ILLEGAL_USE** This reason code is used to indicate that the requested function is not allowed.
`static short`	**ILLEGAL_VALUE** This reason code is used to indicate that one or more input parameters is out of allowed bounds.
`static short`	**NO_RESOURCE** This reason code is used to indicate that there is insufficient resource in the Card for the request.
`static short`	**NO_TRANSIENT_SPACE** This reason code is used by the `makeTransient..()` methods to indicate that no room is available in volatile memory for the requested object.

Constructor Summary

SystemException`(short reason)`
Constructs a SystemException.

Method Summary

`static void`	**throwIt**`(short reason)` Throws an instance of `SystemException` with the specified reason.

Methods inherited from class javacard.framework.CardRuntimeException

`getReason, setReason`

Methods inherited from class java.lang.Throwable

`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object

```
clone, equals, finalize, getClass, hashCode, notify, notifyAll,
wait, wait, wait
```

Field Detail

ILLEGAL_VALUE

`public static final short` **`ILLEGAL_VALUE`**

This reason code is used to indicate that one or more input parameters is out of allowed bounds.

See Also:
Constant Field Values

NO_TRANSIENT_SPACE

`public static final short` **`NO_TRANSIENT_SPACE`**

This reason code is used by the `makeTransient..()` methods to indicate that no room is available in volatile memory for the requested object.

See Also:
Constant Field Values

ILLEGAL_TRANSIENT

`public static final short` **`ILLEGAL_TRANSIENT`**

This reason code is used to indicate that the request to create a transient object is not allowed in the current applet context. See *Java Card Runtime Environment Specification*, section 6.2.1 for details.

See Also:
Constant Field Values

ILLEGAL_AID

`public static final short` **`ILLEGAL_AID`**

This reason code is used by the `javacard.framework.Applet.register()` method to indicate that the input AID parameter is not a legal AID value.

See Also:
Constant Field Values

NO_RESOURCE

`public static final short` **`NO_RESOURCE`**

This reason code is used to indicate that there is insufficient resource in the Card for the request.

For example, the virtual machine for the Java Card platform may `throw` this exception reason when there is insufficient heap space to create a new instance.

See Also:
Constant Field Values

ILLEGAL_USE

`public static final short` **`ILLEGAL_USE`**

This reason code is used to indicate that the requested function is not allowed. For example, `JCSystem.requestObjectDeletion()` method throws this exception if the object deletion mechanism is not implemented.

See Also:
Constant Field Values

Constructor Detail

SystemException

```
public SystemException(short reason)
```

Constructs a SystemException.

Parameters:
`reason` - the reason for the exception.

Method Detail

throwIt

```
public static void throwIt(short reason)
                    throws SystemException
```

Throws an instance of `SystemException` with the specified reason.

Parameters:
`reason` - the reason for the exception.
Throws:
`SystemException` - always.

javacard.framework

Class PINException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.framework.PINException
```

All Implemented Interfaces:

Direct Known Subclasses:
PINExceptionSubclass

`public class` **`PINException`**

extends CardRuntimeException

`PINException` represents a `OwnerPIN` class access-related exception.

See Also:

Field Summary

`static short`	**ILLEGAL_VALUE** This reason code is used to indicate that one or more input parameters is out of allowed bounds.

Constructor Summary
`PINException(short reason)` Constructs a PINException.

Method Summary	
`static void`	`throwIt(short reason)` Throws an instance of `PINException` with the specified reason.

Methods inherited from class javacard.framework.CardRuntimeException
`getReason, setReason`

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Field Detail

ILLEGAL_VALUE

```
public static final short ILLEGAL_VALUE
```

This reason code is used to indicate that one or more input parameters is out of allowed bounds.

See Also:
Constant Field Values

Constructor Detail

PINException

```
public PINException(short reason)
```

Constructs a PINException. To conserve on resources use `throwIt()` to use the Java Card runtime-owned instance of this class.

Parameters:
`reason` - the reason for the exception.

Method Detail

throwIt

```
public static void throwIt(short reason)
```

Throws an instance of `PINException` with the specified reason.

Parameters:
`reason` - the reason for the exception.
Throws:
`PINException` - always.

javacard.framework

Class ISOException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.framework.ISOException
```

All Implemented Interfaces:

Direct Known Subclasses:
ISOExceptionSubclass

`public class` **`ISOException`**

extends CardRuntimeException

`ISOException` class encapsulates an ISO 7816-4 response status word as its `reason` code.

See Also:

Constructor Summary

`ISOException``(short sw)` Constructs an ISOException instance with the specified status word.

Method Summary

`static void`	**`throwIt`**`(short sw)` Throws an instance of the ISOException class with the specified status word.

Methods inherited from class javacard.framework.CardRuntimeException
`getReason, setReason`

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Constructor Detail

ISOException

`public ` **`ISOException`**`(short sw)`

Constructs an ISOException instance with the specified status word.

Parameters:

`sw` - the ISO 7816-4 defined status word

Method Detail

throwIt

`public static void ` **`throwIt`**`(short sw)`

Throws an instance of the ISOException class with the specified status word.

Parameters:

`sw` - ISO 7816-4 defined status word

Throws:

`ISOException` - always.

Constant Field Values

Contents

com.sun.*

com.sun.javacard.javax.smartcard.rmiclient.CardObjectFactory		
public static final byte	REF_FORMAT_CLASS	1
public static final byte	REF_FORMAT_INTERFACES	2
public static final byte	REF_FORMAT_NONE	0

com.sun.javacard.rmiclientlib.JCRMIConnect		
public static final byte	REF_WITH_CLASS_NAME	0
public static final byte	REF_WITH_INTERFACE_NAMES	16

javacard.framework.*

javacard.framework.APDUException		
public static final short	BAD_LENGTH	3
public static final short	BUFFER_BOUNDS	2

public static final short	ILLEGAL_USE	1
public static final short	IO_ERROR	4
public static final short	NO_T0_GETRESPONSE	170
public static final short	NO_T0_REISSUE	172
public static final short	T1_IFD_ABORT	171

javacard.framework.PINException		
public static final short	ILLEGAL_VALUE	1

javacard.framework.SystemException		
public static final short	ILLEGAL_AID	4
public static final short	ILLEGAL_TRANSIENT	3
public static final short	ILLEGAL_USE	6
public static final short	ILLEGAL_VALUE	1
public static final short	NO_RESOURCE	5
public static final short	NO_TRANSIENT_SPACE	2

javacard.framework.TransactionException		
public static final short	BUFFER_FULL	3
public static final short	IN_PROGRESS	1
public static final short	INTERNAL_FAILURE	4
public static final short	NOT_IN_PROGRESS	2

javacard.framework.service.ServiceException		
public static final short	CANNOT_ACCESS_IN_COMMAND	4
public static final short	CANNOT_ACCESS_OUT_COMMAND	5
public static final short	COMMAND_DATA_TOO_LONG	3

public static final short	COMMAND_IS_FINISHED	6
public static final short	DISPATCH_TABLE_FULL	2
public static final short	ILLEGAL_PARAM	1
public static final short	REMOTE_OBJECT_NOT_EXPORTED	7

javacard.security.*

javacard.security.CryptoException		
public static final short	ILLEGAL_USE	5
public static final short	ILLEGAL_VALUE	1
public static final short	INVALID_INIT	4
public static final short	NO_SUCH_ALGORITHM	3
public static final short	UNINITIALIZED_KEY	2

javacardx.biometry.*

javacardx.biometry.BioException		
public static final short	ILLEGAL_USE	5
public static final short	ILLEGAL_VALUE	1
public static final short	INVALID_DATA	2
public static final short	NO_SUCH_BIO_TEMPLATE	3
public static final short	NO_TEMPLATES_ENROLLED	4

javacardx.external.*

javacardx.external.ExternalException		
public static final short	INTERNAL_ERROR	3

public static final short	INVALID_PARAM	2
public static final short	NO_SUCH_SUBSYSTEM	1

javacardx.framework.*

javacardx.framework.tlv.TLVException		
public static final short	EMPTY_TAG	3
public static final short	EMPTY_TLV	4
public static final short	ILLEGAL_SIZE	2
public static final short	INSUFFICIENT_STORAGE	7
public static final short	INVALID_PARAM	1
public static final short	MALFORMED_TAG	5
public static final short	MALFORMED_TLV	6
public static final short	TAG_NUMBER_GREATER_THAN_32767	9
public static final short	TAG_SIZE_GREATER_THAN_127	8
public static final short	TLV_LENGTH_GREATER_THAN_32767	11
public static final short	TLV_SIZE_GREATER_THAN_32767	10

javacardx.framework.util.UtilException		
public static final short	ILLEGAL_VALUE	1
public static final short	TYPE_MISMATCHED	2

com.sun.javacard.rmiclientlib

Class JCRMIConnect

```
java.lang.Object
  └com.sun.javacard.rmiclientlib.JCRMIConnect
```

`public class` **`JCRMIConnect`**

extends java.lang.Object

The main class of the Java Card RMI client API. Provides functionality to select a card applet and to obtain an initial reference.

Field Summary

`protected byte`	**`format`** Format of the remote references.
`static byte`	**`REF_WITH_CLASS_NAME`** Constant used as the 2nd parameter to selectApplet method.
`static byte`	**`REF_WITH_INTERFACE_NAMES`** Constant used as the 2nd parameter to selectApplet method.
`protected byte[]`	**`selectResponse`** Response to the SELECT command is stored in this field.

Constructor Summary

`JCRMIConnect``(CardAccessor ca)` Creates a new instance of JCRMIConnect

Method Summary

`java.rmi.Remote`	**`getInitialReference`**`()` Parses the R-APDU which was returned during selecting an applet, returns a reference to an initial remote object
`byte[]`	**`selectApplet`**`(byte[] aid, byte format)` Selects an applet, requesting initial reference in the format specified by the 2nd parameter.

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Field Detail

selectResponse

`protected byte[] `**`selectResponse`**

Response to the SELECT command is stored in this field.

---

format

`protected byte `**`format`**

Format of the remote references.

REF_WITH_CLASS_NAME

`public static final byte `**`REF_WITH_CLASS_NAME`**

Constant used as the 2nd parameter to selectApplet method.

See Also:
[Constant Field Values]

REF_WITH_INTERFACE_NAMES

```
public static final byte REF_WITH_INTERFACE_NAMES
```

Constant used as the 2nd parameter to selectApplet method.

See Also:
Constant Field Values

Constructor Detail

JCRMIConnect

```
public JCRMIConnect(CardAccessor ca)
```

Creates a new instance of JCRMIConnect

Parameters:
`ca` - Implementaion of a CardAccessor

Method Detail

getInitialReference

```
public java.rmi.Remote getInitialReference()
                                    throws java.lang.Exception
```

Parses the R-APDU which was returned during selecting an applet, returns a reference to an initial remote object

Returns:
Stub or proxy for the initial remote object
Throws:

`java.lang.Exception` - If a problem occured

---

selectApplet

```
public byte[] selectApplet(byte[] aid,
                           byte format)
                    throws java.lang.Exception
```

Selects an applet, requesting initial reference in the format specified by the 2nd parameter. The R-APDU is returned and also stored internally for further processing by the getInitialReference () method.

Parameters:

`aid` - AID of the applet to be selected

`format` - Format of the remote references

Returns:

R-APDU

Throws:

`java.lang.Exception` - If a problem occured

---

com.sun.javacard.rmiclientlib

Class JCRemoteRefImpl

```
java.lang.Object
  └com.sun.javacard.rmiclientlib.JCRemoteRefImpl
```

All Implemented Interfaces:

java.io.Externalizable, java.lang.reflect.InvocationHandler, java.rmi.server.RemoteRef

---

`public class` **`JCRemoteRefImpl`**

extends java.lang.Object
implements java.rmi.server.RemoteRef, java.lang.reflect.InvocationHandler

Represents a reference to a card object. This class is a Java Card RMI implementation of the `RemoteRef` interface. It is used in conjunction with Java RMIC generated stubs or dynamically generated proxies for Java Card RMI method invocations.

See Also:

Field Summary

Fields inherited from interface java.rmi.server.RemoteRef
`packagePrefix`

Constructor Summary

JCRemoteRefImpl(short objID, java.lang.String a_string, CardAccessor ca, CardObjectFactory cOF) Creates new JCRemoteRefImpl

Method Summary	
void	**done**(java.rmi.server.RemoteCall remoteCall) **Deprecated.**
java.lang.String	**getRefClass**(java.io.ObjectOutput objectOutput) Unsupported operation.
java.lang.Object	**invoke**(java.lang.Object obj, java.lang.reflect.Method method, java.lang.Object[] params) This method is used by dynamically generated proxies.
void	**invoke**(java.rmi.server.RemoteCall remoteCall) **Deprecated.**
java.lang.Object	**invoke**(java.rmi.Remote remote, java.lang.reflect.Method method, java.lang.Object[] params, long unused) This method is used by rmic-generated stubs.
java.rmi.server.RemoteCall	**newCall**(java.rmi.server.RemoteObject remoteObject, java.rmi.server.Operation[] operation, int param, long param3) **Deprecated.**
void	**readExternal**(java.io.ObjectInput objectInput) Unsupported operation.
boolean	**remoteEquals**(java.rmi.server.RemoteRef remoteRef) Compares two remote objects for being identical.
int	**remoteHashCode**() Unsupported operation.
java.lang.String	**remoteToString**() String representation of remote object.
void	**writeExternal**(java.io.ObjectOutput objectOutput) Unsupported operation.

Methods inherited from class java.lang.Object

```
clone, equals, finalize, getClass, hashCode, notify, notifyAll,
toString, wait, wait, wait
```

Constructor Detail

JCRemoteRefImpl

```
public JCRemoteRefImpl(short objID,
                       java.lang.String a_string,
                       CardAccessor ca,
                       CardObjectFactory cOF)
```

Creates new `JCRemoteRefImpl`

Parameters:

`objID` - 2 byte Object ID from card remote reference descriptor
`a_string` - Anticollision string for the class of the remote object
`cA` - `CardAccessor`
`cOF` - `CardObjectFactory`

Method Detail

getRefClass

```
public java.lang.String getRefClass(java.io.
ObjectOutput objectOutput)
```

Unsupported operation.

Specified by:

`getRefClass` in interface `java.rmi.server.RemoteRef`

Parameters:

`objectOutput` -

---

invoke

```
public java.lang.Object invoke(java.rmi.Remote remote,
                               java.lang.reflect.Method method,
                               java.lang.Object[] params,
                               long unused)
                        throws java.io.IOException,
                               java.rmi.RemoteException,
                               java.lang.Exception
```

This method is used by rmic-generated stubs.

Specified by:
`invoke` in interface `java.rmi.server.RemoteRef`

Parameters:
`remote` - Reference to the stub - not used.
`method` - `java.reflect.Method` object containing information about the method to be invoked.
`params` - Array of parameters. Primitives are wrapped.
`unused` - rmic-generated hash of the method. Not used.

Returns:
The result returned from the card.

Throws:
`java.io.IOException` - If a communication error occured.
`java.rmi.RemoteException` - If an RMI error occured.
`java.lang.Exception` - Exception corresponding to the one that was thrown on the card.

---

remoteHashCode

```
public int remoteHashCode()
```

Unsupported operation.

Specified by:
`remoteHashCode` in interface `java.rmi.server.RemoteRef`

Returns:
A number which is the same for all objects.

---

remoteToString

```
public java.lang.String remoteToString()
```

String representation of remote object.

Specified by:
`remoteToString` in interface `java.rmi.server.RemoteRef`
Returns:
A `String` representation of the remote object.

---

readExternal

```
public void readExternal(java.io.ObjectInput objectInput)
                  throws java.io.IOException,
                         java.lang.ClassNotFoundException
```

Unsupported operation.

Specified by:
`readExternal` in interface `java.io.Externalizable`
Parameters:
`objectInput` -
Throws:
`java.io.IOException`
`java.lang.ClassNotFoundException`

---

writeExternal

```
public void writeExternal(java.io.ObjectOutput objectOutput)
                   throws java.io.IOException
```

Unsupported operation.

Specified by:
`writeExternal` in interface `java.io.Externalizable`
Parameters:

`objectOutput` -

Throws:

`java.io.IOException`

---

newCall

```
public java.rmi.server.RemoteCall newCall(java.rmi.server.
RemoteObject remoteObject,
                                           java.rmi.server.Operation
[] operation,
                                           int param,
                                           long param3)
                                    throws java.rmi.RemoteException
```

Deprecated.

Deprecated and not implemented

Specified by:

`newCall` in interface `java.rmi.server.RemoteRef`

Parameters:

`remoteObject` -
`operation` -
`param` -
`param3` -

Throws:

`java.rmi.RemoteException`

---

invoke

```
public void invoke(java.rmi.server.RemoteCall remoteCall)
            throws java.lang.Exception
```

Deprecated.

Deprecated and not implemented

Specified by:

`invoke` in interface `java.rmi.server.RemoteRef`

Parameters:

`remoteCall` -

Throws:

`java.lang.Exception`

---

remoteEquals

```
public boolean remoteEquals(java.rmi.server.RemoteRef remoteRef)
```

Compares two remote objects for being identical.

Specified by:

`remoteEquals` in interface `java.rmi.server.RemoteRef`

Parameters:

`remoteRef` - `RemoteRef` to the other remote object.

Returns:

`true` if corresponding remote objects are identical.

---

done

```
public void done(java.rmi.server.RemoteCall remoteCall)
          throws java.rmi.RemoteException
```

Deprecated.

Deprecated and not implemented

Specified by:

`done` in interface `java.rmi.server.RemoteRef`

Parameters:

`remoteCall` -

Throws:

`java.rmi.RemoteException`

---

invoke

```
public java.lang.Object invoke(java.lang.Object obj,
                               java.lang.reflect.Method method,
                               java.lang.Object[] params)
                        throws java.io.IOException,
                               java.rmi.RemoteException,
                               java.lang.Throwable
```

This method is used by dynamically generated proxies.

Specified by:

`invoke` in interface `java.lang.reflect.InvocationHandler`

Parameters:

`obj` - The reference to the Proxy - not used.

`method` - `Method` object containing information about the method.

`params` - Array of parameters for the method.

Returns:

The result returned from the card.

Throws:

`java.io.IOException` - If a communication error occured.

`java.rmi.RemoteException` - If an RMI error occured.

`java.lang.Throwable` - Exception corresponding to the one that was thrown on the card.

---

com.sun.javacard.rmiclientlib

Class JCCardProxyFactory

```
java.lang.Object
  └com.sun.javacard.javax.smartcard.rmiclient.CardObjectFactory
      └com.sun.javacard.rmiclientlib.JCCardProxyFactory
```

`public class` **`JCCardProxyFactory`**

extends CardObjectFactory

Processes the data returned from the card in the format defined for Java Card RMI. Object references must contain lists of interface names. Extends `CardObjectFactory`.

Field Summary

Fields inherited from class com.sun.javacard.javax.smartcard.rmiclient.CardObjectFactory
`REF_FORMAT_CLASS, REF_FORMAT_INTERFACES, REF_FORMAT_NONE`

Constructor Summary

`JCCardProxyFactory``()`   Constructor for the factory.

Method Summary

`protected java.rmi.Remote`	**`getRemoteObject`**`(byte[] buffer, int tagOffset, CardAccessor cardAccessor)` Creates the stub instance for object reference returned from the card, assuming the card returned a reference with list of interface names.
`byte`	**`getRemoteRefFormat`**`()` Returns constant `REF_FORMAT_INTERFACES` defined in class `CardObjectFactory`.

Methods inherited from class com.sun.javacard.javax.smartcard.rmiclient.CardObjectFactory
`getINSByte, getObject, setINSByte`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

JCCardProxyFactory

`public` **`JCCardProxyFactory`**`()`

Constructor for the factory.

Method Detail

getRemoteRefFormat

`public byte` **`getRemoteRefFormat`**`()`

Returns constant `REF_FORMAT_INTERFACES` defined in class `CardObjectFactory`.

Specified by:

`getRemoteRefFormat` in class `CardObjectFactory`

Returns:

The format constant.

---

getRemoteObject

```
protected java.rmi.Remote getRemoteObject(byte[] buffer,
                                          int tagOffset,
                                          CardAccessor cardAccessor)
                                   throws java.lang.Exception
```

Creates the stub instance for object reference returned from the card, assuming the card returned a reference with list of interface names.

Specified by:
: `getRemoteObject` in class `CardObjectFactory`

Parameters:
: `cardAccessor` -
: `buffer` - `APDU` buffer.
: `tagOffset` - Offset to tag.

Returns:
: The instance of the proxy.

Throws:
: `java.lang.Exception` - Thrown if the proxy instance cannot be instantiated

---

com.sun.javacard.rmiclientlib

Class JCCardObjectFactory

```
java.lang.Object
  └ com.sun.javacard.javax.smartcard.rmiclient.CardObjectFactory
      └ com.sun.javacard.rmiclientlib.JCCardObjectFactory
```

`public class` **`JCCardObjectFactory`**

extends CardObjectFactory

Processes the data returned from the card in the format defined for Java Card RMI. Object references must contain class names. Extends `CardObjectFactory`.

Field Summary

Fields inherited from class com.sun.javacard.javax.smartcard.rmiclient.CardObjectFactory
`REF_FORMAT_CLASS, REF_FORMAT_INTERFACES, REF_FORMAT_NONE`

Constructor Summary

`JCCardObjectFactory``()` The constructor.

Method Summary

protected java.rmi.Remote	**getRemoteObject**(byte[] buffer, int tagOffset, CardAccessor cardAccessor) Creates the stub instance for object reference returned from the card, assuming the card returned a reference with class name.
byte	**getRemoteRefFormat**() Returns constant `REF_FORMAT_CLASS` defined in class `CardObjectFactory`.

Methods inherited from class com.sun.javacard.javax.smartcard.rmiclient.CardObjectFactory
getINSByte, getObject, setINSByte

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

JCCardObjectFactory

```
public JCCardObjectFactory()
```

The constructor.

Method Detail

getRemoteObject

```
protected java.rmi.Remote getRemoteObject(byte[] buffer,
                                          int tagOffset,
                                          CardAccessor cardAccessor)
                                   throws java.lang.Exception
```

Creates the stub instance for object reference returned from the card, assuming the card returned a reference with class name.

Specified by:

`getRemoteObject` in class `CardObjectFactory`

Parameters:

`cardAccessor` - used to instantiate stubs of remote objects

`buffer` - APDU buffer

`tagOffset` - Offset to tag

Returns:

The resulting stub.

Throws:

`java.lang.Exception` - Failed to instantiate a stub

---

getRemoteRefFormat

```
public byte getRemoteRefFormat()
```

Returns constant `REF_FORMAT_CLASS` defined in class `CardObjectFactory`.

Specified by:

`getRemoteRefFormat` in class `CardObjectFactory`

Returns:

REF_FORMAT_CLASS value defined above

---

com.sun.javacard.clientlib

Interface CardAccessor

All Known Implementing Classes:
ApduIOCardAccessor

```
public interface CardAccessor
```

The `CardAccessor` interface represents a generic smartcard communication API. The interface based definition makes it platform and framework independent. This interface is used by Java Card RMI stubs to access the smart card.

Method Summary

`void`	**`closeCard()`** This method closes and resets the card
`byte[]`	**`exchangeAPDU(byte[] sendData)`** This method sends the specified data to the smartcard, waits for the response and returns the response in the return data.

Method Detail

exchangeAPDU

```
byte[] exchangeAPDU(byte[] sendData)
                    throws java.io.IOException
```

This method sends the specified data to the smartcard, waits for the response and returns the response in the return data. The input data is assumed to be formatted for ISO 7816-4 APDU communication as follows : [0] = CLA, [1]= INS, [2] = P1, [3]= P2, [4]=Lc, [4..]= command data. The response data is formatted for ISO 7816-4 APDU communication as follows : [0] = SW1, [1]= SW2, [2..]= response data.

Parameters:
: `sendData` - the ISO 7816-4 formatted command APDU data with 5 bytes of header followed by the command data.

Returns:
: responseData contains the response received from card with the 2 status bytes followed by the response data.

Throws:
: `java.io.IOException` - if communication error occurs

---

closeCard

```
void closeCard()
               throws java.lang.Exception
```

This method closes and resets the card

Throws:
: `java.lang.Exception` - Exception with a message about the problem.

---

com.sun.javacard.clientlib

Class ApduIOCardAccessor

```
java.lang.Object
  └com.sun.javacard.clientlib.ApduIOCardAccessor
```

All Implemented Interfaces:

CardAccessor

`public class` **`ApduIOCardAccessor`**

extends java.lang.Object
implements CardAccessor

Implementation of CardAccessor using ApduIO library

Field Summary	
`protected com.sun.javacard.apduio.CadClientInterface`	**`cad`** Reference to underlying ApduIO object.

Constructor Summary
`ApduIOCardAccessor``()` Creates a new instance of ApduIOCardAccessor

Method Summary

void	**closeCard**() Close and powerdown the card.
byte[]	**exchangeAPDU**(byte[] capdu) Implementation of exchangeAPDU method of CardAccessor interface

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

cad

```
protected com.sun.javacard.apduio.CadClientInterface cad
```

Reference to underlying ApduIO object.

Constructor Detail

ApduIOCardAccessor

```
public ApduIOCardAccessor()
                   throws java.lang.Exception
```

Creates a new instance of ApduIOCardAccessor

Throws:

`java.lang.Exception` - Thrown is config file jcclient.properties is not found in classpath or communication error happened.

Method Detail

exchangeAPDU

```
public byte[] exchangeAPDU(byte[] capdu)
```

```
                          throws java.io.IOException
```

Implementation of exchangeAPDU method of CardAccessor interface

Specified by:
: `exchangeAPDU` in interface `CardAccessor`

Parameters:
: `capdu` - byte array containing C-APDU

Returns:
: R-APDU in the format described in the javadoc for CardAccessor

Throws:
: `java.io.IOException` - In case of I/O error

---

closeCard

```
public void closeCard()
               throws java.lang.Exception
```

Close and powerdown the card.

Specified by:
: `closeCard` in interface `CardAccessor`

Throws:
: `java.lang.Exception` - Thrown if a problem occured

---

javacard.framework.service

Class ServiceException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.framework.service.ServiceException
```

All Implemented Interfaces:

Direct Known Subclasses:

ServiceExceptionSubclass

`public class` **`ServiceException`**

extends CardRuntimeException

`ServiceException` represents a service framework related exception.

See Also:

Field Summary

`static short`	**`CANNOT_ACCESS_IN_COMMAND`** This reason code is used to indicate that the command in the `APDU` object cannot be accessed for input processing.

static short	**CANNOT_ACCESS_OUT_COMMAND** This reason code is used to indicate that the command in the `APDU` object cannot be accessed for output processing.
static short	**COMMAND_DATA_TOO_LONG** This reason code is used to indicate that the incoming data for a command in the `APDU` object does not fit in the APDU buffer.
static short	**COMMAND_IS_FINISHED** This reason code is used to indicate that the command in the `APDU` object has been completely processed.
static short	**DISPATCH_TABLE_FULL** This reason code is used to indicate that a dispatch table is full
static short	**ILLEGAL_PARAM** This reason code is used to indicate that an input parameter is not allowed.
static short	**REMOTE_OBJECT_NOT_EXPORTED** This reason code is used by RMIService to indicate that the remote method returned an remote object which has not been exported.

Constructor Summary

ServiceException(short reason)
Constructs a `ServiceException`.

Method Summary

static void	**throwIt**(short reason) Throws an instance of `ServiceException` with the specified reason.

Methods inherited from class javacard.framework.CardRuntimeException

`getReason, setReason`

Methods inherited from class java.lang.Throwable

`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Field Detail

ILLEGAL_PARAM

`public static final short` **`ILLEGAL_PARAM`**

This reason code is used to indicate that an input parameter is not allowed.

See Also:
Constant Field Values

DISPATCH_TABLE_FULL

`public static final short` **`DISPATCH_TABLE_FULL`**

This reason code is used to indicate that a dispatch table is full

See Also:
Constant Field Values

COMMAND_DATA_TOO_LONG

`public static final short` **`COMMAND_DATA_TOO_LONG`**

This reason code is used to indicate that the incoming data for a command in the `APDU` object does not fit in the APDU buffer.

See Also:
Constant Field Values

CANNOT_ACCESS_IN_COMMAND

`public static final short` **`CANNOT_ACCESS_IN_COMMAND`**

This reason code is used to indicate that the command in the `APDU` object cannot be accessed for input processing.

See Also:
Constant Field Values

CANNOT_ACCESS_OUT_COMMAND

`public static final short` **`CANNOT_ACCESS_OUT_COMMAND`**

This reason code is used to indicate that the command in the `APDU` object cannot be accessed for output processing.

See Also:
Constant Field Values

COMMAND_IS_FINISHED

`public static final short` **`COMMAND_IS_FINISHED`**

This reason code is used to indicate that the command in the `APDU` object has been completely processed.

See Also:
Constant Field Values

REMOTE_OBJECT_NOT_EXPORTED

```
public static final short REMOTE_OBJECT_NOT_EXPORTED
```

This reason code is used by RMIService to indicate that the remote method returned an remote object which has not been exported.

See Also:

Constant Field Values

Constructor Detail

ServiceException

```
public ServiceException(short reason)
```

Constructs a `ServiceException`.

Parameters:

`reason` - the reason for the exception.

Method Detail

throwIt

```
public static void throwIt(short reason)
                    throws ServiceException
```

Throws an instance of `ServiceException` with the specified reason.

Parameters:

`reason` - the reason for the exception.

Throws:

`ServiceException` - always.

Package javacard.framework.service

Provides Java Card API exceptions that may be re-thrown on the client.

See:
Description

Exception Summary	
ServiceException	`ServiceException` represents a service framework related exception.

Package javacard.framework.service Description

Provides Java Card API exceptions that may be re-thrown on the client.

Hierarchy For Package javacard.framework.service

Package Hierarchies:
All Packages

Class Hierarchy

- java.lang.Object
  - java.lang.Throwable
    - java.lang.Exception
      - java.lang.RuntimeException
        - javacard.framework.**CardRuntimeException**
          - javacard.framework.service.**ServiceException**

Hierarchy For Package javacard.security

Package Hierarchies:
All Packages

Class Hierarchy

- java.lang.Object
  - java.lang.Throwable
    - java.lang.Exception
      - java.lang.RuntimeException
        - javacard.framework.**CardRuntimeException**
          - javacard.security.**CryptoException**

Package javacard.security

Provides Java Card API exceptions that may be re-thrown on the client.

See:
Description

Exception Summary	
CryptoException	`CryptoException` represents a cryptography-related exception.

Package javacard.security Description

Provides Java Card platform exceptions that may be re-thrown on the client.

Package javacardx.biometry

Provides Java Card API exceptions that may be re-thrown on the client.

See:
Description

Exception Summary	
BioException	The `BioException` class encapsulates specific exceptions which can be thrown by the methods of the `javacardx.biometry` package in case of error.

Package javacardx.biometry Description

Extension package that contains functionality for implementing a biometric framework on the Java Card platform. The platform must support this optional package only if biometry support is included in the implementation.

The `javacardx.biometry` package contains classes and interfaces which can be used to build a biometric server application. These classes also enable a client application on the card to obtain biometric services from the biometric server application.

Hierarchy For Package javacardx.biometry

Package Hierarchies:

All Packages

Class Hierarchy

- java.lang.Object
  - java.lang.Throwable
    - java.lang.Exception
      - java.lang.RuntimeException
        - javacard.framework.**CardRuntimeException**
          - javacardx.biometry.**BioException**

Hierarchy For Package javacardx.external

Package Hierarchies:

All Packages

Class Hierarchy

- java.lang.Object
  - java.lang.Throwable
    - java.lang.Exception
      - java.lang.RuntimeException
        - javacard.framework.**CardRuntimeException**
          - javacardx.external.**ExternalException**

Package javacardx.external

Provides Java Card API exceptions that may be re-thrown on the client.

See:
Description

Exception Summary	
ExternalException	`ExternalException` represents an external subsystem related exception.

Package javacardx.external Description

Extension package that provides mechanisms to access memory subsystems which are not directly addressable by the Java Card runtime environment (Java Card RE) on the Java Card platform. The platform must support this optional package if an external memory access feature is included in the implementation.

The `javacardx.external` package contains the `Memory` class and the `MemoryAccess` interface. The `Memory` class provides a factory method for creating an instance of the `MemoryAccess` interface suitable for accessing supported memory subsystems.

Package javacardx.framework.tlv

Provides Java Card API exceptions that may be re-thrown on the client.

See:
 Description

Exception Summary	
TLVException	`TLVException` represents a TLV-related exception.

Package javacardx.framework.tlv Description

Extension package that contains functionality, for managing storage for BER TLV formatted data, based on the ASN.1 BER encoding rules of ISO/IEC 8825-1:2002, as well as parsing and editing BER TLV formatted data in I/O buffers.

The `javacardx.framework.tlv` package contains the `BERTag` abstract class, and its concrete subclasses `PrimitiveBERTag` and `ConstructedBERTag`. These classes encapsulate the BER tag functionality.

The `javacardx.framework.tlv` package also contains the `BERTLV` abstract class, and its concrete subclasses `PrimitiveBERTLV` and `ConstructedBERTLV`. These classes encapsulate the BER TLV functionality.

Hierarchy For Package javacardx.framework.tlv

Package Hierarchies:

All Packages

Class Hierarchy

- java.lang.Object
  - java.lang.Throwable
    - java.lang.Exception
      - java.lang.RuntimeException
        - javacard.framework.**CardRuntimeException**
          - javacardx.framework.tlv.**TLVException**

Hierarchy For Package javacardx.framework.util

Package Hierarchies:
All Packages

Class Hierarchy

- java.lang.Object
  - java.lang.Throwable
    - java.lang.Exception
      - java.lang.RuntimeException
        - javacard.framework.**CardRuntimeException**
          - javacardx.framework.util.**UtilException**

Package javacardx.framework.util

Provides Java Card API exceptions that may be re-thrown on the client.

See:
Description

Exception Summary	
UtilException	`UtilException` represents a util related exception.

Package javacardx.framework.util Description

Extension package that contains common utility functions for manipulating arrays of primitive components - `byte`, `short` or `int`. If the `int` primitive type is supported by the platform, the `intx` sub-package must be included.

The `javacardx.framework.util` package contains the `ArrayLogic` class. The `ArrayLogic` class provides methods for functionality similar to that of the `javacard.framework.Util` class but with generic `Object` component equivalents.

javacardx.framework.util

Class UtilException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacardx.framework.util.UtilException
```

All Implemented Interfaces:

Direct Known Subclasses:

UtilExceptionSubclass

`public class` **`UtilException`**

extends CardRuntimeException

`UtilException` represents a util related exception.

The API classes throw Java Card runtime environment-owned instances of `UtilException`.

Java Card runtime environment-owned instances of exception classes are temporary Java Card runtime environment Entry Point Objects and can be accessed from any applet context. References to these temporary objects cannot be stored in class variables, instance variables, or array components.

See Also:

Field Summary	
`static short`	**`ILLEGAL_VALUE`** This reason code is used to indicate that one or more input parameters is not the correct type or is out of allowed bounds.
`static short`	**`TYPE_MISMATCHED`** This reason code is used to indicate that input parameters are not the same type.

Constructor Summary
`UtilException``(short reason)` Constructs a `UtilException` with the specified reason.

Method Summary	
`static void`	**`throwIt`**`(short reason)` Throws the Java Card runtime environment-owned instance of `UtilException` with the specified reason.

Methods inherited from class javacard.framework.CardRuntimeException
`getReason, setReason`

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Field Detail

ILLEGAL_VALUE

`public static final short` **`ILLEGAL_VALUE`**

This reason code is used to indicate that one or more input parameters is not the correct type or is out of allowed bounds.

See Also:
Constant Field Values

TYPE_MISMATCHED

`public static final short` **`TYPE_MISMATCHED`**

This reason code is used to indicate that input parameters are not the same type.

See Also:
Constant Field Values

Constructor Detail

UtilException

`public` **`UtilException`**`(short reason)`

Constructs a `UtilException` with the specified reason. To conserve on resources use `throwIt()` to use the Java Card runtime environment-owned instance of this class.

Parameters:
`reason` - the reason for the exception

Method Detail

throwIt

`public static void` **`throwIt`**`(short reason)`

Throws the Java Card runtime environment-owned instance of `UtilException` with the specified reason.

Java Card runtime environment-owned instances of exception classes are temporary Java Card runtime environment Entry Point Objects and can be accessed from any applet context. References to these temporary objects cannot be stored in class variables or instance variables or array components. See *Runtime Environment Specification for the Java Card Platform*, section 6.2.1 for details.

Parameters:

`reason` - the reason for the exception

Throws:

`UtilException` - always

javacardx.framework.tlv

Class TLVException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacardx.framework.tlv.TLVException
```

All Implemented Interfaces:

Direct Known Subclasses:

TLVExceptionSubclass

`public class` **`TLVException`**

extends CardRuntimeException

`TLVException` represents a TLV-related exception.

The API classes throw Java Card runtime environment-owned instances of `TLVException`.

Java Card runtime environment-owned instances of exception classes are temporary Java Card runtime environment Entry Point Objects and can be accessed from any applet context. References to these temporary objects cannot be stored in class variables, instance variables, or array components.

See Also:

Field Summary	
`static short`	**EMPTY_TAG** This reason code is used to indicate that the Tag object is empty
`static short`	**EMPTY_TLV** This reason code is used to indicate that the TLV object is empty
`static short`	**ILLEGAL_SIZE** This reason code is used to indicate that the size of a TLV or Tag representation in the input parameter is greater than the supported size or will result in in a TLV struture of greater than supported size
`static short`	**INSUFFICIENT_STORAGE** This reason code is used to indicate that the configured storage capacity of the object will be exceeded
`static short`	**INVALID_PARAM** This reason code is used to indicate that one or more input parameters is invalid.
`static short`	**MALFORMED_TAG** This reason code is used to indicate that the tag representation is not a well-formed BER Tag
`static short`	**MALFORMED_TLV** This reason code is used to indicate that the TLV representation is not a well-formed BER TLV
`static short`	**TAG_NUMBER_GREATER_THAN_32767** This reason code is used to indicate that the tag number value greater than 32767
`static short`	**TAG_SIZE_GREATER_THAN_127** This reason code is used to indicate that the size of the tag representation is greater than 127 bytes
`static short`	**TLV_LENGTH_GREATER_THAN_32767** This reason code is used to indicate that the Length component value in the TLV is greater than 32767
`static short`	**TLV_SIZE_GREATER_THAN_32767** This reason code is used to indicate that the TLV requires more that 32767 bytes to represent

Constructor Summary
TLVException(short reason) Constructs a `TLVException` with the specified reason.

Method Summary	
static void	**throwIt**(short reason) Throws the Java Card runtime environment-owned instance of `TLVException` with the specified reason.

Methods inherited from class javacard.framework.CardRuntimeException
getReason, setReason

Methods inherited from class java.lang.Throwable
fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Field Detail

INVALID_PARAM

```
public static final short INVALID_PARAM
```

This reason code is used to indicate that one or more input parameters is invalid.

See Also:

Constant Field Values

ILLEGAL_SIZE

```
public static final short ILLEGAL_SIZE
```

This reason code is used to indicate that the size of a TLV or Tag representation in the input parameter is greater than the supported size or will result in in a TLV struture of greater than supported size

See Also:
Constant Field Values

EMPTY_TAG

```
public static final short EMPTY_TAG
```

This reason code is used to indicate that the Tag object is empty

See Also:
Constant Field Values

EMPTY_TLV

```
public static final short EMPTY_TLV
```

This reason code is used to indicate that the TLV object is empty

See Also:
Constant Field Values

MALFORMED_TAG

```
public static final short MALFORMED_TAG
```

This reason code is used to indicate that the tag representation is not a well-formed BER Tag

See Also:
Constant Field Values

MALFORMED_TLV

public static final short **MALFORMED_TLV**

This reason code is used to indicate that the TLV representation is not a well-formed BER TLV

See Also:
Constant Field Values

INSUFFICIENT_STORAGE

public static final short **INSUFFICIENT_STORAGE**

This reason code is used to indicate that the configured storage capacity of the object will be exceeded

See Also:
Constant Field Values

TAG_SIZE_GREATER_THAN_127

public static final short **TAG_SIZE_GREATER_THAN_127**

This reason code is used to indicate that the size of the tag representation is greater than 127 bytes

See Also:
Constant Field Values

TAG_NUMBER_GREATER_THAN_32767

`public static final short` **`TAG_NUMBER_GREATER_THAN_32767`**

This reason code is used to indicate that the tag number value greater than 32767

See Also:
Constant Field Values

TLV_SIZE_GREATER_THAN_32767

`public static final short` **`TLV_SIZE_GREATER_THAN_32767`**

This reason code is used to indicate that the TLV requires more that 32767 bytes to represent

See Also:
Constant Field Values

TLV_LENGTH_GREATER_THAN_32767

`public static final short` **`TLV_LENGTH_GREATER_THAN_32767`**

This reason code is used to indicate that the Length component value in the TLV is greater than 32767

See Also:
Constant Field Values

Constructor Detail

TLVException

`public` **`TLVException`**`(short reason)`

Constructs a `TLVException` with the specified reason. To conserve on resources use `throwIt()` to use the Java Card runtime environment-owned instance of this class.

Parameters:

`reason` - the reason for the exception

Method Detail

throwIt

```
public static void throwIt(short reason)
```

Throws the Java Card runtime environment-owned instance of `TLVException` with the specified reason.

Java Card runtime environment-owned instances of exception classes are temporary Java Card runtime environment Entry Point Objects and can be accessed from any applet context. References to these temporary objects cannot be stored in class variables or instance variables or array components. See *Runtime Environment Specification for the Java Card Platform*, section 6.2.1 for details.

Parameters:

`reason` - the reason for the exception

Throws:

`TLVException` - always

---

javacardx.external

Class ExternalException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacardx.external.ExternalException
```

All Implemented Interfaces:

Direct Known Subclasses:

ExternalExceptionSubclass

`public class` **`ExternalException`**

extends CardRuntimeException

`ExternalException` represents an external subsystem related exception.

The API classes throw Java Card runtime environment-owned instances of `ExternalException`.

Java Card runtime environment-owned instances of exception classes are temporary Java Card runtime environment Entry Point Objects and can be accessed from any applet context. References to these temporary objects cannot be stored in class variables or instance variables or array components.

Since:

2.2.2

See Also:

Field Summary

static short	**INTERNAL_ERROR** This reason code is used to indicate that an unrecoverable external access error occurred.
static short	**INVALID_PARAM** This reason code is used to indicate that an input parameter is invalid.
static short	**NO_SUCH_SUBSYSTEM** This reason code is used to indicate that specified external subsystem is not available.

Constructor Summary

ExternalException(short reason) Constructs a `ExternalException` with the specified reason.

Method Summary

static void	**throwIt**(short reason) Throws the Java Card runtime environment-owned instance of `ExternalException` with the specified reason.

Methods inherited from class javacard.framework.CardRuntimeException
`getReason, setReason`

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Field Detail

NO_SUCH_SUBSYSTEM

```
public static final short NO_SUCH_SUBSYSTEM
```

This reason code is used to indicate that specified external subsystem is not available.

See Also:
Constant Field Values

INVALID_PARAM

```
public static final short INVALID_PARAM
```

This reason code is used to indicate that an input parameter is invalid.

See Also:
Constant Field Values

INTERNAL_ERROR

```
public static final short INTERNAL_ERROR
```

This reason code is used to indicate that an unrecoverable external access error occurred.

See Also:
Constant Field Values

Constructor Detail

ExternalException

```
public ExternalException(short reason)
```

Constructs a `ExternalException` with the specified reason. To conserve on resources use `throwIt()` to use the Java Card runtime environment-owned instance of this class.

Parameters:
`reason` - the reason for the exception

Method Detail

throwIt

`public static void throwIt(short reason)`

Throws the Java Card runtime environment-owned instance of `ExternalException` with the specified reason.

Java Card runtime environment-owned instances of exception classes are temporary Java Card runtime environment Entry Point Objects and can be accessed from any applet context. References to these temporary objects cannot be stored in class variables or instance variables or array components. See *Runtime Environment Specification for the Java Card Platform*, section 6.2.1 for details.

Parameters:
`reason` - the reason for the exception
Throws:
`ExternalException` - always

---

javacardx.biometry

Class BioException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacardx.biometry.BioException
```

All Implemented Interfaces:

Direct Known Subclasses:

BioExceptionSubclass

`public class` **`BioException`**

extends CardRuntimeException

The `BioException` class encapsulates specific exceptions which can be thrown by the methods of the `javacardx.biometry` package in case of error.

See Also:

Field Summary	
`static short`	**`ILLEGAL_USE`** This reason code is used to indicate that the method should not be invoked based on the current state of the card.

`static short`	**ILLEGAL_VALUE** This reason code is used to indicate that one or more input parameters is out of allowed bounds.
`static short`	**INVALID_DATA** This reason code is used to indicate that the data the system encountered is illegible.
`static short`	**NO_SUCH_BIO_TEMPLATE** This reason code is used to indicate that the provided bio template type is not supported by the template builder.
`static short`	**NO_TEMPLATES_ENROLLED** This reason code is used to indicate that no reference template is available for matching, or that the reference template is uninitialized.

Constructor Summary
BioException`(short reason)` Construct a new biometric exception using a provided reason code.

Method Summary	
`static void`	**throwIt**`(short reason)` Throws the Java Card runtime environment owned instance of BioException with the specified reason.

Methods inherited from class javacard.framework.CardRuntimeException
`getReason, setReason`

Methods inherited from class java.lang.Throwable
`fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString`

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait`

Field Detail

ILLEGAL_VALUE

`public static final short` **`ILLEGAL_VALUE`**

This reason code is used to indicate that one or more input parameters is out of allowed bounds.

See Also:
Constant Field Values

INVALID_DATA

`public static final short` **`INVALID_DATA`**

This reason code is used to indicate that the data the system encountered is illegible.

See Also:
Constant Field Values

NO_SUCH_BIO_TEMPLATE

`public static final short` **`NO_SUCH_BIO_TEMPLATE`**

This reason code is used to indicate that the provided bio template type is not supported by the template builder.

See Also:
Constant Field Values

NO_TEMPLATES_ENROLLED

`public static final short` **`NO_TEMPLATES_ENROLLED`**

This reason code is used to indicate that no reference template is available for matching, or that the reference template is uninitialized.

See Also:
Constant Field Values

ILLEGAL_USE

`public static final short` **`ILLEGAL_USE`**

This reason code is used to indicate that the method should not be invoked based on the current state of the card.

See Also:
Constant Field Values

Constructor Detail

BioException

`public` **`BioException`**`(short reason)`

Construct a new biometric exception using a provided reason code. To conserve on resources use `throwIt()` to use the Java Card runtime environment instance of this class.

Parameters:
`reason` - the reason code for this exception.

Method Detail

throwIt

```
public static void throwIt(short reason)
                    throws BioException
```

Throws the Java Card runtime environment owned instance of BioException with the specified reason. Java Card runtime environment owned instances of exception classes are temporary Java Card runtime environment Entry Point Objects and can be accessed from any applet context. References to these objects cannot be stored in class variables or instance variables or array components.

Parameters:

`reason` - the reason for the exception.

Throws:

`BioException` - always.

---

javacard.security

Class CryptoException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └java.lang.RuntimeException
              └javacard.framework.CardRuntimeException
                  └javacard.security.CryptoException
```

All Implemented Interfaces:

Direct Known Subclasses:
CryptoExceptionSubclass

`public class` **`CryptoException`**

extends CardRuntimeException

`CryptoException` represents a cryptography-related exception.

See Also:

Field Summary	
`static short`	**ILLEGAL_USE** This reason code is used to indicate that the signature or cipher algorithm does not pad the incoming message and the input message is not block aligned.

static short	**ILLEGAL_VALUE** This reason code is used to indicate that one or more input parameters is out of allowed bounds.
static short	**INVALID_INIT** This reason code is used to indicate that the signature or cipher object has not been correctly initialized for the requested operation.
static short	**NO_SUCH_ALGORITHM** This reason code is used to indicate that the requested algorithm or key type is not supported.
static short	**UNINITIALIZED_KEY** This reason code is used to indicate that the key is uninitialized.

Constructor Summary

CryptoException(short reason) Constructs a `CryptoException` with the specified reason.

Method Summary

static void	**throwIt**(short reason) Throws an instance of `CryptoException` with the specified reason.

Methods inherited from class javacard.framework.CardRuntimeException
getReason, setReason

Methods inherited from class java.lang.Throwable
fillInStackTrace, getCause, getLocalizedMessage, getMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Field Detail

ILLEGAL_VALUE

```
public static final short ILLEGAL_VALUE
```

This reason code is used to indicate that one or more input parameters is out of allowed bounds.

See Also:
Constant Field Values

UNINITIALIZED_KEY

```
public static final short UNINITIALIZED_KEY
```

This reason code is used to indicate that the key is uninitialized.

See Also:
Constant Field Values

NO_SUCH_ALGORITHM

```
public static final short NO_SUCH_ALGORITHM
```

This reason code is used to indicate that the requested algorithm or key type is not supported.

See Also:
Constant Field Values

INVALID_INIT

```
public static final short INVALID_INIT
```

This reason code is used to indicate that the signature or cipher object has not been correctly initialized for the requested operation.

See Also:
Constant Field Values

ILLEGAL_USE

```
public static final short ILLEGAL_USE
```

This reason code is used to indicate that the signature or cipher algorithm does not pad the incoming message and the input message is not block aligned.

See Also:
Constant Field Values

Constructor Detail

CryptoException

```
public CryptoException(short reason)
```

Constructs a `CryptoException` with the specified reason. To conserve on resources use `throwIt()` to use the JCRE owned instance of this class.

Parameters:
`reason` - the reason for the exception.

Method Detail

throwIt

```
public static void throwIt(short reason)
```

Throws an instance of `CryptoException` with the specified reason.

Parameters:

`reason` - the reason for the exception.

Throws:

`CryptoException` - always.

com.sun.javacard.clientlib

Interfaces

CardAccessor

Classes

ApduIOCardAccessor

com.sun.javacard.javax.smartcard.rmiclient

Classes

CardObjectFactory

Exceptions

APDUExceptionSubclass
BioExceptionSubclass
CardExceptionSubclass
CardRuntimeExceptionSubclass
CryptoExceptionSubclass
ExternalExceptionSubclass
ISOExceptionSubclass
PINExceptionSubclass
ServiceExceptionSubclass
SystemExceptionSubclass
TLVExceptionSubclass
TransactionExceptionSubclass
UserExceptionSubclass
UtilExceptionSubclass

com.sun.javacard.rmiclientlib

Classes

JCCardObjectFactory

JCCardProxyFactory

JCRemoteRefImpl

JCRMIConnect

javacard.framework

Exceptions

APDUException
CardException
CardRuntimeException
ISOException
PINException
SystemException
TransactionException
UserException

javacard.framework.service

Exceptions

ServiceException

javacard.security

Exceptions

CryptoException

[javacardx.biometry]

Exceptions

[BioException]

javacardx.external

Exceptions

ExternalException

javacardx.framework.tlv

Exceptions

TLVException

javacardx.framework.util

Exceptions

UtilException