Java Card 2.0 API

Other Packages

- package javacard.framework
- package javacardx.crypto
- package javacardx.framework

Class Hierarchy

- class java.lang.Object
  - class javacard.framework.AID
  - class javacard.framework.APDU
  - class javacard.framework.Applet
  - class javacardx.crypto.AsymKey
    - class javacardx.crypto.RSA_Key
  - class javacardx.framework.File
    - class javacardx.framework.DedicatedFile
      - class javacardx.framework.FileSystem
    - class javacardx.framework.ElementaryFile
      - class javacardx.framework.LinearVariableFile
        - class javacardx.framework.LinearFixedFile
          - class javacardx.framework.CyclicFile
      - class javacardx.framework.TransparentFile
  - class javacardx.crypto.Key
    - class javacardx.crypto.RSA_CRT_Key
    - class javacardx.crypto.SymKey
      - class javacardx.crypto.DES_CBC_Key
      - class javacardx.crypto.DES_Key
  - class javacardx.crypto.MessageDigest
    - class javacardx.crypto.Sha1MessageDigest
  - class javacard.framework.PIN
  - class javacardx.crypto.RandomData
  - class javacard.framework.System
  - class java.lang.Throwable (implements java.io.Serializable)
    - class java.lang.Exception
      - class javacard.framework.CardException
        - class javacard.framework.UserException
      - class java.lang.RuntimeException
        - class javacard.framework.CardRuntimeException
          - class javacard.framework.ApduException
          - class javacard.framework.FileIoException
          - class javacard.framework.PINException
          - class javacard.framework.SystemException
        - class javacardx.crypto.CryptoException
  - class javacard.framework.Util

All Packages Class Hierarchy

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Index of all Fields and Methods

A

abortTransaction(). Static method in class javacard.framework.System
    Aborts the atomic transaction.
ACCESS_READ. Static variable in class javacardx.framework.File
ACCESS_WRITE. Static variable in class javacardx.framework.File
addChildFile(File). Method in class javacardx.framework.DedicatedFile
    Add (append) a new chile file to the file.
addRecord(byte[]). Method in class javacardx.framework.CyclicFile
    Add (append) a new record.
addRecord(byte[]). Method in class javacardx.framework.LinearFixedFile
    Add (append) a new record to the file.
addRecord(byte[]). Method in class javacardx.framework.LinearVariableFile
    Add (append) a new record to the file.
addRecord(short). Method in class javacardx.framework.CyclicFile
    Add (append) a new record.
addRecord(short). Method in class javacardx.framework.LinearFixedFile
    Add (append) a new record to the file.
addRecord(short). Method in class javacardx.framework.LinearVariableFile
    Add (append) a new record to the file.
ALLOW_ANY. Static variable in class javacardx.framework.File
ALLOW_AUTH1. Static variable in class javacardx.framework.File
ALLOW_AUTH2. Static variable in class javacardx.framework.File
ALLOW_NONE. Static variable in class javacardx.framework.File
ApduException(byte). Constructor for class javacard.framework.ApduException
    Constructs an ApduException.
appendRecord(APDU). Method in class javacardx.framework.FileSystem
    Handles APPEND RECORD command APDU as specified by ISO 7816-4.
Applet(). Constructor for class javacard.framework.Applet
arrayCompare(byte[], short, byte[], short, short). Static method in class javacard.framework.Util
    Compares an array from the specified source array, beginning at the specified position, with the specified position of the destination array.
arrayCopy(byte[], short, byte[], short, short). Static method in class javacard.framework.Util
    Copies an array from the specified source array, beginning at the specified position, to the specified position of the destination array.
AsymKey(). Constructor for class javacardx.crypto.AsymKey

B

BAD_LENGTH. Static variable in class javacard.framework.ApduException
beginTransaction(). Static method in class javacard.framework.System
    Begins an atomic transaction.
bitLength(). Method in class javacardx.crypto.Key
    the length of the key in bits
BUFFER_BOUNDS. Static variable in class javacard.framework.ApduException
BUFFER_BOUNDS. Static variable in class javacard.framework.PINException

C

CardException(). Constructor for class javacard.framework.CardException
    Constructs a Card exception instance To conserve on resources use `getInstance()` to re-use the System instance of this class.
CardRuntimeException(). Constructor for class javacard.framework.CardRuntimeException
    Constructs a Card Runtime exception instance.
check(byte[], short, byte). Method in class javacard.framework.PIN
    Compares `pin` against the PIN value.
clearIV(). Method in class javacardx.crypto.DES_CBC_Key
    Clears the initialization vector used in CBC mode DES operations.
clearKey(). Method in class javacardx.crypto.Key
    clears the key and sets its initialized state to false.
commitTransaction(). Static method in class javacard.framework.System
    Commits an atomic transaction.
create(APDU). Method in class javacardx.framework.FileSystem
    Handles create APDUs.
CryptoException(). Constructor for class javacardx.crypto.CryptoException
CyclicFile(short, byte, byte). Constructor for class javacardx.framework.CyclicFile
    Constructor.

D

DATAFILE_FID. Static variable in class javacardx.framework.FileSystem
decrypt(byte[], short, short, byte[], short). Method in class javacardx.crypto.DES_CBC_Key
    Decrypts using DES in CBC mode.
decrypt(byte[], short, short, byte[], short). Method in class javacardx.crypto.DES_Key
    Decrypts using DES in ECB mode.
decrypt(byte[], short, short, byte[], short). Method in class javacardx.crypto.SymKey
    Decrypts data using this key.
DedicatedFile(short, byte[], byte). Constructor for class javacardx.framework.DedicatedFile
    Constructor.
DES_CBC_Key(byte). Constructor for class javacardx.crypto.DES_CBC_Key
DES_Key(byte). Constructor for class javacardx.crypto.DES_Key
DIRECTION_FIRST. Static variable in class javacardx.framework.LinearVariableFile
DIRECTION_LAST. Static variable in class javacardx.framework.LinearVariableFile

DIRECTION_NEXT. Static variable in class javacardx.framework.LinearVariableFile
DIRECTION_PREV. Static variable in class javacardx.framework.LinearVariableFile

---

E

ElementaryFile(). Constructor for class javacardx.framework.ElementaryFile
encrypt(byte[], short, short, byte[], short). Method in class javacardx.crypto.DES_CBC_Key
Encrypts using DES in CBC mode.
encrypt(byte[], short, short, byte[], short). Method in class javacardx.crypto.DES_Key
Encrypts using DES in ECB mode.
encrypt(byte[], short, short, byte[], short). Method in class javacardx.crypto.SymKey
Encrypts data using this key.
eraseBinary(APDU). Method in class javacardx.framework.FileSystem
Handles ERASE BINARY command APDU as specified by ISO 7816-4.

F

File(). Constructor for class javacardx.framework.File
FileIoException(byte). Constructor for class javacard.framework.FileIoException
FileSystem(byte). Constructor for class javacardx.framework.FileSystem
Constructs an instance of an ISO 7816-4 file system.
FIND_ANY. Static variable in class javacardx.framework.FileSystem
FIND_CHILD. Static variable in class javacardx.framework.FileSystem
FIND_CHILD_DF. Static variable in class javacardx.framework.FileSystem
FIND_CHILD_EF. Static variable in class javacardx.framework.FileSystem
FIND_PARENT. Static variable in class javacardx.framework.FileSystem
findDedicatedFile(byte[], short, byte). Method in class javacardx.framework.FileSystem
Find the DF with the specified name.
findElementaryFile(byte). Method in class javacardx.framework.DedicatedFile
Under this DF, find the EF with the specified SFI.
findFile(byte, short). Method in class javacardx.framework.FileSystem
According to findType, find the file with the specified FID.

G

generateData(byte[], short, short). Static method in class javacardx.crypto.RandomData
generates random data.
generateDigest(byte[], short, short, byte[], short, short). Static method in class javacardx.crypto.MessageDigest
generates a hash of the input data.
generateDigest(byte[], short, short, byte[], short, short). Static method in class javacardx.crypto.Sha1MessageDigest
generates a hash of the input data using the SHA1 algorithm.
generateMAC(byte[], short, short, byte[], short, byte). Method in class javacardx.crypto.DES_CBC_Key
Generates a MAC using DES in CBC mode.

Java Card 2.0 API, copyright Sun Microsystems Inc

getAID(). Static method in class javacard.framework.System
Returns the unique Applet Identifier (AID) object associated with the current applet.
getArray(byte[], short). Method in class javacard.framework.AID
Called to obtain a copy of the byte array within AID object.
getAuthFlag(byte). Method in class javacardx.framework.FileSystem
Get authorization flag.
getBuffer(). Method in class javacard.framework.APDU
Returns the APDU buffer byte array.
getChildFile(byte). Method in class javacardx.framework.DedicatedFile
Get the File object for the specified child file.
getCurrentDedicatedFile(). Method in class javacardx.framework.FileSystem
Get current DF.
getCurrentElementaryFile(). Method in class javacardx.framework.FileSystem
Get current EF.
getCurrentRecNum(). Method in class javacardx.framework.FileSystem
Get current record number.
getData(). Method in class javacardx.framework.TransparentFile
Gets the byte array containing the data for this file.
getData(APDU). Method in class javacardx.framework.FileSystem
Handles GET DATA command APDU as specified by ISO 7816-4.
getException(byte). Static method in class javacard.framework.ApduException
Returns the system instance of ApduException
getException(byte). Static method in class javacard.framework.FileIoException
Returns the System instance of FileIoException
getException(byte). Static method in class javacard.framework.PINException
Returns the system instance of PINException
getException(byte). Static method in class javacard.framework.SystemException
Returns the system instance of SystemException
getException(byte). Static method in class javacard.framework.UserException
Returns the system instance of UserException
getFCI(). Method in class javacardx.framework.File
Get this file's FCI (if any).
getFID(). Method in class javacardx.framework.File
Get this file's 16-bit FID.
getIFSC(). Static method in class javacard.framework.APDU
Returns the configured IFSC (information field size for ICC).
getInstance(byte). Static method in class javacard.framework.CardException
Returns the re-used system instance of card exception.
getInstance(byte). Static method in class javacard.framework.CardRuntimeException
Returns the System instance of the card runtime exception.
getMaxChildFiles(). Method in class javacardx.framework.DedicatedFile
Get the maximum number of child files in this DF.
getMaxCommitCapacity(). Static method in class javacard.framework.System
Returns the total number of bytes in the commit buffer.
getMaxNumRecords(). Method in class javacardx.framework.LinearVariableFile
Get the maximum number of records in this file.
getName(). Method in class javacardx.framework.DedicatedFile
Get the file's name.
getNewFirstRecord(). Method in class javacardx.framework.CyclicFile
Recycles the oldest record as the new most recent record (record number 1).

getNumChildFiles(). Method in class javacardx.framework.DedicatedFile
Get the actual number of child files in this DF.
getNumRecords(). Method in class javacardx.framework.LinearVariableFile
Get the actual number of records in this file.
getParent(). Method in class javacardx.framework.File
Get this file’s parent DF.
getReason(). Method in class javacard.framework.CardException
Returns the reason of the failure.
getReason(). Method in class javacard.framework.CardRuntimeException
Returns the reason of the failure.
getRecord(byte). Method in class javacardx.framework.CyclicFile
Get the record byte array for the specified record.
getRecord(byte). Method in class javacardx.framework.LinearVariableFile
Get the record byte array for the specified record number.
getRecord(byte, byte, byte). Method in class javacardx.framework.LinearVariableFile
Get the record byte array for the specified record identifier using the specified starting point and direction.
getSecurity(byte). Method in class javacardx.framework.File
Get this file’s external read or write security.
getSFI(). Method in class javacardx.framework.ElementaryFile
Get this file’s 5-bit SFI.
getTriesLeft(). Method in class javacard.framework.PIN
Returns the number of times remaining that an incorrect PIN can be presented before the PIN is blocked.
getUnusedCommitCapacity(). Static method in class javacard.framework.System
Returns the number of bytes left in the commit buffer.
getVersion(). Static method in class javacard.framework.System
Returns the current major and minor version of the Java Card API.

I

ILLEGAL_USE. Static variable in class javacard.framework.ApduException
ILLEGAL_USE. Static variable in class javacard.framework.PINException
ILLEGAL_USE. Static variable in class javacard.framework.SystemException
ILLEGAL_VALUE. Static variable in class javacard.framework.PINException
increaseMaxChildFiles(byte). Method in class javacardx.framework.DedicatedFile
Increase the maximum number of child files in this DF.
increaseMaxNumRecords(byte). Method in class javacardx.framework.CyclicFile
Increase the maximum number of records.
increaseMaxNumRecords(byte). Method in class javacardx.framework.LinearVariableFile
Increase the maximum number of records in this file.
install(APDU, byte). Static method in class javacard.framework.Applet
Installs this applet.
IO_ERROR. Static variable in class javacard.framework.ApduException
isAllowed(byte). Method in class javacardx.framework.File
Check this file’s external read or write security.
isEqual(AID). Method in class javacard.framework.AID
Checks if the specified AID object is the same as `this`.

isEqual(byte[], short, byte). Method in class javacard.framework.AID
Checks if the specified AID byte array is the same as `this`.
isInitialized(). Method in class javacardx.crypto.Key
Reports the initialized state of the key.
isInitialized(). Method in class javacardx.crypto.RSA_CRT_Key
Reports the initialized state of the key.
isInitialized(). Method in class javacardx.crypto.RSA_Key
Reports the initialized state of the key.
isInitialized(). Method in class javacardx.crypto.SymKey
Reports the initialized state of the key.
isTransient(Object). Method in class javacard.framework.System
Used to check if the object is in volatile memory.
isValidated(). Method in class javacard.framework.PIN
Returns true if a valid PIN has been presented since the last card reset or last call to `reset()`.

K

Key(short). Constructor for class javacardx.crypto.Key
constructs a key with a specific bit length

L

LinearFixedFile(short, byte, byte). Constructor for class javacardx.framework.LinearFixedFile
Constructor.
LinearVariableFile(short, byte). Constructor for class javacardx.framework.LinearVariableFile
Constructor.

M

makeTransient(Object). Method in class javacard.framework.System
Called to put the referenced object in volatile memory.
MessageDigest(). Constructor for class javacardx.crypto.MessageDigest

P

PIN(byte). Constructor for class javacard.framework.PIN
Constructor.
PINException(byte). Constructor for class javacard.framework.PINException
Constructs a PINException.
process(APDU). Method in class javacard.framework.Applet
Processes an incoming APDU.
process(APDU). Method in class javacardx.framework.FileSystem
Handles FileSystem APDUs as specified by ISO 7816-4.
putData(APDU). Method in class javacardx.framework.FileSystem
Handles PUT DATA command APDU as specified by ISO 7816-4.

R

RandomData(). Constructor for class javacardx.crypto.RandomData
readBinary(APDU). Method in class javacardx.framework.FileSystem
Handles READ BINARY command APDU as specified by ISO 7816-4.
readRecord(APDU). Method in class javacardx.framework.FileSystem
Handles READ RECORD command APDU as specified by ISO 7816-4.
reason. Variable in class javacard.framework.CardException
reason. Variable in class javacard.framework.CardRuntimeException
receiveBytes(short). Method in class javacard.framework.APDU
Gets more bytes into this APDU buffer at specified offset `bOff`.
register(Applet). Static method in class javacard.framework.System
Register an applet with the System.
reset(). Method in class javacard.framework.PIN
Resets validation flag
RSA_CRT_Key(). Constructor for class javacardx.crypto.RSA_CRT_Key
Constructs an empty RSA_CRT_Key
RSA_Key(). Constructor for class javacardx.crypto.RSA_Key
Constructs an empty RSA_Key that stores its key data in modulus/exponent form.

S

select(APDU, boolean). Method in class javacardx.framework.FileSystem
This method handles SELECT APDUs as specified by ISO 7816-4.
select(APDU, byte, boolean). Method in class javacard.framework.Applet
Called by the System to inform this applet that it should execute.
selectFile(File). Method in class javacardx.framework.FileSystem
Make the specified file the current DF or current EF.
sendBytes(short, short). Method in class javacard.framework.APDU
Sends `len` more bytes from `apdu.buffer` at specified offset `bOff`.
sendBytesLong(byte[], short, short). Method in class javacard.framework.APDU
Sends `len` more bytes from `outData` at specified offset `bOff`.
setAuthFlag(byte, boolean). Method in class javacardx.framework.FileSystem
Set authorization flag.
setCurrentDedicatedFile(DedicatedFile). Method in class javacardx.framework.FileSystem
Set current DF.
setCurrentElementaryFile(ElementaryFile). Method in class javacardx.framework.FileSystem
Set current EF.
setCurrentRecNum(byte). Method in class javacardx.framework.FileSystem
Set current record number.
setDP1(byte[], short, short). Method in class javacardx.crypto.RSA_CRT_Key
Sets the value of the DP1 parameter.
setDQ1(byte[], short, short). Method in class javacardx.crypto.RSA_CRT_Key
Sets the value of the P key.
setExponent(byte[], short, short). Method in class javacardx.crypto.RSA_Key
Sets the exponent value of the key.

Java Card 2.0 API, copyright Sun Microsystems Inc

setFCI(byte[]). Method in class javacardx.framework.File
Set this file's FCI.
setIncoming(). Method in class javacard.framework.APDU
Indicates that this command has incoming data.
setIncomingAndReceive(). Method in class javacard.framework.APDU
Indicates that this command has incoming data and gets data bytes into the APDU buffer following the header.
setIV(byte[], short, short). Method in class javacardx.crypto.DES_CBC_Key
Sets the initialization vector used in CBC mode DES operations.
setKey(byte[], short, short). Method in class javacardx.crypto.SymKey
initializes a key from raw key data bytes.
setModulus(byte[], short, short). Method in class javacardx.crypto.RSA_Key
Sets the modulus value of the key.
setOutgoing(). Method in class javacard.framework.APDU
This method is used to set the data transfer direction to outbound and to obtain the expected length of response (Le).
setOutgoingLength(short). Method in class javacard.framework.APDU
Sets the expected length of response data.
setP(byte[], short, short). Method in class javacardx.crypto.RSA_CRT_Key
Sets the value of the P parameter.
setPQ(byte[], short, short). Method in class javacardx.crypto.RSA_CRT_Key
Sets the value of the PQ parameter.
setQ(byte[], short, short). Method in class javacardx.crypto.RSA_CRT_Key
Sets the value of the Q parameter.
setReason(byte). Method in class javacard.framework.CardException
Sets the reason for the failure.
setReason(byte). Method in class javacard.framework.CardRuntimeException
Sets the reason for the failure.
setSecurity(byte, byte). Method in class javacardx.framework.File
Set this file's external read or write security.
setSeed(byte[], short, short). Static method in class javacardx.crypto.RandomData
seeds random data generator.
Sha1MessageDigest(). Constructor for class javacardx.crypto.Sha1MessageDigest
share(Object). Static method in class javacard.framework.System
Makes the specified object instance available for access from any installed applet on the card.
share(Object, AID). Static method in class javacard.framework.System
Makes the specified object instance available for access from the applet identified by the specified AID object.
Sign(byte[], short, short, byte[], short). Method in class javacardx.crypto.AsymKey
Signs data using this key.
sign(byte[], short, short, byte[], short, short). Method in class javacardx.crypto.RSA_CRT_Key
Signs data using this key.
sign(byte[], short, short, byte[], short, short). Method in class javacardx.crypto.RSA_Key
Signs data using this key.
SW_BYTES_REMAINING. Static variable in class javacard.framework.APDU
SW_CLA_NOT_SUPPORTED. Static variable in class javacard.framework.APDU
SW_DATA_INVALID. Static variable in class javacard.framework.APDU
SW_FILE_INVALID. Static variable in class javacard.framework.APDU
SW_FILE_NOT_FOUND. Static variable in class javacard.framework.APDU

SW_FUNC_NOT_SUPPORTED. Static variable in class javacard.framework.APDU
SW_INCORRECT_P1P2. Static variable in class javacard.framework.APDU
SW_INS_NOT_SUPPORTED. Static variable in class javacard.framework.APDU
SW_NO_ERROR. Static variable in class javacard.framework.APDU
SW_PIN_REQUIRED. Static variable in class javacard.framework.APDU
SW_RECORD_NOT_FOUND. Static variable in class javacard.framework.APDU
SW_WRONG_DATA. Static variable in class javacard.framework.APDU
SW_WRONG_LENGTH. Static variable in class javacard.framework.APDU
SW_WRONG_P1P2. Static variable in class javacard.framework.APDU
SymKey(). Constructor for class javacardx.crypto.SymKey
System(). Constructor for class javacard.framework.System
SystemException(byte). Constructor for class javacard.framework.SystemException
Constructs a SystemException.

T

TransparentFile(short, byte[]). Constructor for class javacardx.framework.TransparentFile
Constructor, with data byte array specified.
TransparentFile(short, short). Constructor for class javacardx.framework.TransparentFile
Constructor, with data byte array size specified.

U

updateAndUnblock(byte[], short, byte). Method in class javacard.framework.PIN
This method sets a new value for the PIN and resets the PIN try counter to the value of the PIN try limit.
updateBinary(APDU). Method in class javacardx.framework.FileSystem
Handles UPDATE BINARY command APDU as specified by ISO 7816-4.
updateRecord(APDU). Method in class javacardx.framework.FileSystem
Handles UPDATE RECORD command APDU as specified by ISO 7816-4.
UserException(byte). Constructor for class javacard.framework.UserException
Constructs a UserException.
Util(). Constructor for class javacard.framework.Util

V

Verify(byte[], short, short, byte[], short, short). Method in class javacardx.crypto.AsymKey
Verifies signed data using this key.
verify(byte[], short, short, byte[], short, short). Method in class javacardx.crypto.RSA_Key
Verifies signed data using this key.
verifyMAC(byte[], short, short, byte[], short, short). Method in class javacardx.crypto.DES_CBC_Key
Verifies signed data using this key.

W

wait(byte). Method in class javacard.framework.APDU
Requests additional processsing time from Terminal.
writeBinary(APDU). Method in class javacardx.framework.FileSystem
Handles WRITE BINARY command APDU as specified by ISO 7816-4.
writeRecord(APDU). Method in class javacardx.framework.FileSystem
Handles WRITE RECORD command APDU as specified by ISO 7816-4.

package javacard.framework

Class Index

- AID
- APDU
- Applet
- PIN
- System
- Util

Exception Index

- ApduException
- CardException
- CardRuntimeException
- FileIoException
- PINException
- SystemException
- UserException

Class javacard.framework.AID

```
java.lang.Object
   |
   +----javacard.framework.AID
```

public class **AID**
extends Object

This class encapsulates the Application Identifier associated with an applet. It contains a byte array : ISO 7816-5 defined 11..16 bytes.

Method Index

o **getArray**(byte[], short)
    Called to obtain a copy of the byte array within AID object.
o **isEqual**(AID)
    Checks if the specified AID object is the same as `this`.
o **isEqual**(byte[], short, byte)
    Checks if the specified AID byte array is the same as `this`.

Methods

o **getArray**

```
public byte getArray(byte dest[],
                     short offset)
```

Called to obtain a copy of the byte array within AID object.

Parameters:
dest - byte array to copy to.
offset - within dest to start the copy.
Returns:
the length of the AID byte array.

o **isEqual**

```
public boolean isEqual(AID aid)
```

Checks if the specified AID object is the same as `this`.

Parameters:
aid - the object to compare against
Returns:
true if equal, false otherwise.

o **isEqual**

```
public boolean isEqual(byte bArray[],
                       short offset,
                       byte length)
```

Checks if the specified AID byte array is the same as `this`.

Parameters:
bArray - to compare against
offset - within bArray to begin
length - of AID byte array

Returns:
true if equal, false otherwise.

Java Card 2.0 API, copyright Sun Microsystems Inc

Class javacard.framework.APDU

```
java.lang.Object
   |
   +----javacard.framework.APDU
```

public final class **APDU**
extends Object

An ISO 7816 APDU is an Application PDU (Protocol Data Unit) that is the communication format between the card and the off-card applications. The format of the APDU is defined in ISO specification 7816-4.

APDU objects are owned by the System.

The applet will receive an APDU to process from the system in `process(APDU)`. When the APDU object is handed to `process(APDU)`, the first five bytes [cla, ins, p1, p2, p3] are available in the APDU buffer.

Depending on the size of the incoming APDU, it may not fit inside the buffer and may need to be read in portions by the applet. There exist methods in `APDU` to do so.

For sending large byte arrays as response data, the APDU class provides a special method `sendBytesLong()` which manages the APDU buffer.

This class provides constants (named `SW_..`) for ISO 7816-4 defined response APDU status codes.

```
 // The purpose of this example is to show most of the methods
 // in use and not to depict any particular APDU processing
public short process(APDU apdu){
  // ...
  byte[] buffer = apdu.getBuffer();
  byte cla = buffer[0];
  byte ins = buffer[1];
  ...
  // assume this command has incoming data
  apdu.setIncoming();
  //
  // Lc tells us the incoming apdu command length
  byte bytesLeft = buffer[4]();
  byte readCount;
  //
  while ( bytesLeft > 0){
      *  // see if we can fill it all into the buffer
      if ( bytesLeft > buffer.length() ) readCount = buffer.length();
      else readCount = bytesLeft;
      bytesRead = receiveBytes ( (byte) 0 );
      // process buffer[0..bytesRead-1];
      bytesLeft -= readCount;
      }
  //
  //...
  //
  // construct a reply APDU
```

```
// build response data in apdu.buffer[ 0.. outCount-1 ];
short le = apdu.setOutgoing();
apdu.setOutgoingLength( 2 );
buffer[0] = 1; buffer[1] = 2;
sendBytes ( 0 , 2 );
// return good complete status 90 00
return (APDU.SW_NO_ERROR);
}
```

Variable Index

o **SW_BYTES_REMAINING**
o **SW_CLA_NOT_SUPPORTED**
o **SW_DATA_INVALID**
o **SW_FILE_INVALID**
o **SW_FILE_NOT_FOUND**
o **SW_FUNC_NOT_SUPPORTED**
o **SW_INCORRECT_P1P2**
o **SW_INS_NOT_SUPPORTED**
o **SW_NO_ERROR**
o **SW_PIN_REQUIRED**
o **SW_RECORD_NOT_FOUND**
o **SW_WRONG_DATA**
o **SW_WRONG_LENGTH**
o **SW_WRONG_P1P2**

Method Index

o **getBuffer**()
Returns the APDU buffer byte array.
o **getIFSC**()
Returns the configured IFSC (information field size for ICC).
o **receiveBytes**(short)
Gets more bytes into this APDU buffer at specified offset `bOff`.
o **sendBytes**(short, short)
Sends `len` more bytes from `apdu.buffer` at specified offset `bOff`.
o **sendBytesLong**(byte[], short, short)
Sends `len` more bytes from `outData` at specified offset `bOff`.
o **setIncoming**()
Indicates that this command has incoming data.
o **setIncomingAndReceive**()
Indicates that this command has incoming data and gets data bytes into the APDU buffer following the header.
o **setOutgoing**()
This method is used to set the data transfer direction to outbound and to obtain the expected length of response (Le).
o **setOutgoingLength**(short)
Sets the expected length of response data.
o **wait**(byte)
Requests additional processsing time from Terminal.

Variables

o **SW_NO_ERROR**

`public static final short SW_NO_ERROR`

o **SW_BYTES_REMAINING**

`public static final short SW_BYTES_REMAINING`

o **SW_WRONG_LENGTH**

`public static final short SW_WRONG_LENGTH`

o **SW_PIN_REQUIRED**

`public static final short SW_PIN_REQUIRED`

o **SW_FILE_INVALID**

`public static final short SW_FILE_INVALID`

o **SW_DATA_INVALID**

`public static final short SW_DATA_INVALID`

o **SW_WRONG_DATA**

`public static final short SW_WRONG_DATA`

o **SW_FUNC_NOT_SUPPORTED**

`public static final short SW_FUNC_NOT_SUPPORTED`

o **SW_FILE_NOT_FOUND**

`public static final short SW_FILE_NOT_FOUND`

o **SW_RECORD_NOT_FOUND**

`public static final short SW_RECORD_NOT_FOUND`

o **SW_INCORRECT_P1P2**

`public static final short SW_INCORRECT_P1P2`

o **SW_WRONG_P1P2**

`public static final short SW_WRONG_P1P2`

o **SW_INS_NOT_SUPPORTED**

`public static final short SW_INS_NOT_SUPPORTED`

o **SW_CLA_NOT_SUPPORTED**

`public static final short SW_CLA_NOT_SUPPORTED`

Methods

o **getBuffer**

```
public byte[] getBuffer()
```

Returns the APDU buffer byte array.

Returns:
byte array containing the APDU buffer

o **getIFSC**

```
public static byte getIFSC()
```

Returns the configured IFSC (information field size for ICC).

Note: On receiveBytes() the length param should account for this potential blocksize.

Returns:
IFSC, the maximum size of incoming blocks (T=1) from terminal.

o **setOutgoing**

```
public short setOutgoing() throws ApduException
```

This method is used to set the data transfer direction to outbound and to obtain the expected length of response (Le).

Notes.

- The remaining incoming data if any, will be discarded.
- T=0 (Case 4) will return 256.
- The APDU buffer at offset 0 will be used to read the unread incoming data.

Returns:
the Le.
Throws: ApduException
if method already invoked.

o **setOutgoingLength**

```
public void setOutgoingLength(short len) throws ApduException
```

Sets the expected length of response data. Default is 0.

Note. Used in T=0 (Case 4) protocol to prompt terminal for read command.

Parameters:
len - the length of response data.
Throws: ApduException
if setOutgoing() not called or this method already invoked.

o **setIncoming**


```
public void setIncoming() throws ApduException
```

Indicates that this command has incoming data.

Note. Used in T=0 (Case 3 or 4) protocol to assume P3 param is Lc.

Throws: ApduException
if this method already invoked.

o **receiveBytes**

```
public short receiveBytes(short bOff) throws ApduException
```

Gets more bytes into this APDU buffer at specified offset `bOff`.

Notes:

- The space in the buffer must allow for the IFSC blocksize(information field size for terminal).
- User must manage the APDU buffer.

Parameters:
bOff - the offset into APDU buffer.
Returns:
number of bytes read. returns 0 if no bytes available.
Throws: ApduException
on potential APDU buffer overflow OR if `setIncoming()` not called.

o **setIncomingAndReceive**

```
public short setIncomingAndReceive() throws ApduException
```

Indicates that this command has incoming data and gets data bytes into the APDU buffer following the header.

Notes:

- Used in T=0 (Case 3 or 4) protocol to assume P3 param is Lc.
- Data is read into the buffer at offset 5.
- This method is a combination of setIncoming() and receiveBytes(5) APIs.

Returns:
number of bytes read. returns 0 if no bytes available.
Throws: ApduException
if `setIncoming()` already invoked.

o **sendBytes**

```
public void sendBytes(short bOff,
                      short len) throws ApduException
```

Sends `len` more bytes from `apdu.buffer` at specified offset `bOff`.

User must manage the APDU buffer.

Parameters:
bOff - the offset into APDU buffer.
len - the bytelength of the data to send.
Returns:
void.
Throws: ApduException
if `bOff` or `len` is too large and the APDU buffer overflows OR responseLength inconsistency.

o **sendBytesLong**

```
public void sendBytesLong(byte outData[],
                          short bOff,
                          short len) throws ApduException
```

Sends `len` more bytes from `outData` at specified offset `bOff`.

System will manage the APDU buffer.

Parameters:
outData - the large byte array source.
bOff - the offset into OutData array.
len - the bytelength of the data to send.
Returns:
void.
Throws: ApduException
if responseLength inconsistency

o **wait**

```
public void wait(byte waitUnits)
```

Requests additional processsing time from Terminal.

Note. In T=0 waitUnits is ignored. The standard additional time is requested.

Parameters:
waitUnits - number of BWT(byte wait time) units.

Java Card 2.0 API, copyright Sun Microsystems Inc

Class javacard.framework.ApduException

```
java.lang.Object
   |
   +----java.lang.Throwable
           |
           +----java.lang.Exception
                   |
                   +----java.lang.RuntimeException
                           |
                           +----javacard.framework.CardRuntimeException
                                   |
                                   +----javacard.framework.ApduException
```

---

public class **ApduException**
extends CardRuntimeException

The `ApduException` signals a APDU access related exception.

Variable Index

o **BAD_LENGTH**
o **BUFFER_BOUNDS**
o **ILLEGAL_USE**
o **IO_ERROR**

Constructor Index

o **ApduException**(byte)
Constructs an ApduException.

Method Index

o **getException**(byte)
Returns the system instance of ApduException

Variables

o **ILLEGAL_USE**

```
public static final byte ILLEGAL_USE
```

o **BUFFER_BOUNDS**

```
public static final byte BUFFER_BOUNDS
```

o **BAD_LENGTH**

```
public static final byte BAD_LENGTH
```

o **IO_ERROR**

```
public static final byte IO_ERROR
```

Constructors

o **ApduException**

```
public ApduException(byte reason)
```

Constructs an ApduException. Reason codes:

Table ApduException

Byte	Description
1	APDU Illegal Use
2	APDU buffer bounds error
3	APDU outGoingLength inconsistency
4	APDU I/O Error

Methods

o **getException**

```
public static ApduException getException(byte reason)
```

Returns the system instance of ApduException

Class javacard.framework.Applet

```
java.lang.Object
   |
   +----javacard.framework.Applet
```

public abstract class **Applet**
extends Object

This abstract class defines an applet in a smart card.

The `Applet` should be extended by any applet wishing to be loaded onto, installed into and executed on a Java Card compliant smart card.

Example usage of `Applet`

```
public class MyApplet extends javacard.framework.Applet{
  byte b[];
  public static short install( APDU apdu, byte lenRcvd ) {
    // make all my allocations here, so I do not run
    // out of memory later
    b[] = new byte[100];
    // we need a commit buffer 255 byte long
    return ( apdu.getBuffer().length >= 16 );
  }
  public short select(APDU apdu, byte lenRcvd, boolean selectedFlag) {
    // selection initialization
    b[17] = 42;
    if (selectedFlag) {
      // send FCI as reply to select command
      byte Le =  setOutgoing();
      byte[] fciData = myFCITransparentFile.getData();
      apdu.sendBytesLong(fciData, 0, fciData.length);
      }
    return (APDU.SW_NO_ERROR);
  }
  public short process(APDU apdu){
  // .. process the incoming data and reply
  }
}
```

Constructor Index

o **Applet**()

Method Index

o **install**(APDU, byte)
Installs this applet.

o **process**(APDU)
    Processes an incoming APDU.
o **select**(APDU, byte, boolean)
    Called by the System to inform this applet that it should execute.

Constructors

o **Applet**

```
public Applet()
```

Methods

o **install**

```
public static short install(APDU apdu,
                            byte lengthReceived)
```

Installs this applet. Any specific installation calls by the applet will be issued here, e.g., calls to check system resources, such as:

```
if ( apdu.getBuffer().length >= 16 ) return true;
else ...
```

(This method is called by the System at install time.)

Notes:

- A return of `APDU.SW_NO_ERROR` signals to the System that this applet should be installed.
- *The system has received the APDU's incoming data before this call.*
- *The incoming header is available in buffer[0]..buffer[4] and data in buffer[5]..*
- *The number of data bytes received so far is available in* `lengthReceived`.

Parameters:
    apdu - the incoming APDU containing the INSTALL command.
    length - of data bytes previously received and available in buffer[5]...
Returns:
    status, short, response bytes per ISO 7816-3.
See Also:
    APDU

o **process**

```
public short process(APDU apdu)
```

Processes an incoming APDU. An Applet is expected to return the status bytes response to be returned to terminal.

This method is called as soon as the System has received the header of the incoming message. See the APDU class for details.
The five header bytes of the APDU are available in `APDU.buffer` at the time this method is called.

Parameters:
    apdu - the incoming APDU
Returns:
    status, short, response bytes per ISO 7816-3
See Also:
    APDU

o **select**

```
public short select(APDU apdu,
                    byte lengthReceived,
                    boolean selectedFlag)
```

Called by the System to inform this applet that it should execute. It is called when a 'select' command is received and the applet is selected. The applet is expected to return the associated response data (such as FCI) and status bytes to terminal. (For more information about 'selected', please refer to the ISO 7816 specification.)

A subclass of `Applet` should override this method if it has any operation that it wants to perform when a select command is received. If the `selectedFlag` is true, this applet has just been selected by the Executive.

The implementation of this method provided by `Applet` class does nothing.

Notes:

- *The system has received the APDU's incoming data before this call.*
- *The incoming header is available in buffer[0]..buffer[4] and data in buffer[5]..*
- *The number of data bytes received so far is available in* `lengthReceived`.

Parameters:
    apdu - the incoming APDU
    length - of data bytes previously received and available in buffer[5]...
    selectedFlag, - true if the applet just got selected, false otherwise
Returns:
    status, short, response bytes per ISO 7816-3.

See Also:
    APDU

All Packages Class Hierarchy This Package Previous Next Index

Class javacard.framework.CardException

```
java.lang.Object
   |
   +----java.lang.Throwable
           |
           +----java.lang.Exception
                   |
                   +----javacard.framework.CardException
```

public class **CardException**
extends Exception

The `CardException` signals a general Card exception in `javacard.framework`.

Variable Index

o **reason**

Constructor Index

o **CardException**()
Constructs a Card exception instance To conserve on resources use `getInstance()` to re-use the System instance of this class.

Method Index

o **getInstance**(byte)
Returns the re-used system instance of card exception.
o **getReason**()
Returns the reason of the failure.
o **setReason**(byte)
Sets the reason for the failure.

Variables

o **reason**

```
protected byte reason
```

Constructors

o **CardException**

```
public CardException()
```


Java Card 2.0 API, copyright Sun Microsystems Inc

Constructs a Card exception instance To conserve on resources use `getInstance()` to re-use the System instance of this class.

Methods

o **getInstance**

```
public static CardException getInstance(byte reason)
```

Returns the re-used system instance of card exception.

Parameters:
reason - the reason for the failure.

o **getReason**

```
public byte getReason()
```

Returns the reason of the failure. See tables of reasons and values.

Returns:
the reason of the failure.

o **setReason**

```
public void setReason(byte reason)
```

Sets the reason for the failure. See tables of reasons and values.

Parameters:
reason - the failure reason.

All Packages Class Hierarchy This Package Previous Next Index

All Packages Class Hierarchy This Package Previous Next Index

Class javacard.framework.CardRuntimeException

```
java.lang.Object
   |
   +----java.lang.Throwable
           |
           +----java.lang.Exception
                   |
                   +----java.lang.RuntimeException
                           |
                           +----javacard.framework.CardRuntimeException
```

public class **CardRuntimeException**
extends RuntimeException

The `CardRuntimeException` signals a general Runtime exception in `javacard.framework`.

Variable Index

o **reason**

Constructor Index

o **CardRuntimeException**()
  Constructs a Card Runtime exception instance.

Method Index

o **getInstance**(byte)
  Returns the System instance of the card runtime exception.
o **getReason**()
  Returns the reason of the failure.
o **setReason**(byte)
  Sets the reason for the failure.

Variables

o **reason**

```
protected byte reason
```

Constructors

o **CardRuntimeException**

```
public CardRuntimeException()
```


Java Card 2.0 API, copyright Sun Microsystems Inc

Constructs a Card Runtime exception instance. To conserve on resources use `getInstance()` to re-use the System instance of this class.

Methods

o **getInstance**

```
public static CardRuntimeException getInstance(byte reason)
```

Returns the System instance of the card runtime exception.

Parameters:
reason - the reason for the failure.

o **getReason**

```
public byte getReason()
```

Returns the reason of the failure. See tables of reasons and values.

Returns:
the reason of the failure.

o **setReason**

```
public void setReason(byte reason)
```

Sets the reason for the failure. See tables of reasons and values.

Parameters:
reason - the failure reason.

All Packages Class Hierarchy This Package Previous Next Index

Class javacard.framework.FileIoException

```
java.lang.Object
   |
   +----java.lang.Throwable
           |
           +----java.lang.Exception
                   |
                   +----java.lang.RuntimeException
                           |
                           +----javacard.framework.CardRuntimeException
                                   |
                                   +----javacard.framework.FileIoException
```

public class **FileIoException**
extends CardRuntimeException

The `FileIoException` signals a File access related exception.

Constructor Index

o **FileIoException**(byte)

Method Index

o **getException**(byte)
  Returns the System instance of FileIoException

Constructors

o **FileIoException**

```
public FileIoException(byte reason)
```

Methods

o **getException**

```
public static FileIoException getException(byte reason)
```

Returns the System instance of FileIoException

Class javacard.framework.PIN

```
java.lang.Object
   |
   +----javacard.framework.PIN
```

public class **PIN**
extends Object

This class represents a PIN. It maintains these internal values:

- PIN value
- try limit, the maximum number of times an incorrect PIN can be presented before the PIN is blocked
- try counter, the remaining number of times an incorrect PIN presentation is permitted
- validated flag, true if a valid PIN has been presented

This class does not make any assumptions about where the data for the PIN comparison is stored.

Constructor Index

o **PIN**(byte)
  Constructor.

Method Index

o **check**(byte[], short, byte)
  Compares `pin` against the PIN value.
o **getTriesLeft**()
  Returns the number of times remaining that an incorrect PIN can be presented before the PIN is blocked.
o **isValidated**()
  Returns true if a valid PIN has been presented since the last card reset or last call to `reset()`.
o **reset**()
  Resets validation flag
o **updateAndUnblock**(byte[], short, byte)
  This method sets a new value for the PIN and resets the PIN try counter to the value of the PIN try limit.

Constructors

o **PIN**

```
public PIN(byte pinSize) throws PINException
```

Constructor. Allocates a PIN as a `pinSize` byte array initialized to 0s.

Parameters:
pinSize - the length of the PIN bytearray.
Throws: PINException
on illegal parameter

Methods

o **updateAndUnblock**

```
public void updateAndUnblock(byte pin[],
                             short offset,
                             byte length) throws PINException
```

This method sets a new value for the PIN and resets the PIN try counter to the value of the PIN try limit. It also resets the validation flag.

Parameters:
pin - the bytearray containing the new pin value
offset - the starting offset in the pin array
length - the length of the new pin.
Throws: PINException
if param invalid or update failed.

o **getTriesLeft**

```
public byte getTriesLeft()
```

Returns the number of times remaining that an incorrect PIN can be presented before the PIN is blocked.

Returns:
the number of times remaining

o **check**

```
public boolean check(byte pin[],
                     short offset,
                     byte length)
```

Compares `pin` against the PIN value. If they match, it sets the validated flag and resets the try counter to its maximum. If not, it decrements the try counter, and if the counter has reached zero, blocks the PIN.

Parameters:
pin - the PIN value being checked
offset - the starting offset in the pin array
length - the length of pin.
Returns:
true if PIN is OK; false otherwise

Java Card 2.0 API, copyright Sun Microsystems Inc

o **isValidated**

```
public boolean isValidated()
```

Returns true if a valid PIN has been presented since the last card reset or last call to `reset()`.

Returns:
true if validated; false otherwise

o **reset**

```
public void reset()
```

Resets validation flag

All Packages Class Hierarchy This Package Previous Next Index

Class javacard.framework.PINException

```
java.lang.Object
   |
   +----java.lang.Throwable
           |
           +----java.lang.Exception
                   |
                   +----java.lang.RuntimeException
                           |
                           +----javacard.framework.CardRuntimeException
                                   |
                                   +----javacard.framework.PINException
```

public class **PINException**
extends CardRuntimeException

The `PINException` signals a PIN access-related exception.

Variable Index

o **BUFFER_BOUNDS**
o **ILLEGAL_USE**
o **ILLEGAL_VALUE**

Constructor Index

o **PINException**(byte)
Constructs a PINException.

Method Index

o **getException**(byte)
Returns the system instance of PINException

Variables

o **BUFFER_BOUNDS**

`public static final byte BUFFER_BOUNDS`

o **ILLEGAL_VALUE**

`public static final byte ILLEGAL_VALUE`

o **ILLEGAL_USE**

Java Card 2.0 API, copyright Sun Microsystems Inc

`public static final byte ILLEGAL_USE`

Constructors

o **PINException**

`public PINException(byte reason)`

Constructs a PINException. Reason codes:

Table PINException

Byte	Description
1	PIN buffer bounds error
2	PIN Illegal value
2	PIN Illegal usage

Methods

o **getException**

`public static PINException getException(byte reason)`

Returns the system instance of PINException

All Packages Class Hierarchy This Package Previous Next Index

Class javacard.framework.System

```
java.lang.Object
   |
   +----javacard.framework.System
```

public final class **System**
extends Object

The `System` class is a centralized point of applet execution, resource management and security in the smart card.

On startup, System initializes the commit buffer (used for `commitTransaction()/abortTransaction()`).

Constructor Index

o **System**()

Method Index

o **abortTransaction**()
 Aborts the atomic transaction.
o **beginTransaction**()
 Begins an atomic transaction.
o **commitTransaction**()
 Commits an atomic transaction.
o **getAID**()
 Returns the unique Applet Identifier (AID) object associated with the current applet.
o **getMaxCommitCapacity**()
 Returns the total number of bytes in the commit buffer.
o **getUnusedCommitCapacity**()
 Returns the number of bytes left in the commit buffer.
o **getVersion**()
 Returns the current major and minor version of the Java Card API.
o **isTransient**(Object)
 Used to check if the object is in volatile memory.
o **makeTransient**(Object)
 Called to put the referenced object in volatile memory.
o **register**(Applet)
 Register an applet with the System.
o **share**(Object)
 Makes the specified object instance available for access from any installed applet on the card.
o **share**(Object, AID)
 Makes the specified object instance available for access from the applet identified by the specified AID object.

Java Card 2.0 API, copyright Sun Microsystems Inc

Constructors

o **System**

```
public System()
```

Methods

o **register**

```
public static void register(Applet a)
```

Register an applet with the System.

Parameters:
 a - the applet being added to the Card framework.

o **share**

```
public static void share(Object object,
                         AID otherAID) throws SystemException
```

Makes the specified object instance available for access from the applet identified by the specified AID object.

Parameters:
 object - the object which we want to share.
 otherAID, - identifies the other applet to share with.
Throws: SystemException
 if illegal parameter

o **share**

```
public static void share(Object object)
```

Makes the specified object instance available for access from any installed applet on the card.

Parameters:
 object - the object which we want to share with all others.

o **isTransient**

```
public boolean isTransient(Object obj)
```

Used to check if the object is in volatile memory.

Parameters:
 obj - the object being queried.
Returns:
 true if the object is loaded in volatile memory.

o **makeTransient**

```
public byte makeTransient(Object obj)
```

Called to put the referenced object in volatile memory.

Parameters:
obj - the object to be made available in volatile memory.
Returns:
0 if successful. Non zero error code(TBD) otherwise.

o **getVersion**

```
public static short getVersion()
```

Returns the current major and minor version of the Java Card API.

Returns:
version number as byte.byte (major.minor)

o **getAID**

```
public static AID getAID()
```

Returns the unique Applet Identifier (AID) object associated with the current applet.

Returns:
the AID object reference.

o **beginTransaction**

```
public static void beginTransaction()
```

Begins an atomic transaction. The System object maintains a commit buffer into which data is written so that System always can guarantee, at commit time, that everything in the buffer is written, or nothing at all.

See Also:
commitTransaction, abortTransaction

o **abortTransaction**

```
public static void abortTransaction()
```

Aborts the atomic transaction. The contents of the commit buffer is discarded.

See Also:
beginTransaction, commitTransaction

o **commitTransaction**

```
public static boolean commitTransaction()
```

Commits an atomic transaction. The contents of commit buffer is atomically commited. If the buffer cannot be commited, false is returned and no write will occur.

Returns:
true if commit was successful; false otherwise

See Also:
beginTransaction, abortTransaction

o **getUnusedCommitCapacity**

```
public static short getUnusedCommitCapacity()
```

Returns the number of bytes left in the commit buffer.

Returns:
the number of bytes left in the commit buffer

o **getMaxCommitCapacity**

```
public static short getMaxCommitCapacity()
```

Returns the total number of bytes in the commit buffer.

Returns:
the total number of bytes in the commit buffer

All Packages Class Hierarchy This Package Previous Next Index

All Packages Class Hierarchy This Package Previous Next Index

Class javacard.framework.SystemException

```
java.lang.Object
   |
   +----java.lang.Throwable
           |
           +----java.lang.Exception
                   |
                   +----java.lang.RuntimeException
                           |
                           +----javacard.framework.CardRuntimeException
                                   |
                                   +----javacard.framework.SystemException
```

public class **SystemException**
extends CardRuntimeException

The `SystemException` signals a System access-related exception.

Variable Index

o **ILLEGAL_USE**

Constructor Index

o **SystemException**(byte)
Constructs a SystemException.

Method Index

o **getException**(byte)
Returns the system instance of SystemException

Variables

o **ILLEGAL_USE**

```
public static final byte ILLEGAL_USE
```

Constructors

o **SystemException**

```
public SystemException(byte reason)
```

Constructs a SystemException. Reason codes :

Java Card 2.0 API, copyright Sun Microsystems Inc

Table SystemException

Byte	Description
1	AID Illegal Use

Methods

o **getException**

```
public static SystemException getException(byte reason)
```

Returns the system instance of SystemException

All Packages Class Hierarchy This Package Previous Next Index

Class javacard.framework.UserException

```
java.lang.Object
   |
   +----java.lang.Throwable
           |
           +----java.lang.Exception
                   |
                   +----javacard.framework.CardException
                           |
                           +----javacard.framework.UserException
```

public class **UserException**
extends CardException

The `UserException` signals a User exception. This class also provides a resource-saving mechanism for user exceptions by re-using a singleton instance.

Constructor Index

o **UserException**(byte)
  Constructs a UserException.

Method Index

o **getException**(byte)
  Returns the system instance of UserException

Constructors

o **UserException**

```
public UserException(byte reason)
```

Constructs a UserException.

Methods

o **getException**

```
public static UserException getException(byte reason)
```

Returns the system instance of UserException

Java Card 2.0 API, copyright Sun Microsystems Inc

All Packages Class Hierarchy This Package Previous Next Index

Class javacard.framework.Util

```
java.lang.Object
   |
   +----javacard.framework.Util
```

public class **Util**
extends Object

The `util` class contains common utility functions. Some of the methods may be implemented as native functions for performance reasons.

Constructor Index

o **Util**()

Method Index

o **arrayCompare**(byte[], short, byte[], short, short)
  Compares an array from the specified source array, beginning at the specified position, with the specified position of the destination array.
o **arrayCopy**(byte[], short, byte[], short, short)
  Copies an array from the specified source array, beginning at the specified position, to the specified position of the destination array.

Constructors

o **Util**

```
public Util()
```

Methods

o **arrayCopy**

```
public static short arrayCopy(byte src[],
                              short srcOff,
                              byte dest[],
                              short destOff,
                              short length)
```

Copies an array from the specified source array, beginning at the specified position, to the specified position of the destination array.

Parameters:
  src - source byte array.
  srcOff - offset within source byte array to start copy from.
  dest - destination byte array.

destOff - offset within destination byte array to start copy into.
length - byte length to be copied.

Returns:
actualLength - the number of bytes copied. 0 if insufficient space.

o **arrayCompare**

```
public static boolean arrayCompare(byte src[],
                                   short srcOff,
                                   byte dest[],
                                   short destOff,
                                   short length)
```

Compares an array from the specified source array, beginning at the specified position, with the specified position of the destination array. Returns true if identical, false otherwise.

Parameters:
src - source byte array.
srcOff - offset within source byte array to start copy from.
dest - destination byte array.
destOff - offset within destination byte array to start copy into.
length - byte length to be copied.

Returns:
boolean - the result of the comparison.

All Packages Class Hierarchy This Package Previous Next Index

All Packages Class Hierarchy Index

---

package javacardx.framework

Class Index

- CyclicFile
- DedicatedFile
- ElementaryFile
- File
- FileSystem
- LinearFixedFile
- LinearVariableFile
- TransparentFile

All Packages Class Hierarchy This Package Previous Next Index

Class javacardx.framework.CyclicFile

```
java.lang.Object
   |
   +----javacardx.framework.File
           |
           +----javacardx.framework.ElementaryFile
                   |
                   +----javacardx.framework.LinearVariableFile
                           |
                           +----javacardx.framework.LinearFixedFile
                                   |
                                   +----javacardx.framework.CyclicFile
```

public class **CyclicFile**
extends LinearFixedFile

Cyclic fixed-length record files.

Constructor Index

o **CyclicFile**(short, byte, byte)
Constructor.

Method Index

o **addRecord**(byte[])
Add (append) a new record.
o **addRecord**(short)
Add (append) a new record.
o **getNewFirstRecord**()
Recycles the oldest record as the new most recent record (record number 1).
o **getRecord**(byte)
Get the record byte array for the specified record.
o **increaseMaxNumRecords**(byte)
Increase the maximum number of records.

Constructors

o **CyclicFile**

```
public CyclicFile(short FID,
                  byte maxNumRecords,
                  byte recordLength)
```

Constructor.

Java Card 2.0 API, copyright Sun Microsystems Inc

Parameters:
FID - the file's 16-bit FID
maxNumRecords - the maximum number of records in this file
recordLength - the fixed record length for this file

Methods

o **getRecord**

```
public byte[] getRecord(byte recordNum)
```

Get the record byte array for the specified record. Records are in the reverse order that they were updated in the file.

Parameters:
recordNum - the number (most recent record = 1) of the record. 0 means get "current" record.
Returns:
record (or null)
Overrides:
getRecord in class LinearVariableFile

o **getNewFirstRecord**

```
public byte[] getNewFirstRecord()
```

Recycles the oldest record as the new most recent record (record number 1).

Returns:
record, the byte array contains the old data and must be updated by the caller.

o **increaseMaxNumRecords**

```
public boolean increaseMaxNumRecords(byte number)
```

Increase the maximum number of records. Not allowed for cyclic files.

Returns:
false
Overrides:
increaseMaxNumRecords in class LinearVariableFile

o **addRecord**

```
public void addRecord(byte record[])
```

Add (append) a new record. Not allowed for cyclic files.

Overrides:
addRecord in class LinearFixedFile

o **addRecord**

```
public void addRecord(short length)
```

Add (append) a new record. Not allowed for cyclic files.

Overrides:
addRecord in class LinearFixedFile

All Packages Class Hierarchy This Package Previous Next Index

Java Card 2.0 API, copyright Sun Microsystems Inc

All Packages Class Hierarchy This Package Previous Next Index

Class javacardx.framework.DedicatedFile

```
java.lang.Object
   |
   +----javacardx.framework.File
           |
           +----javacardx.framework.DedicatedFile
```

public class **DedicatedFile**
extends File

DFs.

Constructor Index

o **DedicatedFile**(short, byte[], byte)
Constructor.

Method Index

o **addChildFile**(File)
Add (append) a new chile file to the file.
o **findElementaryFile**(byte)
Under this DF, find the EF with the specified SFI.
o **getChildFile**(byte)
Get the File object for the specified child file.
o **getMaxChildFiles**()
Get the maximum number of child files in this DF.
o **getName**()
Get the file's name.
o **getNumChildFiles**()
Get the actual number of child files in this DF.
o **increaseMaxChildFiles**(byte)
Increase the maximum number of child files in this DF.

Constructors

o **DedicatedFile**

```
public DedicatedFile(short FID,
                     byte name[],
                     byte maxChildFiles)
```

Constructor.

Parameters:
FID - the file's 16-bit FID

name - the name byte array of this file (or null if none)
maxChildFiles - the maximum number of child files for this DF

Methods

o **getName**

```
public byte[] getName()
```

Get the file's name.

Returns:
name

o **getMaxChildFiles**

```
public byte getMaxChildFiles()
```

Get the maximum number of child files in this DF.

Returns:
maxChildFiles

o **increaseMaxChildFiles**

```
public boolean increaseMaxChildFiles(byte number)
```

Increase the maximum number of child files in this DF.

Returns:
true if the increase was successful, false otherwise

o **getNumChildFiles**

```
public byte getNumChildFiles()
```

Get the actual number of child files in this DF.

Returns:
numChildFiles

o **getChildFile**

```
public File getChildFile(byte childNum)
```

Get the File object for the specified child file. Child file are in the order that they were added to the file.

Parameters:
childNum - the index (first child = 1) of the child file.
Returns:
File (or null)

Java Card 2.0 API, copyright Sun Microsystems Inc

o **addChildFile**

```
public void addChildFile(File child)
```

Add (append) a new chile file to the file.

Parameters:
child - the reference to the child file.

o **findElementaryFile**

```
public ElementaryFile findElementaryFile(byte SFI)
```

Under this DF, find the EF with the specified SFI. Resets the current record pointer

Parameters:
SFI - the short file identifier
Returns:
the EF selected (or null)

All Packages Class Hierarchy This Package Previous Next Index

Class javacardx.framework.ElementaryFile

```
java.lang.Object
   |
   +----javacardx.framework.File
           |
           +----javacardx.framework.ElementaryFile
```

public abstract class **ElementaryFile**
extends File

This is the abstract base class for all elementary files (EFs). For simplicity, the SFI of an EF is the last 5 bits of the FID.

Constructor Index

o **ElementaryFile**()

Method Index

o **getSFI**()
Get this file's 5-bit SFI.

Constructors

o **ElementaryFile**

```
public ElementaryFile()
```

Methods

o **getSFI**

```
public byte getSFI()
```

Get this file's 5-bit SFI. The SFI is the last 5 bits of the FID.

Returns:
SFI

Class javacardx.framework.File

```
java.lang.Object
   |
   +----javacardx.framework.File
```

public abstract class **File**
extends Object

This is the abstract base class for all files (DFs and EFs) in an applet's file system. See `FileSystem` class and ISO 7816-4 for additional details.

All files have:

- a FID (16-bit file identifer)
- a parent DF (which is null if the file has no parent)
- external read/write security attributes

Since an explicit security model is not defined in 7816-4, this class defines a simple yet extensible scheme. Each file has two attributes, one for "external read access" (such as a READ RECORD command) and one for "external write access" (such as a WRITE BINARY command). In each attribute the user can set one of the ALLOW_xxx values to specify what conditions must be true in order to allow that type of access (see tables below).

Table - Access Types

Constant	Value	Description
ACCESS_READ	0	External read
ACCESS_WRITE	1	External write

Table - Allow Types

Constant	Value	Description
ALLOW_ANY	0	Any external access allowed
ALLOW_AUTH1	1	External access allowed only if Auth1 flag is true
ALLOW_AUTH2	2	External access allowed only if Auth2 flag is true
ALLOW_NONE	3	No external access allowed

For example, a value of 0 in the read attribute means that this file can be read externally at any time. A value of 3 in the write attribute means that this file never be written externally.

The two Auth flags are defined in the `FileSystem` class and allow for a certain amount of applet customization. When a security attribute is set to 1 or 2, the access is allowed only if the appropriate Auth flags maintained by the FileSystem is true. For example, a applet may set Auth1 when a valid PIN is presented. After that point, all files with ALLOW_AUTH1 in the read attribute can now be read externally.

Note that this security checking is done programatically and is not enforced by the VM. That is, the `FileSystem.readRecord` method will perform read access checking on the accessed file. But internal applet access to the file is not checked unless the applet specifically does so, using the `isAllowed` method in this class.

Variable Index

o **ACCESS_READ**
o **ACCESS_WRITE**
o **ALLOW_ANY**
o **ALLOW_AUTH1**
o **ALLOW_AUTH2**
o **ALLOW_NONE**

Constructor Index

o **File**()

Method Index

o **getFCI**()
Get this file’s FCI (if any).
o **getFID**()
Get this file’s 16-bit FID.
o **getParent**()
Get this file’s parent DF.
o **getSecurity**(byte)
Get this file’s external read or write security.
o **isAllowed**(byte)
Check this file’s external read or write security.
o **setFCI**(byte[])
Set this file’s FCI.
o **setSecurity**(byte, byte)
Set this file’s external read or write security.

Variables

o **ACCESS_READ**

```
public static final byte ACCESS_READ
```

o **ACCESS_WRITE**

Java Card 2.0 API, copyright Sun Microsystems Inc

```
public static final byte ACCESS_WRITE
```

o **ALLOW_ANY**

```
public static final byte ALLOW_ANY
```

o **ALLOW_AUTH1**

```
public static final byte ALLOW_AUTH1
```

o **ALLOW_AUTH2**

```
public static final byte ALLOW_AUTH2
```

o **ALLOW_NONE**

```
public static final byte ALLOW_NONE
```

Constructors

o **File**

```
public File()
```

Methods

o **getFID**

```
public short getFID()
```

Get this file’s 16-bit FID.

Returns:
FID

o **getParent**

```
public DedicatedFile getParent()
```

Get this file’s parent DF.

Returns:
parent DF (or null)

o **getFCI**

```
public byte[] getFCI()
```

Get this file’s FCI (if any).

Returns:
the FCI byte array (or null)

o **setFCI**

```
public void setFCI(byte FCI[])
```

Set this file's FCI.

Parameters:
FCI - the byte array containing the FCI

o **getSecurity**

```
public byte getSecurity(byte access)
```

Get this file's external read or write security.

Parameters:
access - ACCESS_READ or ACCESS_WRITE
Returns:
one of the ALLOW_xxx constants

o **setSecurity**

```
public void setSecurity(byte access,
                        byte allow)
```

Set this file's external read or write security.

Parameters:
access - ACCESS_READ or ACCESS_WRITE
allow - one of the ALLOW_xxx constants

o **isAllowed**

```
public boolean isAllowed(byte access)
```

Check this file's external read or write security. This method always returns true for ALLOW_ANY and false for ALLOW_NONE. For ALLOW_AUTHn, it returns the state of the Auth flag maintained in the `FileSystem` class.

Parameters:
access - ACCESS_READ or ACCESS_WRITE
Returns:
true if the specified access is allowed, false otherwise

All Packages Class Hierarchy This Package Previous Next Index

All Packages Class Hierarchy This Package Previous Next Index

---

Class javacardx.framework.FileSystem

```
java.lang.Object
   |
   +----javacardx.framework.File
           |
           +----javacardx.framework.DedicatedFile
                   |
                   +----javacardx.framework.FileSystem
```

---

public class **FileSystem**
extends DedicatedFile

This is a subclass of DedicatedFile which is the "root" DF of the applet. It contains several kinds of methods:

- get and set state values: Auth1 and Auth2 flags and current DF, EF, and record
- find files via name or FID
- handle ISO 7618-4 file-oriented APDUs

Variable Index

o **DATAFILE_FID**
o **FIND_ANY**
o **FIND_CHILD**
o **FIND_CHILD_DF**
o **FIND_CHILD_EF**
o **FIND_PARENT**

Constructor Index

o **FileSystem**(byte)
Constructs an instance of an ISO 7816-4 file system.

Method Index

o **appendRecord**(APDU)
Handles APPEND RECORD command APDU as specified by ISO 7816-4.
o **create**(APDU)
Handles create APDUs.
o **eraseBinary**(APDU)
Handles ERASE BINARY command APDU as specified by ISO 7816-4.
o **findDedicatedFile**(byte[], short, byte)
Find the DF with the specified name.
o **findFile**(byte, short)
According to findType, find the file with the specified FID.

o **getAuthFlag**(byte)
  Get authorization flag.
o **getCurrentDedicatedFile**()
  Get current DF.
o **getCurrentElementaryFile**()
  Get current EF.
o **getCurrentRecNum**()
  Get current record number.
o **getData**(APDU)
  Handles GET DATA command APDU as specified by ISO 7816-4.
o **process**(APDU)
  Handles FileSystem APDUs as specified by ISO 7816-4.
o **putData**(APDU)
  Handles PUT DATA command APDU as specified by ISO 7816-4.
o **readBinary**(APDU)
  Handles READ BINARY command APDU as specified by ISO 7816-4.
o **readRecord**(APDU)
  Handles READ RECORD command APDU as specified by ISO 7816-4.
o **select**(APDU, boolean)
  This method handles SELECT APDUs as specified by ISO 7816-4.
o **selectFile**(File)
  Make the specified file the current DF or current EF.
o **setAuthFlag**(byte, boolean)
  Set authorization flag.
o **setCurrentDedicatedFile**(DedicatedFile)
  Set current DF.
o **setCurrentElementaryFile**(ElementaryFile)
  Set current EF.
o **setCurrentRecNum**(byte)
  Set current record number.
o **updateBinary**(APDU)
  Handles UPDATE BINARY command APDU as specified by ISO 7816-4.
o **updateRecord**(APDU)
  Handles UPDATE RECORD command APDU as specified by ISO 7816-4.
o **writeBinary**(APDU)
  Handles WRITE BINARY command APDU as specified by ISO 7816-4.
o **writeRecord**(APDU)
  Handles WRITE RECORD command APDU as specified by ISO 7816-4.

Variables

o **FIND_ANY**

```
public static final byte FIND_ANY
```

o **FIND_CHILD_DF**

```
public static final byte FIND_CHILD_DF
```

o **FIND_CHILD_EF**


```
public static final byte FIND_CHILD_EF
```

o **FIND_PARENT**

```
public static final byte FIND_PARENT
```

o **FIND_CHILD**

```
public static final byte FIND_CHILD
```

o **DATAFILE_FID**

```
public static final short DATAFILE_FID
```

Constructors

o **FileSystem**

```
public FileSystem(byte maxChildFiles)
```

Constructs an instance of an ISO 7816-4 file system.

Parameters:
  maxChildFiles - the maximum number of child files for this DF

Methods

o **getCurrentDedicatedFile**

```
public DedicatedFile getCurrentDedicatedFile()
```

Get current DF.

Returns:
  DF the DF reference maintained in RAM

o **setCurrentDedicatedFile**

```
public void setCurrentDedicatedFile(DedicatedFile DF)
```

Set current DF.

Parameters:
  DF - the DF reference maintained in RAM

o **getCurrentElementaryFile**

```
public ElementaryFile getCurrentElementaryFile()
```

Get current EF.

Returns:
  EF the EF reference maintained in RAM

o **setCurrentElementaryFile**

```
public void setCurrentElementaryFile(ElementaryFile EF)
```

Set current EF.

Parameters:
EF - the EF reference maintained in RAM

o **getCurrentRecNum**

```
public byte getCurrentRecNum()
```

Get current record number.

Returns:
recNum the recNum maintained in RAM

o **setCurrentRecNum**

```
public void setCurrentRecNum(byte recNum)
```

Set current record number.

Parameters:
recNum - the recNum maintained in RAM

o **getAuthFlag**

```
public boolean getAuthFlag(byte number)
```

Get authorization flag.

Parameters:
number - the number (1 or 2) of the authorization flag in RAM
Returns:
the value of the authorization flag

o **setAuthFlag**

```
public void setAuthFlag(byte number,
                        boolean value)
```

Set authorization flag.

Parameters:
number - the number (1 or 2) of the authorization flag in RAM
the - value of the authorization flag

o **findDedicatedFile**

```
public DedicatedFile findDedicatedFile(byte data[],
                                       short offset,
                                       byte length)
```


Find the DF with the specified name.

Parameters:
data - a byte array containing the name
offset - byte offset of name in data
length - byte length of name in data
Returns:
the DedicatedFile found (or null)

o **findFile**

```
public File findFile(byte findType,
                     short FID)
```

According to findType, find the file with the specified FID. The FIND_xxx constants allow different ways to find a file based as specified by the ISO 7816-4 SELECT command.

Parameters:
findType - one of the FIND_xxx constants
FID - the file identifier
Returns:
the File found (or null)

o **selectFile**

```
public void selectFile(File file)
```

Make the specified file the current DF or current EF.

Parameters:
file - the file reference

o **select**

```
public short select(APDU apdu,
                    boolean appletSelected)
```

This method handles SELECT APDUs as specified by ISO 7816-4. The command data is in the APDU buffer since the Executive has already read and examined it.

Parameters:
apdu - the APDU object
appletSelected - true if this applet has just been selected, false otherwise
Returns:
status the resulting SW

o **process**

```
public short process(APDU apdu)
```

Handles FileSystem APDUs as specified by ISO 7816-4. This method simply dispatches to other methods in this class (except SELECT) based on the INS in the APDU.

Parameters:
apdu - the APDU object

Returns:
status the resulting SW

o **create**

```
public short create(APDU apdu)
```

Handles create APDUs. TBD !! create is not defined by ISO, should we define it?

Parameters:
apdu - the APDU object
Returns:
status the resulting SW

o **readBinary**

```
protected short readBinary(APDU apdu)
```

Handles READ BINARY command APDU as specified by ISO 7816-4.

Parameters:
apdu - the APDU object
Returns:
status the resulting SW

o **writeBinary**

```
protected short writeBinary(APDU apdu)
```

Handles WRITE BINARY command APDU as specified by ISO 7816-4.

Parameters:
apdu - the APDU object
Returns:
status the resulting SW

o **updateBinary**

```
protected short updateBinary(APDU apdu)
```

Handles UPDATE BINARY command APDU as specified by ISO 7816-4.

Parameters:
apdu - the APDU object
Returns:
status the resulting SW

o **eraseBinary**

```
protected short eraseBinary(APDU apdu)
```

Handles ERASE BINARY command APDU as specified by ISO 7816-4.

Parameters:
apdu - the APDU object

Returns:
status the resulting SW

o **readRecord**

```
protected short readRecord(APDU apdu)
```

Handles READ RECORD command APDU as specified by ISO 7816-4.

Parameters:
apdu - the APDU object
Returns:
status the resulting SW

o **writeRecord**

```
protected short writeRecord(APDU apdu)
```

Handles WRITE RECORD command APDU as specified by ISO 7816-4.

Parameters:
apdu - the APDU object
Returns:
status the resulting SW

o **updateRecord**

```
protected short updateRecord(APDU apdu)
```

Handles UPDATE RECORD command APDU as specified by ISO 7816-4.

Parameters:
apdu - the APDU object
Returns:
status the resulting SW

o **appendRecord**

```
protected short appendRecord(APDU apdu)
```

Handles APPEND RECORD command APDU as specified by ISO 7816-4.

Parameters:
apdu - the APDU object
Returns:
status the resulting SW

o **getData**

```
protected short getData(APDU apdu)
```

Handles GET DATA command APDU as specified by ISO 7816-4.

Parameters:
apdu - the APDU object

Returns:
status the resulting SW

o **putData**

```
protected short putData(APDU apdu)
```

Handles PUT DATA command APDU as specified by ISO 7816-4.

Parameters:
apdu - the APDU object
Returns:
status the resulting SW

All Packages Class Hierarchy This Package Previous Next Index

Java Card 2.0 API, copyright Sun Microsystems Inc

All Packages Class Hierarchy This Package Previous Next Index

---

Class javacardx.framework.LinearFixedFile

```
java.lang.Object
   |
   +----javacardx.framework.File
           |
           +----javacardx.framework.ElementaryFile
                   |
                   +----javacardx.framework.LinearVariableFile
                           |
                           +----javacardx.framework.LinearFixedFile
```

---

public class **LinearFixedFile**
extends LinearVariableFile

Linear fixed-length record files.

Constructor Index

o **LinearFixedFile**(short, byte, byte)
Constructor.

Method Index

o **addRecord**(byte[])
Add (append) a new record to the file.
o **addRecord**(short)
Add (append) a new record to the file.

Constructors

o **LinearFixedFile**

```
public LinearFixedFile(short FID,
                       byte maxNumRecords,
                       byte recordLength)
```

Constructor.

Parameters:
FID - the file's 16-bit FID
maxNumRecords - the maximum number of records in this file
recordLength - the fixed record length for this file

DRAFT

Methods

o **addRecord**

```
public void addRecord(byte record[])
```

Add (append) a new record to the file. Note that the record reference is stored in the file object. A copy of the record byte array is not made.

Parameters:
record - the record byte array
Overrides:
addRecord in class LinearVariableFile

o **addRecord**

```
public void addRecord(short length)
```

Add (append) a new record to the file. This creates a new record byte array.

Parameters:
length - the size of the new record byte array to be added
Overrides:
addRecord in class LinearVariableFile

Java Card 2.0 API, copyright Sun Microsystems Inc

Class javacardx.framework.LinearVariableFile

```
java.lang.Object
   |
   +----javacardx.framework.File
           |
           +----javacardx.framework.ElementaryFile
                   |
                   +----javacardx.framework.LinearVariableFile
```

public class **LinearVariableFile**
extends ElementaryFile

This is the class for all linear variable-length files, and the base class for linear variable-fixed and cyclic files.

Table - Starting point and direction (for getRecord method)

Constant	Value	Description
DIRECTION_FIRST	0	Start at first record in file
DIRECTION_LAST	1	Start at last record in file
DIRECTION_NEXT	2	Start at current record and move forward
DIRECTION_PREV	3	Start at current record and move backward

Variable Index

o **DIRECTION_FIRST**
o **DIRECTION_LAST**
o **DIRECTION_NEXT**
o **DIRECTION_PREV**

Constructor Index

o **LinearVariableFile**(short, byte)
Constructor.

Method Index

o **addRecord**(byte[])
Add (append) a new record to the file.
o **addRecord**(short)
Add (append) a new record to the file.

o **getMaxNumRecords**()
Get the maximum number of records in this file.
o **getNumRecords**()
Get the actual number of records in this file.
o **getRecord**(byte)
Get the record byte array for the specified record number.
o **getRecord**(byte, byte, byte)
Get the record byte array for the specified record identifier using the specified starting point and direction.
o **increaseMaxNumRecords**(byte)
Increase the maximum number of records in this file.

Variables

o **DIRECTION_FIRST**

```
public static final byte DIRECTION_FIRST
```

o **DIRECTION_LAST**

```
public static final byte DIRECTION_LAST
```

o **DIRECTION_NEXT**

```
public static final byte DIRECTION_NEXT
```

o **DIRECTION_PREV**

```
public static final byte DIRECTION_PREV
```

Constructors

o **LinearVariableFile**

```
public LinearVariableFile(short FID,
                          byte maxNumRecords)
```

Constructor.

Parameters:
FID - the file's 16-bit FID
maxNumRecords - the maximum number of records in this file

Methods

o **getMaxNumRecords**

```
public byte getMaxNumRecords()
```

Get the maximum number of records in this file.

Returns:
maxNumRecords

o **increaseMaxNumRecords**

```
public boolean increaseMaxNumRecords(byte number)
```

Increase the maximum number of records in this file.

Returns:
true if the increase was successful, false otherwise

o **getNumRecords**

```
public byte getNumRecords()
```

Get the actual number of records in this file.

Returns:
numRecords

o **addRecord**

```
public void addRecord(byte record[])
```

Add (append) a new record to the file. Note that the record reference is stored in the file object. A copy of the record byte array is not made.

Parameters:
record - the record byte array

o **addRecord**

```
public void addRecord(short length)
```

Add (append) a new record to the file. This creates a new record byte array.

Parameters:
length - the size of the new record byte array to be added

o **getRecord**

```
public byte[] getRecord(byte recordNum)
```

Get the record byte array for the specified record number. This is a reference to the actual file data, not a copy of the file data. Records are in the order that they were added to the file.

Parameters:
recordNum - the index (first record = 1) of the record. 0 means get "current" record.
Returns:
record (or null)

o **getRecord**

```
public byte[] getRecord(byte direction,
                        byte firstByte,
                        byte secondByte)
```

Get the record byte array for the specified record identifier using the specified starting point and direction. Records are in the order that they were added to the file.

Parameters:
direction - one of the DIRECTION_xxx constants
firstByte - if non-0, the record's first byte must match this value
secondByte - if non-0, the record's second byte must match this value
Returns:
record (or null)

All Packages Class Hierarchy This Package Previous Next Index

Java Card 2.0 API, copyright Sun Microsystems Inc

All Packages Class Hierarchy This Package Previous Next Index

---

Class javacardx.framework.TransparentFile

```
java.lang.Object
   |
   +----javacardx.framework.File
           |
           +----javacardx.framework.ElementaryFile
                   |
                   +----javacardx.framework.TransparentFile
```

---

public class **TransparentFile**
extends ElementaryFile

This is the class for all transparent files.

Constructor Index

o **TransparentFile**(short, byte[])
Constructor, with data byte array specified.
o **TransparentFile**(short, short)
Constructor, with data byte array size specified.

Method Index

o **getData**()
Gets the byte array containing the data for this file.

Constructors

o **TransparentFile**

```
 public TransparentFile(short FID,
                        byte data[])
```

Constructor, with data byte array specified. Note that the data reference is stored in the file object. A copy of the data byte array is not made.

Parameters:
FID - the file's 16-bit FID
data - the data byte array of this file

o **TransparentFile**

```
 public TransparentFile(short FID,
                        short length)
```

Constructor, with data byte array size specified. This creates a new data byte array.

Parameters:
FID - the file's 16-bit FID
length - the length of the data byte array

Methods

o **getData**

```
public byte[] getData()
```

Gets the byte array containing the data for this file. This is a reference to the actual file data, not a copy of the file data.

Returns:
data

All Packages Class Hierarchy This Package Previous Next Index

All Packages Class Hierarchy Index

---

package javacardx.crypto

Class Index

- AsymKey
- DES_CBC_Key
- DES_Key
- Key
- MessageDigest
- RSA_CRT_Key
- RSA_Key
- RandomData
- Sha1MessageDigest
- SymKey

Exception Index

- CryptoException

Class javacardx.crypto.AsymKey

```
java.lang.Object
   |
   +----javacardx.crypto.AsymKey
```

public abstract class **AsymKey**
extends Object

The AsymKey class is the base class for keys used in asymetric algorithms.

Constructor Index

o **AsymKey**()

Method Index

o **Sign**(byte[], short, short, byte[], short)
Signs data using this key.
o **Verify**(byte[], short, short, byte[], short, short)
Verifies signed data using this key.

Constructors

o **AsymKey**

```
public AsymKey()
```

Methods

o **Sign**

```
public abstract native void Sign(byte inBuff[],
                                 short inOffset,
                                 short inLength,
                                 byte outBuff[],
                                 short outOffset)
```

Signs data using this key.

Parameters:
inBuff - the input buffer containing this key.
inOffset - the offset into the input buffer
inLength - the length
outBuff - the output buffer, may be the same as the input buffer
outOffset - the offset into the output buffer

Java Card 2.0 API, copyright Sun Microsystems Inc

o **Verify**

```
public abstract native boolean Verify(byte msgDigest[],
                                      short msgOffset,
                                      short msgLength,
                                      byte signedData[],
                                      short signOffset,
                                      short signLength)
```

Verifies signed data using this key.

Parameters:
msgDigest - the buffer containing the hash result.
msgOffset - the offset into the hash result buffer
msgLength - the length of the hash
signedData - the buffer containing the signed data.
signOffset - the offset into the signed data buffer
signLength - the of the signed data buffer
Returns:
true if the data is properly signed.

All Packages Class Hierarchy This Package Previous Next Index

Class javacardx.crypto.CryptoException

```
java.lang.Object
   |
   +----java.lang.Throwable
           |
           +----java.lang.Exception
                   |
                   +----java.lang.RuntimeException
                           |
                           +----javacardx.crypto.CryptoException
```

public class **CryptoException**
extends RuntimeException

CryptoException. The types of crypto exceptions are TBD.

Constructor Index

o **CryptoException**()

Constructors

o **CryptoException**

```
public CryptoException()
```


Class javacardx.crypto.DES_CBC_Key

```
java.lang.Object
   |
   +----javacardx.crypto.Key
           |
           +----javacardx.crypto.SymKey
                   |
                   +----javacardx.crypto.DES_CBC_Key
```

public class **DES_CBC_Key**
extends SymKey

DES_key contains a DES key and provides operations for encryption and decryption using that key. The length of the key determines whether single or triple DES is performed. DES operates on a block size of 8 and all input parameters to these methods are expected to be multiples of 8. In each case the caller is responisble for any padding. Chaining the results of one call to a CBC mode function to another, is the responsiblity of the caller.

Constructor Index

o **DES_CBC_Key**(byte)

Method Index

o **clearIV**()
  Clears the initialization vector used in CBC mode DES operations.
o **decrypt**(byte[], short, short, byte[], short)
  Decrypts using DES in CBC mode.
o **encrypt**(byte[], short, short, byte[], short)
  Encrypts using DES in CBC mode.
o **generateMAC**(byte[], short, short, byte[], short, byte)
  Generates a MAC using DES in CBC mode.
o **setIV**(byte[], short, short)
  Sets the initialization vector used in CBC mode DES operations.
o **verifyMAC**(byte[], short, short, byte[], short, short)
  Verifies signed data using this key.

Constructors

o **DES_CBC_Key**

```
public DES_CBC_Key(byte length)
```

Parameters:
length - the length of the key in bytes

Methods

o **setIV**

```
public native void setIV(byte buff[],
                         short offset,
                         short length)
```

Sets the initialization vector used in CBC mode DES operations.

Parameters:
buff - the input buffer
offset - the offset into the input buffer at which to begin decryption
length - the length to encrypt

o **clearIV**

```
public native void clearIV()
```

Clears the initialization vector used in CBC mode DES operations.

o **encrypt**

```
public native void encrypt(byte inBuff[],
                           short inOffset,
                           short inLength,
                           byte outBuff[],
                           short outOffset)
```

Encrypts using DES in CBC mode. If the key is 8 bytes long single DES is performed. If the key is 16 bytes long triple DES is performed.

Parameters:
inBuff - the input buffer
inOffset - the offset into the input buffer at which to begin encryption
inLength - the length to encrypt
outBuff - the output buffer, may be the same as the input buffer
outOffset - the offset into the output buffer
Overrides:
encrypt in class SymKey

o **decrypt**

```
public native void decrypt(byte inBuff[],
                           short inOffset,
                           short inLength,
                           byte outBuff[],
                           short outOffset)
```

Decrypts using DES in CBC mode. If the key is 8 bytes long single DES is performed. If the key is 16 bytes long triple DES is performed.

Parameters:
inBuff - the input buffer
inOffset - the offset into the input buffer at which to begin decryption
inLength - the length to encrypt

outBuff - the output buffer, may be the same as the input buffer
outOffset - the offset into the output buffer
Overrides:
decrypt in class SymKey

o **generateMAC**

```
public native void generateMAC(byte inBuff[],
                               short inOffset,
                               short inLength,
                               byte outBuff[],
                               short outOffset,
                               byte length)
```

Generates a MAC using DES in CBC mode. If the key is 8 bytes long single DES is performed. If the key is 16 bytes long triple DES is performed.

Parameters:
inBuff - the input buffer
inOffset - the offset into the input buffer at which to begin encryption
inLength - the length to encrypt
outBuff - the output buffer, may be the same as the input buffer
outOffset - the offset into the output buffer
outLength - the length of the MAC to generate

o **verifyMAC**

```
public native boolean verifyMAC(byte macBuffer[],
                                short macOffset,
                                short macLength,
                                byte inBuff[],
                                short inOffset,
                                short inLength)
```

Verifies signed data using this key.

Parameters:
msgDigest - the buffer containing the MAC to verify.
msgOffset - the offset into the MAC buffer
msgLength - the length of the MAC
signedData - the buffer containing the input data.
signOffset - the offset into the input data buffer
signLength - the of the input data buffer
Returns:
true if the data is MAC is correct.

Class javacardx.crypto.DES_Key

```
java.lang.Object
   |
   +----javacardx.crypto.Key
           |
           +----javacardx.crypto.SymKey
                   |
                   +----javacardx.crypto.DES_Key
```

public class **DES_Key**
extends SymKey

DES_Key contains a DES key and provides operations for encryption and decryption using that key. The length of the key determines whether single or triple DES is performed. DES operates on a block size of 8 and all input parameters to these methods are expected to be multiples of 8. In each case the caller is responisble for any padding. Keys in this class are used for ECB mode operation only.

See Also:
DES_CBC_Key

Constructor Index

o **DES_Key**(byte)

Method Index

o **decrypt**(byte[], short, short, byte[], short)
Decrypts using DES in ECB mode.
o **encrypt**(byte[], short, short, byte[], short)
Encrypts using DES in ECB mode.

Constructors

o **DES_Key**

```
public DES_Key(byte length)
```

Parameters:
length - the length of the key in bytes

Methods

o **encrypt**

Java Card 2.0 API, copyright Sun Microsystems Inc

```
public native void encrypt(byte inBuff[],
                           short inOffset,
                           short inLength,
                           byte outBuff[],
                           short outOffset)
```

Encrypts using DES in ECB mode. If the key is 8 bytes long single DES is performed. If the key is 16 bytes long triple DES is performed.

Parameters:
inBuff - the input buffer
inOffset - the offset into the input buffer at which to begin encryption
inLength - the length to encrypt
outBuff - the output buffer, may be the same as the input buffer
outOffset - the offset into the output buffer
Overrides:
encrypt in class SymKey

o **decrypt**

```
public native void decrypt(byte inBuff[],
                           short inOffset,
                           short inLength,
                           byte outBuff[],
                           short outOffset)
```

Decrypts using DES in ECB mode. If the key is 8 bytes long single DES is performed. If the key is 16 bytes long triple DES is performed.

Parameters:
inBuff - the input buffer
inOffset - the offset into the input buffer at which to begin decryption
inLength - the length to decrypt
outBuff - the output buffer, may be the same as the input buffer
outOffset - the offset into the output buffer
Overrides:
decrypt in class SymKey

All Packages Class Hierarchy This Package Previous Next Index

Class javacardx.crypto.Key

```
java.lang.Object
   |
   +----javacardx.crypto.Key
```

public abstract class **Key**
extends Object

The Key class is the base class for keys.

Constructor Index

o **Key**(short)
constructs a key with a specific bit length

Method Index

o **bitLength**()
the length of the key in bits
o **clearKey**()
clears the key and sets its initialized state to false.
o **isInitialized**()
Reports the initialized state of the key.

Constructors

o **Key**

```
public Key(short length)
```

constructs a key with a specific bit length

Parameters:
length - the length of the key in bits

Methods

o **bitLength**

```
public final short bitLength()
```

the length of the key in bits

Returns:
the length of the key in bits

Java Card 2.0 API, copyright Sun Microsystems Inc

o **isInitialized**

```
public boolean isInitialized()
```

Reports the initialized state of the key. Keys must be initialized before being used.

Returns:
true if the key has been initialized.

o **clearKey**

```
public void clearKey()
```

clears the key and sets its initialized state to false.

All Packages Class Hierarchy This Package Previous Next Index

Class javacardx.crypto.MessageDigest

```
java.lang.Object
   |
   +----javacardx.crypto.MessageDigest
```

public class **MessageDigest**
extends Object

The MessageDigest class is the base class for hashing algorthims.

Constructor Index

o **MessageDigest**()

Method Index

o **generateDigest**(byte[], short, short, byte[], short, short)
generates a hash of the input data.

Constructors

o **MessageDigest**

```
public MessageDigest()
```

Methods

o **generateDigest**

```
public static native void generateDigest(byte inBuff[],
                                          short inOffset,
                                          short inLength,
                                          byte outBuff[],
                                          short outOffset,
                                          short outLength)
```

generates a hash of the input data.

Parameters:
inBuff - the input buffer
inOffset - the offset into the input buffer at which to begin encryption
inLength - the length to encrypt
outBuff - the output buffer, may be the same as the input buffer
outOffset - the offset into the output buffer
outLength - the length of resulting hash

All Packages Class Hierarchy This Package Previous Next Index

Class javacardx.crypto.RSA_CRT_Key

```
java.lang.Object
   |
   +----javacardx.crypto.Key
           |
           +----javacardx.crypto.RSA_CRT_Key
```

---

public class **RSA_CRT_Key**
extends Key

The RSA_CRT_Key class is used to sign data using the RSA algorithm in its Chinese Remainder Theorem form.

Let $S = m^d \bmod n$, where m is the data to be signed, d is the private key exponent, and n is private key modulus composed of two prime numbers p and q. The following names are used in the setter methods in this class:

P, the prime factor p
Q, the prime factor q.
PQ = $p^{-1} \bmod q$
DP1 = $d \bmod (p - 1)$
DQ1 = $d \bmod (q - 1)$

See Also:
RSA_Key

Constructor Index

o **RSA_CRT_Key**()
Constructs an empty RSA_CRT_Key

Method Index

o **isInitialized**()
Reports the initialized state of the key.
o **setDP1**(byte[], short, short)
Sets the value of the DP1 parameter.
o **setDQ1**(byte[], short, short)
Sets the value of the P key.
o **setP**(byte[], short, short)
Sets the value of the P parameter.
o **setPQ**(byte[], short, short)
Sets the value of the PQ parameter.
o **setQ**(byte[], short, short)
Sets the value of the Q parameter.

o **sign**(byte[], short, short, byte[], short, short)
Signs data using this key.

Constructors

o **RSA_CRT_Key**

```
public RSA_CRT_Key()
```

Constructs an empty RSA_CRT_Key

Methods

o **isInitialized**

```
public boolean isInitialized()
```

Reports the initialized state of the key. All five CRT parameter must be initialized before the key can be used.

Returns:
true if the key has been initialized.
Overrides:
isInitialized in class Key

o **setP**

```
public void setP(byte buffer[],
                 short offset,
                 short length)
```

Sets the value of the P parameter.

Parameters:
buff - the input buffer
offset - the offset into the input buffer at which modulus value begins
length - the length to encrypt

o **setQ**

```
public void setQ(byte buffer[],
                 short offset,
                 short length)
```

Sets the value of the Q parameter.

Parameters:
buff - the input buffer
offset - the offset into the input buffer at which modulus value begins
length - the length to encrypt

o **setDP1**

```
public void setDP1(byte buffer[],
                   short offset,
                   short length)
```

Sets the value of the DP1 parameter.

Parameters:
buff - the input buffer
offset - the offset into the input buffer at which modulus value begins
length - the length to encrypt

o **setDQ1**

```
public void setDQ1(byte buffer[],
                   short offset,
                   short length)
```

Sets the value of the P key.

Parameters:
buff - the input buffer
offset - the offset into the input buffer at which modulus value begins
length - the length to encrypt

o **setPQ**

```
public void setPQ(byte buffer[],
                  short offset,
                  short length)
```

Sets the value of the PQ parameter.

Parameters:
buff - the input buffer
offset - the offset into the input buffer at which modulus value begins
length - the length to encrypt

o **sign**

```
public native void sign(byte msgDigest[],
                        short msgOffset,
                        short msgLength,
                        byte signedData[],
                        short signOffset,
                        short signLength)
```

Signs data using this key.

Parameters:
msgDigest - the input buffer containing the hash result.
msgOffset - the offset into the hash result buffer
msgLength - the length of the hash
signedData - the output buffer containing the signed data.
signOffset - the offset into the signed data buffer
signLength - the of the signed data buffer
Returns:
true if the data is properly signed.

Class javacardx.crypto.RSA_Key

```
java.lang.Object
   |
   +----javacardx.crypto.AsymKey
           |
           +----javacardx.crypto.RSA_Key
```

public class **RSA_Key**
extends AsymKey

The RSA_Key is used to sign data (typically a hash value) and verify signatures on signed data using the RSA algorithm in its modulus/exponent form.

See Also:
RSA_CRT_Key

Constructor Index

o **RSA_Key**()
Constructs an empty RSA_Key that stores its key data in modulus/exponent form.

Method Index

o **isInitialized**()
Reports the initialized state of the key.
o **setExponent**(byte[], short, short)
Sets the exponent value of the key.
o **setModulus**(byte[], short, short)
Sets the modulus value of the key.
o **sign**(byte[], short, short, byte[], short, short)
Signs data using this key.
o **verify**(byte[], short, short, byte[], short, short)
Verifies signed data using this key.

Constructors

o **RSA_Key**

```
public RSA_Key()
```

Constructs an empty RSA_Key that stores its key data in modulus/exponent form.

Java Card 2.0 API, copyright Sun Microsystems Inc

Methods

o **isInitialized**

```
public boolean isInitialized()
```

Reports the initialized state of the key. Both the modulus and exponent must be initialized before the key can be used.

Returns:
true if the key has been initialized.

o **setModulus**

```
public void setModulus(byte buffer[],
                       short offset,
                       short length)
```

Sets the modulus value of the key. When both the modulus and exponent are set the key is initialized and ready for use.

Parameters:
buff - the input buffer
offset - the offset into the input buffer at which modulus value begins
length - the length to encrypt

o **setExponent**

```
public void setExponent(byte buffer[],
                        short offset,
                        short length)
```

Sets the exponent value of the key. When both the modulus and exponent are set the key is initialized and ready for use.

Parameters:
buff - the input buffer
offset - the offset into the input buffer at which the exponet value begins
length - the length to encrypt

o **sign**

```
public native void sign(byte msgDigest[],
                        short msgOffset,
                        short msgLength,
                        byte signedData[],
                        short signOffset,
                        short signLength)
```

Signs data using this key.

Parameters:
msgDigest - the input buffer containing the hash result.
msgOffset - the offset into the hash result buffer
msgLength - the length of the hash
signedData - the output buffer containing the signed data.

signOffset - the offset into the signed data buffer
signLength - the of the signed data buffer

Returns:
true if the data is properly signed.

o **verify**

```
public native boolean verify(byte msgDigest[],
                             short msgOffset,
                             short msgLength,
                             byte signedData[],
                             short signOffset,
                             short signLength)
```

Verifies signed data using this key.

Parameters:
msgDigest - the buffer containing the hash result.
msgOffset - the offset into the hash result buffer
msgLength - the length of the hash
signedData - the buffer containing the signed data.
signOffset - the offset into the signed data buffer
signLength - the of the signed data buffer

Returns:
true if the data is properly signed.

Java Card 2.0 API, copyright Sun Microsystems Inc

Class javacardx.crypto.RandomData

```
java.lang.Object
   |
   +----javacardx.crypto.RandomData
```

public class **RandomData**
extends Object

The RandomData class provides a source of randomness.

Constructor Index

o **RandomData**()

Method Index

o **generateData**(byte[], short, short)
generates random data.
o **setSeed**(byte[], short, short)
seeds random data generator.

Constructors

o **RandomData**

```
public RandomData()
```

Methods

o **generateData**

```
public static native void generateData(byte buffer[],
                                       short offset,
                                       short length)
```

generates random data.

Parameters:
buffer - the output buffer
offset - the offset into the output buffer
length - the length of random data to generate

o **setSeed**

```
public static native void setSeed(byte buffer[],
                                  short offset,
                                  short length)
```

seeds random data generator. Implementation dependent (e.g. may not be required on platforms with hardware support for random number generation).

Parameters:
buffer - the input buffer
offset - the offset into the input buffer
length - the length of the seed data

Class javacardx.crypto.Sha1MessageDigest

```
java.lang.Object
   |
   +----javacardx.crypto.MessageDigest
           |
           +----javacardx.crypto.Sha1MessageDigest
```

---

public class **Sha1MessageDigest**
extends MessageDigest

The Sha1MessageDigest class implements the SHA1 algorithm.

Constructor Index

o **Sha1MessageDigest**()

Method Index

o **generateDigest**(byte[], short, short, byte[], short, short)
generates a hash of the input data using the SHA1 algorithm.

Constructors

o **Sha1MessageDigest**

```
public Sha1MessageDigest()
```

Methods

o **generateDigest**

```
public static native void generateDigest(byte inBuff[],
                                          short inOffset,
                                          short inLength,
                                          byte outBuff[],
                                          short outOffset,
                                          short outLength)
```

generates a hash of the input data using the SHA1 algorithm.

Parameters:
inBuff - the input buffer
inOffset - the offset into the input buffer at which to begin encryption
inLength - the length to encrypt
outBuff - the output buffer, may be the same as the input buffer
outOffset - the offset into the output buffer
outLength - the length of resulting hash

Class javacardx.crypto.SymKey

```
java.lang.Object
   |
   +----javacardx.crypto.Key
           |
           +----javacardx.crypto.SymKey
```

public abstract class **SymKey**
extends Key

The SymKey class is the base class for keys used in symetric alogrightms (e.g. DES).

Constructor Index

o **SymKey**()

Method Index

o **decrypt**(byte[], short, short, byte[], short)
  Decrypts data using this key.
o **encrypt**(byte[], short, short, byte[], short)
  Encrypts data using this key.
o **isInitialized**()
  Reports the initialized state of the key.
o **setKey**(byte[], short, short)
  initializes a key from raw key data bytes.

Constructors

o **SymKey**

```
public SymKey()
```

Methods

o **isInitialized**

```
public boolean isInitialized()
```

Reports the initialized state of the key. Keys must be initialized before being used.

Returns:
true if the key has been initialized.
Overrides:
isInitialized in class Key

o **setKey**

```
public void setKey(byte buff[],
                   short offset,
                   short length)
```

initializes a key from raw key data bytes. After initialization isInitialized returns true.

Parameters:
buff - the input buffer
offset - the offset into the input buffer at which the key data begins
length - the number of bytes of key data

o **encrypt**

```
public abstract native void encrypt(byte inBuff[],
                                    short inOffset,
                                    short inLength,
                                    byte outBuff[],
                                    short outOffset)
```

Encrypts data using this key.

Parameters:
inBuff - the input buffer
inOffset - the offset into the input buffer at which to begin encryption
inLength - the length to encrypt
outBuff - the output buffer, may be the same as the input buffer
outOffset - the offset into the output buffer

o **decrypt**

```
public abstract native void decrypt(byte inBuff[],
                                    short inOffset,
                                    short inLength,
                                    byte outBuff[],
                                    short outOffset)
```

Decrypts data using this key.

Parameters:
inBuff - the input buffer
inOffset - the offset into the input buffer at which to begin decryption
inLength - the length to decrypt
outBuff - the output buffer, may be the same as the input buffer
outOffset - the offset into the output buffer

Java Card 2.0 API

Table of Contents

Java Card 2.0 API 1
Java Card 2.0 API 1
Other Packages 1
Class Hierarchy 2
Class Hierarchy 2
Index of all Fields and Methods 3
Index of all Fields and Methods 3
A 3
B 4
C 4
D 4
E 5
F 5
G 5
I 7
K 8
L 8
M 8
P 8
R 9
S 9
T 11
U 11
V 11
W 12
Package javacard.framework 13
package javacard.framework 13
Class Index 13
Exception Index 13
Class javacard.framework.AID 14
Class javacard.framework.AID 14
Method Index 14
Methods 14
Class javacard.framework.APDU 16
Class javacard.framework.APDU 16
Variable Index 17
Method Index 17
Variables 18
Methods 19
Class javacard.framework.ApduException 22
Class javacard.framework.ApduException 22
Variable Index 22
Constructor Index 22
Method Index 22
Variables 22
Constructors 23

Methods 23
Class javacard.framework.Applet 24
Class javacard.framework.Applet 24
Constructor Index 24
Method Index 24
Constructors 25
Methods 25
Class javacard.framework.CardException 27
Class javacard.framework.CardException 27
Variable Index 27
Constructor Index 27
Method Index 27
Variables 27
Constructors 27
Methods 28
Class javacard.framework.CardRuntimeException 29
Class javacard.framework.CardRuntimeException 29
Variable Index 29
Constructor Index 29
Method Index 29
Variables 29
Constructors 29
Methods 30
Class javacard.framework.FileIoException 31
Class javacard.framework.FileIoException 31
Constructor Index 31
Method Index 31
Constructors 31
Methods 31
Class javacard.framework.PIN 32
Class javacard.framework.PIN 32
Constructor Index 32
Method Index 32
Constructors 32
Methods 33
Class javacard.framework.PINException 35
Class javacard.framework.PINException 35
Variable Index 35
Constructor Index 35
Method Index 35
Variables 35
Constructors 36
Methods 36
Class javacard.framework.System 37
Class javacard.framework.System 37
Constructor Index 37
Method Index 37
Constructors 38
Methods 38
Class javacard.framework.SystemException 41
Class javacard.framework.SystemException 41

Variable Index . . . 41
Constructor Index . . . 41
Method Index . . . 41
Variables . . . 41
Constructors . . . 41
Methods . . . 42
Class javacard.framework.UserException . . . 43
Class javacard.framework.UserException . . . 43
Constructor Index . . . 43
Method Index . . . 43
Constructors . . . 43
Methods . . . 43
Class javacard.framework.Util . . . 44
Class javacard.framework.Util . . . 44
Constructor Index . . . 44
Method Index . . . 44
Constructors . . . 44
Methods . . . 44
Package javacardx.framework . . . 46
package javacardx.framework . . . 46
Class Index . . . 46
Class javacardx.framework.CyclicFile . . . 47
Class javacardx.framework.CyclicFile . . . 47
Constructor Index . . . 47
Method Index . . . 47
Constructors . . . 47
Methods . . . 48
Class javacardx.framework.DedicatedFile . . . 50
Class javacardx.framework.DedicatedFile . . . 50
Constructor Index . . . 50
Method Index . . . 50
Constructors . . . 50
Methods . . . 51
Class javacardx.framework.ElementaryFile . . . 53
Class javacardx.framework.ElementaryFile . . . 53
Constructor Index . . . 53
Method Index . . . 53
Constructors . . . 53
Methods . . . 53
Class javacardx.framework.File . . . 54
Class javacardx.framework.File . . . 54
Variable Index . . . 55
Constructor Index . . . 55
Method Index . . . 55
Variables . . . 55
Constructors . . . 56
Methods . . . 56
Class javacardx.framework.FileSystem . . . 58
Class javacardx.framework.FileSystem . . . 58
Variable Index . . . 58
Constructor Index . . . 58

Java Card 2.0 API, copyright Sun Microsystems Inc

Method Index . . . 58
Variables . . . 59
Constructors . . . 60
Methods . . . 60
Class javacardx.framework.LinearFixedFile . . . 66
Class javacardx.framework.LinearFixedFile . . . 66
Constructor Index . . . 66
Method Index . . . 66
Constructors . . . 66
Methods . . . 67
Class javacardx.framework.LinearVariableFile . . . 68
Class javacardx.framework.LinearVariableFile . . . 68
Variable Index . . . 68
Constructor Index . . . 68
Method Index . . . 68
Variables . . . 69
Constructors . . . 69
Methods . . . 69
Class javacardx.framework.TransparentFile . . . 72
Class javacardx.framework.TransparentFile . . . 72
Constructor Index . . . 72
Method Index . . . 72
Constructors . . . 72
Methods . . . 73
Package javacardx.crypto . . . 74
package javacardx.crypto . . . 74
Class Index . . . 74
Exception Index . . . 74
Class javacardx.crypto.AsymKey . . . 75
Class javacardx.crypto.AsymKey . . . 75
Constructor Index . . . 75
Method Index . . . 75
Constructors . . . 75
Methods . . . 75
Class javacardx.crypto.CryptoException . . . 77
Class javacardx.crypto.CryptoException . . . 77
Constructor Index . . . 77
Constructors . . . 77
Class javacardx.crypto.DES_CBC_Key . . . 78
Class javacardx.crypto.DES_CBC_Key . . . 78
Constructor Index . . . 78
Method Index . . . 78
Constructors . . . 78
Methods . . . 79
Class javacardx.crypto.DES_Key . . . 81
Class javacardx.crypto.DES_Key . . . 81
Constructor Index . . . 81
Method Index . . . 81
Constructors . . . 81
Methods . . . 81

Class javacardx.crypto.Key . . . 83
Class javacardx.crypto.Key . . . 83
Constructor Index . . . 83
Method Index . . . 83
Constructors . . . 83
Methods . . . 83
Class javacardx.crypto.MessageDigest . . . 85
Class javacardx.crypto.MessageDigest . . . 85
Constructor Index . . . 85
Method Index . . . 85
Constructors . . . 85
Methods . . . 85
Class javacardx.crypto.RSA_CRT_Key . . . 87
Class javacardx.crypto.RSA_CRT_Key . . . 87
Constructor Index . . . 87
Method Index . . . 87
Constructors . . . 88
Methods . . . 88
Class javacardx.crypto.RSA_Key . . . 91
Class javacardx.crypto.RSA_Key . . . 91
Constructor Index . . . 91
Method Index . . . 91
Constructors . . . 91
Methods . . . 92
Class javacardx.crypto.RandomData . . . 94
Class javacardx.crypto.RandomData . . . 94
Constructor Index . . . 94
Method Index . . . 94
Constructors . . . 94
Methods . . . 94
Class javacardx.crypto.Sha1MessageDigest . . . 96
Class javacardx.crypto.Sha1MessageDigest . . . 96
Constructor Index . . . 96
Method Index . . . 96
Constructors . . . 96
Methods . . . 96
Class javacardx.crypto.SymKey . . . 98
Class javacardx.crypto.SymKey . . . 98
Constructor Index . . . 98
Method Index . . . 98
Constructors . . . 98
Methods . . . 98