GlobalPlatform Technology

Open Mobile API – Android Binding

Version 0.0.0.5 for OMAPI v3.3

Public Review

July 2018

Document Reference: GPP_SPE_002

THIS SPECIFICATION OR OTHER WORK PRODUCT IS BEING OFFERED WITHOUT ANY WARRANTY WHATSOEVER, AND IN PARTICULAR, ANY WARRANTY OF NON-INFRINGEMENT IS EXPRESSLY DISCLAIMED. ANY IMPLEMENTATION OF THIS SPECIFICATION OR OTHER WORK PRODUCT SHALL BE MADE ENTIRELY AT THE IMPLEMENTER'S OWN RISK, AND NEITHER THE COMPANY, NOR ANY OF ITS MEMBERS OR SUBMITTERS, SHALL HAVE ANY LIABILITY WHATSOEVER TO ANY IMPLEMENTER OR THIRD PARTY FOR ANY DAMAGES OF ANY NATURE WHATSOEVER DIRECTLY OR INDIRECTLY ARISING FROM THE IMPLEMENTATION OF THIS SPECIFICATION OR OTHER WORK PRODUCT.

Contents

Figures

Tables

1 Introduction

This document presents a Java language API for Open Mobile API Transport Layer specifically targeting the Android platform. It is intended to be read in conjunction with the GlobalPlatform Open Mobile API Specification ([OMAPI]), which contains detailed descriptions of expected API behavior.

Where this document explicitly overrides behavior in [OMAPI], the behavior specified in this document is normative for the Android platform, even where it contradicts [OMAPI].

For Java packages that are mentioned but not described in this document, see [Android Dev] for the normative API definition. This document is written assuming the packages and APIs of Android API Level 28.

Code fragments in this document are written in the Java language.

An Open Source implementation of this API is available in selected Git repositories available at [AOSP] and is part of Android versions from API Level 28 onwards.

- The framework code is available in the `platform/frameworks/base` Git repository. The OMAPI framework itself is in the `core/java/android/se/omapi` folder.
- The service implementation is available in the `platform/packages/apps/SecureElement` Git repository.
- An example of a HAL implementation for an eSE can be found in the `platform/hardware/nxp/secure_element` Git repository.

Note: Some devices supporting Android API Level 27 or earlier may claim support for [OMAPI]. These are not necessarily compliant with this specification.

1.1 Audience

This specification is intended for:

- Device manufacturers wishing to understand the AOSP implementation of this specification.
- Service providers developing a Rich OS application in conjunction with a Secure Element or UICC application.
- Secure Element providers wishing to understand the features available to them on the Android platform.

1.2 IPR Disclaimer

Attention is drawn to the possibility that some of the elements of this GlobalPlatform specification or other work product may be the subject of intellectual property rights (IPR) held by GlobalPlatform members or others. For additional information regarding any such IPR that have been brought to the attention of GlobalPlatform, please visit https://www.globalplatform.org/specificationsipdisclaimers.asp. GlobalPlatform shall not be held responsible for identifying any or all such IPR, and takes no position concerning the possible existence or the evidence, validity, or scope of any such IPR.

1.3 References

Table 1-1: Normative References

Standard / Specification	Description	Ref
GlobalPlatform Secure Element Access Control	GlobalPlatform Technology Secure Element Access Control v1.1	[SE Acc Ctl]
GlobalPlatform Open Mobile API	GlobalPlatform Technology Open Mobile API Specification v3.3	[OMAPI]
Android Platform	https://developer.android.com/reference/classes.html	[Android Dev]
AOSP	Android Open Source Platform https://android.googlesource.com	[AOSP]
ISO/IEC 7816-4:2013	Identification cards – Integrated circuit cards – Part 4: Organization, security and commands for interchange	[7816-4]
IETF RFC 2119	Key words for use in RFCs to Indicate Requirement Levels	[RFC 2119]

Table 1-2: Informative References

Standard / Specification	Description	Ref
Java Card API	Application Programming Interface, Java Card™ Platform, v3.0.1 Classic Edition, Sun Microsystems, Inc., May 2009	[JCAPI]
Java Card VM	Virtual Machine Specification, Java Card™ Platform, v3.0.1 Classic Edition, Sun Microsystems, Inc., May 2009	[JCVM]
Java Card JCRE	Runtime Environment Specification, Java Card™ Platform, v3.0.1 Classic Edition, Sun Microsystems, Inc., May 2009	[JCRE]

1.4 Terminology and Definitions

The following meanings apply to SHALL, SHALL NOT, MUST, MUST NOT, SHOULD, SHOULD NOT, and MAY in this document (refer to [RFC 2119]):

- **SHALL** indicates an absolute requirement, as does **MUST**.
- **SHALL NOT** indicates an absolute prohibition, as does **MUST NOT**.
- **SHOULD** and **SHOULD NOT** indicate recommendations.
- **MAY** indicates an option.

Selected terms used in this document are included in Table 1-3. Additional terms are defined in [OMAPI].

Table 1-3: Terminology and Definitions

Term	Definition
Android Package (APK)	A file format used by the Android operating system for distribution of mobile applications and system middleware.
Binder	Android remote procedure call mechanism (https://developer.android.com/reference/android/os/Binder.html)
Intent	Android mechanism for asynchronous operations (https://developer.android.com/reference/android/content/Intent.html)

1.5 Abbreviations and Notations

Table 1-4: Abbreviations and Notations

Abbreviation / Notation	Meaning
AID	Application IDentifier
AIDL	Android Interface Definition Language
API	Application Programming Interface
APK	Android PacKage
CLF	Contactless Front End
HAL	Hardware Abstraction Layer
HCI	Host Controller Interface
HIDL	Hardware IDL (https://source.android.com/devices/architecture/hidl/)
SE	Secure Element
SHA-1	Secure Hash Algorithm 1

1.6 Revision History

GlobalPlatform technical documents numbered *n*.0 are major releases. Those numbered *n*.1, *n*.2, etc., are minor releases where changes typically introduce supplementary items that do not impact backward compatibility or interoperability of the specifications. Those numbered *n.n*.1, *n.n*.2, etc., are maintenance releases that incorporate errata and precisions; all non-trivial changes are indicated, often with revision marks.

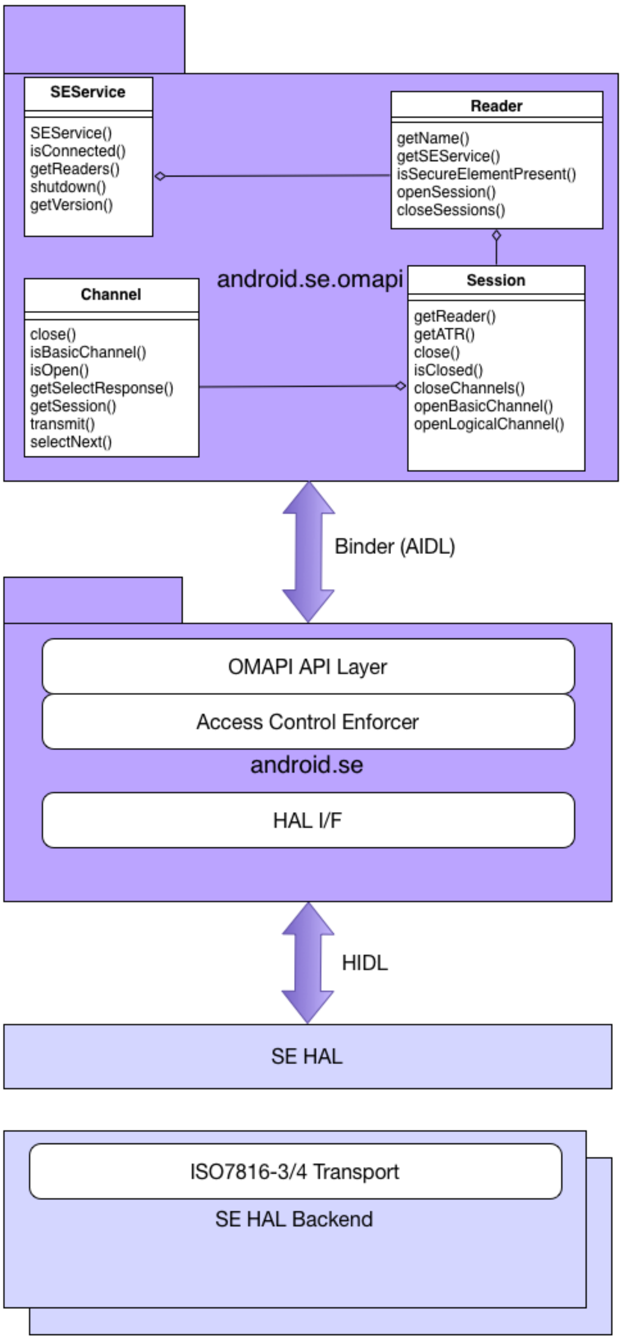
Table 1-5: Revision History

Date	Version	Description
April 2018	0.0.0.3	Member Review
July 2018	0.0.0.5	Public Review
TBD	1.0	Initial release

2 Implementation Architecture (Informative)

The Android implementation of the Open Mobile API is structured as shown in Figure 2-1.

Figure 2-1: OMAPI Implementation in Android



A Java package, `android.se.omapi`, is exposed to applications. This package contains the `SEService`, `Reader`, `Session`, and `Channel` classes defined in [OMAPI], which are implemented as specified in this document.

The `android.se.omapi` package communicates with a separate process space which supports an Open Mobile API Service, `android.se`. Communication between the Service and client application which import `android.se.omapi` uses the Android Binder mechanism.

The `android.se` service communicates with one or more instances of the SE HAL. The HAL provides an abstracted API for communicating with all types of Secure Elements and is responsible for implementing the ISO 7816 Transport Layer behavior defined in [OMAPI], as restricted by this document.

The Access Control Enforcer manages applications' access to applets running on a Secure Element according to [SE Acc Ctl]. On the Android platform, access is granted based on a DeviceAppID which is the SHA-1 hash of the certificate used to sign the APK.

A consequence of layering the Access Control Enforcer above the ISO 7816 Transport Layer is that in scenarios where either an Access Control Exception or a transport layer Exception are possible, the Access Control exception will be raised. [OMAPI] leaves this as an implementation decision, so this is a legal implementation of OMAPI.

3 API Description

This API description assumes the presence of the packages that are available in Android API Level 28.

3.1 Package

The Android platform overrides the namespace specified in [OMAPI] section 1.

```
package android.se.omapi;
```

3.2 Intents

3.2.1 Transaction Event Intent

An Intent SHALL be generated by the Android NFC service when a Contactless transaction event is generated by a Contactless Front End (CLF) which is connected to a Secure Element instance managed by the Android OMAPI service.

Note: The NFC service is responsible for generating the Intent because the transaction is reported to the NFC Service by means of the Host Controller entity in the CLF, which is owned by the NFC service.

The generated Intent SHALL include the following data:

Table 3-1: Transaction Event Definition

ACTION_TRANSACTION_DETECTED		android.nfc.action.TRANSACTION_DETECTED
Extra	EXTRA_AID	Mandatory ByteArray containing AID of the SELECTed applet.
	EXTRA_DATA	Optional ByteArray containing payload from the HCI EVT_TRANSACTION
	EXTRA_SE_NAME	Mandatory String containing the name of the SE that generated the transaction detected event.

ACTION_TRANSACTION_DETECTED is only sent to applications for which both of the following are true:

- The application has the NFC_TRANSACTION_EVENT permission.
- The Access Control Enforcer grants access.

3.2.2 OnConnected Listener

This section overrides [OMAPI] section 4.2.3.

When an application instantiates `SEService`, it SHALL pass in a listener class implementing the `OnConnectedListener` interface.

```
package android.se.omapi;

public interface OnConnectedListener {
  void onConnected();
}
```

The `onConnected()` method is called when the service is connected.

`SEService` provides `SEListener`, a default implementation of the `OnConnectedListener` interface.

3.2.3 Reader Event Listener

This section overrides [OMAPI] section 4.2.4.

Android API Level 28 does not support `Reader:EventCallback`.

3.3 Classes

In general, Android marks OMAPI classes as final, meaning that they cannot be extended by Apps.

3.3.1 SEService

See [OMAPI] section 4.2.2 for a description of the `SEService` class.

On the Android platform, the `SEService CallBack` is implemented using an instance of `OnConnectedListener`.

Applications should not perform any processing which requires the `SEService` instance to be connected until `onConnected()` has been called by the service or `isConnected()` returns `true`.

```
package android.se.omapi;

public final class SEService {
    /* Implementation */
}
```

3.3.1.1 Constructor

See [OMAPI] section 4.2.2.1 for a description of the `SEService` constructor.

```
public SEService(@NonNull Context    context,
                 @NonNull Executor   executor,
                 OnConnectedListener listener);
```

The Android OMAPI service can take some time to establish an instance which can be used by the caller. The caller can optionally provide an `OnConnectedListener` to provide a notification when the service is connected.

The Android Platform requires an instance of `Executor` in the `SEService` constructor which is not described in [OMAPI] section 4.2.2.1. This allows a background thread to run when the main thread is blocked waiting for the `OnConnectedListener` if the use case requires it.

3.3.1.1.1 Exceptions

`NullPointerException` is thrown if `context` is `null`.

3.3.1.2 Method: isConnected()

See [OMAPI] section 4.2.2.3 for a description of the `isConnected()` method.

```
public boolean isConnected();
```

3.3.1.3 Method: getReaders()

See [OMAPI] section 4.2.2.2 for a description of the `getReaders()` method.

```
public @NonNull Reader[] getReaders();
```

The returned array is never `null`. If there are no Readers, an array of length 0 is returned.

3.3.1.3.1 Exceptions

`IllegalStateException` is thrown if the `SEService` instance is not yet valid.

`RuntimeException` is thrown if the service instance exists, but the proxy is unable to obtain the reader names from the service.

3.3.1.4 Method: shutdown()

See [OMAPI] section 4.2.2.4 for a description of the `shutdown()` method.

```
public void shutdown();
```

3.3.1.5 Method: getVersion()

See [OMAPI] section 4.2.2.5 for a description of the `getVersion()` method.

```
public String getVersion();
```

Android API Level 28 returns `"3.3"` for the Open Mobile API version.

3.3.2 Reader

See [OMAPI] section 4.2.6 for a description of the Reader class.

Android does not support the `Reader:EventCallback` mechanism.

```
package android.se.omapi;

public final class Reader {
    /* Implementation */
}
```

3.3.2.1 Method: getName()

See [OMAPI] section 4.2.6.1 for a description of the `getName()` method.

```
public @NonNull String getName();
```

The Android platform uses the Reader instance naming convention specified in [OMAPI], with the exception that slot numbering is mandatory, with slot numbers for each reader type counting upwards from 1.

For a UICC reader, the UICC slot numbering SHALL be the same as the value returned by SubscriptionInfo.getSimSlotIndex().

3.3.2.2 Method: getSEService()

See [OMAPI] section 4.2.6.2 for a description of the `getSEService()` method.

```
public @NonNull SEService getSEService();
```

3.3.2.3 Method: isSecureElementPresent()

See [OMAPI] section 4.2.6.3 for a description of the `isSecureElementPresent()` method.

```
public boolean isSecureElementPresent();
```

3.3.2.3.1 Exceptions

`IllegalStateException` is thrown if the `SEService` instance is not valid or if the Reader instance cannot be obtained.

3.3.2.4 Method: openSession()

See [OMAPI] section 4.2.6.4 for a description of the `openSession()` method.

On the Android platform, reference counting is used to determine whether power is applied to the SE. As such, if there is currently no open session on a given SE anywhere in the system, the OMAPI service will power up the SE.

```
public @NonNull Session openSession() throws IOException;
```

3.3.2.4.1 Exceptions

`IllegalStateException` is thrown if the `SEService` instance is not yet valid.

`IOException` is thrown if there is a failure of communication with `Reader` or SE hardware.

3.3.2.5 Method: closeSessions()

See [OMAPI] section 4.2.6.5 for a description of the `closeSessions()` method.

On the Android platform, reference counting is used to determine whether power is applied to the SE. As such, the OMAPI service will power down the SE only if no other application has a Session open to the same SE.

```
public void closeSessions();
```

3.3.2.5.1 Exceptions

`IllegalStateException` is thrown if the `SEService` instance is not yet valid or if there is a failure of communication with `Reader` or SE hardware

3.3.2.6 Method: registerReaderEventCallback()

This section overrides [OMAPI] section 4.2.6.6.

The Android platform does not implement `registerReaderEventCallback()` or an equivalent.

3.3.2.7 Method: unregisterReaderEventCallback()

This section overrides [OMAPI] section 4.2.6.7.

The Android platform does not implement `unregisterReaderEventCallback()` or an equivalent.

3.3.3 Session

See [OMAPI] section 4.2.7 for a description of the Session class.

```
package android.se.omapi;

public class Session {
    /* Implementation */
}
```

3.3.3.1 Method: getReader()

See [OMAPI] section 4.2.7.1 for a description of the `getReader()` method.

```
public @NonNull Reader getReader();
```

3.3.3.2 Method: getATR()

See [OMAPI] section 4.2.7.2 for a description of the `getATR()` method.

```
public @Nullable byte[] getATR();
```

On the Android platform, `getATR()` may return `null` `for` `Session` instances connected to an eSE.

3.3.3.2.1 Exceptions

`IllegalStateException` is thrown if the owning `SEService` instance is not yet valid or if there is a failure of communication with `Reader` or SE hardware.

3.3.3.3 Method: close()

See [OMAPI] section 4.2.7.3 for a description of the `close()` method.

```
public void close();
```

3.3.3.4 Method: isClosed()

See [OMAPI] section 4.2.7.4 for a description of the `isClosed()` method.

```
public boolean isClosed();
```

3.3.3.5 Method: closeChannels()

See [OMAPI] section 4.2.7.5 for a description of the `closeChannels()` method.

```
public void closeChannels();
```

3.3.3.6 Method: openBasicChannel()

See [OMAPI] sections 4.2.7.6 and 4.2.7.7 for a description of the `openBasicChannel()` method.

On the Android platform, an attempt to call either of the `openBasicChannel()` methods on a UICC Reader instance will return a `null` value.

```
public @Nullable Channel openBasicChannel(byte[] aid, byte p2)
  throws IOException;
public @Nullable Channel openBasicChannel(byte[] aid) throws
  IOException;
```

3.3.3.6.1 Exceptions

`IllegalStateException` is thrown if the SE session is used after being closed.

`IOException` is thrown if there is a communication issue with the Reader or SE.

`IllegalArgumentException` is thrown if `aid` length is not between 5 and 15 bytes.

`SecurityException` is thrown if access cannot be granted to `aid` (or the default applet) on this session.

`NoSuchElementException` is thrown if the AID is not available or is not SELECTed.

`UnsupportedOperationException` is thrown if the provided `p2` value is not supported by the device.

3.3.3.7 Method: openLogicalChannel()

See [OMAPI] section 4.2.7.8 and 4.2.7.9 for a description of the `openLogicalChannel()` method.

```
public @Nullable Channel openLogicalChannel(byte[] aid,
  byte p2) throws IOException;
public @Nullable Channel openLogicalChannel(byte[] aid)
  throws IOException;
```

3.3.3.7.1 Exceptions

`IllegalStateException` is thrown if the SE session is used after being closed.

`IOException` is thrown if there is a communication issue with the Reader or SE.

`IllegalArgumentException` is thrown if `aid` length is not between 5 and 15 bytes.

`SecurityException` is thrown if access cannot be granted to `aid` (or the default applet) on this session.

`NoSuchElementException` is thrown if the AID is not available or is not SELECTed.

`UnsupportedOperationException` is thrown if the provided `p2` value is not supported by the device.

3.3.4 Channel

See [OMAPI] section 4.2.8 for a description of the Channel class.

Android defines Channel as implementing the Java Channel class interface.

```
package android.se.omapi;

public final class Channel implements java.nio.channels.Channel {
    /* Implementation */
}
```

3.3.4.1 Method: close()

See [OMAPI] section 4.2.8.1 for a description of the `close()` method.

```
public void close();
```

3.3.4.2 Method: isBasicChannel()

See [OMAPI] section 4.2.8.2 for a description of the `isBasicChannel()` method.

```
public boolean isBasicChannel();
```

3.3.4.2.1 Exceptions

`IllegalStateException` is thrown if the SE session is used after being closed or before a connection is established.

3.3.4.3 Method: isOpen()

This section overrides [OMAPI] section 4.2.8.3.

```
public boolean isOpen();
```

Android defines the Channel class as an implementation of the Java Channel interface and therefore requires the availability of an `isOpen()` method.

This provides the inverse of the logic defined for the `isClosed()` method in [OMAPI] section 4.2.8.3. As a consequence, Android does not provide an `isClosed()` method.

3.3.4.4 Method: getSelectResponse()

See [OMAPI] section 4.2.8.4 for a description of the `getSelectResponse()` method.

```
public @Nullable byte[] getSelectResponse();
```

3.3.4.4.1 Exceptions

`IllegalStateException` is thrown if the SE Service is not connected.

3.3.4.5 Method: getSession()

See [OMAPI] section 4.2.8.5 for a description of the `getSession()` method.

```
public @NonNull Session getSession();
```

3.3.4.6 Method: setTransmitBehaviour()

This section overrides [OMAPI] section 4.2.8.6.

The Android platform does not implement `setTransmitBehaviour()`.

Applets residing on Secure Elements which rely on this behavior are not supported by the Android platform.

3.3.4.7 Method: transmit()

This section overrides part of [OMAPI] section 4.2.8.7.

See [OMAPI] section 4.2.8.7 for a description of the `transmit()` method. On the Android Platform, received status words are always handled according to [OMAPI] section 4.2.8.6 as if the value of `expectDataWithWarningSW` is false.

The implementation of the `transmit()` method on the Android Platform supports extended L_c and L_e fields as defined in [7816-4]. HAL backend implementations SHALL support at least 2048-byte values of L_c and L_e if the connected Secure Element supports it.

```
public byte[] transmit(byte[] command)
  throws IOException, IllegalStateException,
         IllegalArgumentException, SecurityException, NullPointerException;
```

3.3.4.7.1 Exceptions

`IllegalStateException` is thrown if the SE session is used after being closed.

`IOException` is thrown if there is a communication issue with the Reader or SE.

`IllegalArgumentException` is thrown if `aid` length is not between 5 and 15 bytes.

`SecurityException` is thrown if access cannot be granted to `aid` (or the default applet) on this session.

`NoSuchElementException` is thrown if the AID is not available or is not SELECTed.

`UnsupportedOperationException` is thrown if the provided `p2` value is not supported by the device.

3.3.4.8 Method: selectNext()

See [OMAPI] section 4.2.8.8 for a description of the `selectNext()` method.

```
public boolean selectNext();
```

3.3.4.8.1 Exceptions

`IllegalStateException` is thrown if the SE session is used after being closed.