Android SDK Guide

for SecureMag

#80066816-001

Rev. A

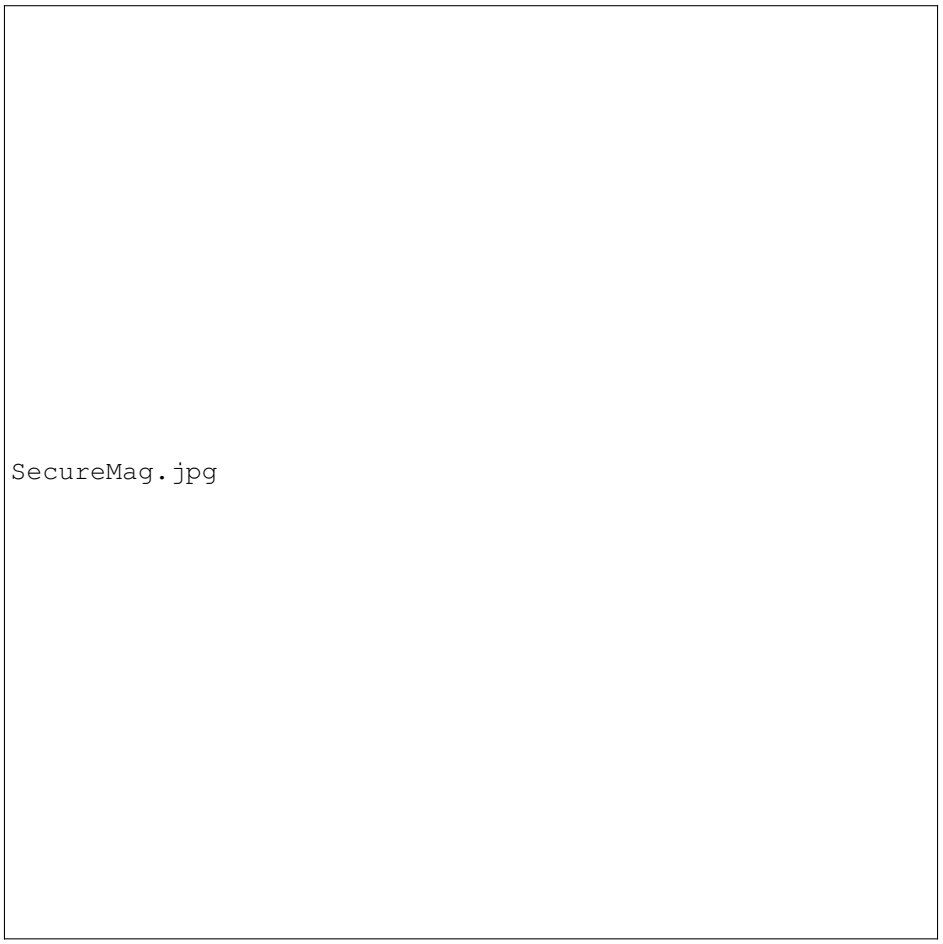
Revision History

Revision	Description and Reason for Change	Date
A	Initial Release - Manual;User;SecureMag;SDK;Android	1/21/2019

Contents

Chapter 1

IDTech Android SDK Reference Guide for SecureMag



Universal_SDK_X.XX.XXX.jar is an Android library that will be provided by IDTech as the main interface between Android applications, the SecureMag and payment processing solutions.

The purpose of this document is to describe the requirements of the library as well as the interface definitions and requirements needed for any Android applications wishing to deploy with the payment application.

Chapter 2

Connecting with SecureMag

The SecureMag connects through Headphone jack or USB on Android. On some models Bluetooth Low Energy is available as a connection option

2.1 Connect with USB

The SecureMag will be recognized as a Human Interface Device once plugged into the Android USB port. The Android must be running firmware 3.1 or greater, and it will need an Android USB host adapter cable, usually referred to as OTG. Please see your manufacturers documentation for the correct part to use. Use a standard USB-miniUSB plug to attach the SecureMag (mini-USB port on left side) to the Android host adapter cable.

Chapter 3

Important Security Notice

The Payment Card Industry Payment Application Data Security Standard (PCI PA-DSS) is comprised of fourteen requirements that support the Payment Card Industry Data Security Standard (PCI DSS). The PCI Security Standards Council (PCI SSC), which was founded by the major card brands in June 2005, set these requirements in order to protect cardholder payment information. The standards set by the council are enforced by the payment card companies who established the Council: American Express, Discover Financial Services, JCB International, MasterCard Worldwide, and Visa, Inc.

PCI PA-DSS is an evolution of Visas Payment Application Best Practices (PABP), which was based on the Visa Cardholder Information Security Program (CISP). In addition to Visa CISP, PCI DSS combines American Express Data Security Operating Policy (DSOP), Discover Networks Information Security and Compliance (DISC), and MasterCards Site Data Protection (SDP) into a single comprehensive set of security standards. The transition to PCI PA-DSS was announced in April 2008. In early October 2008, PCI PA-DSS Version 1.2 was released to align with the PCI DSS Version 1.2, which was released on October 1, 2008. On January 1, 2011, PCI PA-DSS Version 2.0 was released. This extends the PCI DSS Version 1.2, which was released on October 1, 2008 and is effective as of January 1, 2011.

3.1 Applicability

The PCI PA-DSS applies to any payment application that stores, processes, or transmits cardholder data as part of authorization or settlement, unless the application would fall under the merchants PCI DSS validation. It is important to note that PA-DSS validated payment applications alone do not guarantee PCI DSS compliance for the merchant. The validated payment application must be implemented in a PCI DSS compliant environment. If your application runs on Windows XP, you are required to turn off Windows XP System Restore Points.

3.2 What Does PA-DSS Mean to You?

The following table provides opening points to cover in any discussion with merchants on data storage.

	Data Element	Storage Permitted	Protection Required	PCI DSS Req. 3, 4
Cardholder Data	Primary Account Number	Yes	Yes	Yes
	Cardholder Name [1]	Yes	Yes [1]	No
	Service Code [1]	Yes	Yes [1]	No
	Expiration Date [1]	Yes	Yes [1]	No
Sensitive Authentication Data [2]	Full Magnetic Stripe Data [3]	No	N/A	N/A
	CAV2/CID/CVC2/CVV2	No	N/A	N/A
	PIN/PIN Block	No	N/A	N/A

[1] These data elements must be protected if stored in conjunction with the PAN. This protection should be per PCI DSS requirements for general protection of the cardholder environment. Additionally, other legislation (for example, related to consumer personal data protection, privacy, identity theft, or data security) may require specific protection of this data, or proper disclosure of a company's practices if consumer-related personal data is being collected during the course of business. PCI DSS, however, does not apply if PANs are not stored, processed, or transmitted.

[2] Do not store sensitive authentication data after authorization (even if encrypted).

[3] Full track data from the magnetic stripe, magnetic-stripe image on the chip, or elsewhere.

3.3 Third Party Applications

The end-to-end transaction process, beginning with entry into the third party application until the response from the payment engine is returned, must meet the same level of compliance. In order to claim the third party application is end-to-end compliant, the application would need to be submitted to a QSA for a full PA-DSS audit.

The end user and/or P.O.S. developer can integrate and be compliant in the processing portion of a payment transaction. A brief review (given below) of the PA-DSS environmental variables that impact the end user merchant can help the end user merchant obtain and/or maintain PA-DSS compliance. Environmental variables that could prevent passing an audit include without limitation issues involving a secure network connection(s), end user setup location security, users, logging and assigned rights. Remove all testing configurations, samples, and data prior to going into production on your application.

3.4 PA-DSS Guidelines

The following PA-DSS Guidelines are being provided by IDTech as a convenience to its customers. Customers should not rely on these PA-DSS Guidelines, but should instead always refer to the most recent PCI DSS Program Guide published by PCI SSC.

1. Sensitive Data Storage Guidelines

Do not retain full magnetic stripe, card validation code or value (CAV2, CID, CVC2, CVV2), or PIN block data. 1.1 Do not store sensitive authentication data after authorization (even if encrypted): Sensitive authentication data includes the data as cited in the following Requirements 1.1.1 through 1.1.3. PCI Data Security Standard Requirement 3.2

Note: By prohibiting storage of sensitive authentication data after authorization, the assumption is that the transaction has completed the authorization process and the customer has received the final transaction approval. After authorization has completed, this sensitive authentication data cannot be stored.

1.1.1 After authorization, do not store the full contents of any track from the magnetic stripe (located on the back

of a card, contained in a chip, or elsewhere). This data is alternatively called full track, track, track 1, track 2, and magnetic-stripe data.

In the normal course of business, the following data elements from the magnetic stripe may need to be retained:

- The accountholders name,
- Primary account number (PAN),
- Expiration date, and
- Service code
- To minimize risk, store only those data elements needed for business.

Note: See PCI DSS and PA-DSS Glossary of Terms, Abbreviations, and Acronyms for additional information. PCI Data Security Standard Requirement 3.2.1

1.1.2 After authorization, do not store the card-validation value or code (three-digit or four-digit number printed on the front or back of a payment card) used to verify card-not-present transactions. Note: See PCI DSS and P↩ A-DSS Glossary of Terms, Abbreviations, and Acronyms for additional information. PCI Data Security Standard Requirement 3.2.2

1.1.3 After authorization, do not store the personal identification number (PIN) or the encrypted PIN block.

Note: See PCI DSS and PA-DSS Glossary of Terms, Abbreviations, and Acronyms for additional information. PCI Data Security Standard Requirement 3.2.3

1.1.4 Securely delete any magnetic stripe data, card validation values or codes, and PINs or PIN block data stored by previous versions of the payment application, in accordance with industry-accepted standards for secure deletion, as defined, for example by the list of approved products maintained by the National Security Agency, or by other State or National standards or regulations. PCI Data Security Standard Requirement 3.2

Note: This requirement only applies if previous versions of the payment application stored sensitive authentication data.

1.1.5 Securely delete any sensitive authentication data (pre-authorization data) used for debugging or troubleshooting purposes from log files, debugging files, and other data sources received from customers, to ensure that magnetic stripe data, card validation codes or values, and PINs or PIN block data are not stored on software vendor systems. These data sources must be collected in limited amounts and only when necessary to resolve a problem, encrypted while stored, and deleted immediately after use. PCI Data Security Standard Requirement 3.2

2. Protect stored cardholder data

2.1 Software vendor must provide guidance to customers regarding purging of cardholder data after expiration of customer-defined retention period. PCI Data Security Standard Requirement 3.1

2.2 Mask PAN when displayed (the first six and last four digits are the maximum number of digits to be displayed).

Notes:

- This requirement does not apply to those employees and other parties with a legitimate business need to see full PAN;
- This requirement does not supersede stricter requirements in place for displays of cardholder datafor example, for point-of-sale (POS) receipts. PCI Data Security Standard Requirement 3.3

2.3 Render PAN, at a minimum, unreadable anywhere it is stored, (including data on portable digital media, backup media, and in logs) by using any of the following approaches:

- One-way hashes based on strong cryptography with associated key management processes and procedures
- Truncation

- Index tokens and pads (pads must be securely stored)
- Strong cryptography with associated key management processes and procedures. The MINIMUM account information that must be rendered unreadable is the PAN. PCI Data Security Standard Requirement 3.4

The PAN must be rendered unreadable anywhere it is stored, even outside the payment application. Note: Strong cryptography is defined in the PCI DSS and PA-DSS Glossary of Terms, Abbreviations, and Acronyms.

2.4 If disk encryption is used (rather than file- or column-level database encryption), logical access must be managed independently of native operating system access control mechanisms (for example, by not using local user account databases). Decryption keys must not be tied to user accounts. PCI Data Security Standard Requirement 3.4.2

2.5 Payment application must protect cryptographic keys used for encryption of cardholder data against disclosure and misuse. PCI Data Security Standard Requirement 3.5

2.6 Payment application must implement key management processes and procedures for cryptographic keys used for encryption of cardholder data. PCI Data Security Standard Requirement 3.6

2.7 Securely delete any cryptographic key material or cryptogram stored by previous versions of the payment application, in accordance with industry-accepted standards for secure deletion, as defined, for example the list of approved products maintained by the National Security Agency, or by other State or National standards or regulations. These are cryptographic keys used to encrypt or verify cardholder data. PCI Data Security Standard Requirement 3.6

Note: This requirement only applies if previous versions of the payment application used cryptographic key materials or cryptograms to encrypt cardholder data.

3. Provide secure authentication features

3.1 The payment application must support and enforce unique user IDs and secure authentication for all administrative access and for all access to cardholder data. Secure authentication must be enforced to all accounts, generated or managed by the application by the completion of installation and for subsequent changes after the "out of the box" installation (defined at PCI DSS Requirements 8.1, 8.2, and 8.5.88.5.15) for all administrative access and for all access to cardholder data. PCI Data Security Standard Requirements 8.1, 8.2, and 8.5.88.5.15

Note: These password controls are not intended to apply to employees who only have access to one card number at a time to facilitate a single transaction. These controls are applicable for access by employees with administrative capabilities, for access to servers with cardholder data, and for access controlled by the payment application. This requirement applies to the payment application and all associated tools used to view or access cardholder data. 3.1.10 If a payment application session has been idle for more than 15 minutes, the application requires the user to re-authenticate. PCI Data Security Standard Requirement 8.5.15.

3.2 Software vendors must provide guidance to customers that all access to PCs, servers, and databases with payment applications must require a unique user ID and secure authentication. PCI Data Security Standard Requirements 8.1 and 8.2

3.3 Render payment application passwords unreadable during transmission and storage, using strong cryptography based on approved standards

Note: Strong cryptography is defined in PCI DSS and PA-DSS Glossary of Terms, Abbreviations, and Acronyms. PCI Data Security Standard Requirement 8.4

4. Log payment application activity

4.1 At the completion of the installation process, the out of the box default installation of the payment application must log all user access (especially users with administrative privileges), and be able to link all activities to individual users. PCI Data Security Standard Requirement 10.1

4.2 Payment application must implement an automated audit trail to track and monitor access. PCI Data Security Standard Requirements 10.2 and 10.3

5. Develop secure payment applications

5.1 Develop all payment applications in accordance with PCI DSS (for example, secure authentication and logging) and based on industry best practices and incorporate information security throughout the software development life cycle. These processes must include the following: PCI Data Security Standard Requirement 6.3

5.1.1 Live PANS are not used for testing or development. PCI Data Security Standard Requirement 6.4.4.

- Validation of all input (to prevent cross-site scripting, injection flaws, malicious file execution, etc.)
- Validation of proper error handling
- Validation of secure cryptographic storage
- Validation of secure communications
- Validation of proper role-based access control (RBAC)

5.1.2 Separate development/test, and production environments

5.1.3 Removal of test data and accounts before production systems become active development. PCI Data Security Standard Requirement 6.4.4

5.1.4 Review of payment application code prior to release to customers after any significant change, to identify any potential coding vulnerability. Removal of custom payment application accounts, user IDs, and passwords before payment applications are released to customers

Note: This requirement for code reviews applies to all payment application components (both internal and public-facing web applications), as part of the system development life cycle required by PA-DSS Requirement 5.1 and PCI DSS Requirement 6.3. Code reviews can be conducted by knowledgeable internal personnel or third parties.

5.2 Develop all web payment applications (internal and external, and including web administrative access to product) based on secure coding guidelines such as the Open Web Application Security Project Guide. Cover prevention of common coding vulnerabilities in software development processes, to include:

- Injection flaws, with particular emphasis on SQL injection, Cross-site scripting (XSS) OS Command Injection, LDAP and Xpath injection flaws, as well as other injection flaws.
- Buffer Overflow.
- Insecure cryptographic storage.
- Insecure communications.
- Improper error handling.
- All HIGH vulnerabilities as identified in the vulnerability identification process at PA-DSS Requirement 7.1.
- Cross-site scripting (XSS)
- Improper access control such as insecure direct object references, failure to restrict URL access and directory traversal.
- Cross-site request forgery (CSRF)

Note: The vulnerabilities listed in PA-DSS Requirements 5.2.1 through 5.2.9 and in PCI DSS at 6.5.1 through 6.5.9 were current in the OWASP guide when PCI DSS v1.2 / PCI DSS v2.0 (01/01/10) were published. However, if and when the OWASP guide is updated, the current version must be used for these requirements.

5.3 Software vendor must follow change control procedures for all product software configuration changes. PCI Data Security Standard Requirement 6.4. 5.The procedures must include the following:

- Documentation of impact
- Management sign-off by appropriate parties
- Testing functionality to verify the new change(s) does not adversely impact the security of the system. Remove all testing configurations, samples, and data before finalizing the product for production.

- Back-out or product de-installation procedures

5.4 The payment application must not use or require use of unnecessary and insecure services and protocols (for example, NetBIOS, file-sharing, Telnet, unencrypted FTP must be secured via SSH, S-FTP, SSL, IPSec and other technology to implement end to end security). PCI Data Security Standard Requirement 2.2.2

6. Protect wireless transmissions

6.1 For payment applications using wireless technology, the wireless technology must be implemented securely. Payment applications using wireless technology must facilitate use of industry best practices (for example, IE↩ EE 802.11i) to implement strong encryption for authentication and transmission. Controls must be in place to protect the implemented wireless network from unknown wireless access points and clients. This includes testing the end users wireless deployment on a quarterly basis to detect unauthorized access points within the system. Change wireless vendor defaults, including but not limited to default wireless encryption keys, passwords, and S↩ NMP community strings. Maintain a detailed updated hardware list. The end to end wireless implementation must be end to end secure. The use of WEP as a security control was prohibited as of 30 June 2010. PCI Data Security Standard Requirements 1.2.3, 2.1.1, 4.1.1, 6.2, 11.1a-e and 11.4a-c.

7. Test payment applications to address vulnerabilities

7.1 Software vendors must establish a process to identify newly discovered security vulnerabilities (for example, subscribe to alert services freely available on the Internet) and to test their payment applications for vulnerabilities. Any underlying software or systems that are provided with or required by the payment application (for example, web servers, third-party libraries and programs) must be included in this process. Remove all test configurations, samples, and data after testing and before promoting the changes to production. PCI Data Security Standard Requirement 6.2

7.2 Software vendors must establish a process for timely development and deployment of security patches and upgrades, which includes delivery of updates and patches in a secure manner with a known chain-of-trust, and maintenance of the integrity of patch and update code during delivery and deployment.

8. Facilitate secure network implementation

8.1 The payment application must be able to be implemented into a secure network environment. Application must not interfere with use of devices, applications, or configurations required for PCI DSS compliance (for example, payment application cannot interfere with anti-virus protection, firewall configurations, or any other device, application, or configuration required for PCI DSS compliance). PCI Data Security Standard Requirements 1, 3, 4, 5, and 6.

9. Cardholder data must never be stored on a server connected to the Internet

9.1 The payment application must be developed such that the database server and web server are not required to be on the same server, nor is the database server required to be in the DMZ with the web server. PCI Data Security Standard Requirement 1.3.7

10. Facilitate secure remote software updates

10.1 If payment application updates are delivered securely via remote access into customers systems, software vendors must tell customers to turn on remote-access technologies only when needed for downloads from vendor

and to turn off immediately after download completes. Alternatively, if delivered via VPN or other high-speed connection, software vendors must advise customers to properly configure a firewall or a personal firewall product to secure authentication using a two factor authentication mechanism. PCI Data Security Standard Requirement 8.3

10.2 If payment application may be accessed remotely, remote access to the payment application must be authenticated using a two factor authentication mechanism. PCI Data Security Standard Requirement 8.3

10.3 Any remote access into the payment application must be done securely. If vendors, resellers/integrators, or customers can access customers payment applications remotely, the remote access must be implemented securely. PCI Data Security Standard Requirements 1, 8.3 and 12.3.9

11. Encrypt sensitive traffic over public networks

11.1 If the payment application sends, or facilitates sending, cardholder data over public networks, the payment application must support use of strong cryptography and security protocols such as SSL/TLS and Internet protocol security (IPSEC) to safeguard sensitive cardholder data during transmission over open, public networks. Examples of open, public networks that are in scope of the PCI DSS are: The Internet Wireless technologies Global System for Mobile Communications (GSM) General Packet Radio Service (GPRS) PCI Data Security Standard Requirement 4.1

11.2 The payment application must never send unencrypted PANs by end-user messaging technologies (for example, e-mail, instant messaging, and chat). PCI Data Security Standard Requirement 4.2

12. Encrypt all non-console administrative access

12.1 Instruct customers to encrypt all non-console administrative access using technologies such as SSH, VPN, or SSL/TLS for web-based management and other non-console administrative access. Telnet or remote login must never be used for administrative access. PCI Data Security Standard Requirement 2.3

13. Maintain instructional documentation and training programs for customers, resellers, and integrators

13.1 Develop, maintain, and disseminate a PA-DSS Implementation Guide(s) for customers, resellers, and integrators that accomplishes the following:

- Addresses all requirements in this document wherever the PA-DSS Implementation Guide is referenced.
- Includes a review at least annually and updates to keep the documentation current with all major and minor software changes as well as with changes to the requirements in this document.

13.2 Develop and implement training and communication programs to ensure payment application resellers and integrators know how to implement the payment application and related systems and networks according to the PA-DSS Implementation Guide and in a PCI DSS-compliant manner.

- Update the training materials on an annual basis and whenever new payment application versions are released.

3.5 More Information

IDTech Systems, Inc. highly recommends that merchants contact the card association(s) or their processing company and find out exactly what they mandate and/or recommend. Doing so may help merchants protect themselves from fines and fraud.

For more information related to security, visit:

- http://www.pcisecuritystandards.org
- http://www.visa.com/cisp
- http://www.sans.org/resources
- http://www.microsoft.com/security/default.asp
- https://sdp.mastercardintl.com/
- http://www.americanexpress.com/merchantspecs

CAPN questions: capninfocenter@aexp.com

Chapter 4

SecureMag Main Transaction Commands

The methods below are provided as a reference to the main commands needed to collect MSR information from a swipe.

4.1 Transaction Methods

Start Transaction

com.idtechproducts.device.IDT_SecureMag.msr_startMSRSwipe()

- msr = start a transaction on contactless/MSR interfaces ONLY

4.2 MSR

Request Swipe

com.idtechproducts.device.IDT_SecureMag.msr_startMSRSwipe()
Enables MSR to receive Swipe.

Cancel Swipe

com.idtechproducts.device.IDT_SecureMag.msr_cancelMSRSwipe()
Cancels the Swipe request.

Chapter 5

Sending Direct Commands

The main purpose of Android library for SecureMag is to expedite integration to the device by providing the connectivity and communication protocols. It also provides the main functions to get device info and perform swipe transactions.

Chapter 6

Core Implementation SecureMag: Android

Universal_SDK_X.XX.XXX.jar includes class libraries to interface with the SecureMag. This guide assume a fair understanding of Eclipse IDE and general Android programming knowledge.

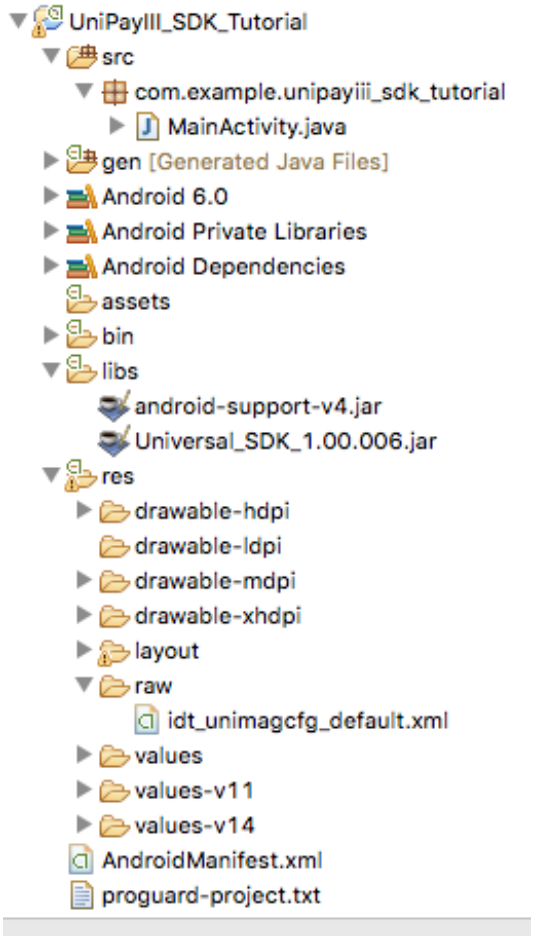
6.1 Integrating with Android SDK

- Import the necessary libraries
- Add Import statements to utilize libraries
- Implement OnReceiverListener for the activity
- Enable permissions for the application
- Allocate/initialize SecureMag objects
- Sample Project Tutorial Eclipse
- Sample Project Tutorial Android Studio

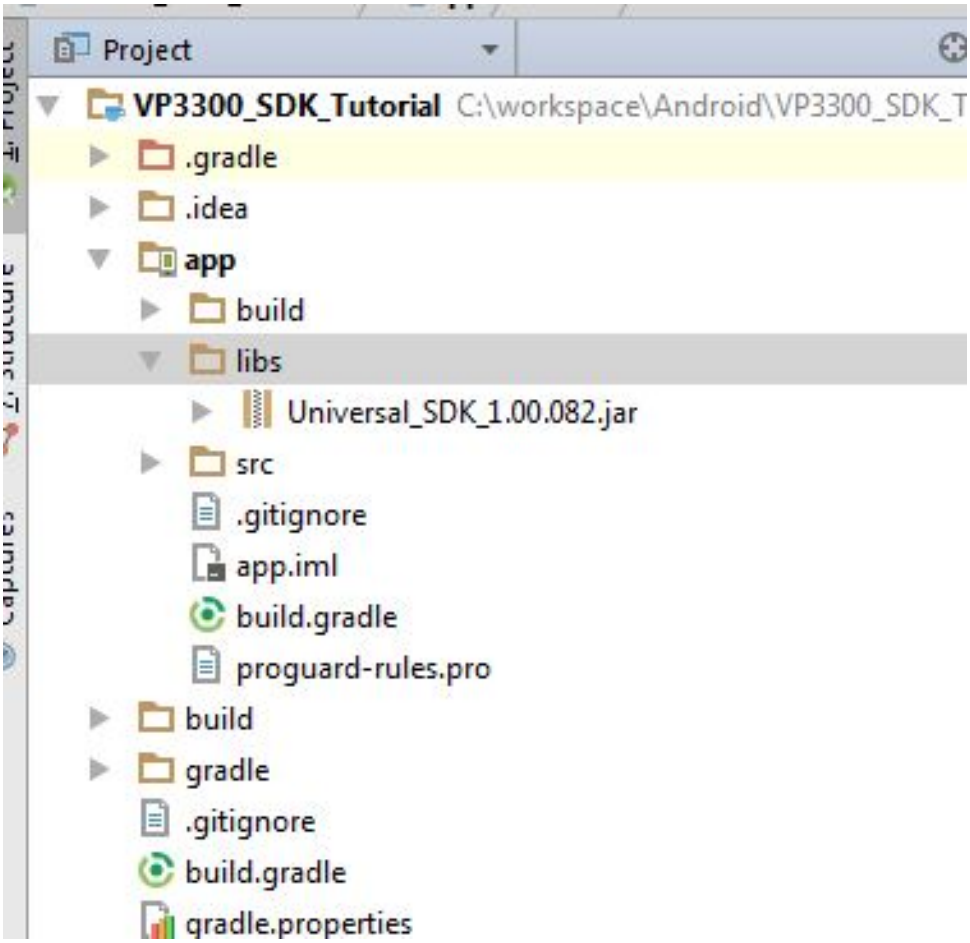
6.2 Import the necessary libraries

Communicating with SecureMag requires the Universal_SDK_X.XX.XXX.jar file be added to the project. While build paths can be created pointing to the Universal_SDK_X.XX.XXX.jar file, the simplest solution is to add the jar file to the projects libs folder.

Eclispe:



Android Studio:



6.3 Add Import statements to utilize libraries

In the header files of the java activity that will access SecureMag, use import statement utilize the library:

```
import com.idtechproducts.device.*;
```

```
1  package com.example.unipayiii_sdk_tutorial;
2
3  import android.app.Activity;
4  import android.os.Bundle;
5  import com.idtechproducts.device.*;
6
```

The complete import list is as follows:

```
import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.util.Set;

import android.app.Activity;
import android.os.Bundle;
import android.os.Handler;
import android.util.Log;
```

```
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

import com.idtechproducts.device.*;
```

6.4 Implement OnReceiverListener for the activity:

In the class that will be a delegate of SecureMag, implement OnReceiverListener. Add the implemented methods to eliminate any error messages.

```
public class MainActivity extends Activity implements  OnReceiverListener {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public void ICCNotifyInfo(byte[] arg0, String arg1) {
        // TODO Auto-generated method stub

    }

    @Override
    public void LoadXMLConfigFailureInfo(int arg0, String arg1) {
        // TODO Auto-generated method stub

    }

    @Override
    public void autoConfigCompleted(StructConfigParameters arg0) {
        // TODO Auto-generated method stub

    }

    @Override
    public void autoConfigProgress(int arg0) {
        // TODO Auto-generated method stub

    }

    @Override
    public void deviceConnected() {
        // TODO Auto-generated method stub

    }

    @Override
    public void deviceDisconnected() {
        // TODO Auto-generated method stub

    }

    @Override
    public void emvTransactionData(IDTEMVData arg0) {
        // TODO Auto-generated method stub

    }

    @Override
    public void lcdDisplay(int arg0, String[] arg1, int arg2) {
        // TODO Auto-generated method stub

    }

    @Override
    public void msgAudioVolumeAjustFailed() {
        // TODO Auto-generated method stub

    }

    @Override
    public void msgRKICompleted(String arg0) {
        // TODO Auto-generated method stub

    }

    @Override
    public void msgToConnectDevice() {
```

```
        // TODO Auto-generated method stub

    }

    @Override
    public void swipeMSRData(IDTMSRData arg0) {
        // TODO Auto-generated method stub

    }

    @Override
    public void timeout(int arg0) {
        // TODO Auto-generated method stub

    }
}
```

6.5 Enable permissions for the application:

```
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS"/>
<uses-permission android:name="android.permission.RECORD_AUDIO"/>
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-feature android:name="android.hardware.usb.host" />
```

6.6 Allocate/initialize SecureMag objects:

Initialize IDT_Device object by passing context and OnReceiverListener delegate.

```
// declaring the instance of the SecureMagReader;
private IDT_SecureMag mySecureMagReader = null;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    if(mySecureMagReader!=null){
        mySecureMagReader.unregisterListen();
        mySecureMagReader.release();
        mySecureMagReader = null;
    }
    mySecureMagReader = new IDT_SecureMag(this,getActivity());
    mySecureMagReader.device_setDeviceType(DEVICE_TYPE.DEVICE_SECUREMAG);
    mySecureMagReader.registerListen();
}
```

6.7 Sample Project Tutorial Eclipse

Using Eclipse, we will create a sample project that will interface with the SecureMag and will perform the following activities:

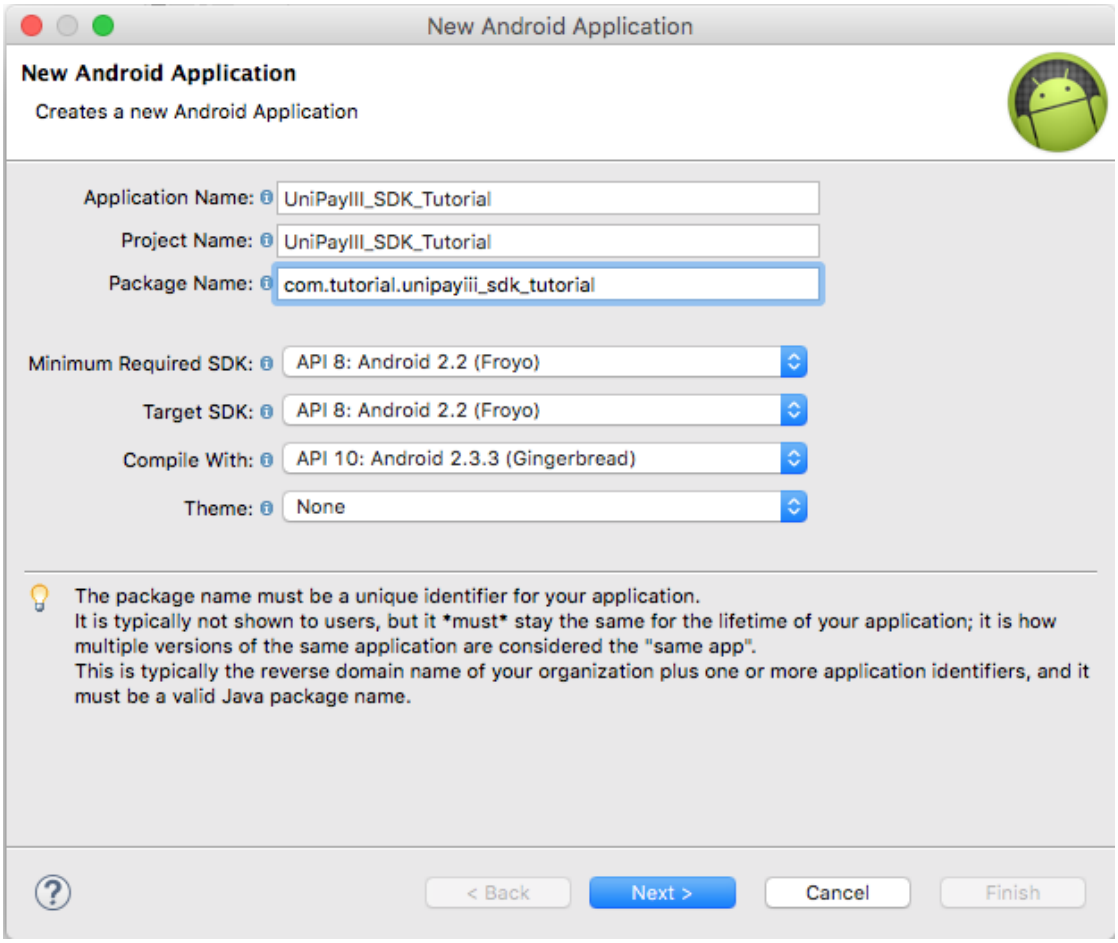
- Auto-Connect and display connection status
- Get Device Firmware
- Start/Stop Transaction Request for MSR (swipe)

Listeners:

- Listener to receive card swipes
- Listener to detect device connected
- Listener to detect device disconnected

6.7.1 Step 1: Create New Project

Create a new Android Application project in Eclipse as an Empty Activity



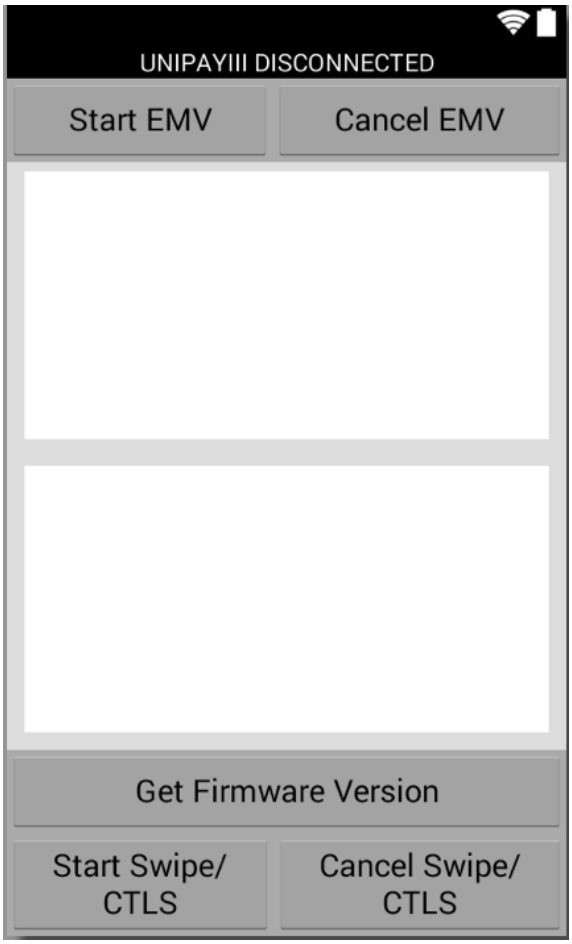
6.7.2 Step 2: Import IDTechSDK for SecureMag

Import the necessary libraries

6.7.3 Step 3: Design Interface

Design the User Interface by editing the main layout XML file
Open your layout and add items to so it contains the following buttons/fields (sample code provide at end of section):

- Add a TextView to the top that will signify connection/disconnection status.
- Add two TextViews to communicate data from the SecureMag. Remove the Editable behavior if you don't want the keyboard to pop up if you accidentally select it.
- Add buttons to execute the following functions:
  - Get Firmware
  - Start MSR
  - Cancel Transaction



6.7.4 Step 4: Configure Activity File

In the activity file, perform the following:

- Add Import statements to utilize libraries
- Implement OnReceiverListener for the activity
- Enable permissions for the application
- Define the association for all the elements on the layout: The connection label, the two text views, and the 5 buttons

Layout Source Code

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#aaaaaa"
    android:orientation="vertical" >
    <TextView
        android:id="@+id/status_text"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:background="#000000"
        android:gravity="center_vertical|center_horizontal"
        android:text="SecureMag DISCONNECTED"
        android:textColor="#FFFFFF" />
    <LinearLayout
        android:id="@+id/linearLayoutEditText"
        android:layout_width="match_parent"
        android:layout_height="fill_parent"
        android:layout_weight="1"
```

```
        android:background="#dddddd"
        android:focusable="true"
        android:focusableInTouchMode="true"
        android:orientation="vertical" >
        <ScrollView
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_marginBottom="10dp"
            android:layout_marginLeft="10dp"
            android:layout_marginRight="10dp"
            android:layout_marginTop="5dp"
            android:background="#ffffff" >
            <TextView
                android:id="@+id/lcdLog"
                android:layout_width="fill_parent"
                android:layout_height="fill_parent"
                android:text=""
                android:textColor="#000000"
                android:textSize="12sp"
                android:typeface="monospace" >
            </TextView>
        </ScrollView>
    </LinearLayout>
        <LinearLayout
        android:id="@+id/linearLayoutEditText2"
        android:layout_width="match_parent"
        android:layout_height="fill_parent"
        android:layout_weight="1"
        android:background="#dddddd"
        android:focusable="true"
        android:focusableInTouchMode="true"
        android:orientation="vertical" >
        <ScrollView
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_marginBottom="10dp"
            android:layout_marginLeft="10dp"
            android:layout_marginRight="10dp"
            android:layout_marginTop="5dp"
            android:background="#ffffff" >
            <TextView
                android:id="@+id/textLog"
                android:layout_width="fill_parent"
                android:layout_height="fill_parent"
                android:text=""
                android:textColor="#000000"
                android:textSize="12sp"
                android:typeface="monospace" >
            </TextView>
        </ScrollView>
    </LinearLayout>
        <LinearLayout
        android:id="@+id/linearLayoutBottom"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        >
        <Button
            android:id="@+id/btn_getFirmware"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.53"
            android:gravity="center_vertical|center_horizontal"
            android:text="Get Firmware Version" />
    </LinearLayout>
    <LinearLayout
        android:id="@+id/linearLayoutBottom4"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        >

        <Button
            android:id="@+id/btn_startSwipe"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.53"
            android:text="Start Swipe" />
        <Button
            android:id="@+id/btn_cancelSwipe"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.53"
            android:text="Cancel Swipe" />
    </LinearLayout>
</LinearLayout>
```

6.7.5 Step 5: Configure Method File

In the activity file, perform the following:

- set delegate and initialize SecureMag object in the onCreate method. Reference: Allocate/initialize SecureMag objects
- set correct device type.

```
// declaring the instance of the SecureMagReader;
private IDT_SecureMag mySecureMagReader = null;
private TextView connectStatusTextView;
private TextView textLog;
private TextView lcdLog;
private Button btnGetFirmware;
private Button btnStartSwipe;
private Button btnCancelSwipe;
private Handler handler = new Handler();
private boolean isReaderConnected = false;
private String info = "";
private String detail = "";
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    handler = new Handler();
    btnGetFirmware = (Button)findViewById(R.id.btn_getFirmware);
    btnStartSwipe = (Button)findViewById(R.id.btn_startSwipe);
    btnCancelSwipe = (Button)findViewById(R.id.btn_cancelSwipe);
    textLog = (TextView)findViewById(R.id.textLog);
    lcdLog = (TextView)findViewById(R.id.lcdLog);
    connectStatusTextView = (TextView)findViewById(R.id.status_text);
    if(mySecureMagReader!=null){
        mySecureMagReader.unregisterListen();
        mySecureMagReader.release();
        mySecureMagReader = null;
    }
    mySecureMagReader = new IDT_SecureMag(this,this);
    mySecureMagReader.device_setDeviceType(DEVICE_TYPE.DEVICE_SECUREMAG);
    mySecureMagReader.registerListen();
}
```

- Implement protocol delegate com.idtechproducts.device.OnReceiverListener.deviceConnected() and com.idtechproducts.device.OnReceiverListener.deviceDisconnected() to monitor connect/disconnect events and modify our connection label upon change. Reference: Implement OnReceiverListener for the activity
  Note: This notification may come back on a thread different that the UI thread, so we want to make sure to use a handler to send to main UI thread.

```
private Runnable doUpdateLabel = new Runnable()
{
    public void run()
    {
        if(!isReaderConnected){
            connectStatusTextView.setText("SecureMag DISCONNECTED");
        }
        else{
            connectStatusTextView.setText("SecureMag CONNECTED");
        }
    }
};
@Override
public void deviceConnected() {
    isReaderConnected = true;
    handler.post(doUpdateLabel);
}

@Override
public void deviceDisconnected() {
    isReaderConnected = false;
    handler.post(doUpdateLabel);
}
```

-Implement protocol delegate com.idtechproducts.device.OnReceiverListener.swipeMSRData() to receive unsolicited card swipe data.

```
private Runnable doUpdateStatus = new Runnable()
```

```
{
    public void run()
    {
        lcdLog.setText(info);
        textLog.setText(detail);
    }
};
@Override
public void swipeMSRData(IDTMSRData card) {
    if (card.cardData[0] != (byte)0x01 && card.track1Length == 0 && card.track2Length == 0 && card.
      track3Length == 0)
        info = "Swipe/Tap data didn't read correctly";
    else
        info = "Swipe/Tap Read Successfully";
    detail = Common.parse_MSRData(myVPSecureMagReader.device_getDeviceType(), card);
    handler.post(doUpdateStatus);

}
```

- Implement the button press methods

```
btnGetFirmware.setOnClickListener(new Button.OnClickListener(){
    public void onClick(View v) {
        info = "Getting Firmware\n";
        detail = "";
        handler.post(doUpdateStatus);
        StringBuilder sb = new StringBuilder();
        int ret = mySecureMagReader.device_getFirmwareVersion(sb);
        if (ret == ErrorCode.SUCCESS) {
            info += "Firmware Version: " + sb.toString();
            detail = "";
            handler.post(doUpdateStatus);
        }
        else {
            info += "GetFirmwareVersion: Failed\n";
            info += "Status: "+ mySecureMagReader.device_getResponseCodeString(ret)+"";
            detail = "";
            handler.post(doUpdateStatus);
            }
    }
});

btnStartSwipe.setOnClickListener(new Button.OnClickListener(){
    public void onClick(View v) {
        detail = "";
        info = "Starting Swipe/Tap Transaction\n";
        handler.post(doUpdateStatus);
        mySecureMagReader.msr_startMSRSwipe();
    }
});

btnCancelSwipe.setOnClickListener(new Button.OnClickListener(){
    public void onClick(View v) {
        detail = "";
        info = "Cancelling Swipe/Tap Transaction\n";
        handler.post(doUpdateStatus);
        mySecureMagReader.msr_cancelMSRSwipe();
    }
});
```

6.7.6 Complete code listing

```
package com.example.securemag_sdk_tutorial;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.util.Set;

import android.app.Activity;
import android.app.Dialog;
import android.os.Bundle;
import android.os.Handler;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.TextView;
```

```
import com.idtechproducts.device.*;
import com.idtechproducts.device.ReaderInfo.DEVICE_TYPE;
import com.idtechproducts.device.ReaderInfo.
      CAPTURE_ENCODE_TYPE;
import com.idtechproducts.device.ReaderInfo.
      CAPTURE_ENCRYPT_TYPE;
import com.idtechproducts.device.ReaderInfo.
      EVENT_MSR_Types;

public class MainActivity extends Activity implements  OnReceiverListener{

    // declaring the instance of the SecureMagReader;
    private IDT_SecureMag mySecureMagReader = null;
    private TextView connectStatusTextView;
    private TextView textLog;
    private TextView lcdLog;
    private Button btnGetFirmware;
    private Button btnStartSwipe;
    private Button btnCancelSwipe;
    private Handler handler = new Handler();
    private boolean isReaderConnected = false;
    private String info = "";
    private String detail = "";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        handler = new Handler();
        btnGetFirmware = (Button)findViewById(R.id.btn_getFirmware);
        btnStartSwipe = (Button)findViewById(R.id.btn_startSwipe);
        btnCancelSwipe = (Button)findViewById(R.id.btn_cancelSwipe);
        textLog = (TextView)findViewById(R.id.textLog);
        lcdLog = (TextView)findViewById(R.id.lcdLog);
        connectStatusTextView = (TextView)findViewById(R.id.status_text);
        if(mySecureMagReader!=null){
            mySecureMagReader.unregisterListen();
            mySecureMagReader.release();
            mySecureMagReader = null;
        }
        mySecureMagReader = new IDT_SecureMag(this,this);
        mySecureMagReader.device_setDeviceType(DEVICE_TYPE.DEVICE_SECUREMAG);
        mySecureMagReader.registerListen();
        loadXMLfile();


        btnGetFirmware.setOnClickListener(new Button.OnClickListener(){
            public void onClick(View v) {
                info = "Getting Firmware\n";
                detail = "";
                handler.post(doUpdateStatus);
                StringBuilder sb = new StringBuilder();
                int ret = mySecureMagReader.device_getFirmwareVersion(sb);
                if (ret == ErrorCode.SUCCESS) {
                    info += "Firmware Version: " + sb.toString();
                    detail = "";
                    handler.post(doUpdateStatus);
                }
                else {
                    info += "GetFirmwareVersion: Failed\n";
                    info += "Status: "+ mySecureMagReader.device_getResponseCodeString(ret)+"";
                    detail = "";
                    handler.post(doUpdateStatus);
                    }
            }
        });

        btnStartSwipe.setOnClickListener(new Button.OnClickListener(){
            public void onClick(View v) {
                detail = "";
                info = "Starting Swipe/Tap Transaction\n";
                handler.post(doUpdateStatus);
                mySecureMagReader.msr_startMSRSwipe();
            }
        });

        btnCancelSwipe.setOnClickListener(new Button.OnClickListener(){
            public void onClick(View v) {
                detail = "";
                info = "Cancelling Swipe/Tap Transaction\n";
                handler.post(doUpdateStatus);
                mySecureMagReader.msr_cancelMSRSwipe();
            }
        });

    }

    @Override
```

```
public void ICCNotifyInfo(byte[] arg0, String arg1) {
    // TODO Auto-generated method stub

}

@Override
public void LoadXMLConfigFailureInfo(int arg0, String arg1) {
    // TODO Auto-generated method stub

}

@Override
public void autoConfigCompleted(StructConfigParameters arg0) {
    // TODO Auto-generated method stub

}

@Override
public void autoConfigProgress(int arg0) {
    // TODO Auto-generated method stub

}

private Runnable doUpdateLabel = new Runnable()
{
    public void run()
    {
        if(!isReaderConnected){
            connectStatusTextView.setText("SecureMag DISCONNECTED");
        }
        else{
            connectStatusTextView.setText("SecureMag CONNECTED");
        }
    }
};
@Override
public void deviceConnected() {
    isReaderConnected = true;
    handler.post(doUpdateLabel);
}

@Override
public void deviceDisconnected() {
    isReaderConnected = false;
    handler.post(doUpdateLabel);
}

private void printTags(IDTEMVData emvData)
{

}

@Override
public void emvTransactionData(IDTEMVData emvData) {

}

public void lcdDisplay(int mode, String[] lines, int timeout) {

}


@Override
public void msgAudioVolumeAjustFailed() {
    // TODO Auto-generated method stub

}

@Override
public void msgRKICompleted(String arg0) {
    // TODO Auto-generated method stub

}

@Override
public void msgToConnectDevice() {
    // TODO Auto-generated method stub

}

private Runnable doUpdateStatus = new Runnable()
{
    public void run()
    {
        lcdLog.setText(info);
        textLog.setText(detail);
    }
```

```
    };
    @Override
    public void swipeMSRData(IDTMSRData card) {
        if (card.cardData[0] != (byte)0x01 && card.track1Length == 0 && card.track2Length == 0 && card.
      track3Length == 0)
            info = "Swipe/Tap data didn't read correctly";
        else
            info = "Swipe/Tap Read Successfully";
        detail = Common.parse_MSRData(mySecureMagReader.device_getDeviceType(), card);
        handler.post(doUpdateStatus);

    }

    @Override
    public void timeout(int arg0) {
        // TODO Auto-generated method stub

    }

    private String getXMLFileFromRaw(String fileName ,int res){
        //the target filename in the application path
        String fileNameWithPath = null;
        fileNameWithPath = fileName;
        String newFilename = fileName;

        try {
            InputStream in = getResources().openRawResource(res);
            int length = in.available();
            byte [] buffer = new byte[length];
            in.read(buffer);
            in.close();
            deleteFile(fileNameWithPath);
            FileOutputStream fout = openFileOutput(fileNameWithPath, MODE_PRIVATE);
            fout.write(buffer);
            fout.close();

            // to refer to the application path
            File fileDir = this.getFilesDir();
            fileNameWithPath = fileDir.getParent() + java.io.File.separator + fileDir.getName();
            fileNameWithPath += java.io.File.separator+newFilename;

        } catch(Exception e){
            e.printStackTrace();
            fileNameWithPath = null;
        }
        return fileNameWithPath;
    }

    private String getConfigurationFileFromRaw( ){
        return getXMLFileFromRaw("idt_unimagcfg_default.xml",R.raw.idt_unimagcfg_default);
        }
    private boolean isFileExist(String path) {
        if(path==null)
            return false;
        File file = new File(path);
        if (!file.exists()) {
          return false ;
        }
        return true;
    }
    private void loadXMLfile(){

        //load the XML configuration file
        String fileNameWithPath = getConfigurationFileFromRaw();
        if(!isFileExist(fileNameWithPath)) {
            fileNameWithPath = null;
        }
        // Network operation is prohibited in the UI Thread if target API is 11 or above.
        // If target API is 11 or above, please use AsyncTask to avoid errors.
        mySecureMagReader.config_setXMLFileNameWithPath(fileNameWithPath);
        Log.d("Demo Info >>>>>","loadingConfigurationXMLFile begin.");
        mySecureMagReader.config_loadingConfigurationXMLFile(true);
    }



}
```

6.8 Sample Project Tutorial Android Studio

Using Android Studio, we will create a sample project that will interface with the SecureMag and will perform the following activities:

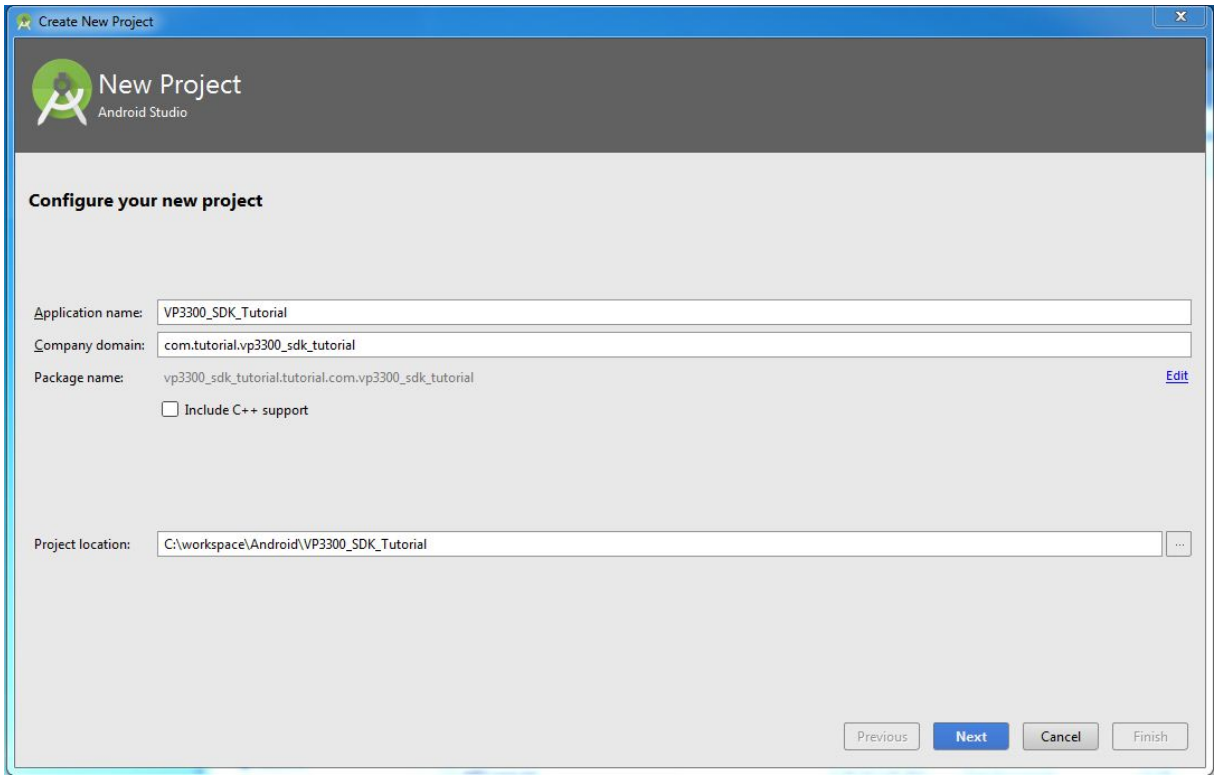
- Auto-Connect and display connection status
- Get Device Firmware
- Start/Stop Transaction Request for MSR (tap/swipe)

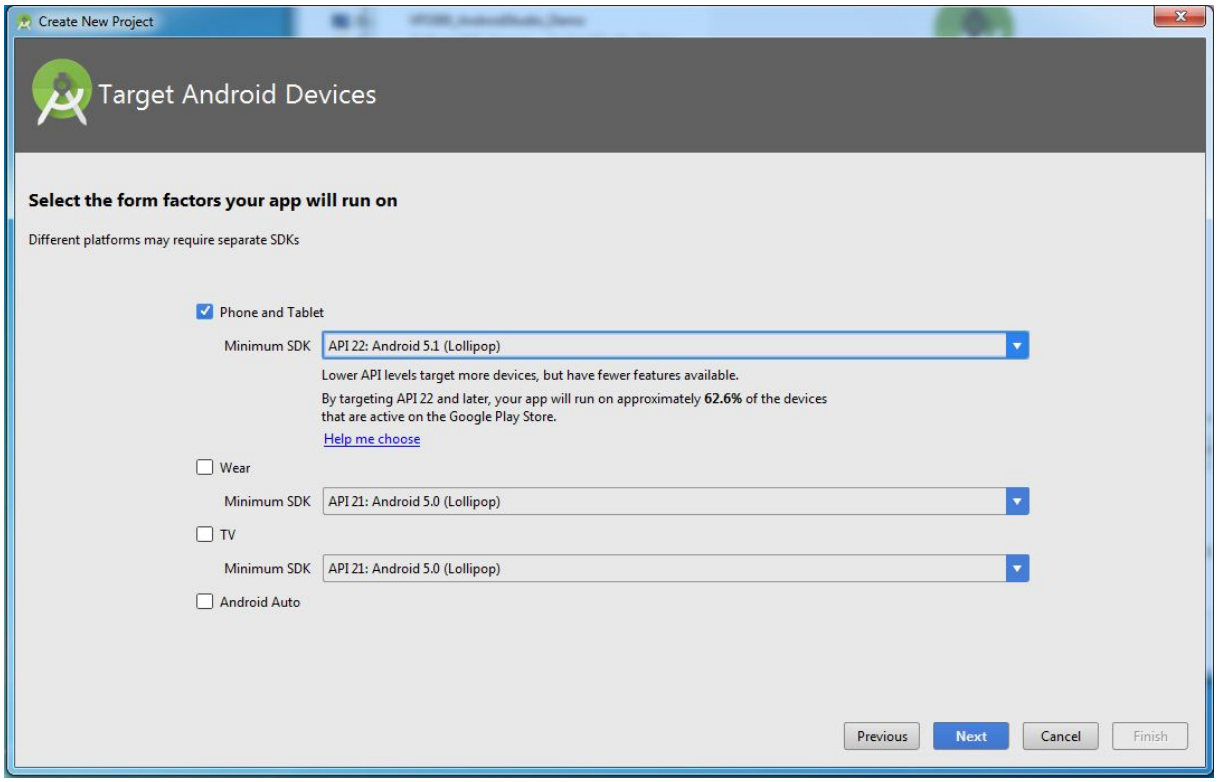
Listeners:

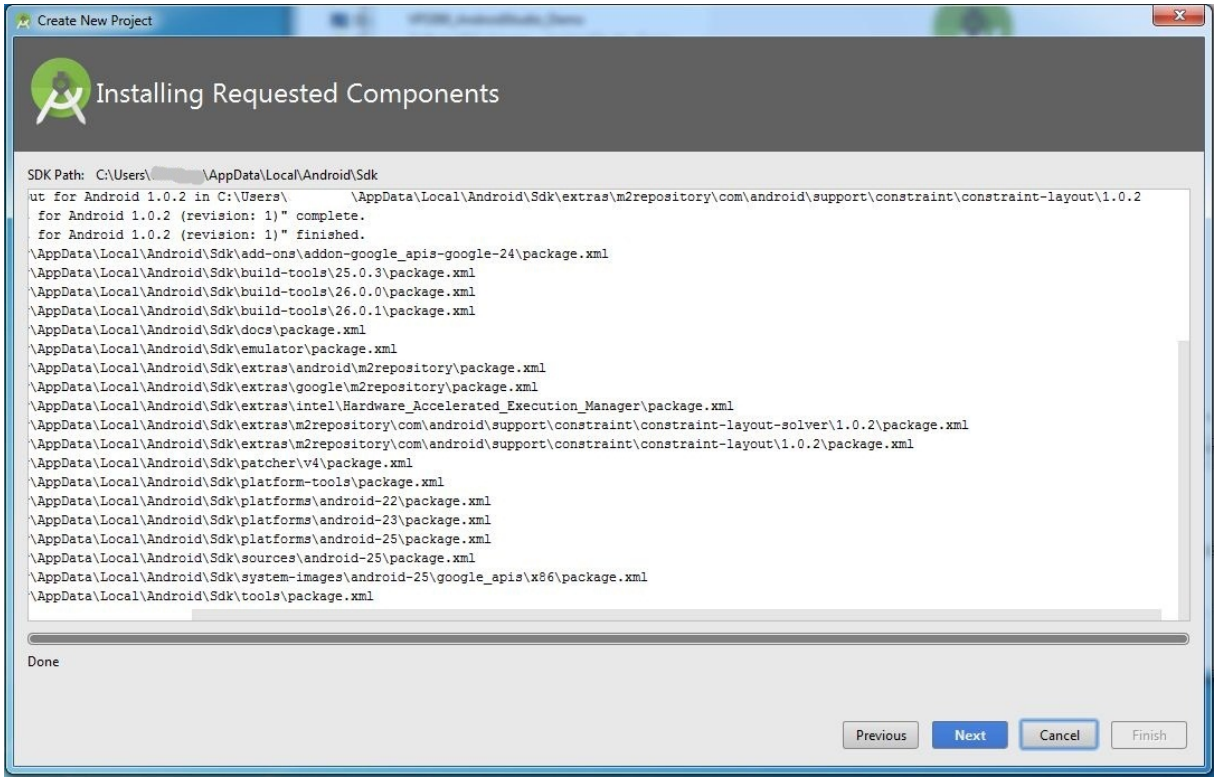
- Listener to receive card swipes
- Listener to detect device connected
- Listener to detect device disconnected

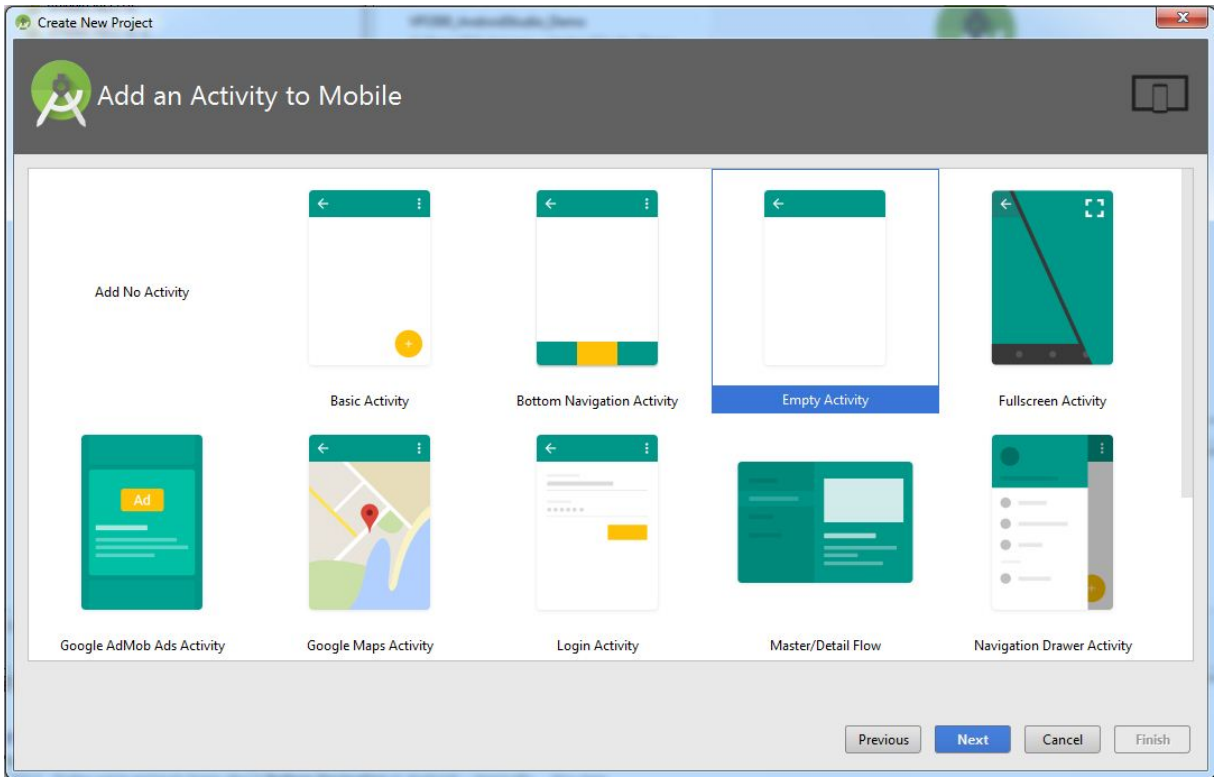
6.8.1 Step 1: Create New Project

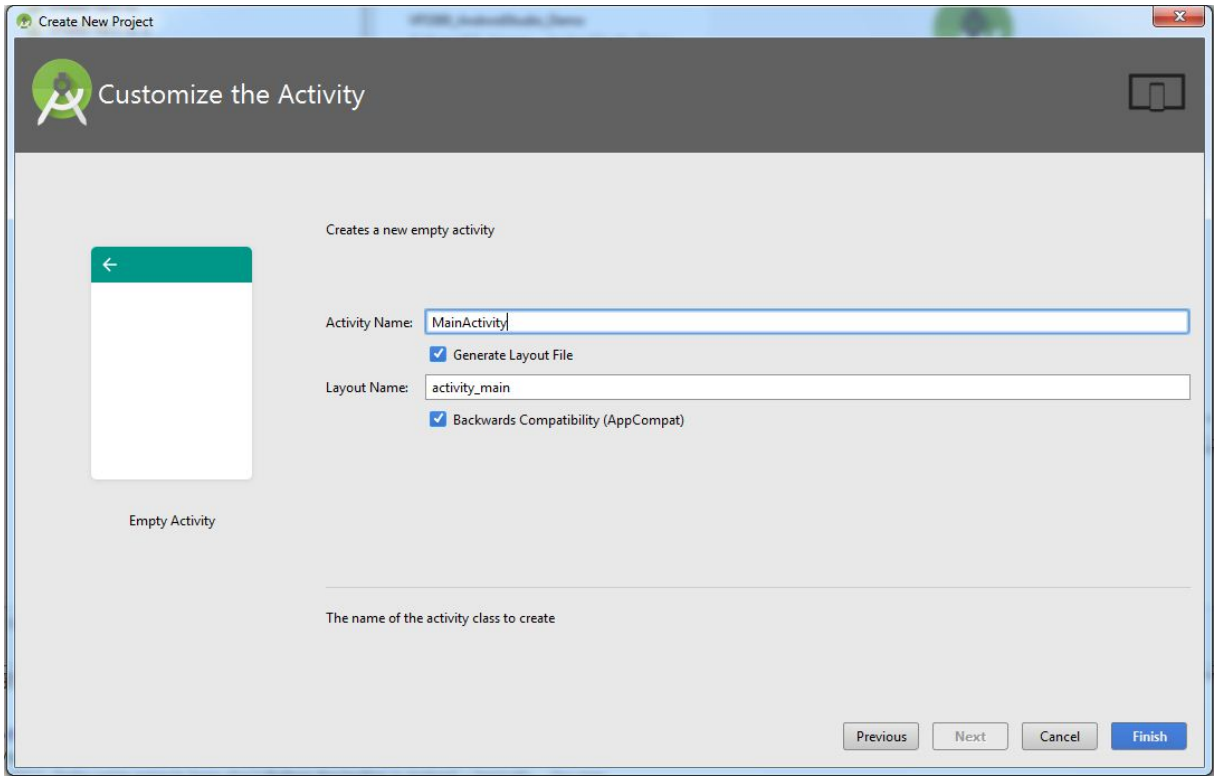
Create a new Android Application project in Android Studio as an Empty Activity




Create New Project
Target Android Devices
Select the form factors your app will run on
Different platforms may require separate SDKs
Phone and Tablet
Minimum SDK
API 22: Android 5.1 (Lollipop)
Lower API levels target more devices, but have fewer features available.
By targeting API 22 and later, your app will run on approximately 62.6% of the devices that are active on the Google Play Store.
Help me choose
Wear
Minimum SDK
API 21: Android 5.0 (Lollipop)
TV
Minimum SDK
API 21: Android 5.0 (Lollipop)
Android Auto
Previous
Next
Cancel
Finish


Create New Project
Installing Requested Components
SDK Path: C:\Users\ \AppData\Local\Android\Sdk
ut for Android 1.0.2 in C:\Users\ \AppData\Local\Android\Sdk\extras\m2repository\com\android\support\constraint\constraint-layout\1.0.2
for Android 1.0.2 (revision: 1)" complete.
for Android 1.0.2 (revision: 1)" finished.
\AppData\Local\Android\Sdk\add-ons\addon-google_apis-google-24\package.xml
\AppData\Local\Android\Sdk\build-tools\25.0.3\package.xml
\AppData\Local\Android\Sdk\build-tools\26.0.0\package.xml
\AppData\Local\Android\Sdk\build-tools\26.0.1\package.xml
\AppData\Local\Android\Sdk\docs\package.xml
\AppData\Local\Android\Sdk\emulator\package.xml
\AppData\Local\Android\Sdk\extras\android\m2repository\package.xml
\AppData\Local\Android\Sdk\extras\google\m2repository\package.xml
\AppData\Local\Android\Sdk\extras\intel\Hardware_Accelerated_Execution_Manager\package.xml
\AppData\Local\Android\Sdk\extras\m2repository\com\android\support\constraint\constraint-layout-solver\1.0.2\package.xml
\AppData\Local\Android\Sdk\extras\m2repository\com\android\support\constraint\constraint-layout\1.0.2\package.xml
\AppData\Local\Android\Sdk\patcher\v4\package.xml
\AppData\Local\Android\Sdk\platform-tools\package.xml
\AppData\Local\Android\Sdk\platforms\android-22\package.xml
\AppData\Local\Android\Sdk\platforms\android-23\package.xml
\AppData\Local\Android\Sdk\platforms\android-25\package.xml
\AppData\Local\Android\Sdk\sources\android-25\package.xml
\AppData\Local\Android\Sdk\system-images\android-25\google_apis\x86\package.xml
\AppData\Local\Android\Sdk\tools\package.xml
Done
Previous
Next
Cancel
Finish





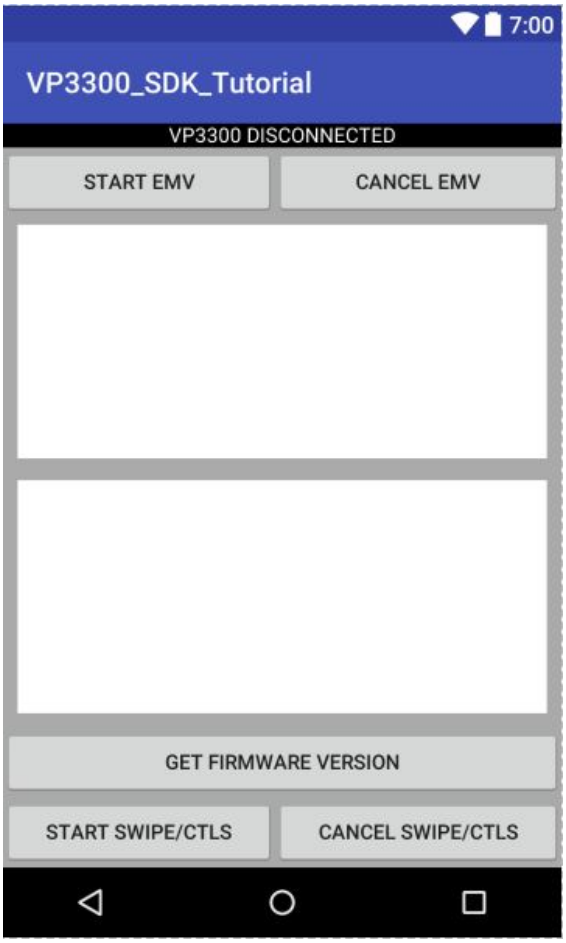
6.8.2 Step 2: Import IDTechSDK for SecureMag

Import the necessary libraries

6.8.3 Step 3: Design Interface

Design the User Interface by editing the main layout XML file
Open your layout and add items to so it contains the following buttons/fields (sample code provide at end of section):

- Add a TextView to the top that will signify connection/disconnection status.
- Add two TextViews to communicate data from the SecureMag. Remove the Editable behavior if you don't want the keyboard to pop up if you accidentally select it.
- Add buttons to execute the following functions:
  - Get Firmware
  - Start MSR/ CTLS
  - Cancel Transaction



6.8.4 Step 4: Configure Activity File

In the activity file, perform the following:

- Add Import statements to utilize libraries
- Implement OnReceiverListener for the activity
- Enable permissions for the application

- Define the association for all the elements on the layout: The connection label, the two text views, and the 5 buttons

Layout Source Code

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#aaaaaa"
    android:orientation="vertical" >
    <TextView
        android:id="@+id/status_text"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:background="#000000"
        android:gravity="center_vertical|center_horizontal"
        android:text="SecureMag DISCONNECTED"
        android:textColor="#FFFFFF" />
    <LinearLayout
        android:id="@+id/linearLayoutEditText"
        android:layout_width="match_parent"
        android:layout_height="fill_parent"
        android:layout_weight="1"
        android:background="#dddddd"
        android:focusable="true"
        android:focusableInTouchMode="true"
        android:orientation="vertical" >
        <ScrollView
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_marginBottom="10dp"
            android:layout_marginLeft="10dp"
            android:layout_marginRight="10dp"
            android:layout_marginTop="5dp"
            android:background="#ffffff" >
            <TextView
                android:id="@+id/lcdLog"
                android:layout_width="fill_parent"
                android:layout_height="fill_parent"
                android:text=""
                android:textColor="#000000"
                android:textSize="12sp"
                android:typeface="monospace" >
            </TextView>
        </ScrollView>
    </LinearLayout>
        <LinearLayout
        android:id="@+id/linearLayoutEditText2"
        android:layout_width="match_parent"
        android:layout_height="fill_parent"
        android:layout_weight="1"
        android:background="#dddddd"
        android:focusable="true"
        android:focusableInTouchMode="true"
        android:orientation="vertical" >
        <ScrollView
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_marginBottom="10dp"
            android:layout_marginLeft="10dp"
            android:layout_marginRight="10dp"
            android:layout_marginTop="5dp"
            android:background="#ffffff" >
            <TextView
                android:id="@+id/textLog"
                android:layout_width="fill_parent"
                android:layout_height="fill_parent"
                android:text=""
                android:textColor="#000000"
                android:textSize="12sp"
                android:typeface="monospace" >
            </TextView>
        </ScrollView>
    </LinearLayout>
        <LinearLayout
        android:id="@+id/linearLayoutBottom"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        >
        <Button
            android:id="@+id/btn_getFirmware"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.53"
```

```
            android:gravity="center_vertical|center_horizontal"
            android:text="Get Firmware Version" />
    </LinearLayout>
    <LinearLayout
        android:id="@+id/linearLayoutBottom4"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        >

        <Button
            android:id="@+id/btn_startSwipe"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.53"
            android:text="Start Swipe/CTLS" />
        <Button
            android:id="@+id/btn_cancelSwipe"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="0.53"
            android:text="Cancel Swipe/CTLS" />
    </LinearLayout>
</LinearLayout>
```

6.8.5 Step 5: Configure Method File

In the activity file, perform the following:

- set delegate and initialize SecureMag object in the onCreate method. Reference: Allocate/initialize SecureMag objects
- set correct device type.

```
// declaring the instance of the SecureMagReader;
private IDT_SecureMag mySecureMagReader = null;
private TextView connectStatusTextView;
private TextView textLog;
private TextView lcdLog;
private Button btnGetFirmware;
private Button btnStartSwipe;
private Button btnCancelSwipe;
private Handler handler = new Handler();
private boolean isReaderConnected = false;
private String info = "";
private String detail = "";
private BluetoothAdapter mBtAdapter = null;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    handler = new Handler();
    btnGetFirmware = (Button)findViewById(R.id.btn_getFirmware);
    btnStartSwipe = (Button)findViewById(R.id.btn_startSwipe);
    btnCancelSwipe = (Button)findViewById(R.id.btn_cancelSwipe);
    textLog = (TextView)findViewById(R.id.textLog);
    lcdLog = (TextView)findViewById(R.id.lcdLog);
    connectStatusTextView = (TextView)findViewById(R.id.status_text);
    if(mySecureMagReader!=null){
        mySecureMagReader.unregisterListen();
        mySecureMagReader.release();
        mySecureMagReader = null;
    }
    mySecureMagReader = new IDT_SecureMag(this,this);
    mySecureMagReader.device_setDeviceType(DEVICE_TYPE.DEVICE_SECUREMAG);
    mySecureMagReader.registerListen();

}
```

- Implement protocol delegate com.idtechproducts.device.OnReceiverListener.deviceConnected() and com.idtechproducts.device.OnReceiverListener.deviceDisconnected() to monitor connect/disconnect events and modify our connection label upon change. Reference: Implement OnReceiverListener for the activity
  Note: This notification may come back on a thread different that the UI thread, so we want to make sure to use a handler to send to main UI thread.

```
private Runnable doUpdateLabel = new Runnable()
{
```

```
    public void run()
    {
        if(!isReaderConnected){
            connectStatusTextView.setText("SecureMag DISCONNECTED");
        }
        else{
            connectStatusTextView.setText("SecureMag CONNECTED");
        }
    }
};
@Override
public void deviceConnected() {
    isReaderConnected = true;
    handler.post(doUpdateLabel);
}

@Override
public void deviceDisconnected() {
    isReaderConnected = false;
    handler.post(doUpdateLabel);
}
```

-Implement protocol delegate com.idtechproducts.device.OnReceiverListener.swipeMSRData() to receive unsolicited card swipe data.

```
private Runnable doUpdateStatus = new Runnable()
{
    public void run()
    {
        lcdLog.setText(info);
        textLog.setText(detail);
    }
};
@Override
public void swipeMSRData(IDTMSRData card) {
    if (card.cardData[0] != (byte)0x01 && card.track1Length == 0 && card.track2Length == 0 && card.
      track3Length == 0)
        info = "Swipe/Tap data didn't read correctly";
    else
        info = "Swipe/Tap Read Successfully";
    detail = Common.parse_MSRData(mySecureMagReader.device_getDeviceType(), card);
    handler.post(doUpdateStatus);

}
```

- Implement the button press methods

```
  btnGetFirmware.setOnClickListener(new Button.OnClickListener(){
      public void onClick(View v) {
          info = "Getting Firmware\n";
          detail = "";
          handler.post(doUpdateStatus);
          StringBuilder sb = new StringBuilder();
          int ret = mSecureMagReader.device_getFirmwareVersion(sb);
          if (ret == ErrorCode.SUCCESS) {
              info += "Firmware Version: " + sb.toString();
              detail = "";
              handler.post(doUpdateStatus);
          }
          else {
              info += "GetFirmwareVersion: Failed\n";
              info += "Status: "+ mySecureMagReader.device_getResponseCodeString(ret)+"";
              detail = "";
              handler.post(doUpdateStatus);
              }
      }
  });

  btnStartSwipe.setOnClickListener(new Button.OnClickListener(){
      public void onClick(View v) {
          detail = "";
          info = "Starting Swipe/Tap Transaction\n";
          handler.post(doUpdateStatus);
          mySecureMagReader.msr_startMSRSwipe();
      }
  });

  btnCancelSwipe.setOnClickListener(new Button.OnClickListener(){
      public void onClick(View v) {
          detail = "";
          info = "Cancelling Swipe/Tap Transaction\n";
          handler.post(doUpdateStatus);
          mySecureMagReader.msr_cancelMSRSwipe();
      }
  });
```

6.8.6 Complete code listing

```
package com.example.securemag_sdk_tutorial;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.util.Set;

import android.app.Activity;
import android.app.Dialog;
import android.os.Bundle;
import android.os.Handler;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.TextView;

import com.idtechproducts.device.*;
import com.idtechproducts.device.ReaderInfo.DEVICE_TYPE;
import com.idtechproducts.device.ReaderInfo.
      CAPTURE_ENCODE_TYPE;
import com.idtechproducts.device.ReaderInfo.
      CAPTURE_ENCRYPT_TYPE;
import com.idtechproducts.device.ReaderInfo.
      EVENT_MSR_Types;

public class MainActivity extends Activity implements  OnReceiverListener{

    // declaring the instance of the SecureMagReader;
    private IDT_SecureMag mySecureMagReader = null;
    private TextView connectStatusTextView;
    private TextView textLog;
    private TextView lcdLog;
    private Button btnGetFirmware;
    private Button btnStartSwipe;
    private Button btnCancelSwipe;
    private Handler handler = new Handler();
    private boolean isReaderConnected = false;
    private String info = "";
    private String detail = "";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        handler = new Handler();
        btnGetFirmware = (Button)findViewById(R.id.btn_getFirmware);
        btnStartSwipe = (Button)findViewById(R.id.btn_startSwipe);
        btnCancelSwipe = (Button)findViewById(R.id.btn_cancelSwipe);
        textLog = (TextView)findViewById(R.id.textLog);
        lcdLog = (TextView)findViewById(R.id.lcdLog);
        connectStatusTextView = (TextView)findViewById(R.id.status_text);
        if(mySecureMagReader!=null){
            mySecureMagReader.unregisterListen();
            mySecureMagReader.release();
            mySecureMagReader = null;
        }
        mySecureMagReader = new IDT_SecureMag(this,this);
        mySecureMagReader.device_setDeviceType(DEVICE_TYPE.DEVICE_SECUREMAG);
        mySecureMagReader.registerListen();
        loadXMLfile();


        btnGetFirmware.setOnClickListener(new Button.OnClickListener(){
            public void onClick(View v) {
                info = "Getting Firmware\n";
                detail = "";
                handler.post(doUpdateStatus);
                StringBuilder sb = new StringBuilder();
                int ret = mySecureMagReader.device_getFirmwareVersion(sb);
                if (ret == ErrorCode.SUCCESS) {
                    info += "Firmware Version: " + sb.toString();
                    detail = "";
                    handler.post(doUpdateStatus);
                }
                else {
                    info += "GetFirmwareVersion: Failed\n";
                    info += "Status: "+ mySecureMagReader.device_getResponseCodeString(ret)+"";
                    detail = "";
                    handler.post(doUpdateStatus);
                    }
            }
        });
```

```
        btnStartSwipe.setOnClickListener(new Button.OnClickListener(){
            public void onClick(View v) {
                detail = "";
                info = "Starting Swipe/Tap Transaction\n";
                handler.post(doUpdateStatus);
                mySecureMagReader.msr_startMSRSwipe();
            }
        });

        btnCancelSwipe.setOnClickListener(new Button.OnClickListener(){
            public void onClick(View v) {
                detail = "";
                info = "Cancelling Swipe/Tap Transaction\n";
                handler.post(doUpdateStatus);
                mySecureMagReader.msr_cancelMSRSwipe();
            }
        });

    }

    @Override
    public void ICCNotifyInfo(byte[] arg0, String arg1) {
        // TODO Auto-generated method stub

    }

    @Override
    public void LoadXMLConfigFailureInfo(int arg0, String arg1) {
        // TODO Auto-generated method stub

    }

    @Override
    public void autoConfigCompleted(StructConfigParameters arg0) {
        // TODO Auto-generated method stub

    }

    @Override
    public void autoConfigProgress(int arg0) {
        // TODO Auto-generated method stub

    }

    private Runnable doUpdateLabel = new Runnable()
    {
        public void run()
        {
            if(!isReaderConnected){
                connectStatusTextView.setText("SecureMag DISCONNECTED");
            }
            else{
                connectStatusTextView.setText("SecureMag CONNECTED");
            }
        }
    };
    @Override
    public void deviceConnected() {
        isReaderConnected = true;
        handler.post(doUpdateLabel);
    }

    @Override
    public void deviceDisconnected() {
        isReaderConnected = false;
        handler.post(doUpdateLabel);
    }

    private void printTags(IDTEMVData emvData)
    {

    }

    @Override
    public void emvTransactionData(IDTEMVData emvData) {

    }

    public void lcdDisplay(int mode, String[] lines, int timeout) {

    }


    @Override
    public void msgAudioVolumeAjustFailed() {
        // TODO Auto-generated method stub
```

```
}

@Override
public void msgRKICompleted(String arg0) {
    // TODO Auto-generated method stub

}

@Override
public void msgToConnectDevice() {
    // TODO Auto-generated method stub

}

private Runnable doUpdateStatus = new Runnable()
{
    public void run()
    {
        lcdLog.setText(info);
        textLog.setText(detail);
    }
};
@Override
public void swipeMSRData(IDTMSRData card) {
    if (card.cardData[0] != (byte)0x01 && card.track1Length == 0 && card.track2Length == 0 && card.
  track3Length == 0)
        info = "Swipe/Tap data didn't read correctly";
    else
        info = "Swipe/Tap Read Successfully";
    detail = Common.parse_MSRData(mySecureMagReader.device_getDeviceType(), card);
    handler.post(doUpdateStatus);

}

@Override
public void timeout(int arg0) {
    // TODO Auto-generated method stub

}

private String getXMLFileFromRaw(String fileName ,int res){
    //the target filename in the application path
    String fileNameWithPath = null;
    fileNameWithPath = fileName;
    String newFilename = fileName;

    try {
        InputStream in = getResources().openRawResource(res);
        int length = in.available();
        byte [] buffer = new byte[length];
        in.read(buffer);
        in.close();
        deleteFile(fileNameWithPath);
        FileOutputStream fout = openFileOutput(fileNameWithPath, MODE_PRIVATE);
        fout.write(buffer);
        fout.close();

        // to refer to the application path
        File fileDir = this.getFilesDir();
        fileNameWithPath = fileDir.getParent() + java.io.File.separator + fileDir.getName();
        fileNameWithPath += java.io.File.separator+newFilename;

    } catch(Exception e){
        e.printStackTrace();
        fileNameWithPath = null;
    }
    return fileNameWithPath;
}

private String getConfigurationFileFromRaw( ){
    return getXMLFileFromRaw("idt_unimagcfg_default.xml",R.raw.idt_unimagcfg_default);
    }
private boolean isFileExist(String path) {
    if(path==null)
        return false;
    File file = new File(path);
    if (!file.exists()) {
      return false ;
    }
    return true;
}
private void loadXMLfile(){

    //load the XML configuration file
    String fileNameWithPath = getConfigurationFileFromRaw();
    if(!isFileExist(fileNameWithPath)) {
```

```
            fileNameWithPath = null;
        }
        // Network operation is prohibited in the UI Thread if target API is 11 or above.
        // If target API is 11 or above, please use AsyncTask to avoid errors.
        mySecureMagReader.config_setXMLFileNameWithPath(fileNameWithPath);
        Log.d("Demo Info >>>>>","loadingConfigurationXMLFile begin.");
        mySecureMagReader.config_loadingConfigurationXMLFile(true);
    }




}
```

Chapter 7

SecureMag Error Code Reference

```
0000    OK
0001    Incorrect Header Tag
0002    Unknown Command
0003    Unknown Sub-Command
0004    CRC Error in Frame
0005    Incorrect Parameter
0006    Parameter Not Supported
0007    Mal-formatted Data
0008    Timeout
000A    Failed / NACK
000B    Command not Allowed
000C    Sub-Command not Allowed
000D    Buffer Overflow (Data Length too large for reader buffer)
000E    User Interface Event
0011    Communication type not supported, VT-1, burst, etc.
0012    Secure interface is not functional or is in an intermediate state.
0013    Data field is not mod 8
0014    Pad 0x80 not found where expected
0015    Specified key type is invalid
0016    Could not retrieve key from the SAM (InitSecureComm)
0017    Hash code problem
0018    Could not store the key into the SAM (InstallKey)
0019    Frame is too large
001A    Unit powered up in authentication state but POS must resend the InitSecureComm command
001B    The EEPROM may not be initialized because SecCommInterface does not make sense
001C    Problem encoding APDU
0020    Unsupported Index (ILM) SAM Transceiver error communicating with the SAM (Key Mgr)
0021    Unexpected Sequence Counter in multiple frames for single bitmap (ILM) Length error in data
      returned from the SAM (Key Mgr)
0022    Improper bit map (ILM)
0023    Request Online Authorization
0024    ViVOCard3 raw data read successful
0025    Message index not available (ILM) ViVOcomm activate transaction card type (ViVOcomm)
0026    Version Information Mismatch (ILM)
0027    Not sending commands in correct index message index (ILM)
0028    Time out or next expected message not received (ILM)
0029    ILM languages not available for viewing (ILM)
002A    Other language not supported (ILM)
0050    Auto-Switch OK
0051    Auto-Switch failed
0060    Data not exist
0061    Data Full
0062    Write Flash Error
0063    Ok and Have Next Command
0090    Account DUKPT Key not exist
0091    Account DUKPT Key KSN exhausted
EE00    OK
EE01    Incorrect Header Tag
EE02    Unknown Command
EE03    Unknown Sub-Command
EE04    CRC Error in Frame
EE05    Incorrect Parameter
EE06    Parameter Not Supported
EE07    Mal-formatted Data
EE08    Timeout
EE0A    Failed / NACK
EE0B    Command not Allowed
EE0C    Sub-Command not Allowed
EE0D    Buffer Overflow (Data Length too large for reader buffer)
```

```
EE0E    User Interface Event
EE11    Communication type not supported, VT-1, burst, etc.
EE12    Secure interface is not functional or is in an intermediate state.
EE13    Data field is not mod 8
EE14    Pad 0x80 not found where expected
EE15    Specified key type is invalid
EE16    Could not retrieve key from the SAM (InitSecureComm)
EE17    Hash code problem
EE18    Could not store the key into the SAM (InstallKey)
EE19    Frame is too large
EE1A    Unit powered up in authentication state but POS must resend the InitSecureComm command
EE1B    The EEPROM may not be initialized because SecCommInterface does not make sense
EE1C    Problem encoding APDU
EE20    Unsupported Index (ILM) SAM Transceiver error problem communicating with the SAM (Key Mgr)
EE21    Unexpected Sequence Counter in multiple frames for single bitmap (ILM) Length error in data
       returned from the SAM (Key Mgr)
EE22    Improper bit map (ILM)
EE23    Request Online Authorization
EE24    ViVOCard3 raw data read successful
EE25    Message index not available (ILM) ViVOcomm activate transaction card type (ViVOcomm)
EE26    Version Information Mismatch (ILM)
EE27    Not sending commands in correct index message index (ILM)
EE28    Time out or next expected message not received (ILM)
EE29    ILM languages not available for viewing (ILM)
EE2A    Other language not supported (ILM)
EE50    Auto-Switch OK
EE51    Auto-Switch failed
EE60    Data not exist
EE61    Data Full
EE62    Write Flash Error
EE63    Ok and Have Next Command
EE90    Account DUKPT Key not exist
EE91    Account DUKPT Key KSN exhausted?problem communicating with the SAM (Key Mgr)
0021    Unexpected Sequence Counter in multiple frames for single bitmap (ILM) Length error in data
      returned from the SAM (Key Mgr)
0022    Improper bit map (ILM)
0023    Request Online Authorization
0024    ViVOCard3 raw data read successful
0025    Message index not available (ILM) ViVOcomm activate transaction card type (ViVOcomm)
0026    Version Information Mismatch (ILM)
0027    Not sending commands in correct index message index (ILM)
0028    Time out or next expected message not received (ILM)
0029    ILM languages not available for viewing (ILM)
002A    Other language not supported (ILM)
0050    Auto-Switch OK
0051    Auto-Switch failed
0060    Data not exist
0061    Data Full
0062    Write Flash Error
0063    Ok and Have Next Command
0090    Account DUKPT Key not exist
0091    Account DUKPT Key KSN exhausted
EE00    OK
EE01    Incorrect Header Tag
EE02    Unknown Command
EE03    Unknown Sub-Command
EE04    CRC Error in Frame
EE05    Incorrect Parameter
EE06    Parameter Not Supported
EE07    Mal-formatted Data
EE08    Timeout
EE0A    Failed / NACK
EE0B    Command not Allowed
EE0C    Sub-Command not Allowed
EE0D    Buffer Overflow (Data Length too large for reader buffer)
EE0E    User Interface Event
EE11    Communication type not supported, VT-1, burst, etc.
EE12    Secure interface is not functional or is in an intermediate state.
EE13    Data field is not mod 8
EE14    Pad 0x80 not found where expected
EE15    Specified key type is invalid
EE16    Could not retrieve key from the SAM (InitSecureComm)
EE17    Hash code problem
EE18    Could not store the key into the SAM (InstallKey)
EE19    Frame is too large
EE1A    Unit powered up in authentication state but POS must resend the InitSecureComm command
EE1B    The EEPROM may not be initialized because SecCommInterface does not make sense
EE1C    Problem encoding APDU
EE20    Unsupported Index (ILM) SAM Transceiver error communicating with the SAM (Key Mgr)
EE21    Unexpected Sequence Counter in multiple frames for single bitmap (ILM) Length error in data
       returned from the SAM (Key Mgr)
EE22    Improper bit map (ILM)
EE23    Request Online Authorization
EE24    ViVOCard3 raw data read successful
EE25    Message index not available (ILM) ViVOcomm activate transaction card type (ViVOcomm)
EE26    Version Information Mismatch (ILM)
EE27    Not sending commands in correct index message index (ILM)
```

```
EE28    Time out or next expected message not received (ILM)
EE29    ILM languages not available for viewing (ILM)
EE2A    Other language not supported (ILM)
EE50    Auto-Switch OK
EE51    Auto-Switch failed
EE60    Data not exist
EE61    Data Full
EE62    Write Flash Error
EE63    Ok and Have Next Command
EE90    Account DUKPT Key not exist
EE91    Account DUKPT Key KSN exhausted?problem communicating with the SAM (Key Mgr)
EE21    Unexpected Sequence Counter in multiple frames for single bitmap (ILM) Length error in data
      returned from the SAM (Key Mgr)
EE22    Improper bit map (ILM)
EE23    Request Online Authorization
EE24    ViVOCard3 raw data read successful
EE25    Message index not available (ILM) ViVOcomm activate transaction card type (ViVOcomm)
EE26    Version Information Mismatch (ILM)
EE27    Not sending commands in correct index message index (ILM)
EE28    Time out or next expected message not received (ILM)
EE29    ILM languages not available for viewing (ILM)
EE2A    Other language not supported (ILM)
EE50    Auto-Switch OK
EE51    Auto-Switch failed
EE60    Data not exist
EE61    Data Full
EE62    Write Flash Error
EE63    Ok and Have Next Command
EE90    Account DUKPT Key not exist
EE91    Account DUKPT Key KSN exhausted
```

Chapter 8

Enumeration Reference

IDTMSRData

```
typedef enum _CAPTURE_ENCODE_TYPE{
    CAPTURE_ENCODE_TYPE_ISOABA=0,
    CAPTURE_ENCODE_TYPE_AAMVA=1,
    CAPTURE_ENCODE_TYPE_Other=3,
    CAPTURE_ENCODE_TYPE_Raw=4,
    CAPTURE_ENCODE_TYPE_JIS_II=5,
    CAPTURE_ENCODE_TYPE_JIS_I=6,
    CAPTURE_ENCODE_TYPE_MANUAL_ENTRY=7
} CAPTURE_ENCODE_TYPE;
```

```
typedef enum{
    CAPTURE_ENCRYPT_TYPE_TDES=0,
    CAPTURE_ENCRYPT_TYPE_AES=1
} CAPTURE_ENCRYPT_TYPE;
```

IDTCommon

```
typedef enum{
    POWER_ON_OPTION_IFS_FLAG=1,
    POWER_ON_OPTION_EXPLICIT_PPS_FLAG=2,
    POWER_ON_OPTION_AUTO_PPS_FLAG=64,
    POWER_ON_OPTION_IFS_RESPONSE_CHECK_FLAG=128
}POWER_ON_OPTION;
```

```
typedef enum{
    LANGUAGE_TYPE_ENGLISH=1,
    LANGUAGE_TYPE_PORTUGUESE,
    LANGUAGE_TYPE_SPANISH,
    LANGUAGE_TYPE_FRENCH
}LANGUAGE_TYPE;
```

```
typedef enum{
    PIN_KEY_TDES_MKSK_extp=0x00,
    PIN_KEY_TDES_DUKPT_extp=0x01,
    PIN_KEY_TDES_MKSK_intl=0x10,
    PIN_KEY_TDES_DUKPT_intl=0x11,
}PIN_KEY_Types;
```

```
typedef enum{
    EVENT_PINPAD_UNKNOWN = 11,
    EVENT_PINPAD_ENCRYPTED_PIN,
    EVENT_PINPAD_NUMERIC,
    EVENT_PINPAD_AMOUNT,
    EVENT_PINPAD_ACCOUNT,
    EVENT_PINPAD_ENCRYPTED_DATA,
    EVENT_PINPAD_CANCEL,
    EVENT_PINPAD_TIMEOUT,
    EVENT_PINPAD_FUNCTION_KEY,
    EVENT_PINPAD_DATA_ERROR
}EVENT_PINPAD_Types;
```

```
typedef enum{
    IDT_DEVICE_BTPAY_IOS = 0,
    IDT_DEVICE_BTPAY_OSX_BT,
    IDT_DEVICE_BTPAY_OSX_USB,
    IDT_DEVICE_UNIPAY_IOS,
    IDT_DEVICE_UNIPAY_OSX_USB,
    IDT_DEVICE_VP3300_IOS,
    IDT_DEVICE_VP3300_OSX_USB,
    IDT_DEVICE_IMAG_IOS,
    IDT_DEVICE_VENDI_MOBILE
}IDT_DEVICE_Types;
```

```
typedef enum{
    EVENT_MSR_UNKNOWN = 31,
    EVENT_MSR_CARD_DATA,
    EVENT_MSR_CANCEL_KEY,
    EVENT_MSR_BACKSPACE_KEY,
    EVENT_MSR_ENTER_KEY,
    EVENT_MSR_DATA_ERROR,
    EVENT_MSR_ICC_START,
    EVENT_BTPAY_CARD_DATA,
    EVENT_VP3300_EMV_NO_ICC_MSR_DATA,
    EVENT_VP3300_EMV_FALLBACK_DATA
}EVENT_MSR_Types;
```

```
typedef enum{
    EVENT_ACTIVE_TRANSACTION = 51
}EVENT_CTLS_Types;
```

```
typedef enum {
    RETURN_CODE_DO_SUCCESS = 0,
    RETURN_CODE_ERR_DISCONNECT,
    RETURN_CODE_ERR_CMD_RESPONSE,
    RETURN_CODE_ERR_TIMEDOUT,
    RETURN_CODE_ERR_INVALID_PARAMETER,
    RETURN_CODE_SDK_BUSY_MSR,
    RETURN_CODE_SDK_BUSY_PINPAD,
    RETURN_CODE_SDK_BUSY_CTLS,
    RETURN_CODE_ERR_OTHER,
    RETURN_CODE_FAILED,
    RETURN_CODE_NOT_ATTACHED,
    RETURN_CODE_MONO_AUDIO,
    RETURN_CODE_CONNECTED,
    RETURN_CODE_LOW_VOLUME,
    RETURN_CODE_CANCELED,


    RETURN_CODE_EMV_AUTHORIZATION_ACCEPTED = 0x0E00,
    RETURN_CODE_EMV_AUTHORIZATION_UNABLE_TO_GO_ONLINE = 0x0E01,
    RETURN_CODE_EMV_AUTHORIZATION_TECHNICAL_ISSUE = 0x0E02,
    RETURN_CODE_EMV_AUTHORIZATION_DECLINED = 0x0E03,
    RETURN_CODE_EMV_AUTHORIZATION_ISSUER_REFERRAL = 0x0E04,
```

```
    RETURN_CODE_EMV_APPROVED = 0x0F00,ction
    RETURN_CODE_EMV_DECLINED = 0x0F01,
    RETURN_CODE_EMV_GO_ONLINE = 0x0F02,
    RETURN_CODE_EMV_FAILED = 0x0F03,
    RETURN_CODE_EMV_SYSTEM_ERROR = 0x0F05,
    RETURN_CODE_EMV_NOT_ACCEPTED = 0x0F07,
    RETURN_CODE_EMV_FALLBACK = 0x0F0A,
    RETURN_CODE_EMV_CANCEL = 0x0F0C,
    RETURN_CODE_EMV_TIMEOUT = 0x0F0D,
    RETURN_CODE_EMV_OTHER_ERROR = 0x0F0F,
    RETURN_CODE_EMV_OFFLINE_APPROVED = 0x0F10,
    RETURN_CODE_EMV_OFFLINE_DECLINED = 0x0F11,



    RETURN_CODE_EMV_NEW_SELECTION = 0x0F21,
    RETURN_CODE_EMV_NO_AVAILABLE_APPS = 0x0F22,
    RETURN_CODE_EMV_NO_TERMINAL_FILE = 0x0F23,
    RETURN_CODE_EMV_NO_CAPK_FILE = 0x0F24,
    RETURN_CODE_EMV_NO_CRL_ENTRY = 0x0F25,
    RETURN_CODE_BLOCKING_DISABLED = 0x0FFE,
    RETURN_CODE_COMMAND_UNAVAILABLE = 0x0FFF


} RETURN_CODE;




typedef enum{
    EMV_RESULT_CODE_V2_APPROVED_OFFLINE = 0x0000,
    EMV_RESULT_CODE_V2_DECLINED_OFFLINE = 0x0001,
    EMV_RESULT_CODE_V2_APPROVED = 0x0002,
    EMV_RESULT_CODE_V2_DECLINED = 0x0003,
    EMV_RESULT_CODE_V2_GO_ONLINE = 0x0004,
    EMV_RESULT_CODE_V2_CALL_YOUR_BANK = 0x0005,
    EMV_RESULT_CODE_V2_NOT_ACCEPTED = 0x0006,
    EMV_RESULT_CODE_V2_USE_MAGSTRIPE = 0x0007,
    EMV_RESULT_CODE_V2_TIME_OUT = 0x0008,
    EMV_RESULT_CODE_V2_START_TRANS_SUCCESS = 0x0010,
    EMV_RESULT_CODE_V2_MSR_SUCCESS = 0x0011,
    EMV_RESULT_CODE_V2_FILE_ARG_INVALID = 0x1001,
    EMV_RESULT_CODE_V2_FILE_OPEN_FAILED = 0x1002,
    EMV_RESULT_CODE_V2_FILE_OPERATION_FAILED = 0X1003,
    EMV_RESULT_CODE_V2_MEMORY_NOT_ENOUGH = 0x2001,
    EMV_RESULT_CODE_V2_SMARTCARD_FAIL = 0x3001,
    EMV_RESULT_CODE_V2_SMARTCARD_INIT_FAILED = 0x3003,
    EMV_RESULT_CODE_V2_FALLBACK_SITUATION = 0x3004,
    EMV_RESULT_CODE_V2_SMARTCARD_ABSENT = 0x3005,
    EMV_RESULT_CODE_V2_SMARTCARD_TIMEOUT = 0x3006,
    EMV_RESULT_CODE_V2_MSR_CARD_ERROR = 0x3007,
    EMV_RESULT_CODE_V2_PARSING_TAGS_FAILED= 0X5001,
    EMV_RESULT_CODE_V2_CARD_DATA_ELEMENT_DUPLICATE = 0X5002,
    EMV_RESULT_CODE_V2_DATA_FORMAT_INCORRECT = 0X5003,
    EMV_RESULT_CODE_V2_APP_NO_TERM = 0X5004,
    EMV_RESULT_CODE_V2_APP_NO_MATCHING = 0X5005,
    EMV_RESULT_CODE_V2_AMANDATORY_OBJECT_MISSING = 0X5006,
    EMV_RESULT_CODE_V2_APP_SELECTION_RETRY = 0X5007,
    EMV_RESULT_CODE_V2_AMOUNT_ERROR_GET = 0X5008,
    EMV_RESULT_CODE_V2_CARD_REJECTED = 0X5009,
    EMV_RESULT_CODE_V2_AIP_NOT_RECEIVED = 0X5010,
    EMV_RESULT_CODE_V2_AFL_NOT_RECEIVEDE = 0X5011,
    EMV_RESULT_CODE_V2_AFL_LEN_OUT_OF_RANGE = 0X5012,
    EMV_RESULT_CODE_V2_SFI_OUT_OF_RANGE = 0X5013,
    EMV_RESULT_CODE_V2_AFL_INCORRECT = 0X5014,
    EMV_RESULT_CODE_V2_EXP_DATE_INCORRECT = 0X5015,
    EMV_RESULT_CODE_V2_EFF_DATE_INCORRECT = 0X5016,
    EMV_RESULT_CODE_V2_ISS_COD_TBL_OUT_OF_RANGE = 0X5017,
    EMV_RESULT_CODE_V2_CRYPTOGRAM_TYPE_INCORRECT = 0X5018,
    EMV_RESULT_CODE_V2_PSE_BY_CARD_NOT_SUPPORTED = 0X5019,
    EMV_RESULT_CODE_V2_USER_LANGUAGE_SELECTED = 0X5020,
    EMV_RESULT_CODE_V2_SERVICE_NOT_ALLOWED = 0X5021,
    EMV_RESULT_CODE_V2_NO_TAG_FOUND = 0X5022,
    EMV_RESULT_CODE_V2_CARD_BLOCKED = 0X5023,
    EMV_RESULT_CODE_V2_LEN_INCORRECT = 0X5024,
    EMV_RESULT_CODE_V2_CARD_COM_ERROR = 0X5025,
    EMV_RESULT_CODE_V2_TSC_NOT_INCREASED = 0X5026,
    EMV_RESULT_CODE_V2_HASH_INCORRECT = 0X5027,
    EMV_RESULT_CODE_V2_ARC_NOT_PRESENCED = 0X5028,
    EMV_RESULT_CODE_V2_ARC_INVALID = 0X5029,
    EMV_RESULT_CODE_V2_COMM_NO_ONLINE = 0X5030,
    EMV_RESULT_CODE_V2_TRAN_TYPE_INCORRECT = 0X5031,
    EMV_RESULT_CODE_V2_APP_NO_SUPPORT = 0X5032,
    EMV_RESULT_CODE_V2_APP_NOT_SELECT = 0X5033,
    EMV_RESULT_CODE_V2_LANG_NOT_SELECT = 0X5034,
    EMV_RESULT_CODE_V2_TERM_DATA_NOT_PRESENCED = 0X5035,
```

```
    EMV_RESULT_CODE_V2_CVM_TYPE_UNKNOWN = 0X6001,
    EMV_RESULT_CODE_V2_CVM_AIP_NOT_SUPPORTED = 0X6002,
    EMV_RESULT_CODE_V2_CVM_TAG_8E_MISSING = 0X6003,
    EMV_RESULT_CODE_V2_CVM_TAG_8E_FORMAT_ERROR = 0X6004,
    EMV_RESULT_CODE_V2_CVM_CODE_IS_NOT_SUPPORTED = 0X6005,
    EMV_RESULT_CODE_V2_CVM_COND_CODE_IS_NOT_SUPPORTED = 0X6006,
    EMV_RESULT_CODE_V2_CVM_NO_MORE = 0X6007,
    EMV_RESULT_CODE_V2_PIN_BYPASSED_BEFORE = 0X6008
} EMV_RESULT_CODE_V2_Types;
```

```
typedef enum{
    EMV_AUTHORIZATION_RESULT_ACCEPTED = 0X00,
    EMV_AUTHORIZATION_RESULT_UNABLE_TO_GO_ONLINE = 0X01,
    EMV_AUTHORIZATION_RESULT_TECHNICAL_ISSUE = 0X02,
    EMV_AUTHORIZATION_RESULT_DECLINED = 0X03,
    EMV_AUTHORIZATION_RESULT_ISSUER_REFERAL = 0X04
} EMV_AUTHORIZATION_RESULT;
```

Chapter 9

EMV Tag Reference

Tag	Description
42	Issuer Identification Number (IIN)
4F	Application Identifier (ADF Name)
50	Application Label
52	Command to perform
56	Track 1 Data
57	Track 2 Equivalent Data
5A	Application Primary Account Number (PAN)
5D	Deleted (see 9D)
5F20	Cardholder Name
5F24	Application Expiration Date
5F28	Issuer Country Code
5F2A	Transaction Currency Code (Default: 08 40)
5F2D	Language Preference
5F30	Service Code
5F34	Application Primary Account Number (PAN) Sequence Number (PSN)
5F36	Transaction Currency Exponent
5F3C	Transaction Reference Currency Code
5F3D	Transaction Reference Currency Exponent
5F50	Issuer URL
5F53	International Bank Account Number (IBAN)
5F54	Bank Identifier Code (BIC)
5F55	Issuer Country Code (alpha2 format)
5F56	Issuer Country Code (alpha3 format)
5F57	Account Type Selection
6F	File Control Information (FCI) Template
61	Application Template
62	File Control Parameters (FCP) Template
70	READ RECORD Response Message Template
71	Issuer Script Template 1
72	Issuer Script Template 2
73	Directory Discretionary Template
77	Response Message Template Format 2
80	Response Message Template Format 1
81	Amount, Authorised (Binary)
82	Application Interchange Profile (AIP)

Tag	Description
83	Command Template
84	Dedicated File (DF) Name
86	Issuer Script Command
87	Application Priority Indicator
88	Short File Identifier (SFI)
89	Authorisation Code
8A	Authorization Response Code
8A	Authorisation Response Code (ARC)
8C	Card Risk Management Data Object List 1 (CDOL1)
8D	Card Risk Management Data Object List 2 (CDOL2)
8E	Cardholder Verification Method (CVM) List
8F	Certification Authority Public Key Index (PKI)
90	Issuer Public Key Certificate
91	Issuer Authentication Data
92	Issuer Public Key Remainder
93	Signed Application Data
94	Application File Locator (AFL)
95	Terminal Verification Results (TVR)
97	Transaction Certificate Data Object List (TDOL)
98	Transaction Certificate (TC) Hash Value
99	Transaction Personal Identification Number (PIN) Data
99	Transaction Personal Identification Number (PIN) Data
98	Transaction Certificate (TC) Hash Value
9A	Transaction Date (YYMMDD)
9A	Transaction Date
9B	Transaction Status Information
9B	Transaction Status Information
9C	Transaction Type
9C	Transaction Type
9D	Directory Definition File (DDF) Name
9F01	Acquirer Identifier
9F02	Amount, Authorized (Numeric)
9F03	Amount, Other (Numeric)
9F04	Amount, Other (Binary)
9F05	Application Discretionary Data
9F06	Application Identifier (AID) terminal
9F07	Application Usage Control (AUC)
9F08	Application Version Number
9F09	Application Version Number (Default: 00 02)
9F0B	Cardholder Name Extended
9F0D	Issuer Action Code - Default
9F0E	Issuer Action Code - Denial
9F0F	Issuer Action Code - Online
9F10	Issuer Application Data (IAD)
9F11	Issuer Code Table Index
9F12	Application Preferred Name
9F13	Last Online Application Transaction Counter (ATC) Register
9F14	Lower Consecutive Offline Limit
9F15	Merchant Category Code

Tag	Description
9F16	Merchant Identifier
9F17	Personal Identification Number (PIN) Try Counter
9F18	Issuer Script Identifier
9F19	Deleted (see 9F49)
9F1A	Terminal Country Code
9F1B	Terminal Floor Limit
9F1C	Terminal Identification
9F1D	Terminal Risk Management Data
9F1E	Interface Device (IFD) Serial Number
9F1F	Track 1 Discretionary Data
9F20	Track 2 Discretionary Data
9F21	Transaction Time (HHMMSS)
9F22	Certification Authority Public Key Index
9F23	Upper Consecutive Offline Limit
9F26	Application Cryptogram (AC)
9F27	Cryptogram Information Data (CID)
9F29	Extended Selection
9F2A	Kernel Identifier
9F2D	Integrated Circuit Card (ICC) PIN Encipherment Public Key Certificate
9F2E	Integrated Circuit Card (ICC) PIN Encipherment Public Key Exponent
9F2F	Integrated Circuit Card (ICC) PIN Encipherment Public Key Remainder
9F32	Issuer Public Key Exponent
9F33	Terminal Capabilities (see below)
9F34	Cardholder Verification Method (CVM) Results
9F35	Terminal Type (see below)
9F36	Application Transaction Counter (ATC)
9F37	Unpredictable Number
9F38	Processing Options Data Object List (PDOL)
9F39	POS Entry Mode (Default: 07)
9F3A	Amount, Reference Currency
9F3B	Application Reference Currency
9F3C	Transaction Reference Currency Code
9F3D	Transaction Reference Currency Exponent
9F40	Additional Terminal Capabilities (see below)
9F41	Transaction Sequence Counter
9F42	Application Currency Code
9F43	Application Reference Currency Exponent
9F44	Application Currency Exponent
9F45	Data Authentication Code
9F46	Integrated Circuit Card (ICC) Public Key Certificate
9F47	Integrated Circuit Card (ICC) Public Key Exponent
9F48	Integrated Circuit Card (ICC) Public Key Remainder
9F49	Dynamic Data Authentication Data Object List (DDOL)
9F4A	Static Data Authentication Tag List (SDA)
9F4B	Signed Dynamic Application Data (SDAD)
9F4C	ICC Dynamic Number
9F4D	Log Entry
9F4E	Merchant Name and Location
9F4E	Merchant Name and Location

Tag	Description
9F4F	Log Format
9F50	Offline Accumulator Balance
9F51	Application Currency Code
9F52	Application Default Action (ADA)
9F53	Transaction Category Code
9F54	DS ODS Card
9F55	Geographic Indicator
9F56	Issuer Authentication Indicator
9F57	Issuer Country Code
9F58	Consecutive Transaction Counter Limit (CTCL)
9F59	Consecutive Transaction Counter Upper Limit (CTCUL)
9F5A	Application Program Identifier (Program ID)
9F5B	Issuer Script Results
9F5C	Magstripe Data Object List (MDOL)
9F5D	Available Offline Spending Amount (AOSA)
9F5D	Application Capabilities Information (ACI)
9F5E	Consecutive Transaction International Upper Limit (CTIUL)
9F5E	DS ID
9F5F	DS Slot Availability
9F60	CVC3 (Track1)
9F61	CVC3 (Track2)
9F62	PCVC3 (Track1)
9F64	NATC (Track1)
9F65	PCVC3 (Track2)
9F66	PUNATC (Track2)
9F67	NATC (Track2)
9F68	Card Additional Processes
9F69	UDOL
9F6A	Unpredictable Number (Numeric)
9F6B	Track 2 Data
9F6C	Card Transaction Qualifiers (CTQ)
9F6D	Mag-stripe Application Version Number (Reader)
9F6E	Third Party Data
9F6E	Terminal Transaction Capabilities
9F6F	DS Slot Management Control
9F70	Protected Data Envelope 1
9F71	Protected Data Envelope 2
9F72	Protected Data Envelope 3
9F73	Protected Data Envelope 4
9F74	Protected Data Envelope 5
9F75	Unprotected Data Envelope 1
9F76	Unprotected Data Envelope 2
9F77	Unprotected Data Envelope 3
9F78	Unprotected Data Envelope 4
9F79	Unprotected Data Envelope 5
9F7A	VLP Terminal Support Indicator
9F7B	VLP Terminal Transaction Limit
9F7C	Customer Exclusive Data (CED)
9F7D	DS Summary 1

Tag	Description
9F7F	DS Unpredictable Number
A5	File Control Information (FCI) Proprietary Template
BF0C	File Control Information (FCI) Issuer Discretionary Data
BF50	Visa Fleet - CDO
BF60	Integrated Data Storage Record Update Template
C3	Card issuer action code -decline
C4	Card issuer action code -default
C5	Card issuer action code online
C6	PIN Try Limit
C7	CDOL 1 Related Data Length
C8	Card risk management country code
C9	Card risk management currency code
CA	Lower cummulative offline transaction amount
CB	Upper cumulative offline transaction amount
CD	Card Issuer Action Code (PayPass) Default
CE	Card Issuer Action Code (PayPass) Online
CF	Card Issuer Action Code (PayPass) Decline
D1	Currency conversion table
D2	Integrated Data Storage Directory (IDSD)
D3	Additional check table
D5	Application Control
D6	Default ARPC response code
D7	Application Control (PayPass)
D8	AIP (PayPass)
D9	AFL (PayPass)
DA	Static CVC3-TRACK1
DB	Static CVC3-TRACK2
DC	IVCVC3-TRACK1
DD	IVCVC3-TRACK2
DF01	ApplePay VAS Protocol
DF02	ApplePay VAS Failure Report
DF10	Terminal Languages Supported
DF10	Multi Language (Default: “enfr
DF11	Enable Transaction Logging
DF13	Terminal Action Code - Default
DF14	Terminal Action Code - Denial
DF15	Terminal Action Code - Online
DF17	Threshold Value for Biased Random Selection
DF18	Target Percentage to be Used for Random Selection
DF19	Maximum Target Percentage to be used for Biased Random Selection
DF1F	Last 4 digits of Primary Account Number (PAN)
DF21	Issuer Script Results
DF22	Force Online (1-Enable, 0-Disable)
DF25	Default DDOL (1-Enable, 0-Disable)
DF26	Revocation List Support (Default: Enable - 1)
DF27	Exception File Support (Default: Disable - 0)
DF28	Default TDOL
DF29	Terminal Capabilities - CVM Required
DF2A	Threshold Value for Biased Random Selection (Interac)

Tag	Description
DF2B	Maximum Target Percentage for Biased Random Selection (Interac)
DF2C	Target Percentage for Random Selection (Interac)
DF30	Track Data Source
DF31	DD Card Track 1
DF32	DD Card Track 2
DF33	Interac Receipt Required
DF34	TTK Customer - Firmware Version
DF40	Message to be displayed by EMV Kernel on “PIN Try Limit Exceededcondition
DF41	Message to be displayed by EMV Kernel on “Last PIN Trycondition
DF42	Message to be displayed by EMV Kernel on “Please Try Againcondition
DF43	Message to be displayed by EMV Kernel on “Call Your Bankcondition
DF45	GMEDS Secret Keys
DF46	GMAD MIDs
DF47	ISIS Read Cmd Data
DF48	ISIS Write Data
DF49	ISIS Transaction Data
DF4A	TTK Customer - Current KSN of Data encryption Key
DF4B	TTK Customer - MSR all track data
DF4C	TTK Customer - Masked PAN
DF4D	TTK Customer - Additional POS Info
DF4E	Polling Options
DF4F	TTK Customer - Fallback Reason
DF50	Special Flow
DF51	Amex Terminal Capability
DF52	Transaction CVM
DF55	RID
DF56	Activate Trans for DESFireViVOComm Flows
DF57	Reader Primary Language
DF57	2nd usage: Remaining Candidates
DF58	Reader Secondary Language
DF5A	TLV Exclusion List
DF5B	Terminal Entry Capability
DF5C	RF Deactivate Period
DF5D	D-PAS Issuer Script Response status
DF5E	Transaction Timing Information
DF5F	Encrypted PAN for remote PIN Pad
DF60	Product ID
DF61	Processor ID
DF61	CVMRequiredLimit_JCBScheme
DF62	Main Firmware Build ID
DF63	CB Enhanced DDA Indicator (same block as DF03)
DF64	CB Wave 2 CVM Requirements (same block as DF04)
DF65	Build ID Num (Cxx)
DF65	CB Display Offline Funds Indicator (same block as DF05)
DF65	Serial heartbeat Required
DF66	SVN Number
DF66	CB Terminal Type (same block as 9F35)
DF66	Display Unsupported Card
DF68	Enable/Disable STOP command processing
DF69	ConfigureProprietaryTags

Tag	Description
DF6A	Enable/Disable Comm Error Recovery
DF6C	Cubic FTP Phase 2 Mode Options
DF6D	Cubic Mode 3 Match AID
DF6E	Cubic Fixed Fare Amounts
DF6F	Cubic Timestamp Data
DF70	Loyalty Program ID
DF70	Generic Name String
DF71	Value Added Tax 1
DF71	Generic Numeric
DF72	Value Added Tax 2
DF72	Generic Specification String
DF73	Merchant Category Code
DF73	Generic Implementation String
DF74	Discover Optional Features
DF75	Communications Error Message Delay
DF76	TVR from GenAC
DF77	ViVOpay MSR Custom Data Output Tag
DF78	MC Timing Performance Enable
DF79	Card Disable Mask
DF7A	Card Disable Interval
DF7B	Serial Port (UART) Inter-character Timeout Period
DF7C	Auto Switch Feature
DF7D	Track Formatting Feature
DF7F	Improved Collision Detection & Media Removal Feature
DF891B	Poll Mode
DF891C	Interac Retry Limit
DFDE04	MSR Encryption Option
DFEE0C	PPSE Terminate Flags
DFEE12	KID
DFEE15	Application Selection Indicator
DFEE16	DUKPT Key or MKSK Select for Online PIN Encrypted
DFEE17	ICC Terminal Entry Mode
DFEE18	MSR Terminal Entry Mode
DFEE19	Online DOL
DFEE1A	Output data element
DFEE1B	Authorization Request data elements
DFEE1E	Contact Terminal Configuration (see below)
DFEE1F	Issuer script device limit, Range: 0∼255 (Default: 128)
DFEE20	ICC Power on detect waiting time. (Unit: Sec) (Default: 60S)
DFEE21	ICC L1 waiting time. (Unit: Sec)(Default: 10 S)
DFEE22	Driver (Menu, Get PIN, Get MSR) Timeout. (Unit: Sec) (see below)
DFEE23	MSR Track Data
DFEE24	Force Acceptance (Default: 00)
DFEE25	ICC Response Code
DFEE26	Encryption Status Information
DFEE27	MSR Control
DFEF1A	TLV available
DFEF1A	Encrypted Sensitive Tags
DFEF1A	Auto Authenticate
DFEF20	MAC option in reponse data

Tag	Description
DFEF21	BIN
DFEF22	AID
DFEF23	HMAC
DFEF24	HMAC KSN
DFEF25	Output Data Format Select
DFEF26	MSR fallback
DFEF27	Online capability
DFEF28	Disable Encrypt ON
DFEF2C	Terminal AID List
DFEF2E	Terminal Transaction Log
DFEF2F	CUP configuration
DFEF30	White List
DFEF31	Black List
DFEF32	Auto-Switch
DFEF34	Antenna Detection Switch
DFEF35	Communications Watchdog Period
DFEF36	Media Control & Status Tracking
DFEF37	Interface Select
DFEF38	Timeout for Next Command
DFEF39	Network Indicate
DFEF3A	Reader Behavior Mode
DFEF3B	Autopoll Transaction Separation Interval
DFEF40	Ascii-code encryption Tag57 TLV
DFEF41	MAC Verification Data for SRED
DFEF42	MAC Verification KSN for SRED
DFEF43	Local TZ/DST information.
DFEF44	Combination Options
DFEF45	Removal Timeout
DFEF46	ACT Pass Response DOL
DFEF47	CDA Hash Input
DFEF48	Indicate - retrieve transaction result again due to Output RAM is Not enough.
DFEF49	Outcome Parameter Set
DFE↩ F4A	User Interface Request Data
DFEF4B	MSR Equivalent Data Option
DFEF4C	MSR Equivalent Data Track Lengths
DFEF4D	MSR Equivalent Data
DFEF4E	ACT MSD Response DOL
DFEF4F	ACT Decline Response DOL
DFEF50	Terminal Interchange Profile (JCB)
DFEF51	Bypass EMV Completion Output
DFEF52	Re-FallBack times
DFEF53	Dynamic Reader Limits
DFEF54	SmartTap AID Index
DFEF55	Kernel Specific Features
DFEF56	Retry Limit
DFEF57	PPSE Terminate Flags
DFEF59	Terminal Data Setting - Default Amount
DFEF5A	Terminal Data Setting - Tags to Return
DFE↩ F5B	Mask for Tag5A

Tag	Description
DFEF5C	Mask for Tag56
DFEF5D	Mask for Tag57
DFEF5E	Mask for Tag9F6B
DFEF5F	Mask for TagFFEE13
DFEF60	Mask for TagFFEE14
DFEF61	Error Code
DFEF62	Allow MSR Swipe data from ICC Card
DFEF63	Tags To Read Yet
DFEF64	Referral Timeout
DFEF6E	USB-KB Output Data Postfix
DFEF6F	Inter-character Delay for USB-KB Interface
DFEF70	PISCES dual interface interference prevention mechanism fine-tune parameters.
DFEF71	Waiting ICC insert time
DFEF72	Pre-poll card mechanism control in ACT cmd & config setting
DFEF73	Transaction Message Type
DFEF74	Reference amplitude value
DFEF75	Reference delta value
DFEF76	Transaction Interface Type to activate
DFEF77	Timeout for waiting next command
DFEF78	EMV contact L2 display messages option
DFEF79	PIN block format (when TDES)
DFEF7A	Enable Apple Pay Check
DFEF7B	Apple Pay Status
DFEF7C	Track Bit Encoding
DFEF7D	Re-power on times
DFEF7E	Fallback response code list
FF69	ViVOpay Proprietary Tag List
FF70	Serial Finite State Machine Version
FF71	Transaction Finite State Machine Version
FF72	System Information Suite
FF73	Serial Protocol Version
FF74	Serial Protocol Suite
FF75	L1 Paypass Version
FF76	L1 LCR Version
FF77	L2 Card App Version
FF78	L2 Card App Suite
FF79	GMEDs Data
FF79	User Experience Version
FF7A	User Experience Suite
FF7B	ViVOtech Proprietary Suite
FF7C	VIUDS Scheme IDs Supported
FF7D	VIUDS Scheme ID Selection Criteria
FFE0	Registered Application Provider Identifier (RID)
FFE1	Partial Selection Allowed
FFE2	Application Flow
FFE3	Selection Features - GR 1.2.10
FFE4	Group Number / Fallback Group
FFE5	Max AID Length
FFE6	AID Disabled

Tag	Description
FFE7	Interface Support
FFE8	Exclude from Processing
FFE9	Kernel ID Transaction Type Group List
FFEA	Default Kernel ID
FFEE01	ViVOpay TLV Group Tag
FFEE02	ViVOpay Pre-PPSE Special Flow Group Tag
FFEE03	ViVOpay Post-PPSE Special Flow Group Tag
FFEE04	M/Chip3 Intermediate Message Data
FFEE05	M/Chip3 Intermediate Message Marker
FFEE06	ApplePay VAS Container
FFEE07	Encrypted Sensitive Tags
FFEE08	Masked Tags
FFEE0A	BIN Range
FFEE0B	AID Range
FFEE0C	White List
FFEE10	ViVOpay MChip Group Tag
FFEE11	ViVOpay Discover Group Tag
FFEE12	KID
FFEE12	Cash Reader Risk Record
FFEE13	Track 1 Data
FFEE13	Cashback Reader Risk Record
FFEE14	Track 2 Data
FFEE14	DRL Record 1
FFEE15	DRL Record 2
FFEE16	DRL Record 3
FFEE17	DRL Record 4
FFEE18	Tags To Write Yet Before GenAC
FFEE19	Tags To Write Yet After GenAC
FFEE1A	Terminal App DET Data
FFEE1C	Unpredictable Number Range
FFEE1D	Sensitive Data Mask
FFE↩ E1E	Group 0 Initialize Flag
FFE↩ E1F	Error Code Table
FFEE20	Restart Deactivation Time
FFF0	Specific Features Switch
FFF1	Terminal Contactless Transaction Limit
FFF2	Terminal IFD
FFF3	Application Capability
FFF4	Visa Reader Risk Flags
FFF6	Torn Transaction Log Clean Interval (minutes)
FFF7	Burst Mode
FFF8	UI Scheme
FFF9	LCD Font Size
FFFA	LCD Delay Time
FFFB	Language Option for LCD
FFFC	Force MagStripe

9F33 Terminal Capabilities

```
Byte 1
```

```
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
1   x   x   x   x   x   x   x   Manual key entry
x   1   x   x   x   x   x   x   Magnetic stripe
x   x   1   x   x   x   x   x   IC with contacts
x   x   x   0   x   x   x   x   RFU
x   x   x   x   0   x   x   x   RFU
x   x   x   x   x   0   x   x   RFU
x   x   x   x   x   x   0   x   RFU
x   x   x   x   x   x   x   0   RFU

Byte 2
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
1   x   x   x   x   x   x   x   Plaintext PIN for IC verification
x   1   x   x   x   x   X   x   Enciphered PIN for online verification
x   x   1   x   x   x   X   x   Signature(paper)
x   x   x   1   x   x   X   x   Enciphered PIN for offline verification
x   x   x   x   1   x   X   x   No CVM Required
x   x   x   x   x   0   x   x   RFU
x   x   x   x   x   x   0   x   RFU
x   x   x   x   x   x   X   0   RFU

Byte 3
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
1   x   x   x   x   x   x   x   SDA
X   1   x   x   x   x   x   x   DDA
X   x   1   x   x   x   x   x   Card capture
x   x   x   0   x   x   x   x   RFU
x   x   x   x   1   x   x   X   CDA
x   x   x   x   x   0   x   x   RFU
x   x   x   x   x   x   0   x   RFU
x   x   x   x   X   X   x   0   RFU
```

9F40 Additional Terminal Capabilities

```
Byte 1
b1  b2  b3  b4  b5  b6  b7  b8  Meaning
1   x   x   x   x   x   x   x   Cash
x   1   x   x   x   x   x   x   Goods
x   x   1   x   x   x   x   x   Services
x   x   x   1   x   x   x   x   Cashback
x   x   x   x   1   x   x   x   Inquiry
x   x   x   x   x   1   x   x   Transfer
x   x   x   x   x   x   1   x   Payment
x   x   x   x   x   x   x   1   Administrative

Byte 2
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
1   x   x   x   x   x   x   x   Cash Deposit
x   0   x   x   x   x   x   x   RFU
x   x   0   x   x   x   x   x   RFU
x   x   x   0   x   x   x   x   RFU
x   x   x   x   0   x   x   x   RFU
x   x   x   x   x   0   x   x   RFU
x   x   x   x   x   x   0   x   RFU
x   x   x   x   x   x   x   0   RFU

Byte 3
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
1   x   x   x   x   x   x   x   Numeric keys
x   1   x   x   x   x   x   x   Alphabetic and special characters keys
x   x   1   x   x   x   x   x   Command keys
x   x   x   1   x   x   x   x   Function Keys
x   x   x   x   0   x   x   x   RFU
x   x   x   x   x   0   x   x   RFU
x   x   x   x   x   x   0   x   RFU
x   x   x   x   x   x   x   0   RFU

Byte 4
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
1   x   x   x   x   x   x   x   Print, attendant
x   1   x   x   x   x   x   x   Print, cardholder
x   x   1   x   x   x   x   x   Display, attendant
x   x   x   1   x   x   x   x   Display, cardholder
x   x   x   x   0   x   x   x   RFU
x   x   x   x   x   0   x   x   RFU
x   x   x   x   x   x   1   x   Code table 10
x   x   x   x   x   x   x   1   Code table 9

Byte 5
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
1   x   x   x   x   x   x   x   Code table 8
x   1   x   x   x   x   x   x   Code table 7
x   x   1   x   x   x   x   x   Code table 6
x   x   x   1   x   x   x   x   Code table 5
x   x   x   x   1   x   x   x   Code table 4
x   x   x   x   x   1   x   x   Code table 3
```

```
x   x   x   x   x   x   1   x   Code table 2
x   x   x   x   x   x   x   1   Code table 1
```

9F35 Terminal Type

Environment	Financial Institution	Merchant	Cardholder
Attended			
Online only	11	21	
Offline with online capability	12	22	
Offline only	13	23	
Unattended			
Online only	14	24	34
Offline with online capability	15	25	35
Offline only	16	26	36

DFEE1E Contact Terminal Configuration (Default: F0 DC 3C F0 C2 9E 94 00)

```
Byte 1
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
1   x   x   x   x   x   x   x   Key Pad support
x   1   x   x   x   x   x   x   LCD support
x   x   1   x   x   x   x   x   PIN Pad support
x   x   x   1   x   x   x   x   Print Support
x   x   x   x   0   x   x   x   RFU
x   x   x   x   x   0   x   x   RFU
x   x   x   x   x   x   0   x   RFU
x   x   x   x   x   x   X   0   RFU

Byte 2
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
1   x   x   x   x   x   x   x   PSE support
x   1   x   x   x   x   x   x   Cardholder confirmation
x   x   1   x   x   x   x   x   Preferred display order
x   x   x   1   x   x   x   x   Multi language
x   x   x   x   1   x   x   x   EMV language selection method
x   x   x   x   x   1   x   x   Default DDOL
x   x   x   x   x   x   0   x   RFU
x   x   x   x   x   x   x   0   RFU

Byte 3
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
0   x   x   x   x   x   x   x   RFU
(Revocation of Issuer Public Key Certificate (DF26))
x   1   x   x   x   x   x   x   Manual action when CA PK loading fails
x   x   1   x   x   x   x   x   CA PK verified with check sum
x   x   x   1   x   x   x   x   Bypass PIN Entry
x   x   x   x   1   x   x   x   Subsequent bypass PIN Entry
x   x   x   x   x   1   x   x   Get data for pin try counter
x   x   x   x   x   x   0   x   RFU
x   x   x   x   x   x   x   0   RFU

Byte 4
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
1   x   x   x   x   x   x   x   Amount before CVM processing
x   1   x   x   x   x   x   x   Floor limit checking
x   x   1   x   x   x   x   x   Random transaction selection
x   x   x   1   x   x   x   x   Velocity checking
x   x   x   x   0   x   x   x   RFU
(Transaction Log (DF11))
x   x   x   x   x   0   x   x   RFU
(Exception File (DF27))
x   x   x   x   x   x   0   x   RFU
x   x   x   x   x   x   x   0   RFU

Byte 5
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
1   x   x   x   x   x   x   X   Terminal action code support
x   1   x   x   x   x   x   x   Terminal action code can be change
x   x   1   x   x   x   x   x   Terminal action code can be deleted or disable
x   x   x   1   x   x   x   x   Default Action code processing before 1st GAC
x   x   x   x   1   x   x   x   Default Action code processing after 1st GAC
x   x   x   x   x   1   x   x   TAC/IAC default process when unable to go online (Skipped)
x   x   x   x   x   x   1   x   TAC/IAC default process when unable to go online (Normal)
x   x   x   x   x   x   x   0   RFU
```

```
Byte 6
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
1   x   x   x   x   x   x   x   Forced Online support
x   1   x   x   x   x   x   x   Forced acceptance support
x   x   1   x   x   x   x   x   Advices support
x   x   x   1   x   x   x   x   Issuer referrals support
X   x   x   x   1   x   x   x   Batch data capture
x   x   x   x   x   1   x   x   Online data capture
X   x   x   x   x   x   1   x   Default TDOL
X   x   x   x   x   x   x   0   RFU


Byte 7
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
1   x   x   x   x   x   x   x   amount and pin entered on the same keypad
x   1   x   x   x   x   x   x   ICC/Magstripe reader combined
x   x   1   x   x   x   x   x   Magstripe read first
x   x   x   1   x   x   x   x   Support account type selection
x   x   x   x   1   x   x   x   On fly script processing
x   x   x   x   x   1   x   x   Internal date management
x   x   x   x   x   x   1   x   Reversal Mode
(1)Unable go online
(2) ARC Error
0: (3) Online Approved but reader not approved.
1: (3) Online Approved but card response AAC.
x   x   x   x   x   x   x   0   RFU

Byte 8
b8  b7  b6  b5  b4  b3  b2  b1  Meaning
x   x   x   x   x   x   x   x   RFU
```

DFEE22 Driver (Menu, Get PIN, Get MSR) Timeout. (Unit: Sec)

```
Byte1: Timeout for Menu. (Default: 30 S)
Byte2: Timeout for Get PIN. (Default: 60 S)
Byte3: Timeout for Get MSR. (Default: 60 S)
```

Chapter 10

LCD Foreign Language Mapping Table

ID	Message ID	English	French	Spanish	Chinese
0	MSG_NULL	-	-	-	-
1	MSG_AMOUNT	AMOUNT	MONTANT	CANTIDAD	金
2	MSG_AMOUNT_OK	AMOUNT OK?	MONTANT OK	MONTO CORRECTO?	确定金
3	MSG_APPROVED	APPROVED	APPROUVE	APROVADO	通
4	MSG_CALL_YOUR_BANK	CALL YOUR BANK	APPE VOTRE BANQE	LLAME A SU BANCO	系您的
5	MSG_CANCEL_OR_ENTER	CANCEL OR ENTER	ANNULE OU ENTRER	CANCEL O ENTRAR	取消或确
6	MSG_CARD_ERROR	CARD ERROR	ERREUR CARTE	ERROR DE TARJETA	卡
7	MSG_DECLINED	DECLINED	REFUSE	DECLINADO	卡被
8	MSG_ENTER_AMOUNT	ENTER AMOUNT	ENTRER MONTANT	INGRESE MONTO	入金
9	MSG_ENTER_PIN	ENTER PIN:	ENTRER PIN:	ENTRAR NPI:	入密
10	MSG_INCORRECT_PIN	INCORRECT PIN	NIP INCORRECT	NPI INCORRECTO	密
11	MSG_ICC_MSR1	SWIPE OR INSERT	PASSER OU INSERT	MOVER O INSERT	刷卡或插卡
12	MSG_ICC_MSR2	CARD	CARTE	TARJETA	
13	MSG_INSERT_CARD	INSERT CARD	INSERT CARTE	INSERTAR TARJETA	插
14	MSG_USE_CHIP_READER	USE CHIP READER UTI	LECTEUR CHIP	USO CHIP LECTOR	使用芯片
15	MSG_NOT_ACCEPTED	NOT ACCEPTED	PAS ACCEPTE	DENEGADO	法接受
16	MSG_PIN_OK	GET PIN OK			密正确
17	MSG_PLEASE_WAIT	PLEASE WAIT...	ATTENDRE...	POR FAVOR ESPERE	等候中
18	MSG_PROCESSING_ERROR	PROCESSING ERROR	ERREUR DE TRAITE	ERROR PROCESANDO	理
19	MSG_USE_MAGSTRIPE	USE MAGSTRIPE	USAGE MAGSTRIPE	USO DE MAGSTRIPE	使用磁
20	MSG_TRY_AGAIN	TRY AGAIN	REESSAYER	VUELV INTENTARLO	重
21	MSG_ONLINE	GO ONLINE	GO LIGNE	GO LINEA	在

ID	Message ID	English	French	Spanish	Chinese
22	MSG_TRANSAC↩ TION_ERROR	TRANSACTION ERR	ERREUR DE TR↩ ANS	ERROR DE TRA↩ NSAC	交易
23	MSG_TERMINA↩ TE	TERMINATE	RESILIER	TERMINAR	止
24	MSG_ADVICE	ADVICE	CONSEILS	CONSEJOS	建
25	MSG_TIMEOUT	TIME OUT	TIMEOUT	TIEMPO DE ESP↩ ERA	超
26	MSG_PROCES↩ SING	PROCESSING...	PROCESSUS...	PROCESANDO...	理中。。
27	MSG_PIN_TRY_↩ EX	PIN TRY LIMIT EX	PIN TRY DEPAS↩ SE	TRY PIN SUPER↩ ADA	密次多
28	MSG_ISSUER_↩ AUTH_FAIL	ISSUER AUTH F↩ AIL	EMETTEUR FAIL	EMISOR FALLA	与卡机构
29	MSG_CONTINU↩ E_PROCESS	CONTINUE PRO↩ CESS	CONTINUER LA	CONTINUAR PR↩ OCES	理
30	MSG_GET_PIN_↩ ERROR	GET PIN ERROR	GET PIN ERROR	OBTENER PIN E↩ RR	密
31	MSG_GET_PIN_↩ FAIL	GET PIN FAIL	GET PIN FAIL	OBTENER PIN F↩ ALL	取密
32	MSG_NOKEY_G↩ ET_PIN	NO KEY GET PIN	NO KEY GET PIN	NO CLAVE GET PIN	法入密
33	MSG_CANCELL↩ ED	CANCELLED	ANNULE	CANCELADO	取消
34	MSG_LAST_PIN↩ _TRY	LAST PIN TRY	-	-	最后一次入密

Chapter 11

Class Index

11.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 12

Class Documentation

12.1 com.idtechproducts.device.OnReceiverListener.EMV_RESULT_CODE_Types Enum Reference

Public Attributes

- **EMV_RESULT_CODE_OFFLINE_APPROVED**
- **EMV_RESULT_CODE_OFFLINE_DECLINED**
- **EMV_RESULT_CODE_APPROVED**
- **EMV_RESULT_CODE_DECLINED**
- **EMV_RESULT_CODE_GO_ONLINE**
- **EMV_RESULT_CODE_CALL_YOUR_BANK**
- **EMV_RESULT_CODE_NOT_ACCEPTED**
- **EMV_RESULT_CODE_USE_MAGSTRIPE**
- **EMV_RESULT_CODE_TIME_OUT**
- **EMV_RESULT_CODE_TRANSACTION_SUCCESS**
- **EMV_RESULT_CODE_TERMINATE**

The documentation for this enum was generated from the following file:

- Source_Android/OnReceiverListener.java

12.2 com.idtechproducts.device.IDT_SecureMag Class Reference

Public Member Functions

- IDT_SecureMag (OnReceiverListener callback, Context context)
- boolean device_setDeviceType (ReaderInfo.DEVICE_TYPE deviceType)
- void setIDT_Device (FirmwareUpdateTool fwTool)
- DEVICE_TYPE device_getDeviceType ()
- void registerListen ()
- void unregisterListen ()
- void release ()
- String config_getSDKVersion ()
- String config_getXMLVersionInfo ()
- String phone_getInfoManufacture ()
- String phone_getInfoModel ()
- void log_setVerboseLoggingEnable (boolean enable)

- void log_setSaveLogEnable (boolean enable)
- int log_deleteLogs ()
- boolean device_isConnected ()
- int device_startRKI ()
- int device_getFirmwareVersion (StringBuilder version)
- int config_getSerialNumber (StringBuilder serialNumber)
- String device_getResponseCodeString (int errorCode)
- int device_sendDataCommand (String cmd, boolean calcLRC, String data, ResDataStruct respData, int timeout)
- int device_sendDataCommand (String cmd, boolean calcLRC, String data, ResDataStruct respData)
- int device_reviewAllSetting (ResDataStruct respData)
- int device_enableTDES (ResDataStruct respData)
- int device_enableAES (ResDataStruct respData)
- int device_getKSN (ResDataStruct ksn)
- int device_enableErrorNotification (ResDataStruct respData, boolean enable)
- int device_enableExpDate (ResDataStruct respData, boolean enable)
- int device_setEnhancedMode (ResDataStruct respData, boolean enable)
- int device_switchToKeyboardMode (ResDataStruct respData)
- int device_switchToHidMode (ResDataStruct respData)
- int msr_defaultAllSetting ()
- int msr_cancelMSRSwipe ()
- int msr_startMSRSwipe ()

Static Public Member Functions

- static IDT_Device getSDKInstance ()
- static void useUSBIntentFilter ()
- static IDT_Device **getIDT_Device** ()

12.2.1 Constructor & Destructor Documentation

12.2.1.1 IDT_SecureMag()

```
com.idtechproducts.device.IDT_SecureMag.IDT_SecureMag (
            OnReceiverListener callback,
            Context context )
```

It is the constructor of the main class IDT_BTMag. When it is called, the SDK will create the Instance for IDT_BTMag device. The interface OnReceiverListner needs to be implemented in the application.

Parameters

callback	OnReceiverListener callback
context	Application context

12.2.2 Member Function Documentation

12.2.2.1 config_getSDKVersion()

```
String com.idtechproducts.device.IDT_SecureMag.config_getSDKVersion ( )
```

READER CONFIG API LIST Get the version of SDK.

Parameters

sdkVersion	for version string.

Returns

success or error code.

See also

ErrorCode

12.2.2.2 config_getSerialNumber()

```
int com.idtechproducts.device.IDT_SecureMag.config_getSerialNumber (
            StringBuilder serialNumber )
```

Get the serial number of device.

Parameters

serialNumber	returns Serial Number string.

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.3 config_getXMLVersionInfo()

```
String com.idtechproducts.device.IDT_SecureMag.config_getXMLVersionInfo ( )
```

Get XML configuration version.

Returns

the version info.

12.2.2.4 device_enableAES()

```
int com.idtechproducts.device.IDT_SecureMag.device_enableAES (
            ResDataStruct respData )
```

enable AES encryption.

Parameters

respData	response data from reader

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.5 device_enableErrorNotification()

```
int com.idtechproducts.device.IDT_SecureMag.device_enableErrorNotification (
            ResDataStruct respData,
            boolean enable )
```

To enable or disable MSR error notification.

Parameters

respData	response data from reader
enable	to enable or disable the notification

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.6 device_enableExpDate()

```
int com.idtechproducts.device.IDT_SecureMag.device_enableExpDate (
            ResDataStruct respData,
            boolean enable )
```

To enable or disable Expiration date in MSR data

Parameters

respData	response data from reader
enable	to enable or disable expiration date

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.7 device_enableTDES()

```
int com.idtechproducts.device.IDT_SecureMag.device_enableTDES (
            ResDataStruct respData )
```

enable TDES encryption.

Parameters

respData	response data from reader

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.8 device_getDeviceType()

```
DEVICE_TYPE com.idtechproducts.device.IDT_SecureMag.device_getDeviceType ( )
```

Gets type of device

12.2.2.9 device_getFirmwareVersion()

```
int com.idtechproducts.device.IDT_SecureMag.device_getFirmwareVersion (
            StringBuilder version )
```

start Auto Config to search the profile.

Parameters

strXMLFilename	Input the customized XML file as the templates to search the profile.

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode stop Auto Config.

Returns

null. DEVICE INFO API Get the firmware version of device.

Parameters

version	for version string.

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.10 device_getKSN()

```
int com.idtechproducts.device.IDT_SecureMag.device_getKSN (
            ResDataStruct ksn )
```

Get the Account DUKPT Key KSN of device.

Parameters

10-byte	KSN

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.11 device_getResponseCodeString()

```
String com.idtechproducts.device.IDT_SecureMag.device_getResponseCodeString (
            int errorCode )
```

Get Response Code String

Interpret a response code and return string description.

Parameters

errorCode	Error code, range 0x0000 - 0xFFFF, example 0x0300

Returns

Verbose error description

12.2.2.12 device_isConnected()

```
boolean com.idtechproducts.device.IDT_SecureMag.device_isConnected ( )
```

set XML Configuration File Name with the full path.

Parameters

xmlFilename,XML	Configuration File Name.

Returns

none Load XML Configuration File.

Parameters

xmlFilename,XML	Configuration File Name.

Returns

none connect the device with Profile.

Parameters

profile,the	profile is the one which is the result from Auto config.

Returns

true: success, false: fail. get the status if the device connected.
true: connected, false: disconnected

12.2.2.13 device_reviewAllSetting()

```
int com.idtechproducts.device.IDT_SecureMag.device_reviewAllSetting (
            ResDataStruct respData )
```

/** Review All Configuration Settings

it returns the current values for all the parameters that can be set using the Set Configuration command. Each parameter is returned as a TLV data object.

Parameters

respData	Returns TLV in ResDataStruct.resData. Status Code in ResDataStruct.statusCode.

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.14 device_sendDataCommand() [1/2]

```
int com.idtechproducts.device.IDT_SecureMag.device_sendDataCommand (
            String cmd,
            boolean calcLRC,
            String data,
            ResDataStruct respData,
            int timeout )
```

Send a NSData object to device

Sends a command represented by the provide NSData object to the device through the accessory protocol.

Parameters

cmd	NSData representation of command to execute
calcLRC	If TRUE, this will wrap command with start/length/lrc/sum/end: '{STX}{Len_Low}{Len_High} data {CheckLRC} {CheckSUM} {ETX}'
data	Command data (if applicable) for IDG, not used for NGA
response	Returns response ResDataStruct.respData
timeout	Command timeout in seconds

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.15 device_sendDataCommand() [2/2]

```
int com.idtechproducts.device.IDT_SecureMag.device_sendDataCommand (
            String cmd,
            boolean calcLRC,
            String data,
            ResDataStruct respData )
```

Send a NSData object to device

Sends a command represented by the provide NSData object to the device through the accessory protocol.

Parameters

cmd	NSData representation of command to execute

Parameters

calcLRC	If TRUE, this will wrap command with start/length/lrc/sum/end: '{STX}{Len_Low}{Len_High} data {CheckLRC} {CheckSUM} {ETX}'
data	Command data (if applicable) for IDG, not used for NGA
response	Returns response ResDataStruct.respData

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.16 device_setDeviceType()

```
boolean com.idtechproducts.device.IDT_SecureMag.device_setDeviceType (
            ReaderInfo.DEVICE_TYPE deviceType )
```

Defines connection Bluetooth

Parameters

deviceType	DEVICE_TYPE.DEVICE_BTMAG

12.2.2.17 device_setEnhancedMode()

```
int com.idtechproducts.device.IDT_SecureMag.device_setEnhancedMode (
            ResDataStruct respData,
            boolean enable )
```

To enable or disable Enhanced Encryption mode

Parameters

respData	response data from reader
enable	to enable or disable enhanced encryption mode

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.18 device_startRKI()

```
int com.idtechproducts.device.IDT_SecureMag.device_startRKI ( )
```

Start remote key injection.

Returns

success or error code.

See also

ErrorCode

12.2.2.19 device_switchToHidMode()

```
int com.idtechproducts.device.IDT_SecureMag.device_switchToHidMode (
            ResDataStruct respData )
```

switch to HID-USB Mode.

Parameters

respData	response data from reader

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.20 device_switchToKeyboardMode()

```
int com.idtechproducts.device.IDT_SecureMag.device_switchToKeyboardMode (
            ResDataStruct respData )
```

switch to Keyboard Mode.

Parameters

respData	response data from reader

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.21 getSDKInstance()

```
static IDT_Device com.idtechproducts.device.IDT_SecureMag.getSDKInstance ( )  [static]
```

Returns an instance of the currently initialized IDT_Device class.

Returns

IDT_Device instance

12.2.2.22 log_deleteLogs()

```
int com.idtechproducts.device.IDT_SecureMag.log_deleteLogs ( )
```

delete the log in the root path of SD card.

Returns

number of log files deleted

See also

log_setSaveLogEnable

12.2.2.23 log_setSaveLogEnable()

```
void com.idtechproducts.device.IDT_SecureMag.log_setSaveLogEnable (
            boolean enable )
```

Enable/Disable save the log into the root path of SD card.

Parameters

enableShowLog,true	enable save the log, the log includes the .txt text log and .wav signals file. false: disable save the log.

Returns

none

See also

deleteLogs

12.2.2.24 log_setVerboseLoggingEnable()

```
void com.idtechproducts.device.IDT_SecureMag.log_setVerboseLoggingEnable (
            boolean enable )
```

Enable/Disable Verbose Logging show in the logcat view window.

Parameters

enableShowLog,true	enable to show the log in the logcat view window. false: disable to show the log in the logcat view window.

Returns

none

12.2.2.25 msr_cancelMSRSwipe()

```
int com.idtechproducts.device.IDT_SecureMag.msr_cancelMSRSwipe ( )
```

Get single setting of Mask and Encryption by Function ID.

Parameters

funcID	function ID. 0x49:Leading PAN digits to display(0x00∼0x06). 0x4A:Last PAN digits to display(0x00∼0x04). 0x4B:Mask ASCII code track data(0x20∼0x7E). 0x4C:Encryption type ('1'-'2'). '1' 3DES, '2' AES. 0x50:Mask or display expiration date(0x30 or 0x31);0x31:don't mask expiration date. 0x7E:Security Level ID. 0x84:Encryption Option (Forced encryption or not) Bit 0 : T1 force encrypt Bit 1 : T2 force encrypt Bit 2 : T3 force encrypt Bit 3 : T3 force encrypt when card type is 0

0x86:Masked / clear data sending option Bit 0 : T1 mask allowed
Bit 1 : T2 mask allowed
Bit 2 : T3 mask allowed

NOTE:
UniPay support 0x49,0x50,0x4C,0x7E,0x84 and 0x86.
UniPay II support 0x49,0x50,0x4A, 0x4B, 0x4C,0x7E,0x84 and 0x86.

Parameters

response	response[0] for setting data.

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode Set single setting of Mask and Encryption by Function ID.

Parameters

funcID	function ID. 0x49:Leading PAN digits to display(0x00∼0x06). 0x4A:Last PAN digits to display(0x00∼0x04). 0x4B:Mask ASCII code track data(0x20∼0x7E). 0x4C:Encryption type ('1'-'2'). '1' 3DES, '2' AES. 0x50:Mask or display expiration date(0x30 or 0x31);0x31:don't mask expiration date. 0x7E:Security Level ID. 0x84:Encryption Option (Forced encryption or not) Bit 0 : T1 force encrypt Bit 1 : T2 force encrypt Bit 2 : T3 force encrypt Bit 3 : T3 force encrypt when card type is 0

0x86:Masked / clear data sending option Bit 0 : T1 mask allowed
Bit 1 : T2 mask allowed
Bit 2 : T3 mask allowed

NOTE:
UniPay support 0x49,0x50,0x4C,0x7E,0x84 and 0x86.
UniPay II support 0x49,0x50,0x4A, 0x4B, 0x4C,0x7E,0x84 and 0x86.

Parameters

setData	for setting data.

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode Disable MSR swipe card.

Cancels MSR swipe request.

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.26 msr_defaultAllSetting()

```
int com.idtechproducts.device.IDT_SecureMag.msr_defaultAllSetting ( )
```

Default all setting of Mask and Encryption.

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.27 msr_startMSRSwipe()

```
int com.idtechproducts.device.IDT_SecureMag.msr_startMSRSwipe ( )
```

Enable MSR swipe card. Returns encrypted MSR data or function key value by call back function. The function swipeMSRData in interface OnReceiverListener will be called if swiping card data received.

See also

OnReceiverListener

Returns

success or error code. Values can be parsed with device_getResponseCodeString

See also

ErrorCode

12.2.2.28 phone_getInfoManufacture()

```
String com.idtechproducts.device.IDT_SecureMag.phone_getInfoManufacture ( )
```

Get manufacture version.

Returns

the manufacture info

12.2.2.29 phone_getInfoModel()

```
String com.idtechproducts.device.IDT_SecureMag.phone_getInfoModel ( )
```

Get phones's model number information.

Returns

the model number information.

12.2.2.30 registerListen()

```
void com.idtechproducts.device.IDT_SecureMag.registerListen ( )
```

General API:registerListen.

registerListen to enable SDK detect the phone jack plug in/off notification

12.2.2.31 release()

```
void com.idtechproducts.device.IDT_SecureMag.release ( )
```

release, make the SDK in the idle status.

12.2.2.32 setIDT_Device()

```
void com.idtechproducts.device.IDT_SecureMag.setIDT_Device (
            FirmwareUpdateTool fwTool )
```

For System Use Only

Parameters

fwTool	Parameter for firmware update

12.2.2.33 unregisterListen()

```
void com.idtechproducts.device.IDT_SecureMag.unregisterListen ( )
```

unregisterListen to disable the detect

12.2.2.34 useUSBIntentFilter()

```
static void com.idtechproducts.device.IDT_SecureMag.useUSBIntentFilter ( )  [static]
```

Use USB Intent Filter For USB Devices, you may opt to incorporate an Intent Filter that will automatically start your application when a specific USB device is attached. The SDK must be informed to bypass it's normal Enumeration of USB Devices when an Intent Filter is being use. This function MUST be called BEFORE device_setDeviceType() is executed if a USB Intent Filter is being utilized. https://developer.android.com/guide/topics/connectivity/usb/host.html

The documentation for this class was generated from the following file:

- Source_Android/IDT_SecureMag.java

12.3 com.idtechproducts.device.IDTMSRData Class Reference

Public Attributes

- EVENT_MSR_Types event
- byte **cardDataFlag**
- boolean isCTLS
- byte [] cardData
- byte **t1DecodeStatus**
- byte **t2DecodeStatus**
- byte **t3DecodeStatus**
- byte [] encTrack1
- byte [] encTrack2
- byte [] encTrack3
- String track1
- String track2

- String track3
- byte [] serialNumber
- byte [] KSN
- int track1Length
- int track2Length
- int track3Length
- boolean iccPresent
- CAPTURE_ENCODE_TYPE cardType
- CTLS_APPLICATION ctlsApplication
- byte [] optionalBytes
- byte captureEncodeStatus
- CAPTURE_ENCRYPT_TYPE captureEncryptType
- byte hasDE055
- int DE055Len
- byte [] DE055Data
- int TLVLen
- byte [] TLVData
- byte [] rawTrackData
- Map< String, byte[]> unencryptedTags
- Map< String, byte[]> encryptedTags
- Map< String, byte[]> maskedTags
- int result = ErrorCode.SUCCESS
- String fastEMV = null

12.3.1 Detailed Description

This class provides all information of card data.
Application can get the card data by calling the Properties of class IDTMSRData when finish swiping.

12.3.2 Member Data Documentation

12.3.2.1 captureEncodeStatus

```
byte com.idtechproducts.device.IDTMSRData.captureEncodeStatus
```

Get the swiped card decoded status.
0x00:decoded data success;
Bit0:1-track1 data error;
Bit1:1-track2 data error;
Bit2:1-track3 data error;
Bit3:1-track1 encrypted data error;
Bit4:1-track2 encrypted data error;
Bit5:1-track3 encrypted data error;
Bit6:1-KSN error;

12.3.2.2 captureEncryptType

```
CAPTURE_ENCRYPT_TYPE com.idtechproducts.device.IDTMSRData.captureEncryptType
```

Get the swiped card encrypted type,please see CAPTURE_ENCRYPT_TYPE for more information.
CAPTURE_ENCRYPT_TYPE_TDES:TDES;
CAPTURE_ENCRYPT_TYPE_AES:AES;

12.3.2.3 cardData

```
byte [] com.idtechproducts.device.IDTMSRData.cardData
```

Get the swiped card data.
Containing complete unparsed swipe data as received from MSR.
NOTE:
Just refer to this item cardData if the card data is the clear data.

12.3.2.4 cardType

```
CAPTURE_ENCODE_TYPE com.idtechproducts.device.IDTMSRData.cardType
```

Get the swiped card type,please see CAPTURE_ENCODE_TYPE for more information.
MSR card type:
CAPTURE_ENCODE_TYPE_ISOABA:ISO/ABA format
CAPTURE_ENCODE_TYPE_AAMVA:AAMVA format
CAPTURE_ENCODE_TYPE_Other:Other
CAPTURE_ENCODE_TYPE_Raw:Raw; undecoded format
CAPTURE_ENCODE_TYPE_JisI_II:JIS I or JIS II

12.3.2.5 ctlsApplication

```
CTLS_APPLICATION com.idtechproducts.device.IDTMSRData.ctlsApplication
```

CTLS Application

12.3.2.6 DE055Data

```
byte [] com.idtechproducts.device.IDTMSRData.DE055Data
```

Get the swiped card of DE055 data.

12.3.2.7 DE055Len

```
int com.idtechproducts.device.IDTMSRData.DE055Len
```

Get the swiped card length of DE055 data.

12.3.2.8 encryptedTags

```
Map<String, byte[]> com.idtechproducts.device.IDTMSRData.encryptedTags
```

Encrypted card data provided via TLV.

12.3.2.9 encTrack1

```
byte [] com.idtechproducts.device.IDTMSRData.encTrack1
```

Get the swiped card Track1 encrypted data.
A byte array containing Track1 encrypted data.

12.3.2.10 encTrack2

```
byte [] com.idtechproducts.device.IDTMSRData.encTrack2
```

Get the swiped card Track2 encrypted data.
A byte array containing Track2 encrypted data.

12.3.2.11 encTrack3

```
byte [] com.idtechproducts.device.IDTMSRData.encTrack3
```

Get the swiped card Track3 encrypted data.
A byte array containing Track3 encrypted data.

12.3.2.12 event

```
EVENT_MSR_Types com.idtechproducts.device.IDTMSRData.event
```

MSR type,please see EVENT_MSR_Types for more information.

12.3.2.13 fastEMV

```
String com.idtechproducts.device.IDTMSRData.fastEMV = null
```

Fast EMV String.

12.3.2.14 hasDE055

```
byte com.idtechproducts.device.IDTMSRData.hasDE055
```

The flag to indicate the availability of the swiped card DE055 data.

12.3.2.15 iccPresent

```
boolean com.idtechproducts.device.IDTMSRData.iccPresent
```

Determines if ICC is present in card (service code starts with "2" or "6".

12.3.2.16 isCTLS

```
boolean com.idtechproducts.device.IDTMSRData.isCTLS
```

Track data was captured via CTLS interface

12.3.2.17 KSN

```
byte [] com.idtechproducts.device.IDTMSRData.KSN
```

Get the swiped card KSN (Key Serial Number).
A byte array containing 10 bytes.

12.3.2.18 maskedTags

```
Map<String, byte[]> com.idtechproducts.device.IDTMSRData.maskedTags
```

Masked card data provided via TLV.

12.3.2.19 optionalBytes

```
byte [] com.idtechproducts.device.IDTMSRData.optionalBytes
```

Get optional bytes of the swiped card data.

12.3.2.20 rawTrackData

```
byte [] com.idtechproducts.device.IDTMSRData.rawTrackData
```

Get the DFEE23 MSR raw data.

12.3.2.21 result

```
int com.idtechproducts.device.IDTMSRData.result = ErrorCode.SUCCESS
```

Return error code.

12.3.2.22 serialNumber

```
byte [] com.idtechproducts.device.IDTMSRData.serialNumber
```

Get the Reader Serial Number.

12.3.2.23 TLVData

```
byte [] com.idtechproducts.device.IDTMSRData.TLVData
```

Get the swiped card TLV data.

12.3.2.24 TLVLen

```
int com.idtechproducts.device.IDTMSRData.TLVLen
```

Get the swiped card length of TLV data.

12.3.2.25 track1

```
String com.idtechproducts.device.IDTMSRData.track1
```

Get the swiped card Track1 data.
A string containing Track1 masked data expressed as hex characters.

12.3.2.26 track1Length

```
int com.idtechproducts.device.IDTMSRData.track1Length
```

Get the swiped card length of Track1 data.

12.3.2.27 track2

```
String com.idtechproducts.device.IDTMSRData.track2
```

Get the swiped card Track2 data.
A string containing Track2 masked data expressed as hex characters.

12.3.2.28 track2Length

```
int com.idtechproducts.device.IDTMSRData.track2Length
```

Get the swiped card length of Track2 data.

12.3.2.29 track3

```
String com.idtechproducts.device.IDTMSRData.track3
```

Get the swiped card Track3 data.
A string containing Track3 masked data expressed as hex characters.

12.3.2.30 track3Length

```
int com.idtechproducts.device.IDTMSRData.track3Length
```

Get the swiped card length of Track3 data.

12.3.2.31 unencryptedTags

```
Map<String, byte[]> com.idtechproducts.device.IDTMSRData.unencryptedTags
```

Unencrypted card data provided via TLV.

The documentation for this class was generated from the following file:

- Source_Android/IDTMSRData.java

12.4 com.idtechproducts.device.OnReceiverListener Interface Reference

Classes

- enum EMV_RESULT_CODE_Types

Public Member Functions

- void swipeMSRData (IDTMSRData card)
- void lcdDisplay (int mode, String[] lines, int timeout)
- void lcdDisplay (int mode, String[] lines, int timeout, byte[] languageCode, byte messageId)
- void ctlsEvent (byte event, byte scheme, byte data)
- void emvTransactionData (IDTEMVData emvData)
- void deviceConnected ()
- void deviceDisconnected ()
- void timeout (int errorCode)
- void autoConfigCompleted (StructConfigParameters profile)
- void autoConfigProgress (int progressValue)
- void msgRKICompleted (String MACResult)
- void ICCNotifyInfo (byte[] dataNotify, String strMessage)
- void msgBatteryLow ()
- void LoadXMLConfigFailureInfo (int index, String strMessage)
- void msgToConnectDevice ()
- void msgAudioVolumeAjustFailed ()
- void dataInOutMonitor (byte[] data, boolean isIncoming)

12.4.1 Detailed Description

The interface includes the callback functions for card data, PIN data and EMV data. The android activity should implement this interface then implement callback functions.

12.4.2 Member Function Documentation

12.4.2.1 autoConfigCompleted()

```
void com.idtechproducts.device.OnReceiverListener.autoConfigCompleted (
            StructConfigParameters profile )
```

The auto config process finished, and succeeded to get one profile to connect the device.

12.4.2.2 autoConfigProgress()

```
void com.idtechproducts.device.OnReceiverListener.autoConfigProgress (
            int progressValue )
```

The auto config process percent value.

12.4.2.3 ctlsEvent()

```
void com.idtechproducts.device.OnReceiverListener.ctlsEvent (
            byte event,
            byte scheme,
            byte data )
```

Contactless Event Asynchronous UI Message Event

Parameters

event	Asynchronous UI Message Event: • 0x01: LED event • 0x02: Buzzer event • 0x03: LCD event
scheme	• 0x00: ViVOtech UI Scheme • 0x02: VisaWave UI Scheme • 0x03: EMEA UI Scheme
data	Event Data: For LED event: Higher nibble: LED # 00: LED0 01: LED1 02: LED2 03: LED3 FF: all Lower nibble: 00: Off 01: On 11: No change For Buzzer event: Higher nibble: 1: short beeps 2: long beeps Lower nibble, short beep: 0: No change 1: Single beep 2: Double beep 3: Triple beep Lower nibble, long beep: 0: 200ms 1: 400ms 2: 600ms For LCD event: LCD message index

12.4.2.4 dataInOutMonitor()

```
void com.idtechproducts.device.OnReceiverListener.dataInOutMonitor (
            byte [] data,
            boolean isIncoming )
```

The input/output data notification,

Parameters

data	the input/output data.
isIncoming	true if is incoming data, false if it is out going data.

12.4.2.5 deviceConnected()

```
void com.idtechproducts.device.OnReceiverListener.deviceConnected ( )
```

Fires when device connects.

12.4.2.6 deviceDisconnected()

```
void com.idtechproducts.device.OnReceiverListener.deviceDisconnected ( )
```

Fires when device disconnects.

12.4.2.7 emvTransactionData()

```
void com.idtechproducts.device.OnReceiverListener.emvTransactionData (
            IDTEMVData emvData )
```

EMV Transaction Data

This protocol will receive results from IDT_Device::startEMVTransaction:otherAmount:timeout:cashback↩ :additionalTags:()

Parameters

emvData	EMV Results Data. Result code, card type, encryption type, masked tags, encrypted tags, unencrypted tags and KSN

12.4.2.8 ICCNotifyInfo()

```
void com.idtechproducts.device.OnReceiverListener.ICCNotifyInfo (
            byte [] dataNotify,
            String strMessage )
```

The ICC Card seated status notification,

Parameters

dataNotify	the response data.

Parameters

strMessage,the	ICC notification message information.

12.4.2.9 lcdDisplay() [1/2]

```
void com.idtechproducts.device.OnReceiverListener.lcdDisplay (
            int mode,
            String [] lines,
            int timeout )
```

LCD Display Request During an EMV transaction, this delegate will receive data to clear virtual LCD display, display messages, display menu, or display language. Applies to UniPay III

Parameters

mode	LCD Display Mode: • 0x01: Menu Display. A selection must be made to resume the transaction • 0x02: Normal Display get function key. A function must be selected to resume the transaction • 0x03: Display without input. Message is displayed without pausing the transaction • 0x04: List of languages are presented for selection. A selection must be made to resume the transaction • 0x10: Clear Screen. Command to clear the LCD screen
lines	Line(s) of data to display
timeout	Timeout value when displaying dialog box

12.4.2.10 lcdDisplay() [2/2]

```
void com.idtechproducts.device.OnReceiverListener.lcdDisplay (
            int mode,
            String [] lines,
            int timeout,
            byte [] languageCode,
            byte messageId )
```

LCD Display Request During an EMV transaction, this delegate will receive data to clear virtual LCD display, display messages, display menu, or display language. Applies to UniPay III

Parameters

mode	LCD Display Mode: • 0x01: Menu Display. A selection must be made to resume the transaction • 0x02: Normal Display get function key. A function must be selected to resume the transaction • 0x03: Display without input. Message is displayed without pausing the transaction • 0x04: List of languages are presented for selection. A selection must be made to resume the transaction • 0x10: Clear Screen. Command to clear the LCD screen
lines	Line(s) of data to display
timeout	Timeout value when displaying dialog box
languageCode	2 bytes language code ("EN", "ES", "FR", or "ZH") of the LCD message.
messageId	1 byte id (from 1 to 34) for a LCD message string.

12.4.2.11 LoadXMLConfigFailureInfo()

```
void com.idtechproducts.device.OnReceiverListener.LoadXMLConfigFailureInfo (
            int index,
            String strMessage )
```

Get the user grant to continue process ,

Parameters

index	1: "This phone model is not supported by the current SDK. Please contact supporter for assistance."; 2: "Wrong XML file name, please set the filename or enable the auto update."; 3: "The XML file does not exist and the auto update disabled."; 4: "Can't download the XML file. Please make sure the network is accessible.";
strMessage,the	message information when loading the XML file.

12.4.2.12 msgAudioVolumeAjustFailed()

```
void com.idtechproducts.device.OnReceiverListener.msgAudioVolumeAjustFailed ( )
```

The message notify the application failed to adjust the audio volume.

Parameters

strMessage,the	message of description about the failure info when to adjust the audio volume.

12.4.2.13 msgBatteryLow()

```
void com.idtechproducts.device.OnReceiverListener.msgBatteryLow ( )
```

Battery low status notification,

12.4.2.14 msgRKICompleted()

```
void com.idtechproducts.device.OnReceiverListener.msgRKICompleted (
            String MACResult )
```

RKI succeeded; MAC result as return value.

12.4.2.15 msgToConnectDevice()

```
void com.idtechproducts.device.OnReceiverListener.msgToConnectDevice ( )
```

The message notify the application to connect the device.

12.4.2.16 swipeMSRData()

```
void com.idtechproducts.device.OnReceiverListener.swipeMSRData (
            IDTMSRData card )
```

Call back function,this function will be called automatically if Card decode has been completed after swiping card.

Parameters

card	the MSR data. Card data.It is encrypted data and format is following: 1. Data Length low byte - 1 byte; 2. Data length high byte - 1 byte;

1. Card Encode Type - 1 byte.0x00/0x80-ISO/ABA format,0x01/0x81-AAMVA format,0x03/0x83-Other and 0x04/0x84-undecoded format.

2. Track1∼3 Status - 1 byte.Bit0,1,2:Track1∼3 decode and Bit3,4,5:Track1∼3 sampling.

3. Track1 data length - 1 byte.This length is the plain card data's length.

4. Track2 data length - 1 byte.

5. Track3 data length - 1 byte.

6. Clear/mask data sent status - 1 byte.
   Bit0:1–Track1 clear/mask status present,0–not present.
   Bit1:1–Track2 clear/mask status present,0–not present.
   Bit2:1–Track1 clear/mask status present,0–not present.
   Bit3∼Bit7:Reserved.Set to 0.

9.Encrypted/Hash data sent status - 1 byte.
Bit0:1–Track1 encrypted data present.
Bit1:1–Track2 encrypted data present.
Bit2:1–Track3 encrypted data present.
Bit3:1–Track1 hash data present.
Bit4:1–Track2 hash data present.
Bit5:1–Track3 hash data present.
Bit0:0.
Bit7:1–KSN present.

7. Track1 clear/mask data – Var bytes.

8. Track2 clear/mask data – Var bytes.

9. Track3 clear/mask data – Var bytes.

10. Track1 encrypted data – Var bytes.

11. Track2 encrypted data – Var bytes.

12. Track3 encrypted data – Var bytes.

13. Track1 hash data – 20 bytes if exist.

14. Track2 hash data – 20 bytes if exist.

15. Track3 hash data – 20 bytes if exist.

16. KSN – 10 bytes.

12.4.2.17 timeout()

```
void com.idtechproducts.device.OnReceiverListener.timeout (
            int errorCode )
```

Notify the plug status of phone jack. Timeout when wait for the response.
This happens in the process of get PINpad, swipe MSR, EMV Level 2 transaction

The documentation for this interface was generated from the following file:

- Source_Android/OnReceiverListener.java