|  |
| --- |
| Step 1: Sign up for a Test Account. [Sign up for a sandbox account](http://developer.authorize.net/testaccount/) to obtain a Merchant Interface Login ID and Password. These values will allow you to authenticate and make requests to the payment gateway with a session token. |
| Step 2: Download the Authorize.Net Android SDK The [Authorize.Net Android SDK](https://github.com/AuthorizeNet/sdk-mpos-android) will give you access to the full suite of applicable APIs.  Add the Authorize.Net Android SDK as a jar file in your project library folder.   1. Import the library as separate project in your workspace. 2. Export it as a Runnable JAR file (located under the Java tab). 3. Add the Jar file to the Project build path of your sample application. |
| Step 3: Setup your Android Project Create a new Android project and set the following project properties while in the New Project Wizard.   * Application Name: android\_example * Project Name: Authorize Android Example * Package Name: net.authorize * Minimum Required SDK: 14 * Target SDK: 19   Create three Activities: LoginActivity.java, ChargeCardActivity.java, and DevInfoFragment.java  Create three XML files in res/layout: activity\_login.xml, activity\_charge\_card.xml  Now you’re ready to start coding! |
| Step 4: Update the AndroidManifest.xml file If your AndroidManifest.xml file doesn’t already allow access to the Internet and your phone state, add the following snippet of code to the manifest element:   |  | | --- | | <uses-permission android:name="android.permission.INTERNET"/>      <uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE"/>      <uses-permission android:name="android.permission.ACESS\_NETWORK\_STATE"/> |   Note: If you are testing your application on external hardware (i.e. any android phone), be sure to enable Debugging (in the Developer options on your phone). Refer to the Android Developer’s Page for more information on how to properly configure hardware. |
| Step 5: Create your Login Page and Transactions Page Layouts Open your activity\_login.xml (res/layout/activity\_login.xml) and overwrite its contents with the following code.   |  | | --- | | <merge xmlns:android="http://schemas.android.com/apk/res/android"      xmlns:tools="http://schemas.android.com/tools"      android:layout\_width="wrap\_content"      tools:context="com.visa.visasampleapplication.LoginActivity" >  *<!-- Login progress -->*      <LinearLayout          android:id="@+id/login\_status"          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content"          android:layout\_gravity="center"          android:gravity="center\_horizontal"          android:orientation="vertical"          android:visibility="gone" >          <ProgressBar              style="?android:attr/progressBarStyleLarge"              android:layout\_width="wrap\_content"              android:layout\_height="wrap\_content"              android:layout\_marginBottom="8dp" />          <TextView              android:id="@+id/login\_status\_message"              android:layout\_width="wrap\_content"              android:layout\_height="wrap\_content"              android:layout\_marginBottom="16dp"              android:fontFamily="sans-serif-light"              android:text="@string/login\_progress\_signing\_in"              android:textAppearance="?android:attr/textAppearanceMedium" />      </LinearLayout>  *<!-- Login form -->*      <ScrollView          android:id="@+id/login\_form"          android:layout\_width="match\_parent"          android:layout\_height="match\_parent" >          <LinearLayout              style="@style/LoginFormContainer"              android:orientation="vertical"              android:layout\_width="wrap\_content"              android:layout\_height="wrap\_content" >  *<!-- Image of the Visa Logo -->*              <ImageView                  android:id="@+id/logo"                  android:layout\_width="wrap\_content"                  android:layout\_height="wrap\_content"                  android:layout\_gravity="top"                  android:adjustViewBounds="true"                  android:contentDescription="@string/visa\_logo"                  android:paddingLeft="15dp"                  android:paddingRight="15dp"                  android:paddingTop="45dp"                  android:paddingBottom="45dp"                  android:src="@drawable/authorize\_net" />  *<!-- Edit Text for the login ID -->*              <EditText                  android:id="@+id/loginID"                  android:layout\_width="match\_parent"                  android:layout\_height="wrap\_content"                  android:textCursorDrawable="@null"                  android:layout\_marginLeft="15dp"                  android:layout\_marginRight="15dp"                  android:layout\_marginTop="15dp"                  android:background="@drawable/image\_view\_border"                  android:ems="10"                  android:hint="@string/prompt\_loginID"                  android:inputType="text"                  android:maxLines="1"                  android:padding="10dp"                  android:singleLine="true">                  <requestFocus />              </EditText>  *<!-- Edit Text for the password -->*              <EditText                  android:id="@+id/password"                  android:layout\_width="match\_parent"                  android:layout\_height="wrap\_content"                  android:layout\_marginLeft="15dp"                  android:textCursorDrawable="@null"                  android:layout\_marginRight="15dp"                  android:layout\_marginTop="10dp"                  android:layout\_marginBottom="15dp"                  android:background="@drawable/image\_view\_border"                  android:ems="10"                  android:fontFamily="Arial"                  android:hint="@string/prompt\_password"                  android:imeActionId="@+id/login"                  android:imeOptions="actionUnspecified"                  android:inputType="textPassword"                  android:maxLines="1"                  android:padding="10dp"                  android:singleLine="true" />              <RelativeLayout                  android:layout\_width="match\_parent"                  android:layout\_height="48dp"                  android:focusable="false" >                  <View                      android:layout\_width="match\_parent"                      android:layout\_height="2dip"                      android:layout\_alignParentBottom="true"                      android:layout\_alignParentTop="true"                      android:layout\_marginLeft="15dp"                      android:layout\_marginRight="15dp"                      android:background="?android:attr/dividerHorizontal" />  *<!-- Login button -->*                  <Button                      android:id="@+id/log\_in\_button"                      android:layout\_width="match\_parent"                      android:layout\_height="match\_parent"                      android:layout\_alignParentBottom="true"                      android:layout\_marginLeft="15dp"                      android:layout\_marginRight="15dp"                      android:textColor="#FFFFFF"                      android:textAllCaps="true"                      android:background="@drawable/button\_bg"                      android:text="@string/action\_log\_in" />              </RelativeLayout>          </LinearLayout>      </ScrollView>  </merge> |   Open your activity\_charge\_card.xml (res/layout/activity\_charge\_card.xml) and overwrite its contents with the following code.   |  | | --- | | <ScrollView xmlns:android="http://schemas.android.com/apk/res/android"      android:id="@+id/charge\_card\_form"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent"      android:fillViewport="true" >       <RelativeLayout          android:layout\_width="fill\_parent"          android:layout\_height="wrap\_content"          android:paddingTop="50dp">  *<!-- Edit Text for the card number -->*          <EditText              android:id="@+id/card\_number"              android:layout\_width="match\_parent"              android:layout\_height="wrap\_content"              android:ems="10"              android:gravity="center"              android:imeOptions="actionNext"              android:hint="@string/card\_number"              android:maxLength="19"              android:inputType="number"              android:background="@drawable/image\_view\_border"              android:padding="10dp"              android:layout\_marginLeft="15dp"              android:layout\_marginRight="15dp"              android:layout\_marginBottom="10dp" >              <requestFocus />          </EditText>  *<!-- Edit Text for the expiration date -->*          <EditText              android:id="@+id/expiration\_date"              android:layout\_width="wrap\_content"              android:layout\_height="wrap\_content"              android:layout\_alignParentLeft="true"              android:imeOptions="actionNext"              android:layout\_below="@id/card\_number"              android:textCursorDrawable="@null"              android:gravity="center"              android:maxLength="5"              android:hint="@string/expiration\_date"              android:inputType="date"              android:background="@drawable/image\_view\_border"              android:padding="10dp"              android:layout\_marginLeft="15dp"              android:layout\_marginRight="2.5dp" />  *<!-- Edit Text for the CVV2 -->*          <EditText              android:id="@+id/CVV2"              android:layout\_width="wrap\_content"              android:layout\_height="wrap\_content"              android:layout\_below="@id/card\_number"              android:textCursorDrawable="@null"              android:imeOptions="actionNext"              android:layout\_toRightOf="@id/expiration\_date"              android:gravity="center"              android:maxLength="4"              android:hint="@string/CVV2"              android:inputType="number"              android:background="@drawable/image\_view\_border"              android:padding="10dp"              android:layout\_marginLeft="2.5dp"              android:layout\_marginRight="2.5dp" />    *<!-- Edit Text for the zipcode -->*          <EditText              android:id="@+id/zip\_code"              android:layout\_width="wrap\_content"              android:layout\_height="wrap\_content"              android:layout\_alignParentRight="true"              android:layout\_below="@id/card\_number"              android:layout\_marginRight="15dp"              android:layout\_toRightOf="@+id/question\_mark"              android:textCursorDrawable="@null"              android:background="@drawable/image\_view\_border"              android:gravity="center"              android:hint="@string/zip\_code"              android:imeOptions="actionDone"              android:inputType="number"              android:maxLength="5"              android:padding="10dp" />          <View              android:id="@+id/view\_charge\_card"              android:layout\_width="0.5dip"              android:layout\_height="wrap\_content"              android:layout\_alignParentBottom="true"              android:layout\_alignTop="@+id/swipe\_card\_button"              android:layout\_marginBottom="4dip"              android:layout\_centerHorizontal="true"              android:background="?android:attr/dividerVertical" />  *<!-- Swipe card button -->*          <Button              android:id="@+id/swipe\_card\_button"              android:layout\_width="wrap\_content"              android:layout\_height="wrap\_content"              android:layout\_alignParentBottom="true"              android:layout\_alignParentLeft="true"              android:layout\_toLeftOf="@id/view\_charge\_card"              android:text="@string/swipe\_card\_menu\_text"              android:textAllCaps="true"              android:textColor="#FFFFFF"              android:layout\_marginRight="5dp"              android:layout\_marginLeft="15dp"              android:layout\_marginBottom="10dp"              android:background="@drawable/button\_bg" />  *<!-- Submit button -->*          <Button              android:id="@+id/submit\_button"              android:layout\_width="wrap\_content"              android:layout\_height="wrap\_content"              android:layout\_alignParentBottom="true"              android:layout\_alignParentRight="true"              android:layout\_toRightOf="@id/view\_charge\_card"              android:background="@drawable/button\_bg"              android:text="@string/submit\_button\_text"              android:textAllCaps="true"              android:layout\_marginLeft="5dp"              android:layout\_marginRight="15dp"              android:layout\_marginBottom="10dp"              android:textColor="#FFFFFF" />  *<!-- Helper button that is associated with the CVV2 -->*          <ImageButton              android:id="@+id/question\_mark"              android:layout\_width="wrap\_content"              android:layout\_height="wrap\_content"              android:layout\_toRightOf="@id/CVV2"              android:layout\_alignTop="@+id/CVV2"              android:background="#00000000"              android:src="@drawable/ic\_question\_mark" />      </RelativeLayout>  </ScrollView> | |
| Step 6: Update your Login Page and Transactions Page Activities Open your LoginActivity.java and overwrite its contents with the following code.   |  | | --- | | **package** net**.**authorize**.**sampleapplication**;**  **import** com.visa.visasampleapplication.R**;**  **import** android.animation.Animator**;**  **import** android.animation.AnimatorListenerAdapter**;**  **import** android.annotation.TargetApi**;**  **import** android.app.Activity**;**  **import** android.app.DialogFragment**;**  **import** android.content.Context**;**  **import** android.content.Intent**;**  **import** android.net.ConnectivityManager**;**  **import** android.net.NetworkInfo**;**  **import** android.os.AsyncTask**;**  **import** android.os.Build**;**  **import** android.os.Bundle**;**  **import** android.text.TextUtils**;**  **import** android.util.Log**;**  **import** android.view.Gravity**;**  **import** android.view.KeyEvent**;**  **import** android.view.Menu**;**  **import** android.view.MenuItem**;**  **import** android.view.MotionEvent**;**  **import** android.view.View**;**  **import** android.view.View.OnKeyListener**;**  **import** android.view.ViewGroup**;**  **import** android.view.View.OnTouchListener**;**  **import** android.view.inputmethod.EditorInfo**;**  **import** android.view.inputmethod.InputMethodManager**;**  **import** android.widget.Button**;**  **import** android.widget.EditText**;**  **import** android.widget.TextView**;**  **import** android.widget.Toast**;**  **import** net.authorize.Environment**;**  **import** net.authorize.Merchant**;**  **import** net.authorize.auth.PasswordAuthentication**;**  **import** net.authorize.auth.SessionTokenAuthentication**;**  **import** net.authorize.data.mobile.MobileDevice**;**  **import** net.authorize.mobile.TransactionType**;**  **import** net.authorize.mobile.Transaction**;**  **import** net.authorize.mobile.Result**;**  */\*\* Activity which displays a login screen to the user. \*/*  **public** **class** **LoginActivity** **extends** Activity **{**  */\*\* The default email to populate the email field with. \*/*  **public** **static** **final** String EXTRA\_LOGINID **=** "com.example.android.authenticatordemo.extra.LOGINID"**;**  */\*\* Keep track of the login task to ensure we can cancel it if requested. \*/*  **private** UserLoginTask mAuthTask **=** **null;**  */\*\* Values for loginID and password at the time of the login attempt. \*/*  **private** String mLoginID**;**  **private** String mPassword**;**  */\*\* UI references. \*/*  **private** EditText mLoginIDView**;**  **private** EditText mPasswordView**;**  **private** View mLoginFormView**;**  **private** View mLoginStatusView**;**  **private** TextView mLoginStatusMessageView**;**  */\*\* Buttons. \*/*  **private** Button loginButton**;**  */\*\* Strings related to login authentication. \*/*  **private** String deviceNumber **=** ""**;**  **protected** **static** String deviceID **=** "testDeviceID"**;** *// set device ID*  **private** String deviceInfo **=** ""**;**  **protected** **static** Merchant \_merchant **=** **null;**      @Override  **protected** **void** **onCreate(**Bundle savedInstanceState**)** **{**  */\*\* Setup the login form \*/*  **super.**onCreate**(**savedInstanceState**);**          setContentView**(**R**.**layout**.**activity\_login**);**          setupUI**(**findViewById**(**R**.**id**.**login\_form**));**          getActionBar**().**show**();**          mLoginIDView **=** **(**EditText**)** findViewById**(**R**.**id**.**loginID**);**          mPasswordView **=** **(**EditText**)** findViewById**(**R**.**id**.**password**);**          mLoginStatusMessageView **=** **(**TextView**)** findViewById**(**R**.**id**.**login\_status\_message**);**          mLoginFormView **=** findViewById**(**R**.**id**.**login\_form**);**          mLoginStatusView **=** findViewById**(**R**.**id**.**login\_status**);**          loginButton **=** **(**Button**)** findViewById**(**R**.**id**.**log\_in\_button**);**          mLoginID **=** getIntent**().**getStringExtra**(**EXTRA\_LOGINID**);**          mLoginIDView**.**setText**(**mLoginID**);**  */\*\* Respond to Return on keypad. \*/*          mPasswordView  **.**setOnEditorActionListener**(new** TextView**.**OnEditorActionListener**()** **{**                      @Override  **public** **boolean** **onEditorAction(**TextView textView**,** **int** id**,**                              KeyEvent keyEvent**)** **{**  **if** **(**id **==** R**.**id**.**login **||** id **==** EditorInfo**.**IME\_ACTION\_DONE**)** **{**                              attemptLogin**();**  **return** **true;**  **}**  **return** **false;**  **}**  **});**  */\*\* Respond to login button. \*/*          loginButton**.**setOnClickListener**(**  **new** View**.**OnClickListener**()** **{**                      @Override  **public** **void** **onClick(**View view**)** **{**                          attemptLogin**();**  **}**  **});**  **}**  */\*\* Disable the back button. \*/*      @Override  **public** **void** **onBackPressed()** **{**  **}**  */\*\* Creates menu. \*/*      @Override  **public** **boolean** **onCreateOptionsMenu(**Menu menu**)** **{**  **super.**onCreateOptionsMenu**(**menu**);**          getMenuInflater**().**inflate**(**R**.**menu**.**login**,** menu**);**  **return** **super.**onCreateOptionsMenu**(**menu**);**  **}**  */\*\* Responds to each item on the menu. \*/*      @Override  **public** **boolean** **onOptionsItemSelected(**MenuItem item**)** **{**  **if** **(**item**.**getItemId**()** **==** R**.**id**.**dev\_info**)** **{**              DialogFragment devInfoFragment **=** DevInfoFragment**.**newInstance**(**getString**(**R**.**string**.**dev\_info\_message**),** getString**(**R**.**string**.**dev\_info\_title**));**              devInfoFragment**.**show**(**getFragmentManager**(),** "devInfo"**);**  **return** **true;**  **}** **else** **{**  **return** **super.**onOptionsItemSelected**(**item**);**  **}**  **}**  */\*\* Attempts to sign in or register the account specified by the login form.*  *\* If there are form errors (invalid email, missing fields, etc.), the*  *\* errors are presented and no actual login attempt is made. \*/*  **public** **void** **attemptLogin()** **{**  **if** **(**mAuthTask **!=** **null)** **{**  **return;**  **}**  */\*\* Reset errors. \*/*          mLoginIDView**.**setError**(null);**          mPasswordView**.**setError**(null);**  */\*\* Store values at the time of the login attempt. \*/*          mLoginID **=** mLoginIDView**.**getText**().**toString**();**          mPassword **=** mPasswordView**.**getText**().**toString**();**  **boolean** cancel **=** **false;**          View focusView **=** **null;**  */\*\* Check for a valid password. \*/*  **if** **(**TextUtils**.**isEmpty**(**mPassword**))** **{**              mPasswordView**.**setError**(**getString**(**R**.**string**.**error\_field\_required**));**              focusView **=** mPasswordView**;**              cancel **=** **true;**  **}** **else** **if** **(**mPassword**.**length**()** **<** 4**)** **{**              mPasswordView**.**setError**(**getString**(**R**.**string**.**error\_invalid\_password**));**              focusView **=** mPasswordView**;**              cancel **=** **true;**  **}**  */\*\* Check for a valid login ID. \*/*  **if** **(**TextUtils**.**isEmpty**(**mLoginID**))** **{**              mLoginIDView**.**setError**(**getString**(**R**.**string**.**error\_field\_required**));**              focusView **=** mLoginIDView**;**              cancel **=** **true;**  **}**  **if** **(**cancel**)** **{**  */\*\* There was an error; don't attempt login and focus the first*  *\* form field with an error. \*/*  **final** EditText errorEditText **=** **(**EditText**)** focusView**;**              errorEditText**.**requestFocus**();**              errorEditText**.**setOnKeyListener**(new** **OnKeyListener()** **{**                  @Override  **public** **boolean** **onKey(**View v**,** **int** keyCode**,** KeyEvent event**)** **{**  **if** **(**keyCode **==** KeyEvent**.**KEYCODE\_DEL**)** **{**                          errorEditText**.**setText**(**""**);**  **return** **true;**  **}**  **return** **false;**  **}**  **});**  **}** **else** **{**  */\*\* Show a progress spinner, and kick off a background task to*  *\* perform the user login attempt. \*/*              mLoginStatusMessageView**.**setText**(**R**.**string**.**login\_progress\_signing\_in**);**              showProgress**(true);**              mAuthTask **=** **new** **UserLoginTask();**              mAuthTask**.**execute**((**Void**)** **null);**  **}**  **}**  */\*\* Shows the progress UI and hides the login form. \*/*      @TargetApi**(**Build**.**VERSION\_CODES**.**HONEYCOMB\_MR2**)**  **private** **void** **showProgress(final** **boolean** show**)** **{**  **if** **(**Build**.**VERSION**.**SDK\_INT **>=** Build**.**VERSION\_CODES**.**HONEYCOMB\_MR2**)** **{**  **int** shortAnimTime **=** getResources**().**getInteger**(**                      android**.**R**.**integer**.**config\_shortAnimTime**);**              mLoginStatusView**.**setVisibility**(**View**.**VISIBLE**);**              mLoginStatusView**.**animate**().**setDuration**(**shortAnimTime**)**  **.**alpha**(**show **?** 1 **:** 0**)**  **.**setListener**(new** **AnimatorListenerAdapter()** **{**                          @Override  **public** **void** **onAnimationEnd(**Animator animation**)** **{**                              mLoginStatusView**.**setVisibility**(**show **?** View**.**VISIBLE  **:** View**.**GONE**);**  **}**  **});**              mLoginFormView**.**setVisibility**(**View**.**VISIBLE**);**              mLoginFormView**.**animate**().**setDuration**(**shortAnimTime**)**  **.**alpha**(**show **?** 0 **:** 1**)**  **.**setListener**(new** **AnimatorListenerAdapter()** **{**                          @Override  **public** **void** **onAnimationEnd(**Animator animation**)** **{**                              mLoginFormView**.**setVisibility**(**show **?** View**.**GONE  **:** View**.**VISIBLE**);**  **}**  **});**  **}** **else** **{**              mLoginStatusView**.**setVisibility**(**show **?** View**.**VISIBLE **:** View**.**GONE**);**              mLoginFormView**.**setVisibility**(**show **?** View**.**GONE **:** View**.**VISIBLE**);**  **}**  **}**  */\*\* An asynchronous login/registration task used to authenticate*  *\* the user. \*/*  **public** **class** **UserLoginTask** **extends** AsyncTask**<**Void**,** Void**,** Integer**>** **{**          Result loginResult**;**  **public** **static** **final** **int** RESULT\_FAILURE **=** **-**2**;**          @Override  **protected** Integer **doInBackground(**Void**...** params**)** **{**              authenticate**(**findViewById**(**R**.**id**.**login\_form**));**  **int** loginCode**;**  **try** **{**                  Transaction currentTransaction **=** \_merchant**.**createMobileTransaction**(**TransactionType**.**MOBILE\_DEVICE\_LOGIN**);**                  MobileDevice mobileDevice **=** MobileDevice**.**createMobileDevice**(**deviceID**,** deviceInfo**,** deviceNumber**);**                  currentTransaction**.**setMobileDevice**(**mobileDevice**);**                  loginResult **=** **(**Result**)** \_merchant**.**postTransaction**(**currentTransaction**);**                  loginCode **=** loginResult**.**isResponseOk**()** **?** RESULT\_OK **:** RESULT\_FAILURE**;**  **}** **catch** **(**Exception e**)** **{**                  loginCode **=** RESULT\_FAILURE**;**  **}**  **try** **{**                  Thread**.**sleep**(**1000**);**  **}** **catch** **(**InterruptedException e**)** **{**  **return** RESULT\_FAILURE**;**  **}**  **return** loginCode**;**  **}**          @Override  **protected** **void** **onPostExecute(final** Integer success**)** **{**              mAuthTask **=** **null;**              showProgress**(false);**  **if** **(**success **==** **-**1**)** **{**  **if** **(**loginResult**.**isOk**())** **{**                      String sessionToken **=** loginResult**.**getSessionToken**();**                      SessionTokenAuthentication sessionTokenAuthentication **=** SessionTokenAuthentication**.**createMerchantAuthentication**(**\_merchant**.**getMerchantAuthentication**().**getName**(),** sessionToken**,** deviceID**);**                      \_merchant**.**setMerchantAuthentication**(**sessionTokenAuthentication**);**                      Intent chargeCardIntent **=** **new** **Intent(**LoginActivity**.**this**,** ChargeCardActivity**.**class**);**                      startActivity**(**chargeCardIntent**);**                      finish**();**  **}**  **}** **else** **{**                  mPasswordView  **.**setError**(**getString**(**R**.**string**.**error\_incorrect\_password**));**                  mPasswordView**.**requestFocus**();**  **}**  **}**          @Override  **protected** **void** **onCancelled()** **{**              mAuthTask **=** **null;**              showProgress**(false);**  **}**  **}**  */\*\* Utility functions for LoginActivity.java. \*/*  */\*\* Dismisses the soft-key board outside of EditText area. \*/*  **public** **static** **void** **hideSoftKeyboard(**Activity activity**)** **{**          InputMethodManager inputMethodManager **=** **(**InputMethodManager**)** activity**.**getSystemService**(**Activity**.**INPUT\_METHOD\_SERVICE**);**          inputMethodManager**.**hideSoftInputFromWindow**(**activity**.**getCurrentFocus**().**getWindowToken**(),** 0**);**  **}**  */\*\* Iterates through each View in this activity and checks if it is an*  *\* instance of EditText and if it is not, register a setOnTouchlistener*  *\* to that component. \*/*  **public** **void** **setupUI(**View view**)** **{**  **if(!(**view **instanceof** EditText**))** **{**              view**.**setOnTouchListener**(new** **OnTouchListener()** **{**                  @Override  **public** **boolean** **onTouch(**View v**,** MotionEvent event**)** **{**                      hideSoftKeyboard**(**LoginActivity**.**this**);**  **return** **false;**  **}**  **});**  **}**  **if** **(**view **instanceof** ViewGroup**)** **{**  **for** **(int** i **=** 0**;** i **<** **((**ViewGroup**)** view**).**getChildCount**();** i**++)** **{**                  View innerView **=** **((**ViewGroup**)** view**).**getChildAt**(**i**);**                  setupUI**(**innerView**);**  **}**  **}**  **}**  */\*\* Displays a toast with a MESSAGE. \*/*  **public** **void** **displayToast(**String message**)** **{**          hideSoftKeyboard**(this);**          Context currentContext **=** getApplicationContext**();**          String text **=** message**;**  **int** duration **=** Toast**.**LENGTH\_SHORT**;**          Toast completedTransactionToast **=** Toast**.**makeText**(**currentContext**,** text**,** duration**);**          completedTransactionToast**.**setGravity**(**Gravity**.**CENTER\_HORIZONTAL**|**Gravity**.**CENTER\_VERTICAL**,** 0**,** 0**);**          completedTransactionToast**.**show**();**  **}**  */\*\* Authenticates the user-name and password using the VIEW. \*/*  **public** **void** **authenticate(**View view**)** **{**          String loginID **=** **((**EditText**)** findViewById**(**R**.**id**.**loginID**)).**getText**().**toString**();**          String password **=** **((**EditText**)** findViewById**(**R**.**id**.**password**)).**getText**().**toString**();**          PasswordAuthentication loginCredentials **=** PasswordAuthentication**.**createMerchantAuthentication**(**loginID**,** password**,** deviceID**);**          \_merchant **=** Merchant**.**createMerchant**(**Environment**.**SANDBOX**,** loginCredentials**);**          \_merchant**.**setDuplicateTxnWindowSeconds**(**30**);**  **}**  */\*\* Returns the NetworkInfo of the current CONTEXT. \*/*  **public** **static** **boolean** **checkNetwork(**Context context**)** **{**          ConnectivityManager cm **=** **(**ConnectivityManager**)**context**.**getSystemService**(**Context**.**CONNECTIVITY\_SERVICE**);**          NetworkInfo network **=** cm**.**getActiveNetworkInfo**();**  **if** **(**network **!=** **null)** **{**  **return** network**.**isAvailable**();**  **}**  **return** **false;**  **}**  **}** |   Open your ChargeCardActivity.java and overwrite its contents with the following code.   |  | | --- | | **package** net**.**authorize**.**sampleapplication**;**  **import** java.math.BigDecimal**;**  **import** java.util.Calendar**;**  **import** com.visa.visasampleapplication.R**;**  **import** android.app.Activity**;**  **import** android.app.AlertDialog**;**  **import** android.app.Dialog**;**  **import** android.app.DialogFragment**;**  **import** android.app.ProgressDialog**;**  **import** android.content.Context**;**  **import** android.content.DialogInterface**;**  **import** android.content.Intent**;**  **import** android.content.res.Resources**;**  **import** android.graphics.Bitmap**;**  **import** android.graphics.drawable.BitmapDrawable**;**  **import** android.graphics.drawable.Drawable**;**  **import** android.os.AsyncTask**;**  **import** android.os.Bundle**;**  **import** android.text.Editable**;**  **import** android.text.TextUtils**;**  **import** android.text.TextWatcher**;**  **import** android.util.DisplayMetrics**;**  **import** android.util.Log**;**  **import** android.view.Gravity**;**  **import** android.view.KeyEvent**;**  **import** android.view.Menu**;**  **import** android.view.MenuItem**;**  **import** android.view.MotionEvent**;**  **import** android.view.View**;**  **import** android.view.View.OnTouchListener**;**  **import** android.view.ViewGroup**;**  **import** android.view.inputmethod.EditorInfo**;**  **import** android.view.inputmethod.InputMethodManager**;**  **import** android.widget.Button**;**  **import** android.widget.EditText**;**  **import** android.widget.ImageButton**;**  **import** android.widget.TextView**;**  **import** android.widget.Toast**;**  **import** net.authorize.Merchant**;**  **import** net.authorize.TransactionType**;**  **import** net.authorize.aim.Result**;**  **import** net.authorize.aim.Transaction**;**  **import** net.authorize.aim.cardpresent.DeviceType**;**  **import** net.authorize.aim.cardpresent.MarketType**;**  **import** net.authorize.data.Order**;**  **import** net.authorize.data.creditcard.CreditCard**;**  **import** net.authorize.data.creditcard.CreditCardPresenceType**;**  **import** net.authorize.data.swiperdata.SwiperEncryptionAlgorithmType**;**  **import** net.authorize.util.Luhn**;**  **import** net.authorize.data.Address**;**  **import** net.authorize.data.Customer**;**  **import** net.authorize.data.OrderItem**;**  **import** net.authorize.data.ShippingCharges**;**  */\*\* Activity which displays screen to enter credit card information. \*/*  **public** **class** **ChargeCardActivity** **extends** Activity **{**  */\*\* Credit Card total length. \*/*  **private** **static** **final** **int** CREDIT\_CARD\_LENGTH\_W\_SPACE **=** 19**;**  */\*\* Current length of the card number in real time \*/*  **private** **int** cardNumberLen **=** 0**;**  */\*\* Credit Card \*/*  **private** **static** CreditCard creditCard**;**  */\*\* Credit Card Information \*/*  **private** EditText cardNumber**;**  **private** EditText expDate**;**  **private** EditText cvv2**;**  **private** EditText zipcode**;**  */\*\* Buttons. \*/*  **private** Button swipeCardButton**;**  **private** ImageButton questionMarkButton**;**  **private** Button submitButton**;**  */\*\* Credit Card Number converted into String \*/*  **private** String cardNumText**;**  */\*\* Current length of the expiration date in real time \*/*  **private** **int** expDateLen **=** 0**;**  */\*\* Test Order - used only for testing \*/*  **private** **static** Order testOrder**;**  */\*\* Test Order details - used only for testing \*/*  **private** BigDecimal**[]** testOrderInfo**;**  */\*\* TransactionTask to be executed \*/*  **private** ExecuteTransactionTask transactionTask**;**  */\*\* True if processing a card swipe, otherwise false. \*/*  **private** **boolean** cardSwipe**;**  */\*\* Merchant associated with transaction. \*/*  **private** Merchant merchant**;**  */\*\* Saves the default DeviceType and MarketType. \*/*  **private** DeviceType defaultDeviceType**;**  **private** MarketType defaultMarketType**;**      @Override  **protected** **void** **onCreate(**Bundle savedInstanceState**)** **{**  */\*\* Setup login page \*/*  **super.**onCreate**(**savedInstanceState**);**          setContentView**(**R**.**layout**.**activity\_charge\_card**);**          setupUI**(**findViewById**(**R**.**id**.**charge\_card\_form**));**          getActionBar**().**show**();**          cardSwipe **=** **false;**          cardNumber **=** **(**EditText**)** findViewById**(**R**.**id**.**card\_number**);**          expDate **=** **(**EditText**)** findViewById**(**R**.**id**.**expiration\_date**);**          cvv2 **=** **(**EditText**)** findViewById**(**R**.**id**.**CVV2**);**          zipcode **=** **(**EditText**)** findViewById**(**R**.**id**.**zip\_code**);**          swipeCardButton **=** **(**Button**)** findViewById**(**R**.**id**.**swipe\_card\_button**);**          questionMarkButton **=** **(**ImageButton**)** findViewById**(**R**.**id**.**question\_mark**);**          submitButton **=** **(**Button**)** findViewById**(**R**.**id**.**submit\_button**);**  */\*\* Current merchant. \*/*          merchant **=** LoginActivity**.**\_merchant**;**          testOrderInfo **=** createTestOrder**();**          defaultDeviceType **=** merchant**.**getDeviceType**();**          defaultMarketType **=** merchant**.**getMarketType**();**          formatCreditCard**();**          formatExpDate**();**          setupOnClick**();**  **}**  */\*\* Setup all on click listeners. \*/*  **public** **void** **setupOnClick()** **{**  */\*\* Respond to swipe card button \*/*          swipeCardButton**.**setOnClickListener**(new** View**.**OnClickListener**()** **{**              @Override  **public** **void** **onClick(**View view**)** **{**                  cardSwipe **=** **true;**                  ExecuteSwipeTask swipeTask **=** **new** **ExecuteSwipeTask();**                  swipeTask**.**execute**();**                  startTransaction**();**  **}**  **});**  */\*\* Displays CVV2 information \*/*          questionMarkButton**.**setOnClickListener**(new** View**.**OnClickListener**()** **{**              @Override  **public** **void** **onClick(**View view**)** **{**                  DialogFragment cvv2Info **=** DevInfoFragment**.**newInstance**(**getString**(**R**.**string**.**cvv2\_info\_message**),** getString**(**R**.**string**.**cvv2\_info\_title**));**                  cvv2Info**.**show**(**getFragmentManager**(),** "swipecard"**);**  **}**  **});**  */\*\* Respond to Done on the keypad \*/*          zipcode  **.**setOnEditorActionListener**(new** TextView**.**OnEditorActionListener**()** **{**                  @Override  **public** **boolean** **onEditorAction(**TextView v**,** **int** actionId**,** KeyEvent event**)** **{**  **if** **(**actionId **==** EditorInfo**.**IME\_ACTION\_DONE**)** **{**                          attemptSubmit**();**  **return** **true;**  **}**  **return** **false;**  **}**  **});**  */\*\* Respond to submit button \*/*          submitButton**.**setOnClickListener**(new** View**.**OnClickListener**()** **{**              @Override  **public** **void** **onClick(**View v**)** **{**                  attemptSubmit**();**  **}**  **});**  **}**  */\*\* Auto-format expiration date at real time \*/*  **public** **void** **formatExpDate()** **{**          expDate**.**addTextChangedListener**(new** **TextWatcher()** **{**  **public** **void** **afterTextChanged(**Editable s**)** **{** **}**  **public** **void** **beforeTextChanged(**CharSequence s**,** **int** start**,** **int** count**,** **int** after**)** **{** **}**  **public** **void** **onTextChanged(**CharSequence s**,** **int** start**,** **int** before**,** **int** count**)** **{**                  String expDateText **=** expDate**.**getText**().**toString**();**                  expDateLen **=** expDateText**.**length**();**  **if** **(**expDateLen **==** 3 **&&** **!(**String**.**valueOf**(**expDate**.**getText**().**toString**().**charAt**(**expDateLen **-** 1**)).**equals**(**"/"**)))** **{**                      expDate**.**setText**(new** **StringBuilder(**expDateText**).**insert**(**expDateText**.**length**()** **-** 1**,** "/"**).**toString**());**                      expDate**.**setSelection**(**expDateLen**);**  **}**  **if** **(**expDateText**.**endsWith**(**" "**))** **{**  **return;**  **}**  **}**  **});**  **}**  */\*\* Auto-format credit card text field at real time \*/*  **public** **void** **formatCreditCard()** **{**          cardNumber**.**addTextChangedListener**(new** **TextWatcher()** **{**  **public** **void** **afterTextChanged(**Editable s**)** **{** **}**  **public** **void** **beforeTextChanged(**CharSequence s**,** **int** start**,** **int** count**,** **int** after**)** **{** **}**  **public** **void** **onTextChanged(**CharSequence s**,** **int** start**,** **int** before**,** **int** count**)** **{**                  cardNumText **=** cardNumber**.**getText**().**toString**();**                  cardNumberLen **=** cardNumber**.**getText**().**length**();**  **if** **((**cardNumberLen **==** 5 **||** cardNumberLen **==** 10 **||** cardNumberLen **==** 15**)**  **&&** **!(**String**.**valueOf**(**cardNumber**.**getText**().**toString**().**charAt**(**cardNumberLen **-** 1**))**  **.**equals**(**" "**)))** **{**                      cardNumber**.**setText**(new** **StringBuilder(**cardNumText**).**insert**(**cardNumText**.**length**()** **-** 1**,** " "**).**toString**());**                      cardNumber**.**setSelection**(**cardNumber**.**getText**().**length**());**  **}**  **if** **(**cardNumText**.**endsWith**(**" "**))** **{**  **return;**  **}**  **if** **((**cardNumberLen **==** CREDIT\_CARD\_LENGTH\_W\_SPACE**))** **{**                      cardNumText **=** cardNumber**.**getText**().**toString**();**                      cardNumText**.**replace**(**" "**,** ""**);**  **if** **(**Luhn**.**isCardValid**(**cardNumText**))** **{**                          Drawable checkMark **=** getResources**().**getDrawable**(**R**.**drawable**.**ic\_check\_mark**);**                          Bitmap checkMarkBitmap **=** **((**BitmapDrawable**)** checkMark**).**getBitmap**();**                          Drawable scaledCheckMark **=** **new** **BitmapDrawable(**getResources**(),** Bitmap**.**createScaledBitmap**(**checkMarkBitmap**,** 50**,** 50**,** **true));**                          cardNumber**.**setCompoundDrawablesWithIntrinsicBounds**(null,** **null,** scaledCheckMark**,** **null);**  **}** **else** **{**                          Drawable delete **=** getResources**().**getDrawable**(**R**.**drawable**.**ic\_delete**);**                          Bitmap deleteBitmap **=** **((**BitmapDrawable**)** delete**).**getBitmap**();**                          Drawable scaledDelete **=** **new** **BitmapDrawable(**getResources**(),** Bitmap**.**createScaledBitmap**(**deleteBitmap**,** 50**,** 50**,** **true));**                          cardNumber**.**setCompoundDrawablesWithIntrinsicBounds**(null,** **null,** scaledDelete**,** **null);**  **}**  **}** **else** **{**                      cardNumber**.**setCompoundDrawablesWithIntrinsicBounds**(null,** **null,** **null,** **null);**  **}**  **}**  **});**  **}**  */\*\* Disable the back button. \*/*      @Override  **public** **void** **onBackPressed()** **{**  **}**  */\*\* Creates menu. \*/*      @Override  **public** **boolean** **onCreateOptionsMenu(**Menu menu**)** **{**          getMenuInflater**().**inflate**(**R**.**menu**.**charge\_card**,** menu**);**  **return** **super.**onCreateOptionsMenu**(**menu**);**  **}**  */\*\* Responds to each item on the menu. \*/*      @Override  **public** **boolean** **onOptionsItemSelected(**MenuItem item**)** **{**  **switch(**item**.**getItemId**())** **{**  **case** R**.**id**.**dev\_info**:**                  DialogFragment devInfoFragment **=**                      DevInfoFragment**.**newInstance**(**getString**(**R**.**string**.**dev\_info\_message**),**                              getString**(**R**.**string**.**dev\_info\_title**));**                  devInfoFragment**.**show**(**getFragmentManager**(),** "devInfo"**);**  **return** **true;**  **case** R**.**id**.**logout**:**                  startLogout**();**                  Intent logoutIntent **=** **new** **Intent(this,** LoginActivity**.**class**);**                  startActivity**(**logoutIntent**);**  **return** **true;**  **default:**  **return** **super.**onOptionsItemSelected**(**item**);**  **}**  **}**  */\*\* Functions for processing credit card transaction \*/*  */\*\* Process all credit card information and continue to submit a transaction. \*/*  **public** **void** **processCreditCardInfo()** **{**          creditCard **=** CreditCard**.**createCreditCard**();**          String**[]** expDateArray **=** expDate**.**getText**().**toString**().**split**(**"/"**);**          creditCard**.**setExpirationMonth**(**expDateArray**[**0**]);**  **int** expYear **=** Integer**.**valueOf**(**expDateArray**[**1**])** **+** 2000**;**          creditCard**.**setExpirationYear**(**String**.**valueOf**(**expYear**));**          creditCard**.**setCreditCardNumber**(**cardNumText**);**          creditCard**.**setCardCode**(**cvv2**.**getText**().**toString**());**          creditCard**.**setAnetDuplicatemaskedCardNumber**(**cardNumText**.**substring**(**15**,** 19**));**  **}**  */\*\* Create a test order. - for testing purposes only \*/*  **public** BigDecimal**[]** **createTestOrder()** **{**          testOrder **=** Order**.**createOrder**();**          testOrder**.**setTotalAmount**(new** **BigDecimal(**20**));**          OrderItem testItem1 **=** OrderItem**.**createOrderItem**();**          testItem1**.**setItemId**(**"test1ID"**);**          testItem1**.**setItemName**(**"test1Name"**);**          testItem1**.**setItemDescription**(**"test1Description"**);**          testItem1**.**setItemQuantity**(new** **BigDecimal(**1**));**          testItem1**.**setItemPrice**(new** **BigDecimal(**20**));**          testItem1**.**setItemTaxable**(false);**          testOrder**.**addOrderItem**(**testItem1**);**          testOrder**.**setPurchaseOrderNumber**(**"9999"**);**          BigDecimal**[]** orderInfo **=** **new** BigDecimal**[**2**];**          BigDecimal subtotal **=** **new** **BigDecimal(**0.00**);**          BigDecimal itemCount **=** **new** **BigDecimal(**testOrder**.**getOrderItems**().**size**()** **+** 1**);**  **for** **(**OrderItem i **:** testOrder**.**getOrderItems**())** **{**              subtotal **=** subtotal**.**add**(**i**.**getItemPrice**());**  **}**          orderInfo**[**0**]** **=** itemCount**;**          orderInfo**[**1**]** **=** subtotal**;**  **return** orderInfo**;**  **}**  */\*\* Begin executing the transaction task. \*/*  **public** **void** **startTransaction()** **{**          transactionTask **=** **new** **ExecuteTransactionTask(this);**          transactionTask**.**execute**();**  **}**  */\*\* An AysncTask to process the swipe data request. \*/*  **protected** **class** **ExecuteSwipeTask** **extends** AsyncTask**<**Object**,** Void**,** Void**>** **{**          DialogFragment swipeCardFragment**;**          @Override  **protected** **void** **onPreExecute()** **{**              merchant**.**setDeviceType**(**DeviceType**.**WIRELESS\_POS**);**              merchant**.**setMarketType**(**MarketType**.**RETAIL**);**              swipeCardFragment **=** ProgressDialogSpinner**.**newInstance**(**getString**(**R**.**string**.**swipe\_card\_alert**),** getString**(**R**.**string**.**swipe\_card\_alert\_message**));**              swipeCardFragment**.**show**(**getFragmentManager**(),** "swipe card"**);**  **}**          @Override  **protected** Void **doInBackground(**Object**...** params**)** **{**  **try** **{**                  Thread**.**sleep**(**3000**);**  **}** **catch** **(**InterruptedException e**)** **{** **}**              onReceiveSwipeData**();**  **return** **null;**  **}**          @Override  **protected** **void** **onPostExecute(**Void v**)** **{**              swipeCardFragment**.**dismiss**();**  **}**  **}**  */\*\* An AysncTask to process the transaction request. \*/*  **protected** **class** **ExecuteTransactionTask** **extends** AsyncTask**<**Object**,** Void**,** Void**>** **{**  **public** **static** **final** **int** RESULT\_FAILURE **=** **-**2**;**  **protected** net**.**authorize**.**aim**.**Result result**;**  **private** ChargeCardActivity activity **=** **null;**          ExecuteTransactionTask**(**ChargeCardActivity a**)** **{**              activity **=** a**;**  **}**          @Override  **protected** **void** **onPreExecute()** **{**  **if** **(!**cardSwipe**)** **{**                  displayToast**(**"Processing transaction..."**);**  **}**              cardSwipe **=** **false;**  **}**          @Override  **protected** Void **doInBackground(**Object**...** args**)** **{**  **try** **{**                  Thread**.**sleep**(**3000**);**                  Transaction authorizeTransaction **=** createTransaction**(**TransactionType**.**AUTH\_CAPTURE**);**                  result **=** **(**Result**)** merchant**.**postTransaction**(**authorizeTransaction**);**  **}** **catch** **(**Exception e**)** **{** **}**  **return** **null;**  **}**          @Override  **protected** **void** **onPostExecute(**Void v**)** **{**              setResultIntent**(**result**);**  **if** **(**result**.**isApproved**())** **{**                  notifyActivityLoginTaskCompleted**();**  **}** **else** **{**                  displayToast**(**"ERROR"**);**  **}**              merchant**.**setDeviceType**(**defaultDeviceType**);**              merchant**.**setMarketType**(**defaultMarketType**);**  **}**  */\*\* Display successful transaction fragment. \*/*  **private** **void** **notifyActivityLoginTaskCompleted()** **{**  **if** **(**activity **!=** **null)** **{**                  String message **=** "Your transaction of $" **+** returnTotal**()** **+** " has successfully been processed"**;**                  String title **=** "Transaction Successful!"**;**                  CompletedTransactionFragment devInfoFragment **=** CompletedTransactionFragment**.**newInstance**(**message**,** title**);**                  devInfoFragment**.**show**(**getFragmentManager**(),** "completedTransaction"**);**  **}**  **}**  */\*\* Sets the result as OK or FAILURE. \*/*  **protected** **void** **setResultIntent(**net**.**authorize**.**aim**.**Result result**)** **{**              net**.**authorize**.**aim**.**Result aimTestResult **=** **(**net**.**authorize**.**aim**.**Result**)** result**;**  **if** **(**result **!=** **null)** **{**                  aimTestResult**.**clearRequest**();**                  setResult**(**aimTestResult**.**isApproved**()?** RESULT\_OK:RESULT\_FAILURE**);**  **}**  **}**  */\*\* Returns the total amount of the test Order. - used for testing. \*/*  **public** BigDecimal **returnTotal()** **{**  **return** testOrderInfo**[**1**];**  **}**  */\*\* Returns the total item count of the test order. - used for testing. \*/*  **public** BigDecimal **returnItemCount()** **{**  **return** testOrderInfo**[**0**];**  **}**  */\*\* Creates and returns the transaction associated with an order,*  *\* sets the shipping charges, sets the tax charges, and the sets*  *\* the billing address associated with a customer. \*/*  **public** net**.**authorize**.**aim**.**Transaction **createTransaction(**TransactionType transactionType**)** **{**              BigDecimal total **=** returnTotal**();**              net**.**authorize**.**aim**.**Transaction transaction **=** merchant**.**createAIMTransaction**(**transactionType**,** total**);**              transaction**.**setCreditCard**(**creditCard**);**              ShippingCharges scharges **=** ShippingCharges**.**createShippingCharges**();**              scharges**.**setTaxAmount**(new** **BigDecimal(**5.0**));**              scharges**.**setTaxItemName**(**"Sales Tax"**);**              scharges**.**setFreightAmount**(new** **BigDecimal(**6.0**));**              scharges**.**setFreightItemName**(**"Shipping and Handling"**);**              testOrder**.**setTotalAmount**(**total**);**              testOrder**.**setOrderItems**(**testOrder**.**getOrderItems**());**              transaction**.**setShippingCharges**(**scharges**);**              transaction**.**setOrder**(**testOrder**);**              Customer testCustomer **=** Customer**.**createCustomer**();**              Address billingAddress **=** Address**.**createAddress**();**              billingAddress**.**setZipPostalCode**(**zipcode**.**getText**().**toString**());**              billingAddress**.**setFirstName**(**"John"**);**              billingAddress**.**setLastName**(**"Doe"**);**              billingAddress**.**setAddress**(**"Main Street"**);**              billingAddress**.**setCity**(**"MyCity"**);**              billingAddress**.**setCountry**(**"USA"**);**              billingAddress**.**setState**(**"WA"**);**              transaction**.**setCustomer**(**testCustomer**);**  **return** transaction**;**  **}**  **}**  */\*\* Functions to logout \*/*  */\*\* Begins executing the logout task. \*/*  **public** **void** **startLogout()** **{**          ExecuteLogoutTransactionTask logoutTransaction **=** **new** **ExecuteLogoutTransactionTask();**          logoutTransaction**.**execute**();**  **}**  */\*\* AsyncTask class to process the Logout request. \*/*  **protected** **class** **ExecuteLogoutTransactionTask** **extends** AsyncTask**<**Object**,** Void**,** Void**>** **{**          net**.**authorize**.**mobile**.**Result result**;**          @Override  **protected** Void **doInBackground(**Object**...** params**)** **{**              net**.**authorize**.**mobile**.**Transaction logoutRequestTransaction **=** merchant**.**createMobileTransaction**(**net**.**authorize**.**mobile**.**TransactionType**.**LOGOUT**);**              result **=** **(**net**.**authorize**.**mobile**.**Result**)** merchant**.**postTransaction**(**logoutRequestTransaction**);**  **return** **null;**  **}**  **}**  */\*\* Utility functions for ChargeCardActivity.java. \*/*  */\*\* Fragment that prompts user with a ProgressDialog (spinner) with a cancel button. \*/*  **public** **static** **class** **ProgressDialogSpinner** **extends** DialogFragment **{**  **public** **static** ProgressDialogSpinner **newInstance** **(**String title**,** String message**)** **{**              ProgressDialogSpinner fragment **=** **new** **ProgressDialogSpinner();**              Bundle args **=** **new** **Bundle();**              args**.**putString**(**"title"**,** title**);**              args**.**putString**(**"message"**,** message**);**              fragment**.**setArguments**(**args**);**  **return** fragment**;**  **}**          @Override  **public** Dialog **onCreateDialog(**Bundle savedInstanceState**)** **{**              String title **=** getArguments**().**getString**(**"title"**);**              String message **=** getArguments**().**getString**(**"message"**);**              ProgressDialog alert **=** **new** **ProgressDialog(**getActivity**());**              TextView titleView **=** **new** **TextView(**getActivity**());**              titleView**.**setText**(**title**);**              titleView**.**setPadding**(**15**,**15**,**15**,**15**);**              titleView**.**setTextSize**(**20**);**              titleView**.**setGravity**(**Gravity**.**CENTER**);**              alert**.**setCustomTitle**(**titleView**);**              TextView messageView **=** **new** **TextView(**getActivity**());**              messageView**.**setText**(**message**);**              messageView**.**setTextSize**(**15**);**              messageView**.**setPadding**(**15**,** 15**,** 15**,** 15**);**              messageView**.**setGravity**(**Gravity**.**CENTER**);**              alert**.**setMessage**(**message**);**              alert**.**setButton**(**DialogInterface**.**BUTTON\_NEGATIVE**,** getString**(**R**.**string**.**cancel**),** **new** Dialog**.**OnClickListener**()** **{**                  @Override  **public** **void** **onClick(**DialogInterface dialog**,** **int** which**)** **{**  **return;**  **}**  **});**              setCancelable**(false);**  **return** alert**;**  **}**  **}**  */\*\* Iterates through each View in this activity and checks if it is an*  *\* instance of EditText and if it is not, register a setOnTouchlistener*  *\* to that component. \*/*  **public** **void** **setupUI(**View view**)** **{**  **if(!(**view **instanceof** EditText**))** **{**              view**.**setOnTouchListener**(new** **OnTouchListener()** **{**                  @Override  **public** **boolean** **onTouch(**View v**,** MotionEvent event**)** **{**                      LoginActivity**.**hideSoftKeyboard**(**ChargeCardActivity**.**this**);**  **return** **false;**  **}**  **});**  **}**  **if** **(**view **instanceof** ViewGroup**)** **{**  **for** **(int** i **=** 0**;** i **<** **((**ViewGroup**)** view**).**getChildCount**();** i**++)** **{**                  View innerView **=** **((**ViewGroup**)** view**).**getChildAt**(**i**);**                  setupUI**(**innerView**);**  **}**  **}**  **}**  */\*\* Checks for credit card information errors. \*/*  **public** **void** **attemptSubmit()** **{**          cardNumber **=** **(**EditText**)** findViewById**(**R**.**id**.**card\_number**);**          expDate **=** **(**EditText**)** findViewById**(**R**.**id**.**expiration\_date**);**          cvv2 **=** **(**EditText**)** findViewById**(**R**.**id**.**CVV2**);**          zipcode **=** **(**EditText**)** findViewById**(**R**.**id**.**zip\_code**);**  */\*\* Reset errors \*/*          cardNumber**.**setError**(null);**          expDate**.**setError**(null);**          cvv2**.**setError**(null);**          zipcode**.**setError**(null);**          View focusView **=** **null;**  **boolean** cancel **=** **false;**  */\*\* Store the value at the time of the transaction process \*/*          String cardNumberString **=** cardNumber**.**getText**().**toString**();**          String expDateString **=** expDate**.**getText**().**toString**();**          String cvv2String **=** cvv2**.**getText**().**toString**();**          String zipcodeString **=** zipcode**.**getText**().**toString**();**  */\*\* Check for valid card number \*/*  **if** **(**TextUtils**.**isEmpty**(**cardNumberString**))** **{**              cardNumber**.**setError**(**getString**(**R**.**string**.**error\_field\_required**));**              focusView **=** cardNumber**;**              cancel **=** **true;**  **}** **else** **if** **(**cardNumberString**.**length**()** **!=** 19**)** **{**              cardNumber**.**setError**(**getString**(**R**.**string**.**error\_invalid\_card\_number**));**              focusView **=** cardNumber**;**              cancel **=** **true;**  **}** **else** **{**              cardNumberString **=** cardNumberString**.**replace**(**" "**,** ""**);**  **if** **(!(**TextUtils**.**isDigitsOnly**(**cardNumberString**)))** **{**                  cardNumber**.**setError**(**getString**(**R**.**string**.**error\_invalid\_card\_number**));**                  focusView **=** cardNumber**;**                  cancel **=** **true;**  **}**  **}**          String year **=** String**.**valueOf**(**Calendar**.**getInstance**().**get**(**Calendar**.**YEAR**)).**substring**(**1**);**  */\*\* Check for valid expiration date \*/*  **if** **(**TextUtils**.**isEmpty**(**expDateString**))** **{**              expDate**.**setError**(**getString**(**R**.**string**.**error\_field\_required**));**              focusView **=** expDate**;**              cancel **=** **true;**  **}** **else** **if** **(**expDateString**.**length**()** **<** 5**)** **{**              expDate**.**setError**(**getString**(**R**.**string**.**error\_invalid\_expdate**));**              focusView **=** expDate**;**              cancel **=** **true;**  **}** **else** **if** **(!(**TextUtils**.**isDigitsOnly**(**expDateString**.**substring**(**0**,** 2**)))**  **&&** **!(**TextUtils**.**isDigitsOnly**(**expDateString**.**substring**(**3**,** 5**))))** **{**              expDate**.**setError**(**getString**(**R**.**string**.**error\_invalid\_expdate**));**              focusView **=** expDate**;**              cancel **=** **true;**  **}** **else** **if** **((**Integer**.**valueOf**(**expDateString**.**substring**(**0**,** 2**))** **>** 12**)**  **||** **(**Integer**.**valueOf**(**expDateString**.**substring**(**0**,** 2**))** **<** 0**))** **{**              expDate**.**setError**(**getString**(**R**.**string**.**error\_invalid\_expdate**));**              focusView **=** expDate**;**              cancel **=** **true;**  **}** **else** **if** **((**Integer**.**valueOf**(**expDateString**.**substring**(**3**,** 5**))** **<** Integer**.**valueOf**(**year**)))** **{**              expDate**.**setError**(**getString**(**R**.**string**.**error\_invalid\_expdate**));**              focusView **=** expDate**;**              cancel **=** **true;**  **}**  */\*\* Check for valid cvv2 \*/*  **if** **(**TextUtils**.**isEmpty**(**cvv2String**))** **{**              cvv2**.**setError**(**getString**(**R**.**string**.**error\_field\_required**));**              focusView **=** cvv2**;**              cancel **=** **true;**  **}** **else** **if** **(**cvv2String**.**length**()** **<** 3**)** **{**              cvv2**.**setError**(**getString**(**R**.**string**.**error\_invalid\_cvv2**));**              focusView **=** cvv2**;**              cancel **=** **true;**  **}** **else** **if** **(!(**TextUtils**.**isDigitsOnly**(**cvv2String**)))** **{**              cvv2**.**setError**(**getString**(**R**.**string**.**error\_invalid\_cvv2**));**              focusView **=** cvv2**;**              cancel **=** **true;**  **}**  */\*\* Check for valid zipcode \*/*  **if** **(**TextUtils**.**isEmpty**(**zipcodeString**))** **{**              zipcode**.**setError**(**getString**(**R**.**string**.**error\_field\_required**));**              focusView **=** zipcode**;**              cancel **=** **true;**  **}** **else** **if** **(**zipcode**.**length**()** **<** 5**)** **{**              cvv2**.**setError**(**getString**(**R**.**string**.**error\_invalid\_zipcode**));**              focusView **=** zipcode**;**              cancel **=** **true;**  **}** **else** **if** **(!(**TextUtils**.**isDigitsOnly**(**zipcodeString**)))** **{**              cvv2**.**setError**(**getString**(**R**.**string**.**error\_invalid\_zipcode**));**              focusView **=** zipcode**;**              cancel **=** **true;**  **}**  **if** **(**cancel**)** **{**              focusView**.**requestFocus**();**  **}** **else** **{**              processCreditCardInfo**();**              startTransaction**();**  **}**  **}**  */\*\* Displays a toast after the transaction has been processed and dismisses the keyboard. \*/*  **public** **void** **displayToast(**String message**)** **{**          LoginActivity**.**hideSoftKeyboard**(this);**          Context currentContext **=** getApplicationContext**();**  **int** duration **=** Toast**.**LENGTH\_SHORT**;**          Toast completedTransactionToast **=** Toast**.**makeText**(**currentContext**,** message**,** duration**);**          completedTransactionToast**.**setGravity**(**Gravity**.**CENTER\_HORIZONTAL**|**Gravity**.**CENTER\_VERTICAL**,** 0**,** 0**);**          completedTransactionToast**.**show**();**  **}**  */\*\* Sets up the order of all of the EditTexts. \*/*  **private** **void** **setupEditText()** **{**  **final** EditText cardNumber **=** **(**EditText**)** findViewById**(**R**.**id**.**card\_number**);**  **final** EditText expDate **=** **(**EditText**)** findViewById**(**R**.**id**.**expiration\_date**);**  **final** EditText cvv2 **=** **(**EditText**)** findViewById**(**R**.**id**.**CVV2**);**  **final** EditText zip **=** **(**EditText**)** findViewById**(**R**.**id**.**zip\_code**);**          cardNumber**.**setOnEditorActionListener**(new** TextView**.**OnEditorActionListener**()** **{**              @Override  **public** **boolean** **onEditorAction(**TextView v**,** **int** actionId**,**                      KeyEvent event**)** **{**  **if** **(**actionId **==** EditorInfo**.**IME\_ACTION\_NEXT**)** **{**                      expDate**.**requestFocus**();**  **return** **true;**  **}**  **return** **false;**  **}**  **});**          expDate**.**setOnEditorActionListener**(new** TextView**.**OnEditorActionListener**()** **{**              @Override  **public** **boolean** **onEditorAction(**TextView v**,** **int** actionId**,**                      KeyEvent event**)** **{**  **if** **(**actionId **==** EditorInfo**.**IME\_ACTION\_NEXT**)** **{**                      cvv2**.**requestFocus**();**  **return** **true;**  **}**  **return** **false;**  **}**  **});**          cvv2**.**setOnEditorActionListener**(new** TextView**.**OnEditorActionListener**()** **{**              @Override  **public** **boolean** **onEditorAction(**TextView v**,** **int** actionId**,** KeyEvent event**)** **{**  **if** **(**actionId **==** EditorInfo**.**IME\_ACTION\_NEXT**)** **{**                      zip**.**requestFocus**();**  **return** **true;**  **}**  **return** **false;**  **}**  **});**  **}**  */\*\*Fragment that opens an AlertDialog with message M and title T. \*/*  **public** **static** **class** **CompletedTransactionFragment** **extends** DialogFragment **{**  **public** **static** CompletedTransactionFragment **newInstance(**String m**,** String t**)** **{**              String message **=** m**;**              String title **=** t**;**              CompletedTransactionFragment frag **=** **new** **CompletedTransactionFragment();**              Bundle args **=** **new** **Bundle();**              args**.**putString**(**"title"**,** title**);**              args**.**putString**(**"message"**,** message**);**              frag**.**setArguments**(**args**);**  **return** frag**;**  **}**          @Override  **public** Dialog **onCreateDialog(**Bundle savedInstanceState**)** **{**              String title **=** getArguments**().**getString**(**"title"**);**              String message **=** getArguments**().**getString**(**"message"**);**              AlertDialog**.**Builder builder **=** **new** AlertDialog**.**Builder**(**getActivity**());**              TextView titleView **=** **new** **TextView(**getActivity**());**              titleView**.**setPadding**(**15**,** 15**,** 15**,** 15**);**              titleView**.**setText**(**title**);**              titleView**.**setGravity**(**Gravity**.**CENTER**);**              titleView**.**setTextSize**(**20**);**              builder**.**setCustomTitle**(**titleView**);**              TextView messageView **=** **new** **TextView(**getActivity**());**              messageView**.**setText**(**message**);**              messageView**.**setTextSize**(**15**);**              messageView**.**setPadding**(**15**,** 15**,** 15**,** 15**);**              messageView**.**setGravity**(**Gravity**.**CENTER**);**              builder**.**setView**(**messageView**);**              builder**.**setPositiveButton**(**"OK"**,** **new** DialogInterface**.**OnClickListener**()** **{**                  @Override  **public** **void** **onClick(**DialogInterface dialog**,** **int** which**)** **{**                      dialog**.**dismiss**();**  **}**  **});**              builder**.**setNeutralButton**(**"Return to Login Page"**,** **new** DialogInterface**.**OnClickListener**()** **{**                  @Override  **public** **void** **onClick(**DialogInterface dialog**,** **int** which**)** **{**                      ChargeCardActivity currentActivity **=** **(**ChargeCardActivity**)** getActivity**();**                      currentActivity**.**startLogout**();**                      Intent logoutIntent **=** **new** **Intent(**currentActivity**,** LoginActivity**.**class**);**                      startActivity**(**logoutIntent**);**  **}**  **});**              AlertDialog info **=** builder**.**create**();**              setCancelable**(false);**  **return** info**;**  **}**  **}**  */\*\* Functions related to the card swipe. \*/*    */\*\* Continues processing the card swipe by assigning all credit card credentials. \*/*  **public** **void** **processSwipeData(**String hexData**,** String encryption**)** **{**          creditCard **=** CreditCard**.**createCreditCard**();**          creditCard**.**setCardPresenseType**(**CreditCardPresenceType**.**CARD\_PRESENT\_ENCRYPTED**);**          creditCard**.**getSwipperData**().**setEncryptedData**(**hexData**);**          creditCard**.**getSwipperData**().**setDeviceInfo**(**"testDeviceInfo"**);** *// set device info*          creditCard**.**getSwipperData**().**setEncryptionAlgorithm**(**SwiperEncryptionAlgorithmType**.**getEnum**(**encryption**));**  **}**  */\*\* Starts the process of the encrypted credit card information and the encryption type. \*/*  **public** **void** **onReceiveSwipeData()** **{**          String testHexData **=** "testHexData"**;***// set encrypted data from card swipe*          String encryptionType **=** SwiperEncryptionAlgorithmType**.**TDES**.**getFieldName**();**          processSwipeData**(**testHexData**,** encryptionType**);**  **}**  **}** |   Open your DevInfoFragment.java and overwrite its contents with the following code.   |  | | --- | | **package** net**.**authorize**.**sampleapplication**;**  **import** android.app.AlertDialog**;**  **import** android.app.Dialog**;**  **import** android.app.DialogFragment**;**  **import** android.content.DialogInterface**;**  **import** android.os.Bundle**;**  **import** android.view.Gravity**;**  **import** android.widget.TextView**;**  */\*\*Fragment that opens an AlertDialog with message M and title T. \*/*  **public** **class** **DevInfoFragment** **extends** DialogFragment **{**  **public** **static** DevInfoFragment **newInstance(**String m**,** String t**)** **{**          String message **=** m**;**          String title **=** t**;**          DevInfoFragment frag **=** **new** **DevInfoFragment();**          Bundle args **=** **new** **Bundle();**          args**.**putString**(**"title"**,** title**);**          args**.**putString**(**"message"**,** message**);**          frag**.**setArguments**(**args**);**  **return** frag**;**  **}**      @Override  **public** Dialog **onCreateDialog(**Bundle savedInstanceState**)** **{**          String title **=** getArguments**().**getString**(**"title"**);**          String message **=** getArguments**().**getString**(**"message"**);**          AlertDialog**.**Builder builder **=** **new** AlertDialog**.**Builder**(**getActivity**());**          TextView titleView **=** **new** **TextView(**getActivity**());**          titleView**.**setText**(**title**);**          titleView**.**setGravity**(**Gravity**.**CENTER**);**          titleView**.**setPadding**(**15**,** 15**,** 15**,** 15**);**          titleView**.**setTextSize**(**20**);**          builder**.**setCustomTitle**(**titleView**);**          TextView messageView **=** **new** **TextView(**getActivity**());**          messageView**.**setText**(**message**);**          messageView**.**setTextSize**(**15**);**          messageView**.**setPadding**(**15**,** 15**,** 15**,** 15**);**          messageView**.**setGravity**(**Gravity**.**CENTER**);**          builder**.**setView**(**messageView**);**          builder**.**setNeutralButton**(**"OK"**,** **new** DialogInterface**.**OnClickListener**()** **{**              @Override  **public** **void** **onClick(**DialogInterface dialog**,** **int** which**)** **{**                  dialog**.**dismiss**();**  **}**  **});**          AlertDialog info **=** builder**.**create**();**          setCancelable**(false);**  **return** info**;**  **}**  **}** |   Note: The purpose of the DevInfoFragment.java is used as an example of how to incorporate a Fragment into an application. |
| Step 7: Update Strings.xml and other necessary files Open your strings.xml (res/values/strings.xml) and overwrite its contents with the following code.   |  | | --- | | **<?xml version="1.0" encoding="utf-8"?>**  <resources>  *<!-- Strings related to Application Information -->*      <string name="app\_name">Authorize.Net Sample</string>  *<!-- Strings related to Enter Card Information page -->*      <string name="card\_number">Card Number</string>      <string name="expiration\_date">MM/YY</string>      <string name="CVV2">CVV2</string>      <string name="zip\_code">ZIP</string>      <string name="submit\_button\_text">Submit</string>  *<!-- Strings related to Swipe Card Page -->*      <string name="sim\_instructions">Please swipe card</string>      <string name="charge\_card">Charge Credit Card</string>      <string name="sim\_card">Simulate Credit Card</string>      <string name="swipe\_card\_alert">Please swipe your credit card</string>      <string name="swipe\_card\_alert\_message">Processing credit card information&#8230;</string>      <string name="completed\_transaction">Your transaction has been processed</string>      <string name="cancel">Cancel</string>      <string name="error\_invalid\_card\_number">This card number is invalid</string>      <string name="error\_invalid\_expdate">This expiration is invalid</string>      <string name="error\_invalid\_cvv2">This cvv2 is invalid</string>      <string name="error\_invalid\_zipcode">This zipcode is invalid</string>      <string name="cvv2\_info\_message">The security code (CVV2) is a unique three or four digit number located on the back of a credit card</string>      <string name="cvv2\_info\_title">What is a CVV2?</string>    *<!-- Strings related to Charge/ Swipe Options Menu -->*      <string name="swipe\_card\_menu\_text">Swipe Card</string>      <string name="enter\_card\_menu\_text">Enter Card Information</string>    *<!-- Strings related to Developer information -->*      <string name="dev\_info\_message">This application utilizes the Authorize.Net SDK           available on GitHub under the username AuthorizeNet. Authorize.Net is a            wholly owned subsidiary of Visa.</string>      <string name="dev\_info\_title">Developer Information</string>  *<!-- Strings related to Login -->*      <string name="prompt\_loginID">Login ID</string>      <string name="prompt\_password">Password</string>      <string name="action\_log\_in">Log In</string>      <string name="login\_progress\_signing\_in">Logging In&#8230;</string>      <string name="error\_invalid\_loginID">This login ID is invalid</string>      <string name="error\_invalid\_password">This password is too short</string>      <string name="error\_incorrect\_password">This password is incorrect</string>      <string name="error\_field\_required">This field is required</string>      <string name="visa\_logo">Visa Logo</string>      <string name="title\_activity\_login">Sign In</string>      <string name="error\_network">Unable to complete request. No internet connection.</string>  *<!-- Strings related to Logout -->*      <string name="action\_log\_out">Log Out</string>      <string name="title\_item\_list">Items</string>      <string name="title\_activity\_charge\_card">Charge Credit Card</string>      <string name="action\_settings">Settings</string>  </resources> |   Drag and the drop images into the Drawables folder as you see fit (refer to the [Sample Application](https://github.com/AuthorizeNet/sdk-mpos-android/tree/master/sampleapp) for sample images located in res/drawables).  Style your application accordingly (refer to the [Sample Application](https://github.com/AuthorizeNet/sdk-mpos-android/tree/master/sampleapp) for one styling option located in res/drawables) |
| Step 8: Run your sample application When you application starts, you will be able to enter in your Authorize.Net account login credentials.  You will then be directed to a transactions page where you will be able to enter your credit card information to process a transaction. A test order is hardcoded into the application for demonstration purposes only.  Note that the code associated with the Swipe Card button is only to demonstrate how pass the encrypted card number values into the Authorize.Net SDK for a transaction to process. Please refer to the developer guide of the specific card swiper you are using on how to integrate with their SDK in your sample application. |
| Step 9: Verify the transaction is processed Verify the success of the payment transaction on your transaction report within your Authorize.Net account. |
| Step 10: Congrats! You have successfully integrated payments in your site in 10 easy steps! |