Coin wallet - Crypto currency Android Wallet App

Created: 09/03/2018
By: iTechTheme
Email: itechtheme@gmail.com

Thank you for purchasing my app. If you have any questions that are beyond the scope of this help file, please feel free to email via my user page contact form (**https://codecanyon.net/user/itechtheme**) Thanks so much!

TABLE OF CONTENTS

1) Layout (XML) Structures
2) JAVA Code Explanation
3) Resources
4) Sources & Libraries

1) Layout (XML) Structures

Android provides a straightforward XML vocabulary that corresponds to the View classes and subclasses, such as those for widgets and layouts. You can also use Android Studio's Layout Editor tobuild your XML layout using a drag-and-drop interface.

Now if you want to edit any layout you may do this by drag and drop or by writing xml code.

Here in below one text view is presented as example.

```
<!-- sign in now description -->
    <TextView
        android:layout_width="match_parent"

        android:layout_height="wrap_content"
        android:layout_marginLeft="5dp"
        android:layout_marginRight="5dp"
        android:gravity="center"
        android:text="@string/embarrassing"
        android:textColor="@color/text_color"
        android:textSize="10dp"
        android:textStyle="bold" />
```

In android there are few size measurement available. Here we have used dp (Dot pixel).In android if you want to give a view's width or height to full then you have to use **match parent** or if you want to make the text view's height or width as text view's needs then you have to use **wrap content**.

Now suppose you want to customize this text view then follow this-

Here – the size is measured by dp (Dot Pixel)

- **android:layout_width = "match_parent"** : To change the view's width change match_parent to 15dp or as your requirement.
- **android:layout_height = "wrap_content"** : To change the view's height change wrap_content to 15dp or as your requirement.
- **android:layout_marginRight = "5dp"** : To set margin right change 5dp to your requirement. Like that you may set the margin left, top, bottom.
- **android:gravity = "center"** : To set gravity of the text change center to your requirement.
- **android:text = "your text"** : To set text to the text view put the text on the position of "your text".
- **android:textColor = "#000000"** : To set text color put your hex color code on the position of "#000000".
- **android:textSize = "10dp"** : To set text size of the text change "10dp" to your requirement.

- **android:textStyle="bold"** : To set text style like bold, Italic change "bold" to your requirement.

2) JAVA Code Explanation

There are 5 activities 5 fragments in this project. The java classes are divided into some packages. They are-

A. **Activity Package** – The all activity classes.
B. **Adapter Package** – The all adapter classes for setting data to recycler view and fragments.
C. **Fragment Package** - The all fragment classes.
D. **Util Package** – The all helper classes.

A. Activity Package

In this package the following classes are present-

I. SplashActivity.java
II. LoginActivity.java
III. MainActivity.java
IV. ScanActivity.java
V. SettingsActivity.java

i. SplashActivity.java

Splash screen is the first screen of this project. Initially it will show only for 4 seconds and then the login screen will be opened or the main screen will be opened if previously

Logged in. Basically in splash screen the company logo or banner is shown. To change the time duration of this screen follow below-

```
/**
 * handler for showing splash screen for 4 seconds and after then
finishing this activity and staring the login activity */
new Handler().postDelayed(new Runnable() {
   @Override
   public void run() {
       /**
        * finishing this activity after 4 seconds
        */
       finish();
       /**
        * starting login activity after 4 seconds
        */
       startActivity(new
Intent(SplashActivity.this,LoginActivity.class));
   }
},4000);//this will finish after 4 seconds
```

Here a handler is used to do this job. Handler will run its method after 4 seconds(4000 MS) and then first it will finish this activity and then Will start the login activity. To change the duration just change "4000" to your desired milliseconds. (1 second = 1000 MS). To change the company logo or background image

You will find a layout on this location "res/layout/content_splash.xml" and to edit this follow layout (XML) structures.

ii. **LoginActivity.java**

Login screen is the second screen in this project if the user previously not logged in then. If user previously logged in then the main activity will be opened instead of this screen.

In Login activity there are two text input fields, login button and forgot password & sign up process. To customize this class follow below-

```
/**
 * click listener for login button which will call api for login and if login succeed
then will redirecet to main transaction
 */
 btnLogin.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View view) {
        /**
         * starting the main transaction
         */
        / if you want to make api call then put the api call codes here
        startActivity(new Intent(LoginActivity.this, MainActivity.class));
    }
});
```

This is the click listener of login button. Main screen will be opened if user click on this login button. If you want to do any network operation like API call on click of this button then just put your api call codes on onclick . If you want to change the text of this login button then you will found the login layout file on layout folder then just follow the layout (XML) structures.

```
/**
 * this method will type cast the all views of this screen
 */
public void initialiseViews() {
    tvRegister = (TextView) findViewById(R.id.tv_register);
    tvForgotPassword = (TextView) findViewById(R.id.tv_forgot_password);
    btnLogin = (Button) findViewById(R.id.btn_login);
}
```

In this class there is a method called “initializeViews” which will type cast all the views of sign in layout to this class. By type casting these the activity will know the views of this class and can use them to set text, click listener etc. If you want to change any view in layout of this class or add any view then you have to add here also in this format to use it further.

iii. **MainActivity.java**

Main screen is the screen which will be loaded after app launch if the user is logged in. Here the all main tasks are done. In Main screen there are three tabs with view pager slider. Each tab has a different fragment. This three fragments are setting up with this tab layout and view pager using a custom fragment adapter.

Now if you want to add more tab with fragment then follow below-

```
public void setupFragments() {
   /**
     * setting up the all fragments with the custom fragment adapter
     */
   FragmentAdapter adapter = new
FragmentAdapter(getSupportFragmentManager());
   adapter.addFragment(new ReceiveFragment(), "Receive");
   adapter.addFragment(new TransactionsFragment(), "Activity");
   adapter.addFragment(new SendFragment(), "Send");
   /**
     * setting up the adapter with view pager slider
     */
   viewPager.setAdapter(adapter);
   /**
     * attach the view pager with tab layout so that they will work together
     */
   tableLayout.setupWithViewPager(viewPager);
   /**
     * calling the method of custom vertical icon with text tabs
     */
   setupTabWithIcons();
}
```

Here in setupFragments method there is a reference of FragmentAdapter class which is presented at adapter package. Here the all three fragments are added into the adapter by “addFragment” method And then set this adapter to this view pager and then set this viewpager with tablayout.Now if you want to add more fragment in main screen just add the fragment to this adapter.

B. Adapter Package

In this package the following classes are present-

i. FragmentAdapter.java

ii. ReceiveandSendAdapter.java

iii. SliderSendAdapter.java

iv.SliderTransactionAdapter.java

i. **FragmentAdapter.java**

In Fragment adapter the all given fragments are making ready to setup with any view pager slider. When a fragment is adding to adapter a title is also provided. This title is setting with each
Fragment as title.

ii. **ReceiveandSendAdapter.java**

Receive and send adapter is used to setting up data to the recycler view list in transaction adapter. There is a type variable which value is 0 and 1. If value is 0 then the title will be set to receive and if the value is 1 then the title will be set to send. There is a custom row layout which is used to inflate each row in the recycler view. Now if you want to add or modify the row layout you need to edit this layout. You will found it on layout folder named “recycler_row”. Also u need to edit this adapter class. Follow below-

```
public static class GenericViewHolder extends RecyclerView.ViewHolder {
    /**
     * field instances of all views of the row
     */
    TextView tv_title;
    ImageView row_icon;

    public GenericViewHolder(View itemView) {
        super(itemView);
        /**
         * type casting all views
         * if you want to add more views add them here
         */
        tv_title = (TextView) itemView.findViewById(R.id.tv_title);
        row_icon = (ImageView) itemView.findViewById(R.id.row_icon);
    }

}
```

Here this is a view holder class which extending recyclerview view holder. This is used to hold up the row view. If you add more view to recycler_row.xml then you have to add it into this.

iii. **SliderSendAdapter.java**

SliderSend Adapter is used to set data to the infinite cycler view pager which is presented in “SendFragment.java” And “TransactionFragment.java” to show the coin balance. There is also a row layout
Which will be inflated into infinite cycler view pager. If you add more view to the row or modify the row then just follow below-

```
  /**
     * initiating each item and its all views and setting up the values to the views
     * @param container
     * @param position
     * @return
     */

    @Override
    public Object instantiateItem(ViewGroup container, int position) {
        /**
         * inflating the views with defined layout
         */
        View itemView = mLayoutInflater.inflate(R.layout.slider_layout, container,
false);

/**
 * type casting the views
 */
      TextView tvName = (TextView)
itemView.findViewById(R.id.tv_coin_name);
      /**
       * setting text to the textview name
       */
      tvName.setText(coinNameList.get(position));
      /**
       * adding this view to the item view row
       */
      container.addView(itemView);
      /**
       * returning the whole view
       */
      return itemView;
   }
```

In this instantiateItem method the row layout is inflating into each row position. The all views of row layout are type casted here, and set the text to the “tvname” text field. If you want to add more view to the row layout then you have to type cast the view here and then you can use it for further. The row layout you will found on layout folder named “slider_layout”.

C. Fragment Package

In this package the following classes are present-

i. ForgotPasswordFragment.java

ii. ReceiveFragment.java

iii. SendFragment.java

iv. SignupFragment.java

v. TransactionFragment.java

i. **ForgotPassowordFragment.java**

Forgot password screen will be shown when user will press on the forgot password text on sign in screen. There are one text input field for getting email and on button for send email confirmation. Now if you want to do network operation or API call on send button click then just follow below-

```
/**
 * click listener of send now button
 */
btnSendNow.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View view) {
      /**
       * do api call
       */
   }
});
```

In Send button click listener do you api call.

ii. **RececiveFragment.java**

Receive Screen will be shown in main screen first position. In Receive Screen there are qr code generator which will generate code from given text. If you want to do api call on receive screen for getting data then just follow below-

```
/**
 * this is a callback for when the fragment is visible to user and the qr code is
generated only when this page is visible
 * @param isVisibleToUser
 */

@Override
public void setUserVisibleHint(boolean isVisibleToUser) {
   /**
    * if this page is visible to user then create qr code method will executed
    */
   if (isVisibleToUser) {
      /**
```

```
        * Do API call
        * create the qr code with given string
        */
       createQrCode("this is test QR code");
   }
   super.setUserVisibleHint(isVisibleToUser);
}
```

setUserVisibleHint method will be called when this receive screen will be visible to the user. So here you will do your API call to get data and create qr code and do other using data.

v. **TransactionFragment.java**

Transaction screen is the default screen in main screen.Here is a slider to show the balance of coins.If you want to customize this slide then just follow below-

```
/**
 * setting up the slider with demo data and modifying the infinite cycle view pager
as requirements
 */
public void setupSlider(){
   /**
    * defining the data by array list
    * add coin here
    */
   ArrayList<String> arrayListCoinName = new ArrayList<String>();
   arrayListCoinName.add("BTC");
   arrayListCoinName.add("TCN");
   arrayListCoinName.add("DASH");
   arrayListCoinName.add("DUTCH");
   arrayListCoinName.add("NXT");
   /**
    * setting up the data with adapter in infinite cycle view pager
    */
   infiniteCycleViewPager.setAdapter(new
SliderTransactionAdapter(getActivity(), arrayListCoinName));
```

```
    /**
     * modifying the infinite cycle view pager as requirements
     */
    infiniteCycleViewPager.setScrollDuration(500);
    infiniteCycleViewPager.setMediumScaled(true);
    infiniteCycleViewPager.setMaxPageScale(0.8F);
    infiniteCycleViewPager.setMinPageScale(0.5F);
    infiniteCycleViewPager.setCenterPageScaleOffset(30.0F);
    infiniteCycleViewPager.setMinPageScaleOffset(5.0F);

}
```

setupSlider method will setup the data and setup the infinite cycler view pager slider to show the coins balance. To add more coins just add coin to the array list. Or if you want to get coin list from api call then get coin list from API call and replace this array list with parsed coin list.

There is also another section called “activity”. In activity there are two section “receive” and “send”. There is also a recycler view list to show the receive and sent activity info as a list.

```
/**
 * here 0 is the type which mean receive and the 0 is default
 */
recyclerView.setAdapter(new ReceiveandSendAdapter(getActivity(),0));
```

Here by default adapter is set to the recycler view with type 0 means receive. Now if you want to API call for getting data and then first get data list from API and set adapter to this recycler view.

D. Util Package

In this package the following classes are present-

i. CustomZXingScannerView.java

ii. StatusBarView.java

i. CustomZxingScannerView.java

This is a custom qr scanner view using Zxing library to make custom the qr scan camera design.If you want to change the qr scan camera border to different color just follow below-

```
/**
 * initializing the required things for scanner
 */
private void init() {
    /**
     * setting up three paint by setting its value of required things
     */
    //set up laser paint
    mLaserPaint = new Paint();
    mLaserPaint.setColor(mDefaultLaserColor);
    mLaserPaint.setStyle(Paint.Style.FILL);

    //finder mask paint
    mFinderMaskPaint = new Paint();
    mFinderMaskPaint.setColor(mDefaultMaskColor);

    //border paint
    mBorderPaint = new Paint();
    mBorderPaint.setColor(Color.parseColor("#62B1FC"));
    mBorderPaint.setStyle(Paint.Style.STROKE);
    mBorderPaint.setStrokeWidth(mDefaultBorderStrokeWidth);

    mBorderLineLength = mDefaultBorderLineLength;
}
```

Here change the color ”#62B1FC” of borders as your requirements. Or you can do many other customization here.

ii. StatusBarView.java

This class is used to customize the status bar. If you want to change the status bar of any activity layout then just follow below-

```
                 <!-- contain the custom status bar in app bar layout -->
  <android.support.design.widget.AppBarLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      app:elevation="0dp"
      android:theme="@style/AppTheme.AppBarOverlay">
<!-- status bar -->
      <com.wallet.cryptocurrency.UtilPackage.StatusBarView
          android:layout_width="match_parent"
          android:layout_height="wrap_content"
          android:background="@drawable/status_bar_bg" />
  </android.support.design.widget.AppBarLayout>
```

Put the status bar portion in app bar layout in any activity layout and then set background image or style to it. Then the status bar of that activity will be changed to it.

Or you can put it into content page also.

3) Resources

There is a folder named "Drawable" which contains the all resources in this project. So there you will find your all necessary resources. Now if you want to change the app logo then just put the logo image file in the drawble folder and then just follow below –

There is a file called manifest.xml. You have to edit this file to change your app name or app logo.

```
<application
   android:allowBackup="true"
   android:icon="@drawable/app_icon"
   android:label="Coin Wallet"
   android:supportsRtl="true"
   android:theme="@style/AppTheme">
```

In manifest.xml file there is a tag named “application” .In application tag change this line “android:icon="@drawable/app_icon” to “android:icon="@drawable/your app logo name without extension”. As like that if you want to change your app name then just set your app name on android:label=”your app name”.

If you want to change your app package name then you have to refactor the package name and clean build the project. To do this just follow below –

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.wallet.cryptocurrency">
```

In manifest file there is a section named “package”. You have to change the package name there .Just right click on mouse on the each portion on behalf of “.”, then there you will found refactor option and then select refactor and then another list will open. There you will found rename option. Select rename and then another dialog will be pop up. There change the package name to your desired one and check all check boxes and click on refactor. Wait some moments. Then the package name will be changed to your desired one.

4) Sources & Libraries

We’ve used following third party libraries in this project.

- Infinite cycle view pager for showing coin in a vertical slider (**com.github.devlight:infinitecycleviewpager:1.0.2**)
- Zxing QR scanner for scanning QR code and generating QR (**me.dm7.barcodescanner:zxing:1.8.4**)

Thanks again for purchasing the app and looking forward to helping you if you’ve any questions regarding this app.

iTechtheme.