MyAppTemplates

What's Here documentation

Android Template App

Usage instructions:

On Eclipse:

Make sure that you have configured the Eclipse with Android ADT Plugin and also installed JDK 7 and Android SDK with latest plateforms.
For detail you can visit at: http://developer.android.com/sdk/index.html

After Eclipse is fully setup, open the Eclipse, Go to Files menu then select Import-> Existing project into workspace -> Browse, then select for all the projects (Main Template project and all the related Library files bundled in the Zip provided to you) then click on Finish.
Sometime Eclipse may show errors about Library project missing, for that just Right click on Project -> Properties -> Android-> Scroll down -> Remove all the lib projects that has Red cross -> Click Apply then OK -> Reopen this window and Add back those projects -> Click Apply then OK

Important Class files:

- All the Activity classes can be found in Main package i.e. com.whatshere
- All the Fragments and UI components can be found in *ui* Package i.e. com.whatshere.ui
- All the custom code classes can be found in *custom* package i.e. com.whatshere.custom

- All the Java bean classes are located in *model* package i.e. com.whatshere.model
- You can put all the Utility classes in *utils* package i.e. com.whatshere.utils

Photoshop files:

The Photoshop files can be found in the 'PSD' folder. They are layered - to use, simply
open them up in your design tool of choice (i.e. Adobe Photoshop).
==========
Below you will find detailed source code documentation, that will help you in using and making modifications to this template.
Do not hesitate to email at: contact@myapptemplates.com should you run into any issues.

Contents

Class CustomActivity 6

- Field Detail 6
- Constructor Detail 6
- Method Detail 6

Class CustomFragment 8

- Constructor Detail 8
- Method Detail 8

Class Data 9

- Field Detail 10
- Constructor Detail 10
- Method Detail 10

Class LeftNavAdapter 11

- Field Detail 12
- Constructor Detail 12
- Method Detail 12

Class Login 13

- Constructor Detail 13
- Method Detail 14

Class MainActivity 14

- Field Detail 15
- Constructor Detail 15
- Method Detail 15

Class MainFragment 17

- Nested Class Summary 17
- Field Detail 18
- Constructor Detail 18
- Method Detail 18

Class MapViewer 19

- Nested Class Summary 19
- Field Detail 20
- Constructor Detail 20
- Method Detail 20

Class Place 22

- Field Detail 23
- Constructor Detail 23
- Method Detail 24

Class PlaceDetail 27

- Constructor Detail 27
- Method Detail 27

Class PlaceList 27

- Nested Class Summary 28
- Field Detail 28
- Constructor Detail 29
- Method Detail 29

Class Places 31

- Nested Class Summary 31
- Field Detail 31
- Constructor Detail 32
- Method Detail 32

Class Profile 33

- Constructor Detail 33
- Method Detail 33

Class Settings 33

- Constructor Detail 34
- Method Detail 34

Class SplashScreen 34

- Field Detail 35
- Constructor Detail 35
- Method Detail 35

Class TouchEffect 36

- Constructor Detail 36
- Method Detail 36

com.whatshere.custom

Class CustomActivity

Direct Known Subclasses:

Login, MainActivity, PlaceDetail, PlaceList

```
public class CustomActivity
extends android.support.v4.app.FragmentActivity
implements android.view.View.OnClickListener
```

This is a common activity that all other activities of the app can extend to inherit the common behaviors like setting a Theme to activity.

- **Field Detail**
  - ***TOUCH***

    ```
    public static final TouchEffect TOUCH
    ```

    Apply this Constant as touch listener for views to provide alpha touch effect. The view must have a Non-Transparent background.

- **Constructor Detail**
  - ***CustomActivity***

    ```
    public CustomActivity()
    ```

- **Method Detail**
  - ***onCreate***

    ```
    protected void onCreate(android.os.Bundle arg0)
    ```

    Overrides:

`onCreate` in class `android.support.v4.app.FragmentActivity`

  - ***onOptionsItemSelected***

    ```
    public boolean onOptionsItemSelected(android.view.MenuItem item)
    ```

Overrides:

`onOptionsItemSelected` in class `android.app.Activity`

- ***setupActionBar***

  `protected void setupActionBar()`

  This method will setup the top title bar (Action bar) content and display values. It will also setup the custom background theme for ActionBar. You can override this method to change the behavior of ActionBar for particular Activity

- ***onClick***

  `public void onClick(android.view.View v)`

  Specified by:

`onClick` in interface `android.view.View.OnClickListener`

- ***setTouchNClick***

  `public android.view.View setTouchNClick(int id)`

  Sets the touch and click listeners for a view..

  Parameters:

`id` - the id of View

  Returns:

the view

- ***setClick***

  `public android.view.View setClick(int id)`

  Sets the click listener for a view.

  Parameters:

`id` - the id of View

Returns:

the view

com.whatshere.custom

Class CustomFragment

Direct Known Subclasses:

MainFragment, MapViewer, Profile, Settings

```
public class CustomFragment
extends android.support.v4.app.Fragment
implements android.view.View.OnClickListener
```

The Class CustomFragment is the base Fragment class. You can extend your Fragment classes with this class in case you want to apply common set of rules for those Fragments.

- **Constructor Detail**
  - ***CustomFragment***

    ```
    public CustomFragment()
    ```

- **Method Detail**
  - ***onCreateView***
  - ```
    public android.view.View onCreateView(android.view.LayoutInflater
     inflater,
                                  android.view.ViewGroup container,

    android.os.Bundle savedInstanceState)
    ```
    ```

Overrides:

`onCreateView` in class `android.support.v4.app.Fragment`

- ***setTouchNClick***

  `public android.view.View setTouchNClick(android.view.View v)`

  Set the touch and click listener for a View.

  Parameters:

`v` - the view

  Returns:

the same view

- ***onClick***

  `public void onClick(android.view.View v)`

  Specified by:

`onClick` in interface `android.view.View.OnClickListener`

com.whatshere.model

Class Data

```
public class Data
extends java.lang.Object
```

The Class Data is simple Java bean class that holds two members only to represent dummy data for app. You can customize or can write new bean classes as per you needs.

- **Field Detail**
  - ***texts***

    ```
    private java.lang.String[] texts
    ```

    The texts.

  - ***resources***

    ```
    private int[] resources
    ```

    The resources.

- **Constructor Detail**
  - ***Data***
  - ```
    public Data(java.lang.String[] texts,
        int[] resources)
    ```

    Instantiates a new data.

    Parameters:

`texts` - the texts

`resources` - the resources

- **Method Detail**
  - ***getTexts***

    ```
    public java.lang.String[] getTexts()
    ```

    Gets the texts.

    Returns:

the texts

  - ***setTexts***

    ```
    public void setTexts(java.lang.String[] texts)
    ```
    ```

Sets the texts.

Parameters:

`texts` - the new texts

- ***getResources***

  ```
  public int[] getResources()
  ```

  Gets the resources.

  Returns:

the resources

- ***setResources***

  ```
  public void setResources(int[] resources)
  ```

  Sets the resources.

  Parameters:

`resources` - the new resources

com.whatshere.ui

Class LeftNavAdapter

```
public class LeftNavAdapter
extends android.widget.BaseAdapter
```

The Adapter class for the ListView displayed in the left navigation drawer.

- **Field Detail**
  - ***items***

    ```
    private java.util.ArrayList<Data> items
    ```

    The items.

  - ***context***

    ```
    private android.content.Context context
    ```

    The context.

  - ***selection***

    ```
    private int selection
    ```

    The selection.

- **Constructor Detail**
  - ***LeftNavAdapter***
  - ```
    public LeftNavAdapter(android.content.Context context,
                  java.util.ArrayList<Data> items)
    ```

    Instantiates a new left navigation adapter.

    Parameters:

`context` - the context of activity

`items` - the array of items to be displayed on ListView

- **Method Detail**
  - ***isSelection***

    ```
    public int isSelection()
    ```

    Checks if is selection.

    Returns:

the int
    ```

- ***setSelection***

  ```
  public void setSelection(int selection)
  ```

  Sets the selection.

  Parameters:

`selection` - the new selection

- ***getCount***

  ```
  public int getCount()
  ```

- ***getItem***

  ```
  public Data getItem(int arg0)
  ```

- ***getItemId***

  ```
  public long getItemId(int position)
  ```

- ***getView***
- `public android.view.View getView(int position,`
- `android.view.View convertView,`
  `android.view.ViewGroup parent)`

com.whatshere

Class Login

```
public class Login
extends CustomActivity
```

The Activity Login is launched after the Splash screen. You need to write your logic for actual Login.

- **Constructor Detail**
  - ***Login***

    ```
    public Login()
    ```

- **Method Detail**
  - ***onCreate***

    ```
    protected void onCreate(android.os.Bundle savedInstanceState)
    ```

    Overrides:

`onCreate` in class `CustomActivity`

  - ***setupView***

    ```
    private void setupView()
    ```

    Setup the click & other events listeners for the view components of this screen. You can add your logic for Binding the data to TextViews and other views as per your need.

  - ***onClick***

    ```
    public void onClick(android.view.View v)
    ```

    Specified by:

`onClick` in interface `android.view.View.OnClickListener`

    Overrides:

`onClick` in class `CustomActivity`

com.whatshere

Class MainActivity

```
public class MainActivity
extends CustomActivity
```

The Activity MainActivity will launched after the Login and it is the Home/Base activity of the app which holds all the Fragments and also show the Sliding Navigation drawer. You can write your code for displaying actual items on Drawer layout.

- **Field Detail**
  - ***drawerLayout***

    ```
    private android.support.v4.widget.DrawerLayout drawerLayout
    ```

    The drawer layout.

  - ***drawerLeft***

    ```
    private android.widget.ListView drawerLeft
    ```

    ListView for left side drawer.

  - ***drawerToggle***

    ```
    private android.support.v4.app.ActionBarDrawerToggle drawerToggle
    ```

    The drawer toggle.

- **Constructor Detail**
  - ***MainActivity***

    ```
    public MainActivity()
    ```

- **Method Detail**
  - ***onCreate***

    ```
    protected void onCreate(android.os.Bundle savedInstanceState)
    ```

    Overrides:

    onCreate in class CustomActivity

  - ***setupDrawer***

    ```
    private void setupDrawer()
    ```

Setup the drawer layout. This method also includes the method calls for setting up the Left & Right side drawers.

- ***setupLeftNavDrawer***

  ```
  private void setupLeftNavDrawer()
  ```

  Setup the left navigation drawer/slider. You can add your logic to load the contents to be displayed on the left side drawer. It will also setup the Header and Footer contents of left drawer. This method also apply the Theme for components of Left drawer.

- ***setupContainer***

  ```
  private void setupContainer(int pos)
  ```

  Setup the container fragment for drawer layout. This method will setup the grid view display of main contents. You can customize this method as per your need to display specific content.

  Parameters:

`pos` - the new up container

- ***onPostCreate***

  ```
  protected void onPostCreate(android.os.Bundle savedInstanceState)
  ```

  Overrides:

`onPostCreate` in class `android.app.Activity`

- ***onConfigurationChanged***

  ```
  public void onConfigurationChanged(android.content.res.Configurat
  ion newConfig)
  ```

  Specified by:

`onConfigurationChanged` in interface `android.content.ComponentCallbacks`

Overrides:

`onConfigurationChanged` in class `android.support.v4.app.FragmentActivity`

- ***onOptionsItemSelected***

  ```
  public boolean onOptionsItemSelected(android.view.MenuItem item)
  ```

  Overrides:

`onOptionsItemSelected` in class `CustomActivity`

com.whatshere.ui

Class MainFragment

```
public class MainFragment
extends CustomFragment
```

The Class MainFragment is the base fragment that shows the GridView of various Place categories. You can add your code to do whatever you want related to categories for your app.

- ## Nested Class Summary

Nested Classes

Modifier and Type	Class and Description
private class	**MainFragment.GridAdapter** The Class GridAdapter is the adapter for grid view used in this fragment.

- **Field Detail**
  - ***iList***

    ```
    private java.util.ArrayList<Data> iList
    ```

    The category list.

- **Constructor Detail**
  - ***MainFragment***

    ```
    public MainFragment()
    ```

- **Method Detail**
  - ***onCreateView***
  - `public android.view.View onCreateView(android.view.LayoutInflater inflater,`
  - `android.view.ViewGroup container,`

    `android.os.Bundle savedInstanceState)`

    Overrides:

    onCreateView in class CustomFragment

  - ***onClick***

    ```
    public void onClick(android.view.View v)
    ```

    Specified by:

    `onClick` in interface `android.view.View.OnClickListener`

    Overrides:

    onClick in class CustomFragment

  - ***setupView***

    ```
    private void setupView(android.view.View v)
    ```

    Setup the view components for this fragment. You write your code for initializing the views, setting the adapters, touch and click listeners etc.

    Parameters:

    `v` - the base view of fragment

- ***loadDummyData***

  ```
  private void loadDummyData()
  ```

  Load dummy categories data for displaying on the GridView. You need to write your own code for loading real categories from Web-service or API and displaying them on GridView.

com.whatshere.ui

Class MapViewer

```
public class MapViewer
extends CustomFragment
implements android.view.View.OnClickListener
```

The Class MapViewer is the fragment that shows the Google Map. You can add your code to do whatever you want related to Map functions for your app. For example you can add Map markers here or can show places on map.

- ## Nested Class Summary

Nested Classes

Modifier and Type	Class and Description
private	MapViewer.CustomInfoWindowAdapter

`class`	This class creates a Custom a InfoWindowAdapter that is used to show popup on map when user taps on a pin on the map.

- ## Field Detail
  - ***mMapView***

    ```
    private com.google.android.gms.maps.MapView mMapView
    ```

    The map view.

  - ***mMap***

    ```
    private com.google.android.gms.maps.GoogleMap mMap
    ```

    The Google map.

  - ***pList***

    ```
    private java.util.ArrayList<Place> pList
    ```

    The place list.

- ## Constructor Detail
  - ***MapViewer***

    ```
    public MapViewer()
    ```

- ## Method Detail
  - ***onCreateView***
  - ```
    public android.view.View onCreateView(android.view.LayoutInflater
     inflater,
    ```
  - ```
                                  android.view.ViewGroup container,

    android.os.Bundle savedInstanceState)
    ```

    Overrides:

    onCreateView in class CustomFragment

  - ***initMap***
  - ```
    private void initMap(android.view.View v,
                android.os.Bundle savedInstanceState)
    ```

    Initialize the Map view.

    Parameters:
    ```

`v` - the v

`savedInstanceState` - the saved instance state object passed from OnCreateView method of fragment.

- ***setupMapMarkers***

  `private void setupMapMarkers()`

  This method can be used to show the markers on the map. Current implementation of this method will show only a few dummy Pins with title and snippet. You must customize this method to show the pins as per your need.

- ***loadDummyData***

  `private void loadDummyData()`

  Load dummy places data for displaying on the Map View. You need to write your own code for loading real data from Web-service or API and displaying them on Map View.

- ***onResume***

  `public void onResume()`
  Overrides:

`onResume` in class `android.support.v4.app.Fragment`

- ***onPause***

  `public void onPause()`
  Overrides:

`onPause` in class `android.support.v4.app.Fragment`

- ***onDestroy***

  `public void onDestroy()`
  Overrides:

`onDestroy` in class `android.support.v4.app.Fragment`

- ***onLowMemory***

  `public void onLowMemory()`

  Specified by:

`onLowMemory` in interface `android.content.ComponentCallbacks`

Overrides:

`onLowMemory` in class `android.support.v4.app.Fragment`

- ***onSaveInstanceState***

  `public void onSaveInstanceState(android.os.Bundle outState)`

  Overrides:

`onSaveInstanceState` in class `android.support.v4.app.Fragment`

- ***onClick***

  `public void onClick(android.view.View v)`

  Specified by:

`onClick` in interface `android.view.View.OnClickListener`

Overrides:

onClick in class CustomFragment

com.whatshere.model

Class Place

```
public class Place
extends java.lang.Object
```

The Class Place is java bean class that holds all the properties for a Place.

- **Field Detail**
  - ***title***

    ```
    private java.lang.String title
    ```

    The title.

  - ***address***

    ```
    private java.lang.String address
    ```

    The address.

  - ***icon***

    ```
    private int icon
    ```

    The icon.

  - ***rating***

    ```
    private int rating
    ```

    The rating.

  - ***geo***

    ```
    private com.google.android.gms.maps.model.LatLng geo
    ```

    The geo.

- **Constructor Detail**
  - ***Place***
  - ```
    public Place(java.lang.String title,
    ```
  - ```
         java.lang.String address,
    ```
  - ```
         com.google.android.gms.maps.model.LatLng geo,
         int icon)
    ```

    Instantiates a new place.

    Parameters:
    ```

`title` - the title

`address` - the address

`geo` - the geo

`icon` - the icon

- ***Place***
- `public Place(java.lang.String title,`
- `java.lang.String address,`
- `int rating,`
  `int icon)`

  Instantiates a new place.

  Parameters:

`title` - the title

`address` - the address

`rating` - the rating

`icon` - the icon

- **Method Detail**
  - ***getTitle***

    `public java.lang.String getTitle()`

    Gets the title.

    Returns:

the title

  - ***setTitle***

    `public void setTitle(java.lang.String title)`

    Sets the title.

    Parameters:

`title` - the new title

- ***getAddress***

  `public java.lang.String getAddress()`

  Gets the address.

  Returns:

the address

- ***setAddress***

  `public void setAddress(java.lang.String address)`

  Sets the address.

  Parameters:

`address` - the new address

- ***getIcon***

  `public int getIcon()`

  Gets the icon.

  Returns:

the icon

- ***setIcon***

  `public void setIcon(int icon)`

  Sets the icon.

  Parameters:

`icon` - the new icon

- ***getGeo***

```
public com.google.android.gms.maps.model.LatLng getGeo()
```

Gets the geo.

Returns:

the geo

- ***setGeo***

```
public void setGeo(com.google.android.gms.maps.model.LatLng geo)
```

Sets the geo.

Parameters:

`geo` - the new geo

- ***getRating***

```
public int getRating()
```

Gets the rating.

Returns:

the rating

- ***setRating***

```
public void setRating(int rating)
```

Sets the rating.

Parameters:

`rating` - the new rating

com.whatshere

Class PlaceDetail

```
public class PlaceDetail
extends CustomActivity
```

The PlaceDetail is the activity class that shows details about a selected Place. This activity only shows dummy detail text and Image, you need to load and display actual contents.

- **Constructor Detail**
  - ***PlaceDetail***

    ```
    public PlaceDetail()
    ```

- **Method Detail**
  - ***onCreate***

    ```
    protected void onCreate(android.os.Bundle savedInstanceState)
    ```

    Overrides:

onCreate in class CustomActivity

com.whatshere

Class PlaceList

```
public class PlaceList
extends CustomActivity
```

The PlaceList is the activity class that shows Map fragment and List Fragment. This activity shows Places listing with two option to view places on Map or in List. You need to write actual logic for displaying real place listing and locations. Both List and Map are added in ViewPager to provide Sliding navigation.

-
  - ## Nested Class Summary

Nested Classes

Modifier and Type	Class and Description
private class	**PlaceList.DummyPageAdapter** The Class DummyPageAdapter is a dummy pager adapter for ViewPager.

- ## Field Detail
  - ***searchView***

    ```
    private android.widget.SearchView searchView
    ```

    The search view.

  - ***currentTab***

    ```
    private android.view.View currentTab
    ```

    The current tab.

  - ***pager***

    ```
    private android.support.v4.view.ViewPager pager
    ```

The pager.

- **Constructor Detail**
  - ***PlaceList***

    ```
    public PlaceList()
    ```

- **Method Detail**
  - ***onCreate***

    ```
    protected void onCreate(android.os.Bundle savedInstanceState)
    ```

    Overrides:

`onCreate` in class `CustomActivity`

  - ***onClick***

    ```
    public void onClick(android.view.View v)
    ```

    Specified by:

`onClick` in interface `android.view.View.OnClickListener`

Overrides:

`onClick` in class `CustomActivity`

  - ***initTabs***

    ```
    private void initTabs()
    ```

    Initialize the tabs. You can write your code related to Tabs.

  - ***setCurrentTab***

    ```
    private void setCurrentTab(int page)
    ```

    Sets the current selected tab. Called whenever a Tab is selected either by clicking on Tab button or by swiping the ViewPager. You can write your code related to tab selection actions.

    Parameters:

`page` - the current page of ViewPager

- ***initPager***

  ```
  private void initPager()
  ```

  Initialize the ViewPager. You can customize this method for writing the code related to view pager actions.

- ***onCreateOptionsMenu***

  ```
  public boolean onCreateOptionsMenu(android.view.Menu menu)
  ```

  Overrides:

`onCreateOptionsMenu` in class `android.app.Activity`

- ***onOptionsItemSelected***

  ```
  public boolean onOptionsItemSelected(android.view.MenuItem item)
  ```

  Overrides:

`onOptionsItemSelected` in class `CustomActivity`

- ***setupSearchView***

  ```
  protected void setupSearchView(android.view.Menu menu)
  ```

  Setup the up search view for ActionBar search. The current implementation simply search for Videos from Youtube. You can customize this code to search from your API or any TV API.

  Parameters:

`menu` - the ActionBar Menu

- ***setupSearchViewTheme***

  ```
  protected void setupSearchViewTheme(android.widget.SearchView sea
  rchView)
  ```

  Sets the up search view theme.

  Parameters:

`searchView` - the new up search view theme

com.whatshere.ui

Class Places

```
public class Places
extends android.support.v4.app.Fragment
```

The Class Places is a Fragment that is displayed in the PlaceList activity when the user taps on List tab (2nd tab) or when user swipes to Second page in ViewPager. You can customize this fragment's contents as per your need. It just show a list of dummy places with dummy logo for each place.

-
  - ## Nested Class Summary

Nested Classes

Modifier and Type	Class and Description
private class	**Places.PlaceAdapter** The Class PlaceAdapter is the adapter for list view used in this fragment.

- ## Field Detail
  - ***pList***

```
private java.util.ArrayList<Place> pList
```

The place list.

- **Constructor Detail**
  - ***Places***

    ```
    public Places()
    ```

- **Method Detail**
  - ***onCreateView***
  - ```
    public android.view.View onCreateView(android.view.LayoutInflater
     inflater,
    ```
  - ```
                                  android.view.ViewGroup container,
    ```

    ```
    android.os.Bundle savedInstanceState)
    ```

    Overrides:

`onCreateView` in class `android.support.v4.app.Fragment`

  - ***setupView***

    ```
    private void setupView(android.view.View v)
    ```

    Setup the view components for this fragment. You write your code for initializing the views, setting the adapters, touch and click listeners etc.

    Parameters:

`v` - the base view of fragment

  - ***loadDummyData***

    ```
    private void loadDummyData()
    ```

    Load dummy places data for displaying on the Listview. You need to write your own code for loading real data from Web-service or API and displaying them on ListView.

com.whatshere.ui

Class Profile

```
public class Profile
extends CustomFragment
```

The Class Profile is the fragment that shows the User Profile. You can add your code to do whatever you want related to user profile information for your app. For example you can load and display actual image of user.

- **Constructor Detail**
  - ***Profile***

    ```
    public Profile()
    ```

- **Method Detail**
  - ***onCreateView***
  - ```
    public android.view.View onCreateView(android.view.LayoutInflater
     inflater,
                                          android.view.ViewGroup container,
    android.os.Bundle savedInstanceState)
    ```
    Overrides:

onCreateView in class CustomFragment

com.whatshere.ui

Class Settings
    ```

```
public class Settings
extends CustomFragment
```

The Class Settings is the fragment that shows the App Settings. You need to write functionality for each Setting option and you can change any of the options as per your need.

- **Constructor Detail**
  - ***Settings***

    ```
    public Settings()
    ```

- **Method Detail**
  - ***onCreateView***
  - ```
    public android.view.View onCreateView(android.view.LayoutInflater
     inflater,
                                  android.view.ViewGroup container,

    android.os.Bundle savedInstanceState)
    ```
    Overrides:

onCreateView in class CustomFragment

com.whatshere

Class SplashScreen

```
public class SplashScreen
extends android.app.Activity
```
```

The Class SplashScreen will launched at the start of the application. It will be displayed for 3 seconds and than finished automatically and it will also start the next activity of app.

- **Field Detail**
  - ***isRunning***

    ```
    private boolean isRunning
    ```

    Check if the app is running.

- **Constructor Detail**
  - ***SplashScreen***

    ```
    public SplashScreen()
    ```

- **Method Detail**
  - ***onCreate***

    ```
    public void onCreate(android.os.Bundle savedInstanceState)
    ```

    Overrides:

`onCreate` in class `android.app.Activity`

  - ***startSplash***

    ```
    private void startSplash()
    ```

    Starts the count down timer for 3-seconds. It simply sleeps the thread for 3-seconds.

  - ***doFinish***

    ```
    private void doFinish()
    ```

    If the app is still running than this method will start the Login activity and finish the Splash.

  - ***onKeyDown***
  - ```
    public boolean onKeyDown(int keyCode,
                             android.view.KeyEvent event)
    ```

    Specified by:

`onKeyDown` in interface `android.view.KeyEvent.Callback`
    ```

Overrides:

`onKeyDown` in class `android.app.Activity`

com.whatshere.utils

Class TouchEffect

```
public class TouchEffect
extends java.lang.Object
implements android.view.View.OnTouchListener
```

The Class TouchEffect is a Base Touch listener. It can be attached as touch listener for any view. It simply set alpha value for whole view to create a Touch like effect on View

- **Constructor Detail**
  - ***TouchEffect***

    ```
    public TouchEffect()
    ```

- **Method Detail**
  - ***onTouch***
  - ```
    public boolean onTouch(android.view.View v,
                           android.view.MotionEvent event)
    ```

    Specified by:

`onTouch` in interface `android.view.View.OnTouchListener`
    ```