Course: 405-02: Mobile Application Development – 2

Unit-2 : Basic Attributes and Events of Important Android Widgets(UI)

2.1 ListView, Custom ListView

Android ListView

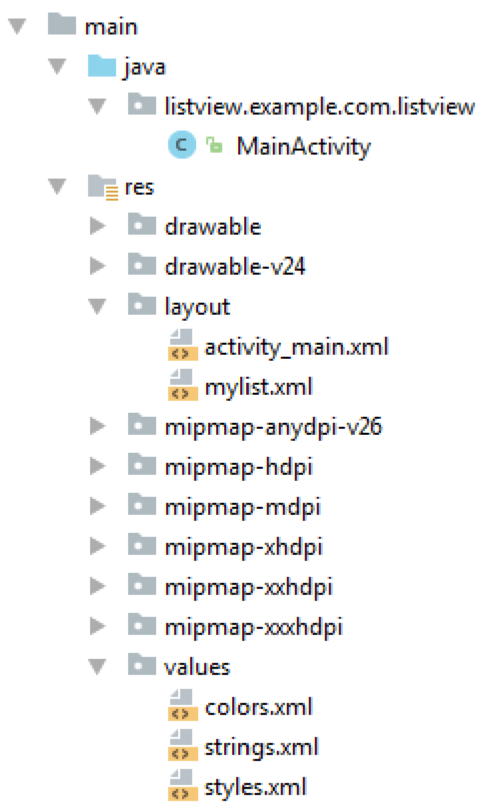
Android **ListView** is a view which contains the group of items and displays in a scrollable list. ListView is implemented by importing *android.widget.ListView* class. ListView is a default scrollable which does not use other scroll view.

ListView uses Adapter classes which add the content from data source (such as string array, array, database etc) to ListView. Adapter bridges data between an *AdapterViews* and other Views (ListView, ScrollView etc).

Example of ListView

Let's implement a simple listview example.

Structure of listview project



```
main
  java
    listview.example.com.listview
      MainActivity
  res
    drawable
    drawable-v24
    layout
      activity_main.xml
      mylist.xml
    mipmap-anydpi-v26
    mipmap-hdpi
    mipmap-mdpi
    mipmap-xhdpi
    mipmap-xxhdpi
    mipmap-xxxhdpi
    values
      colors.xml
      strings.xml
      styles.xml
```

activity_main.xml

First we need to drag and drop ListView component from palette to activity_main.xml file.

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.an
droid.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="listview.example.com.listview.MainActivity">

    <ListView
        android:id="@+id/listView"
        android:layout_width="match_parent"
        android:layout_height="fill_parent"
        />
</android.support.constraint.ConstraintLayout>
```

Create an additional mylist.xml file in layout folder which contains view components displayed in listview.

mylist.xml

File: mylist.xml

```
<?xml version="1.0" encoding="utf-8"?>

<TextView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Medium Text"
    android:textStyle="bold"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:layout_marginLeft="10dp"
```

```
        android:layout_marginTop="5dp"
        android:padding="2dp"
        android:textColor="#4d4d4d"
        />
```

Now place the list of data in strings.xml file by creating string-array.

strings.xml

File:strings.xml

```
<resources>
    <string name="app_name">ListView</string>
    <string-array name="array_technology">
        <item>Android</item>
        <item>Java</item>
        <item>Php</item>
        <item>Hadoop</item>
        <item>Sap</item>
        <item>Python</item>
        <item>Ajax</item>
        <item>C++</item>
        <item>Ruby</item>
        <item>Rails</item>
        <item>.Net</item>
        <item>Perl</item>
    </string-array>
</resources>
```

Activity class

In java class we need to add adapter to listview using setAdapter() method of listview.

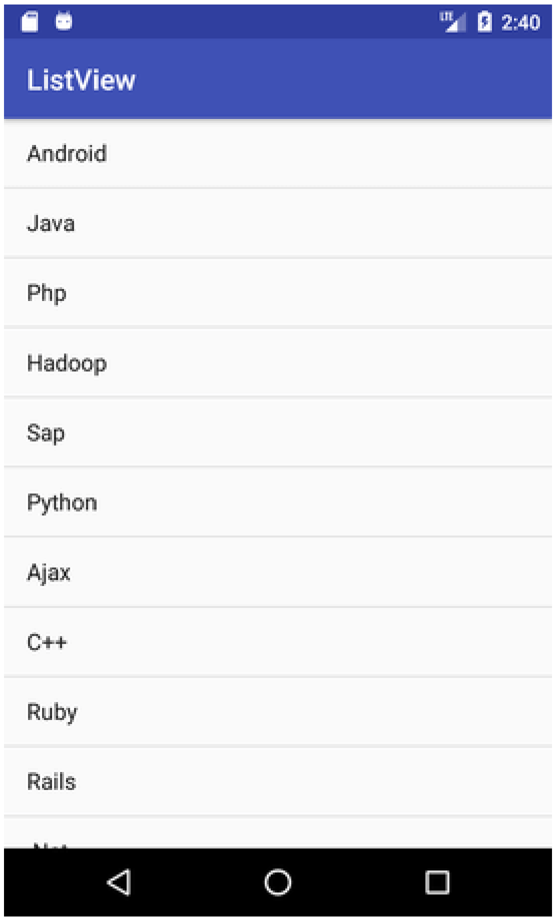
File: MainActivity.java

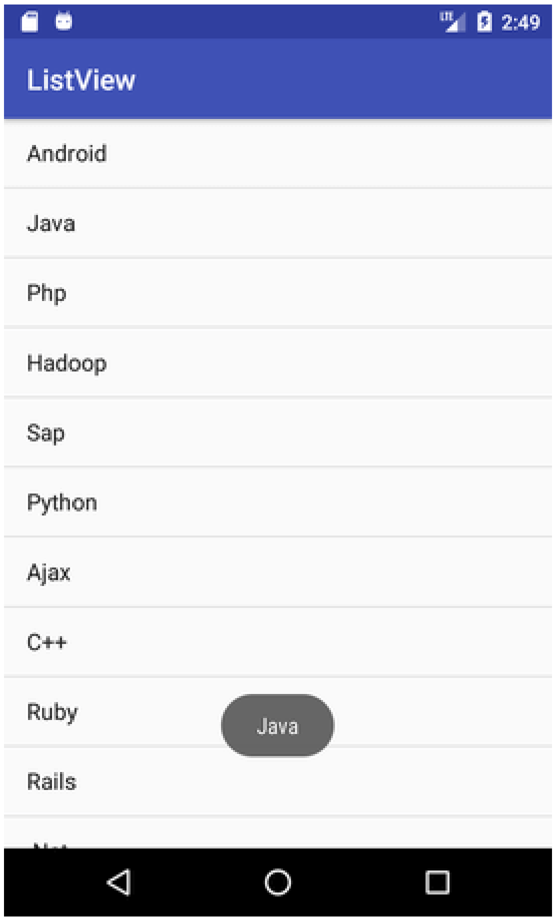
```
package listview.example.com.listview;
```

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    ListView listView;
    TextView textView;
    String[] listItem;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        listView=(ListView)findViewById(R.id.listView);
        textView=(TextView)findViewById(R.id.textView);
        listItem = getResources().getStringArray(R.array.array_technology);
        final ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
                android.R.layout.simple_list_item_1, android.R.id.text1, listItem);
        listView.setAdapter(adapter);

        listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> adapterView, View view, int positio
n, long l) {
                // TODO Auto-generated method stub
                String value=adapter.getItem(position);
                Toast.makeText(getApplicationContext(),value,Toast.LENGTH_SHORT).show
();
```

```
            }
          });
        }
    }
```

Output:


2:40
ListView
Android
Java
Php
Hadoop
Sap
Python
Ajax
C++
Ruby
Rails


2:49
ListView
Android
Java
Php
Hadoop
Sap
Python
Ajax
C++
Ruby
Java
Rails

Android Custom ListView (Adding Images, sub-title)

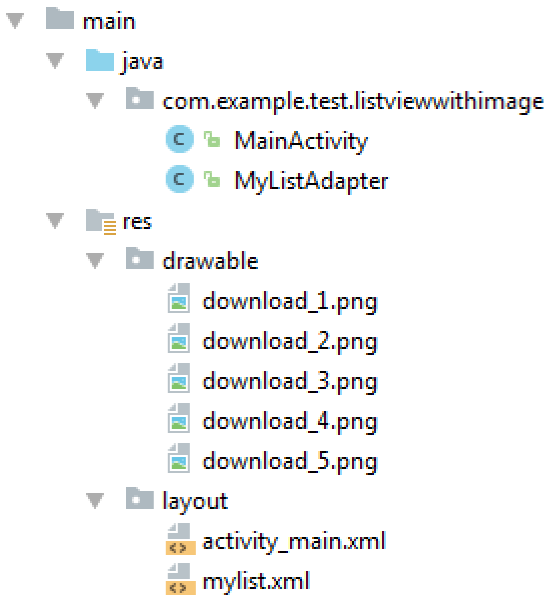
After creating simple ListView, android also provides facilities to customize our ListView.

As the simple ListView, custom ListView also uses Adapter classes which added the content from data source (such as string array, array, database etc). Adapter bridges data between an AdapterViews and other Views

Example of Custom ListView

In this custom listview example, we are adding image, text with title and its sub-title.

Structure of custom listview project



```
main
  java
    com.example.test.listviewwithimage
      MainActivity
      MyListAdapter
  res
    drawable
      download_1.png
      download_2.png
      download_3.png
      download_4.png
      download_5.png
    layout
      activity_main.xml
      mylist.xml
```

activity_main.xml

Create an activity_main.xml file in layout folder.

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingBottom="@dimen/activity_vertical_margin"
```

```xml
        android:paddingLeft="@dimen/activity_horizontal_margin"
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin"
        tools:context="com.example.test.listviewwithimage.MainActivity">

        <ListView
            android:id="@+id/list"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginBottom="50dp">
        </ListView>
    </RelativeLayout>
```

Create an additional mylist.xml file in layout folder which contains view components displayed in listview.

mylist.xml

File: mylist.xml

```xml
    <?xml version="1.0" encoding="utf-8"?>
    <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="horizontal" >

        <ImageView
            android:id="@+id/icon"
            android:layout_width="60dp"
            android:layout_height="60dp"
            android:padding="5dp" />

        <LinearLayout android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:orientation="vertical">
        <TextView
            android:id="@+id/title"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Medium Text"
            android:textStyle="bold"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:layout_marginLeft="10dp"
            android:layout_marginTop="5dp"
            android:padding="2dp"
            android:textColor="#4d4d4d" />
        <TextView
            android:id="@+id/subtitle"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="TextView"
            android:layout_marginLeft="10dp"/>
        </LinearLayout>
    </LinearLayout>
```

Place the all required images in drawable folder.

Activity class

File: MainActivity.java

```
package com.example.test.listviewwithimage;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ListView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    ListView list;

    String[] maintitle ={
            "Title 1","Title 2",
            "Title 3","Title 4",
            "Title 5",
    };

    String[] subtitle ={
            "Sub Title 1","Sub Title 2",
            "Sub Title 3","Sub Title 4",
            "Sub Title 5",
    };

    Integer[] imgid={
            R.drawable.download_1,R.drawable.download_2,
            R.drawable.download_3,R.drawable.download_4,
            R.drawable.download_5,
    };
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        MyListAdapter adapter=new MyListAdapter(this, maintitle, subtitle,imgid);
        list=(ListView)findViewById(R.id.list);
        list.setAdapter(adapter);



        list.setOnItemClickListener(new AdapterView.OnItemClickListener() {

            @Override
            public void onItemClick(AdapterView<?> parent, View view,int position, lon
g id) {
```

```
                // TODO Auto-generated method stub
                if(position == 0) {
                    //code specific to first list item
                    Toast.makeText(getApplicationContext(),"Place Your First Option Code",
Toast.LENGTH_SHORT).show();
                }

                else if(position == 1) {
                    //code specific to 2nd list item
                    Toast.makeText(getApplicationContext(),"Place Your Second Option Cod
e",Toast.LENGTH_SHORT).show();
                }

                else if(position == 2) {

                    Toast.makeText(getApplicationContext(),"Place Your Third Option Code",
Toast.LENGTH_SHORT).show();
                }
                else if(position == 3) {

                    Toast.makeText(getApplicationContext(),"Place Your Forth Option Code",
Toast.LENGTH_SHORT).show();
                }
                else if(position == 4) {

                    Toast.makeText(getApplicationContext(),"Place Your Fifth Option Code",
Toast.LENGTH_SHORT).show();
                }


            }
        });
    }
}
```

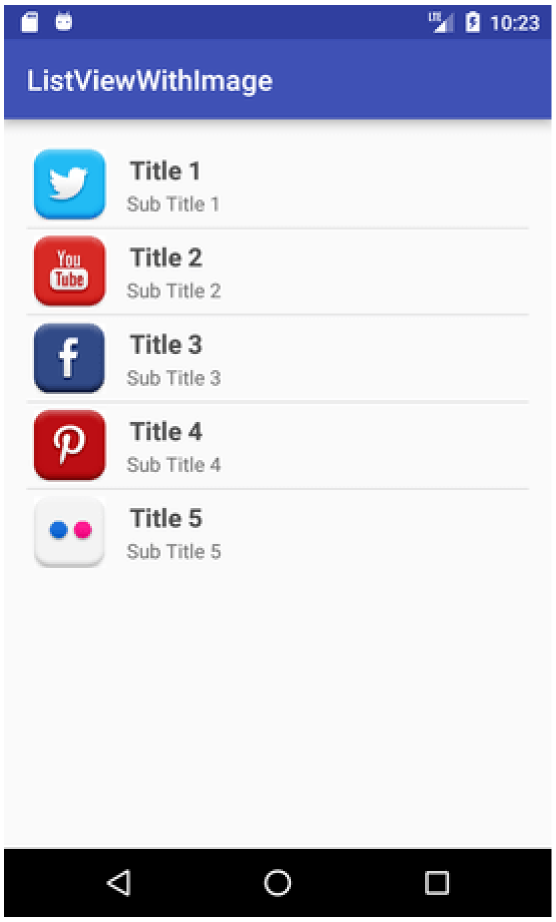
Customize Our ListView

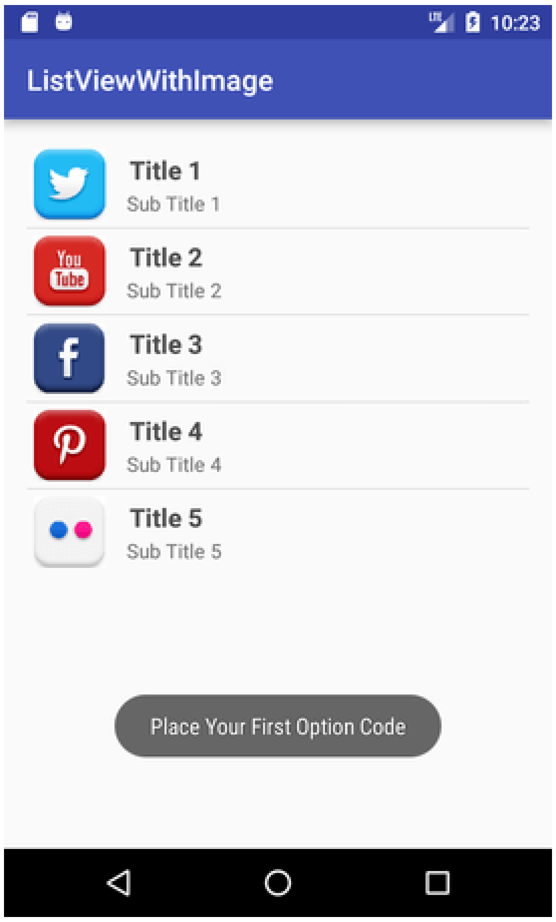
Create another java class MyListView.java which extends ArrayAdapter class. This class customizes our listview.

MyListView.java

```
package com.example.test.listviewwithimage;

import android.app.Activity;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;

public class MyListAdapter extends ArrayAdapter<String> {

   private final Activity context;
   private final String[] maintitle;
   private final String[] subtitle;
   private final Integer[] imgid;

   public MyListAdapter(Activity context, String[] maintitle,String[] subtitle, Integer[] i
mgid) {
      super(context, R.layout.mylist, maintitle);
      // TODO Auto-generated constructor stub

      this.context=context;
      this.maintitle=maintitle;
      this.subtitle=subtitle;
      this.imgid=imgid;

   }
```

```
    public View getView(int position,View view,ViewGroup parent) {
        LayoutInflater inflater=context.getLayoutInflater();
        View rowView=inflater.inflate(R.layout.mylist, null,true);

        TextView titleText = (TextView) rowView.findViewById(R.id.title);
        ImageView imageView = (ImageView) rowView.findViewById(R.id.icon);
        TextView subtitleText = (TextView) rowView.findViewById(R.id.subtitle);

        titleText.setText(maintitle[position]);
        imageView.setImageResource(imgid[position]);
        subtitleText.setText(subtitle[position]);

        return rowView;

    };
}
```

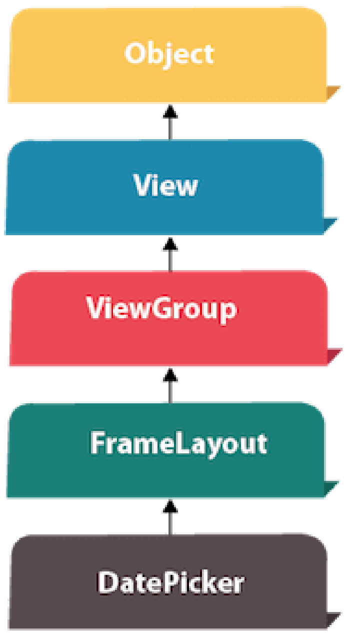




Place Your First Option Code

2.2 DatePicker, TimePicker, ProgressBar

Android DatePicker



Android DatePicker is a widget to select date. It allows you to select date by day, month and year. Like DatePicker, android also provides TimePicker to select time.

The android.widget.DatePicker is the subclass of FrameLayout class.

Android DatePicker Example

Let's see the simple example of datepicker widget in android.

activity_main.xml

File: activity_main.xml

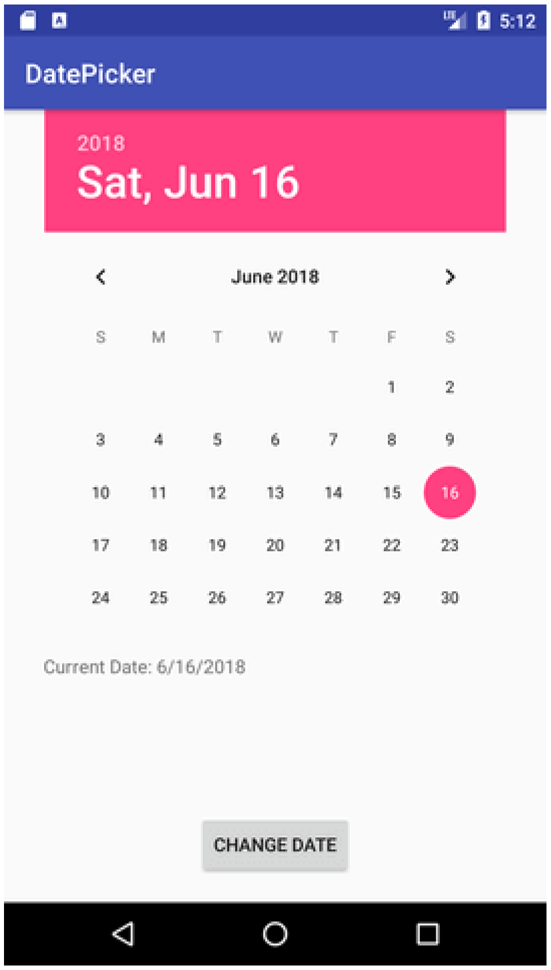
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context="example.javatpoint.com.datepicker.MainActivity">

  <TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_above="@+id/button1"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:layout_marginBottom="102dp"
    android:layout_marginLeft="30dp"
    android:layout_marginStart="30dp"
    android:text="" />
  <Button
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_centerHorizontal="true"
    android:layout_marginBottom="20dp"
    android:text="Change Date" />

  <DatePicker
    android:id="@+id/datePicker"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_above="@+id/textView1"
    android:layout_centerHorizontal="true"
    android:layout_marginBottom="36dp" />

</RelativeLayout>
```

Activity class

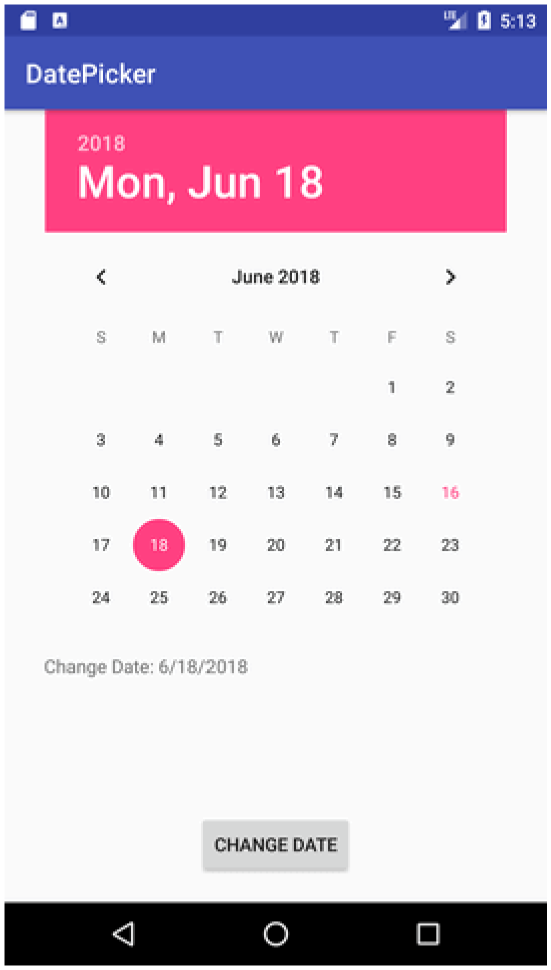
File: MainActivity.java

```
package example.javatpoint.com.datepicker;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    DatePicker picker;
    Button displayDate;
    TextView textview1;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        textview1=(TextView)findViewById(R.id.textView1);
        picker=(DatePicker)findViewById(R.id.datePicker);
        displayDate=(Button)findViewById(R.id.button1);

        textview1.setText("Current Date: "+getCurrentDate());

        displayDate.setOnClickListener(new View.OnClickListener(){
            @Override
            public void onClick(View view) {

                textview1.setText("Change Date: "+getCurrentDate());
            }

        });

    }
    public String getCurrentDate(){
        StringBuilder builder=new StringBuilder();;
        builder.append((picker.getMonth() + 1)+"/");//month is 0 based
        builder.append(picker.getDayOfMonth()+"/");
        builder.append(picker.getYear());
        return builder.toString();
    }
}
```

Output:



Current Date: 6/16/2018



Change Date: 6/18/2018

Android TimePicker

Android TimePicker widget is used to select date. It allows you to select time by hour and minute. You cannot select time by seconds.

The android.widget.TimePicker is the subclass of FrameLayout class.

Android TimePicker Example

Let's see a simple example of android time picker.

activity_main.xml

File: activity_main.xml

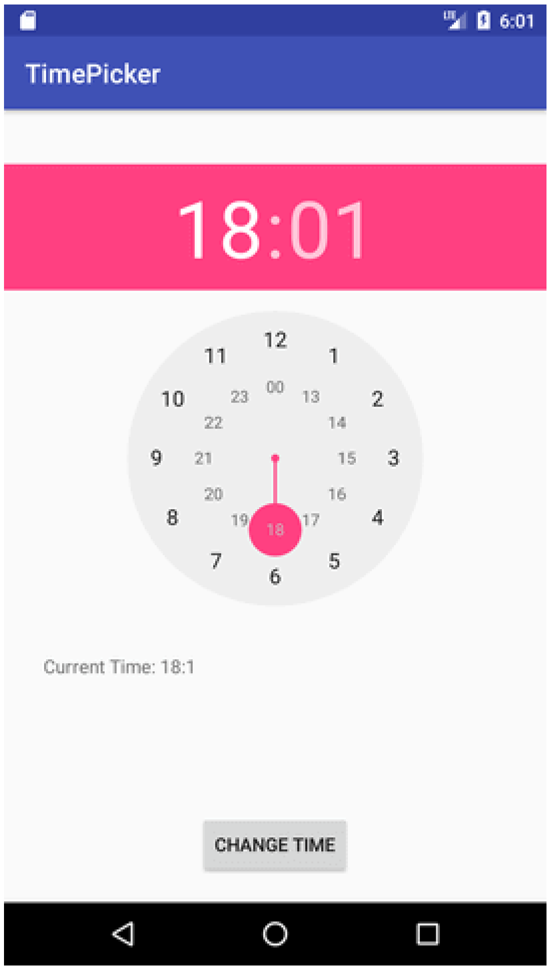
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="example.javatpoint.com.timepicker.MainActivity">

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/button1"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_marginBottom="102dp"
        android:layout_marginLeft="30dp"
        android:layout_marginStart="30dp"
        android:text="" />

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="20dp"
        android:text="Change Time" />

    <TimePicker
        android:id="@+id/timePicker"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_above="@+id/textView1"
        android:layout_centerHorizontal="true"
        android:layout_marginBottom="36dp" />
</RelativeLayout>
```

Activity class

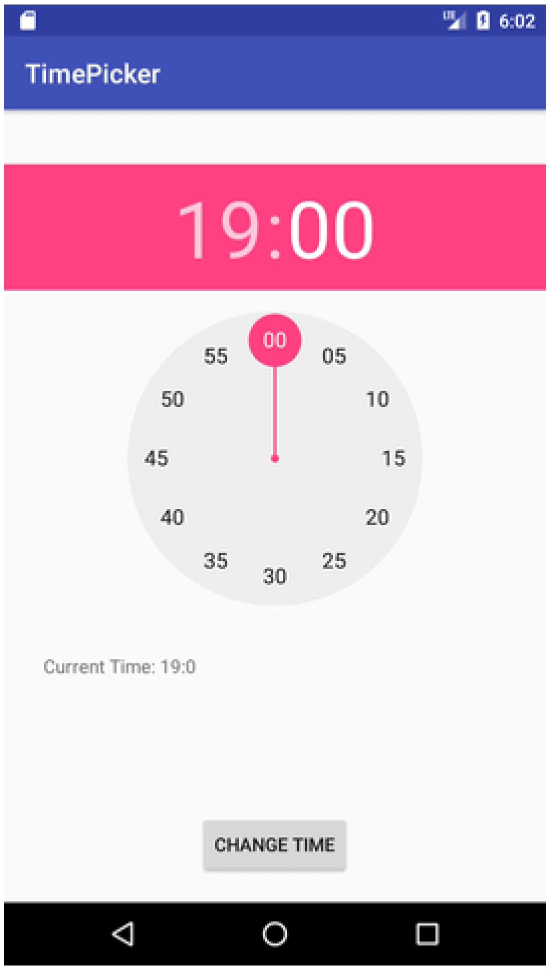
File: MainActivity.java

```
package example.javatpoint.com.timepicker;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.TimePicker;

public class MainActivity extends AppCompatActivity {
    TextView textview1;
    TimePicker timepicker;
    Button changetime;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        textview1=(TextView)findViewById(R.id.textView1);
        timepicker=(TimePicker)findViewById(R.id.timePicker);
        //Uncomment the below line of code for 24 hour view
        timepicker.setIs24HourView(true);
        changetime=(Button)findViewById(R.id.button1);

        textview1.setText(getCurrentTime());

        changetime.setOnClickListener(new View.OnClickListener(){
            @Override
            public void onClick(View view) {
                textview1.setText(getCurrentTime());
            }
        });

    }

    public String getCurrentTime(){
        String currentTime="Current Time: "+timepicker.getCurrentHour()+":"+timepick
er.getCurrentMinute();
        return currentTime;
    }

}
```

Output:



Current Time: 18:1



Current Time: 19:0

Android Analog clock and Digital clock

The **android.widget.AnalogClock** and **android.widget.DigitalClock** classes provides the functionality to display analog and digital clocks.

Android analog and digital clocks are used to show time in android application.

Android AnalogClock is the subclass of View class.

Android DigitalClock is the subclass of TextView class. Since Android API level 17, it is *deprecated*. You are recommended to use **TextClock** Instead.

The AnalogClock was deprecated in API level 23. This widget is no longer supported. Instead if you want to use AnalogClock in your application you need to hard code. It does not appear in API level 27 to drag from palette.

Note: Analog and Digital clocks cannot be used to change the time of the device. To do so, you need to use DatePicker and TimePicker.

In android, you need to drag analog and digital clocks from the pallet to display analog and digital clocks.

activity_main.xml

Now, drag the analog and digital clocks, now the xml file will look like this.

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.an
droid.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context="example.javatpoint.com.analogdigital.MainActivity">

   <AnalogClock
      android:id="@+id/analogClock1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_alignParentTop="true"
      android:layout_centerHorizontal="true"
```

```
        android:layout_marginLeft="136dp"
        android:layout_marginTop="296dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <DigitalClock
        android:id="@+id/digitalClock1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/analogClock1"
        android:layout_centerHorizontal="true"
        android:layout_marginLeft="176dp"
        android:layout_marginTop="84dp"
        android:text="DigitalClock"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>
```

Activity class

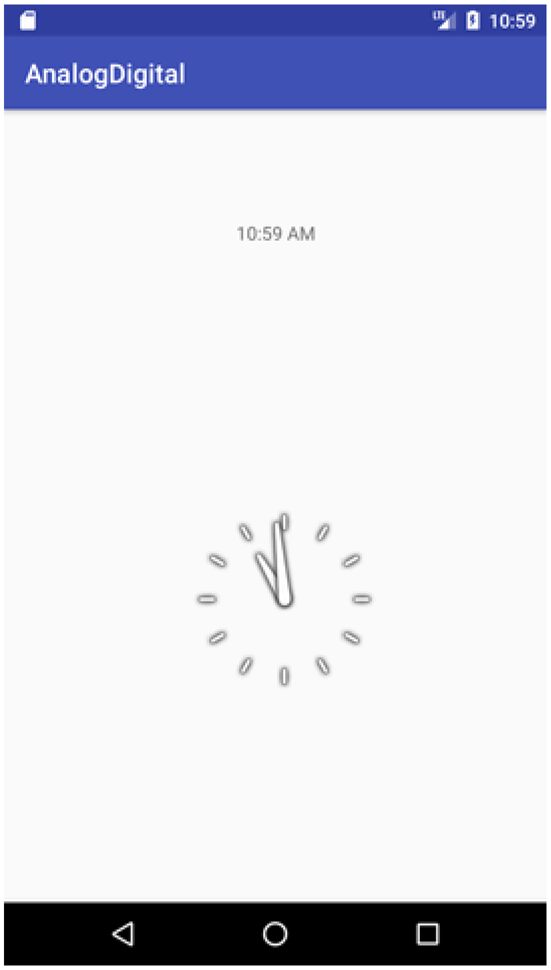
We have not write any code here.

File: MainActivity.java

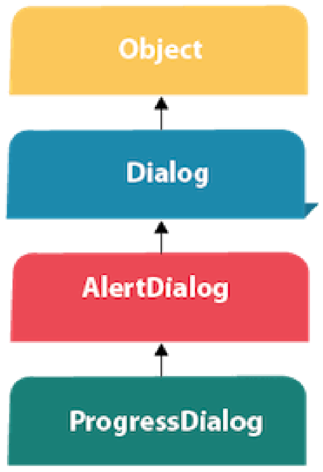
```
package example.javatpoint.com.analogdigital;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
            setContentView(R.layout.activity_main);
        }
    }
```

Output:



Android ProgressBar



We can display the **android progress bar** dialog box to display the status of work being done e.g. downloading file, analyzing status of work etc.

In this example, we are displaying the progress dialog for dummy file download operation.

Here we are using **android.app.ProgressDialog** class to show the progress bar. Android ProgressDialog is the subclass of AlertDialog class.

The **ProgressDialog** class provides methods to work on progress bar like setProgress(), setMessage(), setProgressStyle(), setMax(), show() etc. The progress range of Progress Dialog is 0 to 10000.

Let's see a simple example to display progress bar in android.

```
ProgressDialog progressBar = new ProgressDialog(this);
progressBar.setCancelable(true);//you can cancel it by pressing back button
progressBar.setMessage("File downloading ...");
progressBar.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
progressBar.setProgress(0);//initially progress is 0
progressBar.setMax(100);//sets the maximum value 100
progressBar.show();//displays the progress bar
```

Android Progress Bar Example by ProgressDialog

Let's see a simple example to create progress bar using ProgressDialog class.

activity_main.xml

Drag one button from the pallete, now the activity_main.xml file will look like this:

File: activity_main.xml

```
<RelativeLayout xmlns:androclass="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".MainActivity" >
```

```
    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="116dp"
        android:text="download file" />

</RelativeLayout>
```

Activity class

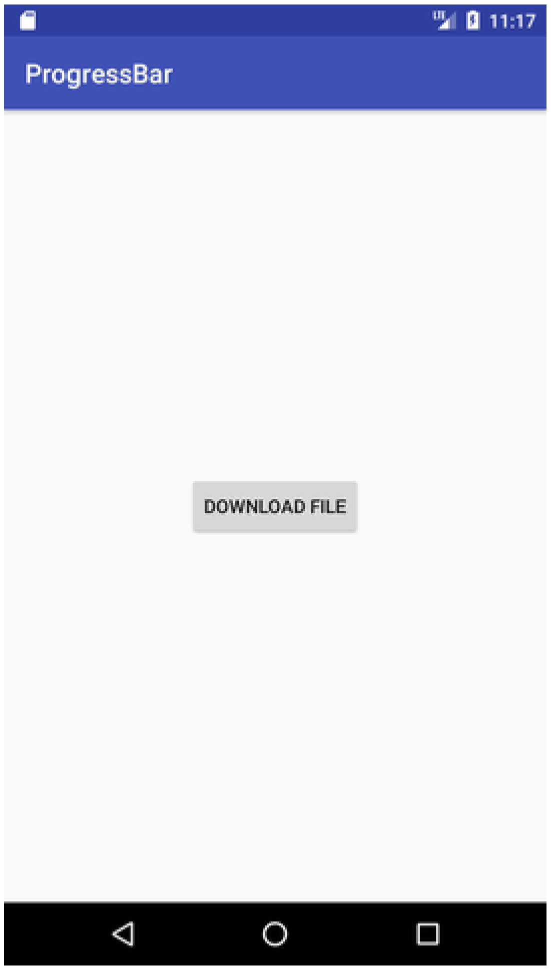
Let's write the code to display the progress bar dialog box.

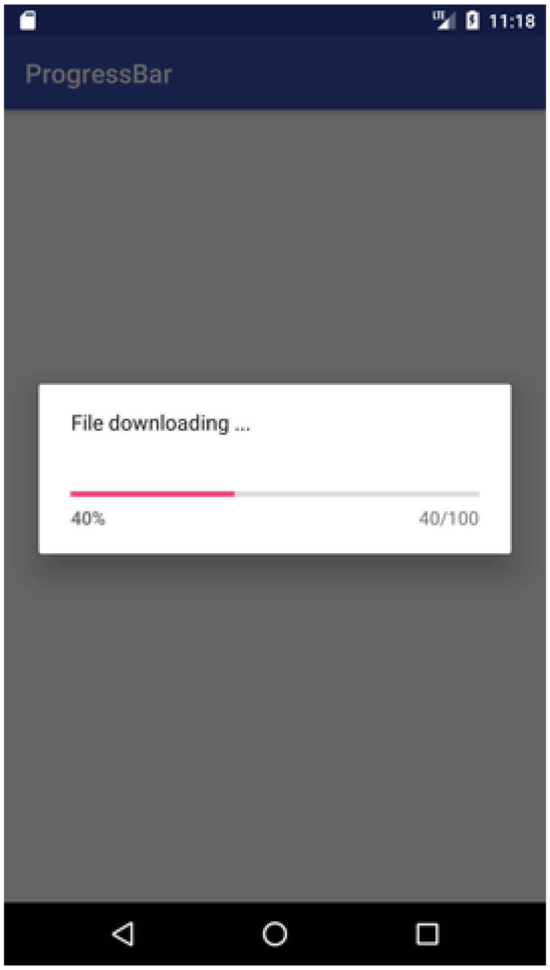
File: MainActivity.java

```
package example.javatpoint.com.progressbar;

import android.app.ProgressDialog;
import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity {
    Button btnStartProgress;
    ProgressDialog progressBar;
    private int progressBarStatus = 0;
    private Handler progressBarHandler = new Handler();
    private long fileSize = 0;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        addListenerOnButtonClick();
    }
    public void addListenerOnButtonClick() {
        btnStartProgress = findViewById(R.id.button);
        btnStartProgress.setOnClickListener(new View.OnClickListener(){

            @Override
            public void onClick(View v) {
                // creating progress bar dialog
                progressBar = new ProgressDialog(v.getContext());
                progressBar.setCancelable(true);
                progressBar.setMessage("File downloading ...");
                progressBar.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
                progressBar.setProgress(0);
                progressBar.setMax(100);
                progressBar.show();
                //reset progress bar and filesize status
                progressBarStatus = 0;
                fileSize = 0;

                new Thread(new Runnable() {
                    public void run() {
                        while (progressBarStatus < 100) {
                            // performing operation
                            progressBarStatus = doOperation();
                            try {
                                Thread.sleep(1000);
                            } catch (InterruptedException e) {
                                e.printStackTrace();
                            }
                            // Updating the progress bar
                            progressBarHandler.post(new Runnable() {
                                public void run() {
```

```
                        progressBar.setProgress(progressBarStatus);
                    }
                });
            }
            // performing operation if file is downloaded,
            if (progressBarStatus >= 100) {
                // sleeping for 1 second after operation completed
                try {
                    Thread.sleep(1000);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
                // close the progress bar dialog
                progressBar.dismiss();
            }
        }
    }).start();
}//end of onClick method
});
}
// checking how much file is downloaded and updating the filesize
public int doOperation() {
    //The range of ProgressDialog starts from 0 to 10000
    while (fileSize <= 10000) {
        fileSize++;
        if (fileSize == 1000) {
            return 10;
        } else if (fileSize == 2000) {
            return 20;
        } else if (fileSize == 3000) {
            return 30;
        } else if (fileSize == 4000) {
            return 40; // you can add more else if
        }
```

```
            /* else {
                return 100;
              }*/
          }//end of while
              return 100;
        }//end of doOperation
    }
```





File downloading ...

40%

40/100

2.3 Horizontal and Vertical ScrollView

Android ScrollView (Vertical)

The **android.widget.ScrollView** class provides the functionality of scroll view. ScrollView is used to scroll the child elements of palette inside ScrollView. Android supports vertical scroll view as default scroll view. Vertical ScrollView scrolls elements vertically.

Android uses *HorizontalScrollView* for horizontal ScrollView.

Let's implement simple example of vertical ScrollView.

activity_main.xml

Now, drag ScrollView from palette to activity_main.xml file and place some palette element inside it.

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingBottom="@dimen/activity_vertical_margin"
  android:paddingLeft="@dimen/activity_horizontal_margin"
  android:paddingRight="@dimen/activity_horizontal_margin"
  android:paddingTop="@dimen/activity_vertical_margin"
  tools:context="com.example.test.scrollviews.MainActivity">


  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="Vertical ScrollView example"
    android:id="@+id/textView"
    android:layout_gravity="center_horizontal"
    android:layout_centerHorizontal="true"
    android:layout_alignParentTop="true" />
```

```
<ScrollView android:layout_marginTop="30dp"
   android:layout_width="fill_parent"
   android:layout_height="wrap_content"
   android:id="@+id/scrollView">


   <LinearLayout
      android:layout_width="fill_parent"
      android:layout_height="fill_parent"
      android:orientation="vertical" >

      <Button
         android:layout_width="fill_parent"
         android:layout_height="wrap_content"
         android:text="Button 1" />
      <Button
         android:layout_width="fill_parent"
         android:layout_height="wrap_content"
         android:text="Button 2" />
      <Button
         android:layout_width="fill_parent"
         android:layout_height="wrap_content"
         android:text="Button 3" />
      <Button
         android:layout_width="fill_parent"
         android:layout_height="wrap_content"
         android:text="Button 4" />
      <Button
         android:layout_width="fill_parent"
         android:layout_height="wrap_content"
         android:text="Button 5" />
      <Button
```

```
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Button 6" />
<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Button 7" />
<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Button 8" />
<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Button 9" />
<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Button 10" />
<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Button 11" />
<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Button 12" />
<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Button 13" />
<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
```

```
            android:text="Button 14" />
        <Button
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="Button 15" />
        <Button
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="Button 16" />
        <Button
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="Button 17" />
        <Button
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="Button 18" />

        <Button
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="Button 19" />
        <Button
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="Button 20" />

      </LinearLayout>

    </ScrollView>

  </RelativeLayout>
```

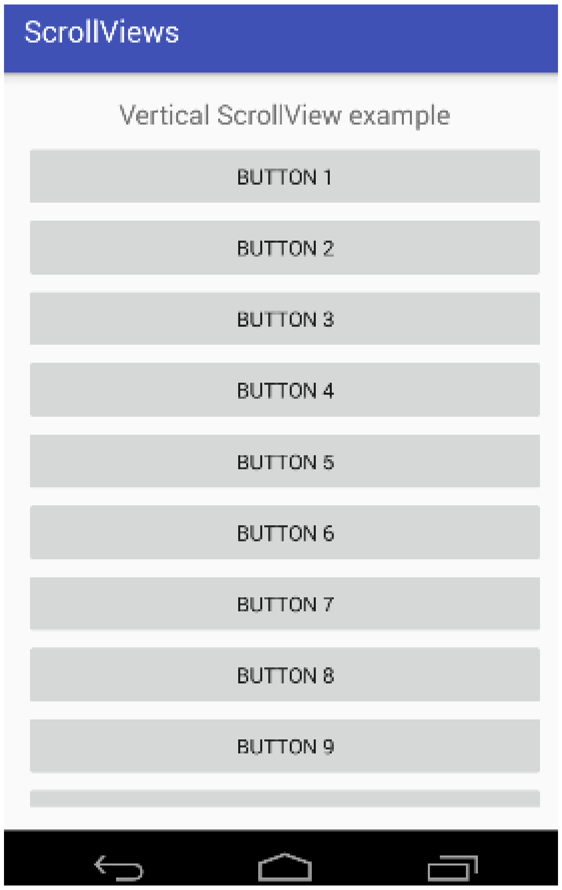
Activity class

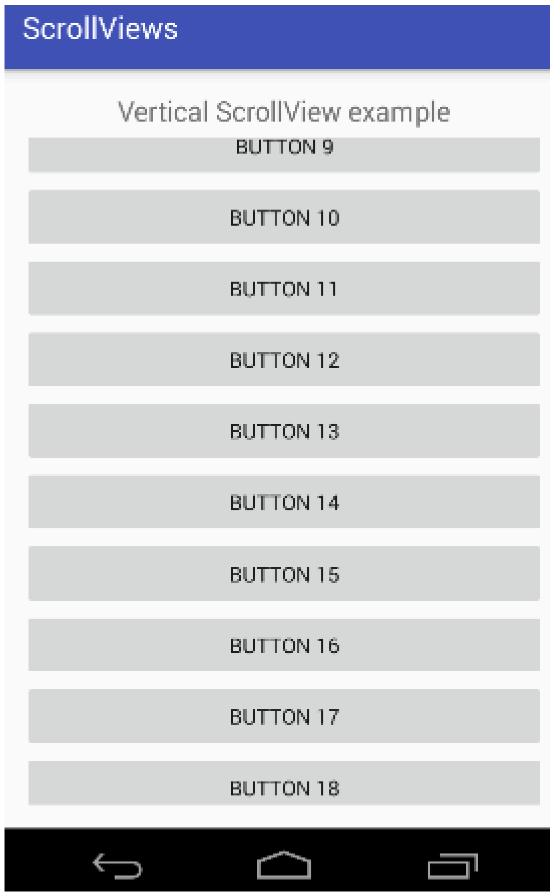
In activity class, we have not changed any code.

File: MainActivity.java

```
package com.example.test.scrollviews;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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

Output:


ScrollViews
Vertical ScrollView example
BUTTON 1
BUTTON 2
BUTTON 3
BUTTON 4
BUTTON 5
BUTTON 6
BUTTON 7
BUTTON 8
BUTTON 9


ScrollViews
Vertical ScrollView example
BUTTON 9
BUTTON 10
BUTTON 11
BUTTON 12
BUTTON 13
BUTTON 14
BUTTON 15
BUTTON 16
BUTTON 17
BUTTON 18

Android HorizontalScrollView

A **HorizontalScrollView** is a *FrameLayout*.
The **android.widget.HorizontalScrollView** class provides the functionality of horizontal scroll view. HorizontalScrollView is used to scroll the child elements or views in a horizontal direction. HorizontalScrollView only supports horizontal scrolling.

For vertical scroll, android uses *ScrollView*.

Let's implement simple example of *HorizontalScrollView*.

activity_main.xml

Now, drag HorizontalScrollView from palette to activity_main.xml file and place some views or elements inside it.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent">


  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceSmall"
    android:text="Horizontal ScrollView Example"
    android:id="@+id/textView"
    android:layout_alignParentTop="true"
    android:layout_centerHorizontal="true" />


  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="25dp">
    <HorizontalScrollView
      android:layout_width="match_parent"
```

```xml
android:layout_height="60dp"
android:id="@+id/horizontalScrollView">

<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New Button1"
        android:id="@+id/button1" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New Button2"
        android:id="@+id/button2" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New Button3"
        android:id="@+id/button3" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New Button4"
        android:id="@+id/button4" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New Button5"
        android:id="@+id/button5" />
    <Button
```

```
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="New Button6"
            android:id="@+id/button6" />
        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="New Button7"
            android:id="@+id/button7" />
        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="New Button8"
            android:id="@+id/button8"/>
    </LinearLayout>

  </HorizontalScrollView>
 </LinearLayout>
</RelativeLayout>
```

Activity class

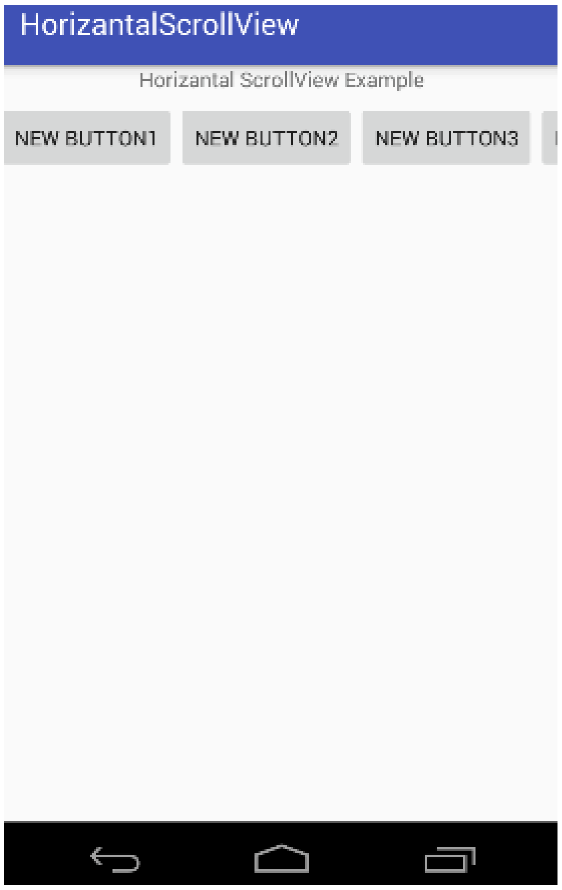
This is auto generated code, we have not written any code here.

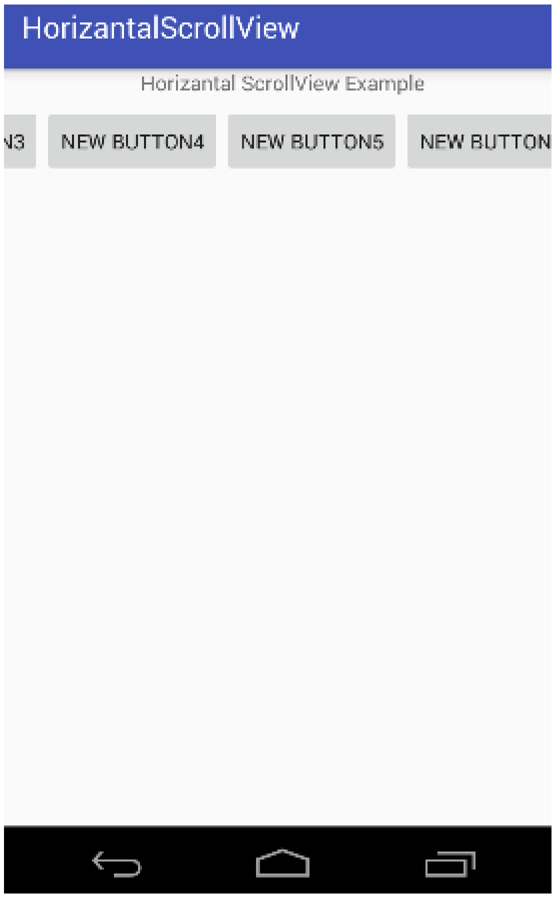
File: MainActivity.java

```
package com.example.test.horizantalscrollview;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
            setContentView(R.layout.activity_main);
        }
    }
```

Output:





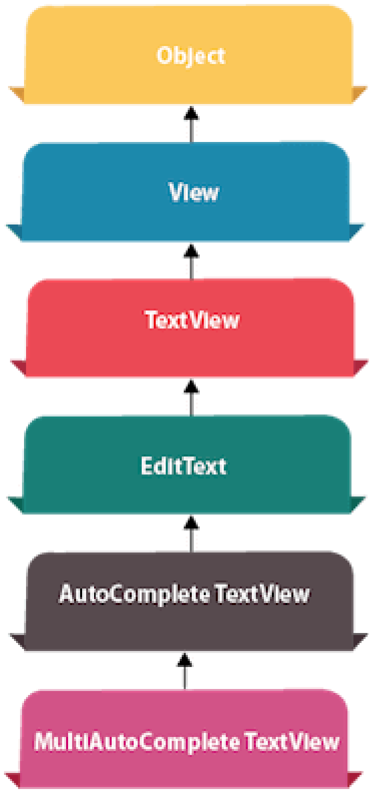
2.4 AutoCompleteTextView, TextWatcher to EditText

Android AutoCompleteTextView

Android AutoCompleteTextView completes the word based on the reserved words, so no need to write all the characters of the word.

Android AutoCompleteTextView is a editable text field, it displays a list of suggestions in a drop down menu from which user can select only one suggestion or value.

Android AutoCompleteTextView is the subclass of EditText class. The MultiAutoCompleteTextView is the subclass of AutoCompleteTextView class.



Android AutoCompleteTextView Example

In this example, we are displaying the programming languages in the autocompletetextview. All the programming languages are stored in string array. We are using the **ArrayAdapter** class to display the array content.

Let's see the simple example of autocompletetextview in android.

activity_main.xml

Drag the AutoCompleteTextView and TextView from the pallete, now the activity_main.xml file will like this:

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.an
droid.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="example.javatpoint.com.autocompletetextview.MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="What is your favourite programming language?"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.032" />

    <AutoCompleteTextView
        android:id="@+id/autoCompleteTextView"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:layout_marginLeft="92dp"
        android:layout_marginTop="144dp"
        android:text=""
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>
```

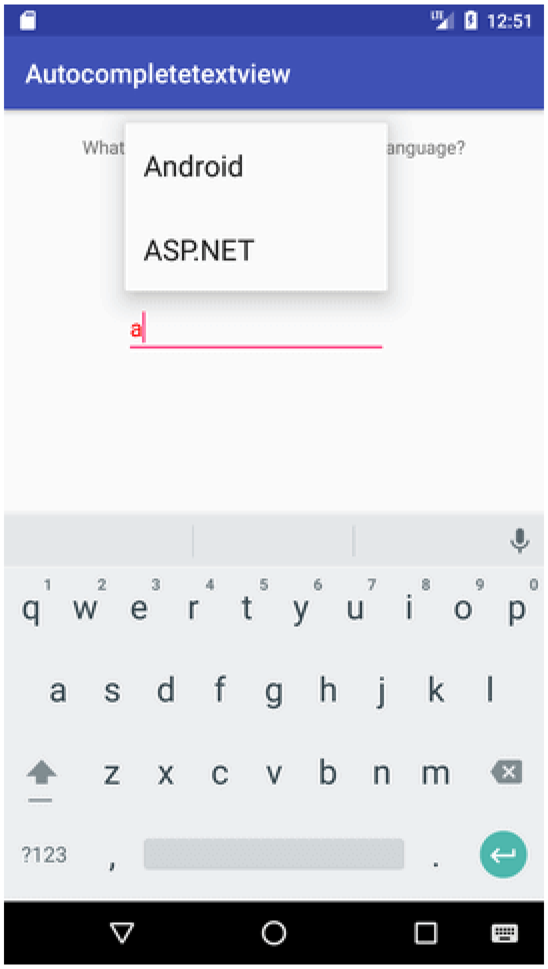
Activity class

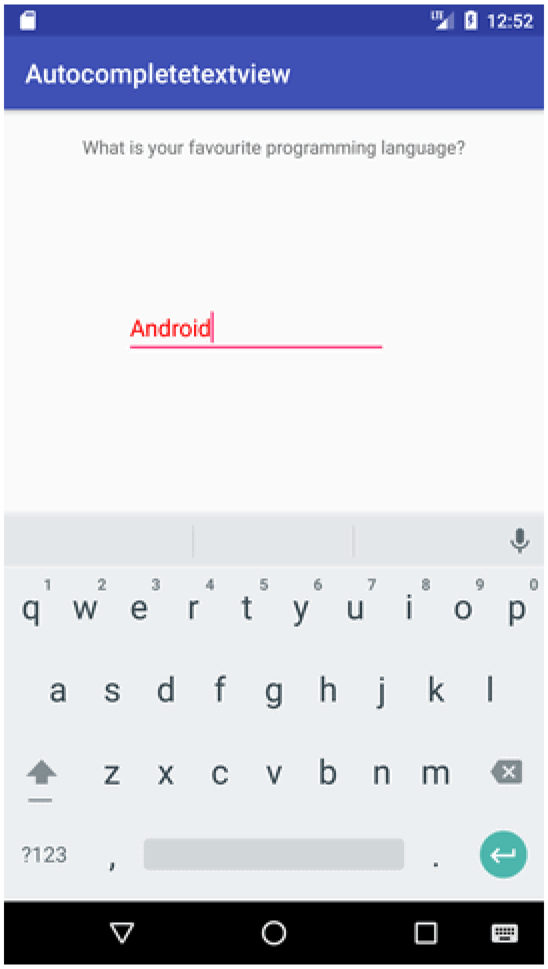
Let's write the code of AutoCompleteTextView.

File: MainActivity.java

```
1. package example.javatpoint.com.autocompletetextview;

import android.graphics.Color;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.AutoCompleteTextView;
```

```
public class MainActivity extends AppCompatActivity {
    String[] language ={"C","C++","Java",".NET","iPhone","Android","ASP.NET","PHP"};
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //Creating the instance of ArrayAdapter containing list of language names
        ArrayAdapter<String> adapter = new ArrayAdapter<String>
                (this,android.R.layout.select_dialog_item,language);
        //Getting the instance of AutoCompleteTextView
        AutoCompleteTextView actv =  (AutoCompleteTextView)findViewById(R.id.auto
CompleteTextView);
        actv.setThreshold(1);//will start working from first character
        actv.setAdapter(adapter);//setting the adapter data into the AutoCompleteText
View
        actv.setTextColor(Color.RED);
    }
}
```





What is your favourite programming language?

Android EditText with TextWatcher (Searching data from ListView)

Android **EditText** is a subclass of *TextView*. EditText is used for entering and modifying text. While using EditText width, we must specify its input type in *inputType* property of EditText which configures the keyboard according to input.

EditText uses **TextWatcher** interface to watch change made over EditText. For doing this, EditText calls the *addTextChangedListener()* method.

Methods of TextWatcher

1. **beforeTextChanged(CharSequence arg0, int arg1, int arg2, int arg3):** It is executed before making any change over EditText.
2. **onTextChanged(CharSequence cs, int arg1, int arg2, int arg3):** It is executed while making any change over EditText.
3. **afterTextChanged(Editable arg0):** It is executed after change made over EditText.

Example of EditText with TextWatcher()

In this example, we will implement EditText with TextWatcher to search data from ListView.

activity_main.xml

Create an activity_main.xml file in layout folder containing EditText and ListView.

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingBottom="@dimen/activity_vertical_margin"
  android:paddingLeft="@dimen/activity_horizontal_margin"
  android:paddingRight="@dimen/activity_horizontal_margin"
  android:paddingTop="@dimen/activity_vertical_margin"
  tools:context="com.example.test.searchfromlistview.MainActivity">
```

```
    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/editText"
        android:inputType="text"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true" />

    <ListView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/listView"
        android:layout_below="@+id/editText"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true" />
</RelativeLayout>
```

Create another file list_item.xml in layout folder which contains data of ListView.

list_item.xm

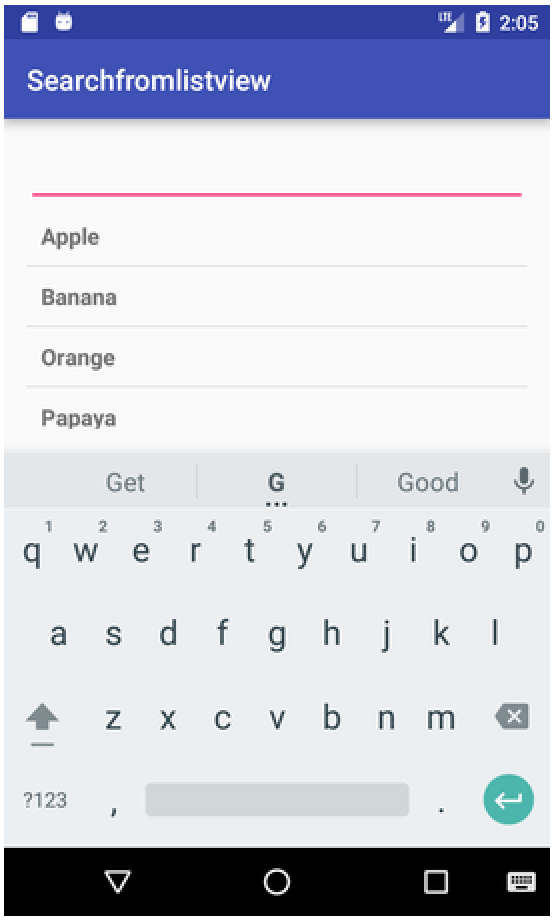
File: list_item.xml

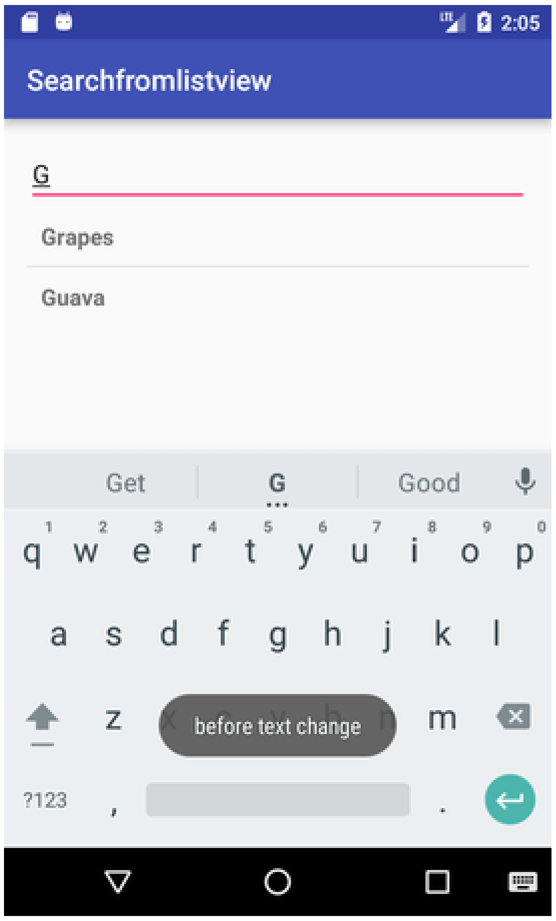
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

<TextView android:id="@+id/product_name"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:padding="10dip"
    android:textSize="16dip"
    android:textStyle="bold"/>
</LinearLayout>
```

Activity class

Activity class

```
package com.example.test.searchfromlistview;

import android.os.Bundle;
import android.text.Editable;
import android.text.TextWatcher;
import android.widget.ArrayAdapter;
import android.widget.EditText;
import android.widget.ListView;
import android.support.v7.app.AppCompatActivity;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    private ListView lv;
    private EditText editText;
    private ArrayAdapter<String> adapter;

    private String products[] = {"Apple", "Banana","Pinapple", "Orange", "Papaya", "M
elon",
          "Grapes", "Water Melon","Lychee", "Guava", "Mango", "Kivi"};
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        lv = (ListView) findViewById(R.id.listView);
        editText = (EditText) findViewById(R.id.editText);
        adapter = new ArrayAdapter<String>(this, R.layout.list_item, R.id.product_name
, products);
        lv.setAdapter(adapter);

        editText.addTextChangedListener(new TextWatcher() {

            @Override
            public void onTextChanged(CharSequence cs, int arg1, int arg2, int arg3) {
                adapter.getFilter().filter(cs);
            }

            @Override
            public void beforeTextChanged(CharSequence arg0, int arg1, int arg2, int ar
g3) {
                Toast.makeText(getApplicationContext(),"before text change",Toast.LENGT
H_LONG).show();
            }

            @Override
            public void afterTextChanged(Editable arg0) {
                Toast.makeText(getApplicationContext(),"after text change",Toast.LENGTH_
LONG).show();
            }
        });
    }
}
```





before text change

2.5 ImageSlider, ImageSwitcher, SearchView

Android Image Slider

Android *image slider* slides one entire screen to another screen. Image slider is created by **ViewPager** which is provided by support library. To implement image slider, you need to inherit ViewPager class which extends PagerAdapter.

Example of Image Slider

Let's see an example of android image slider.

activity_main.xml

In activity_main.xml file, we have wrapped ViewPager inside RelativeLayout.

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:paddingBottom="@dimen/activity_vertical_margin"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
   tools:context="com.example.test.imageslider.MainActivity">


   <android.support.v4.view.ViewPager
      android:id="@+id/viewPage"
      android:layout_width="fill_parent"
      android:layout_height="fill_parent" />

</RelativeLayout>
```

Activity class

File: MainActivity.java

```
package com.example.test.imageslider;

import android.support.v4.view.ViewPager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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

        ViewPager mViewPager = (ViewPager) findViewById(R.id.viewPage);
        ImageAdapter adapterView = new ImageAdapter(this);
        mViewPager.setAdapter(adapterView);
    }
}
```

ImageAdapter class

Now create ImageAdapter class which extends PagerAdapter for android image slider.

Place some images in drawable folder which are to be slid.

File: ImageAdapter.java

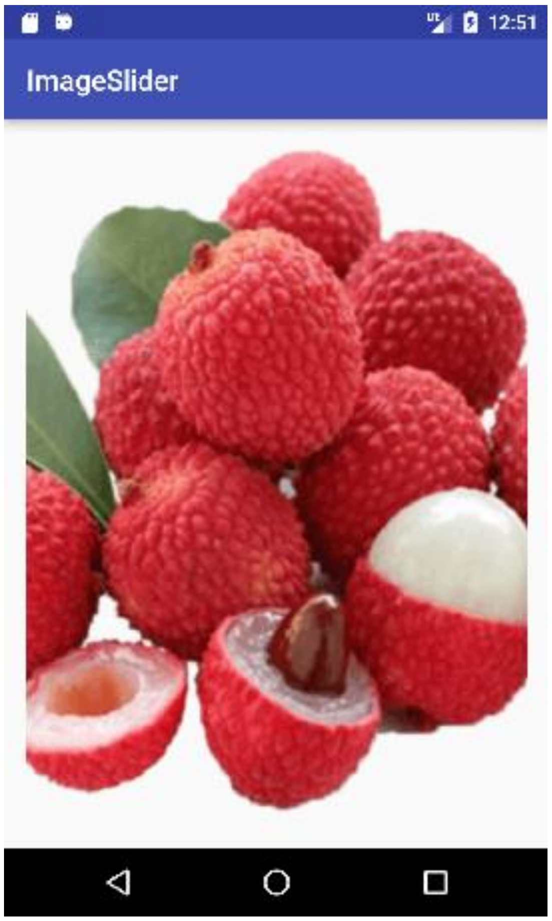
```
package com.example.test.imageslider;

import android.content.Context;
import android.support.v4.view.PagerAdapter;
import android.support.v4.view.ViewPager;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
```

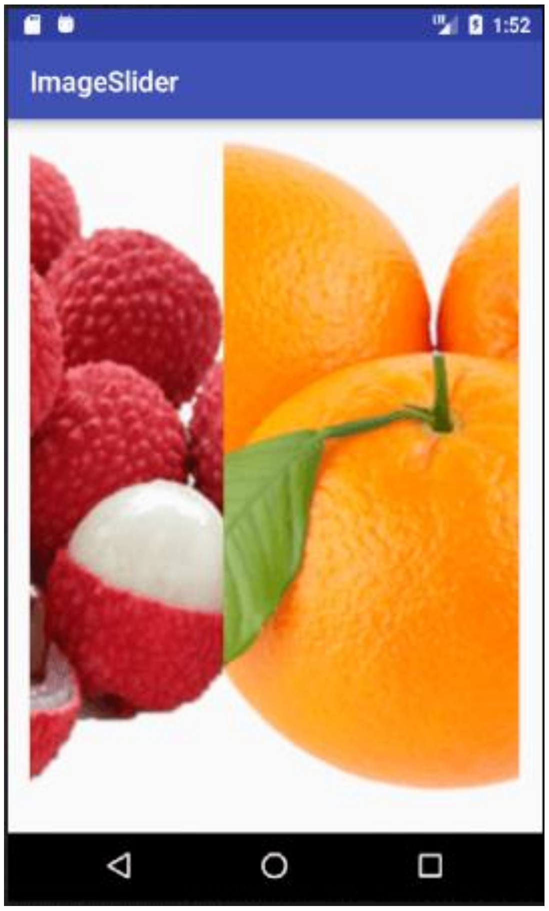
```
public class ImageAdapter extends PagerAdapter{
    Context mContext;

    ImageAdapter(Context context) {
        this.mContext = context;
    }

    @Override
    public boolean isViewFromObject(View view, Object object) {
        return view == ((ImageView) object);
    }

    private int[] sliderImageId = new int[]{
            R.drawable.image1, R.drawable.image2, R.drawable.image3,R.drawable.image
4, R.drawable.image5,
    };

    @Override
    public Object instantiateItem(ViewGroup container, int position) {
        ImageView imageView = new ImageView(mContext);
        imageView.setScaleType(ImageView.ScaleType.CENTER_CROP);
        imageView.setImageResource(sliderImageId[position]);
        ((ViewPager) container).addView(imageView, 0);
        return imageView;
    }

    @Override
    public void destroyItem(ViewGroup container, int position, Object object) {
        ((ViewPager) container).removeView((ImageView) object);
    }

    @Override
    public int getCount() {
        return sliderImageId.length;
        }
    }
```

We need to override following methods of PagerAdapter class.

1. **isViewFromObject(View, Object):** This method checks the view whether it is associated with key and returned by instantiateItem().
2. **instantiateItem(ViewGroup, int):** This method creates the page position passed as an argument.
3. **destroyItem(ViewGroup, int, Object):** It removes the page from its current position from container. In this example we simply removed object using removeView().
4. **getCount():** It returns the number of available views in ViewPager.

Output





Android Image Switcher

Android image switcher provides an animation over images to transition from one image to another. In order to use image switcher, we need to implement **ImageSwitcher** component in .xml file.

The *setFactory()* method of ImageSwitcher provide implementation of *ViewFactory* interface. ViewFactory interface implements its unimplemented method and returns an ImageView.

Example of Image Switcher

Let's implement an image switcher.

Create activity_main.xml and content_main.xml file in layout folder.

Place some images in drawable folder which are to be switch.

activity_main.xml

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.design.widget.CoordinatorLayout xmlns:android="http://sche
mas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:fitsSystemWindows="true"
   tools:context="com.example.test.imageswitcher.MainActivity">

   <android.support.design.widget.AppBarLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:theme="@style/AppTheme.AppBarOverlay">

      <android.support.v7.widget.Toolbar
         android:id="@+id/toolbar"
         android:layout_width="match_parent"
         android:layout_height="?attr/actionBarSize"
         android:background="?attr/colorPrimary"
```

```
        app:popupTheme="@style/AppTheme.PopupOverlay" />

    </android.support.design.widget.AppBarLayout>
    <include layout="@layout/content_main" />

</android.support.design.widget.CoordinatorLayout>
```

content_main.xml

File: content_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:context="com.example.test.imageswitcher.MainActivity"
    tools:showIn="@layout/activity_main">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Image Switcher Example"

        android:id="@+id/textView"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true" />

    <ImageSwitcher
```

```
        android:id="@+id/imageSwitcher"
        android:layout_width="match_parent"
        android:layout_height="250dp"
        android:layout_marginBottom="28dp"
        android:layout_marginTop="40dp" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Next"
        android:id="@+id/button"
        android:layout_marginBottom="47dp"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true" />
</RelativeLayout>
```

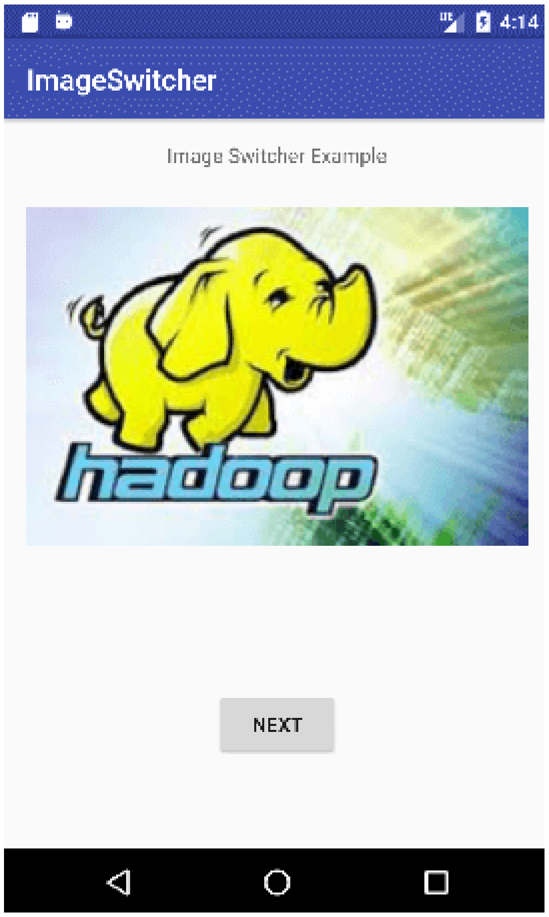
Activity class

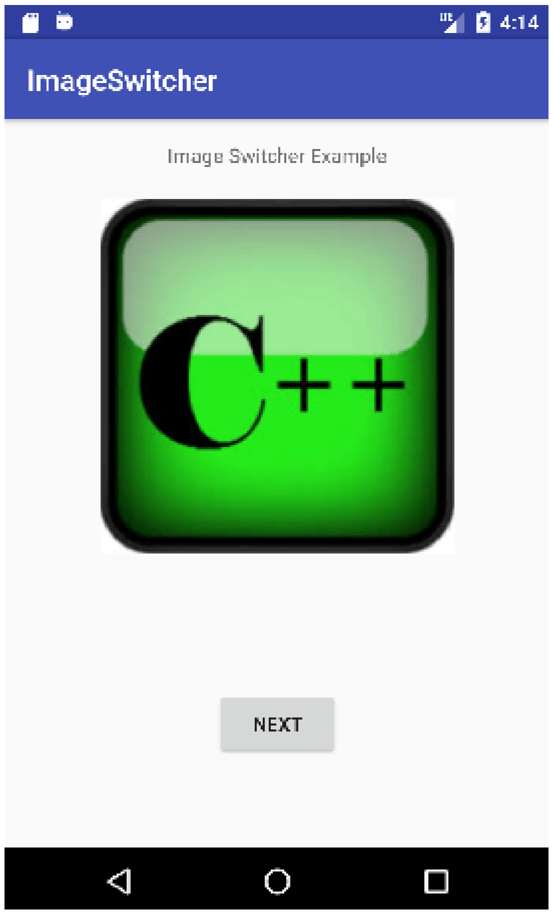
File: MainActivity.java

```
package com.example.test.imageswitcher;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.widget.Button;
import android.widget.ImageSwitcher;
import android.widget.ImageView;
import android.widget.ViewSwitcher;

import android.app.ActionBar;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
```

```
public class MainActivity extends AppCompatActivity {
    ImageSwitcher imageSwitcher;
    Button nextButton;

    int imageSwitcherImages[] =
      {R.drawable.cpp, R.drawable.c_sarp, R.drawable.jsp, R.drawable.mysql, R.drawable.
hadoop};

    int switcherImageLength = imageSwitcherImages.length;
    int counter = -1;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        imageSwitcher = (ImageSwitcher) findViewById(R.id.imageSwitcher);
        nextButton = (Button) findViewById(R.id.button);

        imageSwitcher.setFactory(new ViewSwitcher.ViewFactory() {
            @Override
            public View makeView() {
                ImageView switcherImageView = new ImageView(getApplicationContext())
;
                switcherImageView.setLayoutParams(new ImageSwitcher.LayoutParams(
                        ActionBar.LayoutParams.FILL_PARENT, ActionBar.LayoutParams.FILL_P
ARENT
                ));
                switcherImageView.setScaleType(ImageView.ScaleType.FIT_CENTER);
                switcherImageView.setImageResource(R.drawable.hadoop);
                //switcherImageView.setMaxHeight(100);
                return switcherImageView;
            }
        });

        Animation aniOut = AnimationUtils.loadAnimation(this, android.R.anim.slide_ou
t_right);
        Animation aniIn = AnimationUtils.loadAnimation(this, android.R.anim.slide_in_le
ft);

       imageSwitcher.setOutAnimation(aniOut);
       imageSwitcher.setInAnimation(aniIn);

        nextButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                counter++;
                if (counter == switcherImageLength){
                    counter = 0;
                    imageSwitcher.setImageResource(imageSwitcherImages[counter]);
                }
                else{
                    imageSwitcher.setImageResource(imageSwitcherImages[counter]);
                }
            }
        });
    }

}
```

Output:


4:14
ImageSwitcher
Image Switcher Example
hadoop
NEXT


4:14
ImageSwitcher
Image Switcher Example
C++
NEXT

Android SearchView

Android **SearchView** provides user interface to search query submitted over search provider. SearchView widget can be implemented over ToolBar/ActionBar or inside a layout.

SearchView is by default collapsible and set to be iconified using *setIconifiedByDefault(true)* method of SearchView class. For making search field visible, SearchView uses setIconifiedByDefault(false) method.

Methods of SearchView

1. **public boolean onQueryTextSubmit(String query):** It searches the query on the submission of content over SearchView editor. It is case dependent.
2. **public boolean onQueryTextChange(String newText):** It searches the query at the time of text change over SearchView editor.

Example of SearchView

Let's see the example of SearchView over layout, searching data in a ListView.

activity_main.xml

Create an activity_main.xml file in layout folder containing ScrollView and ListView.

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingBottom="@dimen/activity_vertical_margin"
  android:paddingLeft="@dimen/activity_horizontal_margin"
  android:paddingRight="@dimen/activity_horizontal_margin"
  android:paddingTop="@dimen/activity_vertical_margin"
  tools:context="com.example.test.searchview.MainActivity">

  <ListView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
```

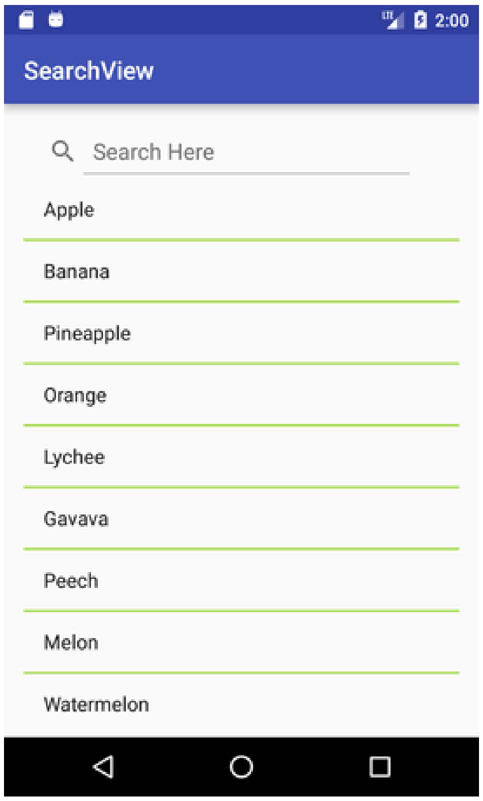
```
        android:id="@+id/lv1"
        android:divider="#ad5"
        android:dividerHeight="2dp"
        android:layout_below="@+id/searchView"/>

    <SearchView
        android:id="@+id/searchView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:queryHint="Search Here"
        android:iconifiedByDefault="false"
        android:layout_alignParentTop="true"
    />

</RelativeLayout>
```

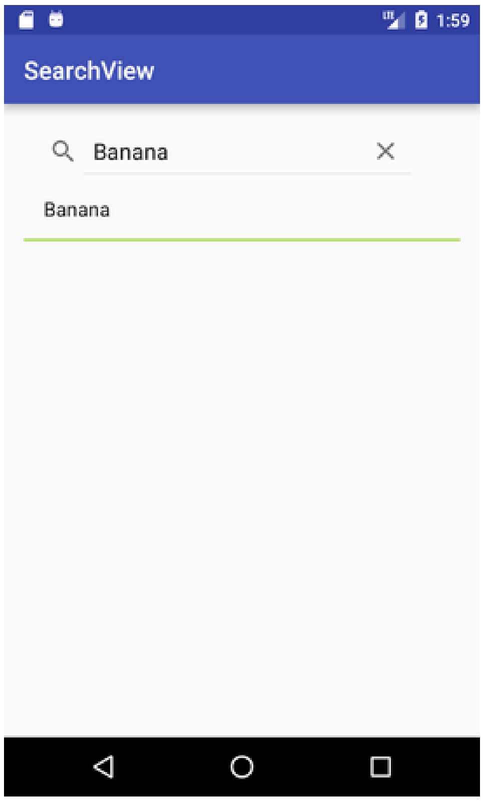
Activity class

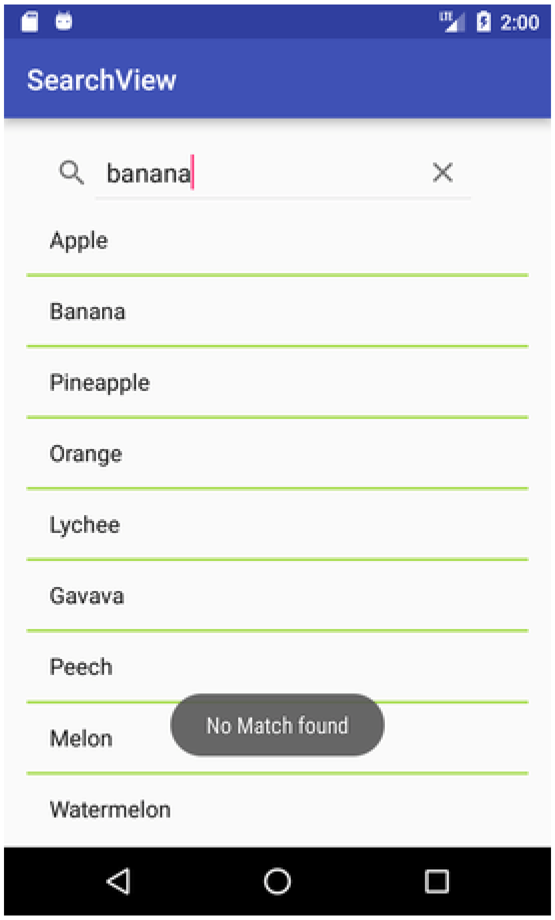
File: MainActivity.java

```
package com.example.test.searchview;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.ArrayAdapter;
import android.widget.Filter;
import android.widget.ListView;
import android.widget.SearchView;
import android.widget.Toast;

import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {
    SearchView searchView;
    ListView listView;
    ArrayList<String> list;
    ArrayAdapter<String > adapter;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        searchView = (SearchView) findViewById(R.id.searchView);
        listView = (ListView) findViewById(R.id.lv1);

        list = new ArrayList<>();
        list.add("Apple");
        list.add("Banana");
        list.add("Pineapple");
        list.add("Orange");
        list.add("Lychee");
        list.add("Gavava");
        list.add("Peech");
        list.add("Melon");
        list.add("Watermelon");
        list.add("Papaya");

        adapter = new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1,li
st);
        listView.setAdapter(adapter);

        searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
            @Override
            public boolean onQueryTextSubmit(String query) {

                if(list.contains(query)){
                    adapter.getFilter().filter(query);
                }else{
                    Toast.makeText(MainActivity.this, "No Match found",Toast.LENGTH_LON
G).show();
                }
                return false;
            }

            @Override
            public boolean onQueryTextChange(String newText) {
            //    adapter.getFilter().filter(newText);
                return false;
            }
        });
    }
}
```

Output:







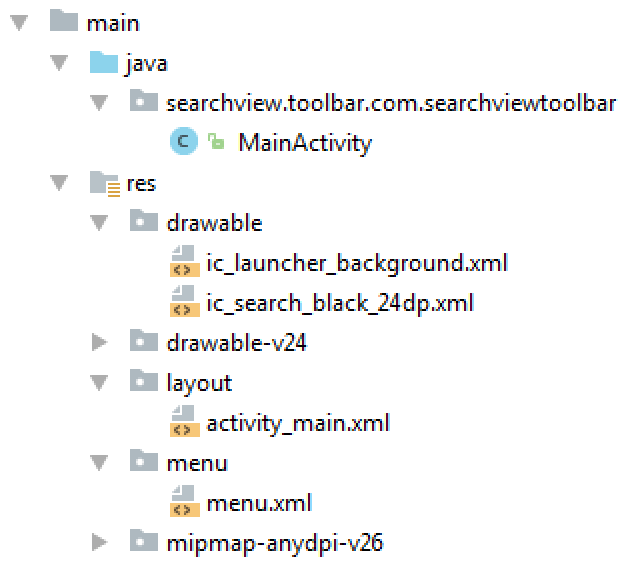
<u>Android SearchView on ToolBar</u>

As we have already implemented SearchView widget over activity layout, it can also be implemented over ToolBar/ActionBar. For implementing SearchView over ToolBar, we need to create menu option and place SearchView widget on it.

Example of SearchView on ToolBar (ActionBar)

Let's us see the example of SearchView over ToolBar and searching data in ListView.

Directory Structure of this Example



```
main
  java
    searchview.toolbar.com.searchviewtoolbar
      MainActivity
  res
    drawable
      ic_launcher_background.xml
      ic_search_black_24dp.xml
    drawable-v24
    layout
      activity_main.xml
    menu
      menu.xml
    mipmap-anydpi-v26
```

activity_main.xml

Create an activity_main.xml file in layout folder containing ListView.

File: activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.an
droid.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="searchview.toolbar.com.searchviewtoolbar.MainActivity">

    <ListView
        android:id="@+id/listView"
        android:layout_width="match_parent"
        android:layout_height="fill_parent"
        />

</android.support.constraint.ConstraintLayout>
```

menu.xml

Create a menu.xml file in menu folder and place the following code. This code places the SearchView widget over ToolBar.

File: menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:android="http://schemas.android.com/apk/res/android">

<item
    android:id="@+id/app_bar_search"
    android:icon="@drawable/ic_search_black_24dp"
    android:title="Search"
    app:showAsAction="ifRoom|withText"
    app:actionViewClass="android.widget.SearchView"/>
</menu>
```

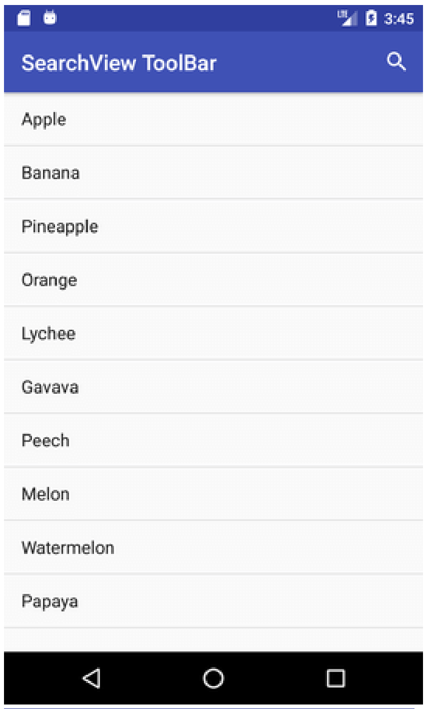
Activity class

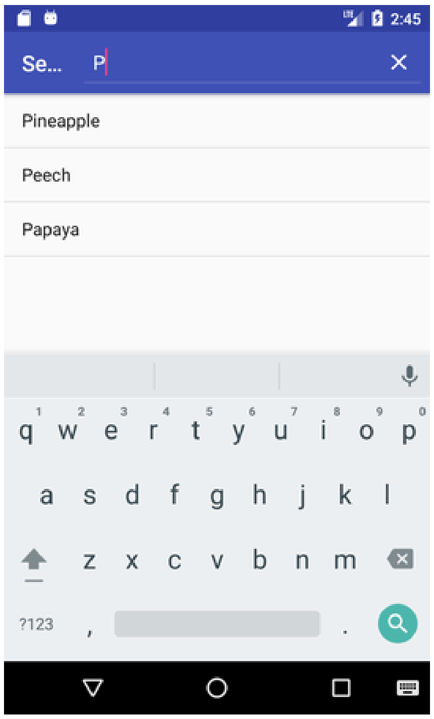
File: MainActivity.java

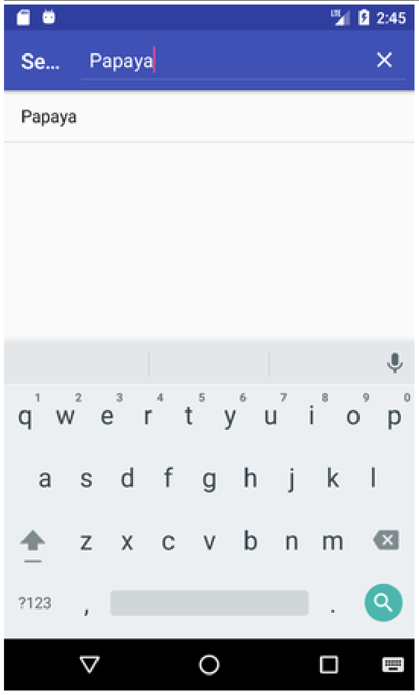
```
package searchview.toolbar.com.searchviewtoolbar;

import android.support.v4.view.MenuItemCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.SearchView;
import android.widget.Toast;

import java.util.ArrayList;
```

```
public class MainActivity extends AppCompatActivity {

    ListView listView;
    ArrayList<String> list;
    ArrayAdapter<String > adapter;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        listView = (ListView) findViewById(R.id.listView);

        list = new ArrayList<>();
        list.add("Apple");
        list.add("Banana");
        list.add("Pineapple");
        list.add("Orange");
        list.add("Lychee");
        list.add("Gavava");
        list.add("Peech");
        list.add("Melon");
        list.add("Watermelon");
        list.add("Papaya");

        adapter = new ArrayAdapter<String>(this, android.R.layout.simple_list_item_1,li
st);
        listView.setAdapter(adapter);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menu, menu);
        MenuItem searchViewItem = menu.findItem(R.id.app_bar_search);
        final SearchView searchView = (SearchView) MenuItemCompat.getActionView(s
earchViewItem);
        searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
            @Override
            public boolean onQueryTextSubmit(String query) {
                searchView.clearFocus();
             /*   if(list.contains(query)){
                    adapter.getFilter().filter(query);
                }else{
                    Toast.makeText(MainActivity.this, "No Match found",Toast.LENGTH_LON
G).show();
                }*/
                return false;

            }

            @Override
            public boolean onQueryTextChange(String newText) {
                adapter.getFilter().filter(newText);
                return false;
            }
        });
        return super.onCreateOptionsMenu(menu);
    }
}
```

Output:


3:45
SearchView ToolBar
Apple
Banana
Pineapple
Orange
Lychee
Gavava
Peech
Melon
Watermelon
Papaya


2:45
Se...
P
Pineapple
Peech
Papaya


2:45
Se...
Papaya
Papaya

2.6 TAbLayout and FrameLayout

Android TabLayout

TabLayout is used to implement horizontal tabs. TabLayout is released by Android after the deprecation of *ActionBar.TabListener (API level 21)*.

TabLayout is introduced in design support library to implement tabs.

Tabs are created using *newTab()* method of TabLayout class. The title and icon of Tabs are set through *setText(int)* and *setIcon(int)* methods of TabListener interface respectively. Tabs of layout are attached over TabLayout using the method addTab(Tab) method.

```
TabLayout tabLayout = (TabLayout)findViewById(R.id.tabLayout);
tabLayout.addTab(tabLayout.newTab().setText("Tab 1"));
tabLayout.addTab(tabLayout.newTab().setText("Tab 2"));
tabLayout.addTab(tabLayout.newTab().setText("Tab 3"));
```

We can also add tab item to TabLayout using TabItem of android design widget.

```
<android.support.design.widget.TabItem
            android:text="@string/tab_text"/>
```

Example of TabLayout using ViewPager

Let's create an example of TabLayout using ViewPager and Fragment.

File: activity.xml

Create an activity.xml file with TabLayout and ViewPager view components.

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.an
droid.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context="tablayout.example.com.tablayout.MainActivity">
```

```
    <android.support.design.widget.TabLayout
        android:id="@+id/tabLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#1db995">
    </android.support.design.widget.TabLayout>

    <android.support.v4.view.ViewPager
        android:id="@+id/viewPager"
        android:layout_width="355dp"
        android:layout_height="455dp"
        app:layout_constraintTop_toBottomOf="@+id/tabLayout"
        tools:layout_editor_absoluteX="8dp" />


</android.support.constraint.ConstraintLayout>
```

File: build.gradle

Now gave the dependency library of TabLayout in build.gradle file.

```
implementation 'com.android.support:design:26.1.0'
```

File: MainActivity.java

In this file, we implement two additional listener *addOnPageChangeListener(listener)* of ViewPager which makes slides the different fragments of tabs and *addOnTabSelectedListener(listener)* of TabLayout which select the current tab on tab selection.

```
package tablayout.example.com.tablayout;

import android.support.design.widget.TabLayout;
import android.support.v4.view.ViewPager;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
```

```
public class MainActivity extends AppCompatActivity {
    TabLayout tabLayout;
    ViewPager viewPager;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        tabLayout=(TabLayout)findViewById(R.id.tabLayout);
        viewPager=(ViewPager)findViewById(R.id.viewPager);

        tabLayout.addTab(tabLayout.newTab().setText("Home"));
        tabLayout.addTab(tabLayout.newTab().setText("Sport"));
        tabLayout.addTab(tabLayout.newTab().setText("Movie"));
        tabLayout.setTabGravity(TabLayout.GRAVITY_FILL);

        final MyAdapter adapter = new MyAdapter(this,getSupportFragmentManager(
), tabLayout.getTabCount());
        viewPager.setAdapter(adapter);

        viewPager.addOnPageChangeListener(new TabLayout.TabLayoutOnPageChang
eListener(tabLayout));

         tabLayout.addOnTabSelectedListener(new TabLayout.OnTabSelectedListener() {

            @Override
            public void onTabSelected(TabLayout.Tab tab) {
                viewPager.setCurrentItem(tab.getPosition());
            }

            @Override
            public void onTabUnselected(TabLayout.Tab tab) {

            }
            @Override
            public void onTabReselected(TabLayout.Tab tab) {

            }
        });

    }
}
```

File: MyAdapter.java

```
package tablayout.example.com.tablayout;

import android.content.Context;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentPagerAdapter;
import android.support.v4.app.FragmentManager;

public class MyAdapter extends FragmentPagerAdapter {

    private Context myContext;
    int totalTabs;

    public MyAdapter(Context context, FragmentManager fm, int totalTabs) {
        super(fm);
        myContext = context;
        this.totalTabs = totalTabs;
    }

    // this is for fragment tabs
    @Override
    public Fragment getItem(int position) {
        switch (position) {
            case 0:
                HomeFragment homeFragment = new HomeFragment();
                return homeFragment;
            case 1:
                SportFragment sportFragment = new SportFragment();
                return sportFragment;
            case 2:
                MovieFragment movieFragment = new MovieFragment();
                return movieFragment;
            default:
                return null;
        }
    }
// this counts total number of tabs
    @Override
    public int getCount() {
        return totalTabs;
    }
}
```

Now create different fragment files for all different tabs.

File: HomeFragment.java

```
package tablayout.example.com.tablayout;

import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

public class HomeFragment extends Fragment {


    public HomeFragment() {
        // Required empty public constructor
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_home, container, false);
    }

}
```

File: fragment_home.xml

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="tablayout.example.com.tablayout.HomeFragment">

    <!-- TODO: Update blank fragment layout -->

    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center"
        android:text="@string/home_fragment" />

</FrameLayout>
```

File: SportFragment.java

```
package tablayout.example.com.tablayout;
```

```java
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

public class SportFragment extends Fragment {


    public SportFragment() {
        // Required empty public constructor
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_sport, container, false);
    }

}
```

File: fragment_sport.xml

```xml
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="tablayout.example.com.tablayout.SportFragment">

    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center"
        android:text="@string/sport_fragment" />

</FrameLayout>
```

File: MovieFragment.java

```
package tablayout.example.com.tablayout;

import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

public class MovieFragment extends Fragment {


    public MovieFragment() {
        // Required empty public constructor
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_movie, container, false);
    }

}
```

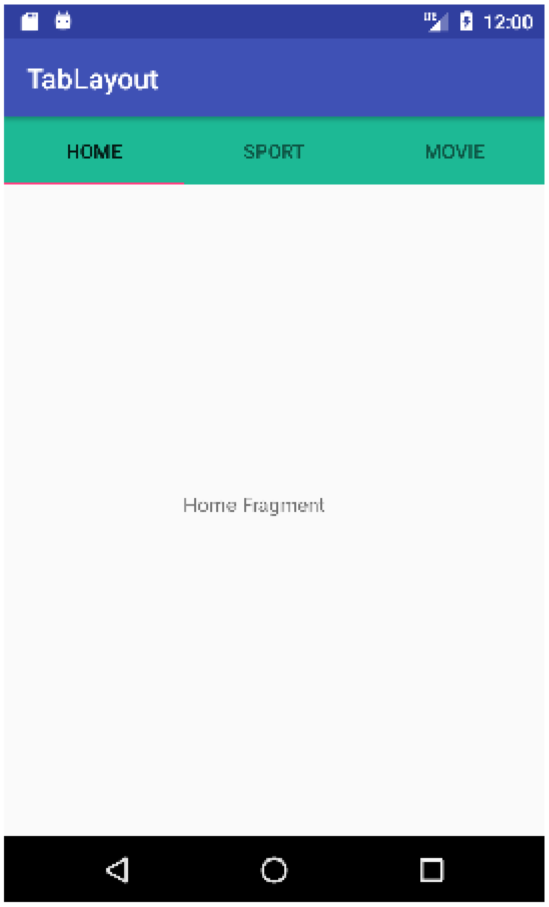
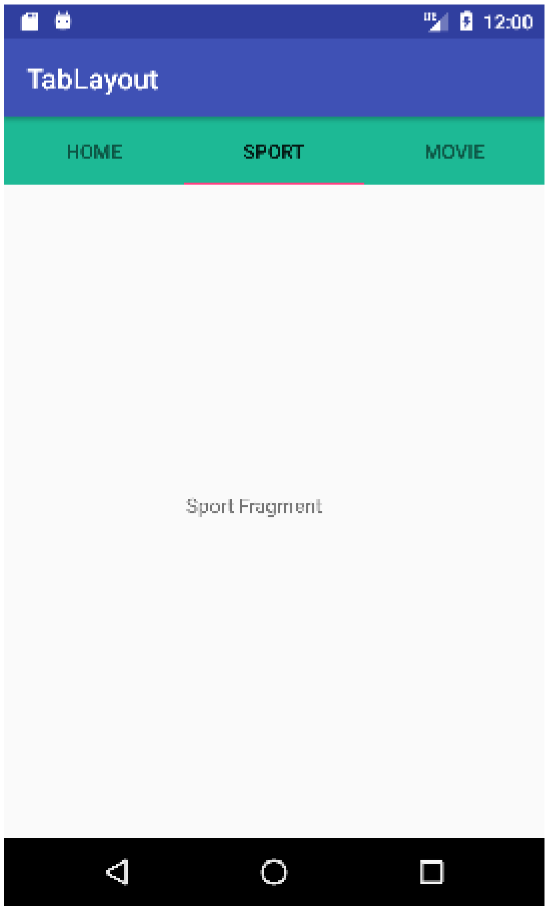
File: fragment_movie.xml

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

```
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="tablayout.example.com.tablayout.MovieFragment">

    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center"
        android:text="@string/movie_fragment" />

</FrameLayout>
```

File: strings.xml

```
<resources>
    <string name="app_name">TabLayout</string>

    <!-- TODO: Remove or change this placeholder text -->
    <string name="home_fragment">Home Fragment</string>
    <string name="sport_fragment">Sport Fragment</string>
    <string name="movie_fragment">Movie Fragment</string>

</resources>
```


12:00
TabLayout
HOME
SPORT
MOVIE
Home Fragment

12:00
TabLayout
HOME
SPORT
MOVIE
Sport Fragment

Android TabLayout with FrameLayout

In the previous page, we created a sliding tabs using TabLayout and ViewPager. Here, we are going to create non sliding tabs using *TabLayout* and *FrameLayout*.

Items of TabLayout are implemented by adding *TabItem* of android support design widget.

Example of TabLayout using FrameLayout

Let's create an example of TabLayout using FrameLayout and Fragment.

File: activity.xml

Create an activity.xml file with TabLayout and FrameLayout view components.

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.an
droid.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context="tablayout.example.com.tablayoutwithframelayout.MainActivity">


   <android.support.design.widget.TabLayout
      android:id="@+id/tabLayout"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:background="#7367">

      <android.support.design.widget.TabItem
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Home" />

      <android.support.design.widget.TabItem
         android:layout_width="wrap_content"
```

```
            android:layout_height="wrap_content"
            android:text="Java" />

        <android.support.design.widget.TabItem
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Android" />

        <android.support.design.widget.TabItem
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Php" />
    </android.support.design.widget.TabLayout>

    <FrameLayout
        android:id="@+id/frameLayout"
        android:layout_width="match_parent"
        android:layout_height="455dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/tabLayout">

    </FrameLayout>
</android.support.constraint.ConstraintLayout>
```

File: build.gradle

Now gave the dependency library of TabLayout in build.gradle file.

```
implementation 'com.android.support:design:26.1.0'
```

File: MainActivity.java

```
package tablayout.example.com.tablayoutwithframelayout;


import android.support.design.widget.TabLayout;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentManager;
import android.support.v4.app.FragmentTransaction;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.FrameLayout;

public class MainActivity extends AppCompatActivity {
   TabLayout tabLayout;
   FrameLayout frameLayout;
   Fragment fragment = null;
   FragmentManager fragmentManager;
   FragmentTransaction fragmentTransaction;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

      tabLayout=(TabLayout)findViewById(R.id.tabLayout);
      frameLayout=(FrameLayout)findViewById(R.id.frameLayout);

      fragment = new HomeFragment();
      fragmentManager = getSupportFragmentManager();
      fragmentTransaction = fragmentManager.beginTransaction();
      fragmentTransaction.replace(R.id.frameLayout, fragment);
      fragmentTransaction.setTransition(FragmentTransaction.TRANSIT_FRAGMENT_O
PEN);
      fragmentTransaction.commit();

      tabLayout.addOnTabSelectedListener(new TabLayout.OnTabSelectedListener() {

         @Override
         public void onTabSelected(TabLayout.Tab tab) {
           // Fragment fragment = null;
```

```
            switch (tab.getPosition()) {
                case 0:
                    fragment = new HomeFragment();
                    break;
                case 1:
                    fragment = new JavaFragment();
                    break;
                case 2:
                    fragment = new AndroidFragment();
                    break;
                case 3:
                    fragment = new PhpFragment();
                    break;
            }
            FragmentManager fm = getSupportFragmentManager();
            FragmentTransaction ft = fm.beginTransaction();
            ft.replace(R.id.frameLayout, fragment);
            ft.setTransition(FragmentTransaction.TRANSIT_FRAGMENT_OPEN);
            ft.commit();
        }

        @Override
        public void onTabUnselected(TabLayout.Tab tab) {

        }

        @Override
        public void onTabReselected(TabLayout.Tab tab) {

        }
    });

  }
}
```

Now create different fragment files for all different tabs.

File: HomeFragment.java

```
package tablayout.example.com.tablayoutwithframelayout;

import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

public class HomeFragment extends Fragment {

    public HomeFragment() {
        // Required empty public constructor
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_home, container, false);
    }

}
```

File: fragment_home.xml

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="tablayout.example.com.tablayoutwithframelayout.HomeFragment"
>
```

```
    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center"
        android:text="@string/home_fragment" />

</FrameLayout>
```

File: JavaFragment.java

```
package tablayout.example.com.tablayoutwithframelayout;

import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

public class JavaFragment extends Fragment {

    public JavaFragment() {
        // Required empty public constructor
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_java, container, false);
    }

}
```

File: fragment_java.xml

```xml
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="tablayout.example.com.tablayoutwithframelayout.JavaFragment">


    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center"
        android:text="@string/java_fragment" />

</FrameLayout>
```

File: AndroidFragment.java

```java
package tablayout.example.com.tablayoutwithframelayout;

import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

public class AndroidFragment extends Fragment {

    public AndroidFragment() {
        // Required empty public constructor
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
```

```
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_android, container, false);
    }

}
```

File: fragment_android.xml

```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="tablayout.example.com.tablayoutwithframelayout.AndroidFragmen
t">

    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center"
        android:text="@string/android_fragment" />

</FrameLayout>
```

File: PhpFragment.java

```
package tablayout.example.com.tablayoutwithframelayout;

import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

public class PhpFragment extends Fragment {
    public PhpFragment() {
        // Required empty public constructor
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_php, container, false);
    }

}
```

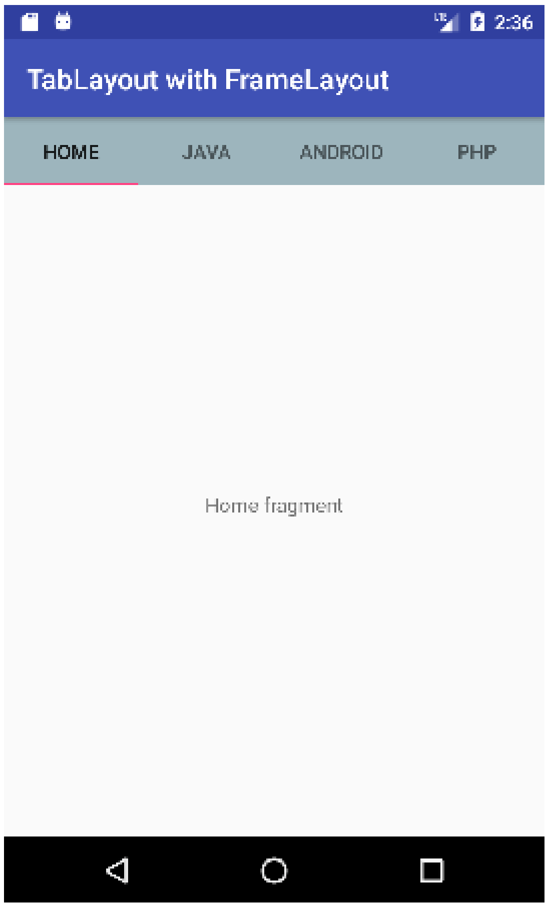
File: fragment_php.xml

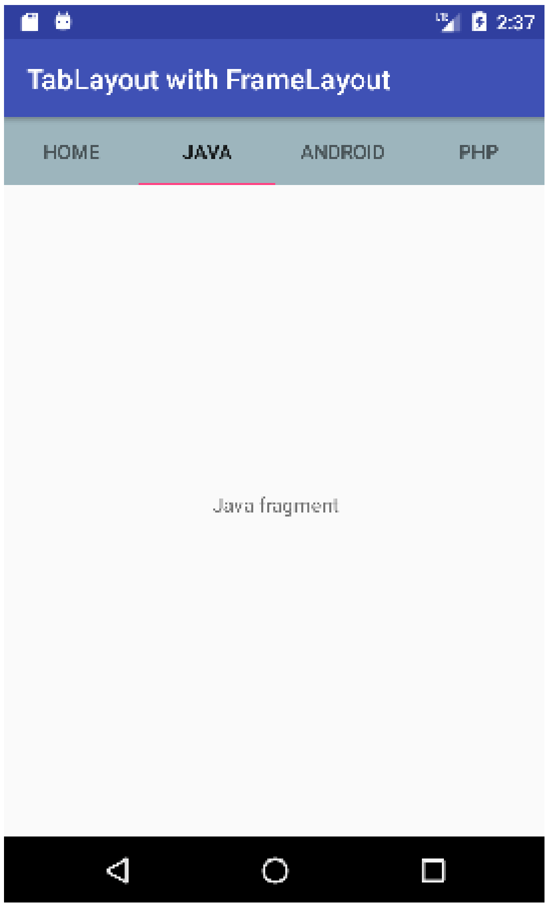
```
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="tablayout.example.com.tablayoutwithframelayout.PhpFragment">

    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center"
        android:text="@string/php_fragment" />

</FrameLayout>
```

File: strings.xml

```
<resources>
    <string name="app_name">TabLayout with FrameLayout</string>
    <!-- TODO: Remove or change this placeholder text -->
    <string name="home_fragment">Home fragment</string>
    <string name="java_fragment">Java fragment</string>
    <string name="android_fragment">Android fragment</string>
    <string name="php_fragment">Php fragment</string>
</resources>
```

Output:


2:36
TabLayout with FrameLayout
HOME
JAVA
ANDROID
PHP
Home fragment


2:37
TabLayout with FrameLayout
HOME
JAVA
ANDROID
PHP
Java fragment