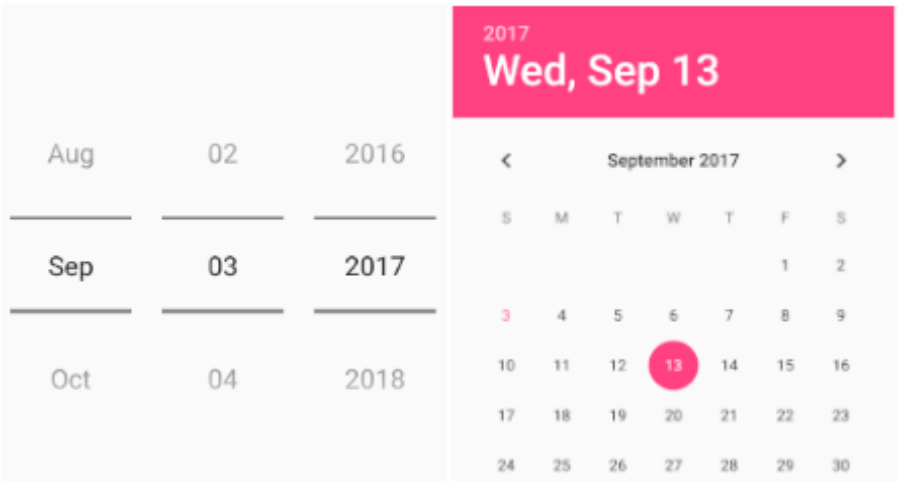
Advance Dialog

Datepicker Dialog

In android, **DatePicker** is a control that will allow users to select the date by a day, month and year in our application user interface.

If we use **DatePicker** in our application, it will ensure that the users will select a valid date.

Following is the pictorial representation of using a datepicker control in android applications.



Generally, in android DatePicker available in two modes, one is to show the complete calendar and another one is to show the dates in spinner view.

Create Android DatePicker in XML Layout File

In android, we can create a DatePicker in XML layout file using **<DatePicker>** element with different attributes like as shown below

```
<DatePicker android:id="@+id/datePicker1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

In anroid, the DatePicker supports a two types of modes, those are **Calendar** and Spinner to show the date details in our application.

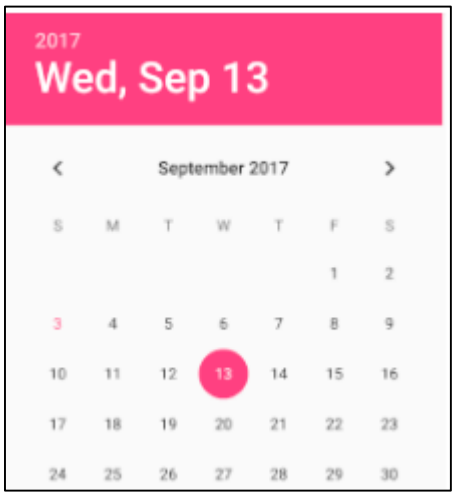
Android DatePicker with Calendar Mode

We can define android DatePicker to show only a calendar view by using DatePicker android:datePickerMode attribute.

Following is the example of showing the DatePicker in **Calendar** mode.

```
<DatePicker
    android:id="@+id/datePicker1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:datePickerMode="calendar"/>
```

The above code snippet will return the DatePicker in android like as shown below



If you observe the above result we got the DatePicker in calendar mode to select a date based on our requirements.

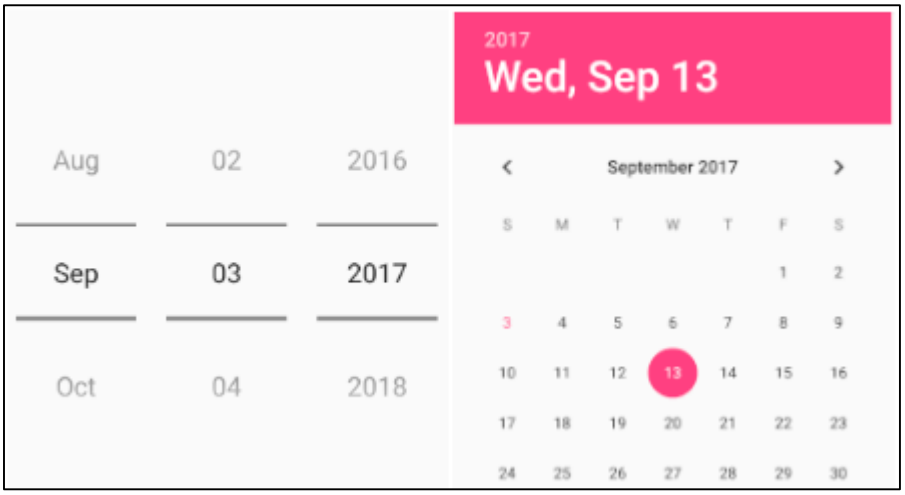
Android DatePicker with Spinner Mode

If we want to show the DatePicker in spinner format like showing day, month and year separately to select the date, then by using DatePicker android:datePickerMode attribute we can achieve this.

Following is the example of showing the DatePicker in Spinner mode.

```
 <DatePicker
    android:id="@+id/datePicker1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:datePickerMode="spinner"/>
```

The above code snippet will return the DatePicker in android like as shown below

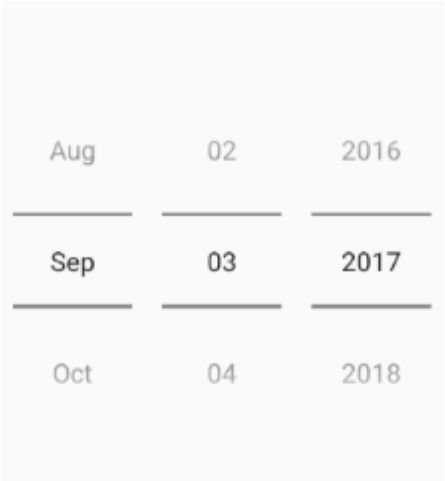


If you observe the above result we got the DatePicker in both Spinner and **Calendar** modes to select the date.

To get only spinner mode date selection, then we need to set android:calendarViewShown="false" attribute in DatePicker control like as shown below.

```
<DatePicker
    android:id="@+id/datePicker1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:datePickerMode="spinner"
    android:calendarViewShown="false"/>
```

The above code will return the DatePicker like as shown below



If you observe the above result we got the DatePicker in spinner mode to select the date separately by day, month and year.

This is how we can use DatePicker in different modes based on our requirements in android applications.

Android DatePicker Control Attributes

The following are some of the commonly used attributes related to **DatePicker** control in android applications.

Attribute	Description
android:id	It is used to uniquely identify the control
android:datePickerMode	It is used to specify datepicker mode either spinner or calendar
android:background	It is used to set the background color for the date picker.
android:padding	It is used to set the padding for left, right, top or bottom of the date picker.

Android DatePicker Example

Following is the example of defining one **DatePicker** control, one TextView control and one Button control in RelativeLayout to show the selected date on Button click in the android application.

Create a new android application using android studio and give names as **DatePickerExample**

Now open an **activity_main.xml** file from **\res\layout** path and write the code like as shown below

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/andro
id"
    android:layout_width="match_parent" android:layout_height="match_pa
rent">
    <DatePicker
        android:id="@+id/datePicker1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="20dp" />
    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:layout_below="@+id/datePicker1"
        android:layout_marginLeft="100dp"
        android:text="Get Date" />
    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/button1"
        android:layout_marginLeft="100dp"
        android:layout_marginTop="10dp"
        android:textStyle="bold"
        android:textSize="18dp"/>
</RelativeLayout>
```

If you observe above code we created a one DatePicker control, one TextView control and one Button control in XML Layout file.

Once we are done with the creation of layout with required controls, we need to load the XML layout resource from our activity **onCreate()** callback method, for that open main activity file **MainActivity.java** from **\java\com.redandwhite.datepickerexample** path and write the code like as shown below.

MainActivity.java

```
package com.redandwhite.datepickerexample;
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 btnGet;
    TextView tvw;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        tvw=(TextView)findViewById(R.id.textView1);
        picker=(DatePicker)findViewById(R.id.datePicker1);
        btnGet=(Button)findViewById(R.id.button1);
        btnGet.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                tvw.setText("Selected Date: "+ picker.getDayOfMonth()+"
/"+ (picker.getMonth() + 1)+"/"+picker.getYear());
            }
        });
```

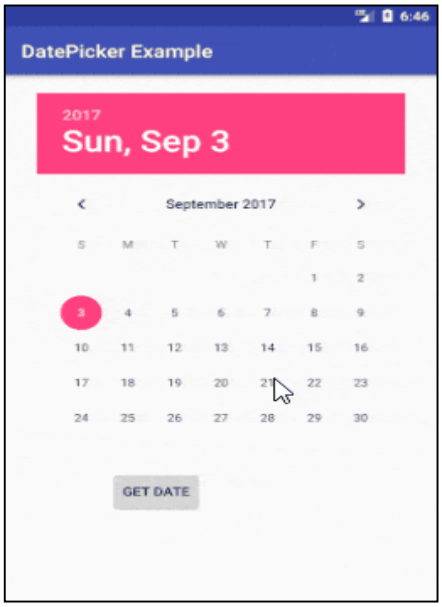
```
    }
}
```

If you observe above code we are calling our layout using **setContentView** method in the form of **R.layout.layout_file_name** in our activity file. Here our xml file name is **activity_main.xml** so we used file name **activity_main** and we are trying to show the selected date of DatePicker on Button click.

Generally, during the launch of our activity, the **onCreate()** callback method will be called by the android framework to get the required layout for an activity.

Output of Android DatePicker Example

When we run the above example using an android virtual device (AVD) we will get a result like as shown below.



If you observe the above result, we are getting the date from DatePicker when we click on Button in the android application.

Now we will see another example of showing the DatePicker control on EditText click event and get the selected date value in the android application.

Android Show DatePicker on EditText Click Example

Following is the example of open or popup datepicker dialog when we click on EditText control and get the selected date value on Button click in the android application.

Create a new android application using android studio and give names as **DatePickerExample**. Now open an **activity_main.xml** file from **\res\layout** path and write the code like as shown below

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/andro
id"
    android:layout_width="match_parent" android:layout_height="match_pa
rent">
    <EditText
        android:id="@+id/editText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="100dp"
        android:layout_marginTop="150dp"
```

```
        android:ems="10"
        android:hint="Enter Date" />
    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/editText1"
        android:layout_marginLeft="100dp"
        android:text="Get Date" />
    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/button1"
        android:layout_marginLeft="100dp"
        android:layout_marginTop="10dp"
        android:textStyle="bold"
        android:textSize="18dp"/>
</RelativeLayout>
```

If you observe above code we created a one EditText control, one TextView control and one Button control in XML Layout file.

Once we are done with the creation of layout with required controls, we need to load the XML layout resource from our activity **onCreate()** callback method, for that open main activity file **MainActivity.java** from **\java\com.redandwhite.datepickerexample** path and write the code like as shown below.

MainActivity.java

```
package com.redandwhite.datepickerexample;
import android.app.DatePickerDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.InputType;
import android.view.View;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.TextView;
import java.util.Calendar;

public class MainActivity extends AppCompatActivity {
    DatePickerDialog picker;
    EditText eText;
    Button btnGet;
    TextView tvw;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
```

```
        tvw=(TextView)findViewById(R.id.textView1);
        eText=(EditText) findViewById(R.id.editText1);
        eText.setInputType(InputType.TYPE_NULL);
        eText.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                final Calendar cldr = Calendar.getInstance();
                int day = cldr.get(Calendar.DAY_OF_MONTH);
                int month = cldr.get(Calendar.MONTH);
                int year = cldr.get(Calendar.YEAR);
                // date picker dialog
                picker = new DatePickerDialog(MainActivity.this,
                        new DatePickerDialog.OnDateSetListener() {
                            @Override
                            public void onDateSet(DatePicker view, in
t year, int monthOfYear, int dayOfMonth) {
                                eText.setText(dayOfMonth + "/" + (month
OfYear + 1) + "/" + year);
                            }
                        }, year, month, day);
                picker.show();
            }
        });
        btnGet=(Button)findViewById(R.id.button1);
        btnGet.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                tvw.setText("Selected Date: "+ eText.getText());
            }
        });
    }
}
```

If you observe above code we are calling our layout using **setContentView** method in the form of **R.layout.layout_file_name** in our activity file. Here our xml file name is **activity_main.xml** so we used file name **activity_main** and we are trying to show the DatePicker on EditText click, get the selected date of EditText control on Button click.

Generally, during the launch of our activity, the **onCreate()** callback method will be called by the android framework to get the required layout for an activity.