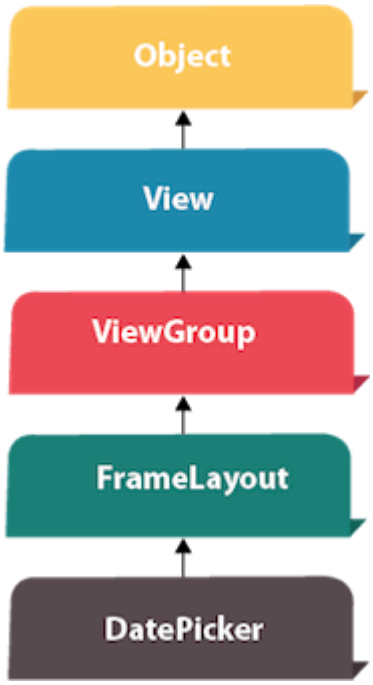
Date Picker:

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.



Methods of DatePicker

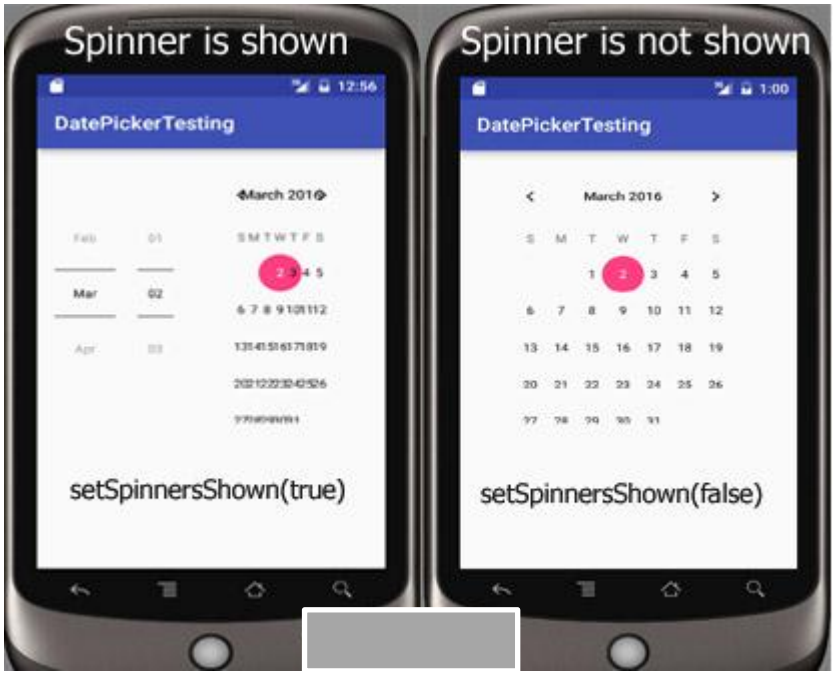
1. setSpinnersShown(boolean shown):

This method is used to set whether the spinner of the date picker in shown or not. In this method you have to set a Boolean value either true or false. True indicates spinner is shown, false value indicates spinner is not shown. Default value for this function is true.

Below we show the use of setSpinnerShown() function by setting false value.

```
DatePicker simpleDatePicker = (DatePicker)findViewById(R.id.simpleDatePicker); //
initiate a date picker
```

```
simpleDatePicker.setSpinnersShown(false); // set false value for the spinner shown function
```



2. getDayOfMonth():

This method is used to get the selected day of the month from a date picker. This method returns an integer value.

Below we get the selected day of the month from a date picker.

```
/*Add in Oncreate() funtion after setContentView()*/

DatePicker simpleDatePicker = (DatePicker) findViewById(R.id.simpleDatePicker); // initiate a date picker

int day = simpleDatePicker.getDayOfMonth(); // get the selected day of the month
```

3. getMonth():

This method is used to get the selected month from a date picker. This method returns an integer value.

Below we get the selected month from a date picker.

```
DatePicker simpleDatePicker = (DatePicker)findViewById(R.id.simpleDatePicker); // initiate a date picker

int month = simpleDatePicker.getMonth(); // get the selected month
```

4. getYear():

This method is used to get the selected year from a date picker. This method returns an integer value.

Below code is used to get the selected year from a date picker.

```
DatePicker simpleDatePicker = (DatePicker)findViewById(R.id.simpleDatePicker); // initiate a date picker

int year = simpleDatePicker.getYear(); // get the selected year
```

5. getFirstDayOfWeek():

This method is used to get the first day of the week. This method returns an integer value.

Below code is used to get the first day of the week.

```
DatePicker simpleDatePicker = (DatePicker)findViewById(R.id.simpleDatePicker);
 // initiate a date picker


int firstDay=simpleDatePicker.getFirstDayOfWeek();
 // get the first day of the week
```

Attributes of DatePicker

Now let's we discuss some important attributes that helps us to configure a DatePicker in your XML file (layout).

1. id: id is an attribute used to uniquely identify a date picker.

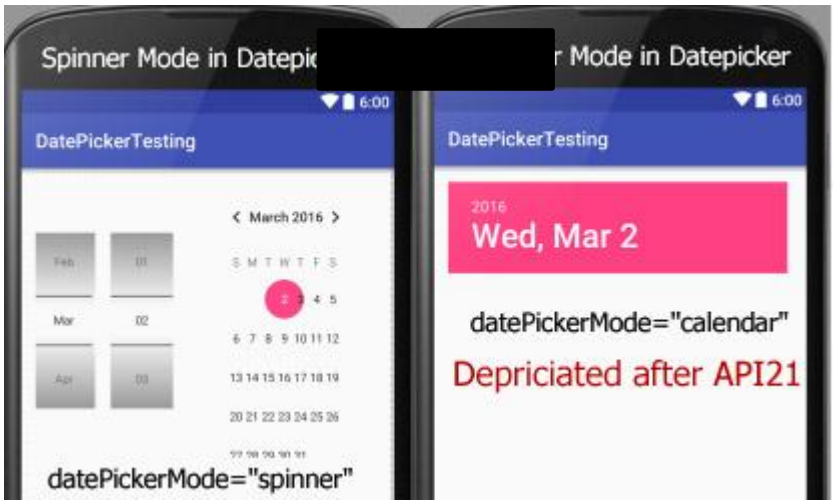
```
<DatePicker
android:id="@+id/simpleDatePicker"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
/>
```

2. datePickerMode: This attribute is used to set the Date Picker in mode either spinner or calendar. Default mode is calendar but this mode is not used after api level 21, so from api level 21 you have to set the mode to spinner.

Below is an example code in which we set the mode to spinner for a date picker.

```
<DatePicker
android:id="@+id/simpleDatePicker"
android:layout_width="wrap_content"
```

```
android:layout_height="wrap_content"
android:datePickerMode="spinner" /> <!-- spinner mode of a date picker -->
```



3. background: background attribute is used to set the background of a date picker. We can set a color or a drawable image in the background.

Below we set the red color for the background of a date picker.

```
<DatePicker
android:id="@+id/simpleDatePicker"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:datePickerMode="spinner"
android:background="#f00"/> <!-- red color for the background of the date picker -
->
```

Setting background of DatePicker In Java Class:

```
DatePicker simpleDatePicker=(DatePicker)findViewById(R.id.simpleDatePicker); // in
itiate a date picker

simpleDatePicker.setBackgroundColor(Color.RED); //  red color for the background o
f a date picker
```

4. padding: padding attribute is used to set the padding from left, right, top or bottom for a date picker.

- **paddingRight:** set the padding from the right side of the date picker.
- **paddingLeft:** set the padding from the left side of the date picker.
- **paddingTop:** set the padding from the top side of the date picker.
- **paddingBottom:** set the padding from the bottom side of the date picker.
- **Padding:** set the padding from the all side's of the date picker.

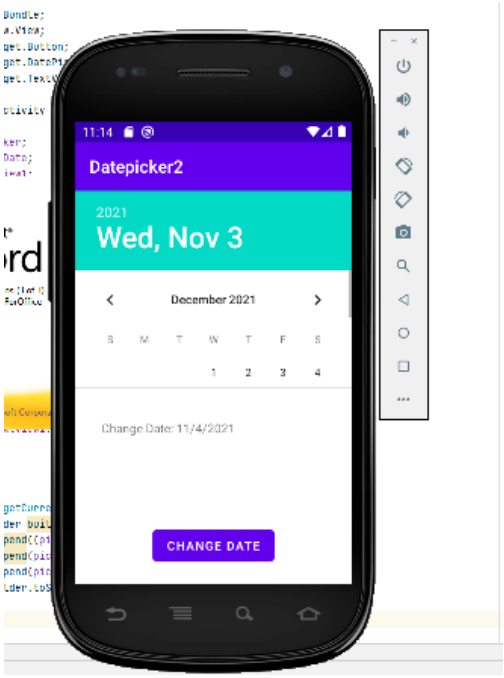
Below code of padding attribute set the 40dp padding from all the side's of the date picker.

```
<DatePicker

android:id="@+id/simpleDatePicker"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:datePickerMode="spinner"

android:padding="40dp"/> <!-- 40dp padding from all the sides of a date picker -->
```


padding in DatePicker
6:00
DatePickerTesting
March 2018
Feb
Mar
Apr
01
02
03
padding="40dp"

Sr.No	Method & description
1	**getDayOfMonth()** This method gets the selected day of month
2	**getMonth()** This method gets the selected month
3	**getYear()** This method gets the selected year
4	**setMaxDate(long maxDate)** This method sets the maximal date supported by this DatePicker in milliseconds since January 1, 1970 00:00:00 in getDefault() time zone
5	**setMinDate(long minDate)** This method sets the minimal date supported by this NumberPicker in milliseconds since January 1, 1970 00:00:00 in getDefault() time zone
6	**setSpinnersShown(boolean shown)** This method sets whether the spinners are shown
7	**updateDate(int year, int month, int dayOfMonth)** This method updates the current date
8	**getCalendarView()** This method returns calendar view
9	**getFirstDayOfWeek()** This Method returns first day of the week

Example :1

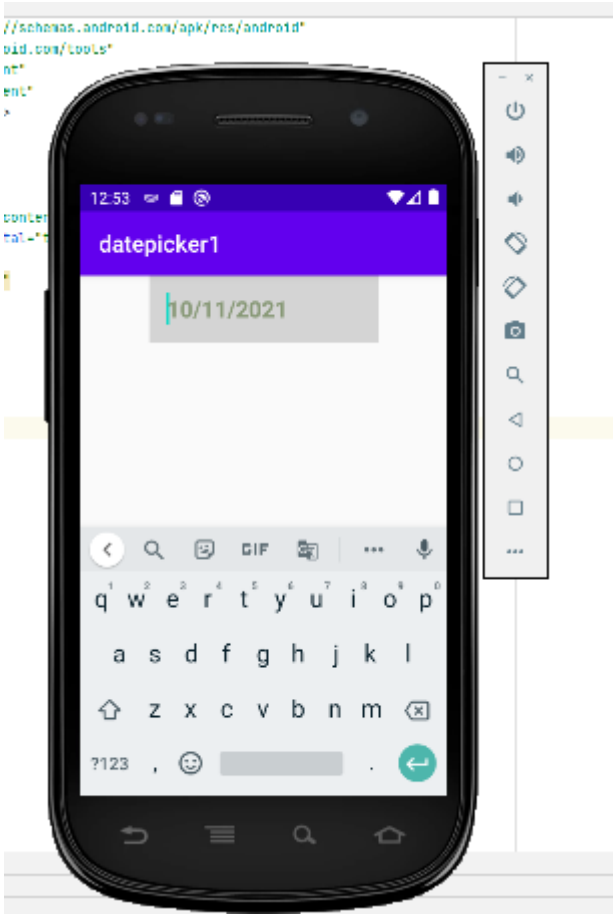



```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"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

    <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>
```

```
package com.example.datepicker2;

import androidx.appcompat.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();
    }
}
```

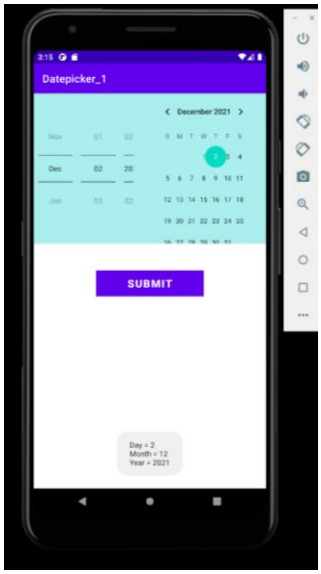
Example2:




```
<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"
       tools:context=".MainActivity">

    <EditText
        android:id="@+id/date"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:background="#d4d4d4"
        android:hint="Select Date..."
        android:padding="15dp"
        android:textColor="#897"
        android:textColorHint="#090"
        android:textSize="20sp"
        android:textStyle="bold" />

</RelativeLayout>
```

```
package com.example.datepicker1;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.app.DatePickerDialog;
import android.view.View;
import android.widget.DatePicker;
import android.widget.EditText;

import java.util.Calendar;

public class MainActivity extends AppCompatActivity {
    EditText date;
    DatePickerDialog datePickerDialog;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // initiate the date picker and a button
        date = (EditText) findViewById(R.id.date);
        // perform click event on edit text
        date.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // calender class's instance and get current date , month
and year from calender
                final Calendar c = Calendar.getInstance();
                int mYear = c.get(Calendar.YEAR); // current year
                int mMonth = c.get(Calendar.MONTH); // current month
                int mDay = c.get(Calendar.DAY_OF_MONTH); // current day
                // date picker dialog
                datePickerDialog = new DatePickerDialog(MainActivity.this,
                        new DatePickerDialog.OnDateSetListener() {

                            @Override
                            public void onDateSet(DatePicker view, int
year,
                                                  int monthOfYear, int
dayOfMonth) {
                                // set day of month , month and year value
in the edit text
                                date.setText(dayOfMonth + "/"
                                        + (monthOfYear + 1) + "/" + year);

                            }
                        }, mYear, mMonth, mDay);
                datePickerDialog.show();
            }
        });

    }


}
```

Example 3:

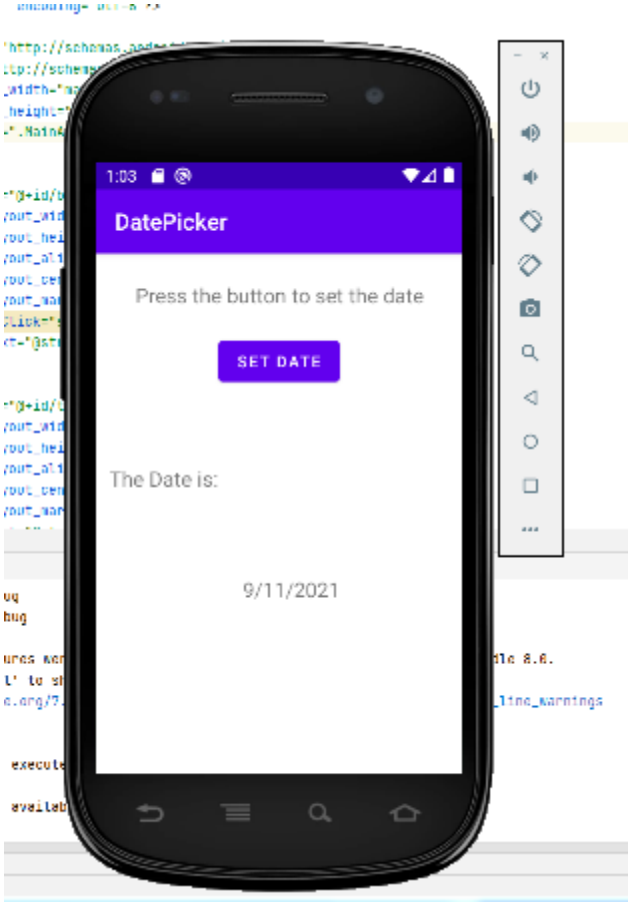



```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity">

<DatePicker
    android:id="@+id/simpleDatePicker"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#aee"
    android:datePickerMode="spinner" />

<Button
    android:id="@+id/submitButton"
    android:layout_width="200dp"
    android:layout_height="wrap_content"
    android:layout_below="@+id/simpleDatePicker"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="50dp"
    android:background="#150"
    android:text="SUBMIT"
    android:textColor="#fff"
    android:textSize="20sp"
    android:textStyle="bold" />
</RelativeLayout>
```

```
package com.example.datepicker3;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
```

```
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    DatePicker simpleDatePicker;
    Button submit;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // initiate the date picker and a button
        simpleDatePicker = (DatePicker)
findViewById(R.id.simpleDatePicker);
        submit = (Button) findViewById(R.id.submitButton);
        // perform click event on submit button
        submit.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // get the values for day of month , month and year from a
date picker
                String day = "Day = " + simpleDatePicker.getDayOfMonth();
                String month = "Month = " + (simpleDatePicker.getMonth() +
1);
                String year = "Year = " + simpleDatePicker.getYear();
                // display the values by using a toast
                Toast.makeText(getApplicationContext(), day + "\n" + month
+ "\n" + year, Toast.LENGTH_LONG).show();
            }
        });

    }


}
```

Example 4:




```
<?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"
     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="70dp"
        android:onClick="setDate"
        android:text="@string/date_button_set" />

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="24dp"
        android:text="@string/date_label_set"
```

```
        android:textAppearance="?android:attr/textAppearanceMedium" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/button1"
        android:layout_marginTop="66dp"
        android:layout_toLeftOf="@+id/button1"
        android:text="@string/date_view_set"
        android:textAppearance="?android:attr/textAppearanceMedium" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignRight="@+id/button1"
        android:layout_below="@+id/textView2"
        android:layout_marginTop="72dp"
        android:text="@string/date_selected"
        android:textAppearance="?android:attr/textAppearanceMedium" />

</RelativeLayout>


<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">DatePicker</string>
    <string name="action_settings">Settings</string>
    <string name="hello_world">Hello world!</string>
    <string name="date_label_set">Press the button to set the date</string>
    <string name="date_button_set">Set Date</string>
    <string name="date_view_set">The Date is: </string>
    <string name="date_selected"></string>
</resources>
```

```java
package com.example.datepicker4;

import androidx.appcompat.app.AppCompatActivity;

import android.app.DatePickerDialog;
import android.app.Dialog;
import android.os.Bundle;
import android.view.View;
import android.widget.DatePicker;
import android.widget.TextView;
import android.widget.Toast;

import java.util.Calendar;

public class MainActivity extends AppCompatActivity {


        private DatePicker datePicker;
        private Calendar calendar;
        private TextView dateView;
        private int year, month, day;

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

            dateView = (TextView) findViewById(R.id.textView3);
            calendar = Calendar.getInstance();
            year = calendar.get(Calendar.YEAR);

            month = calendar.get(Calendar.MONTH);
            day = calendar.get(Calendar.DAY_OF_MONTH);
            showDate(year, month+1, day);
        }

        @SuppressWarnings("deprecation")
        public void setDate(View view) {
            showDialog(999);
            Toast.makeText(getApplicationContext(), "ca",
                    Toast.LENGTH_SHORT)
                    .show();
        }

        @Override
        protected Dialog onCreateDialog(int id) {
            // TODO Auto-generated method stub
            if (id == 999) {
                return new DatePickerDialog(this,
                        myDateListener, year, month, day);
            }
            return null;
        }

        private DatePickerDialog.OnDateSetListener myDateListener = new
                DatePickerDialog.OnDateSetListener() {
                    @Override
                    public void onDateSet(DatePicker arg0,
                                          int arg1, int arg2, int arg3) {
                        // TODO Auto-generated method stub
                        // arg1 = year
```

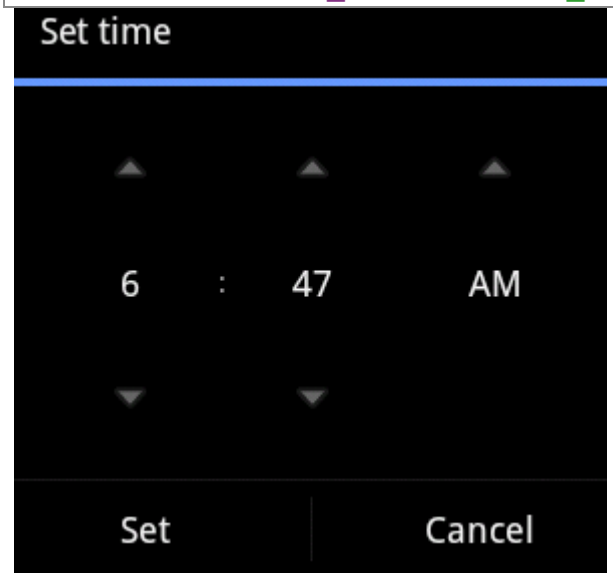
```
                    // arg2 = month
                    // arg3 = day
                    showDate(arg1, arg2+1, arg3);
                }
            };

    private void showDate(int year, int month, int day) {
        dateView.setText(new StringBuilder().append(day).append("/")
                .append(month).append("/").append(year));
    }
}
```

Time picker:

Android Time Picker allows you to select the time of day in either 24 hour or AM/PM mode. The time consists of hours, minutes and clock format. Android provides this functionality through TimePicker class.

In order to use TimePicker class, you have to first define the TimePicker component in your activity.xml. It is define as below −

```
<TimePicker
   android:id="@+id/timePicker1"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content" />
```



After that you have to create an object of TimePicker class and get a reference of the above defined xml component. Its syntax is given below.

```
import android.widget.TimePicker;
private TimePicker timePicker1;
timePicker1 = (TimePicker) findViewById(R.id.timePicker1);
```

In order to get the time selected by the user on the screen, you will use getCurrentHour() and getCurrentMinute() method of the TimePicker Class. Their syntax is given below.

```
int hour = timePicker1.getCurrentHour();
int min = timePicker1.getCurrentMinute();
```

Apart form these methods, there are other methods in the API that gives more control over TimePicker Component. They are listed below.

Sr.No	Method & description

1	**is24HourView()** This method returns true if this is in 24 hour view else false
2	**isEnabled()** This method returns the enabled status for this view
3	**setCurrentHour(Integer currentHour)** This method sets the current hour
4	**setCurrentMinute(Integer currentMinute)** This method sets the current minute
5	**setEnabled(boolean enabled)** This method set the enabled state of this view
6	**setIs24HourView(Boolean is24HourView)** This method set whether in 24 hour or AM/PM mode
7	**setOnTimeChangedListener(TimePicker.OnTimeChangedListener onTimeChangedListener)** This method Set the callback that indicates the time has been adjusted by the user

Example1:


```
<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"
        tools:context=".MainActivity">

    <TimePicker
        android:id="@+id/simpleTimePicker"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="50dp"
        android:background="#090"
        android:padding="20dp"
        android:timePickerMode="spinner" />

    <TextView
        android:id="@+id/time"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:text="Time Is ::"
        android:textColor="#090"
        android:textSize="20sp"
        android:textStyle="bold" />

</RelativeLayout>
```

```
package com.example.timepicker;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
```

```
import android.widget.TextView;
import android.widget.TimePicker;
import android.widget.Toast;



public class MainActivity extends AppCompatActivity {

    TextView time;
    TimePicker simpleTimePicker;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //  initiate the view's
        time = (TextView) findViewById(R.id.time);
        simpleTimePicker = (TimePicker)
findViewById(R.id.simpleTimePicker);
        simpleTimePicker.setIs24HourView(false); // used to display AM/PM
mode
        // perform set on time changed listener event
        simpleTimePicker.setOnTimeChangedListener(new
TimePicker.OnTimeChangedListener() {
            @Override
            public void onTimeChanged(TimePicker view, int hourOfDay, int
minute) {
                // display a toast with changed values of time picker
                Toast.makeText(getApplicationContext(), hourOfDay + "  " +
minute, Toast.LENGTH_SHORT).show();
                time.setText("Time is :: " + hourOfDay + " : " + minute);
// set the current time in text view
            }
        });
    }


}
```

Example2:


```
<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"
       tools:context=".MainActivity">

    <EditText
        android:id="@+id/time"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:hint="Select Time..."
        android:textColor="#090"
        android:textColorHint="#090"
        android:background="#d4d4d4"
        android:padding="15dp"
        android:textSize="20sp"
        android:textStyle="bold" />

</RelativeLayout>
```

```
package com.example.timepicker1;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.app.TimePickerDialog;


import android.view.View;
import android.widget.EditText;

import android.widget.TimePicker;
import java.util.Calendar;

public class MainActivity extends AppCompatActivity {
```

```
    EditText time;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //  initiate the edit text
        time = (EditText) findViewById(R.id.time);
        // perform click event listener on edit text
        time.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Calendar mcurrentTime = Calendar.getInstance();
                int hour = mcurrentTime.get(Calendar.HOUR_OF_DAY);
                int minute = mcurrentTime.get(Calendar.MINUTE);
                TimePickerDialog mTimePicker;
                mTimePicker = new TimePickerDialog(MainActivity.this, new
TimePickerDialog.OnTimeSetListener() {
                    @Override
                    public void onTimeSet(TimePicker timePicker, int
selectedHour, int selectedMinute) {
                        time.setText(selectedHour + ":" + selectedMinute);
                    }
                }, hour, minute, true);//Yes 24 hour time
                mTimePicker.setTitle("Select Time");
                mTimePicker.show();

            }
        });
    }


}
```

Inflating is the process of adding a view (.xml) to activity on runtime.

A layman definition for inflation might be to convert the XML code to Java code. Just a way to understand, e.g., if we have a tag in XML, OS has to create a corresponding Java object in memory, so inflatter reads the XMLtags, and creates the corresponding objects in Java.

Because we make UI into XML but view objects is what we display so we somehow need to convert xml into view objects so inflating means we are converting xml into view objects so that it can be displayed, for this we need a service called layout inflator service and give it an xml and it will be convert for you.

Inflating is the process of adding a view (. xml) to activity on runtime. When we create a listView we inflate each of its items dynamically. If we want to create a ViewGroup with multiple views like buttons and textview, we can create it like so: ... setText =”button text”; txt.

What is inflate method in Android?

inflate(int resource, ViewGroup root) Inflate a new view hierarchy from the specified xml resource. View. inflate(XmlPullParser parser, ViewGroup root) Inflate a new view hierarchy from the specified xml node.

How do you inflate a view on Android?

Just think we specified a button in an XML layout file with its layout width and layout height set to match_parent. On This Buttons Click Event We Can Set Following Code to Inflate Layout on This Activity. LayoutInflater inflater = LayoutInflater. from(getContext()); inflater.

How do you use LayoutInflater?

1. attachToRoot Set to True

<Button xmlns_android=”http://schemas.android.com/apk/res/android” android_layout_width=”match_parent” android_layout_height=”wrap_content” android_text=”@string/action_attach_to_root_true” …

inflater. inflate(R. layout. …

Button btnAttachToRootFalse = (Button) inflater. inflate(R. layout.

Source: https://ostoday.org/android/best-answer-what-is-inflate-in-android.html

How do you inflate a fragment?

Android calls the onCreateView() callback method to display a Fragment . Override this method to inflate the layout for a Fragment , and return a View that is the root of the layout for the Fragment . The container parameter passed to onCreateView() is the parent ViewGroup from the Activity layout.

Why Inflater is used in Android?

What is an Inflater ? To summarize what the LayoutInflater Documentation says… A LayoutInflater is one of the Android System Services that is responsible for taking your XML files that define a layout, and converting them into View objects. The OS then uses these view objects to draw the screen.

What is attach to root in Android?

attaches the views to their parent (includes them in the parent hierarchy), so any touch event that the views recieve will also be transfered to parent view.

Source: https://ostoday.org/android/best-answer-what-is-inflate-in-android.html