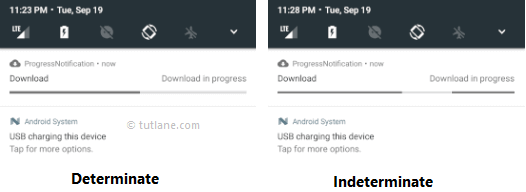
**4. Android Progress Notification with Examples**

In android, **Progress Notification** is used to show the progress of an ongoing operation in notification bar. By using progress notification, we can easily know that how much percentage of the current operation completed and how long the operation will run to complete the remaining operation.

In android, two types of progress indicators available, one is **determinate** and another one is **indeterminate**. If we are known about how long the operation will take to complete, then we can use **determinate** form of the indicator. In case if we are not aware of how long the operation will run then we can use **indeterminate** form of indicator.

Following is the pictorial representation of using different type of android progress indicators (**determinate** and **indeterminate**) to show the progress of current operation in notification.



Generally, the progress indicators in android are implemented by using [ProgressBar](https://www.tutlane.com/tutorial/android/android-progressbar-with-examples" \o "Android ProgressBar Control with Examples" \t "_blank) class. To display the progress indicators in our app, we need to add the [progress bar](https://www.tutlane.com/tutorial/android/android-progressbar-with-examples) to our notification by calling **setProgress(max, progress, false)**method and then issue the notification.

Here the third argument in **setProgress()** method is used to indicate whether the [progress bar](https://www.tutlane.com/tutorial/android/android-progressbar-with-examples) is **determinate** (false) or **indeterminate** (true). As our operation proceeds, we need to increase the value of **progress**, and update the status of [notification](https://www.tutlane.com/tutorial/android/android-notifications-bigtextstyle-bigpicturestyle-inboxstyle). At the end of operation, the **progress** value must be equal to **max** value. The better way to call **setProgress()**is to set **max** value to **100** and then increment **progress** as a percent complete value for the operation.

Once the operation is done leave the [progress bar](https://www.tutlane.com/tutorial/android/android-progressbar-with-examples) showing or remove it by calling **setProgress(0,0, false)** and update the notification text to show that the operation is complete.

Now we will see how to create and show the progress bar in android notification bar with examples.

**Android Progress Notification Example**

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

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"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout\_width="match\_parent"  
    android:layout\_height="match\_parent"  
    android:orientation="vertical" >  
    <Button  
        android:id="@+id/btnShow"  
        android:layout\_width="wrap\_content"  
        android:layout\_height="wrap\_content"  
        android:text="Show Notification"  
        android:layout\_marginTop="100dp" android:layout\_marginLeft="120dp"/>  
</LinearLayout>

If we observe above code we created a one [Button](https://www.tutlane.com/tutorial/android/android-button-with-examples) control in XML Layout file to show the progress indicator in notification when we click on [Button](https://www.tutlane.com/tutorial/android/android-button-with-examples).

Once we are done with creation of layout with required controls, we need to load the XML layout resource from our [activity](https://www.tutlane.com/tutorial/android/android-activity-lifecycle) **onCreate()** callback method, for that open main activity file **MainActivity.java** from **\java\com.tutlane.progressnotification** path and write the code like as shown below.

**MainActivity.java**

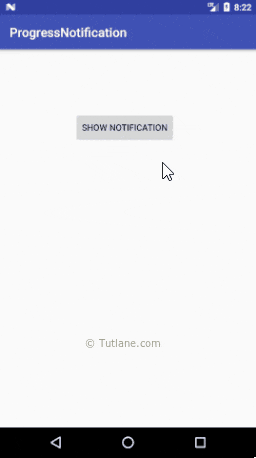
package com.tutlane.progressnotification;  
import android.app.NotificationManager;  
import android.content.Context;  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.support.v7.app.NotificationCompat;  
import android.util.Log;  
import android.view.View;  
import android.widget.Button;  
  
public class MainActivity extends AppCompatActivity {  
    private NotificationManager mNotifyManager;  
    private NotificationCompat.Builder mBuilder;  
    int id = 1;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity\_main);  
        Button b1 = (Button) findViewById(R.id.btnShow);  
        b1.setOnClickListener(new View.OnClickListener() {  
            @Override  
            public void onClick(View v) {  
                mNotifyManager = (NotificationManager) getSystemService(Context.NOTIFICATION\_SERVICE);  
                mBuilder = new NotificationCompat.Builder(MainActivity.this);  
                mBuilder.setContentTitle("File Download")  
                        .setContentText("Download in progress")  
                        .setSmallIcon(R.drawable.download);  
                // Start a the operation in a background thread  
                new Thread(  
                        new Runnable() {  
                            @Override  
                            public void run() {  
                                int incr;  
                                // Do the "lengthy" operation 20 times  
                                for (incr = 0; incr <= 100; incr+=5) {  
                                    // Sets the progress indicator to a max value, the current completion percentage and "determinate" state  
                                    mBuilder.setProgress(100, incr, false);  
                                    // Displays the progress bar for the first time.  
                                    mNotifyManager.notify(id, mBuilder.build());  
                                    // Sleeps the thread, simulating an operation  
                                    try {  
                                        // Sleep for 1 second  
                                        Thread.sleep(1\*1000);  
                                    } catch (InterruptedException e) {  
                                        Log.d("TAG", "sleep failure");  
                                    }  
                                }  
                                // When the loop is finished, updates the notification  
                                mBuilder.setContentText("Download completed")  
                                        // Removes the progress bar  
                                        .setProgress(0,0,false);  
                                mNotifyManager.notify(id, mBuilder.build());  
                            }  
                        }  
                // Starts the thread by calling the run() method in its Runnable  
                ).start();  
            }  
        });  
    }  
}

If we observe above code we are created a progress notification using **setProgress()** method and showing the progress notification on [Button](https://www.tutlane.com/tutorial/android/android-button-with-examples) click.

Generally, during the launch of our [activity](https://www.tutlane.com/tutorial/android/android-activity-lifecycle), **onCreate()** callback method will be called by android framework to get the required layout for an [activity](https://www.tutlane.com/tutorial/android/android-activity-lifecycle).

**Output of Android Progress Notification Example**

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



If we observe above result we created a progress notification and shown it on [Button](https://www.tutlane.com/tutorial/android/android-button-with-examples) click based on our requirements.

This is how we can use progress notification in our android applications based on our requirements.