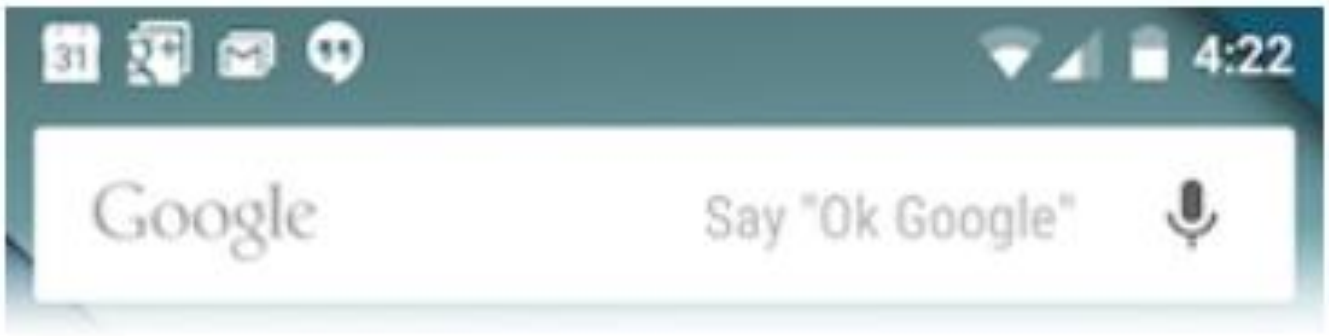
# Mobility Programming

## Lecture 6: Notifications

DR. RAMEZ ALKHATIB

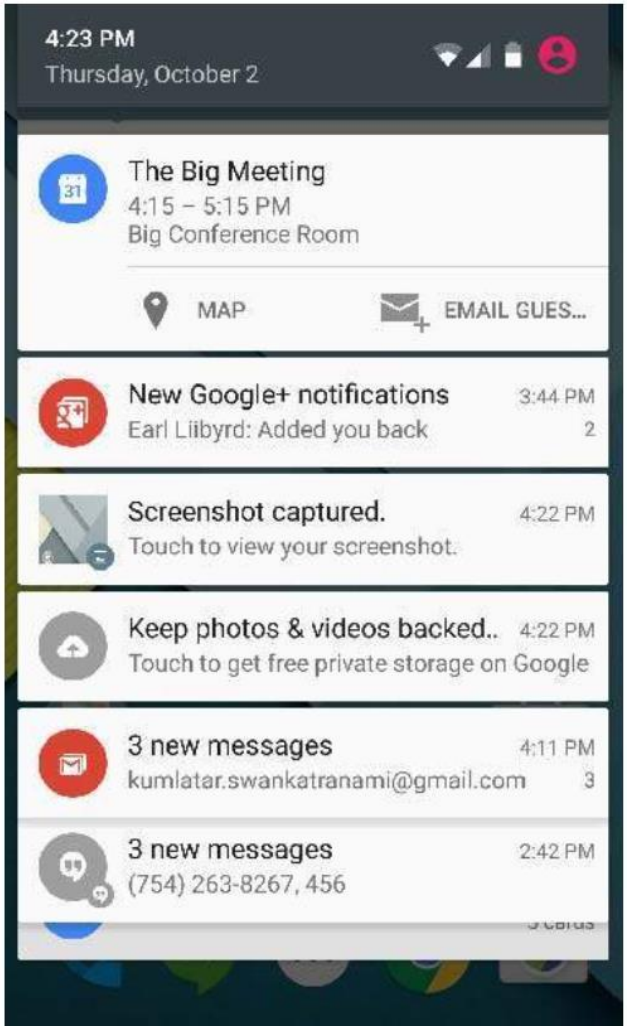
# Notifications

- A **notification** is a message you can display to the user outside of your application's normal UI. When you tell the system to issue a notification, it first appears as an icon in the notification area.



# Notifications

- To see the details of the notification, the user opens the notification drawer. Both the notification area and the notification drawer are system-controlled areas that the user can view at any time.



# Create and Send Notifications

- You have simple way to create a notification. Follow the following steps in your application to create a notification −

  Step 1 - Create Notification Builder

  As a first step is to create a notification builder using *NotificationCompat.Builder.build()*. You will use Notification Builder to set various Notification properties like its small and large icons, title, priority etc.

```
NotificationCompat.Builder mBuilder = new NotificationCompat.Builder(this)
```

# Create and Send Notifications

## Step 2 - Setting Notification Properties

Once you have **Builder** object, you can set its Notification properties using Builder object as per your requirement. But this is mandatory to set at least following −

- A small icon, set by **setSmallIcon()**
- A title, set by **setContentTitle()**
- Detail text, set by **setContentText()**

- You have plenty of optional properties which you can set for your notification. To learn more about them, see the reference documentation for NotificationCompat.Builder.

# Create and Send Notifications

## Step 3 - Attach Actions

- This is an optional part and required if you want to attach an action with the notification. An action allows users to go directly from the notification to an **Activity** in your application, where they can look at one or more events or do further work.
- The action is defined by a **PendingIntent** containing an **Intent** that starts an Activity in your application. To associate the PendingIntent with a gesture, call the appropriate method of *NotificationCompat.Builder*. For example, if you want to start Activity when the user clicks the notification text in the notification drawer, you add the PendingIntent by calling **setContentIntent()**.
- A PendingIntent object helps you to perform an action on your applications behalf, often at a later time, without caring of whether or not your application is running.

# Create and Send Notifications

## Step 4 - Issue the notification

- Finally, you pass the Notification object to the system by calling NotificationManager.notify() to send your notification. Make sure you call **NotificationCompat.Builder.build()** method on builder object before notifying it. This method combines all of the options that have been set and return a new **Notification** object.

```
NotificationManager mNotificationManager = (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
 // notificationID allows you to update the notification later on.
mNotificationManager.notify(notificationID, mBuilder.build());
```

# The NotificationCompat.Builder Class

- The NotificationCompat.Builder class allows easier control over all the flags, as well as help constructing the typical notification layouts. Following are few important and most frequently used methods available as a part of NotificationCompat.Builder class.

# System Events

| Sr.No | Event Constant & Description |
|---|---|
| 1 | **Notification build()**<br>Combine all of the options that have been set and return a new Notification object. |
| 2 | **NotificationCompat.Builder setAutoCancel (boolean autoCancel)**<br>Setting this flag will make it so the notification is automatically canceled when the user clicks it in the panel. |
| 3 | **NotificationCompat.Builder setContent (RemoteViews views)**<br>Supply a custom RemoteViews to use instead of the standard one. |
| 4 | **NotificationCompat.Builder setContentInfo (CharSequence info)**<br>Set the large text at the right-hand side of the notification. |
| 5 | **NotificationCompat.Builder setContentIntent (PendingIntent intent)**<br>Supply a PendingIntent to send when the notification is clicked. |
| 6 | **NotificationCompat.Builder setContentText (CharSequence text)**<br>Set the text (second row) of the notification, in a standard notification. |

# System Events

| Sr.No | Event Constant & Description |
|---|---|
| 7 | **NotificationCompat.Builder setContentTitle (CharSequence title)**<br>Set the text (first row) of the notification, in a standard notification. |
| 8 | **NotificationCompat.Builder setDefaults (int defaults)**<br>Set the default notification options that will be used. |
| 9 | **NotificationCompat.Builder setLargeIcon (Bitmap icon)**<br>Set the large icon that is shown in the ticker and notification. |
| 10 | **NotificationCompat.Builder setNumber (int number)**<br>Set the large number at the right-hand side of the notification. |
| 11 | **NotificationCompat.Builder setOngoing (boolean ongoing)**<br>Set whether this is an ongoing notification. |
| 12 | **NotificationCompat.Builder setSmallIcon (int icon)**<br>Set the small icon to use in the notification layouts. |

# System Events

| Sr.No | Event Constant & Description |
|---|---|
| 13 | **NotificationCompat.Builder setStyle (NotificationCompat.Style style)**<br>Add a rich notification style to be applied at build time. |
| 14 | **NotificationCompat.Builder setTicker (CharSequence tickerText)**<br>Set the text that is displayed in the status bar when the notification first arrives. |
| 15 | **NotificationCompat.Builder setVibrate (long[] pattern)**<br>Set the vibration pattern to use. |
| 16 | **NotificationCompat.Builder setWhen (long when)**<br>Set the time that the event occurred. Notifications in the panel are sorted by this time. |

# EXAMPLE

```
package com.example.notificationdemo;

import android.app.Activity;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.support.v4.app.NotificationCompat;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends Activity {
   Button b1;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

      b1 = (Button)findViewById(R.id.button);
      b1.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
            addNotification();
         }
      });
   }
```

# EXAMPLE - CONTINUE

```
   private void addNotification() {
      NotificationCompat.Builder builder =
         new NotificationCompat.Builder(this)
         .setSmallIcon(R.drawable.abc)
         .setContentTitle("Notifications Example")
         .setContentText("This is a test notification");

      Intent notificationIntent = new Intent(this, MainActivity.class);
      PendingIntent contentIntent = PendingIntent.getActivity(this, 0, notificationIntent,
         PendingIntent.FLAG_UPDATE_CURRENT);
      builder.setContentIntent(contentIntent);

      // Add as notification
      NotificationManager manager = (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
      manager.notify(0, builder.build());
   }
}
```

# res/layout/notification.xml

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

   <TextView
      android:layout_width="fill_parent"
      android:layout_height="400dp"
      android:text="Hi, Your Detailed notification view goes here...." />
</LinearLayout>
```

# NotificationView.java

```
package com.example.notificationdemo;

import android.os.Bundle;
import android.app.Activity;

public class NotificationView extends Activity{
   @Override
   public void onCreate(Bundle savedInstanceState){
      super.onCreate(savedInstanceState);
      setContentView(R.layout.notification);
   }
}
```

# **res/layout/activity_main.xml** file

```
<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="MainActivity">

   <TextView
      android:id="@+id/textView1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Notification Example"
      android:layout_alignParentTop="true"
      android:layout_centerHorizontal="true"
      android:textSize="30dp" />

   <TextView
      android:id="@+id/textView2"
```

# res/values/strings.xml

to define two new constants –

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="action_settings">Settings</string>
    <string name="app_name">tutorialspoint </string>
</resources>
```

# AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.example.notificationdemo" >

   <application
      android:allowBackup="true"
      android:icon="@drawable/ic_launcher"
      android:label="@string/app_name"
      android:theme="@style/AppTheme" >

      <activity
         android:name="com.example.notificationdemo.MainActivity"
         android:label="@string/app_name" >

         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>

      </activity>

      <activity android:name=".NotificationView"
         android:label="Details of notification"
         android:parentActivityName=".MainActivity">
         <meta-data
         android:name="android.support.PARENT_ACTIVITY"
         android:value=".MainActivity"/>
      </activity>

   </application>
</manifest>
```

# View Notification

```
protected void displayNotification() {
   Log.i("Start", "notification");

   /* Invoking the default notification service */
   NotificationCompat.Builder  mBuilder = new NotificationCompat.Builder(this);

   mBuilder.setContentTitle("New Message");
   mBuilder.setContentText("You've received new message.");
   mBuilder.setTicker("New Message Alert!");
   mBuilder.setSmallIcon(R.drawable.woman);

   /* Increase notification number every time a new notification arrives */
   mBuilder.setNumber(++numMessages);

   /* Add Big View Specific Configuration */
   NotificationCompat.InboxStyle inboxStyle = new NotificationCompat.InboxStyle();

   String[] events = new String[6];
   events[0] = new String("This is first line....");
   events[1] = new String("This is second line...");
   events[2] = new String("This is third line...");
   events[3] = new String("This is 4th line...");
   events[4] = new String("This is 5th line...");
   events[5] = new String("This is 6th line...");
```

# View Notification -CONTINUE

```
   // Sets a title for the Inbox style big view
   inboxStyle.setBigContentTitle("Big Title Details:");

   // Moves events into the big view
   for (int i=0; i < events.length; i++) {
      inboxStyle.addLine(events[i]);
   }

   mBuilder.setStyle(inboxStyle);

   /* Creates an explicit intent for an Activity in your app */
   Intent resultIntent = new Intent(this, NotificationView.class);

   TaskStackBuilder stackBuilder = TaskStackBuilder.create(this);
   stackBuilder.addParentStack(NotificationView.class);

   /* Adds the Intent that starts the Activity to the top of the stack */
   stackBuilder.addNextIntent(resultIntent);
   PendingIntent resultPendingIntent =stackBuilder.getPendingIntent(0,PendingIntent.FLAG_UPDATE_CURRENT);

   mBuilder.setContentIntent(resultPendingIntent);
   mNotificationManager = (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);

   /* notificationID allows you to update the notification later on. */
   mNotificationManager.notify(notificationID, mBuilder.build());
}
```

Questions?