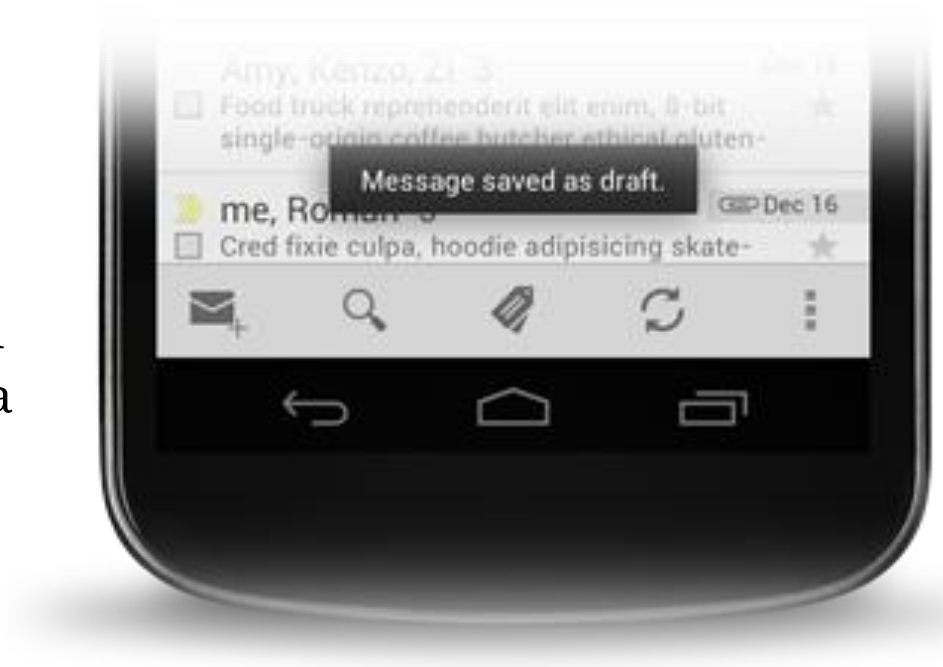
Notifications

Outline

- Toast
- Snackbar
- Notifications
- AlertDialog

Toasts

- A toast provides simple feedback about an operation in a small popup.
- It only fills the amount of space required for the message and the current activity remains visible and interactive.
- For example, navigating away from an email before you send it triggers a "Draft saved" toast to let you know that you can continue editing later.
- Toasts automatically disappear after a timeout.



Toast programming

```
Context context = getApplicationContext();
CharSequence text = "Hello toast!";
int duration = Toast.LENGTH_SHORT;
Toast toast = Toast.makeText(context, text,
duration);
toast.show();

Or chain it like this:
Toast.makeText(getApplicationContext(), “Hello
toast!”, Toast.LENGTH_LONG).show();
```

Returns the context for the entire application (the process all the Activities are running inside of)

Snackbar

- Snackbars provide lightweight feedback about an operation by showing a brief message at the bottom of the screen.
- **Snackbars can contain an action.**

Example with text only :

```
Snackbar snackbar = Snackbar.make(view, "one line",
Snackbar.LENGTH_LONG);

snackbar.show();
```

Add com.android.support:design to the project lib
&
import android.support.design.widget.Snackbar;
in your class

Snackbar

- **Example with text and action :**

```
Snackbar snackbar = Snackbar.make(view, "Message is
deleted", Snackbar.LENGTH_LONG) ;

snackbar.setAction("UNDO", new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Snackbar.make(view, "Message is restored!",
            Snackbar.LENGTH_SHORT).show();
    }
});

snackbar.show();
```

Notifications on the Status Bar

- While the Toast and Snackbar are handy ways to show users alerts, they are not persistent
  - Users may easily miss them if they are not looking at the screen

- For messages that are important, you should use a more persistent method
  - Use the **NotificationManager** to display a persistent message at the top of the device (status bar or notification bar)

Create notification – Step 1

- Add a new activity to your project. Call it **NotificationView**.

```
public class NotificationView extends Activity {

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

activity_notification_view.xml

```
<LinearLayout … >
   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Here are the details for the notification..." />

</LinearLayout>
```

Create notification – Step 2

- Modify the AndroidManifest.xml file
  - Add a permission (if needed)

<uses-permission android:name="android.permission.VIBRATE"/>

  - Add an intent-filter to the activity component

```
<activity
        android:name=".NotificationView"
        android:label="@string/title_activity_notification_view" >
        <intent-filter>
          <action android:name="android.intent.action.MAIN" />
          <category android:name="android.intent.category.DEFAULT" />
          </intent-filter>
</activity>
```

Create notification – Step 3

- Specify when the notification will be displayed.
  - For example on button click on the main activity
  - Add the following button to activity_main.xml

```
<Button
android:id="@+id/btn_displaynotif"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Display Notification"
android:onClick="displayNotification"/>
```

Create notification – Step 4

- Implement the method *displayNotification* in MainActivity.java

```
public void showNotification(View view)
{
//---PendingIntent to launch activity if the user selects this notification
Intent i = new Intent(this, NotificationView.class);
i.putExtra("notificationID", notificationID); // notificationID is any integer
PendingIntent pendingIntent = PendingIntent.getActivity(this, 0, i, 0);

NotificationManager nm = (NotificationManager)getSystemService(NOTIFICATION_SERVICE);

Notification.Builder notifBuilder = new Notification.Builder(this);
notifBuilder.setContentTitle("Reminder");
notifBuilder.setContentText("Meeting with ... starts in 5 minutes");
notifBuilder.setSmallIcon(R.drawable.ic_launcher);
notifBuilder.setContentIntent(pendingIntent);

Notification notif = notifBuilder.build();

notif.sound = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);

nm.notify(notificationID, notif);
notificationID++;
}
```

Create notification – Step 4 (with backward compatibility)

```
public void showNotification(View view)
{
    Intent intent = new Intent(this, NotificationView.class);
    intent.putExtra("notificationID", notificationID);

    PendingIntent pendingIntent = PendingIntent.getActivity(this, (new
Random()).nextInt(), intent, PendingIntent.FLAG_UPDATE_CURRENT);

    NotificationManagerCompat nmc =
        NotificationManagerCompat.from(getApplicationContext());

    NotificationCompat.Builder builder = new
        NotificationCompat.Builder(this);
    builder.setContentTitle("Title of the notif " + notificationID);
    builder.setContentText("Content of the notif " + notificationID);
    builder.setContentIntent(pendingIntent);
    builder.setSmallIcon(R.mipmap.ic_launcher);

    Notification notif = builder.build();
    notif.sound =
RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);

    nmc.notify(notificationID, notif);
    notificationID++;
}
```

Use android.support.v4.app….

Create notification

- To remove the notification when the user open the corresponding activity, modify the onCreate method of the class **NotificationView**

```
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_notification_view);

NotificationManager nm =
(NotificationManager)getSystemService(NOTIFICATION_SERVICE);

nm.cancel(getIntent().getExtras().getInt("notificationID"));
}
```

AlertDialog

```
AlertDialog.Builder alert  = new AlertDialog.Builder(this);
alert.setMessage("This is an alert …");
alert.setTitle("App Title");
alert.setNeutralButton("May be", null);

alert.setNegativeButton("No",  new OnClickListener() {
      public void onClick(DialogInterface dialog, int which) {
          …
        }
    });

alert.setPositiveButton("Yes",  new OnClickListener() {
      public void onClick(DialogInterface dialog, int which) {
          …
        }
    });

alert.setCancelable(false);
alert.create().show();
```

AlertDialog

AlertDialog.Builder alert = **new AlertDialog.Builder(this);**

To show in an alert several choices where one should be selected

Call on alert the method: setSingleChoiceItems

-setSingleChoiceItems (CharSequence[] items, int checkedItem, DialogInterface.OnClickListener listener)

To show in an alert several choices where more than one can be selected

Call on alert the method: setMultipleChoiceItems

-setMultiChoiceItems (CharSequence[] items, boolean[] checkedItems, DialogInterface.OnMultiChoiceClickListener listener)