LA TROBE
UNIVERSITY

All kinds of clever

CSE2MAD

Mobile Application Development
Lecture 5 Part 2

Outline

- Fragments
- UI Polishing

Fragments

- A reusable class that implements a part of an activity
- Dependent on the activity
- Fragments are embedded in activities
- Usually a part of a UI

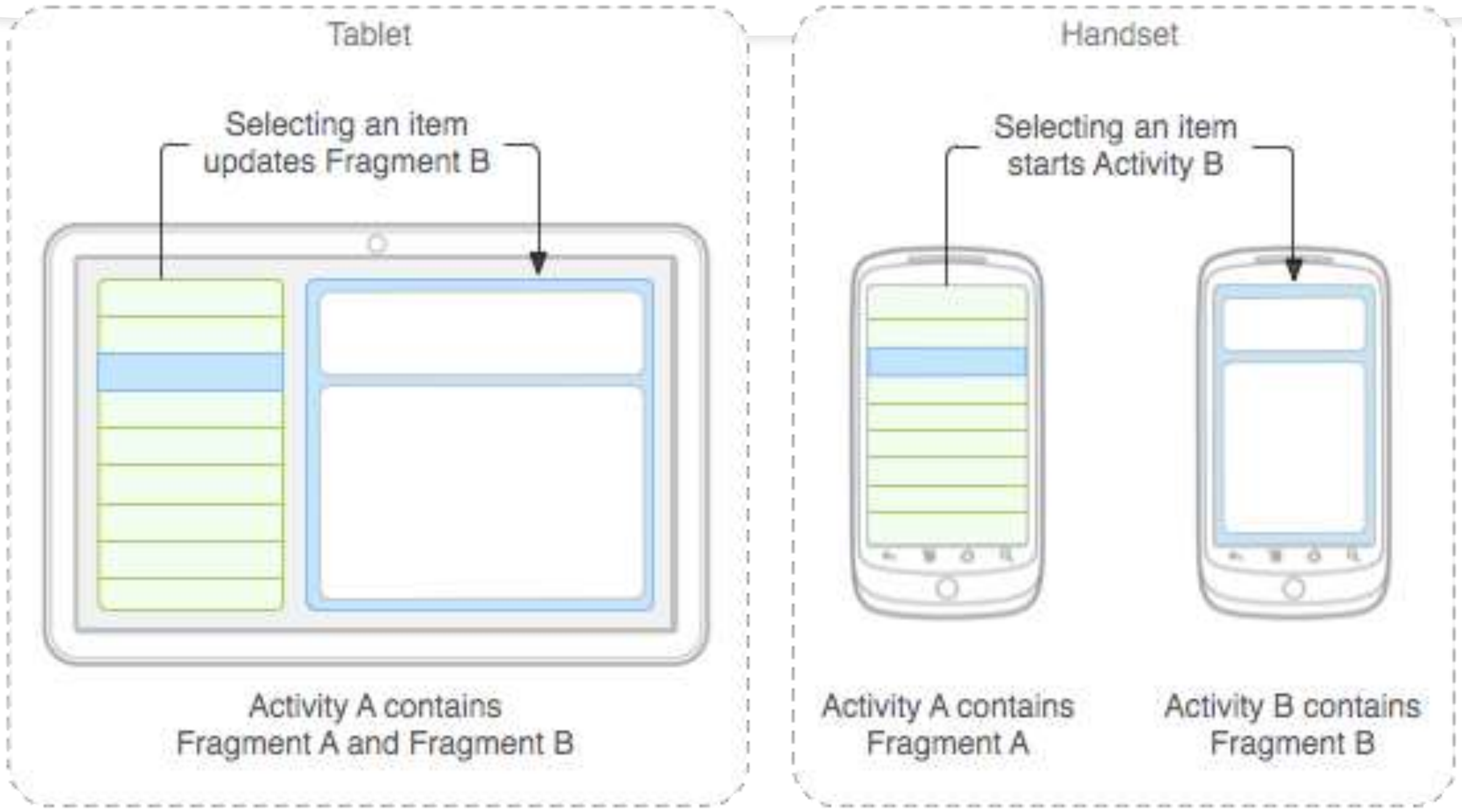
Fragments

- Contain both XML layout file & a Java class
- Encapsulate the view and logic to support reuse
- Useful for supporting multiple device types for an app
  - ➢ Phone & tablet, device specific activities but share reusable fragments
  - ➢ Orientation, as above, different layouts but re-using same fragments
- Fragments Activities = navigational controllers
  - ➢ Point to other activities via intents
  - ➢ Hide/Show fragments
  - ➢ Reveal nav drawer etc
  - ➢ Receive data from intents and forward to other fragments

Fragments

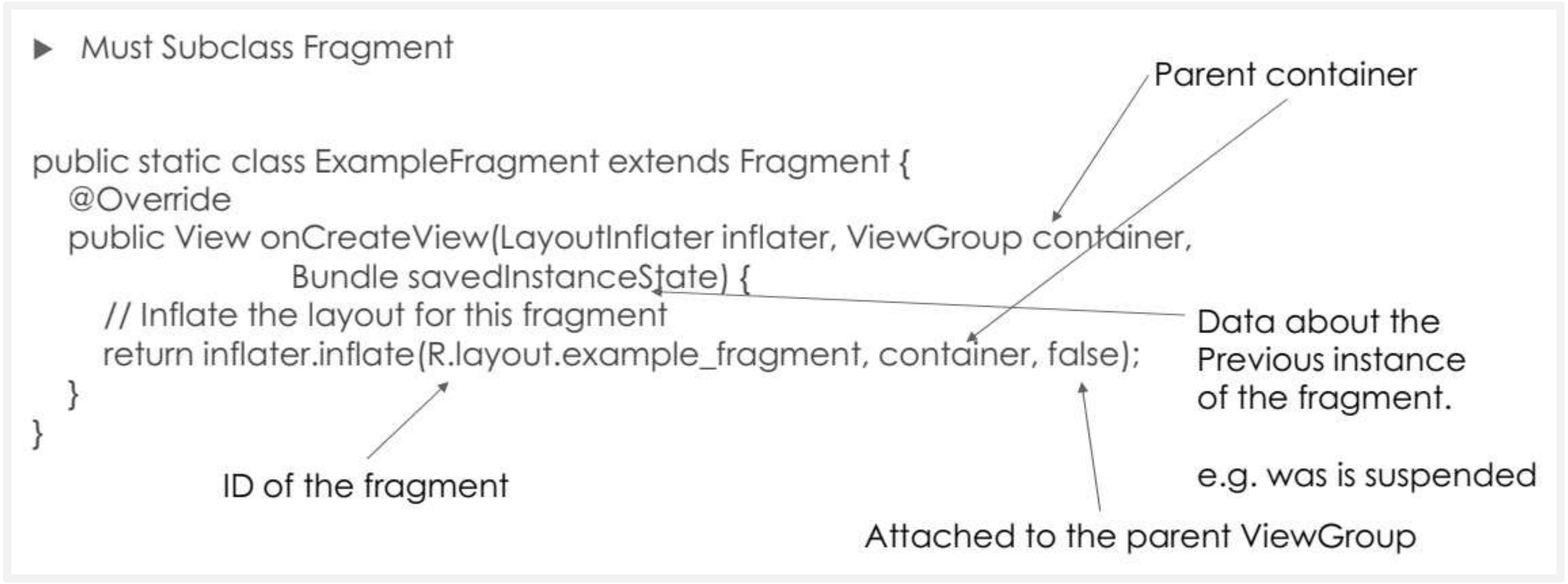
- Should not directly communicate with other fragments
  - ➢ Leave it to the host activity
  - ➢ Therefore: Fragment should define an interface for the to implement
  - ➢ Fragment can check if its host activity is compatible (i.e. has it implemented the interface)

Design Philosophy


Tablet
Selecting an item updates Fragment B
Activity A contains Fragment A and Fragment B
Handset
Selecting an item starts Activity B
Activity A contains Fragment A
Activity B contains Fragment B

https://developer.android.com/guide/components/fragments.html

Let’s Create a Fragment



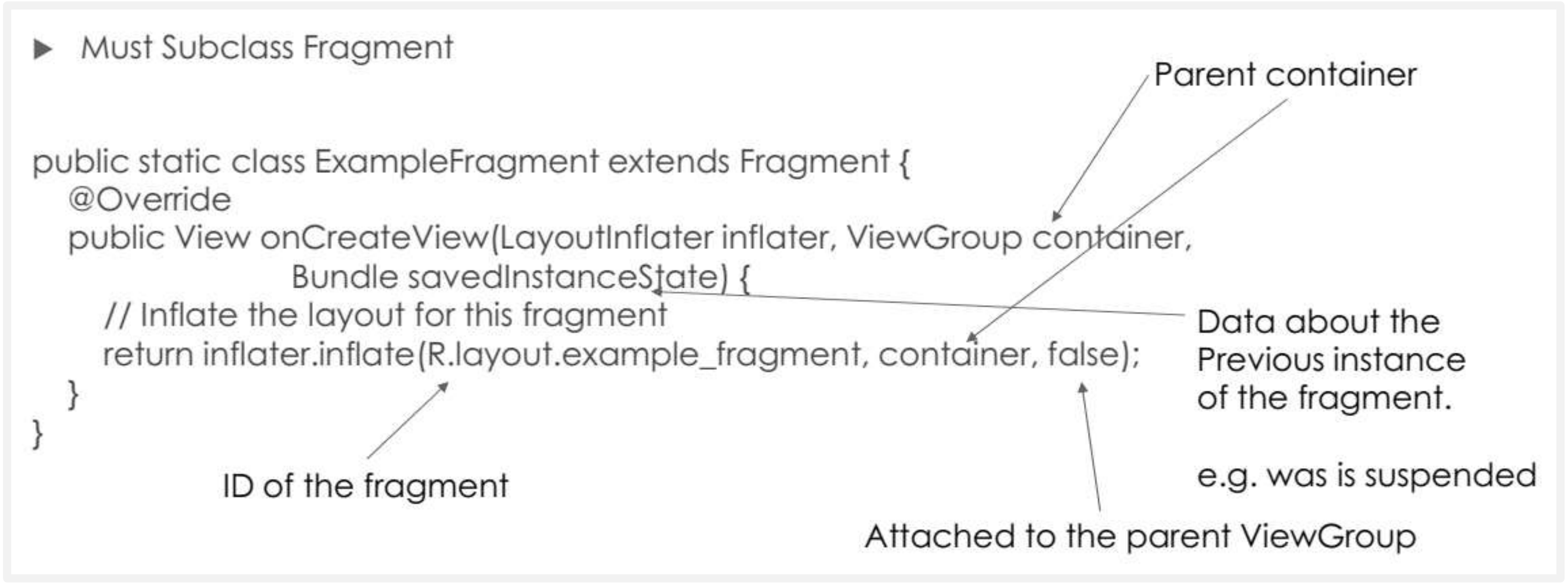
- Must Subclass Fragment

```
public static class ExampleFragment extends Fragment {
   @Override
   public View onCreateView(LayoutInflater inflater, ViewGroup container,
                            Bundle savedInstanceState) {
      // Inflate the layout for this fragment
      return inflater.inflate(R.layout.example_fragment, container, false);
   }
}
```

Parent container
Data about the Previous instance of the fragment.
e.g. was is suspended
ID of the fragment
Attached to the parent ViewGroup

https://developer.android.com/guide/components/fragments.html

Let’s Create a Fragment



- Must Subclass Fragment

```
public static class ExampleFragment extends Fragment {
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.example_fragment, container, false);
    }
}
```

Parent container
Data about the Previous instance of the fragment.
e.g. was is suspended
ID of the fragment
Attached to the parent ViewGroup

https://developer.android.com/guide/components/fragments.html

Fragment UI

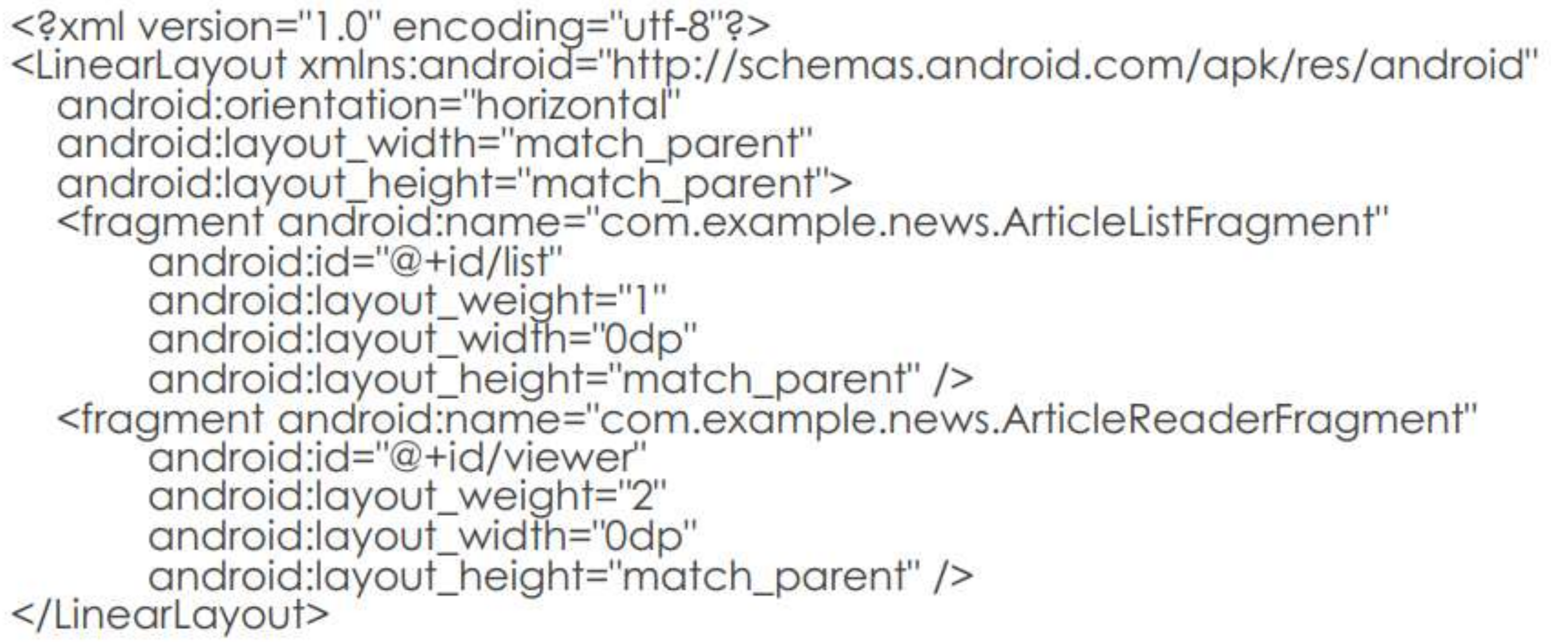
- The XML layout is placed in the project res/layout folder

https://developer.android.com/guide/components/fragments.html

Adding a fragment to an activity

Usually, a fragment contributes a portion of UI to the host activity, which is embedded as a part of the activity's overall view hierarchy. There are two ways you can add a fragment to the activity layout:

Declare the fragment inside the activity's layout file

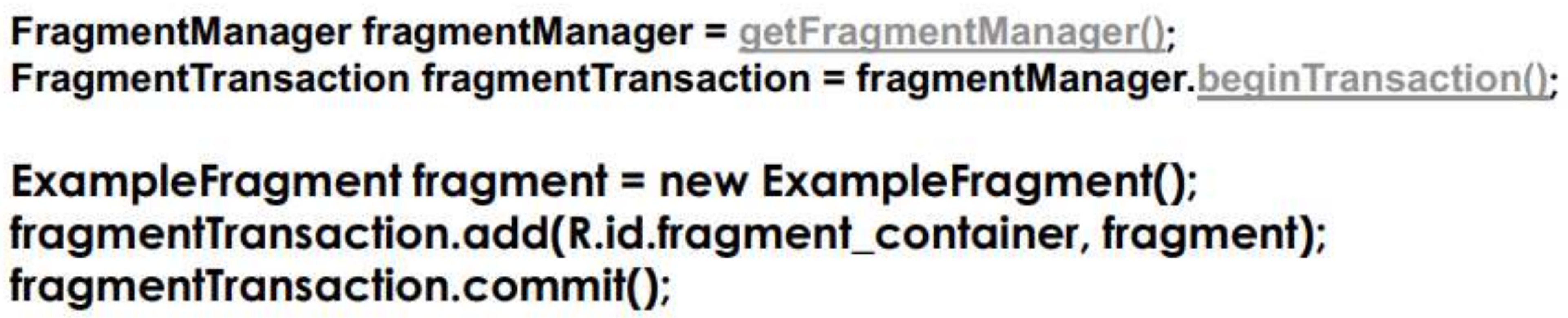


```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="horizontal"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
   <fragment android:name="com.example.news.ArticleListFragment"
         android:id="@+id/list"
         android:layout_weight="1"
         android:layout_width="0dp"
         android:layout_height="match_parent" />
   <fragment android:name="com.example.news.ArticleReaderFragment"
         android:id="@+id/viewer"
         android:layout_weight="2"
         android:layout_width="0dp"
         android:layout_height="match_parent" />
</LinearLayout>
```

https://developer.android.com/guide/components/fragments.html

Adding a fragment to an activity (cont)

Or, programmatically add the fragment to an existing ViewGroup.
At any time while your activity is running, you can add fragments to your activity layout. You simply need to specify a ViewGroup in which to place the fragment.



```
FragmentManager fragmentManager = getFragmentManager();
FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();

ExampleFragment fragment = new ExampleFragment();
fragmentTransaction.add(R.id.fragment_container, fragment);
fragmentTransaction.commit();
```

https://developer.android.com/guide/components/fragments.html

Fragment Management

To manage the fragments in your activity, you need to use **FragmentManager**. To get it, call **getSupportFragmentManager()** from your activity.Obtain it from the host activity.

FragmentManager can;

- Get findFragmentById()
- Pop off the back stack popBackStack()
- Listening to the back stack addOnBackStackChangedListener()

https://developer.android.com/guide/components/fragments.html

Fragment Management

To manage the fragments in your activity, you need to use **FragmentManager**. To get it, call **getSupportFragmentManager()** from your activity.Obtain it from the host activity.

FragmentManager can;

- Get findFragmentById()
- Pop off the back stack popBackStack()
- Listening to the back stack addOnBackStackChangedListener()

https://developer.android.com/guide/components/fragments.html

Communicating with the Activity

Fragment can access the host activity instance

```
getActivity().findViewById(R.id.list); // get the UI element from the parent
```

Likewise, your activity can call methods in the fragment by acquiring a reference to the Fragment from FragmentManager, using findFragmentById() or findFragmentByTag().

```
ExampleFragment fragment = (ExampleFragment)
getSupportFragmentManager().findFragmentById(R.id.example_fragment);
```

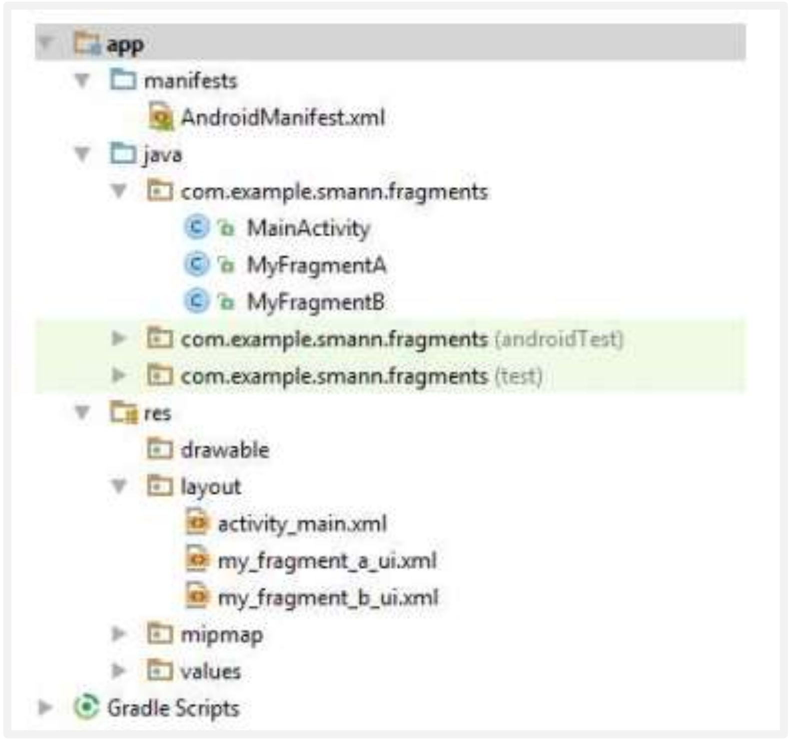
https://developer.android.com/guide/components/fragments.html

Fragments already built

There are many prebuilt fragments!
They extend the base Fragment class: e.g.,:
DialogFragment
Displays a floating dialog. Using this class to create a dialog is a good alternative to using the dialog helper methods in the Activity class, because you can incorporate a fragment dialog into the back stack of fragments managed by the activity, allowing the user to return to a dismissed fragment.
ListFragment
Displays a list of items that are managed by an adapter (such as a SimpleCursorAdapter), similar to ListActivity. It provides several methods for managing a list view, such as the onListItemClick() callback to handle click events.
PreferenceFragment
Displays a hierarchy of Preference objects as a list, similar to PreferenceActivity. This is useful when creating a "settings" activity for your application.
MapFragment
Displays and inserts a google map object in the Activity

https://developer.android.com/guide/components/fragments.html

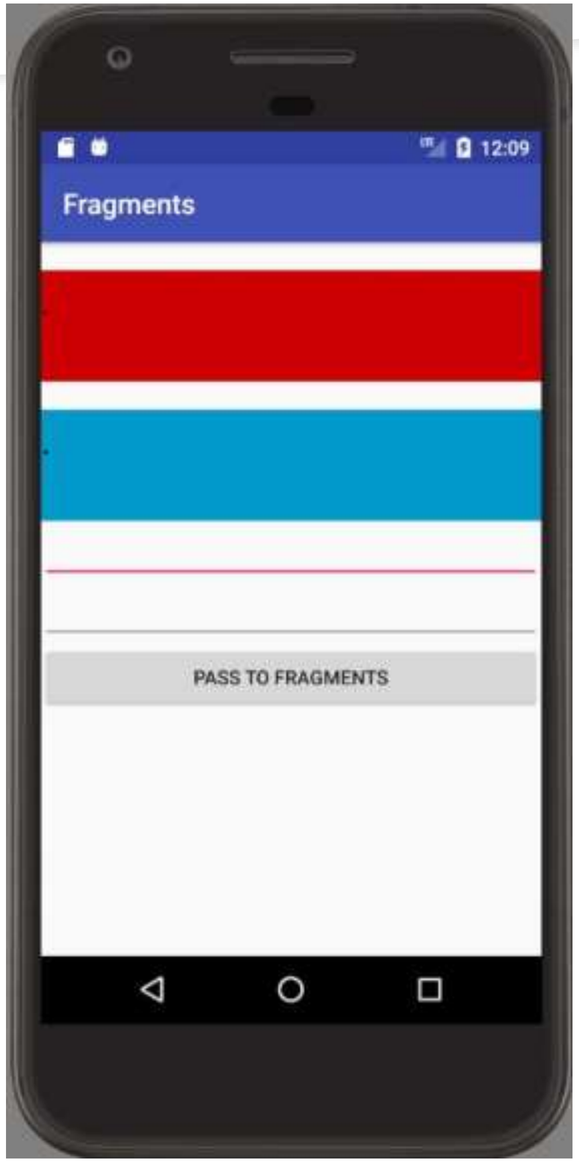
Code Example Using Fragments

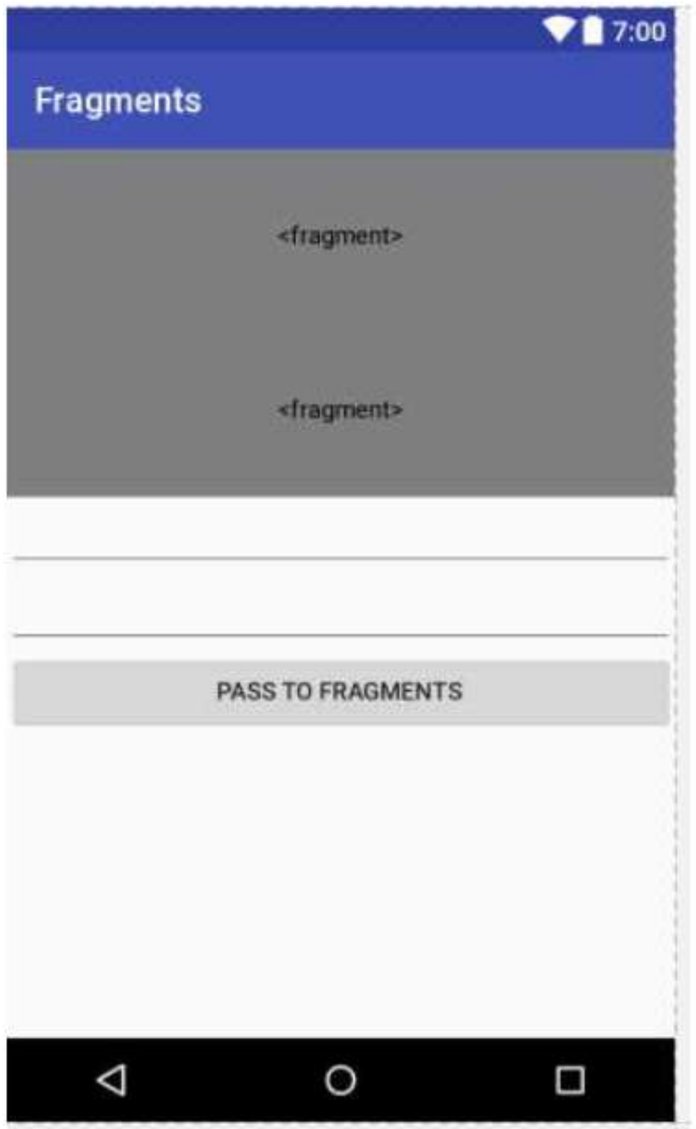


```
app
  manifests
    AndroidManifest.xml
  java
    com.example.smann.fragments
      MainActivity
      MyFragmentA
      MyFragmentB
    com.example.smann.fragments (androidTest)
    com.example.smann.fragments (test)
  res
    drawable
    layout
      activity_main.xml
      my_fragment_a_ui.xml
      my_fragment_b_ui.xml
    mipmap
    values
Gradle Scripts
```

Lets have an activity with 2 fragments

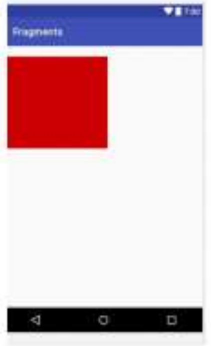
- **MyFragmentA** performs a multiplication of two numbers from the parent
- **MyFragmentB** performs an addition of two numbers from the parent



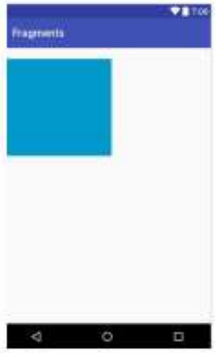


```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/container"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    tools:context="com.example.smann.fragments.MainActivity">

    <fragment
        android:id="@+id/fragment"
        android:name="com.example.smann.fragments.MyFragmentA"
        android:layout_width="match_parent"
        android:layout_height="100dp"
        tools:layout_editor_absoluteX="102dp"
        tools:layout_editor_absoluteY="177dp" />

    <fragment
        android:id="@+id/fragment2"
        android:name="com.example.smann.fragments.MyFragmentB"
        android:layout_width="match_parent"
        android:layout_height="100dp"
        tools:layout_editor_absoluteX="102dp"
        tools:layout_editor_absoluteY="305dp" />

    <EditText
        android:id="@+id/firstOperandET"
        android:layout_width="match_parent"
        android:layout_height="44dp"
        android:ems="10"
        android:inputType="textPersonName"
        android:text=""
        tools:layout_editor_absoluteX="16dp"
        tools:layout_editor_absoluteY="16dp" />

    <EditText
        android:id="@+id/secondOperandET"
        android:layout_width="match_parent"
        android:layout_height="44dp"
        android:ems="10"
        android:inputType="textPersonName"
        android:text=""
        tools:layout_editor_absoluteX="130dp"
        tools:layout_editor_absoluteY="16dp" />

<Button
        android:id="@+id/button"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Pass to Fragments"
        tools:layout_editor_absoluteX="240dp"
        tools:layout_editor_absoluteY="16dp" />

</LinearLayout>
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="200dp"
    android:layout_height="200dp"
    android:orientation="vertical">

    <TextView
        android:id="@+id/resultText"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_gravity="center_horizontal|center_vertical"
        android:layout_marginTop="20dip"
        android:background="@android:color/holo_red_dark"
        android:text="."
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:textSize="30dip" />

</LinearLayout>
```



```java
package com.example.smann.fragments;

import android.app.Fragment;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

/**
 * Created by smann on 06/09/2017.
 */

public class MyFragmentA extends Fragment {

    TextView tv = null;

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.my_fragment_a_ui,
                container, false);

        tv = view.findViewById(R.id.resultText);

        return view;
    }

    public void doCalcDisplay(int a, int b) {

        // perform the multiplication operation and update the textview in this fragment

        tv.setText("" + (a*b));
    }

}
```

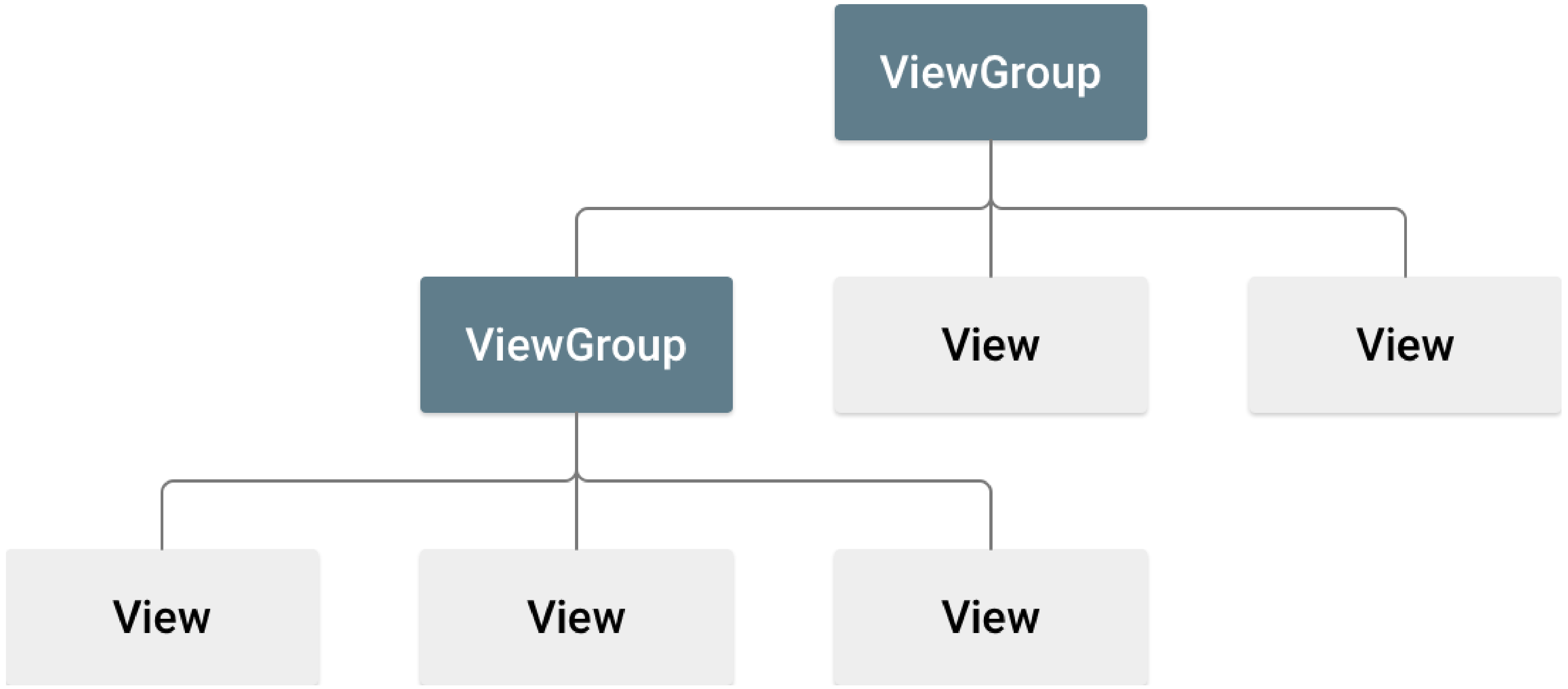


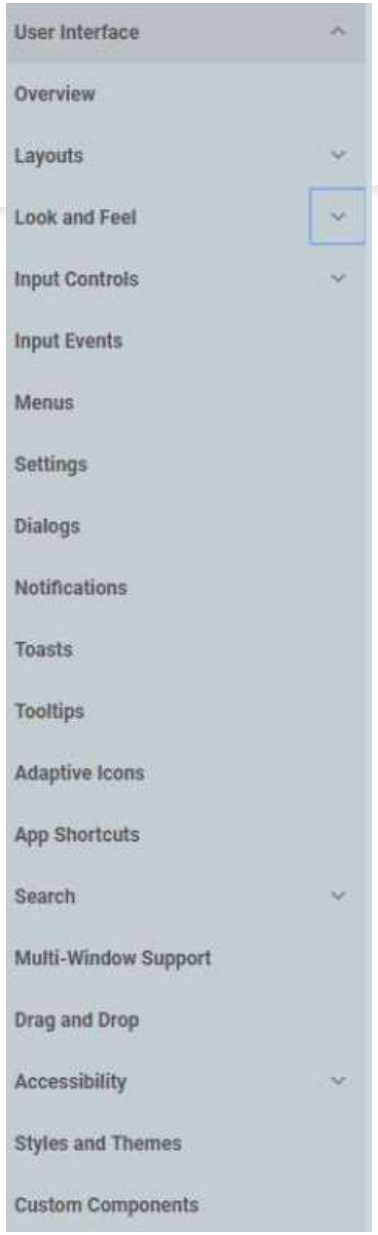
```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="200dp"
    android:layout_height="200dp"
    android:orientation="vertical">

    <TextView
        android:id="@+id/resultText"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_gravity="center_horizontal|center_vertical"
        android:layout_marginTop="20dip"
        android:background="@android:color/holo_blue_dark"
        android:text="."
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:textSize="30dip" />

</LinearLayout>
```

```java
package com.example.smann.fragments;

import android.app.Fragment;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

/**
 * Created by smann on 06/09/2017.
 */

public class MyFragmentB extends Fragment{

    TextView tv = null;


    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.my_fragment_b_ui,
                container, false);

        tv = view.findViewById(R.id.resultText);

        return view;
    }


    public void doCalcDisplay(int a, int b) {

        // perform the addition operation and update the textview in this fragment
        tv.setText("" + (a+b));
    }

}
```

```
package com.example.smann.fragments;

import android.app.Fragment;
import android.app.FragmentTransaction;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    Button resButton = null;
    EditText opA = null;
    EditText opB = null;

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

       resButton = (Button) findViewById(R.id.button);
        opA = (EditText) findViewById(R.id.firstOperandET);
        opB = (EditText) findViewById(R.id.secondOperandET);

        resButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                MyFragmentA fA = (MyFragmentA) getFragmentManager().findFragmentById(R.id.fragment);
                MyFragmentB fB = (MyFragmentB) getFragmentManager().findFragmentById(R.id.fragment2);

                int a = Integer.parseInt(opA.getText().toString());
                int b = Integer.parseInt(opB.getText().toString());

                if(a !=0 && b!=0) {
                    fA.doCalcDisplay(a, b);
                    fB.doCalcDisplay(a, b);
                } else {
                    Toast.makeText(getApplicationContext(),"Please enter non-zero integers", Toast.LENGTH_SHORT);
                }

            }
        });
    }
}
```

UI in Android

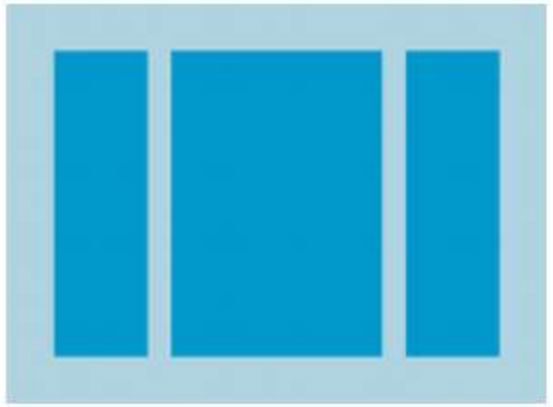




https://developer.android.com/guide/topics/ui

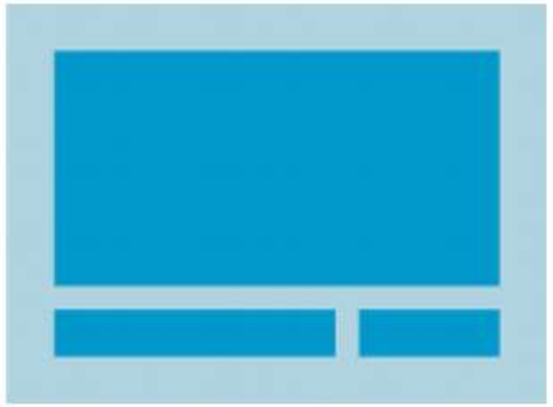
UI Layouts

Linear Layout



A layout that organizes its children into a single horizontal or vertical row. It creates a scrollbar if the length of the window exceeds the length of the screen.

Relative Layout



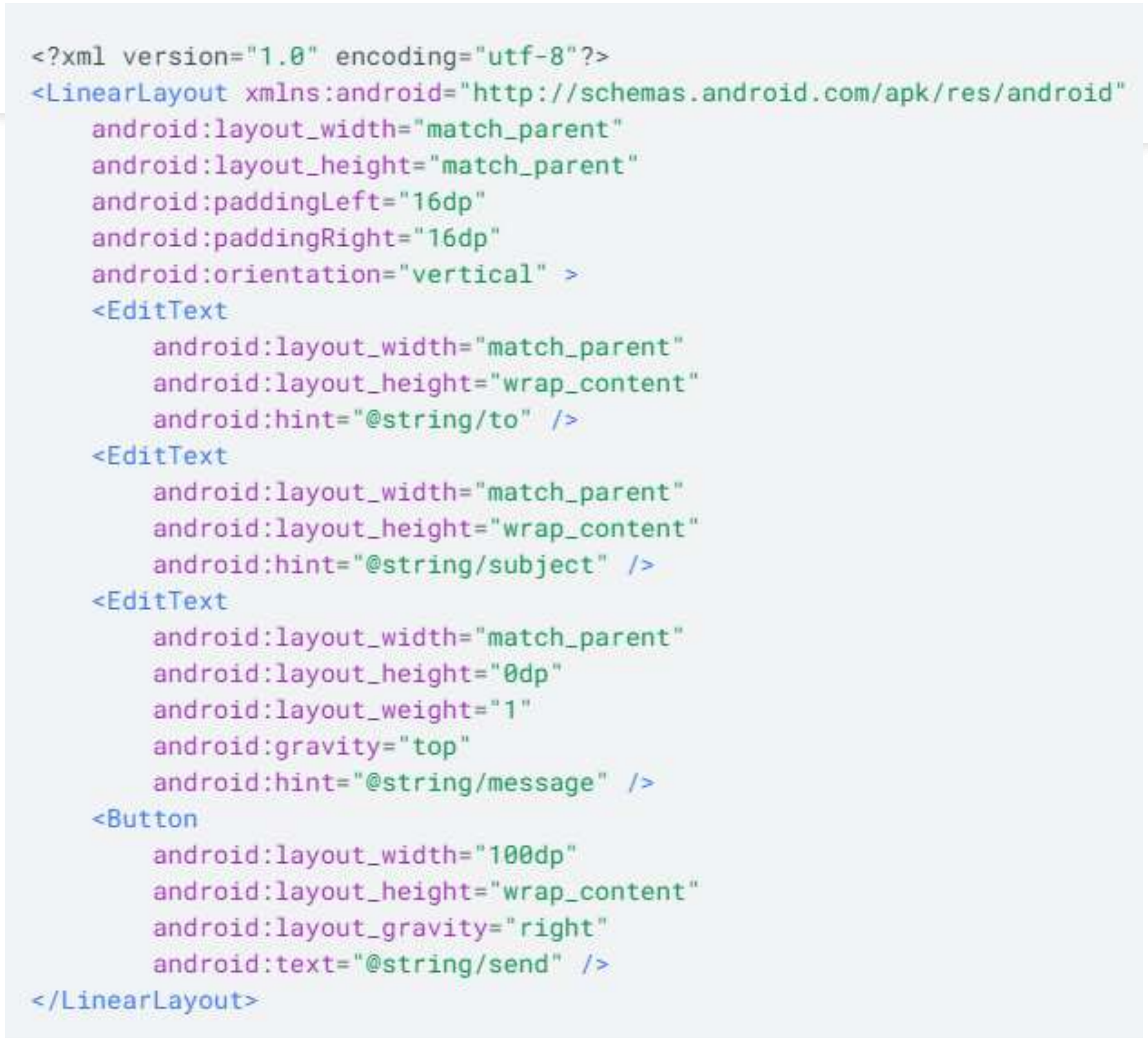
Enables you to specify the location of child objects relative to each other (child A to the left of child B) or to the parent (aligned to the top of the parent).

Web View

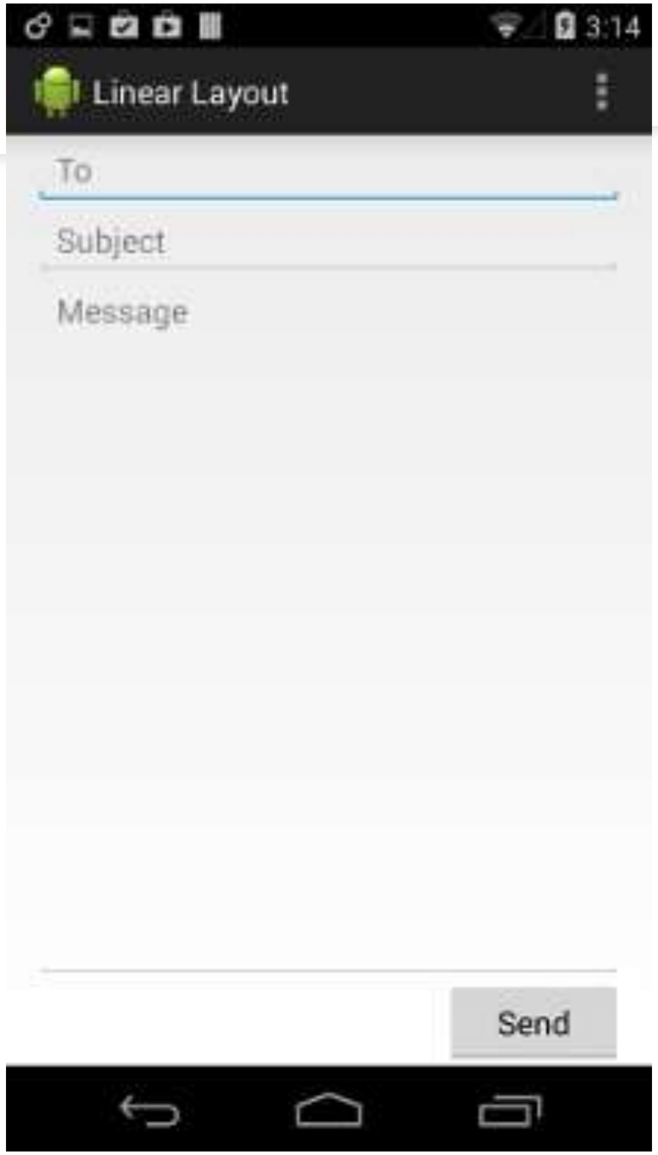
Displays web pages.

https://developer.android.com/guide/topics/ui/declaring-layout

Linear Layout



```
<?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:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:orientation="vertical" >
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/to" />
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="@string/subject" />
    <EditText
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:gravity="top"
        android:hint="@string/message" />
    <Button
        android:layout_width="100dp"
        android:layout_height="wrap_content"
        android:layout_gravity="right"
        android:text="@string/send" />
</LinearLayout>
```

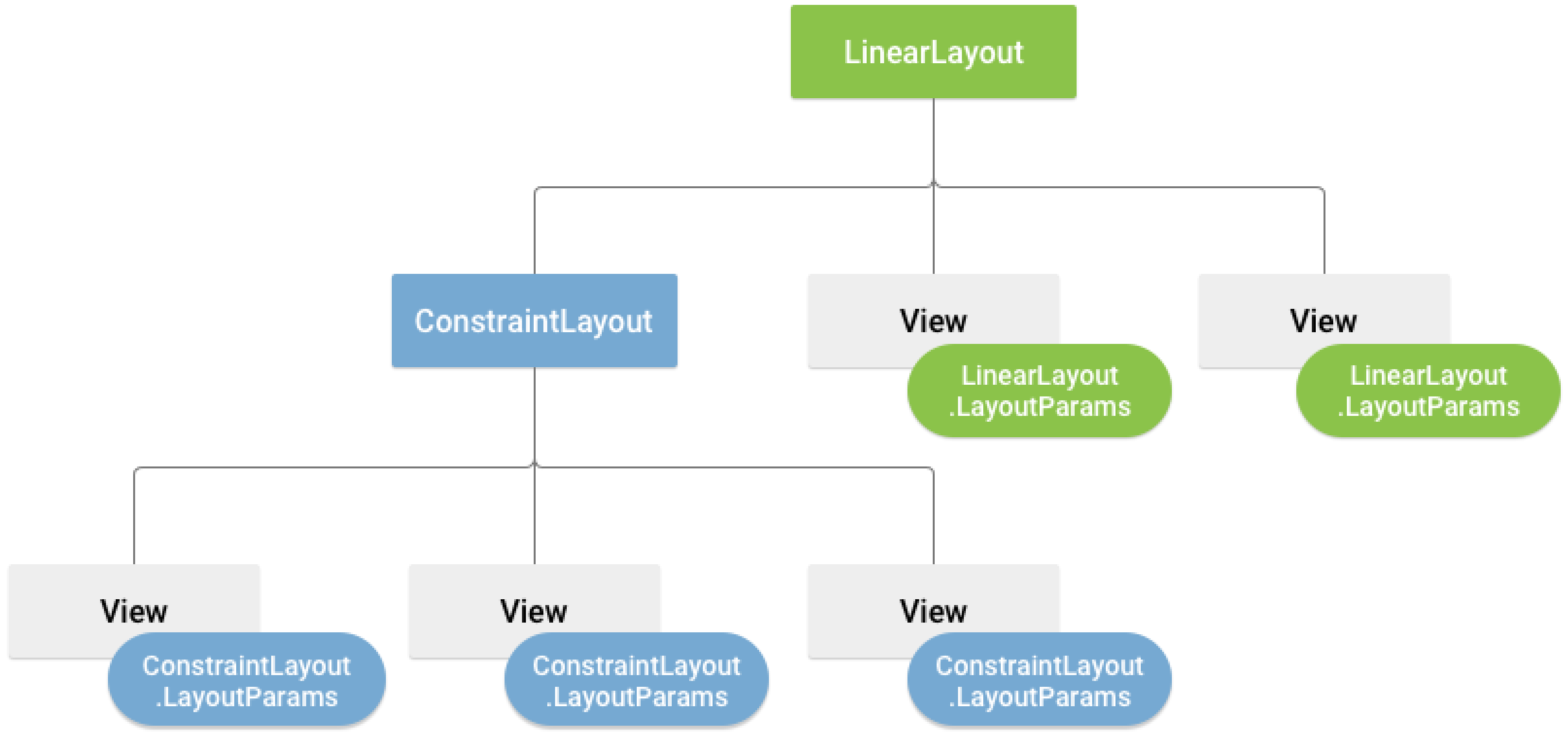


https://developer.android.com/guide/topics/ui/layout/linear

Linear Layout

LinearLayouts

- wrap_content tells your view to size itself to the dimensions required by its content.
- match_parent tells your view to become as big as its parent view group will allow.



LinearLayout
ConstraintLayout
View
LinearLayout .LayoutParams
View
LinearLayout .LayoutParams
View
ConstraintLayout .LayoutParams
View
ConstraintLayout .LayoutParams
View
ConstraintLayout .LayoutParams

https://developer.android.com/guide/topics/ui/declaring-layout

UI Layouts – Constraint Layout

https://youtu.be/XamMbnzI5vE https://developer.android.com/training/constraint-layout

WebView

1

```
<?xml version="1.0" encoding="utf-8"?>
<WebView  xmlns:android="http://schemas.android.com/apk/res/android"
   android:id="@+id/webview"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
/>
```

2

```
WebView myWebView = (WebView)
findViewById(R.id.webview);
myWebView.loadUrl("http://www.example.com");
```

3

```
<manifest ... >
   <uses-permission
android:name="android.permission.INTERNET" />
   ...
</manifest>
```

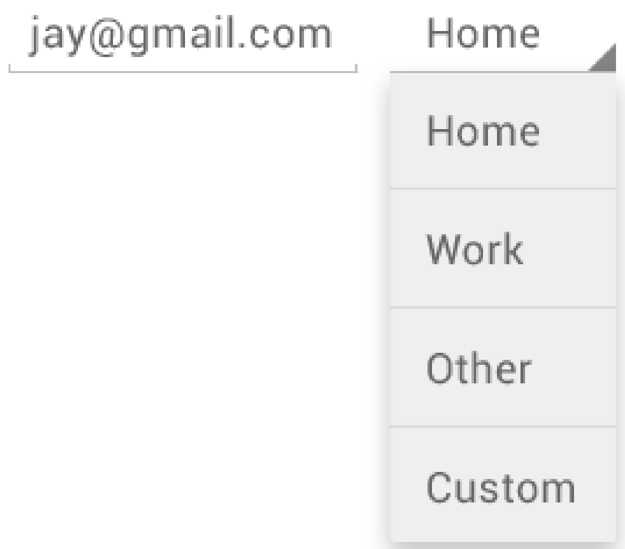
Can use JavaScript and
Bind JS to Android code

https://developer.android.com/guide/topics/ui/declaring-layout

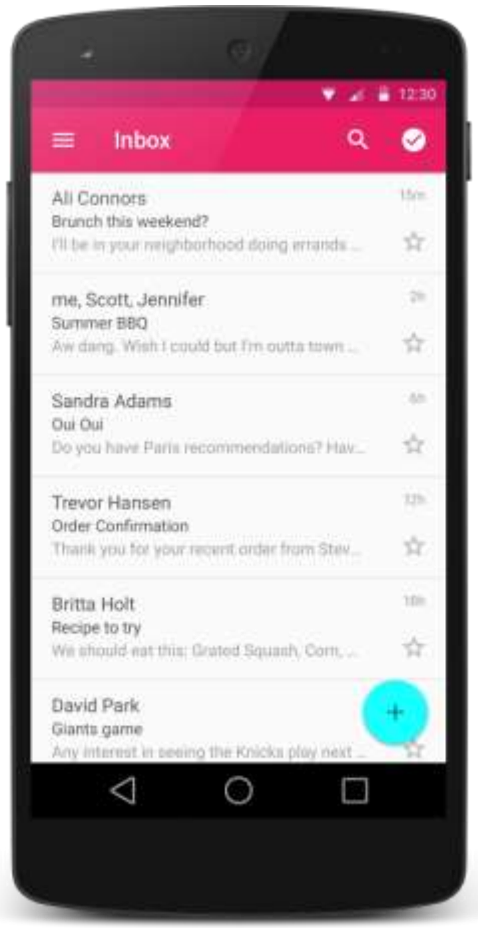
Dynamic Layouts with an Adapter

https://developer.android.com/guide/topics/ui/layout/recyclerview

https://developer.android.com/guide/topics/ui/controls/spinner



Spinner



Ali Connors
Brunch this weekend?
I'll be in your neighborhood doing errands ...

me, Scott, Jennifer
Summer BBQ
Aw dang. Wish I could but I'm outta town ...

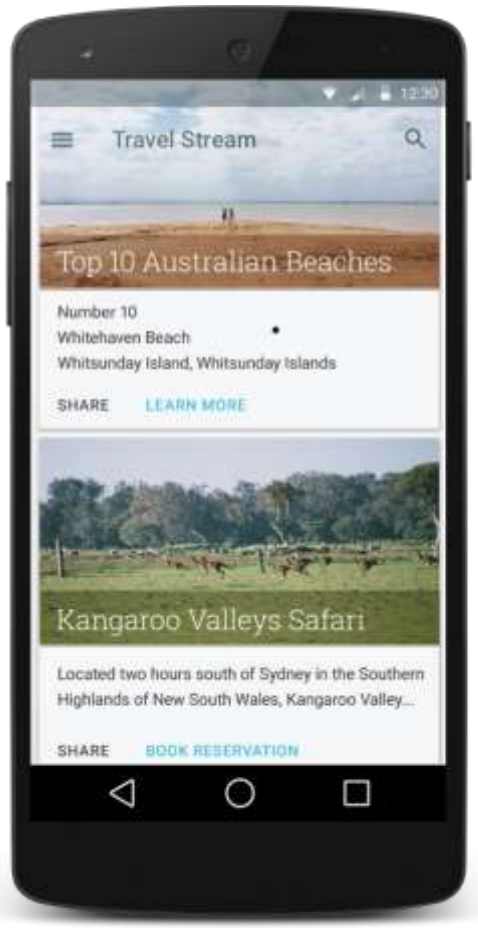
Sandra Adams
Oui Oui
Do you have Paris recommendations? Hav...

Trevor Hansen
Order Confirmation
Thank you for your recent order from Stev...

Britta Holt
Recipe to try
We should eat this: Grated Squash, Corn, ...

David Park
Giants game
Any interest in seeing the Knicks play next ...

RecyclerView



Number 10
Whitehaven Beach
Whitsunday Island, Whitsunday Islands

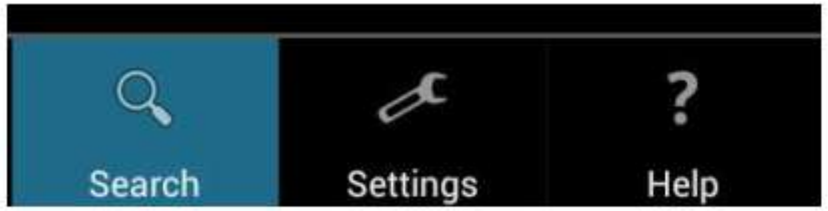
Located two hours south of Sydney in the Southern Highlands of New South Wales, Kangaroo Valley...

CardView

Menus

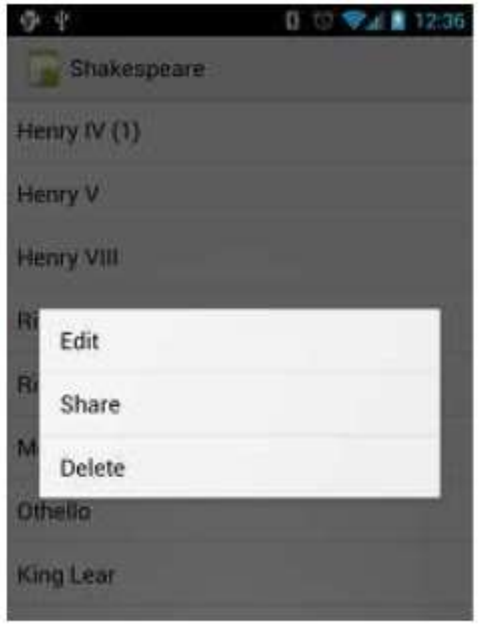
Options menu

Actions that have a global impact on the app, such as "Search", "Compose email", and "Settings"...

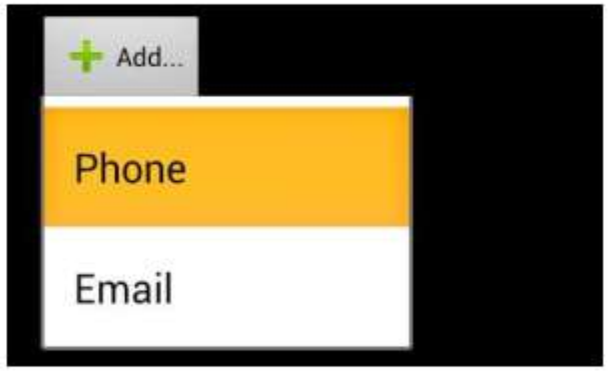


Context menu

A floating menu for a long-click on a selected content.



Popup menu



Defining a Menu in XML

Create an XML file inside the res/menu/ directory.

```
<?xml version="1.0" encoding="utf-8"?>
<menu
xmlns:android="http://schemas.android.com/apk/res/android">
   <item android:id="@+id/new_game"
        android:icon="@drawable/ic_new_game"
        android:title="@string/new_game"
        android:showAsAction="ifRoom"/>
   <item android:id="@+id/help"
        android:icon="@drawable/ic_help"
        android:title="@string/help" />
</menu>
```

https://developer.android.com/guide/topics/ui/menus

Loading/Inflating an Options Menu

Assume that my_options_menu.xml is an XML file defining the options menu.

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
   MenuInflater inflater = getMenuInflater();
   inflater.inflate(R.menu.game_menu, menu);
   return true;
}
```

https://developer.android.com/guide/topics/ui/menus

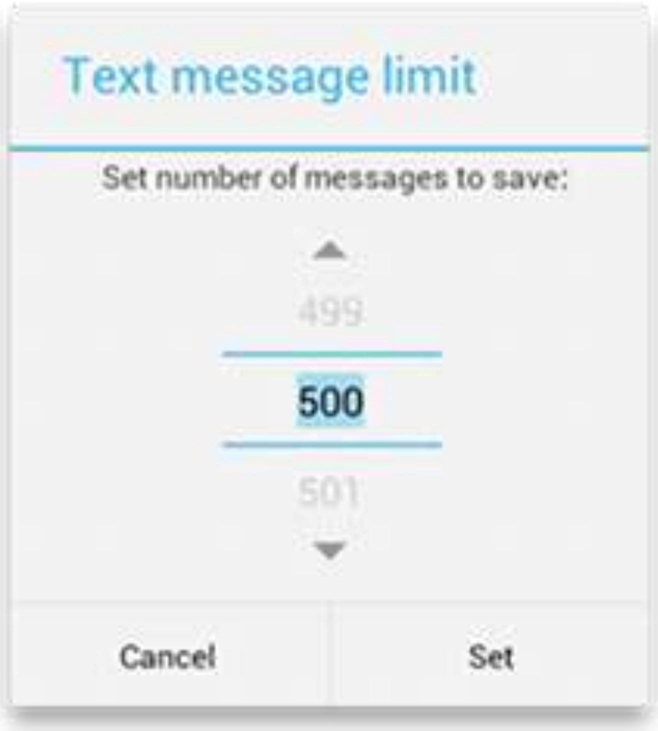
Handling Click Events on a Menu

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle item selection
    switch (item.getItemId()) {
        case R.id.new_game:
            newGame();
            return true;
        case R.id.help:
            showHelp();
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

https://developer.android.com/guide/topics/ui/menus

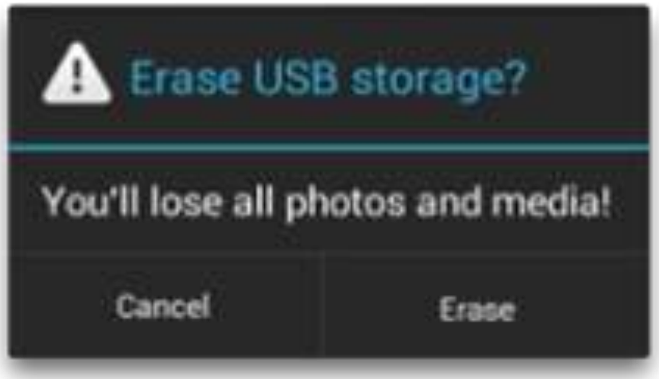
Dialogs

- AlertDialog
- ProgressDialog
- DatePickerDialog
- TimePickerDialog
- Your custom dialog...



Text message limit

Set number of messages to save:



Erase USB storage?

You'll lose all photos and media!

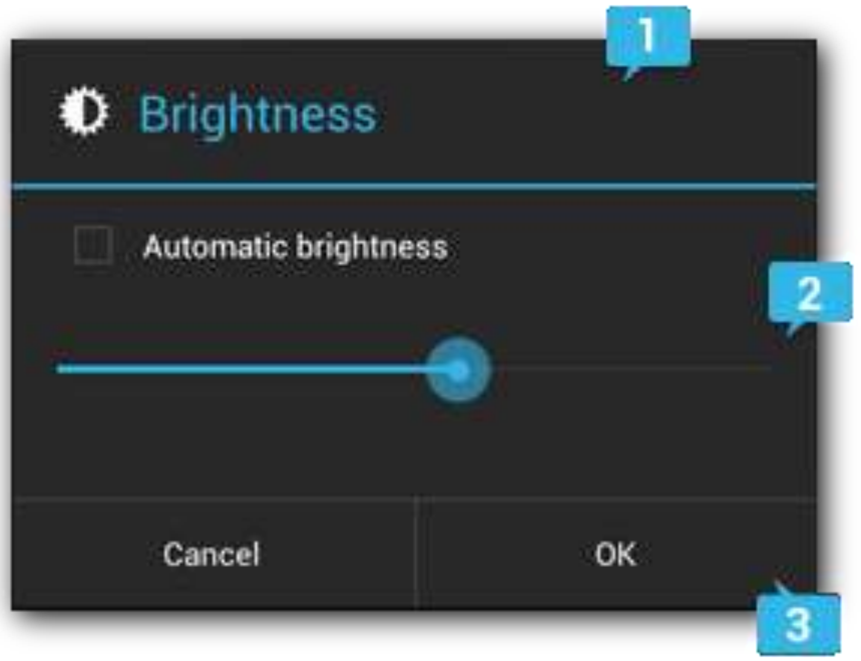
https://developer.android.com/guide/topics/ui/dialogs
https://material.io/components/dialogs#anatomy

Building an Alert Dialog

1. Title
This is optional and should be used only when the content area is occupied by a detailed message, a list, or custom layout. If you need to state a simple message or question (such as the dialog in figure 1), you don't need a title.

2. Content area
This can display a message, a list, or other custom layout.

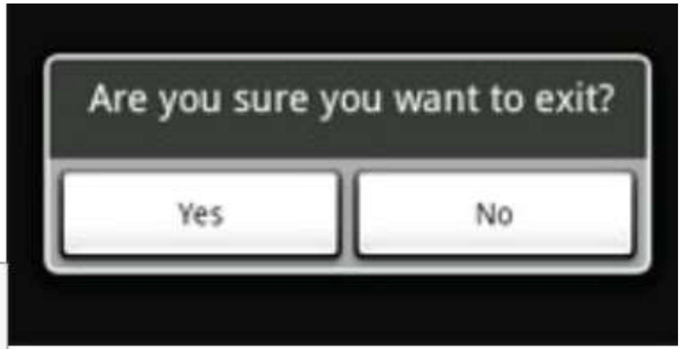
3. Action buttons
There should be no more than three action buttons in a dialog.



https://developer.android.com/guide/topics/ui/dialogs#AlertDialog

Building an Alert Dialog

```java
AlertDialog.Builder builder = new AlertDialog.Builder(MyActivity.this);
builder.setMessage("Are you sure you want to exit?")
       .setCancelable(false)
       .setPositiveButton("Yes", new DialogInterface.OnClickListener() {
         public void onClick(DialogInterface dialog, int id) {
           MyActivity.this.finish();
         }
       }).setNegativeButton("No", new DialogInterface.OnClickListener() {
         public void onClick(DialogInterface dialog, int id) {
           dialog.cancel();
         }
       });
AlertDialog dialog = builder.create();
```



Building an Alert Dialog

Sr.No	Method type & description
1	**setIcon(Drawable icon)** This method set the icon of the alert dialog box.
2	**setCancelable(boolean cancel able)** This method sets the property that the dialog can be cancelled or not
3	**setMessage(CharSequence message)** This method sets the message to be displayed in the alert dialog
4	**setMultiChoiceItems(CharSequence[] items, boolean[] checkedItems, DialogInterface.OnMultiChoiceClickListener listener)** This method sets list of items to be displayed in the dialog as the content. The selected option will be notified by the listener
5	**setOnCancelListener(DialogInterface.OnCancelListener onCancelListener)** This method Sets the callback that will be called if the dialog is cancelled.
6	**setTitle(CharSequence title)** This method set the title to be appear in the dialog

https://www.tutorialspoint.com/android/android_alert_dialoges.htm

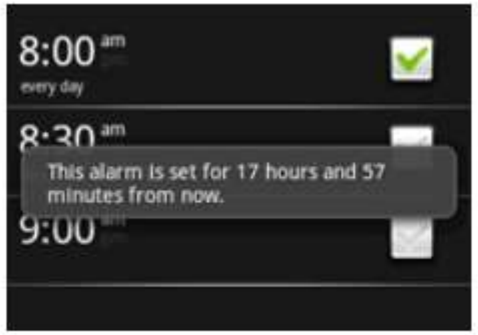
Notifications

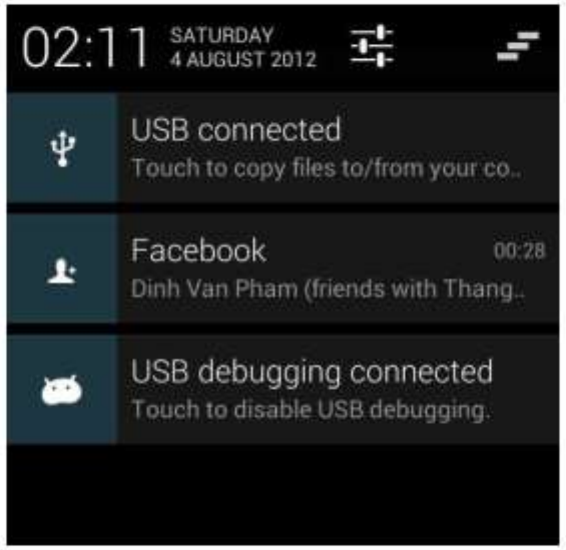
Toast Notification

for brief messages/reports that come from the background operations.

Status Notification

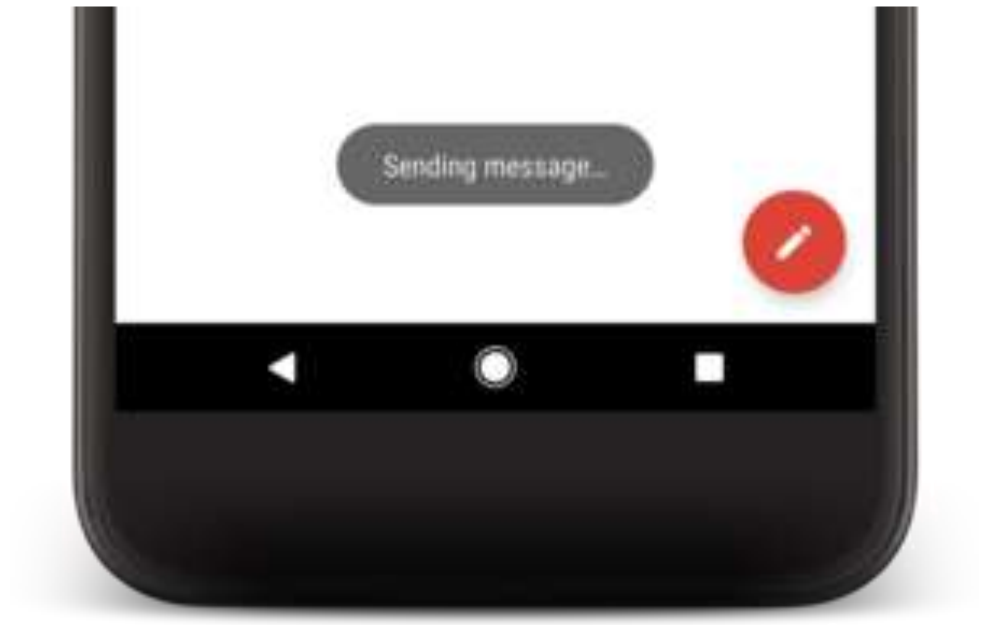
for persistent reminders that come from the background operations and request the user's response


8:00 am
every day
8:30 am
This alarm is set for 17 hours and 57 minutes from now.
9:00


02:11 SATURDAY 4 AUGUST 2012
USB connected
Touch to copy files to/from your co..
Facebook 00:28
Dinh Van Pham (friends with Thang..
USB debugging connected
Touch to disable USB debugging.

Toast Notifications

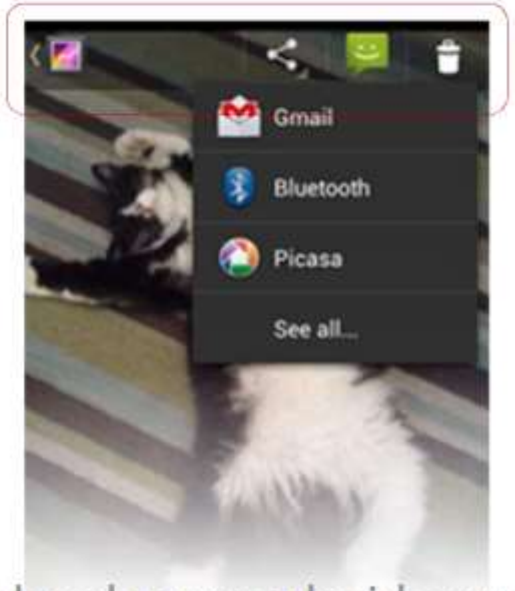
```
Context context = getApplicationContext();
CharSequence text = "Hello toast!";
int duration = Toast.LENGTH_SHORT;

Toast toast = Toast.makeText(context, text, duration);
toast.show();
```

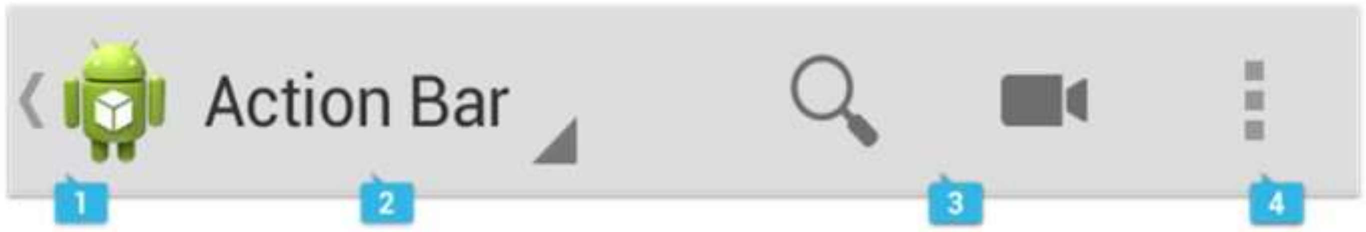


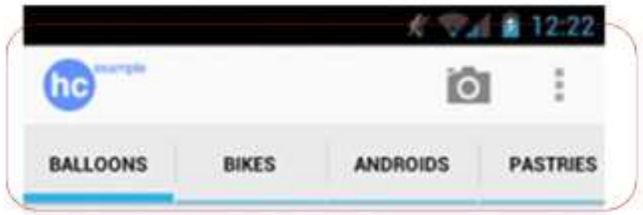
https://developer.android.com/guide/topics/ui/notifiers/toasts

Action Bar/App Bar

- Available in Android >=3.0 (API level >=11).
- Should use Action Bar in most activities that need to prominently present user actions or global navigation.







http://developer.android.com/guide/topics/ui/actionbar.html
http://developer.android.com/design/patterns/actionbar.html