Simple Views and Layouts

ACTIVITY

Model-View-Controller

- A model object holds the application's data and "business logic." In Android applications, model classes are generally custom classes you create.
- View objects know how to draw themselves on the screen and how to respond to user input, like touches. Android provides a wealth of configurable view classes.
- Controller objects tie the view and model objects together. They contain "application logic." In Android, a controller is typically a subclass of Activity, Fragment, or Service.

Activity states

An activity can exist in essentially three states:

Resumed
: The activity is in the foreground of the screen and has user focus. (This state is also sometimes referred to as "running".)

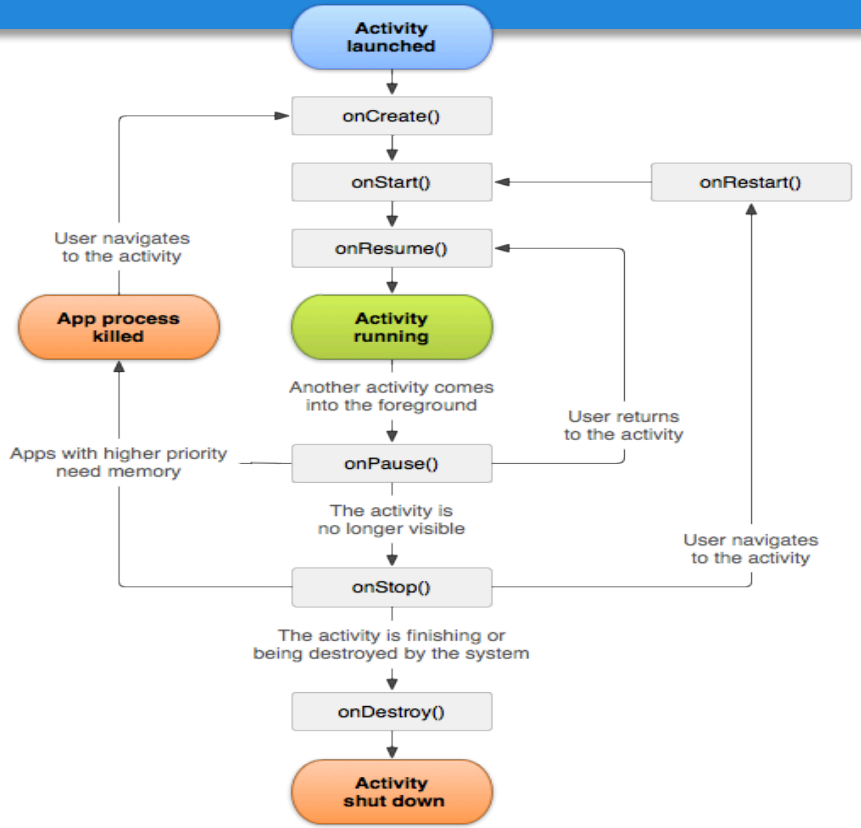
Paused
: Another activity is in the foreground and has focus, but this one is still visible. That is, another activity is visible on top of this one and that activity is partially transparent or doesn't cover the entire screen. A paused activity is completely alive (the `Activity` object is retained in memory, it maintains all state and member information, and remains attached to the window manager), but can be killed by the system in extremely low memory situations.

Stopped
: The activity is completely obscured by another activity (the activity is now in the "background"). A stopped activity is also still alive (the `Activity` object is retained in memory, it maintains all state and member information, but is *not* attached to the window manager). However, it is no longer visible to the user and it can be killed by the system when memory is needed elsewhere.

Activity lifecycle

Activity :

- An activity is an instance of Activity, a class in the Android SDK.
- An activity is responsible for managing user interaction with a screen of information.

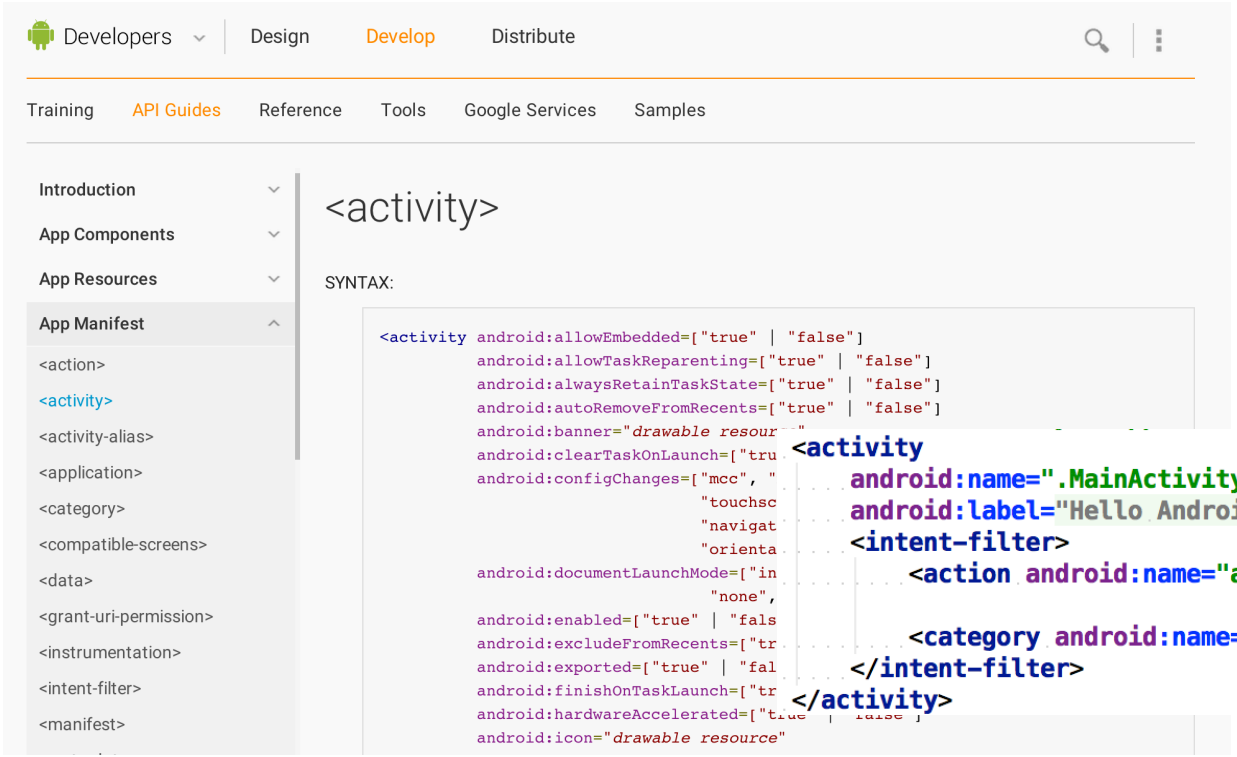


http://developer.android.com/reference/android/app/Activity.html
http://developer.android.com/guide/components/fundamentals.html

Lifecycle callbacks

```
public class ExampleActivity extends Activity {
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // The activity is being created.
    }
    @Override
    protected void onStart() {
        super.onStart();
        // The activity is about to become visible.
    }
    @Override
    protected void onResume() {
        super.onResume();
        // The activity has become visible (it is now "resumed").
    }
```

Configuring Activities




```
<activity
    android:name=".MainActivity"
    android:label="Hello Android" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

Hiding the activity title

```
import android.view.Window;


public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    //---hides the title bar---
    requestWindowFeature(Window.FEATURE_NO_TITLE);

    setContentView(R.layout.main);
    Log.d(tag, "In the onCreate() event");
}
```

Start activities

```
Intent intent = new Intent(this, SignInActivity.class);
startActivity(intent);
```

```
private void pickContact() {
    // Create an intent to "pick" a contact, as defined by the content provider URI
    Intent intent = new Intent(Intent.ACTION_PICK, Contacts.CONTENT_URI);
    startActivityForResult(intent, PICK_CONTACT_REQUEST);
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
```

Finish activities

public void **finish** ()

Call this when your activity is done and should be closed. The Acti

public void **finishActivity** (int requestCode)

Force finish another activity that you had previously started with

Parameters

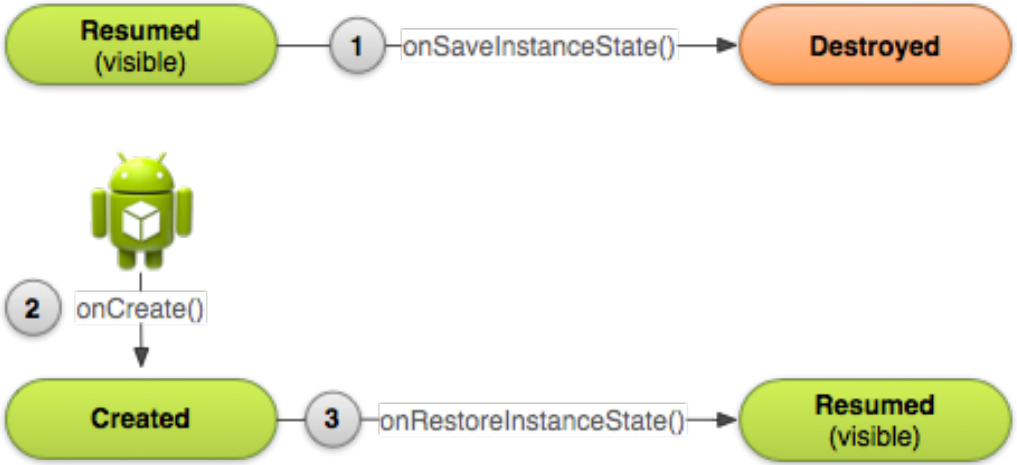
requestCode The request code of the activity that you had giv
they will all be finished.

Recreating an Activity

Your activity will be destroyed and recreated each time the user rotates the screen. When the screen changes orientation, the system destroys and recreates the foreground activity because the screen configuration has changed and your activity might need to load alternative resources (such as the layout).

http://developer.android.com/training/basics/activity-lifecycle/recreating.html

onSaveInstanceState



onSaveInstanceState is not fired when an activity is being unloaded from the stack (such as when the user pressed the back button), because there is no need to restore its state later.

Save Activity state

```
static final String STATE_SCORE = "playerScore";
static final String STATE_LEVEL = "playerLevel";
...

@Override
public void onSaveInstanceState(Bundle savedInstanceState) {
    // Save the user's current game state
    savedInstanceState.putInt(STATE_SCORE, mCurrentScore);
    savedInstanceState.putInt(STATE_LEVEL, mCurrentLevel);

    // Always call the superclass so it can save the view hierarchy state
    super.onSaveInstanceState(savedInstanceState);
}
```

http://developer.android.com/training/basics/activity-lifecycle/recreating.html

Restore Activity state

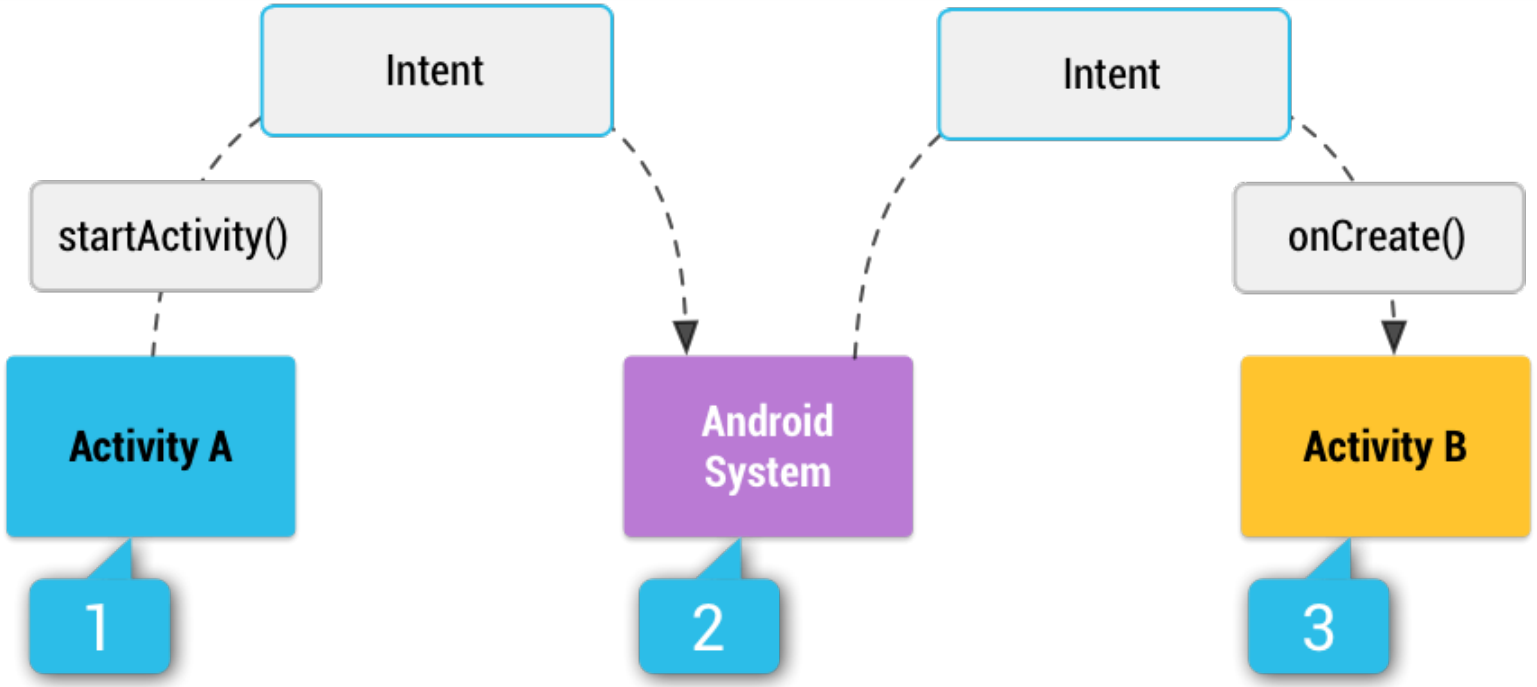
```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState); // Always call the superclass first

    // Check whether we're recreating a previously destroyed instance
    if (savedInstanceState != null) {
        // Restore value of members from saved state
        mCurrentScore = savedInstanceState.getInt(STATE_SCORE);
        mCurrentLevel = savedInstanceState.getInt(STATE_LEVEL);
    } else {
        // Probably initialize members with default values for a new instance
    }
    ...
}
```

INTENT

Intent use-cases

- To start an activity: startActivity(), startActivityForResult()
- To start a service: startService()
- To deliver a broadcast: A broadcast is a message that any app can receive, sendBroadcast()

Intent delivery



http://developer.android.com/guide/components/intents-filters.html#ExampleExplicit

Explicit intent

```
// Executed in an Activity, so 'this' is the Context
// The fileUrl is a string URL, such as "http://www.example.com/image.png"
Intent downloadIntent = new Intent(this, DownloadService.class);
downloadIntent.setData(Uri.parse(fileUrl));
startService(downloadIntent);
```

An explicit intent is one that you use to launch a specific app component, such as a particular activity or service in your app.

Implicit intent

```
// Create the text message with a string
Intent sendIntent = new Intent();
sendIntent.setAction(Intent.ACTION_SEND);
sendIntent.putExtra(Intent.EXTRA_TEXT, textMessage);
sendIntent.setType(HTTP.PLAIN_TEXT_TYPE); // "text/plain" MIME type

// Verify that the intent will resolve to an activity
if (sendIntent.resolveActivity(getPackageManager()) != null) {
    startActivity(sendIntent);
}
```

An implicit intent specifies an action that can invoke any app on the device able to perform the action.

Intent filters

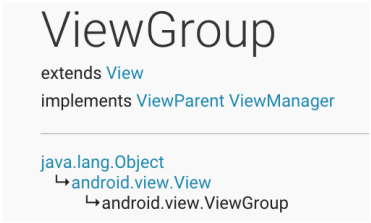
```
<activity android:name="ShareActivity">
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="text/plain"/>
    </intent-filter>
</activity>
```

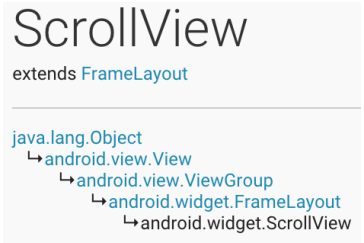
```xml
<activity android:name="MainActivity">
    <!-- This activity is the main entry, should appear in app launcher -->
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>

<activity android:name="ShareActivity">
    <!-- This activity handles "SEND" actions with text data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="text/plain"/>
    </intent-filter>
    <!-- This activity also handles "SEND" and "SEND_MULTIPLE" with media data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <action android:name="android.intent.action.SEND_MULTIPLE"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="application/vnd.google.panorama360+jpg"/>
        <data android:mimeType="image/*"/>
        <data android:mimeType="video/*"/>
    </intent-filter>
</activity>
```

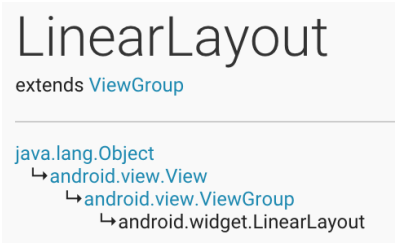
LAYOUT

ViewGroup

- View Groups — View Groups are extensions of the View class that can contain multiple child Views.







Common layouts

Layouts

- FrameLayout
- LinearLayout (Horizontal)
- LinearLayout (Vertical)
- TableLayout
- TableRow
- GridLayout
- RelativeLayout

Containers

- RadioGroup
- ListView
- GridView
- ExpandableListView
- ScrollView
- HorizontalScrollView
- SearchView
- TabHost
- ~~SlidingDrawer~~
- ~~Gallery~~
- VideoView
- ~~TwoLineListItem~~
- DialerFilter

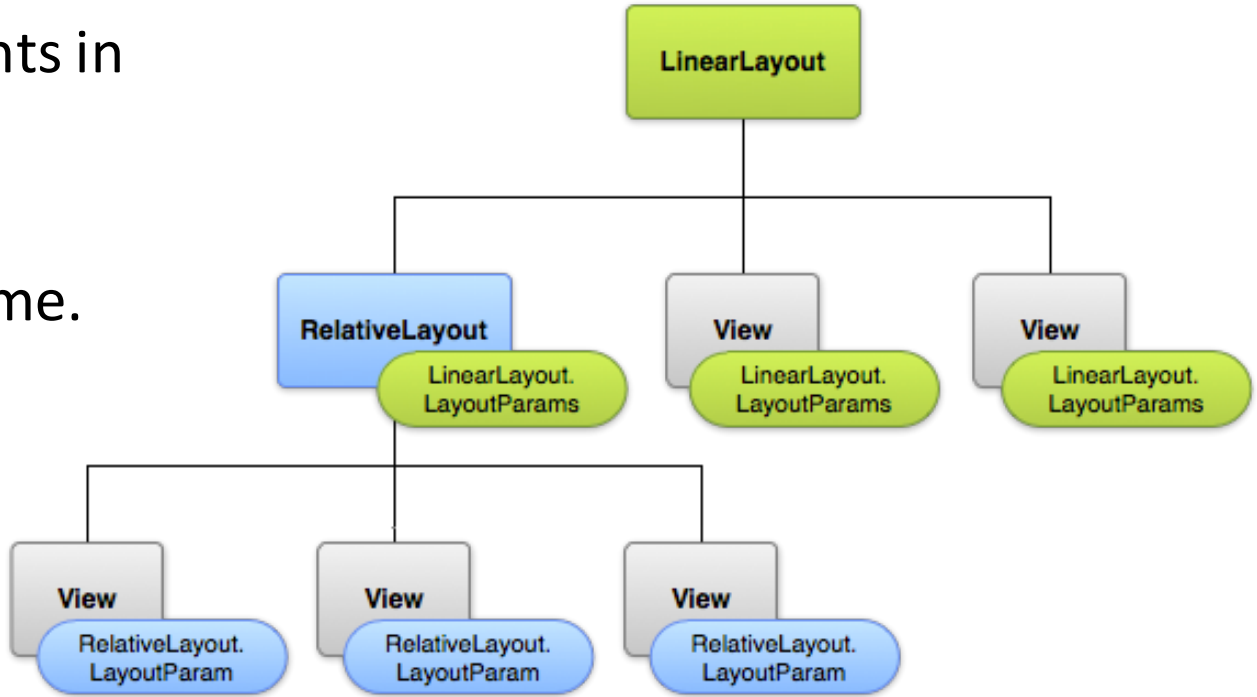
Layout resource

```
<?xml version="1.0" encoding="utf-8"?>
<ViewGroup xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@[+][package:]id/resource_name"
    android:layout_height=["dimension" | "fill_parent" | "wrap_content"]
    android:layout_width=["dimension" | "fill_parent" | "wrap_content"]
    [ViewGroup-specific attributes] >
    <View
        android:id="@[+][package:]id/resource_name"
        android:layout_height=["dimension" | "fill_parent" | "wrap_content"]
        android:layout_width=["dimension" | "fill_parent" | "wrap_content"]
        [View-specific attributes] >
        <requestFocus/>
    </View>
    <ViewGroup >
        <View />
    </ViewGroup>
    <include layout="@layout/layout_resource"/>
</ViewGroup>
```

http://developer.android.com/guide/topics/resources/layout-resource.html

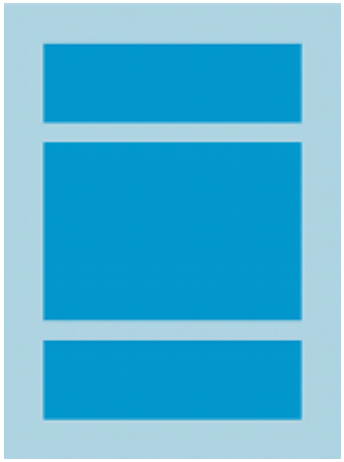
Layouts declaration

- Declare UI elements in XML.
- Instantiate layout elements at runtime.



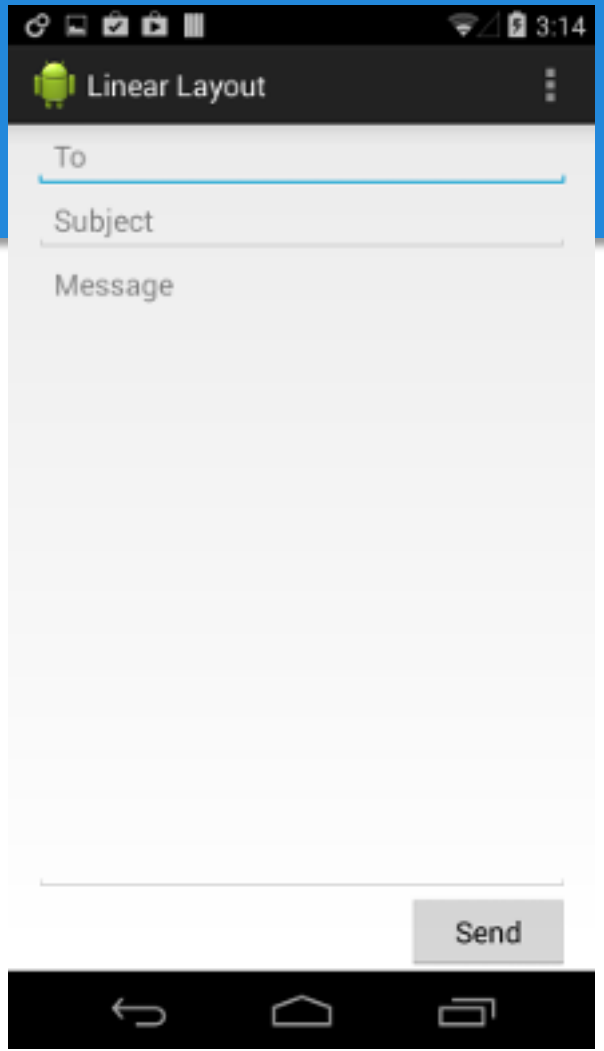
http://developer.android.com/guide/topics/ui/declaring-layout.html#layout-params

LinearLayout



```
android:orientation="vertical" >
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.a
    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>
```



Dimensions

- sp is scale-independent pixels.
- dip is Density-independent pixels.

px
Pixels - corresponds to actual pixels on the screen.

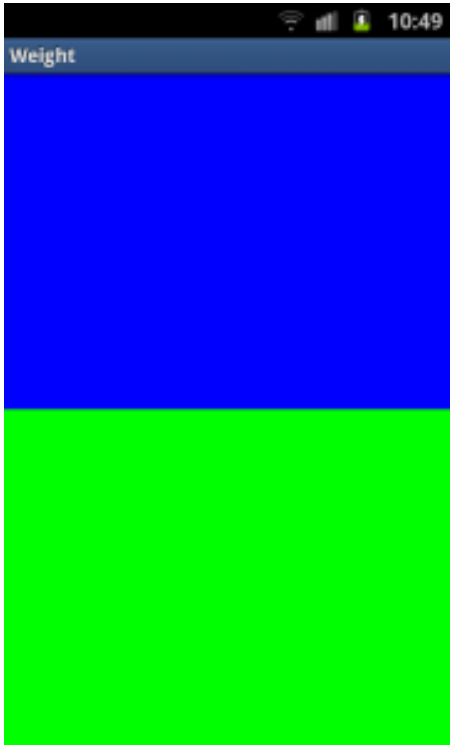
in
Inches - based on the physical size of the screen.
1 Inch = 2.54 centimeters

mm
Millimeters - based on the physical size of the screen.

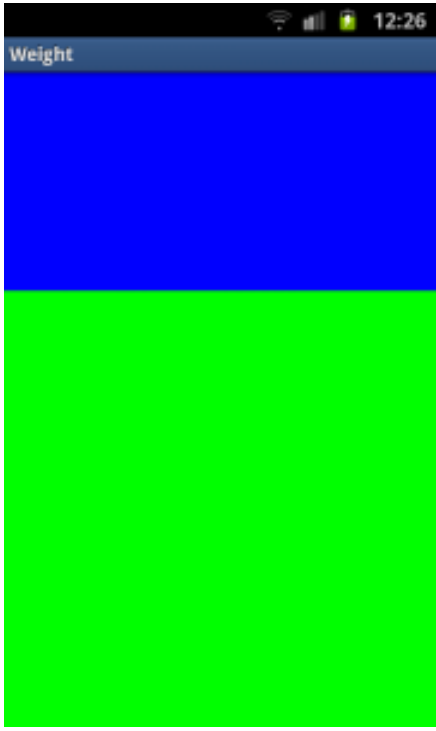
pt (it depends upon physical size of the screen) Points - 1/72 of an inch based on the physical size of the screen.

http://developer.android.com/guide/topics/resources/more-resources.html#Dimension
http://stackoverflow.com/questions/2025282/difference-between-px-dp-dip-and-sp-in-android

layout_weight




```
<LinearLayout xmlns:android="http://schemas.a
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <LinearLayout
        android:background="#0000FF"
        android:layout_height="match_parent"
        android:layout_width="match_parent"
        android:layout_weight="1" />
    <LinearLayout
        android:background="#00FF00"
        android:layout_height="match_parent"
        android:layout_width="match_parent"
        android:layout_weight="1" />
</LinearLayout>
```



http://www.chess-ix.com/blog/the-use-of-layout_weight-with-android-layouts/

Equally weighted children

To create a linear layout in which each child uses the same amount of space on the screen, set the `android:layout_height` of each view to `"0dp"` (for a vertical layout) or the `android:layout_width` of each view to `"0dp"` (for a horizontal layout). Then set the `android:layout_weight` of each view to `"1"`.

Gravity

Gravity and layout_gravity on Android

http://stackoverflow.com/questions/3482742/gravity-and-layout-gravity-on-android

445

134

I know we can set the following value to the `android:gravity` and `android:layout_gravity`.

1. center
2. center_vertical
3. center_horizontal, etc.

But I am confused regarding both of these.

What is the difference between the usage of `android:gravity` and `android:layout_gravity`?

506

Their names should help you:

- `android:gravity` sets the gravity of the content of the `View` its used on.
- `android:layout_gravity` sets the gravity of the `View` or `Layout` in its parent.

RelativeLayout

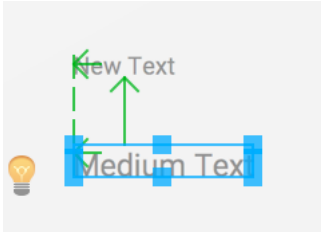
`android:layout_alignParentTop`
If `"true"`, makes the top edge of this view match the top edge of the parent.

`android:layout_centerVertical`
If `"true"`, centers this child vertically within its parent.

`android:layout_below`
Positions the top edge of this view below the view specified with a resource ID.

`android:layout_toRightOf`
Positions the left edge of this view to the right of the view specified with a resource ID.

Positioning Views



Properties	
layout:width	wrap_content
layout:height	wrap_content
layout:margin	[?, ?, 38dp, ?, ?, ?, ?]
all	
left	
top	38dp
right	
bottom	
layout:alignEnd	

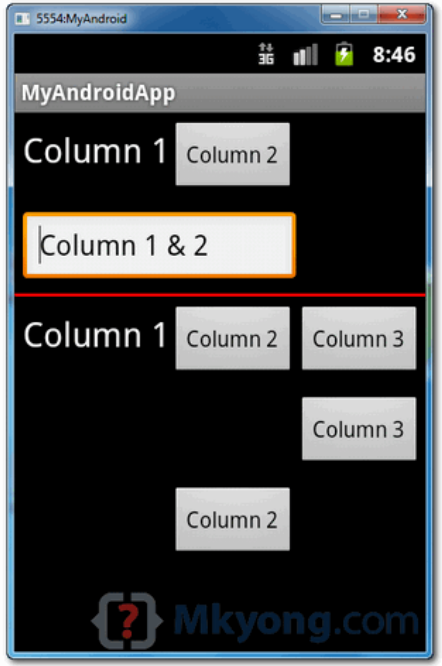
Properties	
layout:toEndOf	
layout:toStartOf	
layout:alignCompone	[top:bottom]
top:top	
top:bottom	Ab textView – "New Text"
left:left	
left:right	
bottom:bottom	
bottom:top	
right:right	
right:left	
baseline:baseline	
layout:alignParent	[]
top	☐
left	☐
bottom	☐
right	☐
missing	☐

```
<TextView
    android:layout_width="wrap_con
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="Medium Text"
    android:id="@+id/textView2"
    android:layout_below="@+id/textView"
    android:layout_alignStart="@+id/textView"
    android:layout_marginTop="38dp" />
```

TableLayout

```
<!-- 2 columns -->
<TableRow
    android:id="@+id/tableRow1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:padding="5dip" >

    <TextView
        android:id="@+id/textView1"
        android:text="Column 1"
        android:textAppearance="?android:attr/textAp

    <Button
        android:id="@+id/button1"
        android:text="Column 2" />
</TableRow>
```

```
<EditText
    android:id="@+id/editText1"
    android:layout_span="2"
    android:text="Column 1 &amp;amp; 2" />
```




```
<Button
    android:id="@+id/button4"
    android:layout_column="2"
    android:text="Column 3" />
```

```
<TextView />
```

ScrollView

```
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
    <TableLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:stretchColumns="1"
        >
        <!-- everything you already have -->
    </TableLayout>
</ScrollView>
```

http://stackoverflow.com/questions/6674341/how-to-use-scrollview-in-android

Layout pitfalls

Stop EditText get focus

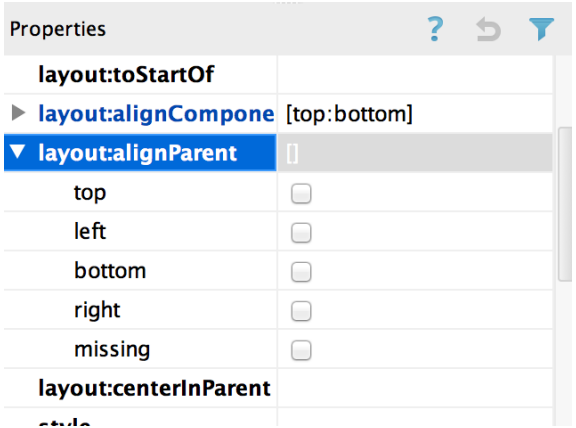
```
android:focusable="true"
android:focusableInTouchMode="true"
```

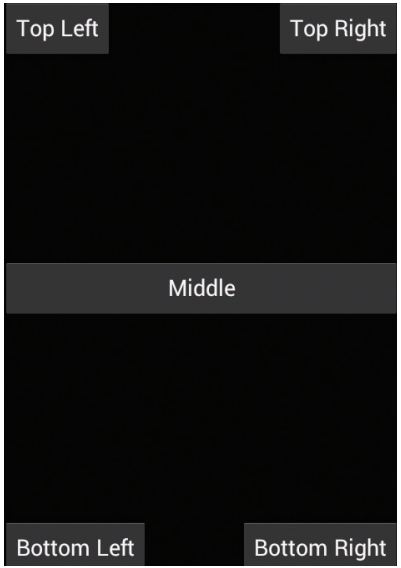
Hide soft keyboard

```
<activity
    android:label="@string/app_name"
    android:name=".LayoutsActivity"
    android:windowSoftInputMode="stateHidden" >
```

Fit screen orientation

Using anchors





Fit screen orientation

Using layouts

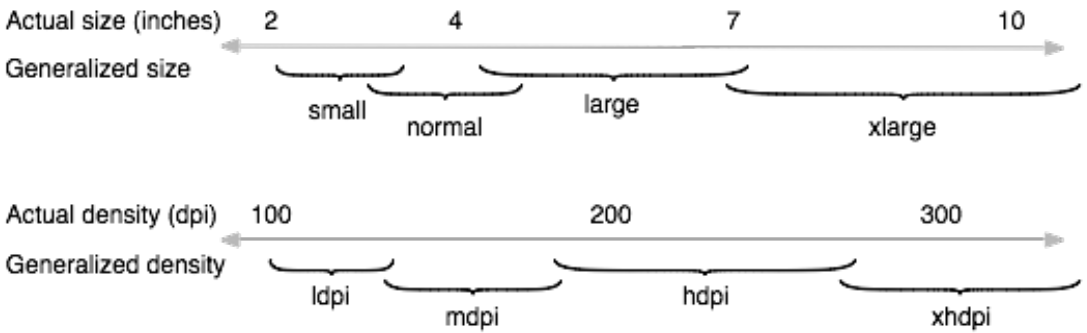
```
MyProject/
    res/
        layout/
            main.xml
        layout-land/
            main.xml
```

```
MyProject/
    res/
        layout/                 # default (portrait)
            main.xml
        layout-land/            # landscape
            main.xml
        layout-large/           # large (portrait)
            main.xml
        layout-large-land/      # large landscape
            main.xml
```

http://developer.android.com/training/basics/supporting-devices/screens.html

Support multiple screens

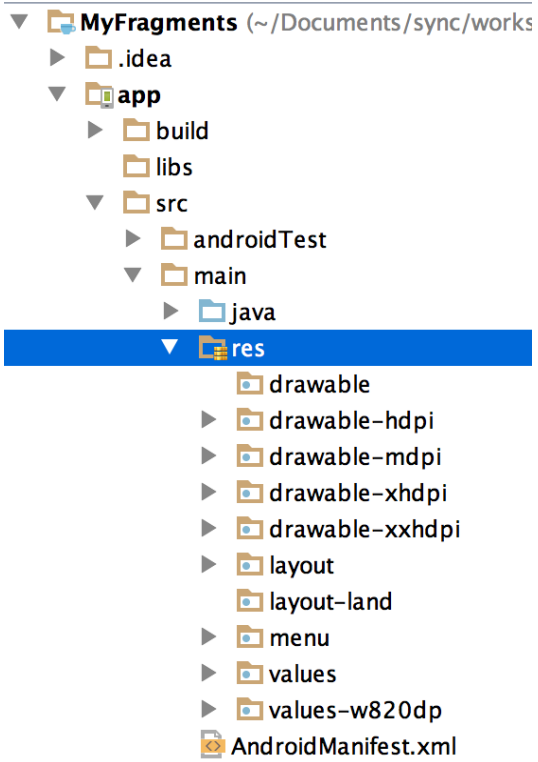
- xlarge screens are at least 960dp x 720dp
- large screens are at least 640dp x 480dp
- normal screens are at least 470dp x 320dp
- small screens are at least 426dp x 320dp



http://developer.android.com/guide/practices/screens_support.html

In Android Studio…

- Switch to 'Project' view
- To create the folder layout-land, right click res (app->src->main->res), and select new->directory
- Enter layout-land in the dialog.



http://forums.bignerdranch.com/viewtopic.php?f=396&t=9107#p25738

Detecting orientation

Deprecated

```
//---get the current display info---
WindowManager wm = getWindowManager();
Display d = wm.getDefaultDisplay();
if (d.getWidth() > d.getHeight())
{
    //---landscape mode---
    Fragment1 fragment1 = new Fragment1();
    // android.R.id.content refers to the content
    // view of the activity
    fragmentTransaction.replace(
            android.R.id.content, fragment1);
}
else
{
    //---portrait mode---
    Fragment2 fragment2 = new Fragment2();
    fragmentTransaction.replace(
            android.R.id.content, fragment2);
}
fragmentTransaction.commit();
```

Detecting orientation

```
WindowManager aWindowManager = getWindowManager();
Display aDisplay = aWindowManager.getDefaultDisplay();

int orientation = aDisplay.getRotation();

if (orientation == Surface.ROTATION_90 || orientation == Surface.ROTATION_270) {
    BlankFragment frag1 = new BlankFragment();
    aFragmentTransaction.replace(R.id.container, frag1);
} else {
```

Not (yet!) deprecated

Windows: left Ctrl + F12

Mac: Fn + Ctrl + F12

View Listeners

Code Implementation:

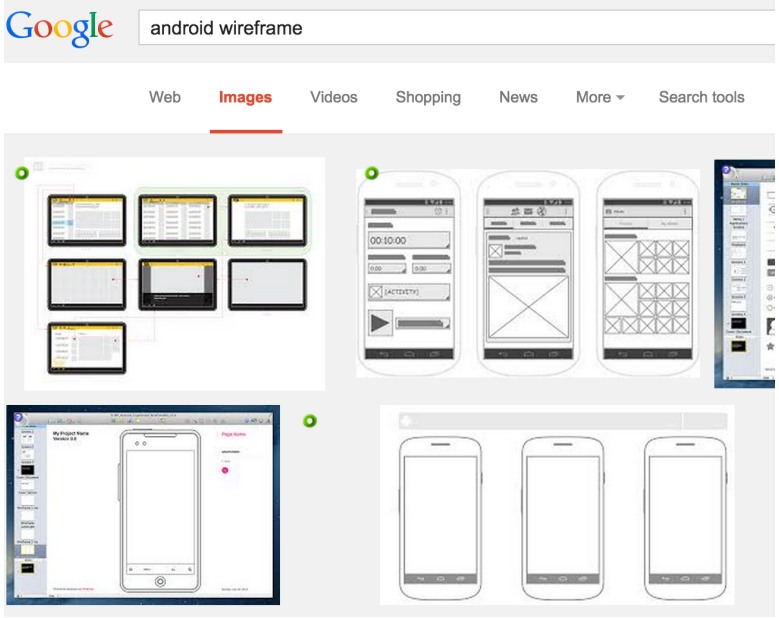
```
Button btn = (Button) findViewById(R.id.mybutton);

btn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        myFancyMethod(v);
    }
});

// some more code

public void myFancyMethod(View v) {
    // does something very interesting
}
```

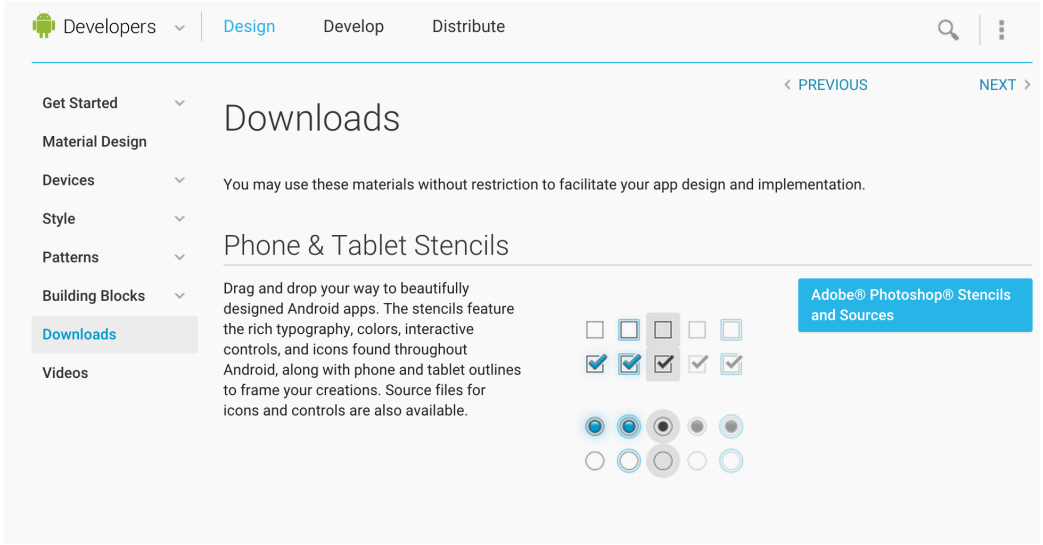
XML Implementation:

```
<?xml version="1.0" encoding="utf-8"?>
<!-- layout elements -->
<Button android:id="@+id/mybutton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Click me!"
    android:onClick="myFancyMethod" />
<!-- even more layout elements -->
```

http://stackoverflow.com/questions/4153517/how-exactly-does-the-androidonclick-xml-attribute-differ-from-setonclicklistene

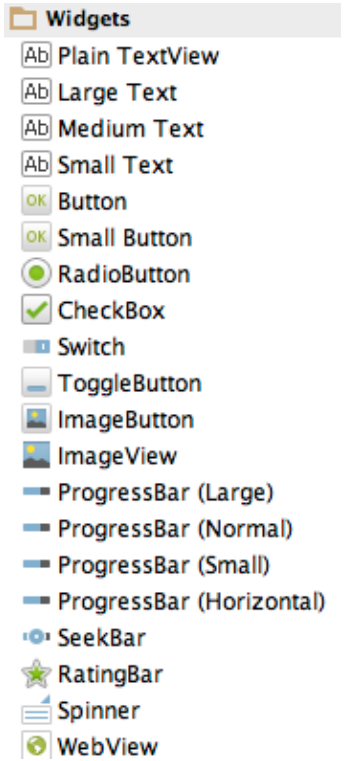
Android stencils

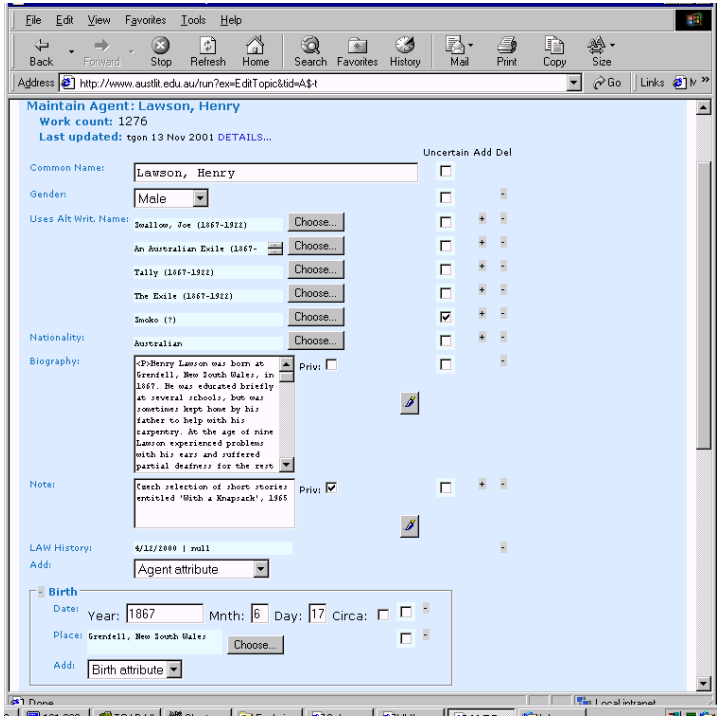




INPUT CONTROLS

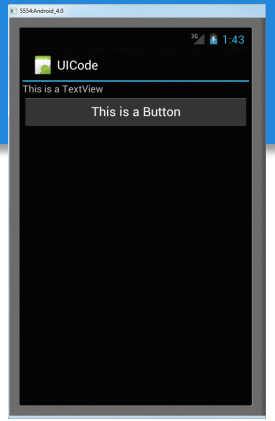
Input Controls





http://ausweb.scu.edu.au/aw02/papers/refereed/fitch/paper.html

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    //setContentView(R.layout.main);
    //---param for views---
    LayoutParams params =
        new LinearLayout.LayoutParams(
            LayoutParams.FILL_PARENT,
            LayoutParams.WRAP_CONTENT);

    //---create a layout---
    LinearLayout layout = new LinearLayout(this);
    layout.setOrientation(LinearLayout.VERTICAL);

    //---create a textview---
    TextView tv = new TextView(this);
    tv.setText("This is a TextView");
    tv.setLayoutParams(params);

    //---create a button---
    Button btn = new Button(this);
    btn.setText("This is a Button");
    btn.setLayoutParams(params);

    //---adds the textview---
    layout.addView(tv);

    //---adds the button---
    layout.addView(btn);

    //---create a layout param for the layout---
    LinearLayout.LayoutParams layoutParam =
        new LinearLayout.LayoutParams(
                LayoutParams.FILL_PARENT,
                LayoutParams.WRAP_CONTENT );

    this.addContentView(layout, layoutParam);
}
```



Buttons


```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/button_text"
    ... />
```

```
<ImageButton
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/button_icon"
    ... />
```

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/button_text"
    android:drawableLeft="@drawable/button_icon"
    ... />
```

Buttons

android:onClick➔

- Be public
- Return void
- Define a View as its only parameter

```
/** Called when the user touches the button */
public void sendMessage(View view) {
    // Do something in response to button click
}
```

OnClickListener➔

```
Button button = (Button) findViewById(R.id.button_send);
button.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        // Do something in response to button click
    }
});
```

Text fields

```
<EditText
    android:id="@+id/email_address"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="@string/email_hint"
    android:inputType="textEmailAddress" />
```

Figure 3. The `phone` input type.

`"text"`
Normal text keyboard.

`"textEmailAddress"`
Normal text keyboard with the @ character.

`"textUri"`
Normal text keyboard with the / character.

`"number"`
Basic number keypad.

`"phone"`
Phone-style keypad.

Text fields

Editable	getText() Return the text the TextView is displaying.

```
public void onClick(View v) {
  String str = eText.getText().toString();
```

void	setText(CharSequence text, TextView.BufferType type) Sets the text that this TextView is to display (see `setText(CharSequence)`) and also sets whether it is stored in a styleable/spannable buffer and whether it is editable.

Text fields

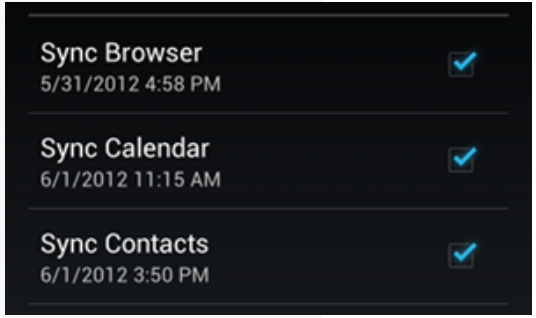
```
EditText editText = (EditText) findViewById(R.id.search);
editText.setOnEditorActionListener(new OnEditorActionListener() {
    @Override
    public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
        boolean handled = false;
        if (actionId == EditorInfo.IME_ACTION_SEND) {
            sendMessage();
            handled = true;
        }
        return handled;
    }
});
```

Specifying Keyboard Actions

```
<EditText
    android:id="@+id/search"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="@string/search_hint"
    android:inputType="text"
    android:imeOptions="actionSend" />
```

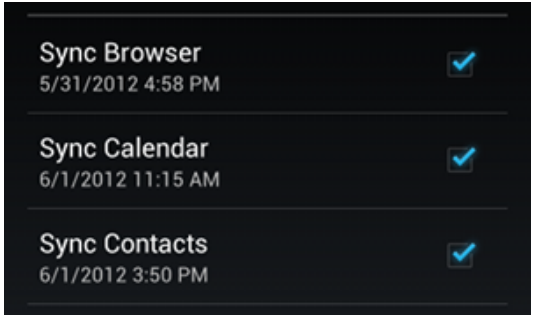
Checkboxes

```
<?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">
    <CheckBox android:id="@+id/checkbox_meat"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/meat"
        android:onClick="onCheckboxClicked"/>
    <CheckBox android:id="@+id/checkbox_cheese"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/cheese"
        android:onClick="onCheckboxClicked"/>
</LinearLayout>
```



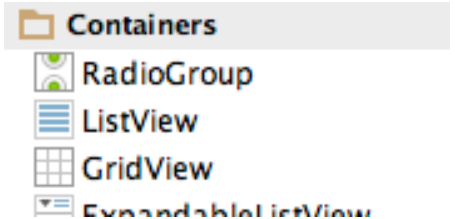
Checkboxes

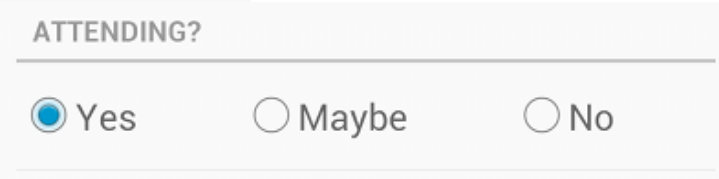
```
public void onCheckboxClicked(View view) {
    // Is the view now checked?
    boolean checked = ((CheckBox) view).isChecked();

    // Check which checkbox was clicked
    switch(view.getId()) {
        case R.id.checkbox_meat:
            if (checked)
                // Put some meat on the sandwich
            else
                // Remove the meat
            break;
        case R.id.checkbox_cheese:
            if (checked)
                // Cheese me
            else
                // I'm lactose intolerant
            break;
        // TODO: Veggie sandwich
    }
}
```



Radio Buttons

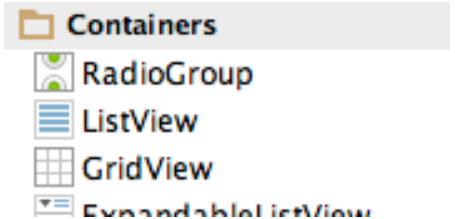
```
<?xml version="1.0" encoding="utf-8"?>
<RadioGroup xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <RadioButton android:id="@+id/radio_pirates"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/pirates"
        android:onClick="onRadioButtonClicked"/>
    <RadioButton android:id="@+id/radio_ninjas"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/ninjas"
        android:onClick="onRadioButtonClicked"/>
</RadioGroup>
```

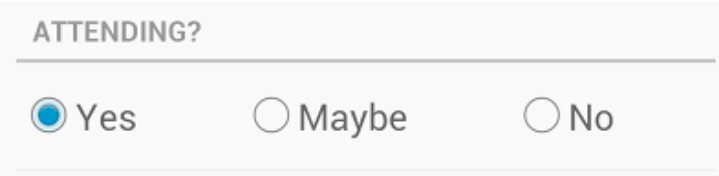




Radio Buttons

```
public void onRadioButtonClicked(View view) {
    // Is the button now checked?
    boolean checked = ((RadioButton) view).isChecked();

    // Check which radio button was clicked
    switch(view.getId()) {
        case R.id.radio_pirates:
            if (checked)
                // Pirates are the best
            break;
        case R.id.radio_ninjas:
            if (checked)
                // Ninjas rule
            break;
    }
}
```





Toggle Buttons

```
public void onToggleClicked(View view) {
    // Is the toggle on?
    boolean on = ((ToggleButton) view).isChecked();

    if (on) {
        // Enable vibrate
    } else {
        // Disable vibrate
    }
}
```


```
<ToggleButton
    android:id="@+id/togglebutton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textOn="Vibrate on"
    android:textOff="Vibrate off"
    android:onClick="onToggleClicked"/>
```

CompoundButton

OnCheckedChangeListener

```
ToggleButton toggle = (ToggleButton) findViewById(R.id.togglebutton);
toggle.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener() {
    public void onCheckedChanged(CompoundButton buttonView, boolean isChecked) {
        if (isChecked) {
            // The toggle is enabled
        } else {
            // The toggle is disabled
        }
    }
});
```

CompoundButton

extends Button

implements Checkable

java.lang.Object
↳android.view.View
↳android.widget.TextView
↳android.widget.Button
↳android.widget.CompoundButton

▶ Known Direct Subclasses

CheckBox, RadioButton, Switch, SwitchCompat, ToggleButton

`onClick()`
: From `View.OnClickListener`. This is called when the user either touches the item (when in touch mode), or focuses upon the item with the navigation-keys or trackball and presses the suitable "enter" key or presses down on the trackball.

`onLongClick()`
: From `View.OnLongClickListener`. This is called when the user either touches and holds the item (when in touch mode), or focuses upon the item with the navigation-keys or trackball and presses and holds the suitable "enter" key or presses and holds down on the trackball (for one second).

`onFocusChange()`
: From `View.OnFocusChangeListener`. This is called when the user navigates onto or away from the item, using the navigation-keys or trackball.

`onKey()`
: From `View.OnKeyListener`. This is called when the user is focused on the item and presses or releases a hardware key on the device.

`onTouch()`
: From `View.OnTouchListener`. This is called when the user performs an action qualified as a touch event, including a press, a release, or any movement gesture on the screen (within the bounds of the item).

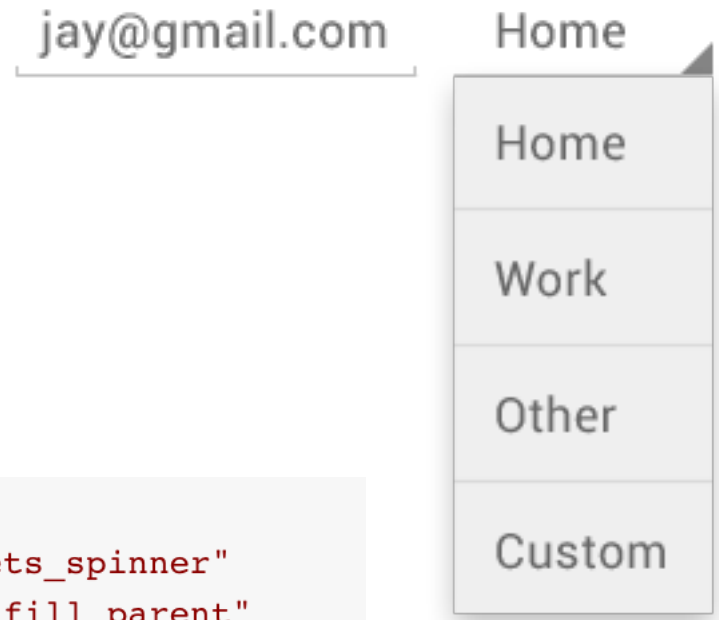
`onCreateContextMenu()`
: From `View.OnCreateContextMenuListener`. This is called when a Context Menu is being built (as the result of a sustained "long click"). See the discussion on context menus in the Menus developer guide.

SPINNERS

Spinners

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="planets_array">
        <item>Mercury</item>
        <item>Venus</item>
        <item>Earth</item>
        <item>Mars</item>
        <item>Jupiter</item>
        <item>Saturn</item>
        <item>Uranus</item>
        <item>Neptune</item>
    </string-array>
</resources>
```

```xml
<Spinner
    android:id="@+id/planets_spinner"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content" />
```



Spinners

```
Spinner spinner = (Spinner) findViewById(R.id.spinner);
// Create an ArrayAdapter using the string array and a default spinner layout
ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(this,
        R.array.planets_array, android.R.layout.simple_spinner_item);
// Specify the layout to use when the list of choices appears
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
// Apply the adapter to the spinner
spinner.setAdapter(adapter);
```

```
public class SpinnerActivity extends Activity implements OnItemSelectedListener {
    ...

    public void onItemSelected(AdapterView<?> parent, View view,
            int pos, long id) {
        // An item was selected. You can retrieve the selected item using
        // parent.getItemAtPosition(pos)
    }

    public void onNothingSelected(AdapterView<?> parent) {
        // Another interface callback
    }
}
```

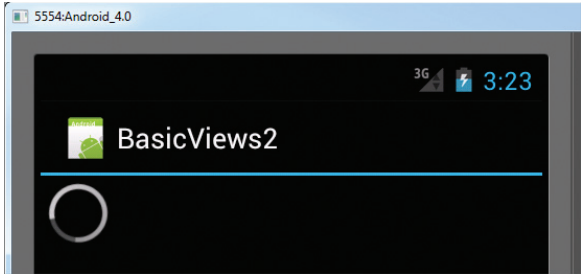
```
public void onItemSelected(AdapterView<?> parent, View view, int pos,long id) {
    Toast.makeText(parent.getContext(),
        "OnItemSelectedListener : " + parent.getItemAtPosition(pos).toString(),
        Toast.LENGTH_SHORT).show();
}
```

http://stackoverflow.com/questions/20151414/how-can-i-use-onitemselected-in-android

PROGRESS BAR AND PICKERS

ProgressBar

```
<LinearLayout
    android:orientation="horizontal"
    ... >
    <ProgressBar
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        style="@android:style/Widget.ProgressBar.Small"
        android:layout_marginRight="5dp" />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/loading" />
</LinearLayout>
```



ProgressBar

```
public class MyActivity extends Activity {
    private static final int PROGRESS = 0x1;

    private ProgressBar mProgress;
    private int mProgressStatus = 0;

    private Handler mHandler = new Handler();

    protected void onCreate(Bundle icicle) {
        super.onCreate(icicle);

        setContentView(R.layout.progressbar_activity);
        mProgress = (ProgressBar) findViewById(R.id.progress_bar);

        // Start lengthy operation in a background thread
        new Thread(new Runnable() {
            public void run() {
                while (mProgressStatus < 100) {
                    mProgressStatus = doWork();

                    // Update the progress bar
                    mHandler.post(new Runnable() {
                        public void run() {
                            mProgress.setProgress(mProgressStatus);
                        }
                    });
                }
            }
        }).start();
    }
}
```

Java threads

To have the run() method executed by a thread, pass an instance of MyRunnable to a Thread in its constructor. Here is how that is done:

```
Thread thread = new Thread(new MyRunnable());
thread.start();
```

http://tutorials.jenkov.com/java-concurrency/creating-and-starting-threads.html

Handler

A Handler allows you to send and process Message and Runnable objects associated with a thread's MessageQueue. Each Handler instance is associated with a single thread and that thread's message queue. When you create a new Handler, it is bound to the thread / message queue of the thread that is creating it -- from that point on, it will deliver messages and runnables to that message queue and execute them as they come out of the message queue.

http://developer.android.com/reference/android/os/Handler.html

Does handler.post(runnable) start a new thread?

5

2

If the handler was instantiated in the main UI thread, does a post with a Runnable create a child thread that gets added to the message queue, or does it just get run in the UI thread?

```
handler.post(new Runnable(){
    public void run() {
        // do stuff
    }
});
```