Android Lecture #3

View

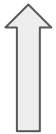
Basic UI building block

- Knows to draw itself
- Used for user interaction

https://developer.android.com/reference/android/view/View.html

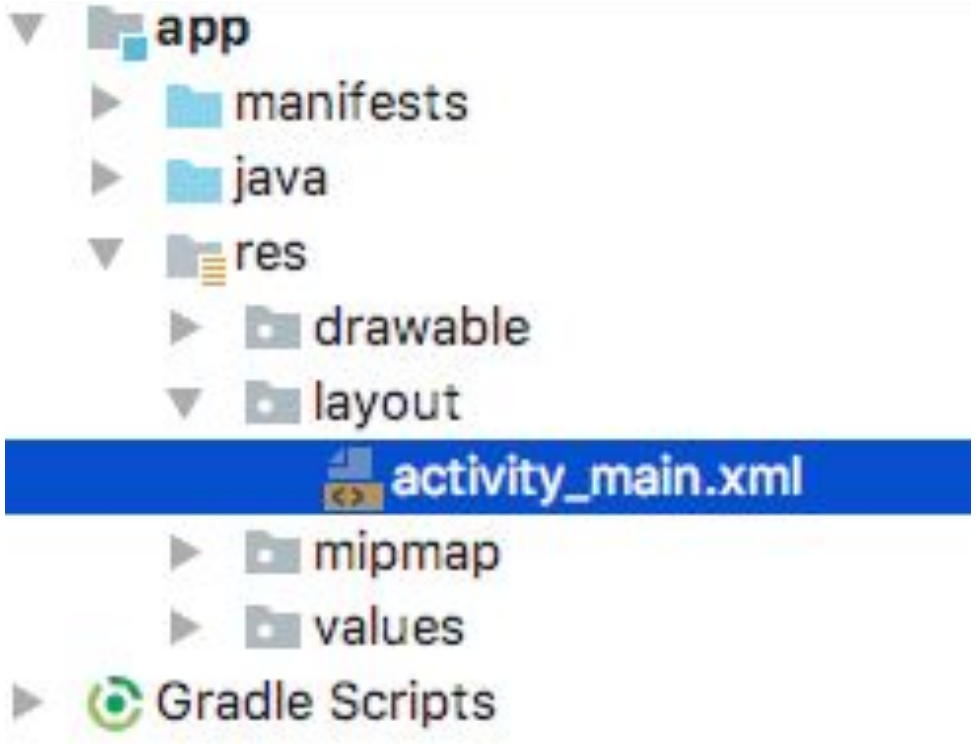
Activity - recap

```
public class MainActivity extends Activity {

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



Setting the Activity content View
using layout resource

Resources - recap



UI building using XML

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<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">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!">
    </TextView>

</RelativeLayout>
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<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">


    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!">
    </TextView>

</RelativeLayout>
```

} View declaration

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<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">


    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!">
    </TextView>

</RelativeLayout>
```

View declaration

- XML tag

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<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">


    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!" />


</RelativeLayout>
```

View declaration

- XML tag

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<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">


    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!">
    </TextView>

</RelativeLayout>
```

View declaration

- XML tag
- View class name

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<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">


    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!">
    </TextView>

</RelativeLayout>
```

View declaration

- XML tag
- View class name
- View properties

Remember!

View properties must contain the view width & height

```
android:layout_width="300dp"
android:layout_height="100dp"
```

Remember!

View properties must contain the view width & height **(in DP)**

```
android:layout_width="300dp"
android:layout_height="100dp"
```

Understanding density-independent pixels

Understanding DP

- **Screen density** = The ratio of resolution and display size
- Measured as Dots Per Inch - **dpi**
- Higher dpi = More details per inch of screen

Understanding DP

Same resolution **does not** mean same dpi

1080 x 1920
5 inch

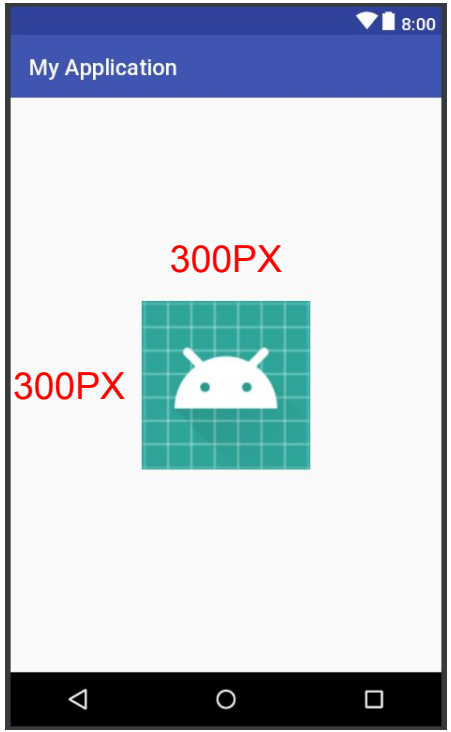
1080 x 1920
7 inch

Understanding DP

Pixel is a density **dependent** unit =

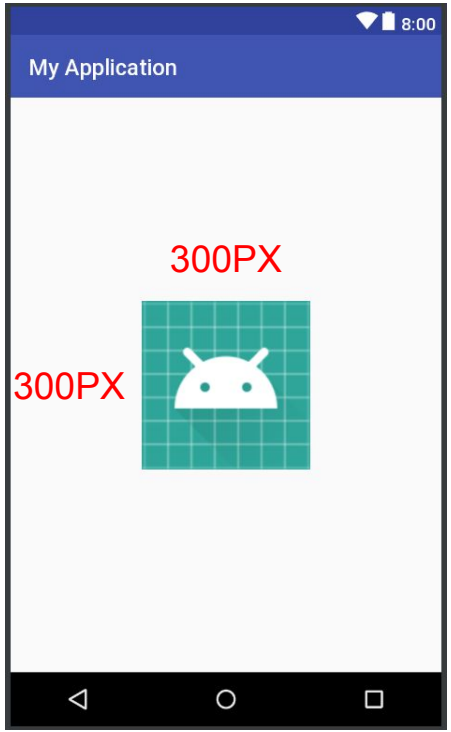
Pixel size changes based on dpi

Layout defined using pixels



Nexus 4
4.7 inch
758 x 1280
318 dpi

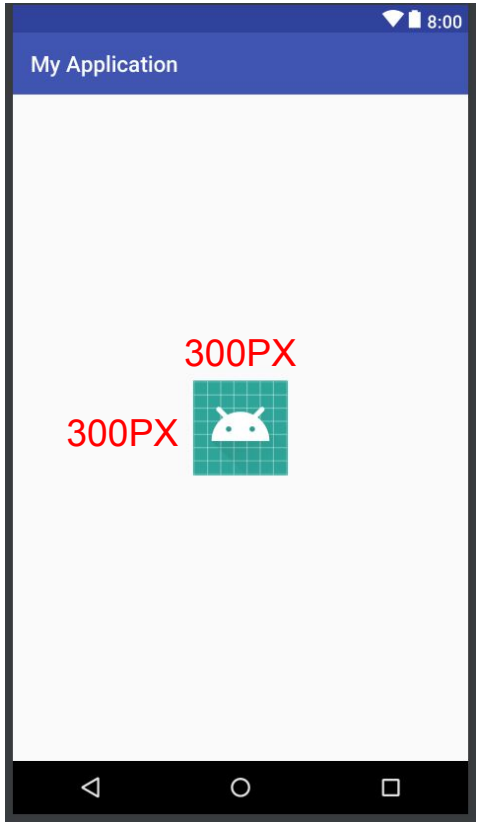
Layout defined using pixels



Nexus 4
4.7 inch
758 x 1280
318 dpi

VS

Nexus 6P
5.7 inch
1440 x 2560
560 dpi



Finding dp value based on pixels

Calculating DP

$$dp =$$

Calculating DP

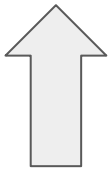
dp = (pixels * 160)

Calculating DP

$$\mathbf{dp} = (\text{pixels} * 160) / \text{dpi}$$

Calculating DP

dp = (pixels * 160) / dpi



But which one???

Android generalized densities

ldpi (low) ~120dpi

mdpi (medium) ~160dpi

hdpi (high) ~240dpi

xhdpi (extra-high) ~320dpi

xxhdpi (extra-extra-high) ~480dpi

xxxhdpi (extra-extra-extra-high) ~640dpi

Android generalized densities

ldpi (low) ~120dpi

mdpi (medium) ~160dpi

hdpi (high) ~240dpi

xhdpi (extra-high) ~320dpi

xxhdpi (extra-extra-high) ~480dpi

xxxhdpi (extra-extra-extra-high) ~640dpi

Calculating DP

$$\mathbf{dp} = (\text{pixels} * 160) / \text{dpi}$$

Calculating DP

? dp = (300px * 160) / dpi

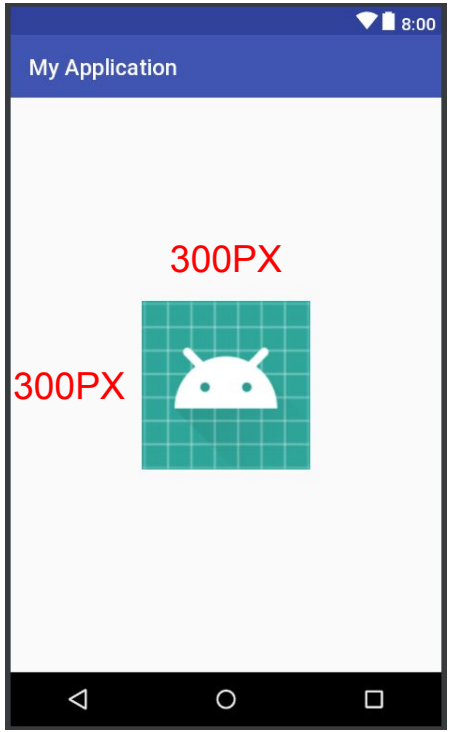
Calculating DP

? dp = (300px * 160) / 240 dpi

Calculating DP

200dp = (300px * 160) / 240 dpi

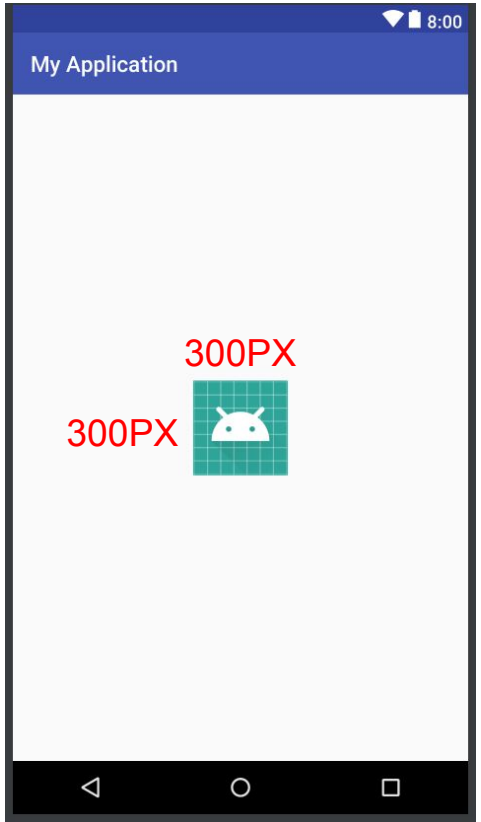
Layout defined using pixels



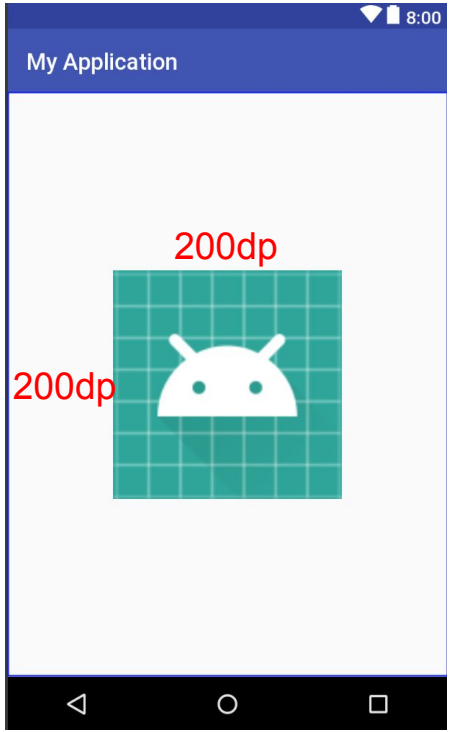
Nexus 4
4.7 inch
758 x 1280
318 dpi

VS

Nexus 6P
5.7 inch
1440 x 2560
560 dpi



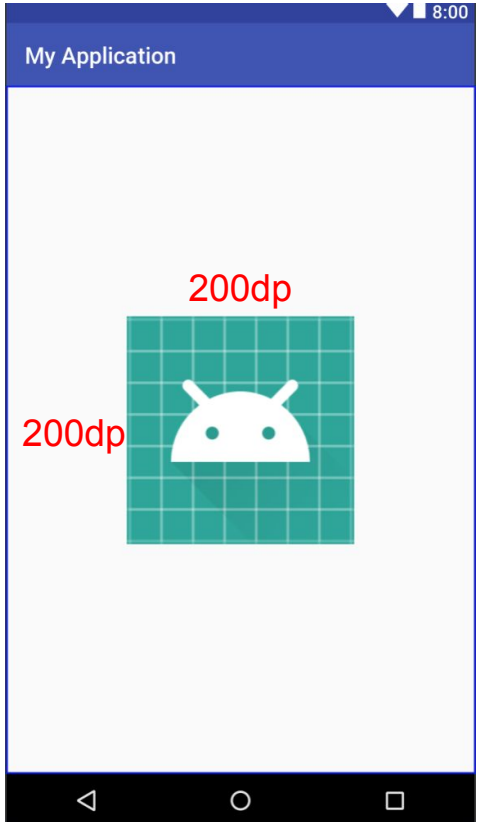
Layout defined using dp



Nexus 4
4.7 inch
758 x 1280
318 dpi

VS

Nexus 6P
5.7 inch
1440 x 2560
560 dpi



Android generalized densities

ldpi (0.75x)	@ 100.00dp = 75.00px
mdpi (1x)	@ 100.00dp = 100.00px
hdpi (1.5x)	@ 100.00dp = 150.00px
xhdpi (2x)	@ 100.00dp = 200.00px
xxhdpi (3x)	@ 100.00dp = 300.00px
xxxhdpi (4x)	@ 100.00dp = 400.00px

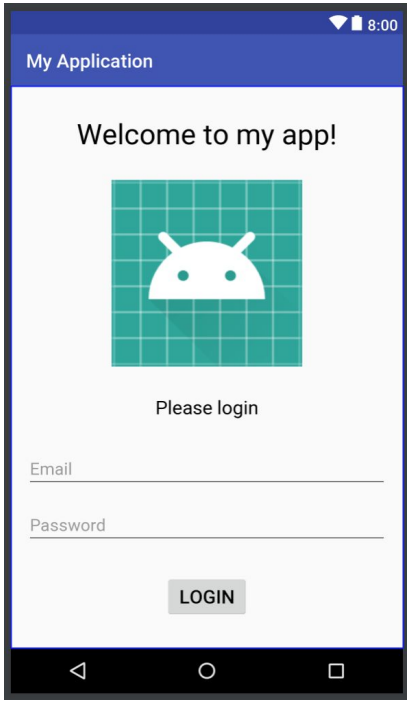
Sometimes,

defining layout in dp is not enough

:(

Understanding dp

This layout requires minimum amount of vertical space



Calculating DP

dp = (pixels * 160) / screen density

Calculating DP

dp = (pixels * 160) / screen density

Nexus 6P (1440 x 2560, 560 dpi)

(2560px * 160) / 560 dpi = 731 dp

Calculating DP

dp = (pixels * 160) / screen density

Nexus 6P (1440 x 2560, 560 dpi)

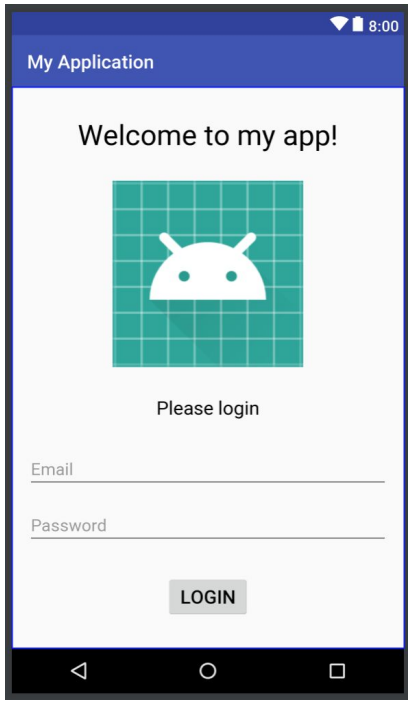
(2560px * 160) / 560 dpi = 731 dp

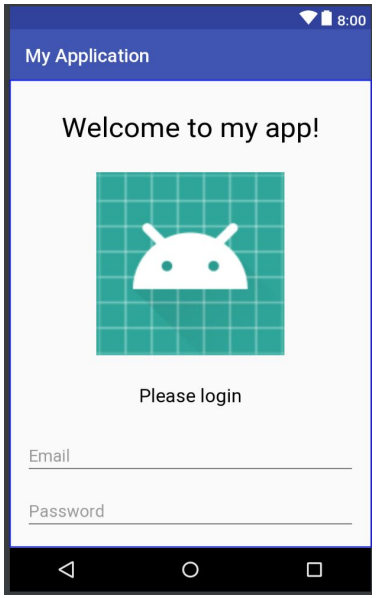
Nexus 4 (768 x 1280, 318 dpi)

(1280px * 160) / 318 dpi = 644 dp

Not enough DPs

731 dp





644 dp

Creating multiple layout files

General screen sizes

The “old way” (pre Android 3.2):

xlarge >= 960dp x 720dp

large >= 640dp x 480dp

normal >= 470dp x 320dp

small >= 426dp x 320dp

General screen sizes

The “old way” (pre Android 3.2):

res/layout/my_layout.xml

res/layout-large/my_layout.xml

res/layout-xlarge/my_layout.xml

General screen sizes

The “old way” (pre Android 3.2):

res/layout/my_layout.xml

res/layout-large/my_layout.xml

res/layout-xlarge/my_layout.xml

Android system scans for match backwards

General screen sizes

The “new way” (since Android 3.2):

sw<N>dp ex: layout-sw600dp	• Smallest possible width • Not changed by orientation
w<N>dp ex: layout-w320dp	• Minimum available required width • Changed by orientation
h<N>dp ex: layout-h700dp	• Minimum available required height • Changed by orientation

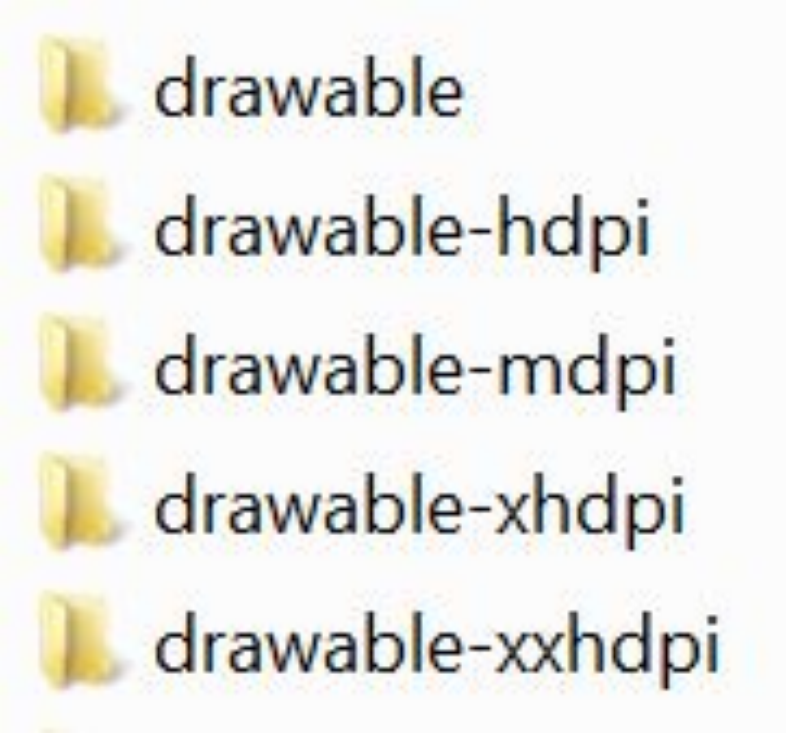
https://developer.android.com/guide/practices/screens_support.html#DeclaringTabletLayouts

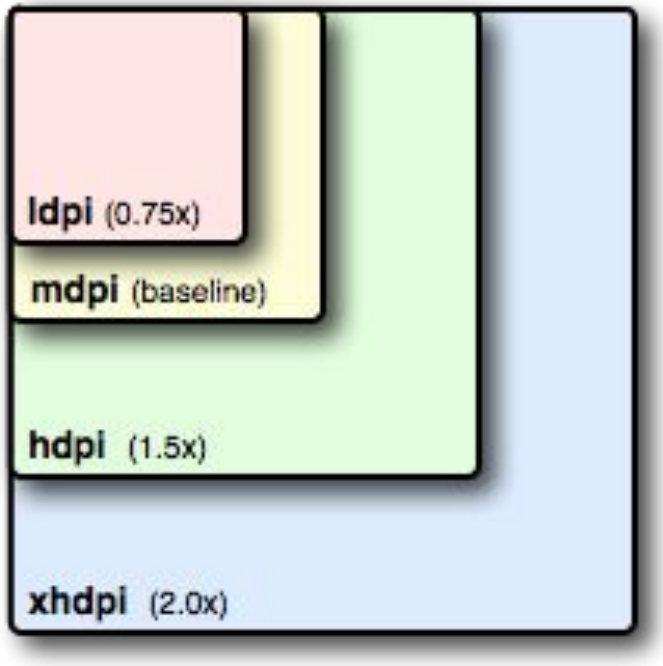
Alternative drawables for different screen sizes

Alternative drawables

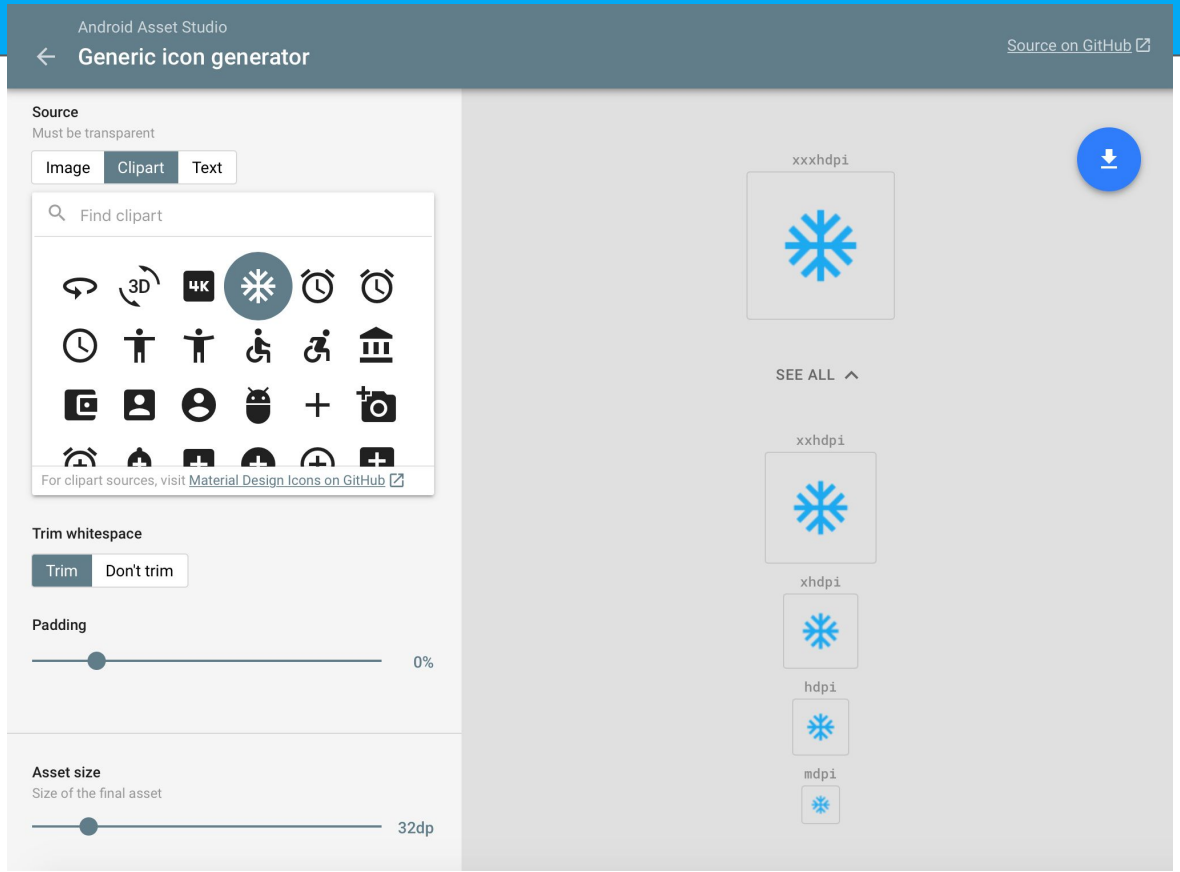
ldpi (0.75x)	@ 100.00dp = 75.00px
mdpi (1x)	@ 100.00dp = 100.00px
hdpi (1.5x)	@ 100.00dp = 150.00px
xhdpi (2x)	@ 100.00dp = 200.00px
xxhdpi (3x)	@ 100.00dp = 300.00px
xxxhdpi (4x)	@ 100.00dp = 400.00px

Alternative drawables





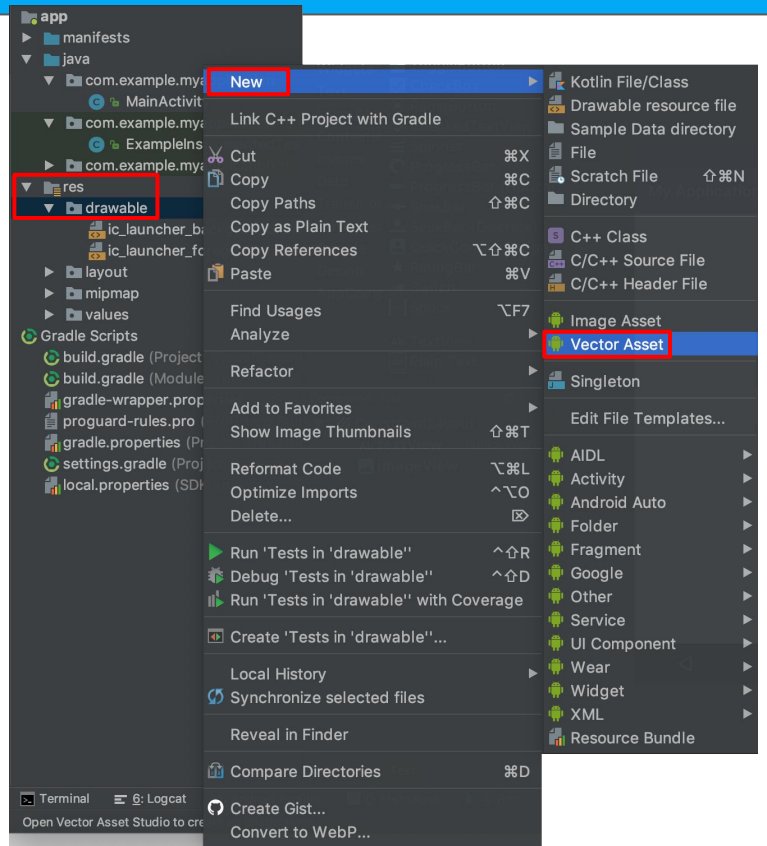
Quick way to make multiple size drawables



[Android Asset Studio]

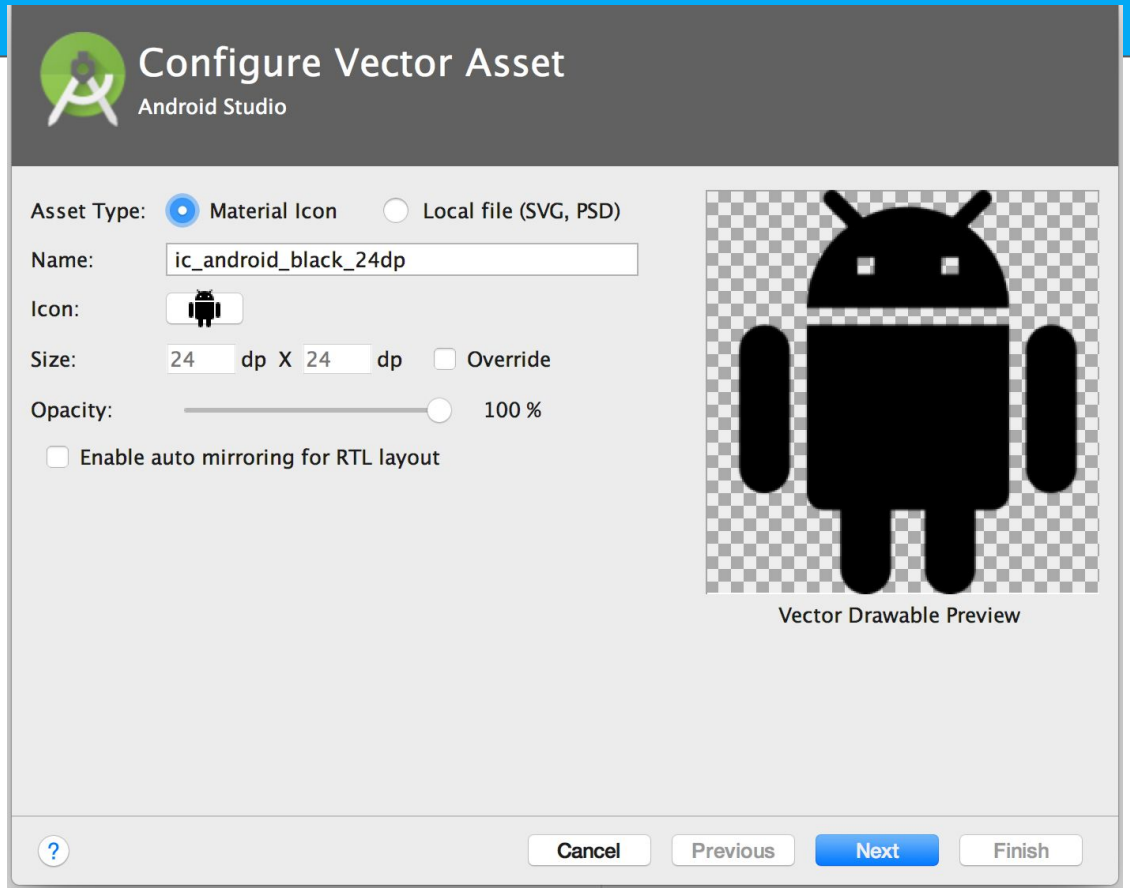
Scalable Vector Graphics
SVG

How to add SVG to our project?



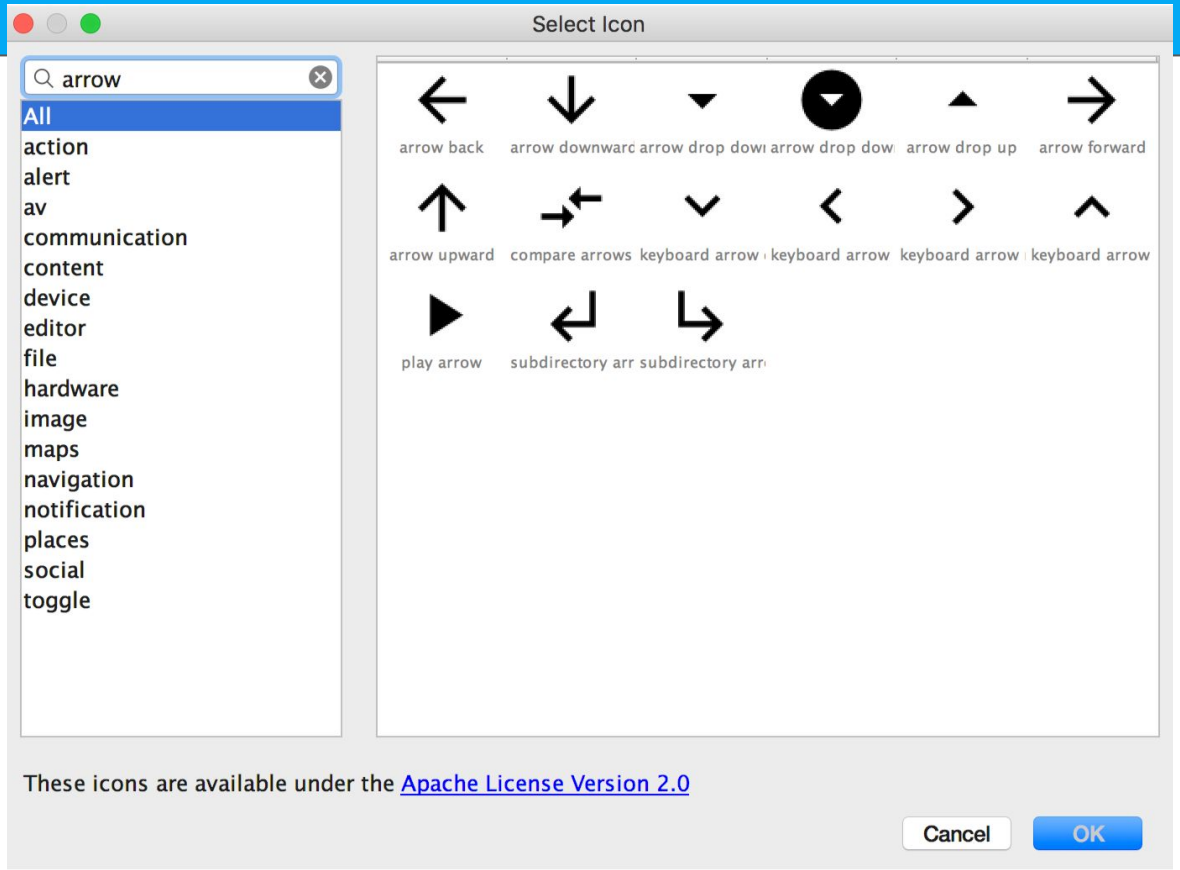
[Vector Asset Studio]

How to add SVG to our project?



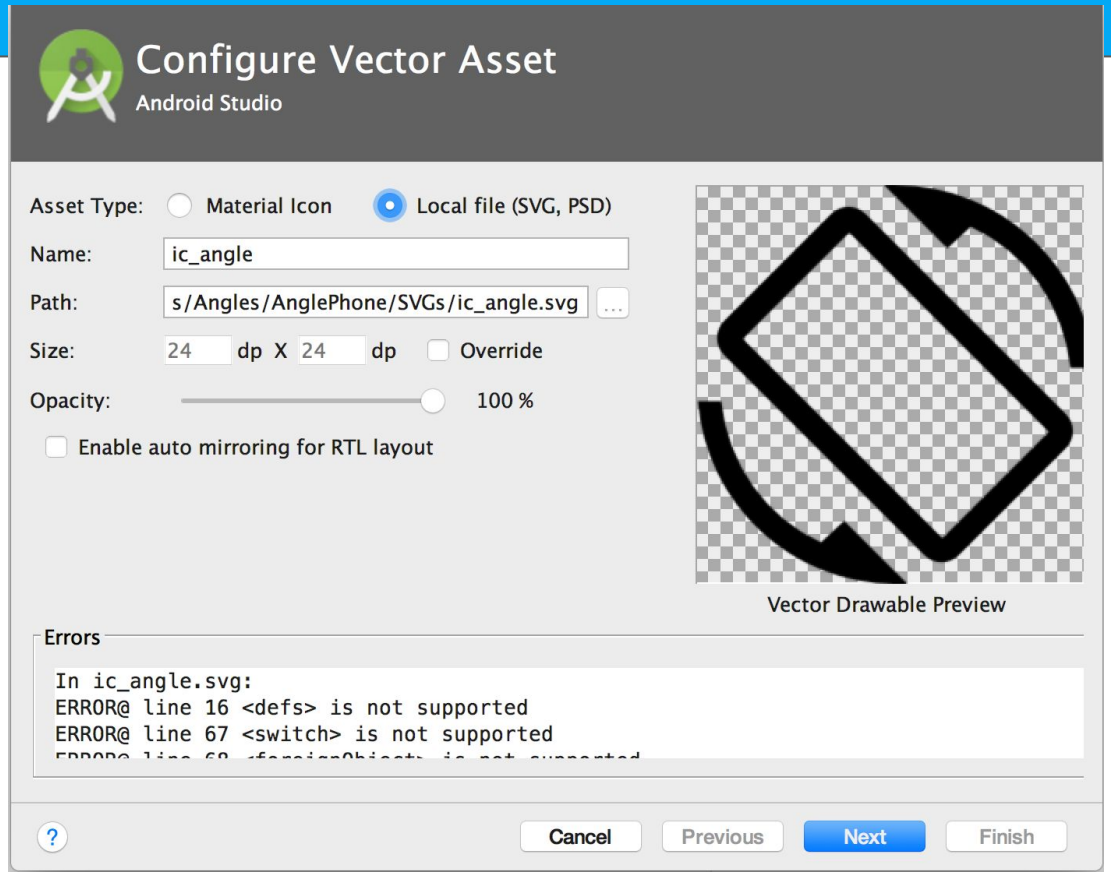
[Vector Asset Studio]

How to add SVG to our project?



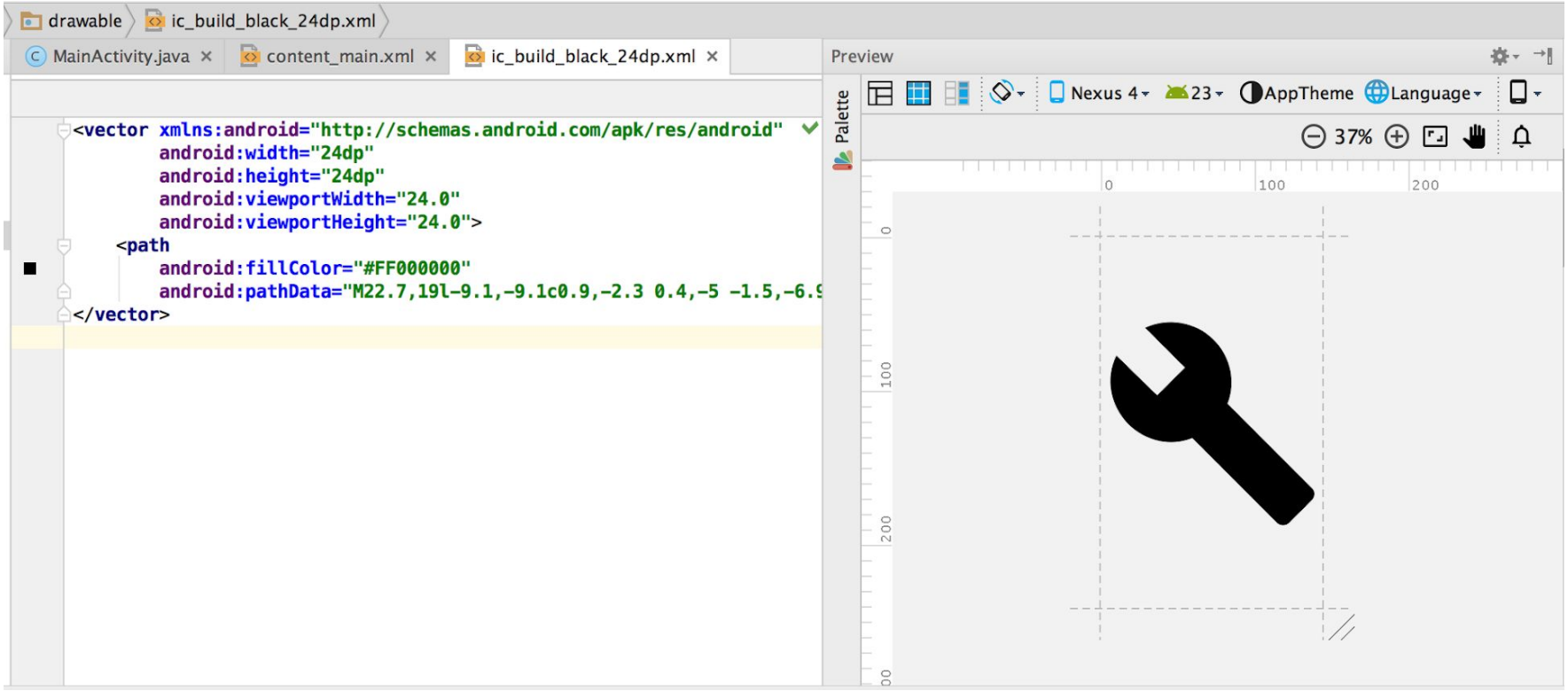
Vector Asset Studio

How to add SVG to our project?



[Vector Asset Studio]

SVG as XML


drawable
ic_build_black_24dp.xml
MainActivity.java
content_main.xml
ic_build_black_24dp.xml
<vector xmlns:android="http://schemas.android.com/apk/res/android"
android:width="24dp"
android:height="24dp"
android:viewportWidth="24.0"
android:viewportHeight="24.0">
<path
android:fillColor="#FF000000"
android:pathData="M22.7,19l-9.1,-9.1c0.9,-2.3 0.4,-5 -1.5,-6.9
</vector>
Preview
Palette
Nexus 4
23
AppTheme
Language
37%
0
100
200

Questions ?

Back to defining View in XML

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<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">


    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!">
    </TextView>

</RelativeLayout>
```

} View declaration

Remember!

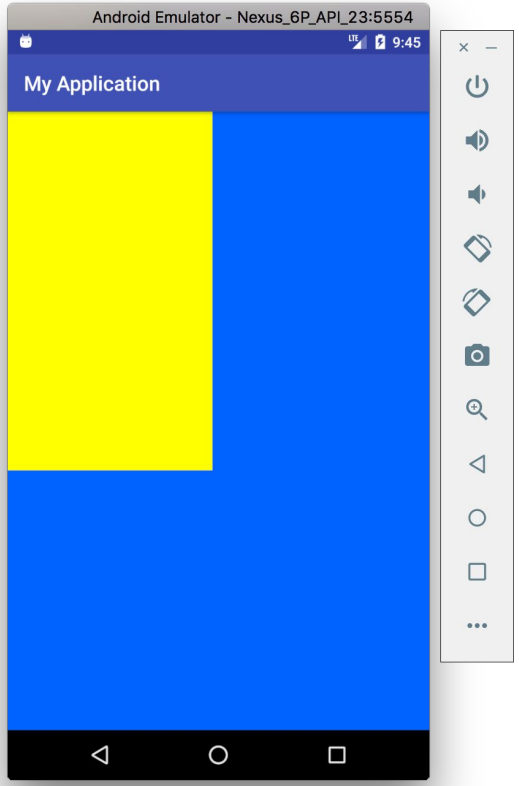
View properties must contain the view width & height

```
android:layout_width="300dp"
android:layout_height="100dp"
```

Width & Height

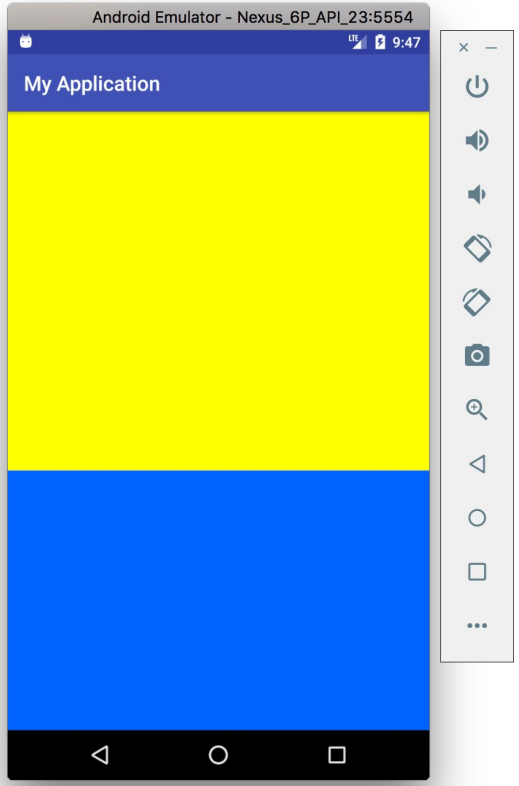
Width & Height - fixed

```
<View
    android:layout_width="200dp"
    android:layout_height="350dp"
/>
```



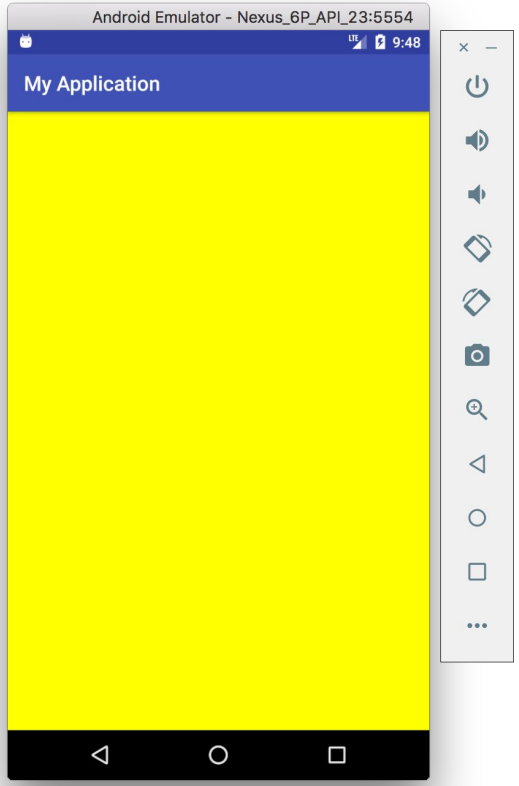
Width & Height - match parent

```
<View
    android:layout_width="match_parent"
    android:layout_height="350dp" />
```



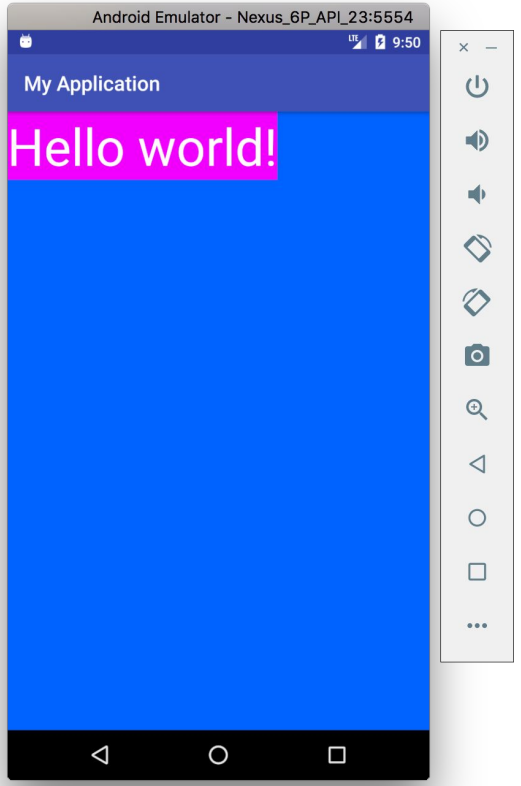
Width & Height - match parent

```
<View
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```



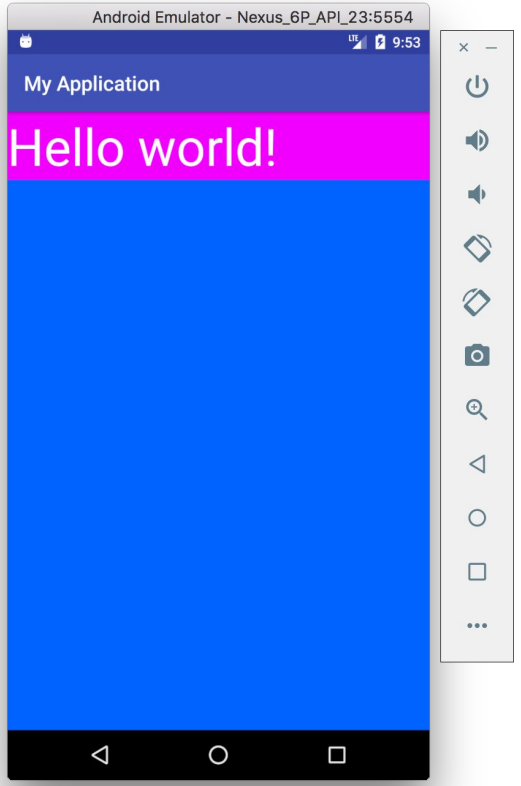
Width & Height - wrap content

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Hello world!"
    android:textSize="50sp"
    android:textColor="#FFFFFF"
    android:background="#f200ff"
    />
```



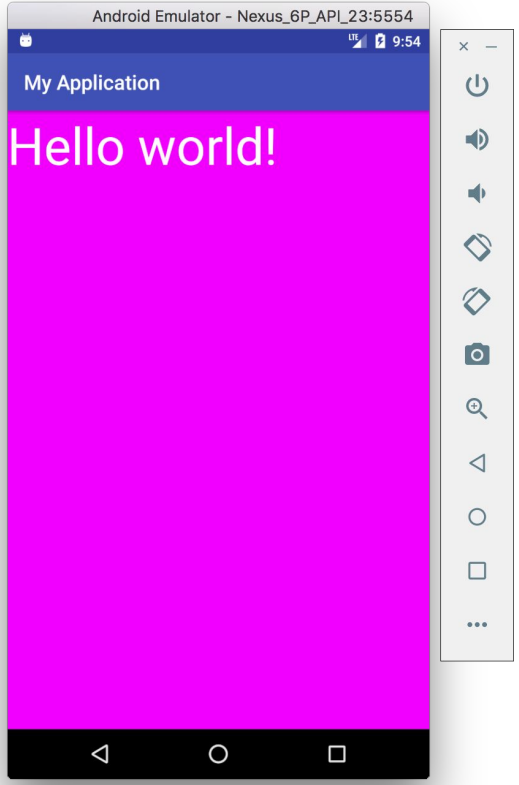
Width & Height

```
<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Hello world!"
    android:textSize="50sp"
    android:textColor="#FFFFFF"
    android:background="#f200ff"
    />
```



Width & Height

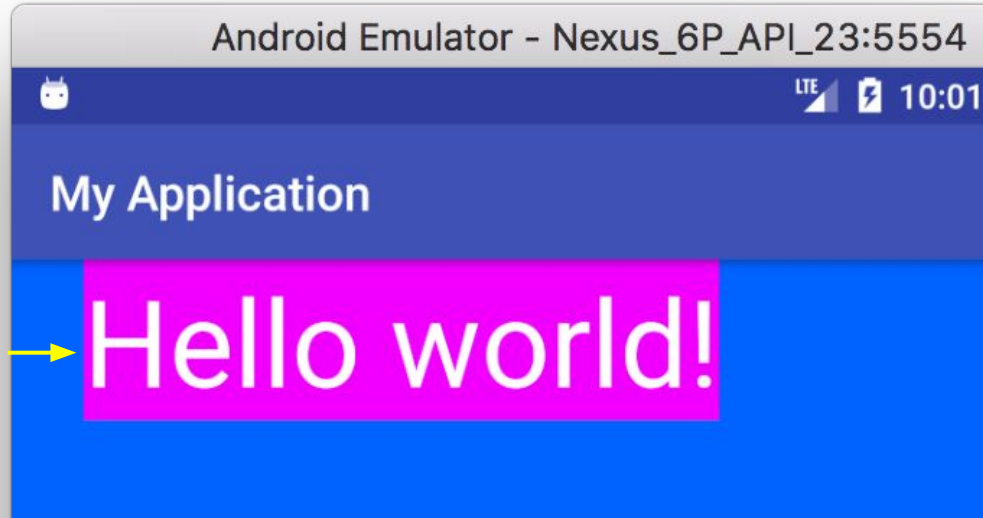
```
<TextView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:text="Hello world!"
    android:textSize="50sp"
    android:textColor="#FFFFFF"
    android:background="#f200ff"
    />
```



Layout Margins

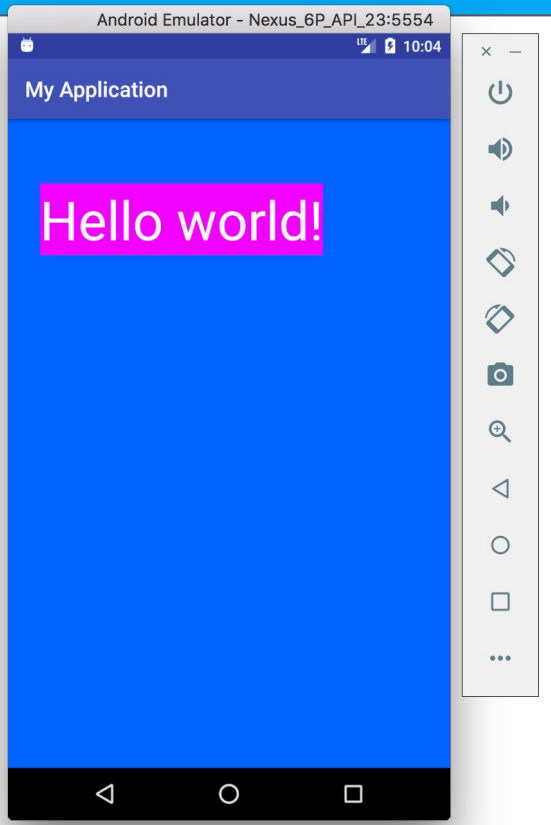
Layout Margin

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:text="Hello world!"
    android:textSize="50sp"
    android:textColor="#FFFFFF"
    android:background="#f200ff"
    />
```



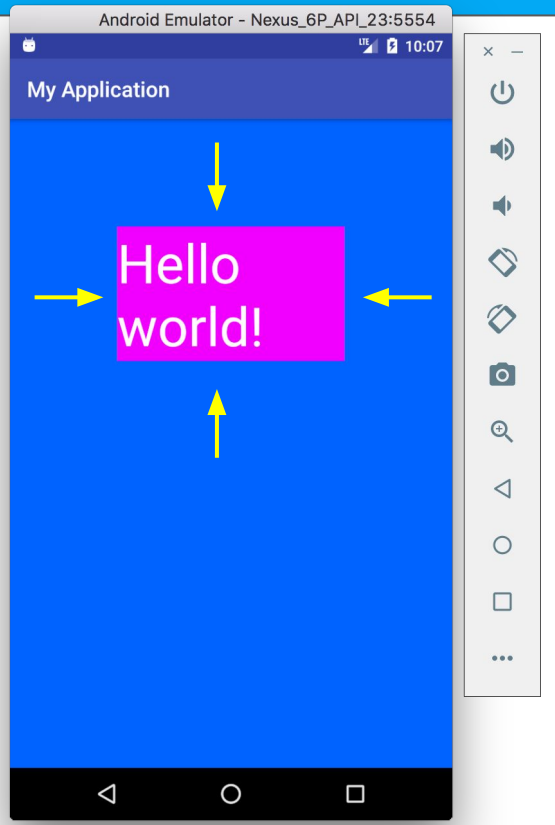
Layout Margin

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="60dp"
    android:text="Hello world!"
    android:textSize="50sp"
    android:textColor="#FFFFFF"
    android:background="#f200ff"
    />
```



Layout Margin

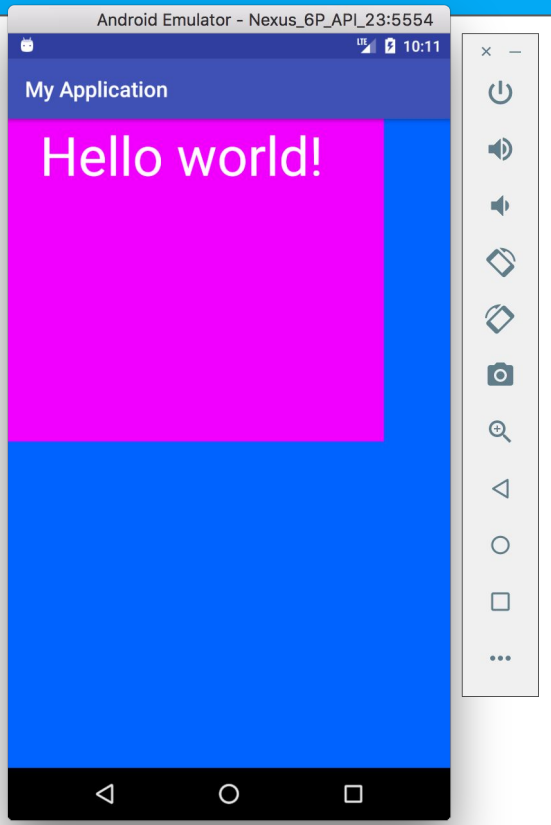
```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_margin="100dp"
    android:text="Hello world!"
    android:textSize="50sp"
    android:textColor="#FFFFFF"
    android:background="#f200ff"
    />
```



Padding

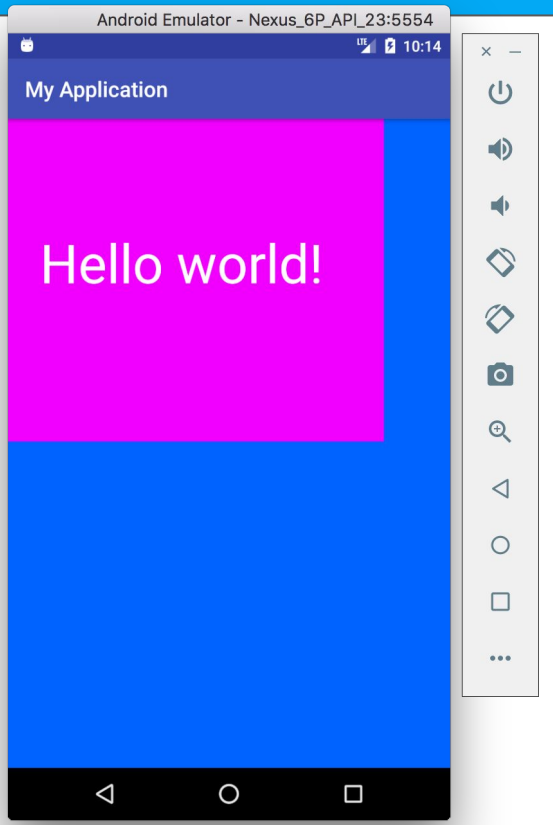
Layout Margin

```
<TextView
    android:layout_width="350dp"
    android:layout_height="300dp"
    android:paddingStart="30dp"
    android:text="Hello world!"
    android:textSize="50sp"
    android:textColor="#FFFFFF"
    android:background="#f200ff"
    />
```



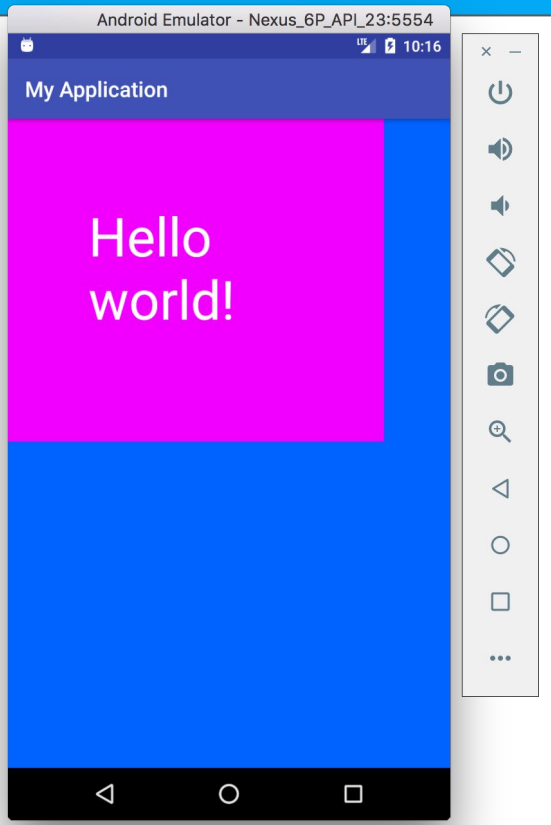
Layout Margin

```
<TextView
    android:layout_width="350dp"
    android:layout_height="300dp"
    android:paddingStart="30dp"
    android:paddingTop="100dp"
    android:text="Hello world!"
    android:textSize="50sp"
    android:textColor="#FFFFFF"
    android:background="#f200ff"
    />
```



Layout Margin

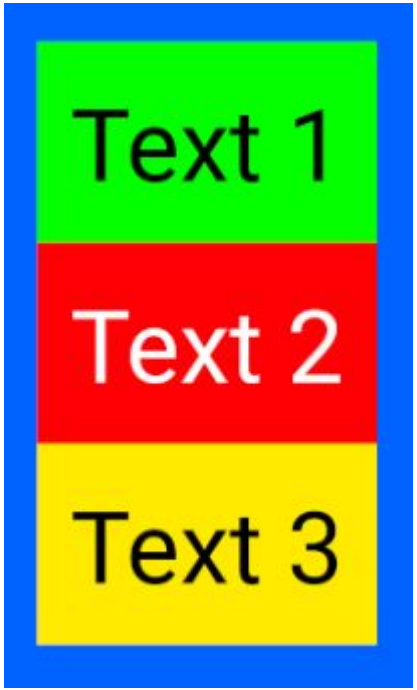
```
<TextView
    android:layout_width="350dp"
    android:layout_height="300dp"
    android:padding="75dp"
    android:text="Hello world!"
    android:textSize="50sp"
    android:textColor="#FFFFFF"
    android:background="#f200ff"
    />
```



Visibility

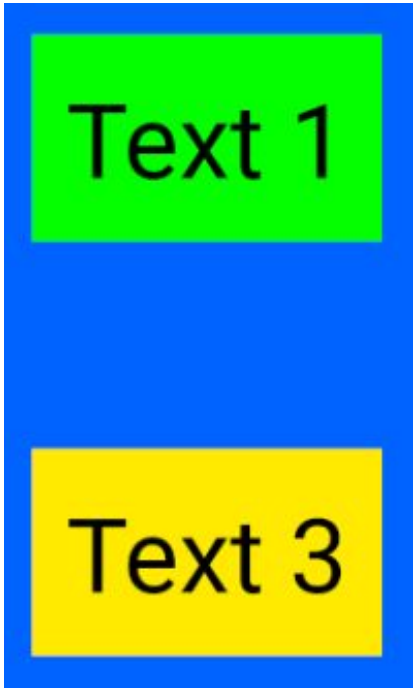
Visibility

```
android:visibility="visible"

android:visibility="invisible"

android:visibility="gone"
```



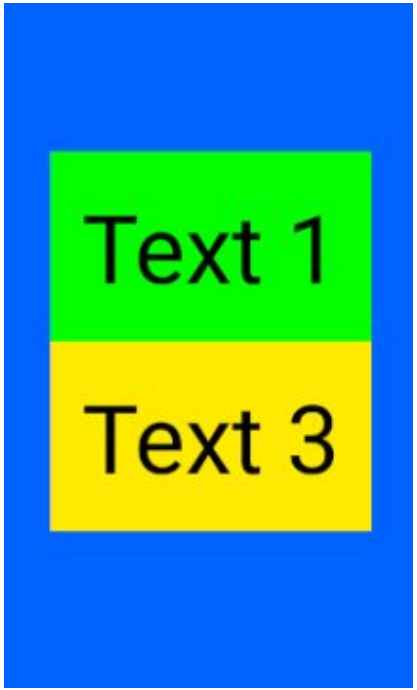
Visibility

```
android:visibility="visible"

android:visibility="invisible"

android:visibility="gone"
```



Visibility

```
android:visibility="visible"

android:visibility="invisible"

android:visibility="gone"
```



Questions ?

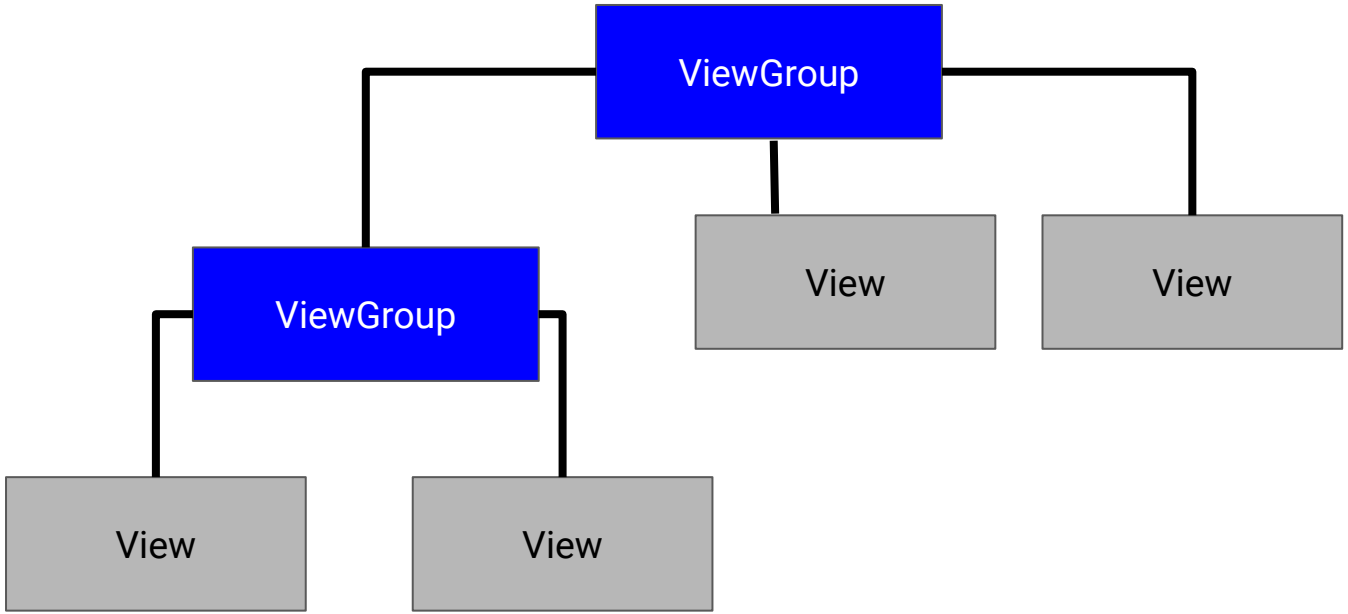
ViewGroups

ViewGroup (layout)

- A Container
- Can hold Views & other ViewGroups
- “Knows” how to position its **children's**

https://developer.android.com/reference/android/view/ViewGroup.html

ViewGroup



Introducing the LinearLayout, the RelativeLayout and the ConstraintLayout

LinearLayout

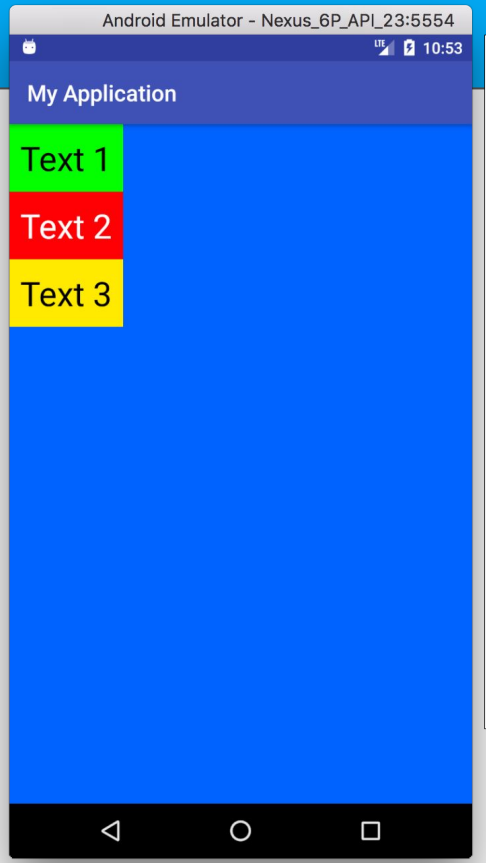
LinearLayout

- Lays its children in a specific **orientation**.
- Can divide existing space by **weight**.
- Can set its children's **gravity**.

LinearLayout - orientation

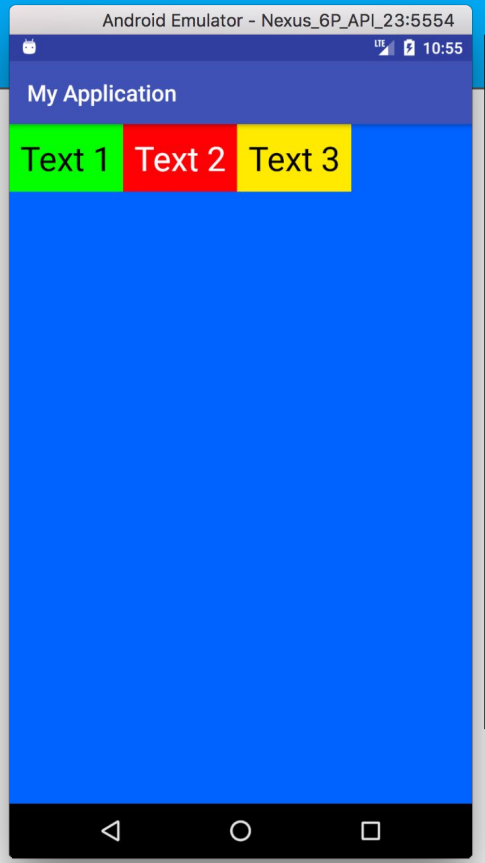
LinearLayout - orientation

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@color/blue">

    <TextView .../>
    <TextView .../>
    <TextView .../>

</LinearLayout>
```



LinearLayout - orientation

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:background="@color/blue">

    <TextView .../>
    <TextView .../>
    <TextView .../>

</LinearLayout>
```



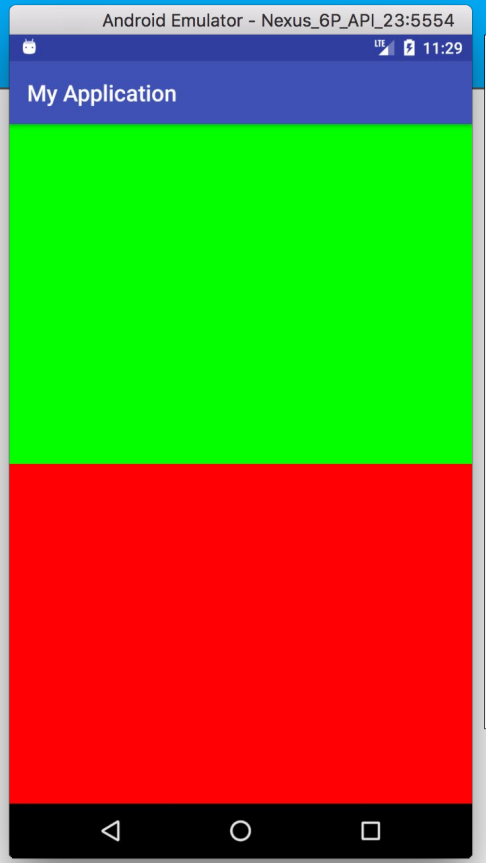
LinearLayout - weight

LinearLayout - layout weight

- Assigns an "importance" value to a view
- “Importance” = how much space the View should occupy
- Default weight = 0

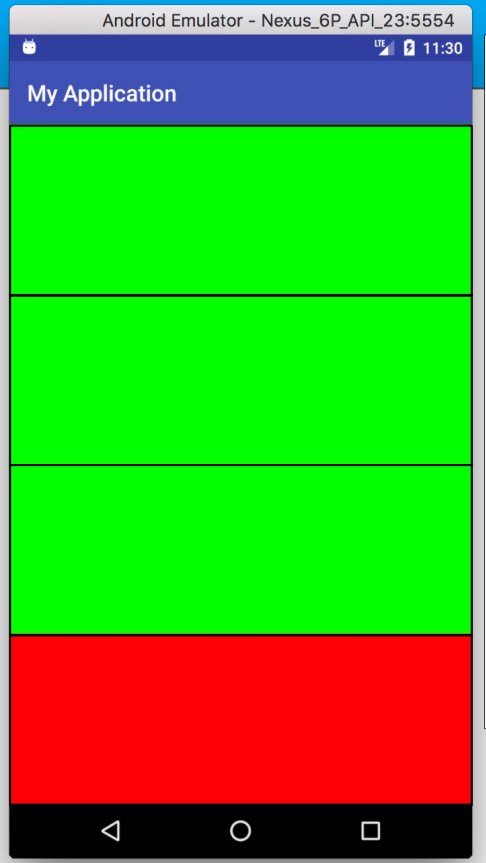
LinearLayout - layout weight

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@color/blue">

    <View
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:layout_weight="1" />
    <View
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:layout_weight="1” />
</LinearLayout>
```



LinearLayout - layout weight

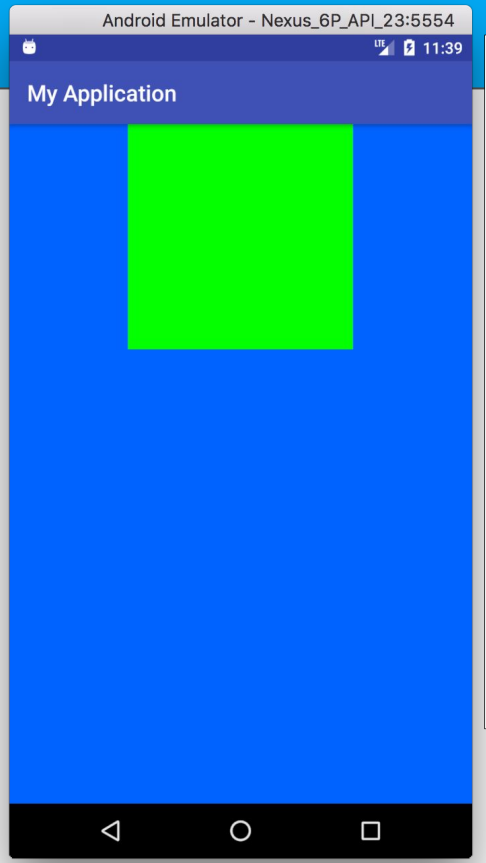
```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@color/blue">

    <View
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:layout_weight="3" />
    <View
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:layout_weight="1” />
</LinearLayout>
```



LinearLayout - gravity

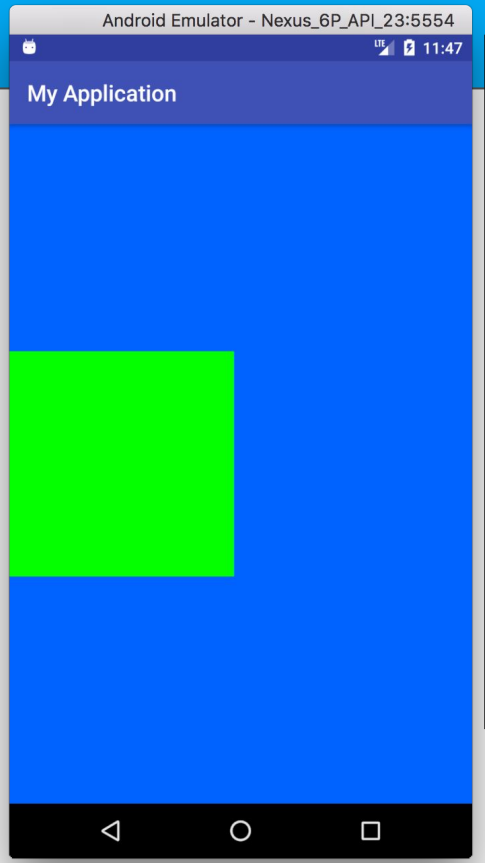
LinearLayout - layout gravity

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@color/blue">

    <View
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_gravity="center_horizontal"
        android:background="#04ff00" />

</LinearLayout>
```



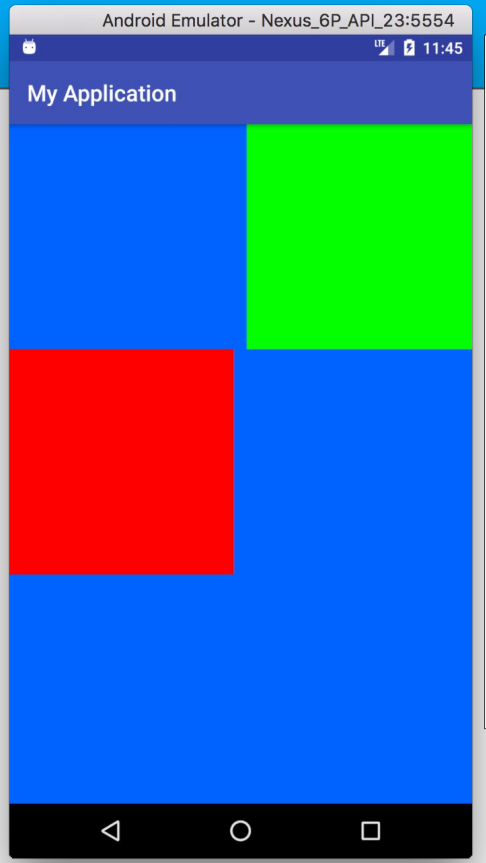
LinearLayout - layout gravity

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:background="@color/blue">

    <View
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_gravity="center_vertical"
        android:background="#04ff00" />

</LinearLayout>
```



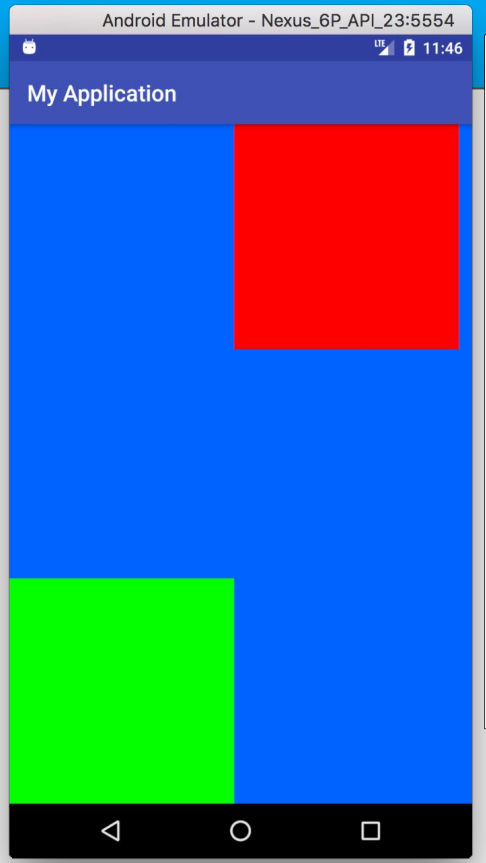
LinearLayout - layout gravity

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@color/blue">

    <View
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_gravity="right" />
    <View
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_gravity="left" />

</LinearLayout>
```



LinearLayout - layout gravity

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    android:background="@color/blue">

    <View
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_gravity="bottom" />
    <View
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_gravity="top" />

</LinearLayout>
```



Questions ?

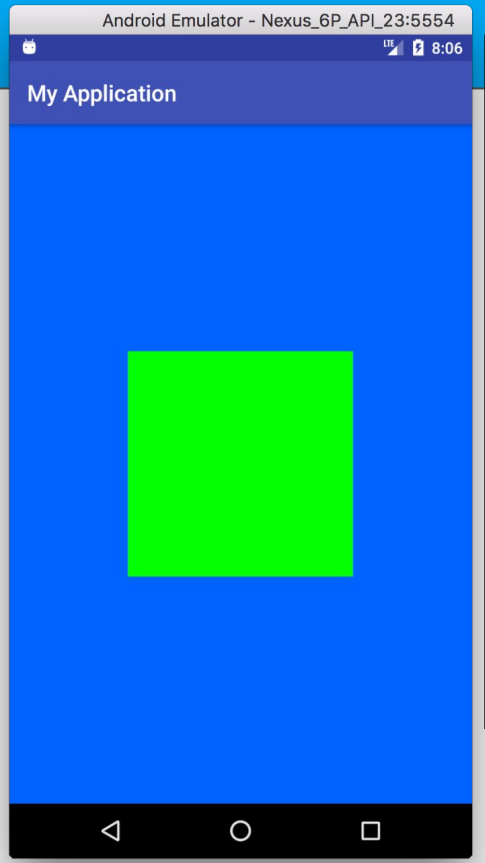
RelativeLayout

RelativeLayout

- Lays its children's **relative** to each other,
  or **relative** to itself.

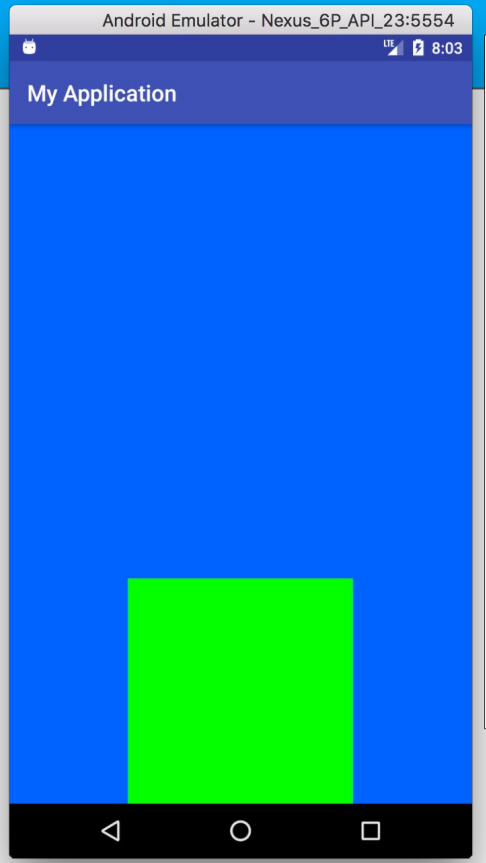
RelativeLayout

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/blue">

    <View
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_centerInParent="true" />

</RelativeLayout>
```



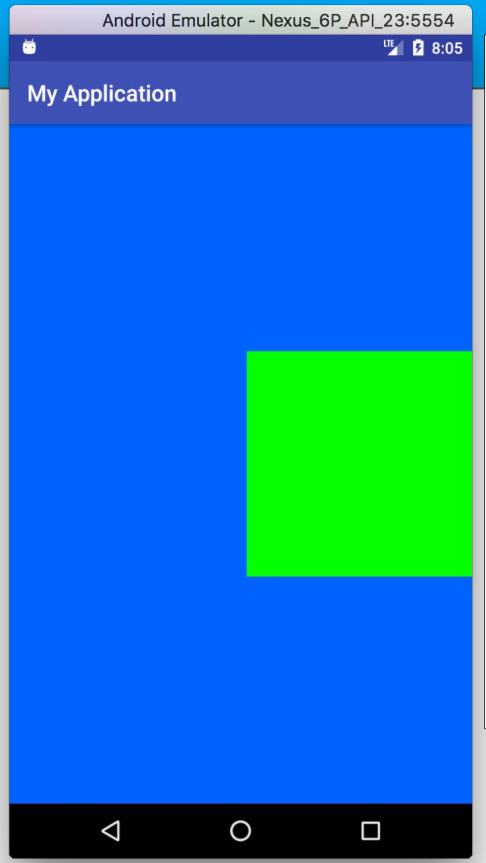
RelativeLayout

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/blue">

    <View
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_centerHorizontal="true"
        android:layout_alignParentBottom="true" />

</RelativeLayout>
```



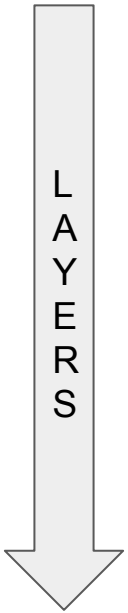
RelativeLayout

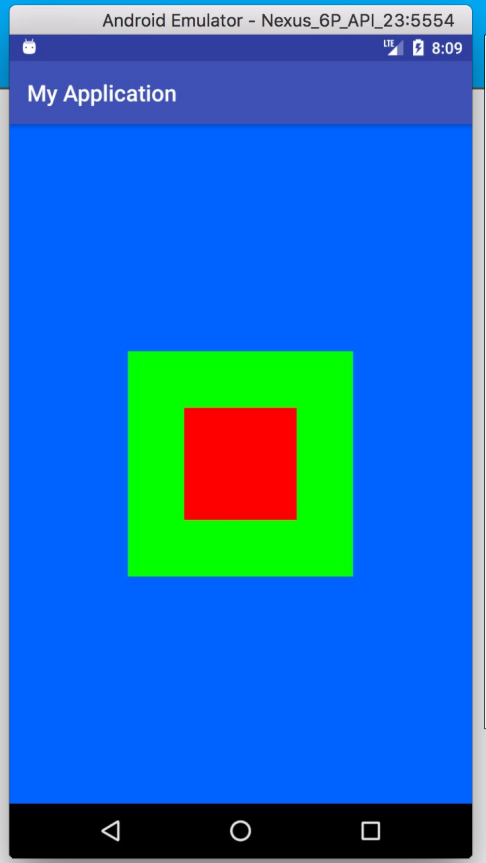
```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/blue">

    <View
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_centerVertical="true"
        android:layout_alignParentEnd="true" />

</RelativeLayout>
```



RelativeLayout

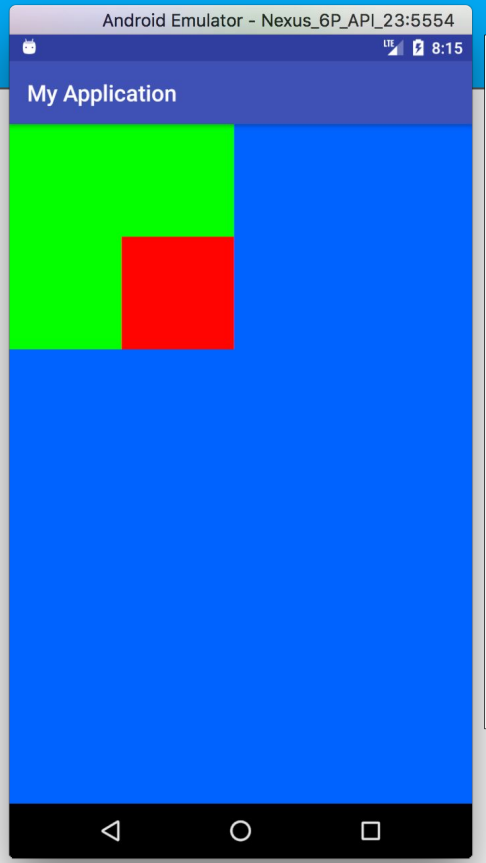
```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/blue">
    <View
         android:id="@+id/green_square"
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:layout_centerInParent="true"
        android:background="#04ff00"/>
    <View
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:layout_centerInParent="true"
        android:background="#ff0000"/>
</RelativeLayout>
```





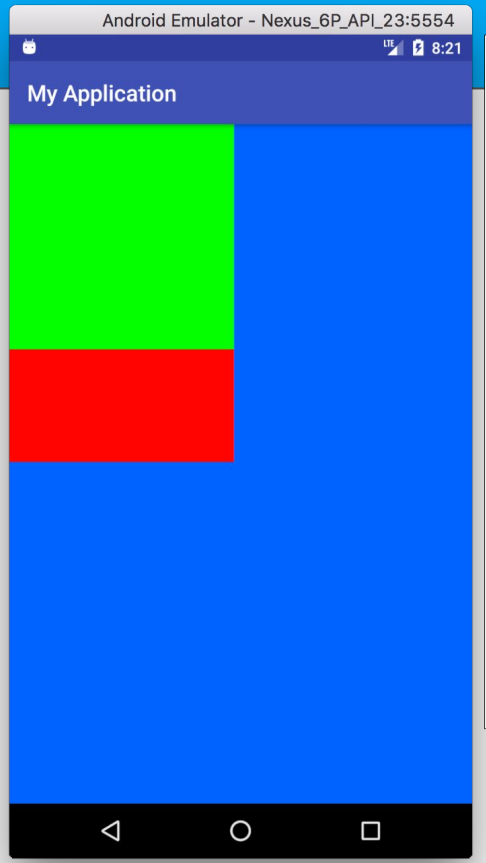
RelativeLayout

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/blue">

    <View
        android:id="@+id/green_square"
        android:layout_width="200dp"
        android:layout_height="200dp" />
    <View
        android:id="@+id/red_square"
        android:layout_width="100dp"
        android:layout_height="100dp"
        android:layout_alignBottom="@id/green_square"
        android:layout_alignEnd="@id/green_square" />

</RelativeLayout>
```



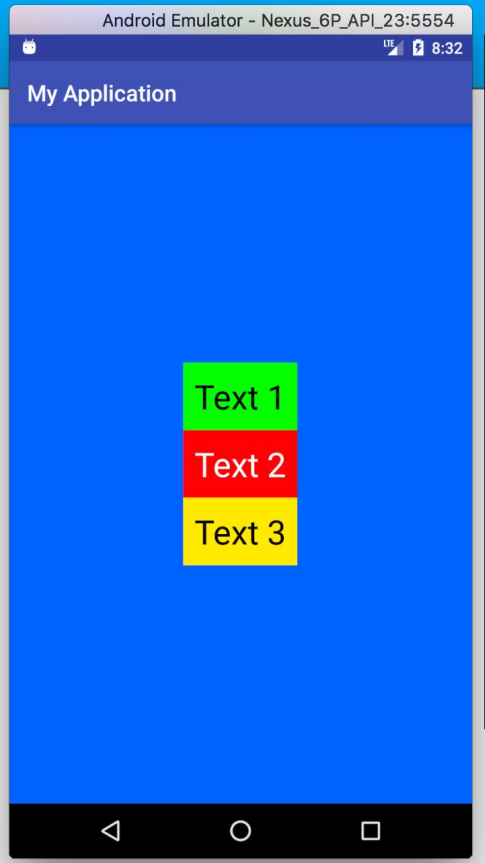
RelativeLayout

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/blue">

    <View
        android:id="@+id/green_square"
        android:layout_width="200dp"
        android:layout_height="200dp" />
    <View
        android:id="@+id/red_square"
        android:layout_width="wrap_content"
        android:layout_height="100dp"
        android:layout_below="@id/green_square"
        android:layout_alignEnd="@id/green_square"
        android:layout_alignStart="@id/green_square" />
</RelativeLayout>
```



RelativeLayout + LinearLayout

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/blue">

        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerInParent="true"
            android:orientation="vertical">

         ...
        </LinearLayout>

</RelativeLayout>
```



Questions ?

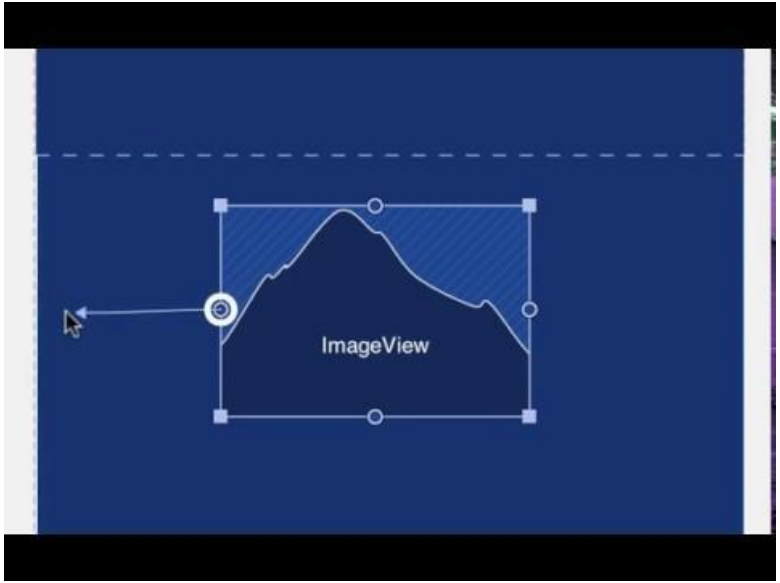
ConstraintLayout

ConstraintLayout

- Every view must have at least two **constraints**:
  one horizontal and one vertical.
- Lays its children's **relative** to each other, or relative to itself.

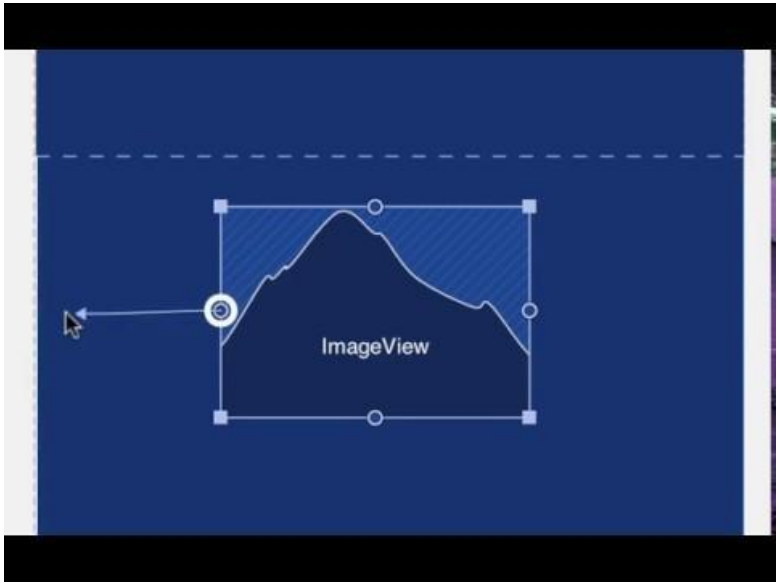
ConstraintLayout - add constraint

- Click a constraint handle and drag it to an available anchor point.



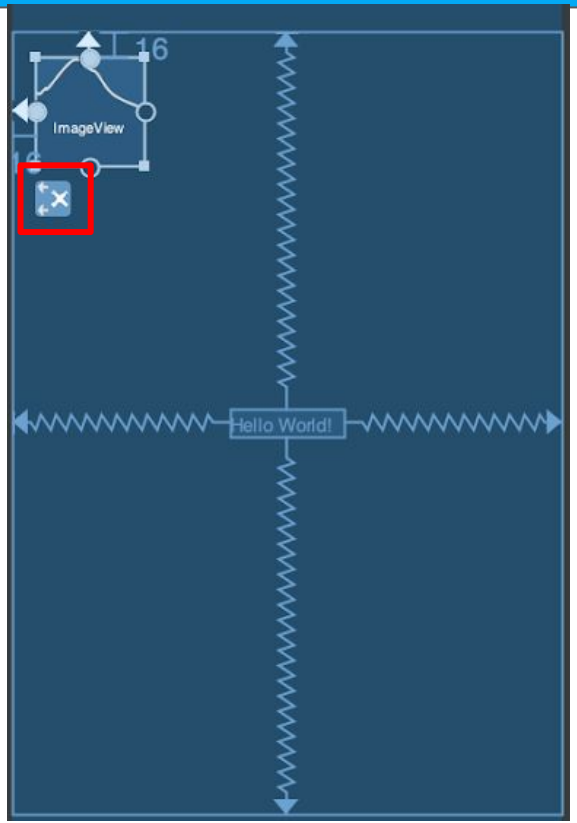
ConstraintLayout - add constraint

- If we want to locate the view in the center we can add the opposite handler.

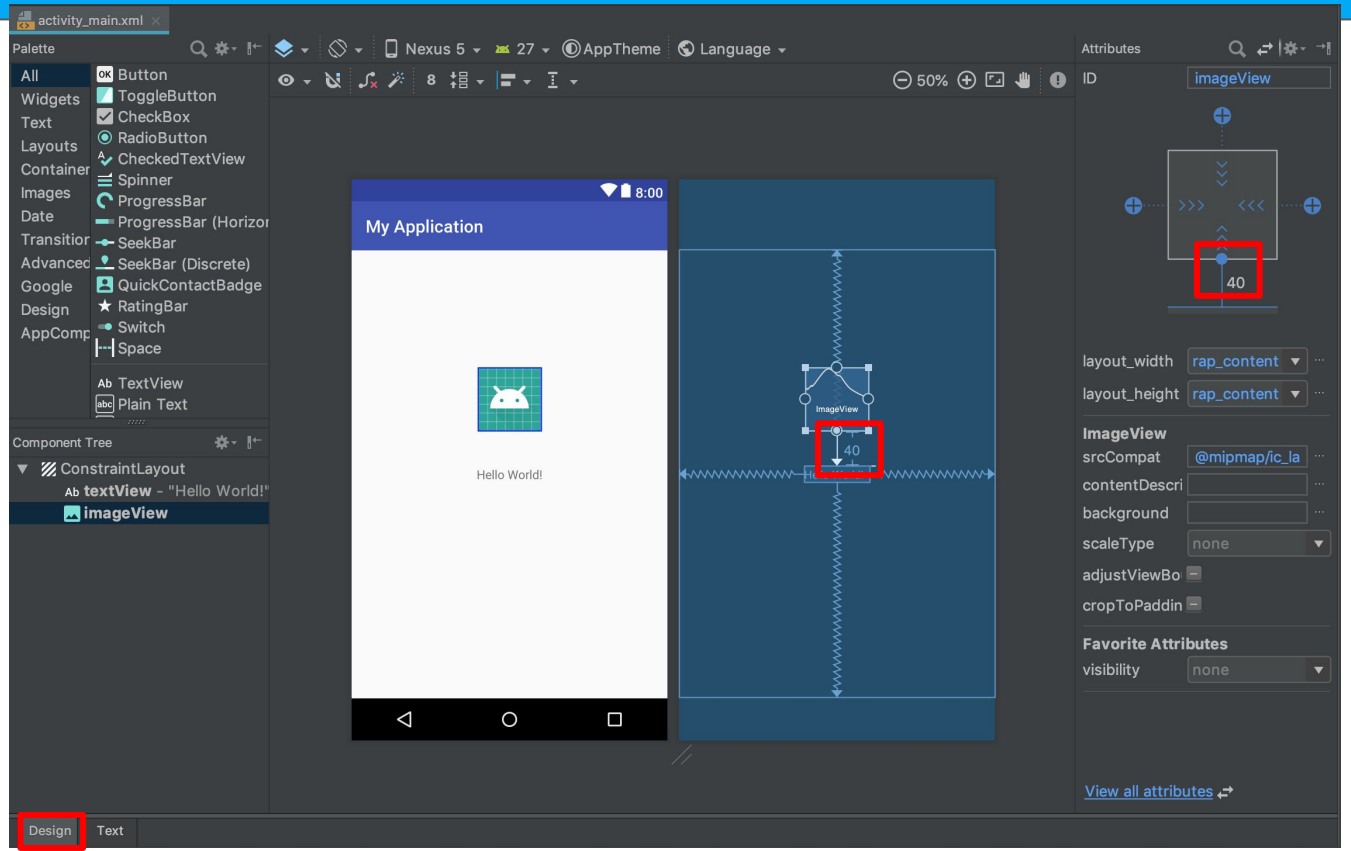


ConstraintLayout - remove constraint

- Hover the view and the remove option will appear.



ConstraintLayout - example design view



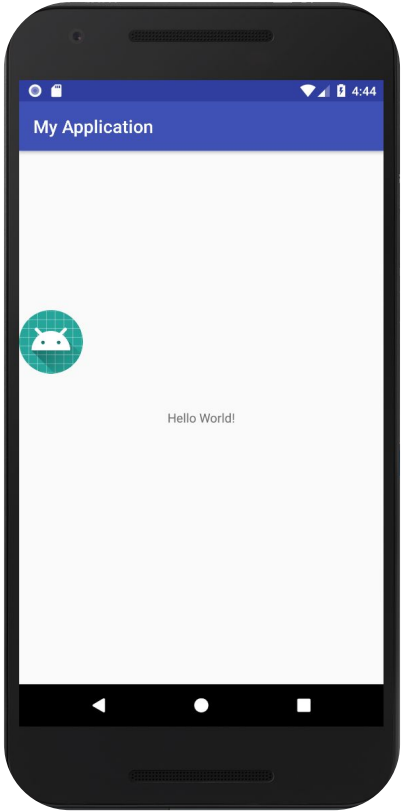
ConstraintLayout - example design view

activity_main.xml

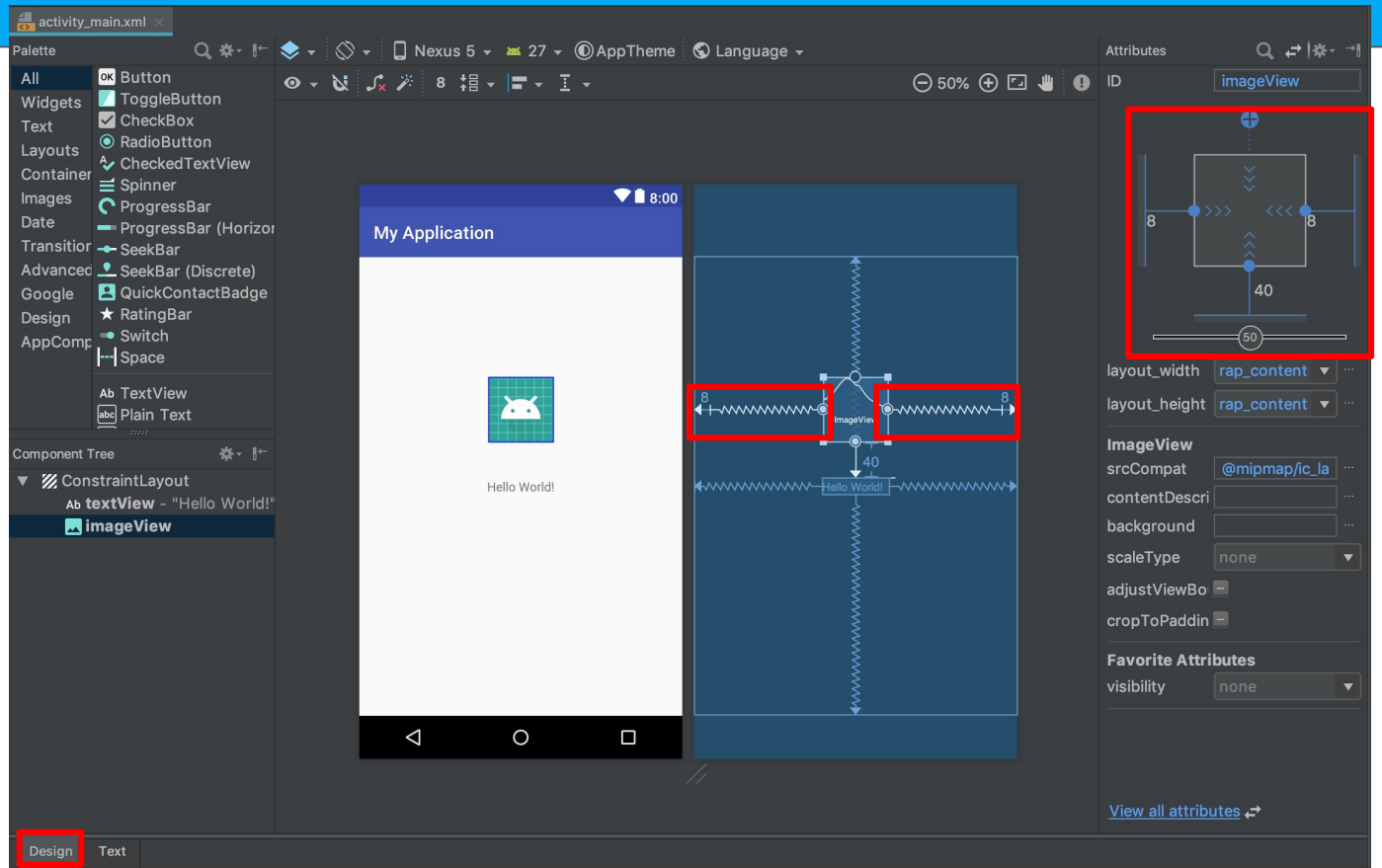
```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.myapplication.MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="40dp"
        app:layout_constraintBottom_toTopOf="@+id/textView"
        app:srcCompat="@mipmap/ic_launcher"
        tools:layout_editor_absoluteX="144dp" />

</android.support.constraint.ConstraintLayout>
```

Design | Text

ConstraintLayout - example emulator view



ConstraintLayout -example design view fixed



ConstraintLayout - example design view fixed

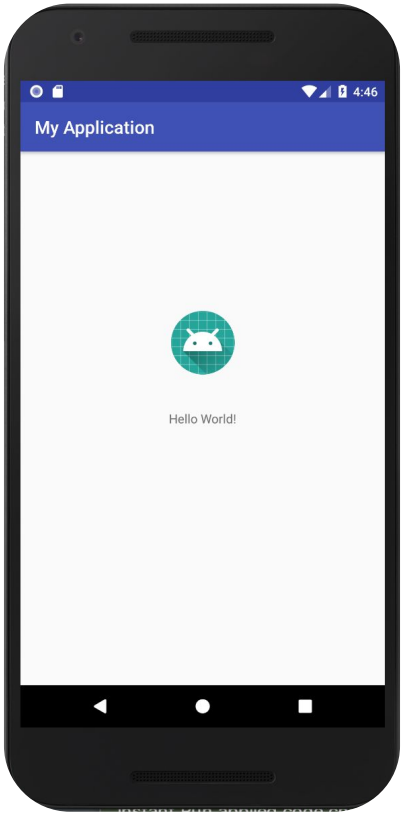
activity_main.xml

android.support.constraint.ConstraintLayout | TextView

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.myapplication.MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="40dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        app:layout_constraintBottom_toTopOf="@+id/textView"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:srcCompat="@mipmap/ic_launcher" />

</android.support.constraint.ConstraintLayout>
```

Design | Text

ConstraintLayout -example emulator view fixed



ConstraintLayout - example text view

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.myapplication.MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <ImageView
```

ConstraintLayout

Notice that in order to use ConstraintLayout you have to ensure:

1. You have the `maven.google.com` repository in your module-level `build.gradle` file:

```
repositories {
    google()
}
```

ConstraintLayout

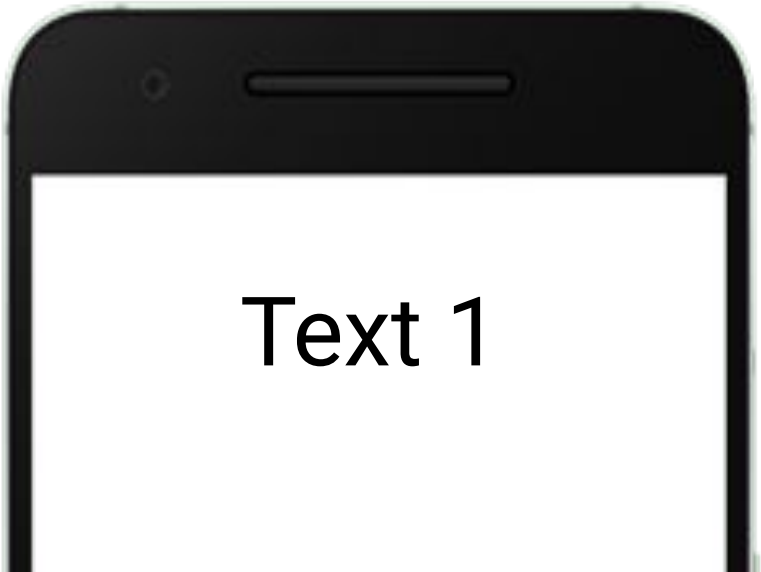
2. Add the library as a dependency in the same `build.gradle` file:

```
dependencies {

    implementation 'com.android.support.constraint:constraint-layout:1.1.3'

}
```

Questions ?

Common Views

TextView




```
android:text="Text 1"

android:textSize="30sp"

android:textColor="#000000"
```

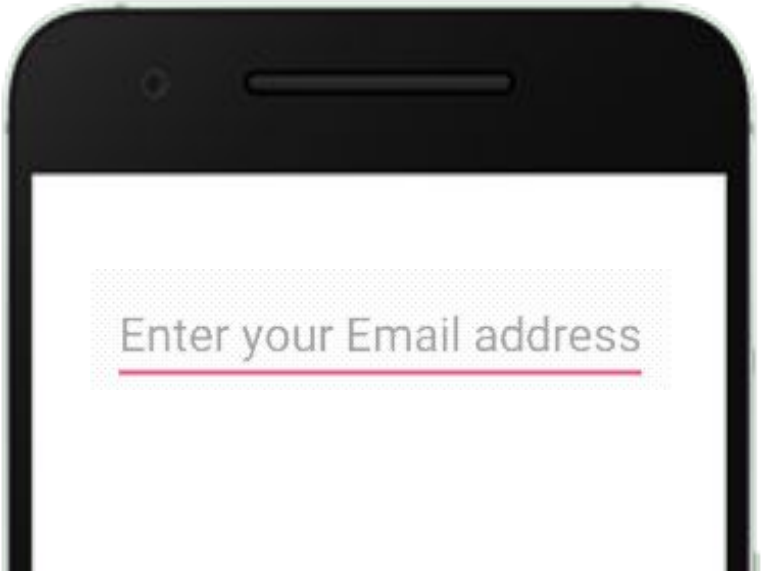
Scale-independent pixel (sp)

- Same as dp.
- Scaled by the user's preferred text size.

ImageView

```
android:src="@drawable/img"
android:scaleType="center"
```

EditText




```
android:hint="Enter your email address"
android:textColorHint="#FF0000"
android:text="Hello world"
android:textColor="#FFFFFF"
android:textSize="18sp"
```

Button


```
android:text="CLICK ME"
android:textColor="#FFFFFF"
android:textSize="18sp"
```

`findViewById()`
From XML to Java

findViewById() inside Activity

Before Android 26 (Oreo) - casting required

```
TextView tv = (TextView) findViewById(R.id.my_view_id);
```

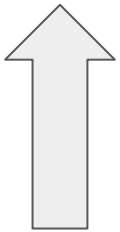
Since Android 26 (Oreo)

```
TextView tv = findViewById(R.id.my_view_id);
```

Creating Views with Java code

Views with Java code

```
TextView tv = new TextView(context);
```



View class basic constructor requires a **Context**

Context

- Interface to global information about the application environment.
- Allows access to application-specific resources.
- Used for application-level operations such as launching activities.

Views with Java code

```
LinearLayout myLayout = findViewById(R.id.content);

TextView tv = new TextView(this);
tv.setText("Created by code!");

LinearLayout.LayoutParams linearLayoutParams = new LinearLayout.LayoutParams(
        ViewGroup.LayoutParams.WRAP_CONTENT, // width
        ViewGroup.LayoutParams.MATCH_PARENT); // height

linearLayoutParams.gravity = Gravity.CENTER_HORIZONTAL;
tv.setLayoutParams(linearLayoutParams);

myLayout.addView(tv);
```

Views with Java code

```
LinearLayout myLayout = findViewById(R.id.content);

TextView tv = new TextView(this);
tv.setText("Created by code!");

LinearLayout.LayoutParams linearLayoutParams = new LinearLayout.LayoutParams(
        ViewGroup.LayoutParams.WRAP_CONTENT, // width
        ViewGroup.LayoutParams.MATCH_PARENT); // height

linearLayoutParams.gravity = Gravity.CENTER_HORIZONTAL;
tv.setLayoutParams(linearLayoutParams);

myLayout.addView(tv);
```

Views with Java code

```
LinearLayout myLayout = findViewById(R.id.content);

TextView tv = new TextView(this);
tv.setText("Created by code!");

LinearLayout.LayoutParams linearLayoutParams = new LinearLayout.LayoutParams(
        ViewGroup.LayoutParams.WRAP_CONTENT, // width
        ViewGroup.LayoutParams.MATCH_PARENT); // height

linearLayoutParams.gravity = Gravity.CENTER_HORIZONTAL;
tv.setLayoutParams(linearLayoutParams);

myLayout.addView(tv);
```

Views with Java code

```
LinearLayout myLayout = findViewById(R.id.content);

TextView tv = new TextView(this);
tv.setText("Created by code!");

LinearLayout.LayoutParams linearLayoutParams = new LinearLayout.LayoutParams(
        ViewGroup.LayoutParams.WRAP_CONTENT, // width
        ViewGroup.LayoutParams.MATCH_PARENT); // height

linearLayoutParams.gravity = Gravity.CENTER_HORIZONTAL;
tv.setLayoutParams(linearLayoutParams);

myLayout.addView(tv);
```

Views with Java code

```
LinearLayout myLayout = findViewById(R.id.content);

TextView tv = new TextView(this);
tv.setText("Created by code!");

LinearLayout.LayoutParams linearLayoutParams = new LinearLayout.LayoutParams(
        ViewGroup.LayoutParams.WRAP_CONTENT, // width
        ViewGroup.LayoutParams.MATCH_PARENT); // height

linearLayoutParams.gravity = Gravity.CENTER_HORIZONTAL;
tv.setLayoutParams(linearLayoutParams);

myLayout.addView(tv);
```

Views with Java code

```
LinearLayout myLayout = findViewById(R.id.content);

TextView tv = new TextView(this);
tv.setText("Created by code!");

LinearLayout.LayoutParams linearLayoutParams = new LinearLayout.LayoutParams(
        ViewGroup.LayoutParams.WRAP_CONTENT, // width
        ViewGroup.LayoutParams.MATCH_PARENT); // height

linearLayoutParams.gravity = Gravity.CENTER_HORIZONTAL;
tv.setLayoutParams(linearLayoutParams);

myLayout.addView(tv);
```

setOnClickListener()

```
tv.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View view) {
        // Do something...
    }
});
```

Questions ?

Theory time!
How it all works?

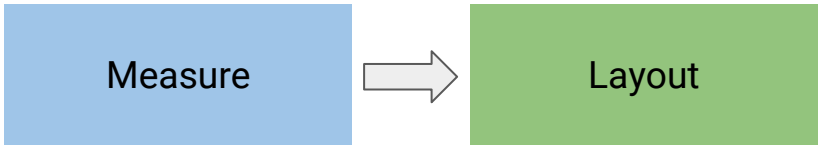
How Views get’s drawn on screen?

How Views get’s drawn on screen?

Measure

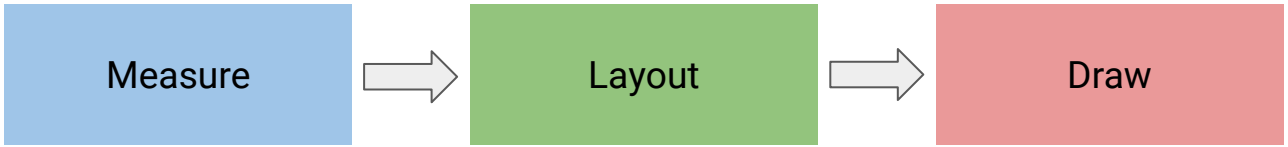
- Traversing the layout hierarchy from the root.
- Parents ask children to measure themselves, **given constraints.**

How Views get’s drawn on screen?



- Parents position their children on screen based on the child measured size.
- Every View now know its place in the layout.

How Views get’s drawn on screen?



- Parents draw themselves, then asks children to draw themselves.

Exercise time!

Let’s practice what we’ve learned!

Start here: Exercise #2