Drawables, and Styles, and android.R

Oh my!

Drawables

- Generic class for drawing content to the screen
- Can represent image resources and “shapes”
- Can be combined and modified via XML to make complex resources
- Easier than subclassing

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

Drawables

- Bitmap
- XML Bitmap
- Nine-patch
- Layer List
- State List
- Level List
- Transition
- Inset
- Clip
- Scale
- Shape

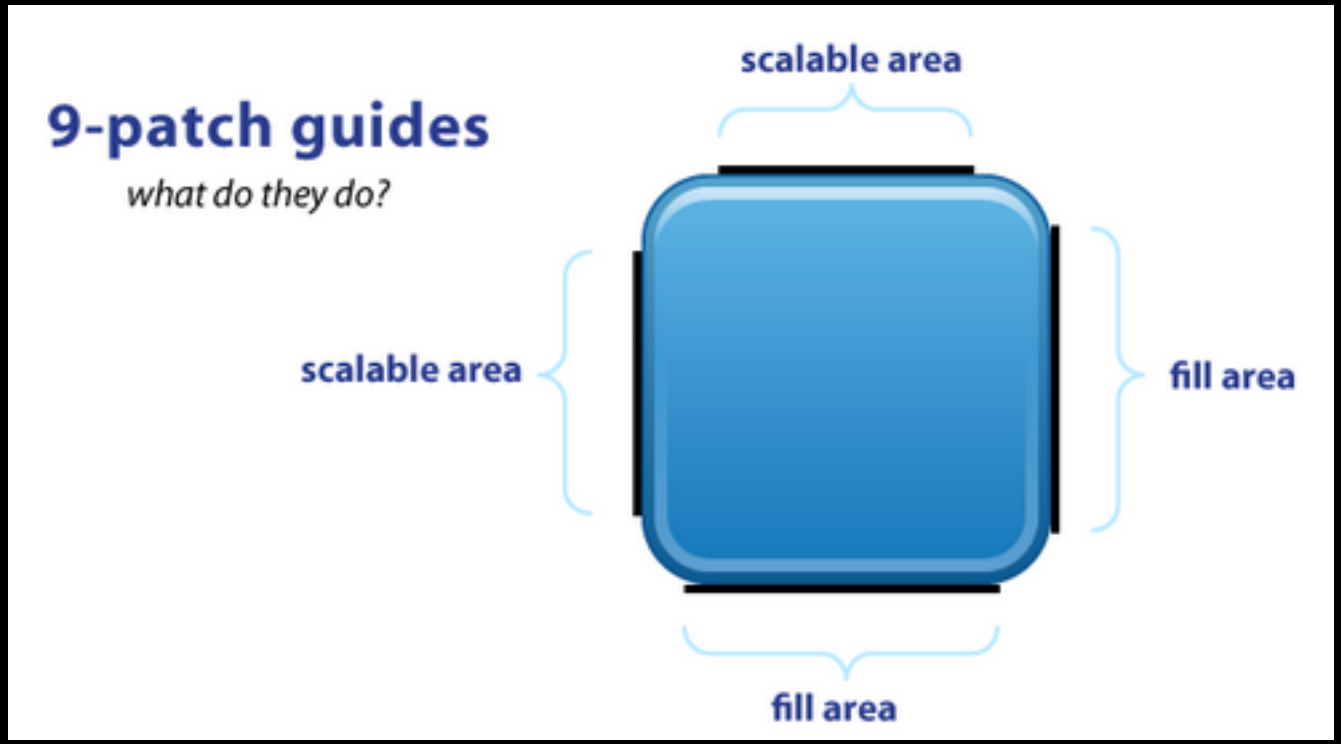
XML Bitmap

- Refers to a bitmap resource
- Defines filter and scaling rules

```
<?xml version="1.0" encoding="utf-8"?>
<bitmap
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:src="@[package:]drawable/drawable_resource"
    android:antialias=["true" | "false"]
    android:dither=["true" | "false"]
    android:filter=["true" | "false"]
    android:gravity=["top" | "bottom" | "left" | "right" | "center_vertical" |
                     "fill_vertical" | "center_horizontal" | "fill_horizontal" |
                     "center" | "fill" | "clip_vertical" | "clip_horizontal"]
    android:tileMode=["disabled" | "clamp" | "repeat" | "mirror"] />
```

Nine-Patch

- PNG with extra pixels to define stretchable regions and content bounds



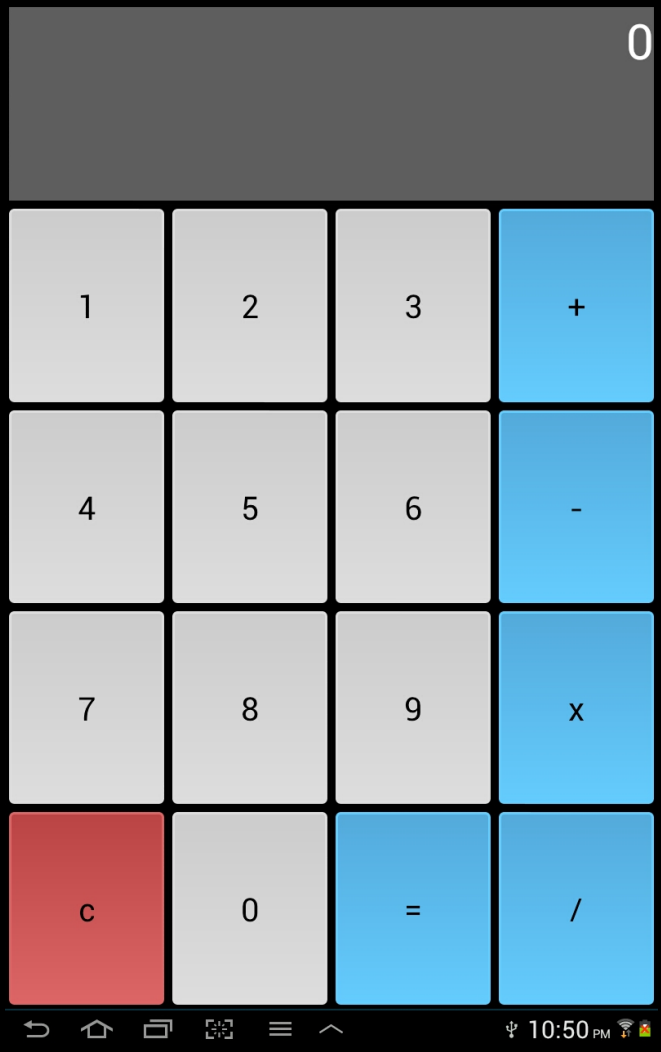
- http://radleymarx.com/blog/simple-guide-to-9-patch/

Shape

- Define a shape, its fill/stroke, color/gradient
- Rect, Oval, Line, Ring

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <gradient
        android:startColor="#FFFF0000"
        android:endColor="#80FF00FF"
        android:angle="45"/>
    <padding android:left="7dp"
        android:top="7dp"
        android:right="7dp"
        android:bottom="7dp" />
    <corners android:radius="8dp" />
</shape>
```

Shape



Layer List

- Combines multiple drawables into a single drawable resource.
- Drawn in order, last on top.
- Can be used to add simple effects to existing drawables.

```
<?xml version="1.0" encoding="utf-8"?>
<layer-list
    xmlns:android="http://schemas.android.com/apk/res/android" >
    <item
        android:drawable="@[package:]drawable/drawable_resource"
        android:id="@[+][package:]id/resource_name"
        android:top="dimension"
        android:right="dimension"
        android:bottom="dimension"
        android:left="dimension" />
</layer-list>
```

State List

- Specifies a set of drawables for different view states.
- Pressed, Enabled, Selected….
- Very useful for buttons (up and down state)

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_pressed="true"
          android:drawable="@drawable/button_pressed" /> <!-- pressed -->
    <item android:state_focused="true"
          android:drawable="@drawable/button_focused" /> <!-- focused -->
    <item android:state_hovered="true"
          android:drawable="@drawable/button_focused" /> <!-- hovered -->
    <item android:drawable="@drawable/button_normal" /> <!-- default -->
</selector>
```

Level List

- Specifies a set of drawables displayed according to a level value.
- Use LevelListDrawable.setLevel(…) to change image.
- Great for 3 state switches.

```
<?xml version="1.0" encoding="utf-8"?>
<level-list
    xmlns:android="http://schemas.android.com/apk/res/android" >
    <item
        android:drawable="@drawable/drawable_resource"
        android:maxLevel="integer"
        android:minLevel="integer" />
</level-list>
```

Others

- Transition
- Inset
- Clip
- Scale
- See documentation at:

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

Combine and Save!

- Most of these can be used in combination.
- Use a smaller set of base images to build out all resources.
- Use shapes instead of images if possible. Takes up less space, scales better.

Styles

- Inspired by CSS
- Lets you define sets of attributes for views
- Reusable, extensible, maintainable
- Dynamic like drawables and layouts
- DRY

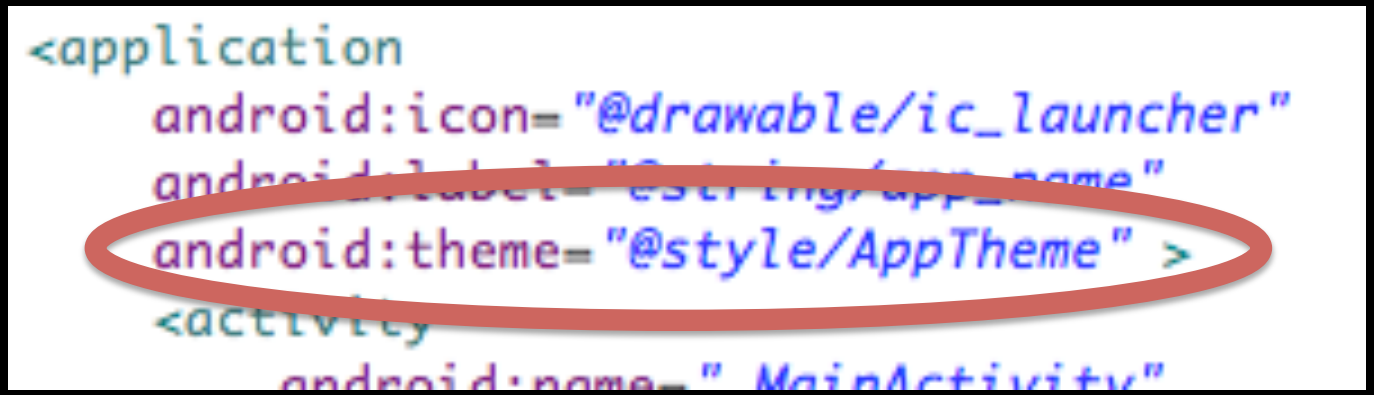
Styles

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="CustomText" parent="@style/Text">
        <item name="android:textSize">20sp</item>
        <item name="android:textColor">#008</item>
    </style>
</resources>

<EditText
    style="@style/CustomText"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="Hello, World!" />
```

Themes

- Themes are styles set on the Activity or Application (in the manifest).
- Overrides default values for Views in the Activity/Appliation.
- Separates the design from the content.



Extending Styles

- Styles can inherit from one another.
- Makes it easy to make small changes to existing styles.

```
<resources>
    <style name="AppTheme" parent="android:Theme.Light" >
        <!-- items go here -->
    </style>
</resources>
```

android.R

- Resources provided by SDK
- Contains many things including simple layouts, and default icons.
- In XML accessed as @android:<type>/<name>
- Defines standard themes such as Holo or the “Ice Cream Sandwich” theme

Questions?