

Libraries

- Many Android developers have produced useful **libraries**.
 - There is a **Maven repository** to store various libraries.
 - This makes it easy to add them to your Android Studio projects.
 - Most libraries use permissive licenses so that you can use them for free and can include them in the code of commercial apps/products.
 - (Some libraries must be downloaded as .JARs and added manually to your project)

maven



Adding a library to your project

- Edit the **build.gradle** file for your 'app' module and add lines to the following section at the bottom.
 - You can usually find out what file name to write below by going to various libraries' home pages / GitHub pages.

```
1 dependencies {  
2     implementation fileTree(dir: 'libs', include: ['*.jar'])  
3     implementation 'com.android.support:appcompat-v7:26.1.0'  
4     implementation 'com.android.support.constraint:constraint-layout:1.1.2'  
5     testImplementation 'junit:junit:4.12'  
6     implementation 'com.beardedhen:androidbootstrap:2.3.1'  
7     implementation 'Your library here...'  
8     implementation 'Your library here...'  
    ...  
}
```

Picasso library

- **Picasso** is a powerful library for manipulating images.
 - written by Square, inc.
 - <http://square.github.io/picasso/>
- To add Picasso to your project:



```
1 // in build.gradle
2 dependencies {
3     ...
4     implementation 'com.squareup.picasso:picasso:2.5.2'
5 }
```

```
1 <!-- in AndroidManifest.xml -->
2 <uses-permission android:name="android.permission.INTERNET" />
```

Displaying a web photo

- In your app's Java code, write:

```
1 Picasso.with(this)
2     .load("url")
3     .into(ImageView);
```

- Example:

```
1 // show a cute puppy photo
2 ImageView img = (ImageView) findViewById(R.id.photo);
3 Picasso.with(this)
4     .load("http://www.martystepp.com/dogs/daisy-01.jpg")
5     .into(img);
```

Picasso image methods

Method	Description
<code>centerCrop()</code>	center and crop image inside view
<code>centerInside()</code>	resize image proportionally inside view
<code>error(<i>id</i>)</code>	show given drawable as error
<code>fetch()</code>	download image in the background
<code>fit()</code>	resize image to fit view bounds
<code>get()</code>	return image as a Bitmap
<code>into(<i>view</i>)</code>	puts image into given view
<code>placeholder(<i>id</i>)</code>	show given drawable while loading
<code>resize(<i>width</i>, <i>height</i>)</code>	change image size in pixels
<code>rotate(<i>degrees</i>)</code>	rotate clockwise
<code>tag("tag")</code>	attaches a "tag" to a loading image (useful for bulk operations shown later)
<code>transform(<i>trans</i>)</code>	apply complex transformations

Picasso methods

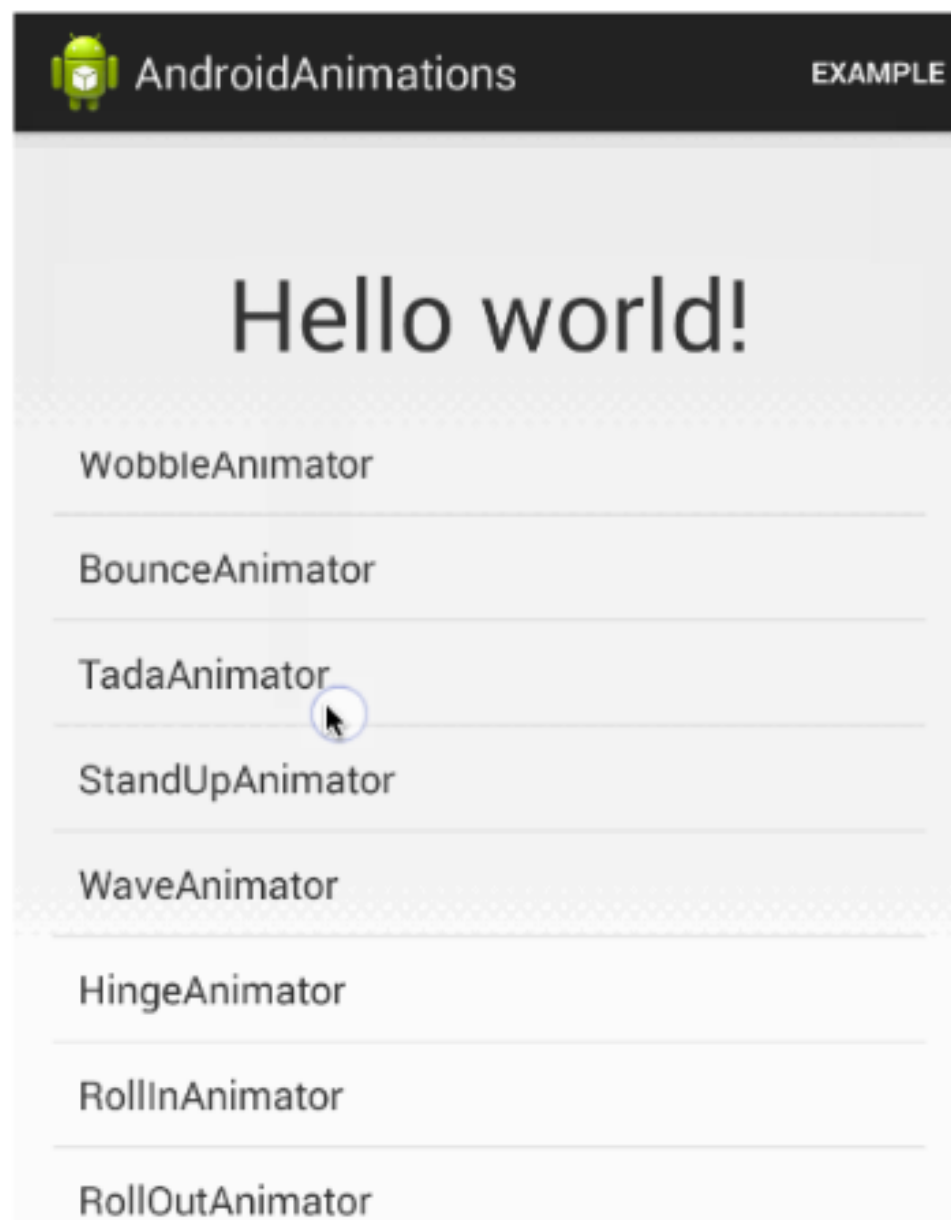
Method	Description
<code>cancelRequest(<i>view</i>)</code>	abort any image loading in that view
<code>cancelTag("tag")</code>	cancel all images with given tag
<code>invalidate("url")</code> <code>invalidate(<i>File</i>)</code>	flush out cache of given image, so it will be re-downloaded the next time
<code>load("url")</code> <code>load(<i>id</i>)</code> <code>load(<i>File</i>)</code>	load an image from various sources
<code>pauseTag("tag")</code>	pause all image loads for given tag
<code>resumeTag("tag")</code>	unpause all image loads for given tag
<code>shutdown()</code>	stop entire Picasso system
<code>with(<i>context</i>)</code>	use given activity/fragment as context

Android Animations Library

- An ambitious Android user named *daimajia* has created several libraries, including one to do **animation effects** on Views.
 - <https://github.com/daimajia/AndroidViewAnimations>
 - To use this library, add the following dependencies:

```
1 dependencies {  
2     ...  
3  
4     compile 'com.nineoldandroids:library:2.4.0'  
5     compile 'com.daimajia.easing:library:1.0.1@aar'  
6     compile 'com.daimajia.androidanimations:library:1.1.3@aar'  
7 }
```

Animations demo



Using an animation

- Anywhere in your app's Java code, write:

```
1 YoYo.with(Techniques.AnimationName)
2     . set various properties of the animation
3     .playOn(View);
```

- Example:

```
1 // play a "tada" animation for 700 ms
2 // that will affect the "edit_area" view
3 YoYo.with(Techniques.Tada)
4     .duration(700)
5     .playOn(findViewById(R.id.edit_area));
```

Animations provided

- Attention
 - Flash, Pulse, RubberBand, Shake, Swing, Wobble, Bounce, Tada, StandUp, Wave
- Special
 - Hinge, RollIn, RollOut, Landing, TakingOff, DropOut
- Bounce
 - BounceIn, BounceInDown, BounceInLeft, BounceInRight, BounceInUp
- Fade
 - FadeIn, FadeInUp, FadeInDown, FadeInLeft, FadeInRight
 - FadeOut, FadeOutDown, FadeOutLeft, FadeOutRight, FadeOutUp
- Flip
 - FlipInX, FlipOutX, FlipOutY

YoYo animation properties

Method	Description
<code>delay(<i>ms</i>)</code>	time to delay before doing animation
<code>duration(<i>ms</i>)</code>	how long the animation should last
<code>interpolate(<i>interpolator</i>)</code>	blend two animations
<code>withListener(<i>listener</i>)</code>	notify a listener on animation events
<code>playOn(<i>view</i>)</code>	start the animation on the given view

```
1 // example
2 YoYo.with(Techniques.Wobble)
3     .delay(500)
4     .duration(2000)
5     .playOn(findViewById(R.id.myview));
```

YoYo animation events

- To hear animation events, pass a class that implements interface `AnimatorListener` (or extends `AnimatorListenerAdapter`) that implements some/all of the following methods:

Method	Description
<code>onAnimationStart</code>	called when animation begins
<code>onAnimationEnd</code>	called when animation ends
<code>onAnimationCancel</code>	called if animation is canceled
<code>onAnimationRepeat</code>	called if a looping animation repeats

```
1 YoYo.with(Techniques.Wobble) // example
2   .withListener(new AnimatorListenerAdapter() {
3       public void onAnimationEnd(Animator anim) {
4           Log.v("demo", "Animation has ended!");
5       }
6   }).playOn(findViewById(R.id.myview));
```

ButterKnife library

- **ButterKnife** is a popular library intended to simplify usage of Android widgets and events in Java code.
 - written by Jake Wharton
 - <http://jakewharton.github.io/butterknife/>
- To add ButterKnife to your Android Studio project:

```
1 dependencies {  
2     ...  
3  
4     compile 'com.jakewharton:butterknife:8.5.1'  
5     annotationProcessor 'com.jakewharton:butterknife-compiler:8.5.1'  
6 }
```



ButterKnife field bindings

- Using the `@Bind` annotation, you can declare a field that will always be set to the value of a widget with a certain ID.
 - equivalent to setting it equal to `findViewById(R.id.id)`;
 - but retains its state if the activity is closed / reopened

```
1 // example: bind TextView and EditText by id
2 public class MyActivity extends Activity {
3     @BindView(R.id.mytext) TextView myText;
4     @BindView(R.id.myedit) EditText myEdit;
5
6     public void onCreate(Bundle bundle) {
7         setContentView(R.layout.activity_my);
8         ButterKnife.bind(this);
9         myEdit.setText("Wow, cool!");
10    }
11 }
```

ButterKnife event bindings

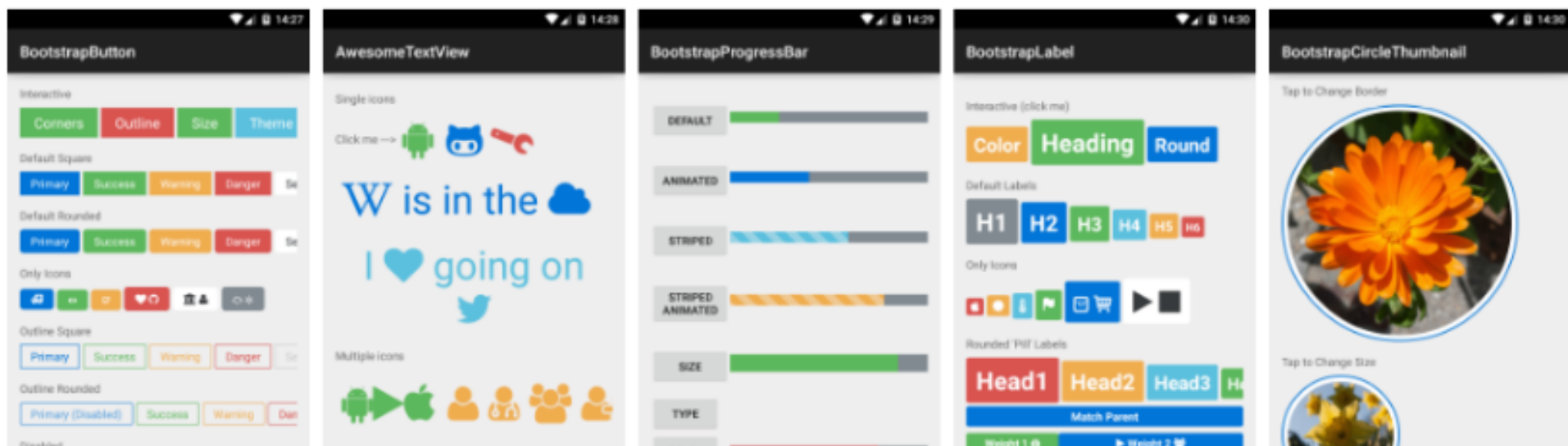
- Using `@OnEvent` annotations, you can easily attach methods to be event handlers for various widget events.
 - equivalent to calling `setOnEventListener` on a given view

```
1 @OnClick(R.id.mybutton)
2 public void handleClick(View view) {
3     Log.v("example", "Clicked the button!");
4 }
5
6 @OnLongClick(R.id.mytextview)
7 public void handleLongClick(View view) {
8     Log.v("example", "Long-clicked text view!");
9 }
```


Android-Bootstrap library

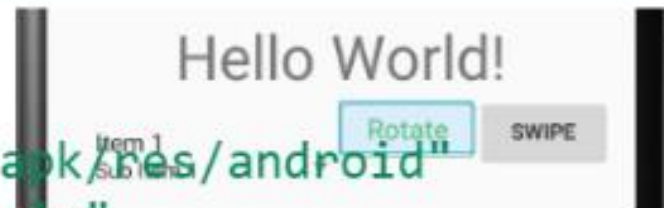
- **Android-Bootstrap** is a library that provides some good-looking customizable widgets not normally available in Android
 - <https://github.com/Bearded-Hen/Android-Bootstrap>
- To add it to your project:

```
1 // in build.gradle
2 dependencies {
3     compile 'com.beardedhen:androidbootstrap:2.3.1'
4 }
```



Using Android-Bootstrap widgets

```
1 <!-- res/layout/activity_main.xml -->
2 <LinearLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     xmlns:tools="http://schemas.android.com/tools"
5     xmlns:app="http://schemas.android.com/apk/res-auto" >
6     ...
7     <com.beardedhen.androidbootstrap.BootstrapButton
8         android:id="@+id/rotate"
9         android:text="Rotate"
10        app:bootstrapBrand="success"
11        app:bootstrapSize="lg"
12        app:buttonMode="regular"
13        app:showOutline="true"
14        app:roundedCorners="true"
15        android:layout_width="wrap_content"
16        android:layout_height="wrap_content" />
```




More about Android-Bootstrap

- Widget types available
 - AwesomeTextView, BootstrapButton, BootstrapButtonGroup, BootstrapCircleThumbnail, BootstrapEditText, BootstrapLabel, BootstrapProgressBar, BootstrapText, BootstrapThumbnail
- Library is not very well documented
 - assumes familiarity with web library Bootstrap, made by Twitter
 - need to dig around in [its source code](#), 'sample' app to see syntax

Branch: master ▾







Android-Bootstrap / sample / src / main / res / layout /

New fileFind fileHistory

 fractalwrench fix #131, update fontawesome to 4.5 (with delicious bluetooth icons)

Latest commit 154b823 on Nov 27, 2015

..

 activity_base.xml	add basic bootstrapprogressview	5 months ago
 activity_main.xml	fix button issues encountered on samsung	4 months ago
 example_awesome_text_view.xml	fix #131, update fontawesome to 4.5 (with delicious bluetooth icons)	2 months ago
 example_bootstrap_button.xml	update readme, add screenshots	4 months ago
 example_bootstrap_button_group.xml	implement bootstrap size in button, using scale factors	4 months ago
 example_bootstrap_circle_thumbnail.xml	implement bootstrapsizes for thumbnails	4 months ago