

# The Future is Here

It's Called "Android Studio"



<http://goo.gl/XISora>

# A Quick Survey



<http://goo.gl/BzRoFy>

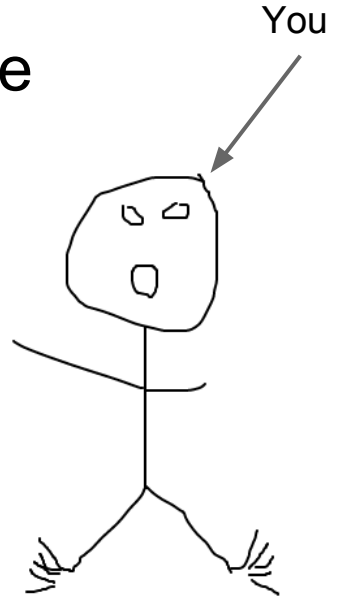
# About Me

- John Lombardo
- @johnnylambada
- john@lombardos.org
- Android for 5 years
- I'm was an Eclipse Snob
- I switched to IntelliJ.
- The keystrokes were different.
- I cried.
- I got over it.



# Android Studio

- Is much better / faster / cheaper than Eclipse
- Should be considered for all new projects
- Is a no-brainer once you've converted to gradle
- Will knock your socks off



# What does an IDE do?

- Makes programming fun.
- Keeps you out of the command line.
- Makes you more productive as a programmer
- Saves your brain cells for more important things.
- Like Beer.



# Android Studio vs Eclipse vs IntelliJ

I'll get straight to the point

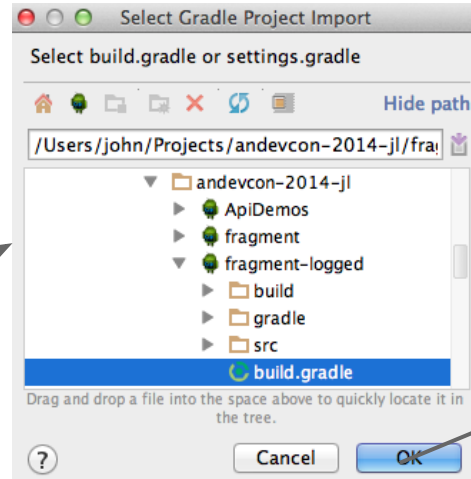
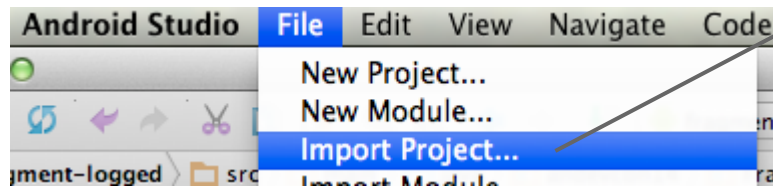
- The number #1 reason to switch to Android Studio is gradle.
- You say: Huh?
- I say: Really.
- Because: With Gradle+AS there is only ONE build system!

# One Build System -- Who Cares!

- You should
- Especially if you have a complex build.
- A half dozen jars
- A Library Project
- A git submodule
- Making that work in Eclipse is a pain.
- Making it work in IntelliJ is a pain too!
- And you have to get your Ant build working.
- Every time someone adds a new component your IDE build must change.

# One Build System - <3

- Some poor schmuck has to understand gradle and how the build actually works.
- No one else on the team does!
- File->Import Project->
- Find the build.gradle file
- That's it!





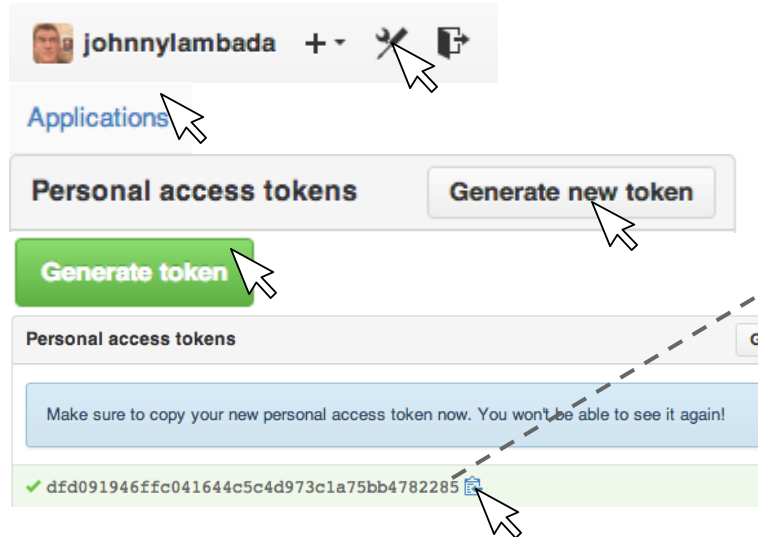
# Ok, that was cool

- All I have to do is File->Import my build.gradle file.
- That's nice, but I'm lazier than that.
- How did the build.gradle file get on my machine to begin with?
- Wait for it.
- Direct integration with GitHub, that's how...

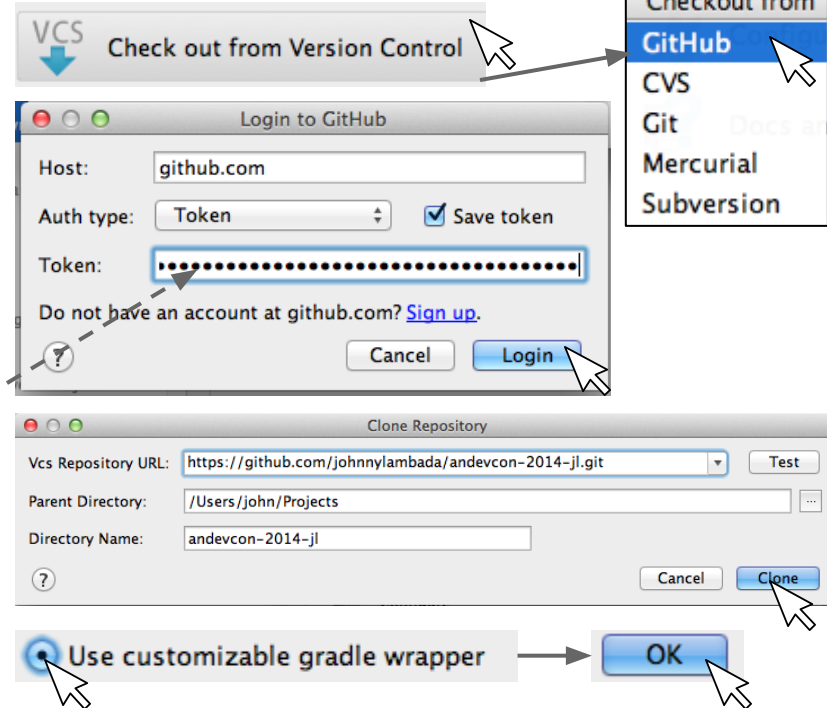
# Github Integration



- github integration is baked right into AS
- First, setup on github:



- In Android Studio:



# Github Integration - fix the command line

- You'll end up with a https based git repository.
- Use 'git remote set-url' to fix
- Great if you're only using Android Studio, not so much for command line

```
git remote set-url origin git@github.com:johnnylambada/andevcon-2014-jl.git
```

# I'm convinced -- let's install Android Studio

Three things to install

- Android Studio
- Android SDK
- Gradle



# Android Studio Tooling -- May 2014

Android Studio

<http://tools.android.com/download/studio/canary/latest>

Android SDK

<https://developer.android.com/sdk/index.html>

Gradle

brew install gradle

or

<http://www.gradle.org/downloads>

## ^ USE AN EXISTING IDE

If you already have an IDE you want to use for Android app development, setting up a new SDK requires that you download the SDK Tools, then select additional Android SDK packages to install (such as the Android platform and system image). If you'll be using an existing version of Eclipse, then you can add the ADT plugin to it.

[Download the SDK Tools for Mac](#)

# Android SDK Setup

Setting up an Android Development Environment takes a bit of time with our friend the command line.

```
cd ~  
unzip ~/Downloads/android-sdk_r22.6.1*zip # it creates android-sdk-macosx  
export ANDROID_HOME=~/android-sdk-macosx  
export PATH=$ANDROID_HOME/tools:$PATH  
export PATH=$ANDROID_HOME/platform-tools:$PATH  
  
# you might want to add the above to ~/.bash_profile  
  
android sdk # install the SDKs you want -- API 19  
android avd # create an Android emulator -- the Intel version is fastest
```

# **Android Studio Awesomeness**

# Android Studio Searching

- Paste this on your wall

## No files are open

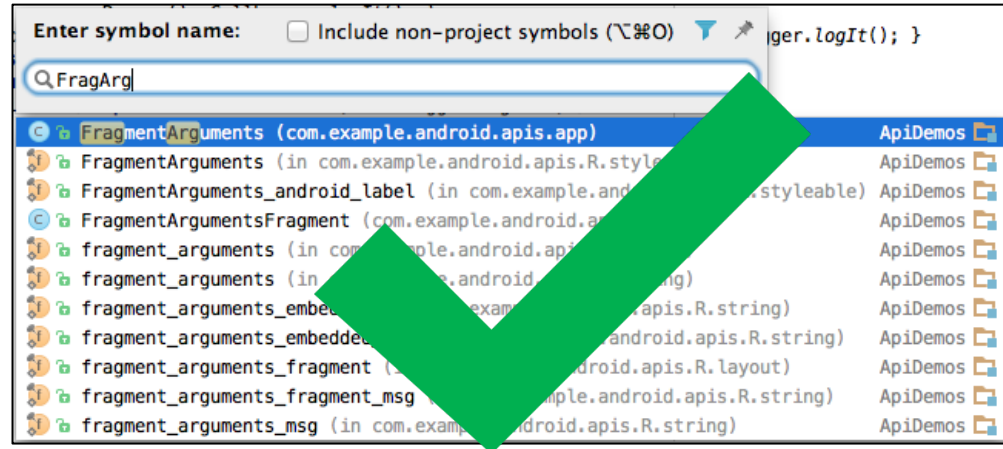
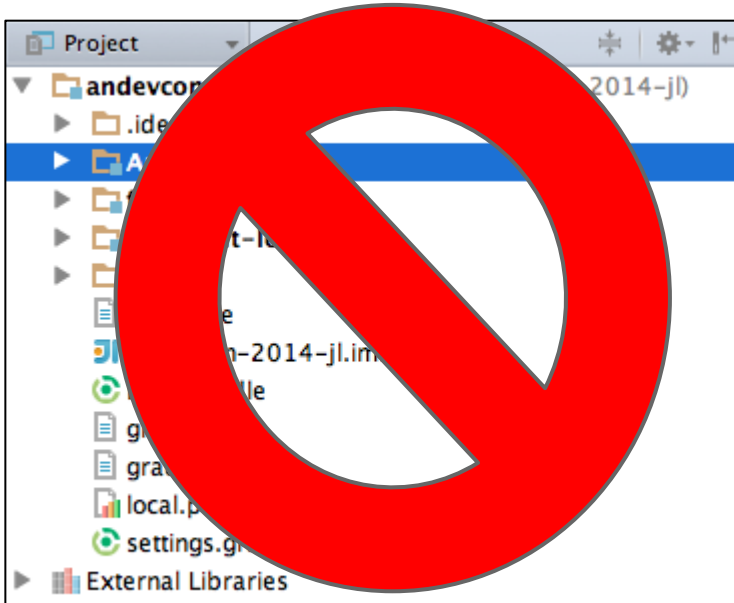
---

- Search Everywhere with Double ⇧
- Open a file by name with ⇧⌘O
- Open Recent Files with ⌘E
- Open Navigation Bar with ⌘↑
- Drag and Drop file(s) here from Finder




# Don't use the Project Hierarchy

- If you do, you're doing it wrong
- Use  instead!



# Browsing code with ⌘+click & ⇧⌘←

- Each identifier is like a link in your browser!
- Instead of just clicking, you ⌘+click
- Instead of a back button you type ⇧⌘←
- It's that easy.
- And it works with XML identifiers!



AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.devcon14.FragmentLogged"
    android:versionCode="1"
    android:versionName="1.0">
    <uses-sdk
        android:minSdkVersion="13"
        android:targetSdkVersion="19"
    />
    <application android:label="Fragments Logged" android:icon="@drawable/ic_launcher">
        <activity android:name=".MenuActivity"
            android:label="Fragments Logged">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".StaticFragmentActivity"/>
        <activity android:name=".DynamicFragmentActivity"/>
    </application>
</manifest>
```

A mouse cursor with a ⌘ key icon is clicking on the `DynamicFragmentActivity` identifier in the XML file.



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

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <FrameLayout
            android:id="@+id/frame_layout"
            android:layout_width="match_parent"
            android:layout_height="match_parent">

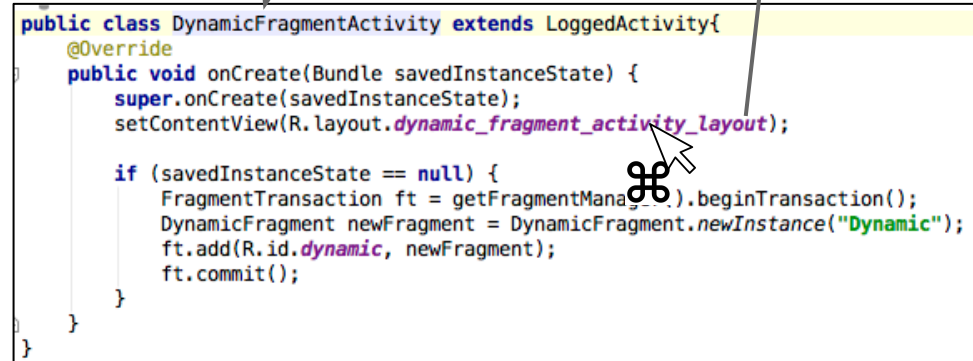
            <include android:name="@layout/content_main"
                android:layout_width="match_parent"
                android:layout_height="match_parent" />

        </FrameLayout>

    </LinearLayout>

</LinearLayout>
```

A mouse cursor with a ⇧⌘ key icon is clicking on the `dynamic_fragment_activity_layout` identifier in the XML file.



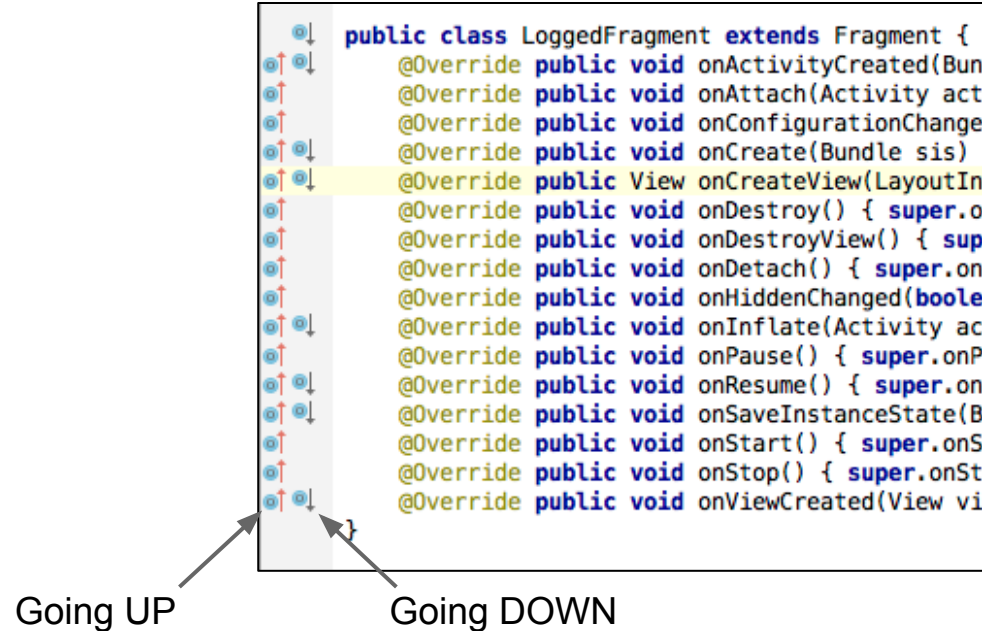
```
public class DynamicFragmentActivity extends LoggedActivity{
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.dynamic_fragment_activity_layout);

        if (savedInstanceState == null) {
            FragmentTransaction ft = getSupportFragmentManager().beginTransaction();
            DynamicFragment newFragment = DynamicFragment.newInstance("Dynamic");
            ft.add(R.id.dynamic, newFragment);
            ft.commit();
        }
    }
}
```

A mouse cursor with a ⇧⌘ key icon is clicking on the `dynamic_fragment_activity_layout` identifier in the Java code.

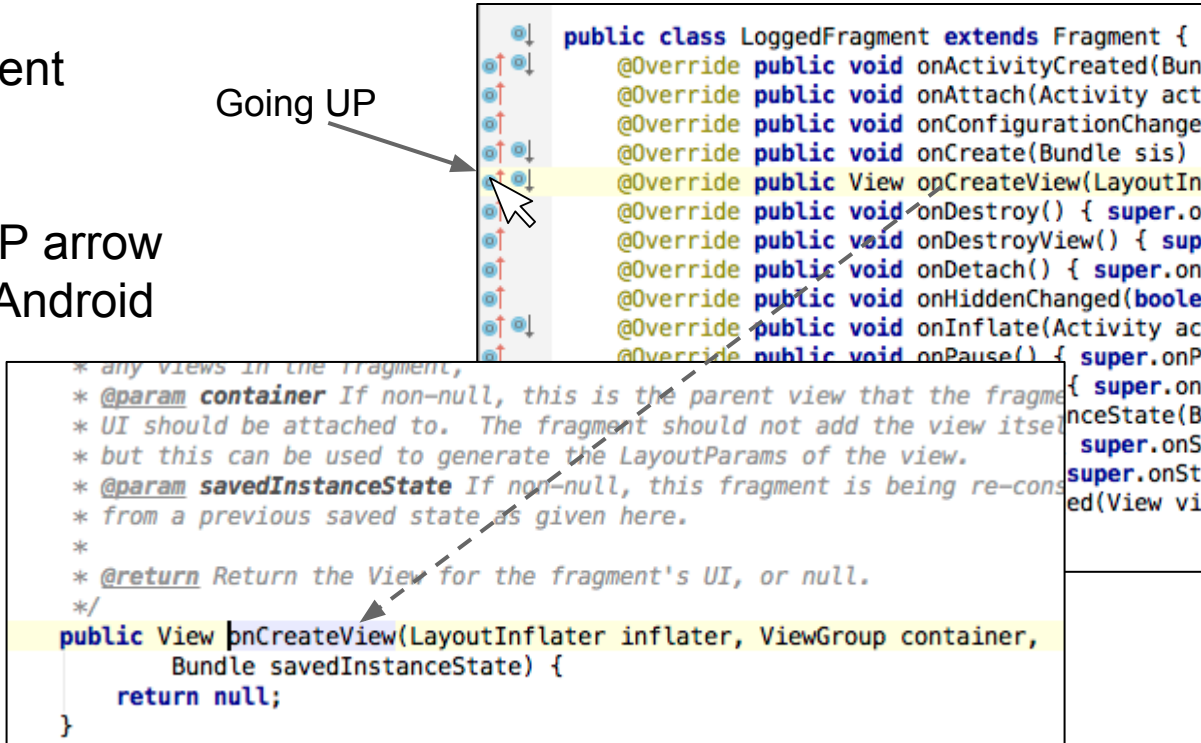
# Browse Up & Down the Class Hierarchy

- Click on the red up arrows to view the same function in the parent class.
- Click on the black down arrows to view the same function in the child class. If there's more than one you'll get a menu



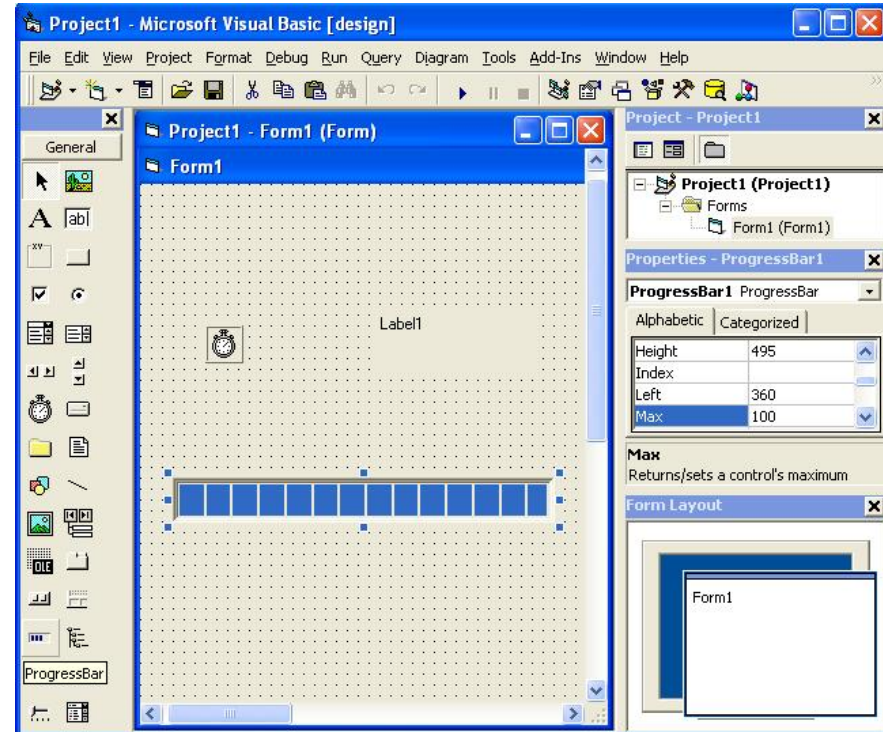
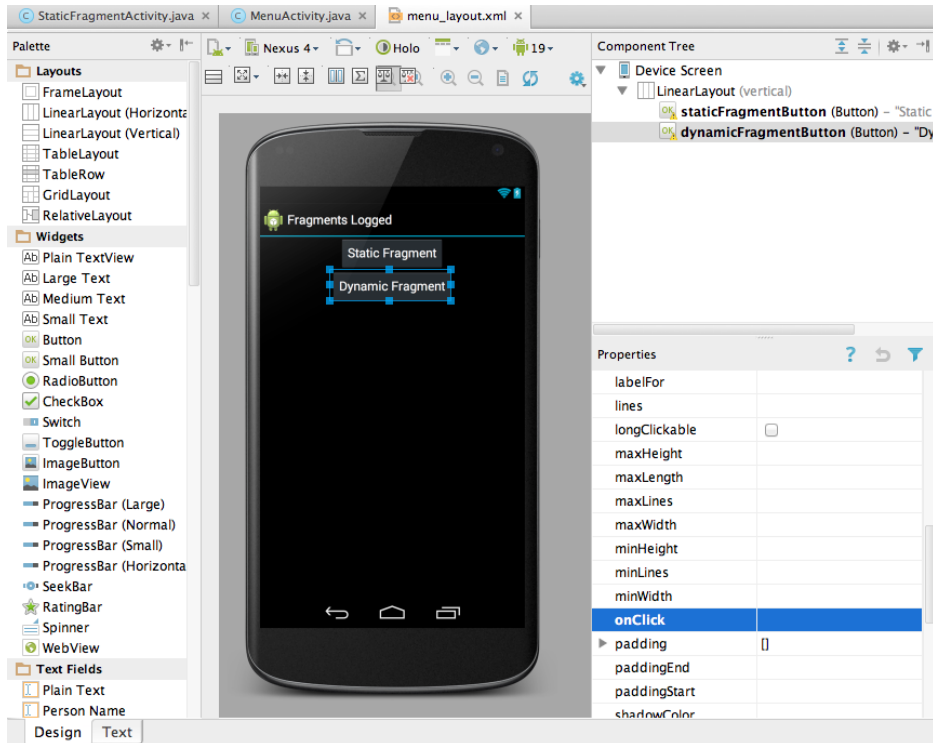
# Browse Right into the Android Source Code

- Note that LoggedFragment extends Fragment
- So clicking on the red UP arrow should bring you to the Android source code.
- And indeed it does!
- A handy way to look up the documentation!



# Drag and Drop controls with properties

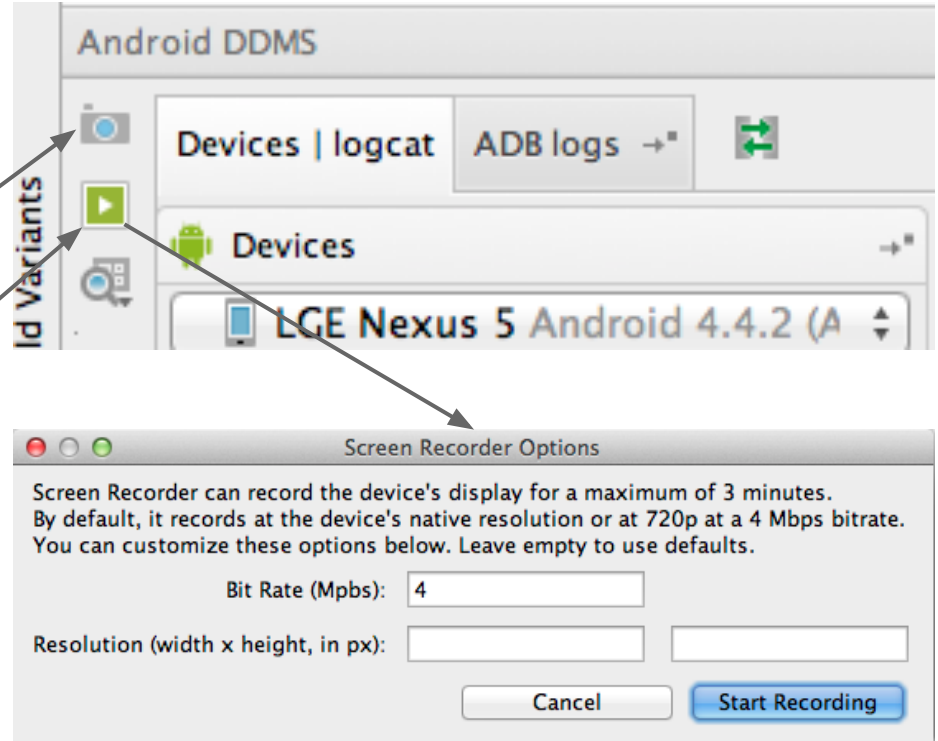
- Coincidence? I think not.



# Screen Capture + Record

Integrated into Android Studio:

- Screen Capture
- Video Capture!





# Hmm, It seems I have a problem

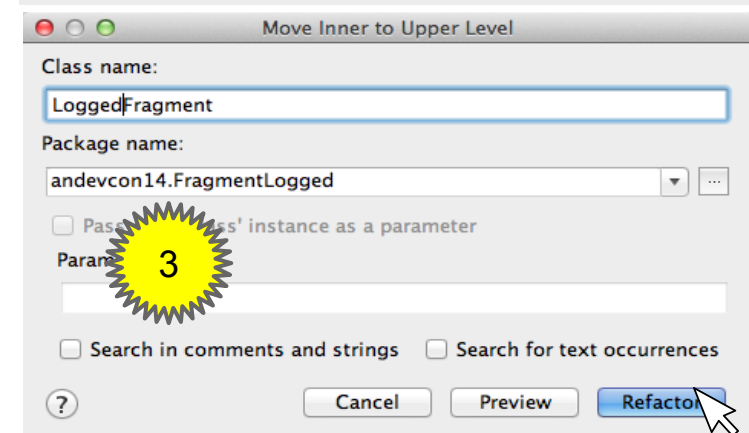
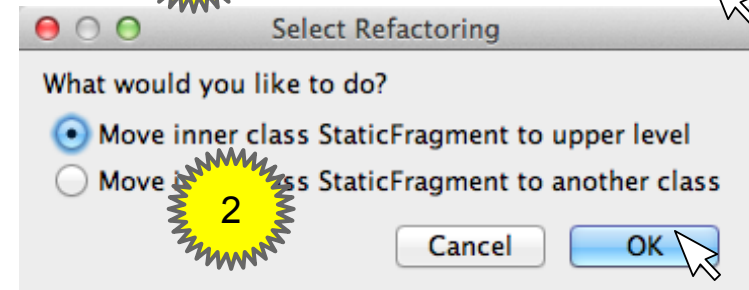
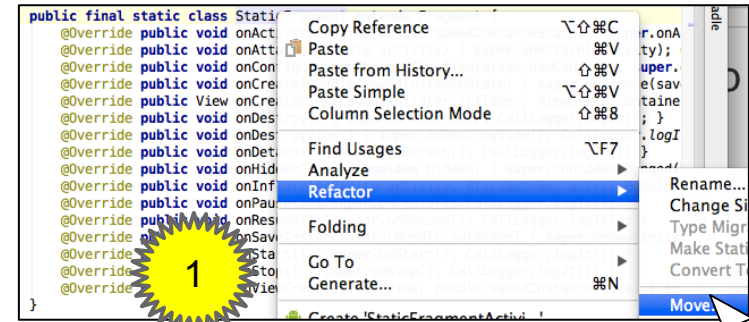
- StaticFragment is an inner class, making reuse impossible.
- Time to refactor!
- Let's make the IDE do it

```
public class StaticFragmentActivity extends Activity {
    @Override public void onCreate(Bundle savedInstanceState) {
    @Override public void onDestroy() { super.onDestroy(); CallLo
    @Override public void onNewIntent(Intent intent) { super.onNe
    @Override public void onPause() { super.onPause(); CallLogger
    @Override public void onPostCreate(Bundle savedInstanceState)
    @Override public void onPostResume() { super.onPostResume();
    @Override public void onRestart() { super.onRestart(); CallLo
    @Override public void onRestoreInstanceState(Bundle savedInstanceState)
    @Override public void onResume() { super.onResume(); CallLogg
    @Override public void onSaveInstanceState(Bundle outState) {
    @Override public void onStart() { super.onStart(); CallLogger
    @Override public void onStop() { super.onStop(); CallLogger.f
    @Override public void onUserLeaveHint() { super.onUserLeaveHi

    public final static class StaticFragment extends Fragment {
        @Override public void onActivityCreated(Bundle savedInstanceState)
        @Override public void onAttach(Activity activity) { super
        @Override public void onConfigurationChanged(Configuration)
        @Override public void onCreate(Bundle savedInstanceState)
        @Override public View onCreateView(LayoutInflater inflater
        @Override public void onDestroy() { super.onDestroy(); Ca
        @Override public void onDestroyView() { super.onDestroyVi
        @Override public void onDetach() { super.onDetach(); Call
        @Override public void onHiddenChanged(boolean hidden) {
        @Override public void onInflate(Activity activity, Attrib
        @Override public void onPause() { super.onPause(); CallLo
        @Override public void onResume() { super.onResume(); Call
        @Override public void onSaveInstanceState(Bundle outState)
        @Override public void onStart() { super.onStart(); CallLo
        @Override public void onStop() { super.onStop(); CallLogg
        @Override public void onViewCreated(View view, Bundle sav
    }
}
```

# Inner Class to Upper Level

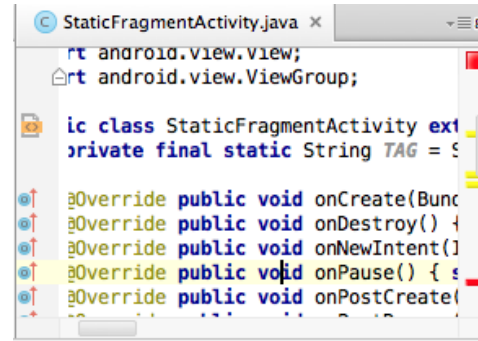
1. Refactor -> Move...
2. Move inner class to upper level
3. Give the new class a name and click Refactor



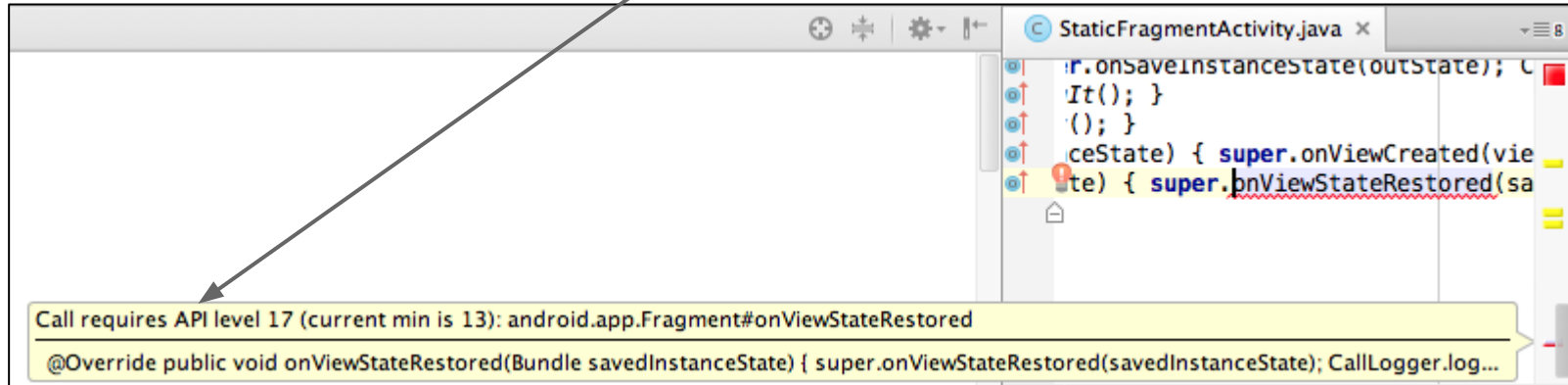


# AS says: You're using the wrong API!

- I'm minding my own business when this nasty red mark shows up in my source code. Yuck.
- The top one shows the # of warnings and errors in the code.
- The bottom one shows the error



Nasty red mark



# Diff your code



/Users/john/Projects/andevcon-2014-jl/fragment-logged/src/main/java/andevcon14/FragmentLogged/StaticFragmentActivity.java

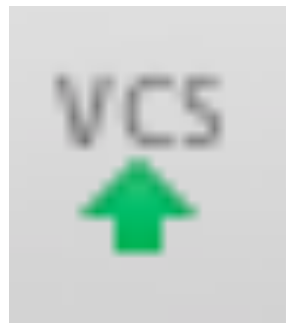
Ignore whitespace:  Highlight:

523c31dc2e0eccb3749f8580c4f54c26445ceeab (Read-only)

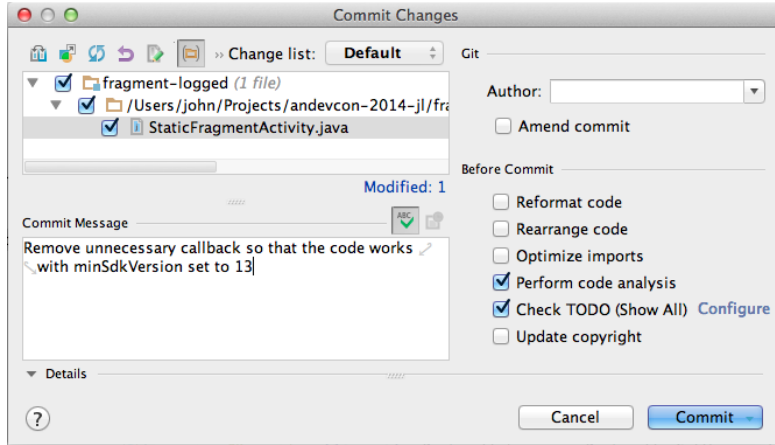
|   | Local   |
|---|---|
| @Override public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) { | @Override public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) { |
| super.onCreate(savedInstanceState);   | super.onCreate(savedInstanceState);   |
| super.onDestroy();  | super.onDestroy();  |
| super.onDestroyView();  | super.onDestroyView();  |
| super.onDetach();   | super.onDetach();   |
| onHiddenChanged(boolean hidden) {   | onHiddenChanged(boolean hidden) {   |
| onInflate(Activity activity, ViewGroup root, boolean attachToRoot) {  | onInflate(Activity activity, ViewGroup root, boolean attachToRoot) {  |
| onPause() {   | onPause() {   |
| onResume() {  | onResume() {  |
| onSaveInstanceState(Bundle savedInstanceState) {  | onSaveInstanceState(Bundle savedInstanceState) {  |
| onStart() {   | onStart() {   |
| onStop() {  | onStop() {  |
| onViewCreated(View view, Bundle savedInstanceState) {   | onViewCreated(View view, Bundle savedInstanceState) {   |
| onViewStateRestored(Bundle savedInstanceState) {  | onViewStateRestored(Bundle savedInstanceState) {  |
| }   | }   |

1 difference Deleted Changed Inserted

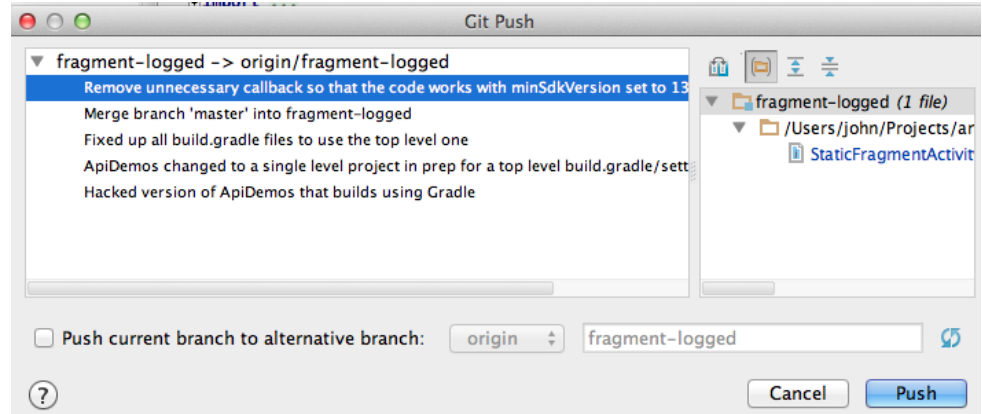
# Commit and Push your changes



- Push



- Commit



# Find bugs before you build

- In the first box, pattern is just a string
- In the second box, I've added code that uses the pattern variable as a regular expression pattern. Notice that there is now an error marker.
- Hover over the error marker for a full description of the error.

```
private String getCustomerPhoneNumber(){
    String description;
    if (project!=null && (description=project.getDescription())!=null){
        String pattern = "tel:([\\d\\s-()]+)*";
    }
    return null;
}
```

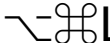

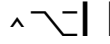
```
private String getCustomerPhoneNumber(){
    String description;
    if (project!=null && (description=project.getDescription())!=null){
        String pattern = "tel:([\\d\\s-()]+)*";
        String phone = description.replaceAll(pattern,"$1");
        return phone;
    }
    return null;
}
```

```
private String getCustomerPhoneNumber(){
    String description;
    if (project!=null && (description=project.getDescription())!=null){
        String pattern = "tel:([\\d\\s-()]+)*";
        String phone = description.replaceAll(pattern,"$1");
        return phone;
    }
    return null;
}
```





Character class (e.g. '\\w') may not be used inside character range

# Android Studio can reformat your code

## Obvious

-  L Reformat selected/file/directory code
-  O Optimize Imports
-  I Indent selected lines properly


## Not so obvious

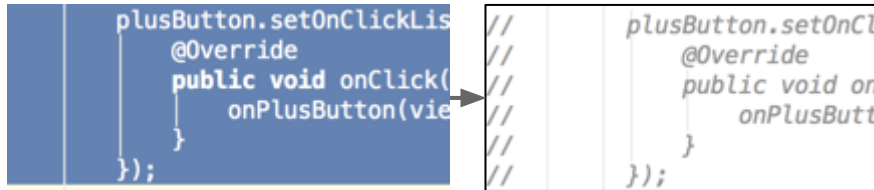
-   when cursor is in a function name, moves the entire function above the previous one. When it's on a statement, move it instead.
-   move it below the next one

# Comment out code

## Line comments (multiple //)

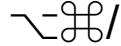
- Select the code

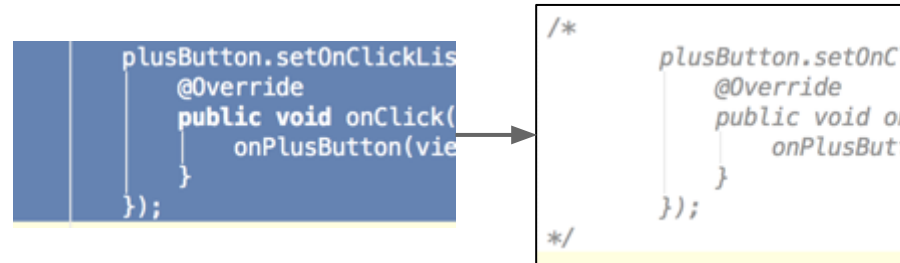
- 



## Block comment (/\* ... \*/)

- Select the code

- 



# Code folding

Wikipedia says:

[Code folding](#) is a feature of some text editors, source code editors and IDEs that allows the user to selectively hide and display – "fold" – sections of a currently-edited file as a part of routine edit operations.

Android Studio has several types:

- Automatic: Code that is automatically folded when you open the class
- Manual: Code that you can fold by pressing a +/- button next to the line in the editor
- Defined: regions can be defined in comments

# Code folding: Automatic

- Anonymous inner classes can be collapsed to just one line.
- One liner functions can be collapsed to just one editor line.

```
minusButton.setOnClickListener((view) -> {  
    ((MinusPlusButtonInterface) getActivity()).onMinusButton(view);  
});
```



```
minusButton.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        ((MinusPlusButtonInterface) getActivity()).onMinusButton(view);  
    }  
});
```

```
public IntentFilter getReportIntentFilter() { return new IntentFilter
```



```
public IntentFilter getReportIntentFilter(){  
    return new IntentFilter(ACTION_REPORT);  
}
```





# Code Folding: Manual

- Click on the  to collapse a whole function.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.local_broadcast);
    Button minusButton = (Button) findViewById(R.id.minusButton);
    Button plusButton = (Button) findViewById(R.id.plusButton);
    final TextView count = (TextView) findViewById(R.id.count);

    if (savedInstanceState != null) {
        counter = (Counter) savedInstanceState.getSerializable(COUNTER_KEY);
    } else {
        counter = new Counter();
        LocalBroadcastFragment fragment = new LocalBroadcastFragment();
        getSupportFragmentManager()
            .beginTransaction()
            .add(R.id.right, fragment)
            .commit();
    }
}
```

-   - to collapse every function in the file. This lets you see the structure of a 100+ line Class at a glance.

```
@Override
protected void onCreate(Bundle savedInstanceState) {...}
```

-   + to re-expand

```
1 package andevcon14.FragmentCommsSupport.Types.LocalBroadcast;
2
3 import ...
4
14 public class LocalBroadcastActivity extends FragmentActivity {
15     private static final String COUNTER_KEY = "COUNTER_KEY";
16     private Counter counter = null;
17     private BroadcastReceiver countReceiver, updateReceiver;
18
19     @Override protected void onCreate(Bundle savedInstanceState) {...}
20
73     @Override protected void onSaveInstanceState(Bundle outState) {...}
77
77     @Override protected void onPause() {...}
84
84     @Override protected void onResume() {...}
95
95     @Override protected void onPostResume() {...}
100 }
```


# Code Folding: Defined regions

- Start the “Fred” region:

// region Fred

- End the region:

// endregion

-  to collapse the defined regions

```
11 public class ObserverPatternActivity extends Fragment
12 implements Counter.Observer, MinusPlusButtonInterface {
13     private static final String COUNTER_KEY = "COUNTER_KEY";
14     private Button minusButton, plusButton;
15     private TextView count;
16     private Counter counter = null;
17
18     // region extends FragmentActivity
19     @Override protected void onCreate(Bundle savedInstanceState) {
20         // ...
21     }
22     @Override protected void onSaveInstanceState(Bundle savedInstanceState) {
23         // ...
24     }
25     @Override protected void onPause() {
26         // ...
27     }
28     @Override protected void onResume() {
29         // ...
30     }
31     // endregion
32     // region implements Counter.Observer
33     @Override public void onCount(int count) { this.count += count; }
34     // endregion
35     // region implements MinusPlusButtonInterface
36     @Override public void onMinusButton(View view) {
37         // ...
38     }
39     @Override public void onPlusButton(View view) {
40         // ...
41     }
42     // endregion
43 }
```



```
11 public class ObserverPatternActivity extends Fragment
12 implements Counter.Observer, MinusPlusButtonInterface {
13     private static final String COUNTER_KEY = "COUNTER_KEY";
14     private Button minusButton, plusButton;
15     private TextView count;
16     private Counter counter = null;
17
18     extends FragmentActivity
19     implements Counter.Observer
20     implements MinusPlusButtonInterface
21 }
```

# Code Generation

Let's build a new project from scratch using the IDE.

We'll call it Coke Zero in honor of the Coke Zero on my desk.

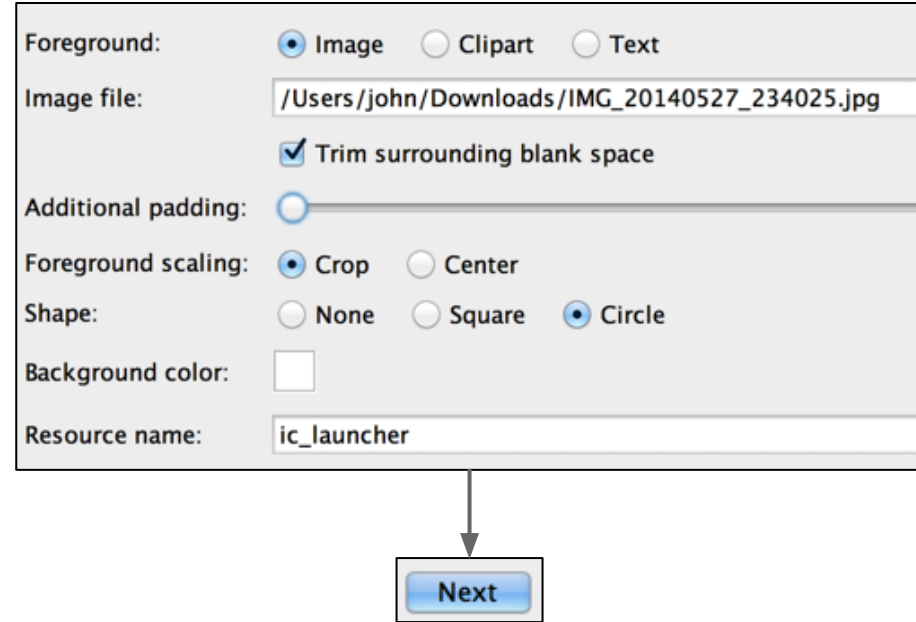
The diagram illustrates the 'New Project...' dialog box in an IDE. It features the following fields and options:

- Application name:** Coke Zero
- Module name:** app
- Package name:** sigseg.cokezero.app
- Project location:** /Users/john/Projects/CokeZero
- Minimum required SDK:** API 8: Android 2.2 (Froyo)
- Target SDK:** API 19: Android 4.4 (KitKat)
- Compile with:** API 19: Android 4.4 (KitKat)
- Theme:** Holo Light with Dark Action Bar
- ☒ Create custom launcher icon
- ☒ Create activity
- ☐ Mark this project as a library
- Support Mode:**
  - ☒ GridLayout
  - ☒ Fragments
  - ☒ Navigation Drawer
  - ☒ Action Bar

A 'Next' button is located at the bottom right of the dialog box.

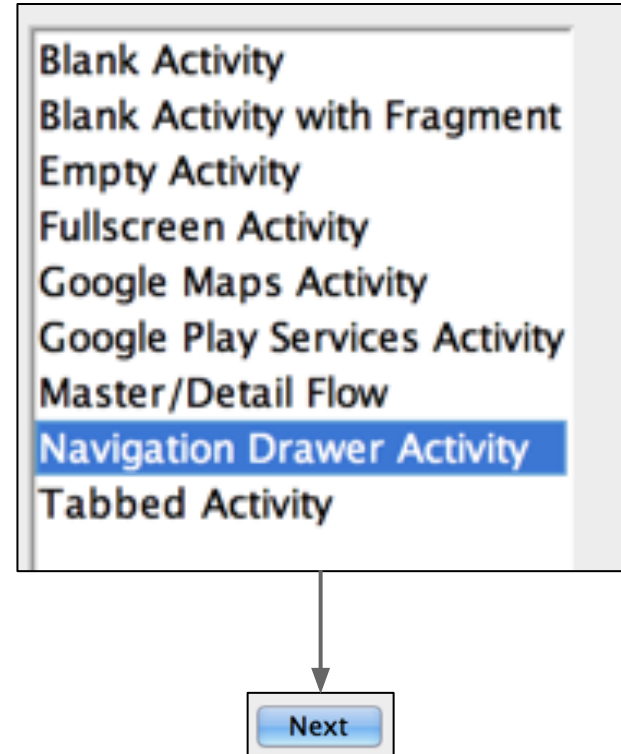
# Generate the Icon

- The green android Icon is so five-years-ago.
- It's simple to create your own.
- You can also use the awesome [Android Asset Studio](#)



# Generate a startup activity

- Navigation Drawers are the new hotness.

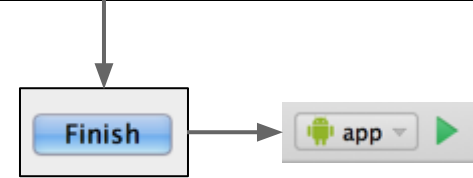


# Name your initial Activity

- You'll get an Activity hosting two fragments:
  - The content fragment
  - The nav drawer fragment

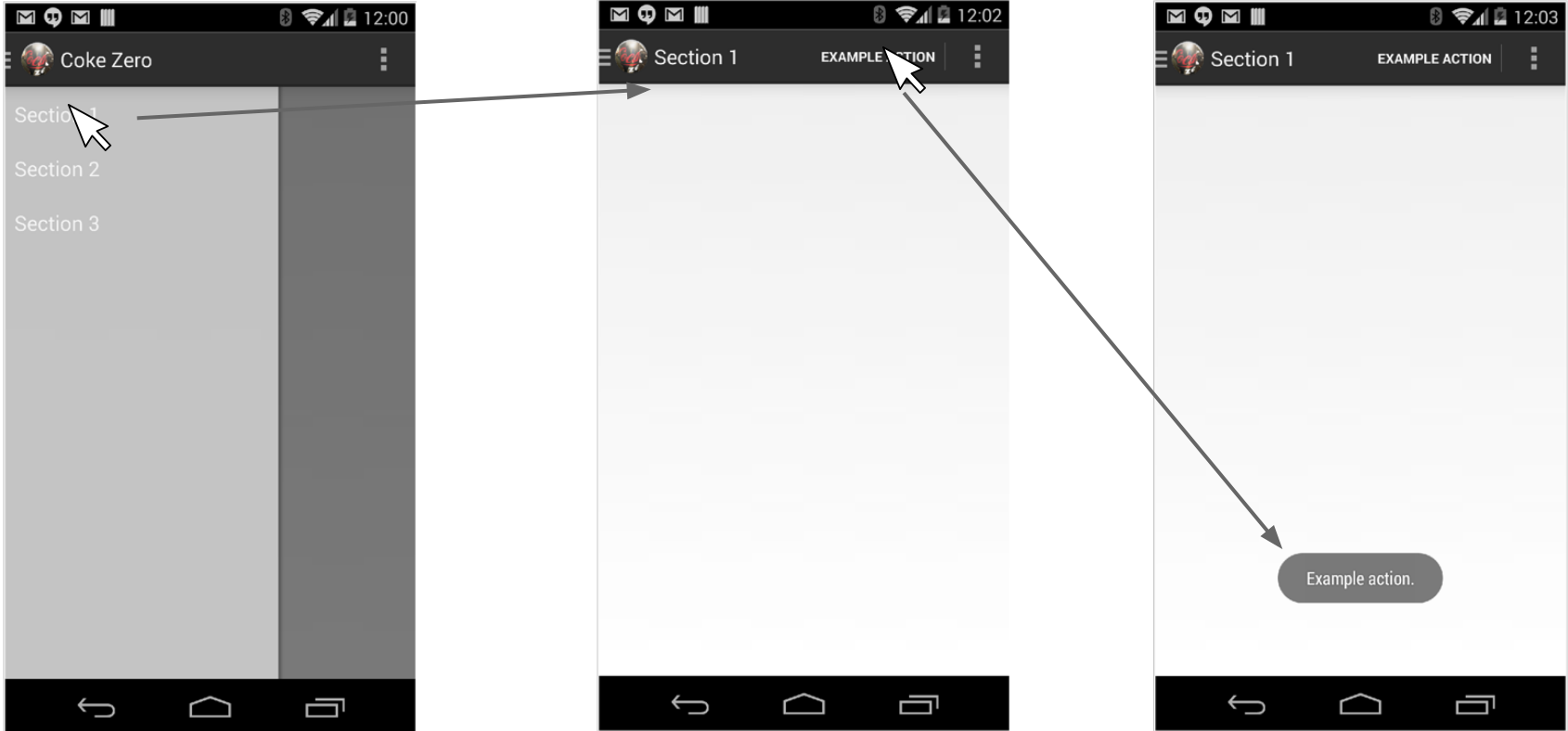
|                                 |                  |
|---------------------------------|------------------|
| Activity Name                   | MainActivity     |
| Layout Name                     | activity_main    |
| Fragment Layout Name            | fragment_content |
| Navigation Drawer Fragment Name | fragment_drawer  |

- Press Finish to generate the app



- In true Google fashion, it'll complain that something went dreadfully wrong, but if you give it time it usually sorts itself out and you can press the green arrow to run it.

# All without a line of code



# Now we hack on the code...

We'll start by changing the navigation drawer to show us various types of Coke Zero

Then, who knows...





# Questions