Android Dev Kotlin

CS 402: Mobile Development

Homework Assignment

During the semester, you will conceive, design and create an Android mobile app that you will release. For this homework assignment, you will come up with five ideas for a mobile app. You will develop this app throughout the semester and it'll become your final project. You might pick an idea that solves a problem you have, or has content about a subject you're passionate about, or takes an available web-based API and does something useful with it.

Emphasize why your app is best on a mobile device vs. a web app.

Homework Assignment

Idea:
Create an app so users can find a place to eat on campus based on location, interests, and who you're with.

Competition:
BSU Food Finder, Bronco Eats, Boise Foodie

Audience:
BSU Student Body or anyone visiting the campus.

Technologies:
GPS to find the user's location. MapView to show map of restaurants. Camera to check into restaurant. XML parser or database to read input. Network to pull in description data. Accelerometer for shake feature that finds a random restaurant.

Homework Assignment

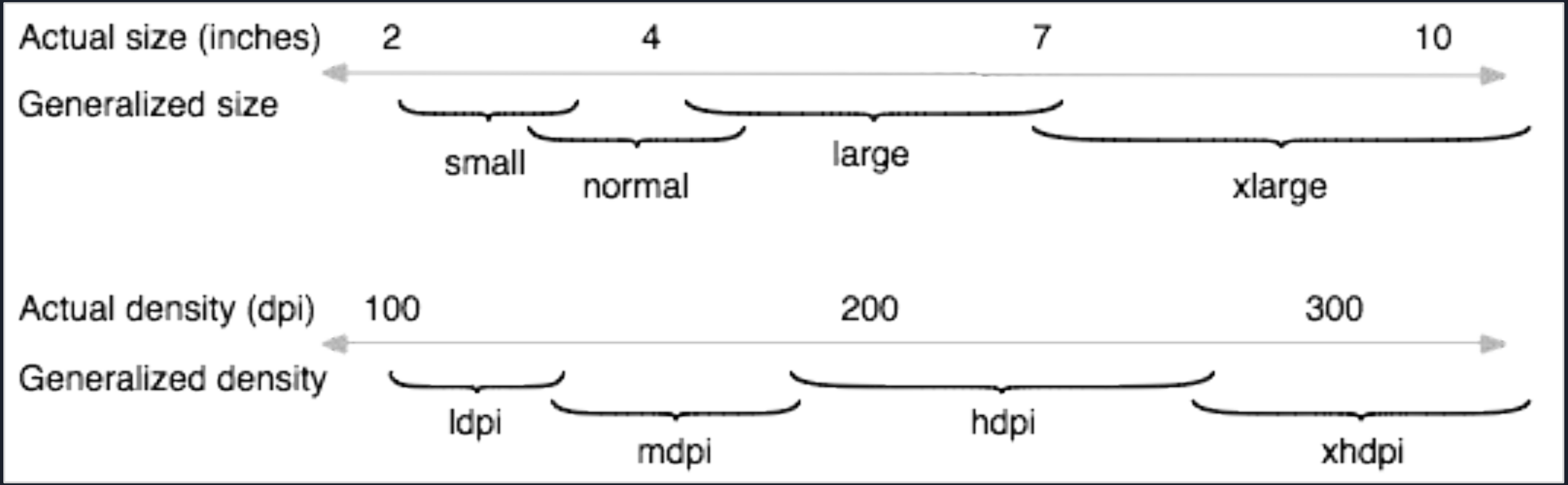
Suggested Price:

$1.99 - All competitor apps are $0.99 and we plan on having a substantial difference on quality, features and updates to the app which will make it worth the extra fee. I'm also getting a designer for a more professional look.

Android Project Structure

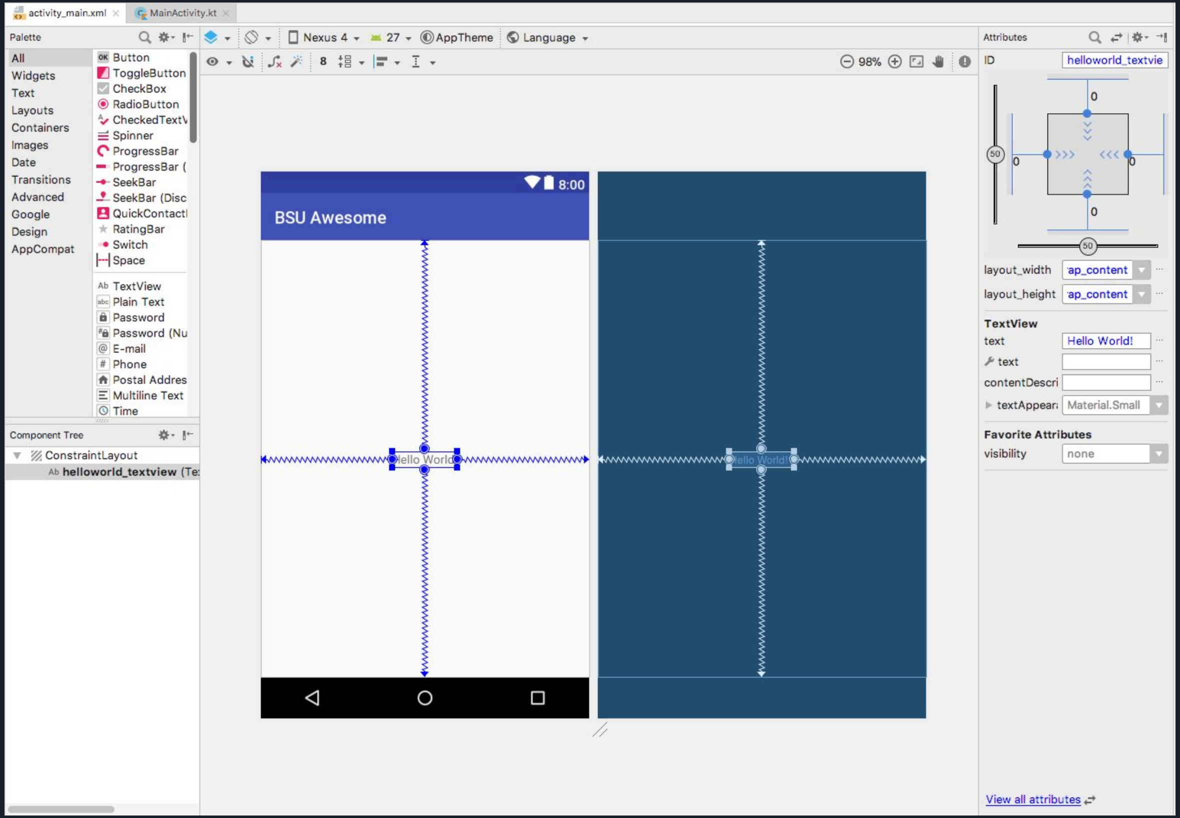
src: Source files for Java/Kotlin files
res: Resources for app
drawable: Images or other graphics
layout: Layout xml files
menu: Menus and menu items xml files
values: All dynamic values
strings: Words, paragraphs, values that are strings (urls, phone numbers, etc)
styles: Styles that affect your UI
dimens: Dimensional numbers (padding, margins, etc)

Kotlin Types



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

Android Studio Visual UI Editor



Android Components

Activities - User interface screens

Fragments - User interface component groups

Widgets - Display and user interactivity

Services - Background jobs

Broadcast Receivers - Messaging system

Android Activities

Subclass of `android.app.Activity`or some version of it

Applications can have 0, 1 or many activities

Defines and manages the user interface

Basically, one activity will define each screen

Android Screen Support

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

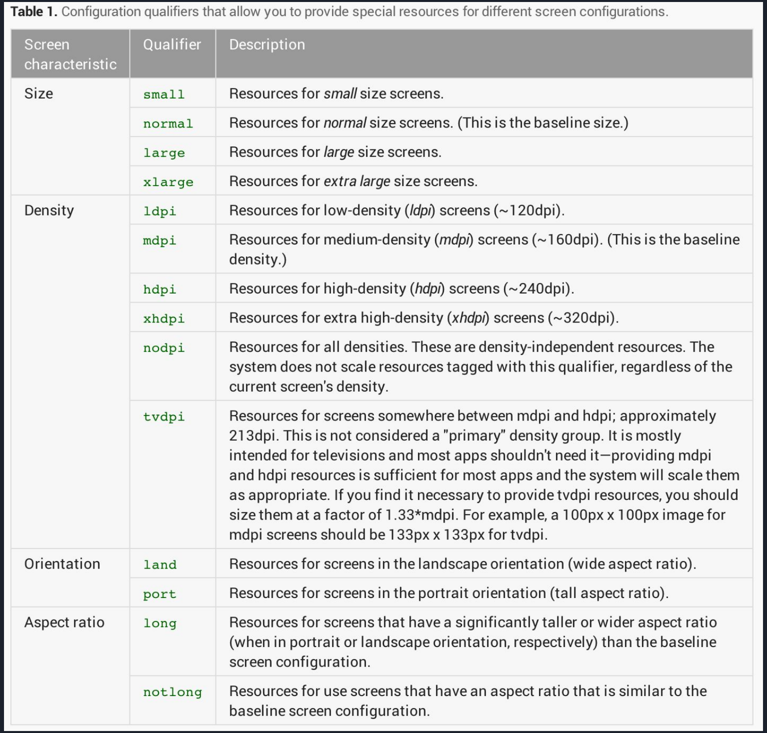
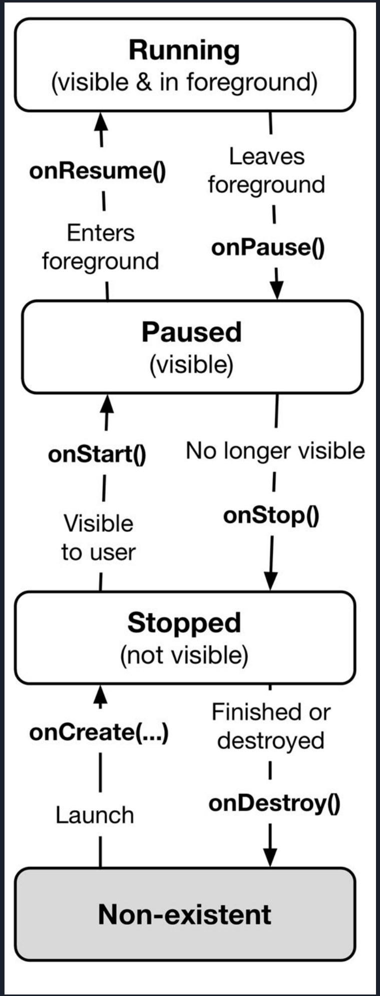


Table 1. Configuration qualifiers that allow you to provide special resources for different screen configurations.

Screen characteristic	Qualifier	Description
Size	`small`	Resources for *small* size screens.
	`normal`	Resources for *normal* size screens. (This is the baseline size.)
	`large`	Resources for *large* size screens.
	`xlarge`	Resources for *extra large* size screens.
Density	`ldpi`	Resources for low-density (*ldpi*) screens (~120dpi).
	`mdpi`	Resources for medium-density (*mdpi*) screens (~160dpi). (This is the baseline density.)
	`hdpi`	Resources for high-density (*hdpi*) screens (~240dpi).
	`xhdpi`	Resources for extra high-density (*xhdpi*) screens (~320dpi).
	`nodpi`	Resources for all densities. These are density-independent resources. The system does not scale resources tagged with this qualifier, regardless of the current screen's density.
	`tvdpi`	Resources for screens somewhere between mdpi and hdpi; approximately 213dpi. This is not considered a "primary" density group. It is mostly intended for televisions and most apps shouldn't need it—providing mdpi and hdpi resources is sufficient for most apps and the system will scale them as appropriate. If you find it necessary to provide tvdpi resources, you should size them at a factor of 1.33*mdpi. For example, a 100px x 100px image for mdpi screens should be 133px x 133px for tvdpi.
Orientation	`land`	Resources for screens in the landscape orientation (wide aspect ratio).
	`port`	Resources for screens in the portrait orientation (tall aspect ratio).
Aspect ratio	`long`	Resources for screens that have a significantly taller or wider aspect ratio (when in portrait or landscape orientation, respectively) than the baseline screen configuration.
	`notlong`	Resources for use screens that have an aspect ratio that is similar to the baseline screen configuration.

Android Widgets

- Running
- Paused
- Stopped
- Doesn't exist



Activities in Android

Controller code goes in Kotlin/Java class that subclasses Activity (ex: MainActivity.kt)

Layout goes in an XML file (ex: layout_mainactivity.xml)

Declare the activity's name (class namespace) in AndroidManifest.xml file. (Android Studio does this for you)

Activities in Android

Creates a subclass of Activity

Creates a layout_activityname.xml

Adds your activity to the AndroidManifest.xml file

```
<activity
android:name="com.example.myapplication.ActivityName"

    android:label="@string/title_for_activity" >

</activity>
```

Activity Death

On device rotation (strange but true)

Reclaiming memory and your activity is paused or stopped

If paused or stop, onSaveInstanceState() has already been called

Back button is pressed

System Reboot

Hasn't been used in a long time (usually a low memory state kills it).

Android Reference to Widgets

```
// Saving an value to the activity's bundle
@Override
public void onSaveInstanceState( Bundle savedInstanceState){
     super.onSaveInstanceState( savedInstanceState);
     savedInstanceState.putInt( "KEY", mSomeIntValue);
}

// Retrieving the value on creation
protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     if (savedInstanceState != null){
        mSomeIntValue = savedInstanceState.getInt( "KEY", 0);
     }
}
```

Saving Activity State

Getters and Putters only support primitive types out of the box

To save state of a class, implement Serializable() on the class

Then put it with: putSerializable(String key, Serializable value)

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

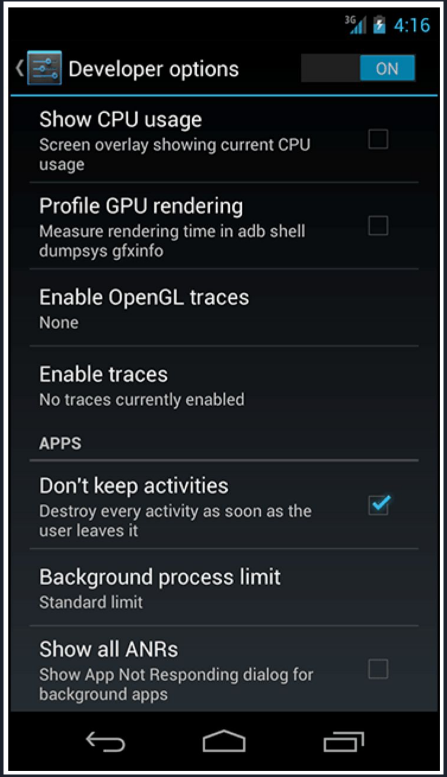
Testing Activity Death

On Hardware:

http://developer.android.com/tools/debugging/debugging-devtools.html

On the Emulator (AVD)

Built in - Settings -> Developer options


4:16

Developer options ON

Show CPU usage
Screen overlay showing current CPU usage ☐

Profile GPU rendering
Measure rendering time in adb shell dumpsys gfxinfo ☐

Enable OpenGL traces
None

Enable traces
No traces currently enabled

APPS

Don't keep activities
Destroy every activity as soon as the user leaves it ☑

Background process limit
Standard limit

Show all ANRs
Show App Not Responding dialog for background apps ☐

Homework 2
Due 1/30/2019 EoD

- PDF ONLY - Every other format gets a 0.
- Submit as Homework 2 in your GitLab repo
- 5 sections for EACH of the app ideas.

Homework Assignment

During the semester, you will conceive, design and create an Android mobile app that you will release. For this homework assignment, you will come up with five ideas for a mobile app. You will develop this app throughout the semester and it’ll become your final project. You might pick an idea that solves a problem you have, or has content about a subject you’re passionate about, or takes an available web-based API and does something useful with it.

Emphasize why your app is best on a mobile device vs. a web app.

Homework Assignment Example

Idea:
Create an app so users can find a place to eat on campus based on location, interests, and who you're with.

Competition:
BSU Food Finder, Bronco Eats, Boise Foodie

Audience:
BSU Student Body or anyone visiting the campus. Ages 16-30 & 40-60

Technologies:
GPS to find the user's location. MapView to show map of restaurants. Camera to check into restaurant. XML parser or database to read input. Network to pull in description data. Accelerometer for shake feature that finds a random restaurant.

Homework Assignment

Suggested Price:

$1.99 - All competitor apps are $0.99 and we plan on having a substantial difference on quality, features and updates to the app which will make it worth the extra fee. I'm also getting a designer for a more professional look.

Homework Grading

0 - No appropriate submission

1 - Not working and very little code

2 - Not working or very little code

3 - Requirements missing

4 - All requirements fully satisfied

5 - All requirements fully satisfied and added design and/or functionality

Homework Grading

Example:

Student satisfies all requirements and adds design elements: 4.2

Missing some features but adds design elements: 3.2

All requirements and added some features and design: 4.9