CS 399: Mobile Application Development
Summer 2020

Homework-03

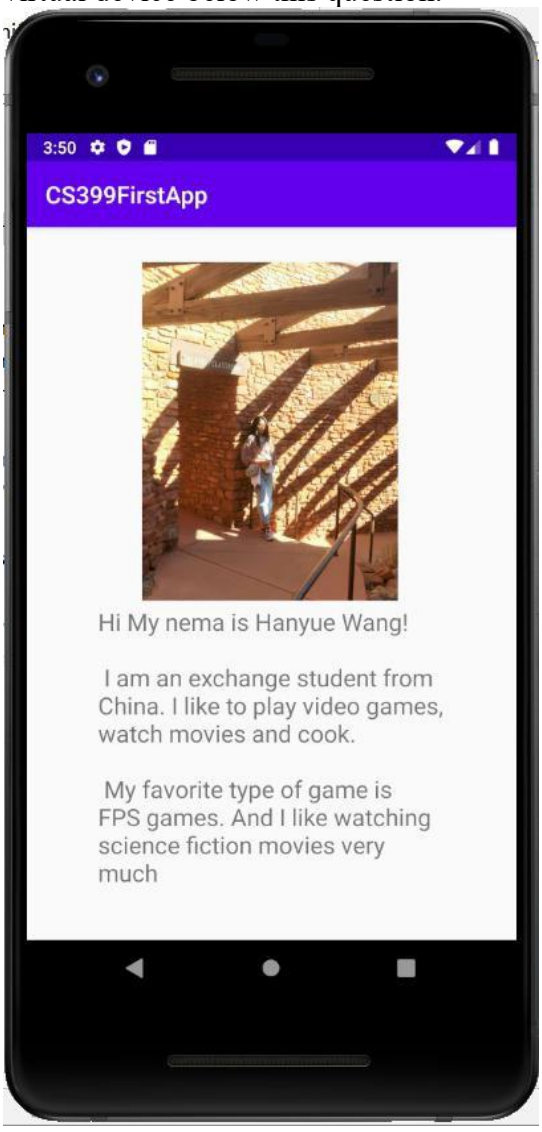
Due Wednesday June 10
30 Points Total

Hanyue Wang

CS-399(1204-2860) SPECIAL TOPICS 002 - MOBILE APPLICATION DEVELOPMENT

Questions:

[0 points] Question#1
Include a view of your Assignment 2 Profile application when it is run on the Pixel 2 virtual device below this question.



[3 points] Question#2

Search the Android online documentation for the *CoordinatorLayout* class. Capture a view of the 4 level class hierarchy including the *CoordinatorLayout* class and its three level superclass hierarchy. Paste the snapshot below.

Android Developers > Docs > Reference

CoordinatorLayout

```
public class CoordinatorLayout
extends ViewGroup implements NestedScrollingParent2, NestedScrollingParent3
```

java.lang.Object
  ↳ android.view.View
    ↳ android.view.ViewGroup
      ↳ androidx.coordinatorlayout.widget.CoordinatorLayout

ViewGroup

```
public abstract class ViewGroup
extends View implements ViewParent, ViewManager
```

java.lang.Object
  ↳ android.view.View
    ↳ android.view.ViewGroup

View

Added

Kc

```
public class View
extends Object implements Drawable.Callback, KeyEvent.Callback, AccessibilityEventSource
```

java.lang.Object
  ↳ android.view.View

Object

Added in API level 1

```
public class Object
```

java.lang.Object

Class `Object` is the root of the class hierarchy. Every class has `Object` as a superclass. All objects, including arrays, implement the methods of this class.

[4 points] Question#3
Review the code generated for your Profile application. Find all the View and ViewGroup subclasses used in your program. List names of the subclasses separately for each superclass below.
Answer:
In my code subclasses of **View**: ImageView
TextView

ViewGroup: ConstraintLayout

[3 points] Question#4
For each one of the classes listed for Question#3 list the number of the instances of the class generated in your program.
Answer:
View: ImageView number:1, TextView nember:1
ViewGroup: ConstraintLayout number: 1

[3 points] Question#5
For each one of the classes listed for Question#3 list if the class belongs to the Core Android SDK or to an Extended library. For each Extended library class include the package name prefix.
Answer:
Core Android SDK: ImageView, TextView
Extended library: androidx.constraintlayout.widget.ConstraintLayout

[4 points] Question#6
Search the Android online documentation for all the classes listed for Question#3. Capture a snapshot of the class definitions and the class hierarchy for these classes. Pay attention to the class name and package prefixes to make sure you pick the right class definitions. Include the snapshots of the class definitions below.

ImageView

Added in API level 1

Kotlin | **Java**

```
public class ImageView
extends View
```

java.lang.Object
↳ android.view.View
↳ android.widget.ImageView

Known direct subclasses
ImageButton, QuickContactBadge

Known indirect subclasses
ZoomButton

Displays image resources, for example `Bitmap` or `Drawable` resources. ImageView is also commonly used to apply tints to an image and handle image scaling.

TextView

Added in API level 1

Kotlin | **Java**

```
public class TextView
extends View implements ViewTreeObserver.OnPreDrawListener
```

java.lang.Object
  ↳ android.view.View
    ↳ android.widget.TextView

Known direct subclasses
Button, CheckedTextView, Chronometer, DigitalClock, EditText, TextClock

Known indirect subclasses
AutoCompleteTextView, CheckBox, CompoundButton, ExtractEditText, MultiAutoCompleteTextView, RadioButton, Switch, ToggleButton

---

A user interface element that displays text to the user. To provide user-editable text, see `EditText`.

ConstraintLayout

Kotlin | **Java**

```
public class ConstraintLayout
extends ViewGroup
```

java.lang.Object
  ↳ ViewGroup
    ↳ androidx.constraintlayout.widget.ConstraintLayout

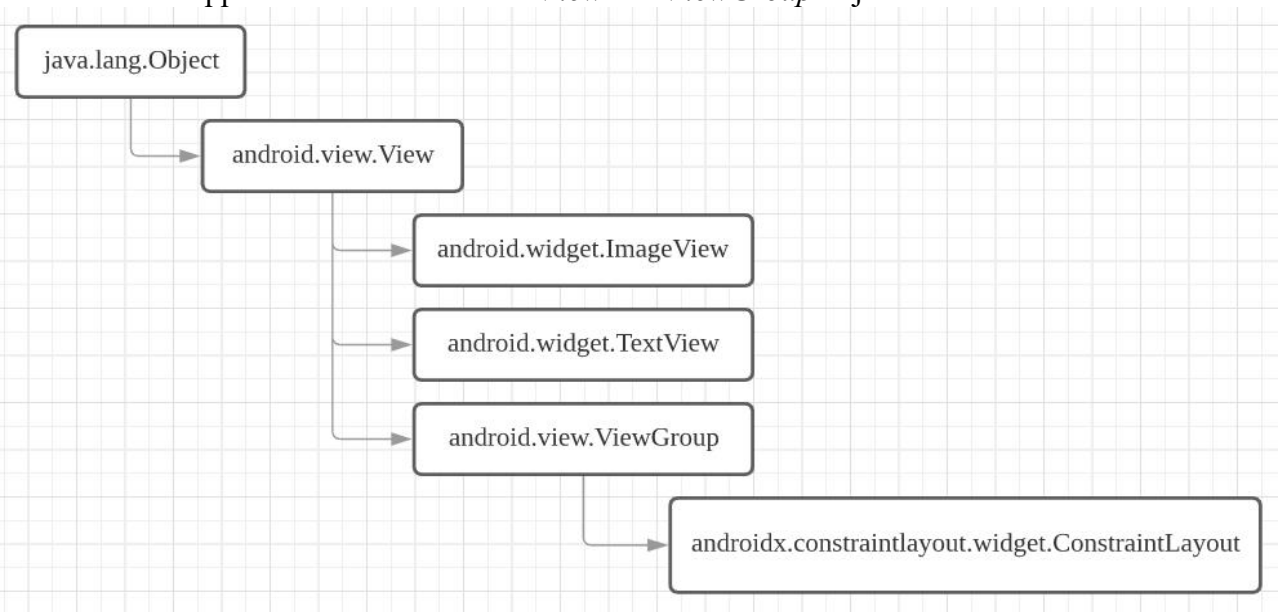
Known direct subclasses
MotionLayout

A `ConstraintLayout` is a `android.view.ViewGroup` which allows you to position and size widgets in a flexible way.

Note: `ConstraintLayout` is available as a support library that you can use on Android systems starting with API level 9 (Gingerbread). As such, we are planning on enriching its API and capabilities over time. This documentation will reflect those changes.

[6 points] Question#7

Draw a hierarchical instance diagram of the *ConstraintLayout* object created in your Profile application. Include all the *View* and *ViewGroup* objects.



[7 points] Question#8

Draw a hierarchical class diagram showing the *inheritance* (class/subclass) relationships among all the classes you have viewed in this assignment (Question#2 to Question#6). Start the root of your class hierarchy at java.lang.Object class. Don't include *implements* relationships with any *interface* in you model. Use the qualified name of the classes in your class diagram.