# Introduction to Professional Android Development

**Project No. 1 –** Waseem Nasim, Mubarak Ahmad

**Due: ~~Sun Feb 28, 2016 – 11:59pm~~**

Note: all programming work should be done on a branch "project1_firstName1_firstName2". The written parts should in a pdf file that is placed in the "assets" folder in the app in your branch. The answers file should be named the same as your branch.pdf

1. Explain what a class is in terms of object-oriented programming (OOP) in Java.

A class in OOP is the **blueprint** that is used to create an object. OOP envisions a program as a set of objects interacting with each other. It is a class that defines the attributes and behaviors of an object.

2. Explain what is meant by an "Object" in OOP and explain its application(s) in Java.

An object is an **instance** of a class. As explained earlier in the OOP paradigm a program is a set of objects interacting with each other. The object is constructed

3. Explain what is meant by an "instance" of a class. Give an example to illustrate.

An **instance** is a particular object created from a class. For example here is an example of a class.

```
public class Car {
     int year;
     String model, make;
     double  mileage;

     Car ( int y, String md, mk, double ml){
           year = y;  model = md; make = mk; mileage = ml;
     }



}
```

Given the class definition above, we can now create two instances of this class using the new keyword.

4. Explain the relationship between the three concepts mentioned in in 1, 2, 3.

Just as explained before a class is the blueprint and object is created from it using the new keyword. The term **instance** means the same as **object**. So for example waseemCar and mubarakCar are instances of the Car class. They are also (references to) objects.

5. Explain the concept of an "Anonymous" class in java. Explain how it is different than a regular class.

An anonymous class is used to create an object on the fly without having to name it. Here is an example where an object is created to handle click on a button and this is done by declaring an anonymous class.

```
someButton.setOnClickListener(     new
View.OnClickListener() {
     public void      onClick(View v) {

     // }     } );
```

An anonymous class is usually used when you have very little code and you are going to be instantiating it just once. To declare a proper class you would have had to write it separately.

```
public class ButtonListener {

}
```

6. What is an **interface** in Java? Give an example interface.

An interface can only contain method signatures and fields. An interface specifies the functionality but doesn't actually implement it. Other classes need to implement all the methods declared in an interface. Here is an example from Java Tutorial of a Bicycle interface which declares 4 methods.

```
interface Bicycle {

   void changeCadence(int newValue);
   void changeGear(int newValue);
   void speedUp(int increment);
   void applyBrakes(int decrement);
}
```

So in order for an object be a bicycles, its class would have to implement the Bicycle interface (by defining the four methods).

7. What is the difference between a class and an interface. Explain how they are related to each other but distinct.

An interface can only contain method signatures and fields but it does not define the method; it doesn't implement the methods. An class contains methods are defined and implement. A class can implement an interface by simply by providing the code for the methods specified in the interface. So here is an example of a class that uses / implements the ***Bicycle*** interface.

```
class ACMEBicycle implements Bicycle {
     int cadence = 0;
     int speed = 0;
     int gear = 1;

void changeCadence(int newValue) {
   cadence = newValue;     }


void changeGear(int newValue) {
    gear = newValue;     }

void speedUp(int increment) {
    speed = speed + increment;
}

void applyBrakes(int decrement) {

    speed = speed - decrement;     }

void printStates() {

   System.out.println("cadence:" +              cadence + "
speed:" +               speed + " gear:" + gear);
}
 }
```

8. How do you define an anonymous class?

This is already answered in question #5 with example.

9. How do instantiate an anonymous class?

This is also answered in question #5 with example. You use the **new** keyword on the class that is being defined.

10. What is the key difference between a "concrete" class and an anonymous class in how it is instantiated into an object?

A "concrete" class is explicitly declared (normally in a separate java file) and then you can make objects from it by using the new keyword. The anonymous class on the other hand is embedded in the code itself but you still use the new keyword.

11. Explain the use of the inheritance concept in Java and give an example to illustrate.

You can extend the properties and functionality of a class by using inheritance. So for example in question 3 we declare a class for Car. Now if we wanted to have automatic cars or manual cars we can create two more classes that extend from the basic car. Similarly, we could create bikes such as MountainBike by extended the regular ACMEBike. The benefit is that the extended classes still have the behaviors and attributes of the parent class.

```
public class ManualCar extends Car {
        int gear;

         public void changeGear(int gear){

        }
}
```

12. Explain what OOP "Design Pattern" does a "listener" implement.

A listener implements the "Observer" design pattern. For example, the button listener observes the button for any clicks and when it is clicked, the button handler method is involved / notified.

https://en.wikipedia.org/wiki/Observer_pattern

13. Give a definition of a "function" in OOP and compare it to a "method".

A function is a group of statements that perform a particular task and usually ends up returning the result. An example could be the average() function that takes in four grades and returns the average.

```
function  double average(double: mark1, mark2, mark3, mark4)
{
      return (mark1+mark2+mark3+mark4)/4;
}
```

Generally, in OOP the term method is used and methods belong to classes and are used to define a particular behavior.

14. Explain what is a "callback" function/method.

A callback method is defined to run when an event takes place. For example, in assignment 2-3, we defined method for handling the click event. So that button handler

method got called when the user clicked on the button.

15. Explain why "callbacks" and "listeners" are used in "modern" systems. What type of programming paradigm do they enable?

Use of callbacks and listener is an efficient way to handle events. Without that you would have to manually and continuously keep checking if the state (of a button for example) has changed. With the listener checking for the events, you do not have to do it. The listener notifies you when an event occurs and the call back method is called automatically.

16. Use the Android Developer documentation to select a list of 3 different listener interfaces and explain where and how they are used in the Android system.

**OnClickListener** - this is used to listed to button clicks using the onClick() method.

**OnTouchListener** - this interface is used to listen to touch. The call back method is **onTouch()**.

**OnDragListener –** is used to detect when a view is being dragged. The call back method is **onDrag()**.

17. In the above list, identify what part is the callback(s).

The call back methods for each interface are already above i.e. onClick(), onTouch(), onDrag()

18. Research the Android View system and give an explanation of how it works from XML to Java in as much technical detail as possible within 250 words. Cover each of the following terms in your description: xml, view, parent-view, child-view, layouts, java objects, inflation, reflection.

An activity is one complete task that the user can interact it (through one screen). Each activity's layout and various views (TextView, Button, group views) are specified using XML tags. All these components are structured hierarchically in a tree. The Android Run Time then converts these into Java objects that are loaded and displayed on the screen. This process is referred to as inflation as the setContentView() is called in the onCreate() method.

19. We have learned about "Activities" so far as being the 1-to-1 for a "screen" that you see. Define how Android "Fragments" are related to Activities in Android.

"A fragment is a self-contained, modular portion of UI that can be embedded within an activity. A fragment can only exist within an activity and can be created via XML or through java coding.

20. State 2 reasons (advantages) that lead to the introduction of Fragments to the Android UI system in v3.0

Fragments provide a mechanism for creating re-usable modules of user interface layout and application behavior, which once created can be embedded in activities.

21. Can you think of a Mobile App UI design that Fragments make possible that with activities would not be possible?

Fragments can be added to the activity and removed during run time.

22. Convert SecondActivity into a Splash screen for our application (it should go away after 5 seconds and launch MainActivity).

Done.

23. Enhance MainActivity to contain an "IPhone" type app bar menu at the bottom that looks like the picture in Figure 1.

Being Worked on – not finished.

24. Make the enhanced MainActivity have the following functionality:

a) Three tabs at the bottom with three 3 different icons and description text.
b) Make the tabs have a persistent selected and unselected state (loook different)
c) Create 3 different Fragments that you will associate with one tab each.
d) Show the appropriate Fragment when the corresponding tab is selected
e) Fragment1 should contain 3 buttons with 1 "WinterJacket" object associated with each button
f) When a button is Fragment1 is pressed - it should stay pressed
g) Fragment2 should have its background be the color of the selected WinterJacket and say what that color is in text on screen
h) Fragment3 should visually indicate the size of the WinterJacket with a different image and should say it in text as well.

**Hint**: you will have to find a way to share the information across the three fragments.

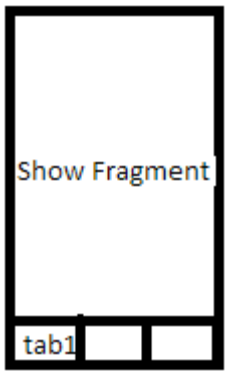


Figure 1

Being Worked on – not finished.