Md Rahman

Week 1 Daily 3 Research

1. Describe the following design principles: Singleton, Factory, Builder, Facade, Prototype

1. A singleton class allows for the creation of only one instance of that class. We define and initialize a static variable object inside the singleton class as well as create a private constructor so multiple instances of the class cannot be created by using Java’s default constructor. Finally, when creating your first and only instance of this class, we call a getInstance method (can be named anything really) that returns that static object variable.
2. Factory design. A way to create an object without exposing the logic behind the creation to the client. We refer to the newly created object using a common interface.

Main difference between factory and builder is that the builder allows freedom for creating the object with whatever parameters you want or don’t want, whereas with the factory it does not.

1. Builder Pattern. A way to create an object by going step by step, allowing the user to initialize only the fields they see fit. Typically, when creating an object, the user must specify all it’s values inside the constructor, or else a default blank object will be created. Then, using getters and setters the fields can be changed. However, using a Builder class, the user can initialize whatever field they see fit at the time of creation, and the rest of the values will just be null or empty. This works really well for creating an object containing multiple fields but the user only wishes to initialize certain ones during the object’s creation.
2. Facade. The complexity of the program will be hidden, and a simple interface will be provided so the user can access the services that the program offers. Involves a single class which provides simplified methods to the user when the user isn’t entirely sure about what they want. Great example taken from Geeks for Geeks:
   1. A user enters a hotel that has multiple restaurants and wants to see the menus for each restaurant. Menus are of three types: vegan, non-vegan and both. Instead of trying to go to each restaurant and asking for the type of food and then asking for the menu, he can ask the hotel keeper (a facade class) who can bring him the menus he chooses.
3. Prototype. Creating a duplicate object while not losing performance; instead, it’s more efficient at runtime because all the attributes from the original object are simply copied over from memory. Great for saving time and resources.

2. What is the differences in ART and Dalvik?

1. Dalvik does Just-in-Time compilation (JIT) which compiles your code to bytecode as it’s running. Then, each time you run your app, the part of the code required for its execution is going to be compiled to machine code. Each time your run your app, additional code will be compiled and cached, allowing for reusability and a smaller use of memory.
2. Android RunTime is the updated version of Dalvik. ART does Ahead-of-Time compilation (AOT) which compiles before your run your code. This removes the lag we typically see when we open an app on our device. In addition, ART runs app machine code directly, putting less strain on the CPU. However, machine code takes up more space than bytecode and more memory at execution time.

3. What is the android manifest used for?

1. All apps must have an AndroidManifest.xml file. This file describes essential information about your app, such as Package name, components of the app such as activities and services, and uses intent filters and permissions to determine how to coordinate between apps.

4. Define the difference in Runtime and Compile Time.

1. Compile time is the instance where your code is converted to an executable. Compile time errors are caught by the compiler as you write your code. These errors can be syntax errors, missing references to files, etc.
2. Runtime. After your code has been compiled into an executable program, you may still experience errors when you run your code. These runtime errors come in the form of exceptions, program crashes, logical errors, etc.

5. How does each of the following units of measure for view work: sp, dp, px, pt, in, mm

1. Sp - Scale independent pixels. Scaled by the users font size preference.
2. Dp - Density independent pixels. Based on the physical density of the screen. For a standard 160 dpi screen, 1dp = 1 pixel.
3. Px - the actual pixels on the screen
4. Pt - Points, 1/72 of an inch based on the physical size of the screen.
5. In - Inches. Based on physical size of screen.
6. Mm - Millimeters. Once again based off physical size of screen. 1 mm = .1 cm = .039 in

6. Describe what each section of the Android Platform arch. Details.

1. Linux Kernel - The foundation of the platform. At its heart, android is running a stripped down version of linux. This allows for features such as threading and allows for the development of hardware drivers.
2. Hardware Abstraction Layer - Provides standard interfaces for accessing hardware layers such as camera.
3. Android Runtime (ART) - Used to be Dalvik which did just-in-time compilation, now it does ahead-of-time compilation. This allows for less strain of the batter and the ability to run many VM’s on low-memory devices.
4. Native C++ libraries - Many core components such as ART and HAL are build using libraries written in C++.
5. Java framework API - All the features we need to develop in Android are provided thru API’s written in Java.
6. System Apps - Core applications build into Android, such as email, calendars, internet, etc.

7. What is reflection in JAVA?

1. An API used to examine or modify the behavior of classes, methods and interfaces at runtime. We can get info about the class to which an object belongs and even the specific methods of that class.
   1. getClass() gets name of class for which an object belongs
   2. getConstructors() gets the public constructs of a class
   3. getMethods() get the public methods of a class

The benefits of reflection is that it allows for a developer to better debug their code because they gain access to external, defined classes that can provide information they may otherwise have a hard time accessing.

8. How does gradle work behind the scene.

Gradle is a JVM based build system that takes all source files like .java and .xml, then applies appropriate build tools to do tasks such as converting java class files into dex (Dalvik executable) files, which are then grouped into one APK file. It’s a way to automate the building process so you don’t need to write your own; however, you can write your own specific scripts in the build.gradle files to automate certain tasks for you.