Research

**1. What are Listeners?**

Listener is an interface in the View class that contains a single call-back method. These listeners must be placed in the views to handle input events, such as button clicks, edittext, etc. Some of the call-back methods used in Android are onClick(), onLongClick(), onFocusChange(), onKey(), onTouch(), and onCreateTextMenu(). When triggered by the user interaction in the UI, Android framework will call these methods. After an event is listened, event registration is also needed to register an event listener with an event handler. So, the event handler can be called when an Event Listener fires the event.

There are different ways to register event listeners in android:

- By specifying event handlers directly in activity\_main.xml.

- By using Activity class that implements a listener interface.

- By using an anonymous class.

Some of most commonly used callback methods for event handlers are as follow:

- onKeyDown()

- onKeyUp()

- onTrackballEvent()

- onTouchEvent()

- onFocusChanged()

**2. How does Java garbage collection work?**

Java garbage collection is the process by Java programs in performing automatic memory management. Java programs are compiled to bytecode which can be run on a Java Virtual Machine (JVM). At this point, objects are created on the heap, which is a portion of memory dedicated to the program. Some of these objects will essentially no longer be needed and it’s the garbage collection’s job to find these unused objects and to remove them to free up memory. This process is done automatically by the Java Garbage Collection whose implementation lives in the JVM. Each JVM can implement garbage collection any way it wants.

There are many benefits of Java garbage collection, but the biggest one is that it automatically handles the removal/deletion of unused objects that are out of reach to free up memory resources. In Java, garbage collection is not something that a programmer needs to consider all the time. However, garbage collection in other languages, like C and C++, programmers must implement their own memory management in their code manually.

**3. What is the android manifest used for?**

Android manifest is used to describe your application’s essential information to the build tools of Android system, OS and Google Play. This essential information about the application is required by Android system before it can run any of the application’s code.

The manifest file is also used for the following:

* To name the Java package for the application.
* To describe the application components, which include the activities, services, broadcast receivers, and content providers.
* To determine the processes that host the application components.
* To declare permissions the application needs to have in order to access protected parts of the API and to interact with other applications.
* It includes the list of Instrumentation classes that provides profiling and other information for when the application runs.
* To declare the minimum level of the Android API the application requires.
* Contains the libraries an application is linked against.

**4. Define the difference in Runtime and Compile Time.**

Runtime is an instance where executable is running. Compile time is an instance where the code you entered is converted to an executable file. Both runtime and compile time are terms to show different stages of software development. Both terms are also used by programmers to refer different type of errors.

Compile time occurs during the compile time and the errors are occurred due to typing mistake caused by the programmers because of the improper use of a particular language syntax. The compiler will not let your program execute a single line until all the syntax errors are removed. Some of the common compile time errors are syntax error, typechecking error, and compiler crashes.

Runtime type checking occurs during program run time after compilation. Runtime errors are generated when the program is a running state. These errors usually cause your program to behave unexpectedly and they are often called Exceptions. Some of the runtime errors are division by zero, running out of memory and dereferencing a null pointer.

**5. What is reflection in JAVA?**

Reflection in Java is an API that can be used to examine or modify the behaviors of classes, methods and interfaces at runtime. In other words, Java Reflection allows the program to inspect classes, interfaces, fields, and method at runtime without knowing the name of the class, method, etc. one can also instantiate new objects, get/set values and invoke methods using reflection. Java reflection can be used to map JSON properties to getter/setter methods in Java objects and it can also be used to map the column names a JDBC ResultSet to getter/setter methods in a Java object. Java Reflection classes are provided in the java.lang.reflect package. Reflection gives us information about the class, its object and the methods of that class. Reflection allows us to invoke methods at runtime irrespective of the access modifiers used.

A Java Reflection example would be:

Method[] methods = MyObject.class.getMethods();

For(Method method : methods){

System.out.println(“method = “ + method.getName());

}

This example shows that using Reflection allows you to obtain a Class object from the class called MyObject. By using the MyObject class object, it will allow the program to get a list of the methods in that class, iterating the methods array and then prints out the method names. More examples are as follow.

Class myObjectClass = MyObject.class;

Field[] fields = myObjectClass.getFields();

The reflection example above will allow you to inspect all the fields within that class along with its names, access modifiers, etc.

**6. How does gradle work behind the scene.**

Gradle is an advanced build management system for general purposes based on Groovy and Kotlin. Gradle supports automatic configuration and download of dependencies or other libraries. Additionally, it also supports Maven and Ivy repositories for retrieving dependencies. In Gradle, a folder of code and resources which is built to produce an artifact, called a project. A project contains a series of tasks, such as compile, test, assembly, etc. These tasks can depend on each other to form a dependency graph. Gradle, behind the scene, keeps track of all of it and ensure that all tasks are run in the right order. It also keeps track of any changes on the tasks since the last run. It can only execute the parts of the task graph that was changed by a task.

A task can have description and group properties, along with two methods called doFirst and doLast. These are the first and last operations to run when a task is started. Working with tasks and dependencies are quite repetitive and that’s where Gradle comes in play. The most common tasks in many projects are compile and test. The main concepts in Gradle are Project, Task and Plugin. Plugin is a collection of tasks that can be imported or included in a project. Each Project for Gradle has a build.gradle file, which is written in Groovy DSL.