Kotlin - Visibility Control

In this chapter, we will learn about different modifiers available in Kotlin language. **Access modifier** is used to restrict the usage of the variables, methods and class used in the application. Like other OOP programming language, this modifier is applicable at multiple places such as in the class header or method declaration. There are four access modifiers available in Kotlin.

Private

The classes, methods, and packages can be declared with a private modifier. Once anything is declared as private, then it will be accessible within its immediate scope. For instance, a private package can be accessible within that specific file. A private class or interface can be accessible only by its data members, etc.

```
private class privateExample {
   private val i = 1
   private val doSomething() {
   }
}
```

In the above example, the class **"privateExample"** and the variable i both can be accessible only in the same Kotlin file, where its mentioned as they all are declared as private in the declaration block.

Protected

Protected is another access modifier for Kotlin, which is currently not available for top level declaration like any package cannot be protected. A protected class or interface is visible to its subclass only.

```
class A() {
   protected val i = 1
}
class B : A() {
   fun getValue() : Int {
      return i
   }
}
```

In the above example, the variable **"i"** is declared as protected, hence, it is only visible to its subclass.

Internal

Internal is a newly added modifier introduced in Kotlin. If anything is marked as internal, then that specific field will be in the internal field. An Internal package is visible only inside the module under which it is implemented. An internal class interface is visible only by other class present inside the same package or the module. In the following example, we will see how to implement an internal method.

```
class internalExample {
   internal val i = 1
   internal fun doSomething() {
   }
}
```

In the above example, the method named “doSomething” and the variable is mentioned as internal, hence, these two fields can be accessible only inside the package under which it is declared.

Public

Public modifier is accessible from anywhere in the project workspace. If no access modifier is specified, then by default it will be in the public scope. In all our previous examples, we have not mentioned any modifier, hence, all of them are in the public scope. Following is an example to understand more on how to declare a public variable or method.

```
class publicExample {
   val i = 1
   fun doSomething() {
   }
}
```

In the above example, we have not mentioned any modifier, thus all these methods and variables are by default public.