Kotlin - Extension

In this chapter, we will learn about another new feature of Kotlin named “Extension”. Using extension, we will be able to add or remove some method functionality even without inheriting or modifying them. Extensions are resolved statistically. It does not actually modify the existing class, but it creates a callable function that can be called with a dot operation.

Function Extension

In function extension, Kotlin allows to define a method outside of the main class. In the following example, we will see how the extension is implemented at the functional level.

Live Demo

```
class Alien {
   var skills : String = "null"

   fun printMySkills() {
      print(skills)
   }
}
fun main(args: Array<String>) {
   var  a1 = Alien()
   a1.skills = "JAVA"
   //a1.printMySkills()

   var  a2 = Alien()
   a2.skills = "SQL"
   //a2.printMySkills()

   var  a3 = Alien()
   a3.skills = a1.addMySkills(a2)
   a3.printMySkills()
}
fun Alien.addMySkills(a:Alien):String{
   var a4 = Alien()
   a4.skills = this.skills + " " +a.skills
   return a4.skills
}
```

In the above example, we don’t have any method inside “Alien” class named as “addMySkills()”, however, we still are implementing the same method somewhere else outside of the class, This is

the magic of extension.

The above piece of code will generate the following output in the browser.

```
JAVA SQL
```

Object Extension

Kotlin provides another mechanism to implement static functionality of Java. This can be achieved using the keyword “companion object”. Using this mechanism, we can create an object of a class inside a factory method and later we can just call that method using the reference of the class name. In the following example, we will create a “companion object”.

Live Demo

```
fun main(args: Array<String>) {
   println("Heyyy!!!"+A.show())
}
class A {
   companion object {
      fun show():String {
         return("You are learning Kotlin from TutorialsPoint.com")
      }
   }
}
```

The above piece of code will yield the following output in the browser.

```
Heyyy!!! You are learning Kotlin from TutorialsPoint.com
```

The above example seems like static in Java, however, in real-time we are creating an object as a member variable of that same class. This is why it is also included under extension property and can be alternatively called as an object extension. You are basically extending the object of the same class to use some of the member functions.