Kotlin - Functions

Kotlin is a statically typed language, hence, functions play a great role in it. We are pretty familiar with function, as we are using function throughout the examples. Function is declared with the keyword “fun”. Like any other OOP, it also needs a return type and an option argument list.

In the following example, we are defining a function called MyFunction and from the main function we are calling this function and passing some argument.

Live Demo

```
fun main(args: Array<String>) {
   println(MyFunction("tutorialsPoint.com"))
}
fun MyFunction(x: String): String {
   var c:String  = "Hey!! Welcome To ---"
   return (c+x)
}
```

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

```
Hey!! Welcome To ---tutorialsPoint.com
```

The function should be declared as follows −

```
fun <nameOfFunction>(<argument>:<argumentType>):<ReturnType>
```

Following are some of the different types of function available in Kotlin.

Lambda Function

Lambda is a high level function that drastically reduces the boiler plate code while declaring a function and defining the same. Kotlin allows you to define your own lambda. In Kotlin, you can declare your lambda and pass that lambda to a function.

Take a look at the following example.

Live Demo

```
fun main(args: Array<String>) {
   val mylambda :(String)->Unit  = {s:String->print(s)}
   val v:String = "TutorialsPoint.com"
   mylambda(v)
}
```

In the above code, we have created our own lambda known as “mylambda” and we have passed one variable to this lambda, which is of type String and contains a value “TutorialsPoint.com”.

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

```
TutorialsPoint.com
```

Inline Function

The above example shows the basic of the lambda expression that we can use in Kotlin application. Now, we can pass a lambda to another function to get our output which makes the calling function an inline function.

Take a look at the following example.

Live Demo

```
fun main(args: Array<String>) {
   val mylambda:(String)->Unit  = {s:String->print(s)}
   val v:String = "TutorialsPoint.com"
   myFun(v,mylambda) //passing lambda as a parameter of another function
}
fun myFun(a :String, action: (String)->Unit) { //passing lambda
   print("Heyyy!!!")
   action(a)// call to lambda function
}
```

The above piece of code will yield the following output in the browser. Using inline function, we have passed a lambda as a parameter. Any other function can be made an inline function using the “inline” keyword.

```
Heyyy!!!TutorialsPoint.com
```