Lambda Expressions

Lambdas Expressions are essentially anonymous functions that we can treat as values – we can, for example, pass them as arguments to methods, return them, or do any other thing we could do with a normal object.

```
val square : (Int) -> Int = { value -> value * value }

val nine = square(3)
```

Lambda Expressions

```
val doNothing : (Int) -> Int = { value -> value }

val add : (Int, Int) -> Int = { a, b -> a + b }

val print : (Int) -> Unit = { value -> println(value) }
```

Higher-Order Functions and Lambdas

A higher-order function is a function that takes functions as parameters, or returns a function.

Higher-Order Functions and Lambdas

```
object AwesomeLambda {
    fun passMeFunction(abc: () -> Unit) {
        // I can take function and execute it
        // do something here
        // execute the function which received as an argument
        abc()
    }
}
```

Higher-Order Functions and Lambdas

```
// You can send the function as a parameter
AwesomeLambda.passMeFunction(
    {
        val user = User()
        user.name = "ABC"
        println("Lambda is awesome")
    }
)
```

Higher-Order Functions and Lambdas

```
class NetworkService {
    fun fetchData(success: (result: String) -> Unit,
                  error: (error: String) -> Unit) {
        val response: Response = fetchResponseFromServer()
        if (response.isSuccessful) {
            // call success function with result as a parameter
            success(response.result)
        } else {
            // call error function with error as a parameter
            error(response.error)
        }
    }
}
```

Higher-Order Functions and Lambdas

```
networkService.fetchData({ result ->
    // do something with the result
    showResult(result)
}, { error ->
    // do something with the error
    showError(error)
})
```

Higher-Order Functions and Lambdas

```
fun add(a: Int, b: Int): Int {
    return a + b
}

fun returnMeAddFunction(): ((Int, Int) -> Int) {
    // can do something and return function as well
    // returning function
    return ::add
}
```

Extension Functions

```
fun ImageView.loadImage(url: String) {
    Glide.with(context).load(url).into(this)
}

// now you can call like this from anywhere
imageView.loadImage(url)
```