# Swift - Optionals

Swift 4 also introduces **Optionals** type, which handles the absence of a value. Optionals say either "there is a value, and it equals x" or "there isn't a value at all".

An Optional is a type on its own, actually one of Swift 4’s new super-powered enums. It has two possible values, **None** and **Some(T)**, where **T** is an associated value of the correct data type available in Swift 4.

Here’s an optional Integer declaration −

var perhapsInt: Int?

Here’s an optional String declaration −

var perhapsStr: String?

The above declaration is equivalent to explicitly initializing it to **nil** which means no value −

var perhapsStr: String? = nil

Let's take the following example to understand how optionals work in Swift 4 −

var myString:String? = nil

if myString != nil {

print(myString)

} else {

print("myString has nil value")

}

When we run the above program using playground, we get the following result −

myString has nil value

Optionals are similar to using **nil** with pointers in Objective-C, but they work for any type, not just classes.

## **Forced Unwrapping**

If you defined a variable as **optional**, then to get the value from this variable, you will have to **unwrap** it. This just means putting an exclamation mark at the end of the variable.

Let's take a simple example −

var myString:String?

myString = "Hello, Swift 4!"

if myString != nil {

print(myString)

} else {

print("myString has nil value")

}

When we run the above program using playground, we get the following result −

Optional("Hello, Swift 4!")

Now let's apply unwrapping to get the correct value of the variable −

var myString:String?

myString = "Hello, Swift 4!"

if myString != nil {

print( myString! )

} else {

print("myString has nil value")

}

When we run the above program using playground, we get the following result.

Hello, Swift 4!

## **Automatic Unwrapping**

You can declare optional variables using exclamation mark instead of a question mark. Such optional variables will unwrap automatically and you do not need to use any further exclamation mark at the end of the variable to get the assigned value. Let's take a simple example −

var myString:String!

myString = "Hello, Swift 4!"

if myString != nil {

print(myString)

} else {

print("myString has nil value")

}

When we run the above program using playground, we get the following result −

Hello, Swift 4!

## **Optional Binding**

Use optional binding to find out whether an optional contains a value, and if so, to make that value available as a temporary constant or variable.

An optional binding for the **if** statement is as follows −

if let constantName = someOptional {

statements

}

Let's take a simple example to understand the usage of optional binding −

var myString:String?

myString = "Hello, Swift 4!"

if let yourString = myString {

print("Your string has - \(yourString)")

} else {

print("Your string does not have a value")

}