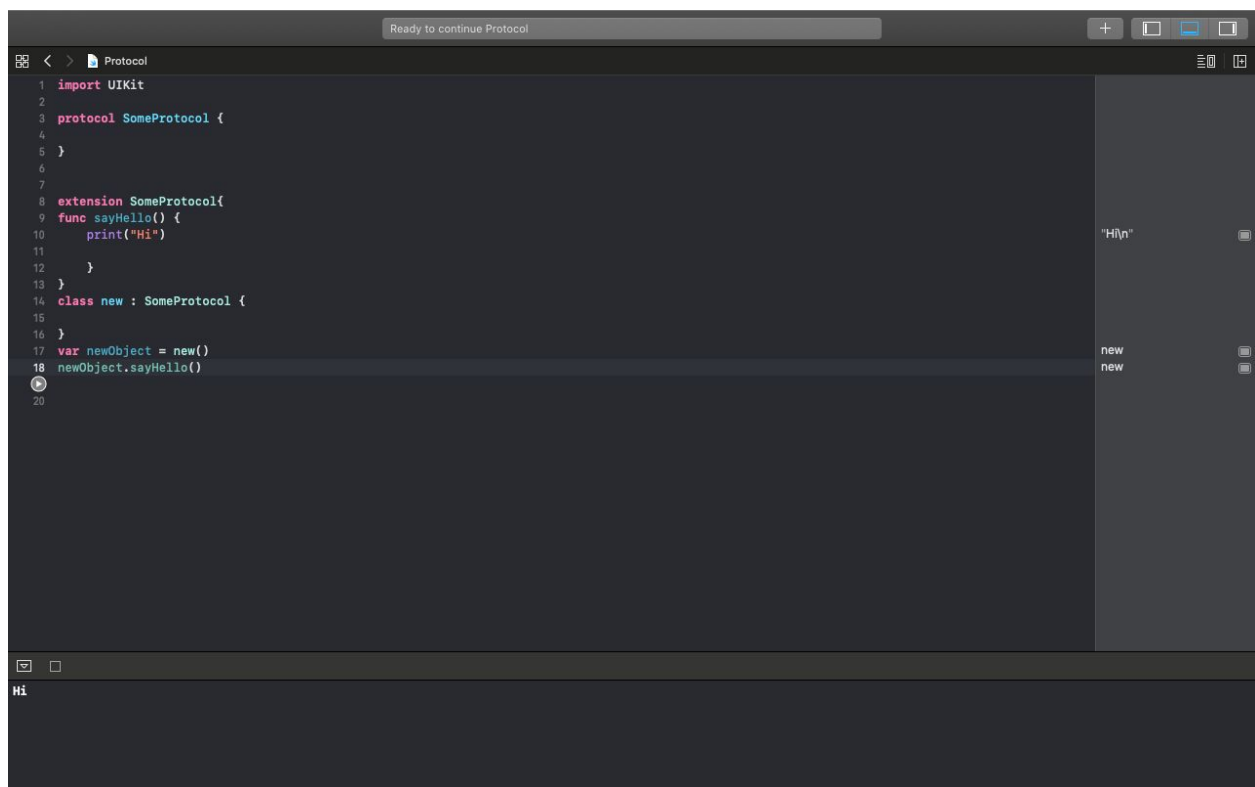


1. What is extension?

Extension is used to add new functionality to an existing class, Structure, Enum or Properties

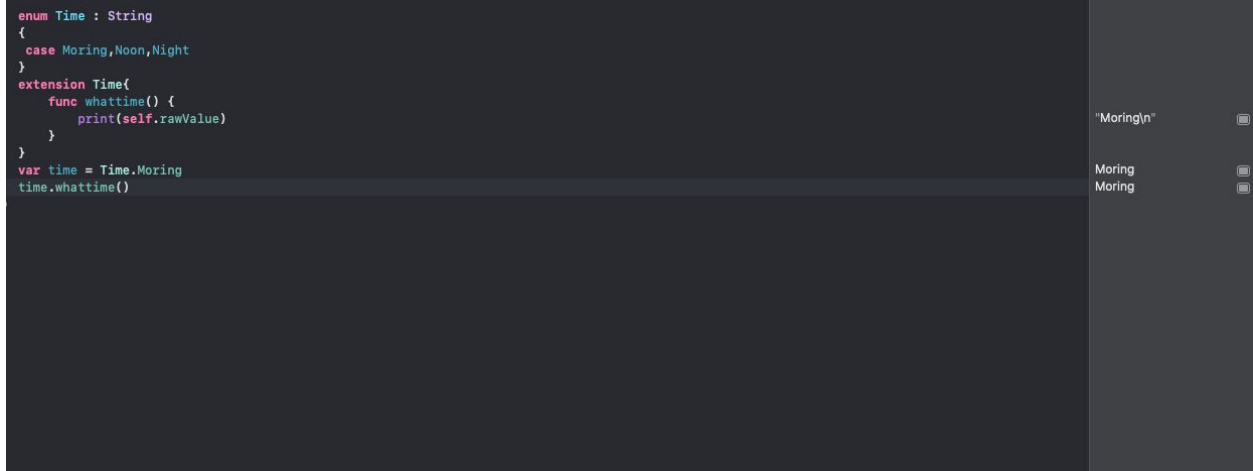
3. Write a protocol and create an extension of the protocol. In extension create a function

```
func sayHello() {  
    print("Hello!")  
}
```



4. Write an enum and create an extension of the enum.

```
enum Time : String
{
    case Moring, Noon, Night
}
extension Time {
    func whattime() {
        print(self.rawValue)
    }
}
var time = Time.Moring
time.whattime()
```



5.What is Generic?

Generic allows us to write flexible and reusable code that can be used with any type of defined by the user.We can write code that avoid duplication

6.Explain Generic with an example

```
func swapTwoInts(_ a: inout Int, _ b: inout Int) {
    let temporaryA = a
    a = b
    b = temporaryA
}
```

```
func swapTwoStrings(_ a: inout String, _ b: inout String) {
    let temporaryA = a
    a = b
    b = temporaryA
}
```

```
func swapTwoValues<T>(_ a: inout T, _ b: inout T) {
```

```
let temporaryA = a

a = b

b = temporaryA

}
```

7. Explain the difference between map and compactMap with an example.

map() is able to return a different type from the one that was originally used. So, this will convert our integer array to a string array

CompactMap() Working with optionals can be annoying, but compactMap() can make life much easier: it performs a transformation (the “map” part of its name), but then unwraps all the optionals and discards any that are nil.

```
31 var someArray = [10,11,nil,12,13]
32 print(someArray.map{$0})
33 print(someArray.compactMap({$0}))
34
35
```

No Completions

[Optional(10), Optional(11), nil, Optional(12), Optional(13)]
[10, 11, 12, 13]

8. Write an example of reduce function with initial value 1000.

```
34 var someArray = [10,11,14,12,13]
35 var newArray=someArray.reduce(1000, {$0 + $1})
36 print(newArray)
37
38
39
40
```

1060

9.

```
38 struct Person {
39
40
41
42     var name : String
43
44     var age : Int
45
46
47
48 }
49
50 let person1 = Person(name: "Sam", age: 23)
51
52 let person2 = Person(name: "John", age: 30)
53
54 let person3 = Person(name: "Rob", age: 27)
55
56 let person4 = Person(name: "Luke", age: 20)
57
58
59
60 let personArray = [person1, person2, person3, person4]
61 func ageMore(_ arr : [Person]) -> [Person]{
62     let newArray = arr.filter{$0.age>25}
63     return newArray
64 }
65
66 print(ageMore(personArray))
67
68
69
```

[__lldb_expr_25.Person(name: "John", age: 30), __lldb_expr_25.Person(name: "Rob", age: 27)]