

(More) Swift Basics

Bas Broek

Force! Casting

```
var statusCode: Int?
```

```
print(statusCode!) // crash 😱
```

```
if statusCode != nil {  
    print(statusCode!) // Please don't do this. 😓  
}
```

```
statusCode = 200
```

```
if let statusCode = statusCode {  
    print(statusCode) // Int, not Optional<Int>  
}
```

??

```
var statusCode: Int?
```

```
let result1 = statusCode // Optional<Int>
```

```
let result2 = statusCode ?? 200 // Int
```

```
// Which is the same as
```

```
let result3 = (statusCode == nil) ? 200 : statusCode! // or
```

```
let result4 = (statusCode != nil) ? statusCode! : 200
```

Enums

```
enum Number: Int {  
    case zero // = 0  
    case one  // = 1  
}
```

```
enum Host: String {  
    case dolores // = "dolores"  
    case bernard // = "bernard"  
}
```

// Here, you get an initializer for :

```
Number(rawValue: 1) // .one 
```

```
Host(rawValue: "dolores") // .dolores 
```

Con(struct)or

Con(struct)or

Initializer

Initializer

```
struct Book {  
    let title: String  
    let author: String  
}
```

```
Book(title: "De IJzeren Wil", author: "Bas Haring")
```


More Enums

```
func track(number: Number) {  
    print(number.rawValue)  
}  
  
// You can call this as  
  
track(number: Number.zero) // 0  
  
// .. but also as  
  
track(number: .zero) // 0
```

Extensions

```
extension String {  
    func localized() -> String {  
        return NSLocalizedString(self, comment: "")  
    }  
}
```

```
"hello".localized()
```

```
extension Int {  
    func times(task: (Int) -> Void) {  
        (1...self).forEach {  
            task($0)  
        }  
    }  
}
```

```
3.times { number in  
    print(number) // 1, 2, 3  
}
```

Typealias

<https://github.com/BasThomas/Analysis>

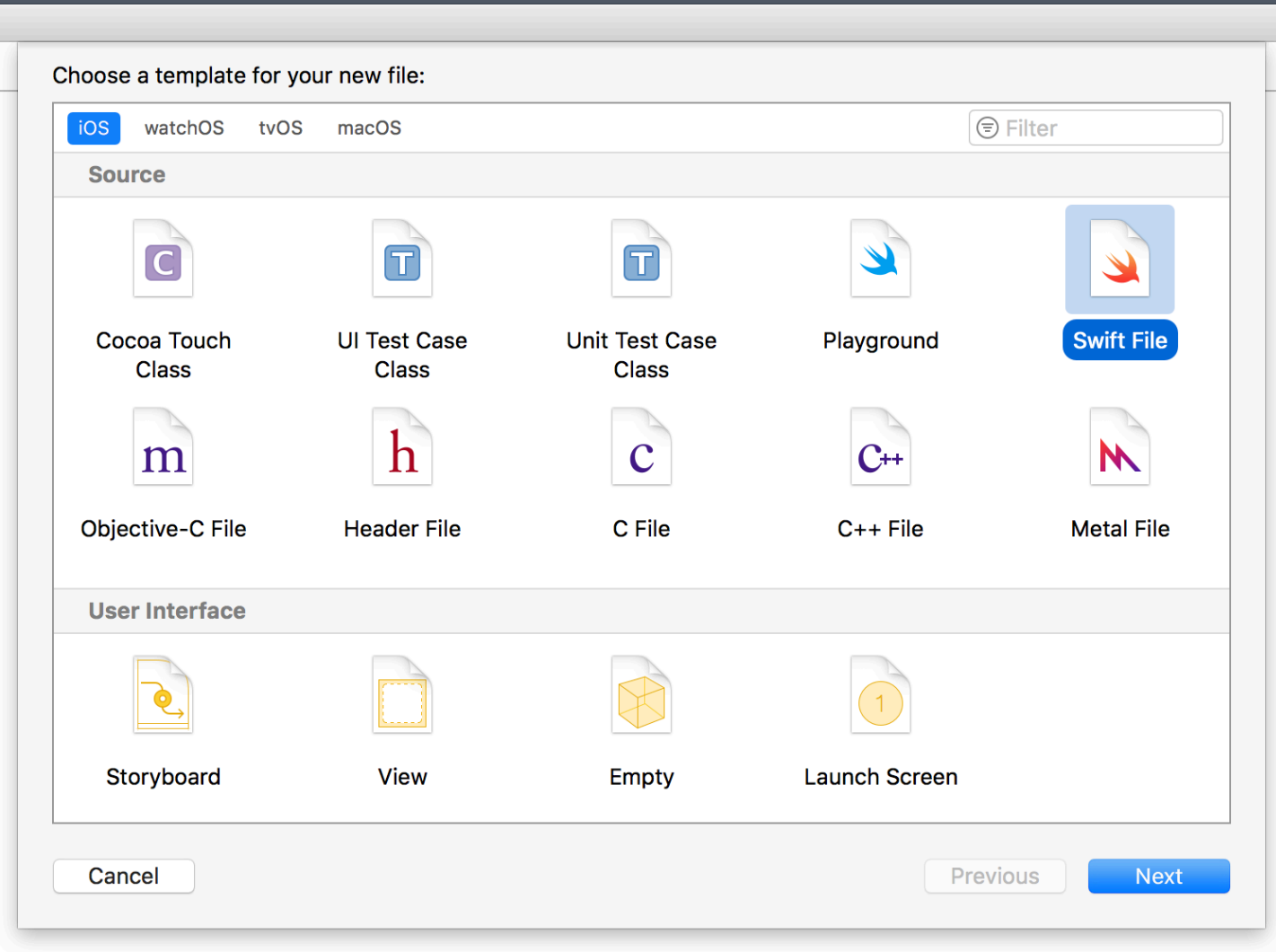
```
public struct Analysis {
    public typealias Percentage = Double

    /// Returns the frequency of the specified word.
    ///
    /// - Parameter caseSensitive: Indicating if words
    /// should be counted regardless of their case sensitivity.
    /// Defaults to `false`.
    ///
    /// - Returns: A percentage based on the `wordCount()`.
    public func frequency(of word: String, caseSensitive: Bool = false) -> Percentage {
        return Double(occurrences(of: word, caseSensitive: caseSensitive)) / Double(wordCount()) * 100.0
    }
}

let analysis = "What is this?".analysed()
analysis.frequency(of: "what") // 33.33
```

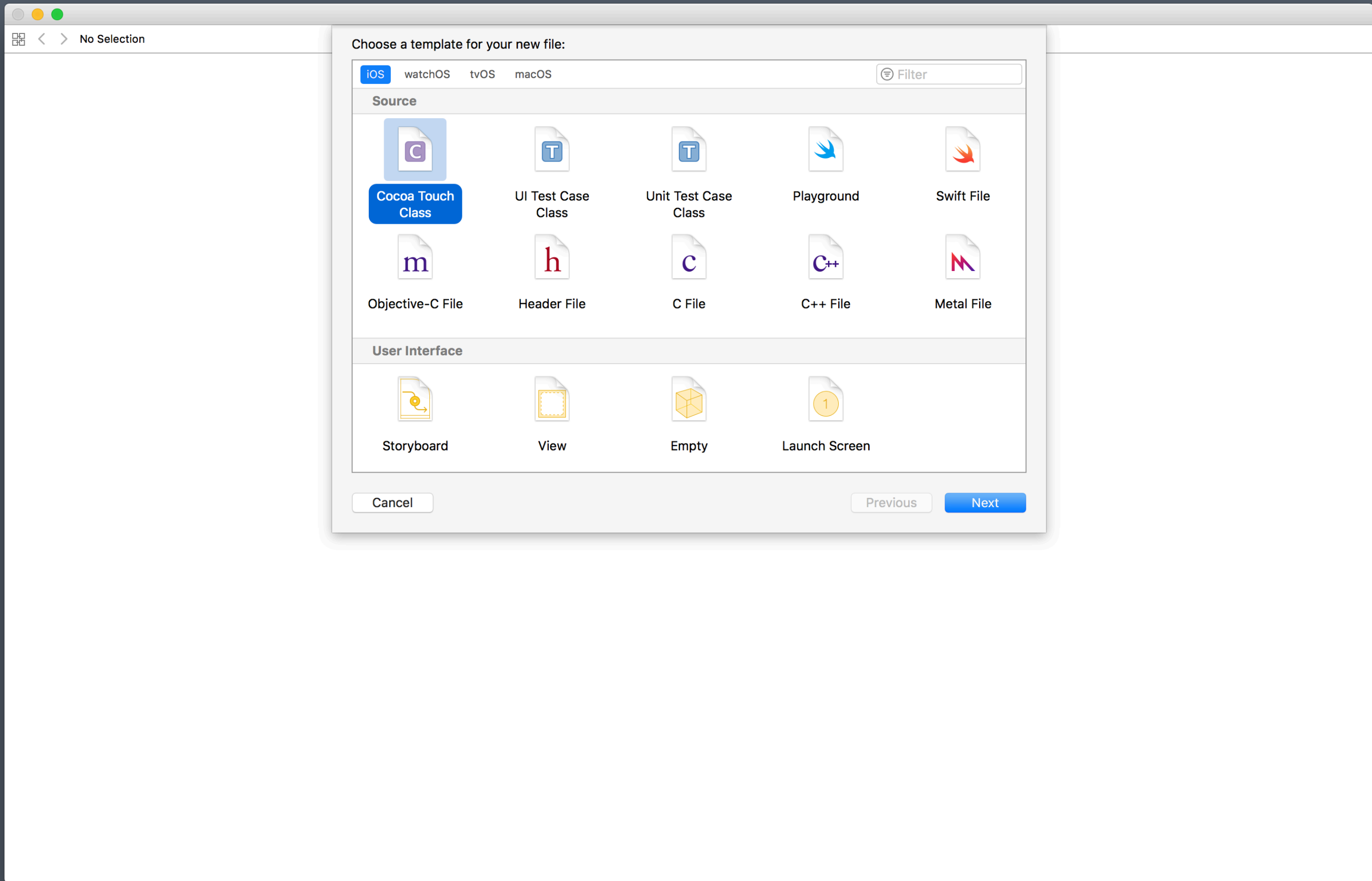
```
typealias 📖 = Book
```

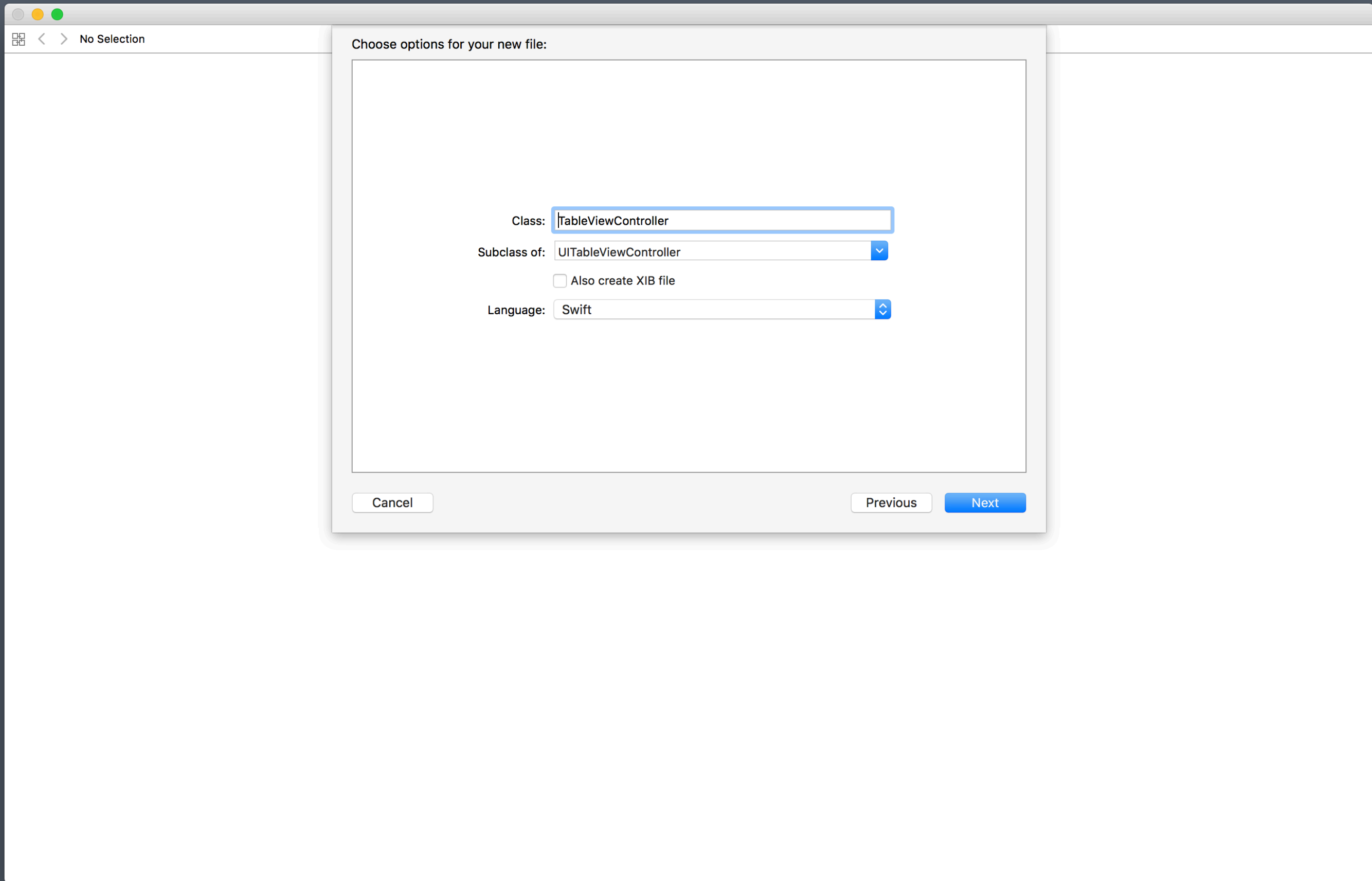
```
📖(title: "A Dream of Spring", author: "George RR Martin") // 🥲
```



File creation

```
1 //  
2 // File.swift  
3 //  
4 //  
5 // Created by Bas Broek on 07/12/2016.  
6 //  
7 //  
8  
9 import Foundation  
10
```





TableViewController.swift > No Selection

```

1 //
2 // TableViewController.swift
3 //
4 //
5 // Created by Bas Broek on 07/12/2016.
6 //
7 //
8
9 import UIKit
10
11 class TableViewController: UITableViewController {
12
13     override func viewDidLoad() {
14         super.viewDidLoad()
15
16         // Uncomment the following line to preserve selection between presentations
17         // self.clearsSelectionOnViewWillAppear = false
18
19         // Uncomment the following line to display an Edit button in the navigation bar for this view controller.
20         // self.navigationItem.rightBarButtonItem = self.editButtonItem()
21     }
22
23     override func didReceiveMemoryWarning() {
24         super.didReceiveMemoryWarning()
25         // Dispose of any resources that can be recreated.
26     }
27
28     // MARK: - Table view data source
29
30     override func numberOfSections(in tableView: UITableView) -> Int {
31         // #warning Incomplete implementation, return the number of sections
32         return 0
33     }
34
35     override func tableView(_ tableView: UITableView, numberOfRowsInSectionSection section: Int) -> Int {
36         // #warning Incomplete implementation, return the number of rows
37         return 0
38     }
39
40     /*
41     override func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
42         let cell = tableView.dequeueReusableCell(withIdentifier: "reuseIdentifier", for: indexPath)
43
44         // Configure the cell...
45
46         return cell
47     }
48     */
49
50     /*
51     // Override to support conditional editing of the table view.
52     override func tableView(_ tableView: UITableView, canEditRowAt indexPath: IndexPath) -> Bool {
53         // Return false if you do not want the specified item to be editable.
54         return true
55     }
56     */
57
58     /*
59     // Override to support editing the table view.
60     override func tableView(_ tableView: UITableView, commit editingStyle: UITableViewCellEditingStyle, forRowAt indexPath: IndexPath) {
61         if editingStyle == .delete {
62             // Delete the row from the data source
63             tableView.deleteRows(at: [indexPath], with: .fade)
64         } else if editingStyle == .insert {
65             // Create a new instance of the appropriate class, insert it into the array, and add a new row to the table view
66         }
67     }
68     */
69
70     /*
71     // Override to support rearranging the table view.
72     override func tableView(_ tableView: UITableView, moveRowAt fromIndexPath: IndexPath, to: IndexPath) {
73
74     }
75     */

```