Getting Started with UITableView in Swift

Muhammad Ariful Islam Follow
May 3 · 6 min read ★

Image by Héctor J. Rivas via Unsplash

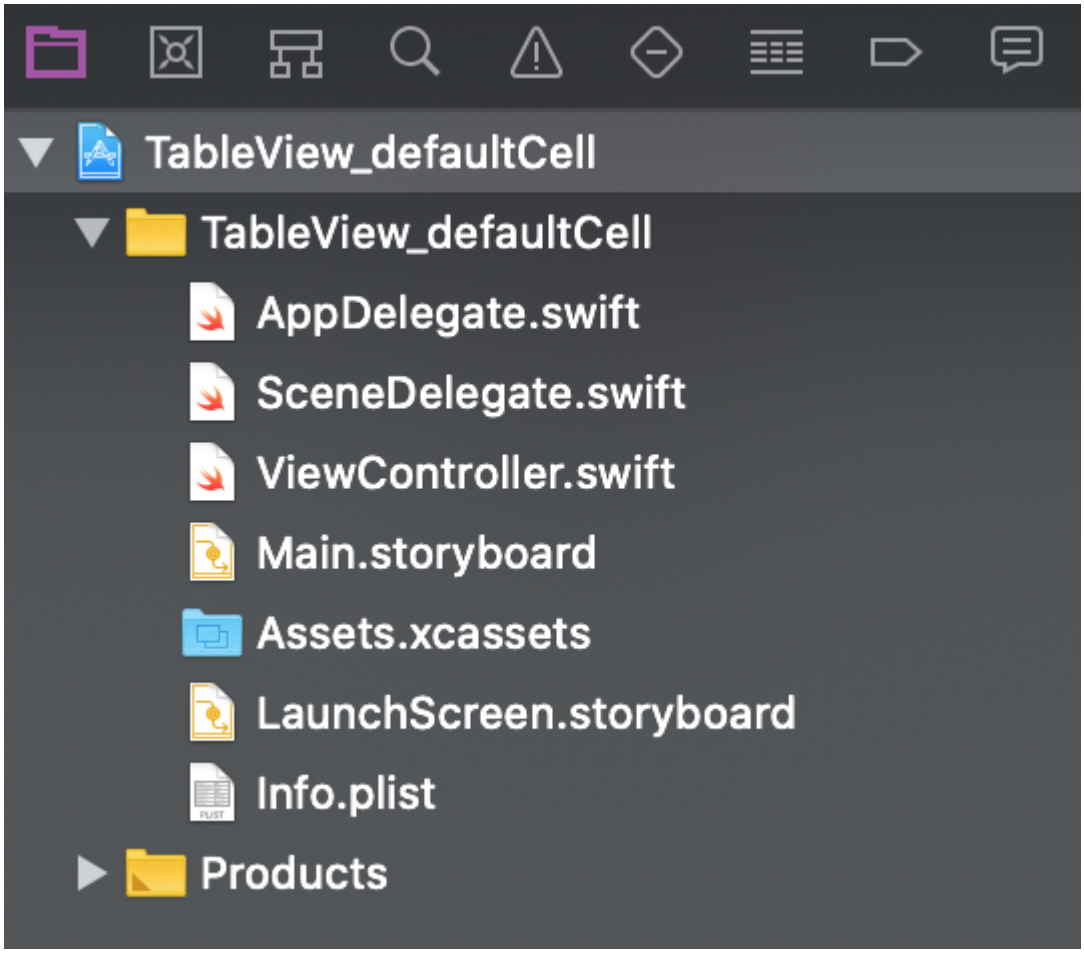
Overview

In iOS development, the table view is one of the most basic and reused design interfaces. It can present a large amount of data using rows arrange in a column. In this tutorial, we will learn how to configure a basic TableView in iOS using default and custom cell. So let's get started

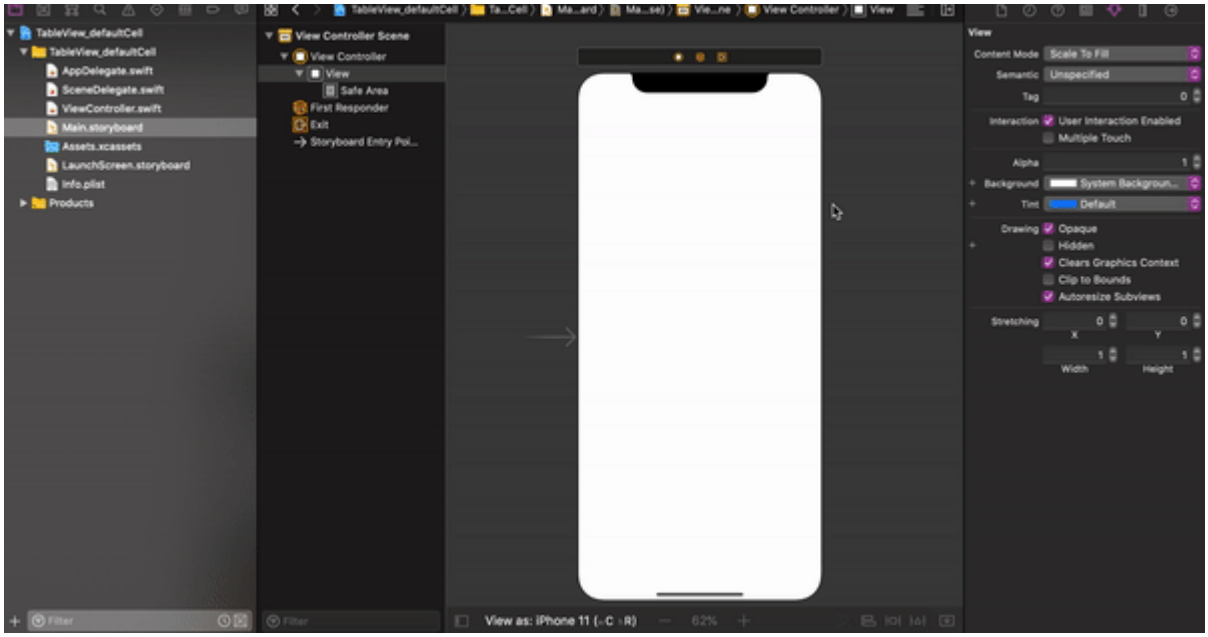
> *This tutorial is written using **Swift 5, Xcode 11.2, iOS 13 & Storyboard Interface.***

1. **Configure TableView using default cell:** Let's see how we can configure a table view using the default cell step by step.

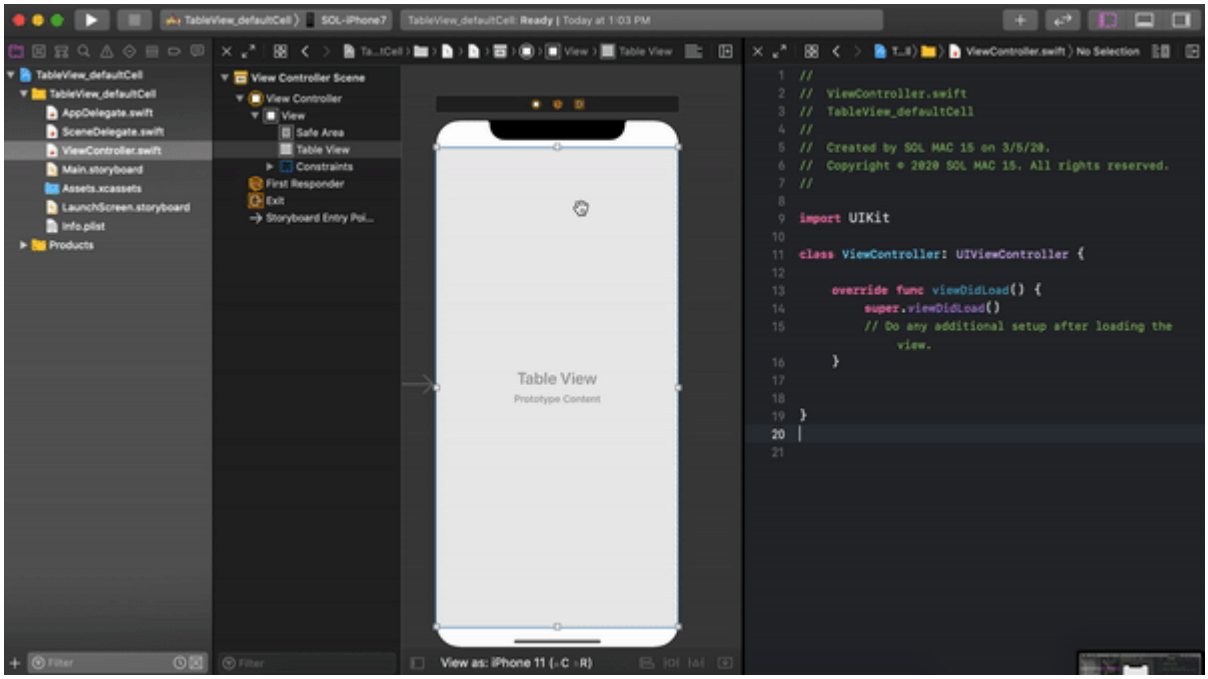
Step 1: **Create an Xcode project & setup TableView**: Open your Xcode -> create a new Xcode project -> Choose ios and single view application as your project template -> name it as you want -> create. After that, your file structure should seem like this.



Open your "Main.storyboard" file, you will find a view controller there. Drag and drop a TableView from the library. Set constrain as (0,0,0,0) as following



Now open your “ViewController.swift” file from the project navigator by “Option + click” on that. Insert outlate of your TableView to the view controller file.



Extends your “ViewController” class to “UITableViewDelegate” and “UITableViewDataSource”. You have to add some protocol stubs, add them. You “ViewController” should seem like this

```
import UIKit
```

```
class ViewController: UIViewController, UITableViewDelegate,
    UITableViewDataSource {

    @IBOutlet weak var exampleTableView: UITableView!
    override func viewDidLoad() {
        super.viewDidLoad()

    }

    func tableView(_ tableView: UITableView,
        numberOfRowsInSection section: Int) -> Int {
        code
    }

    func tableView(_ tableView: UITableView,
        cellForRowAt indexPath: IndexPath) ->
        UITableViewCell {
        code
    }

}
```

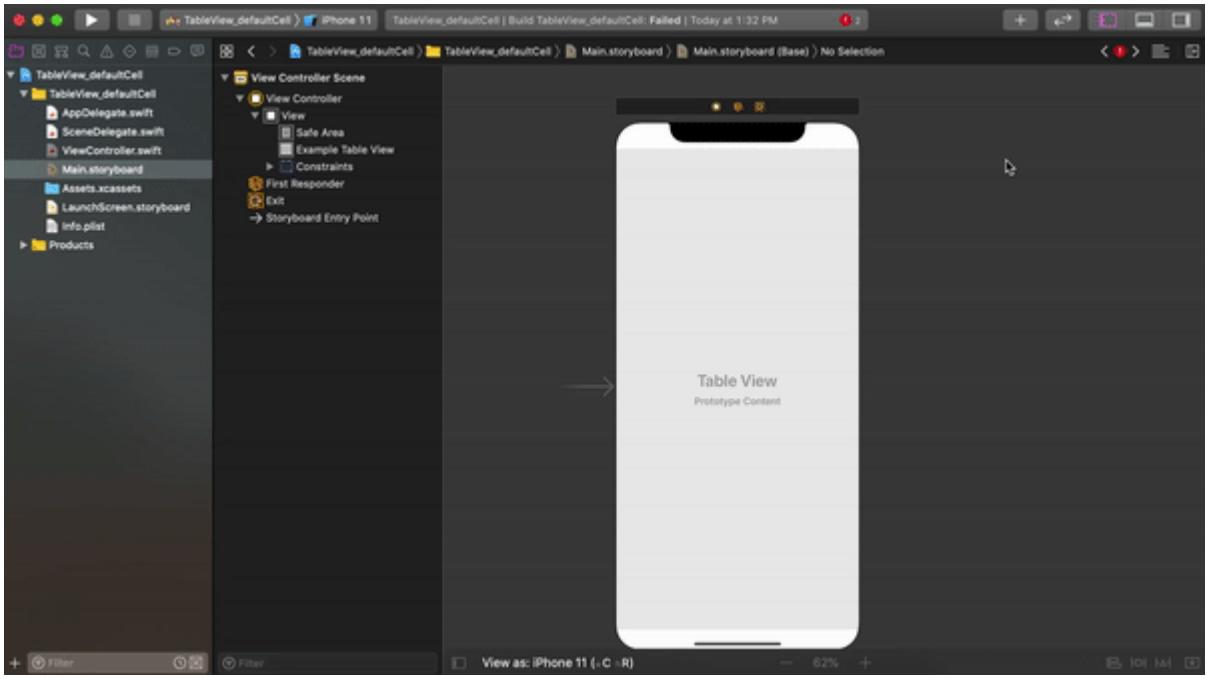
Now set delegate and data source for your TableView. You can do it by writing two lines of code into your “viewDidLoad” method. In my case those are

```
exampleTableView.delegate = self

exampleTableView.dataSource = self
```

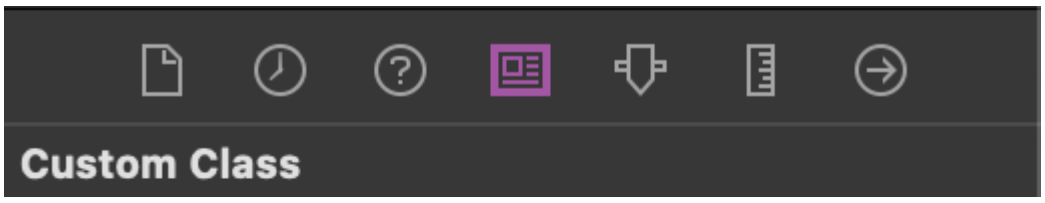
or you can set it using the storyboard like this

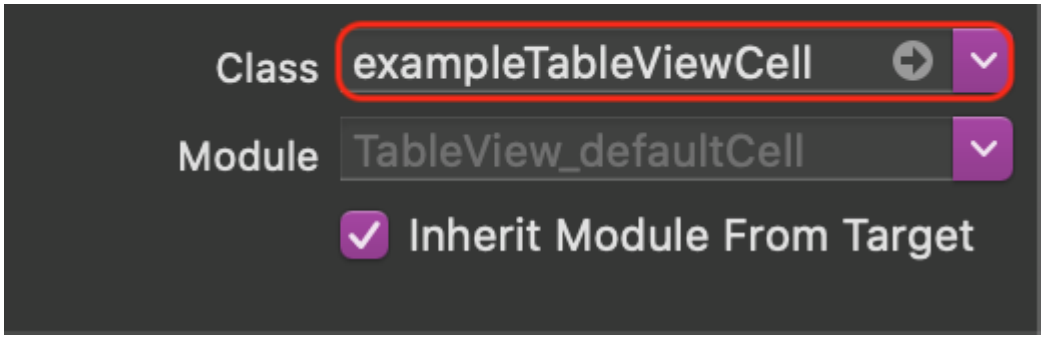
That's it. Your TableView is configured and set to show data.

Step 2: Setup TableView cell: Now drag and drop a TableView cell from the library to your table. Create a new Cocoa Touch file. Create a class subclassing "UITableViewCell".

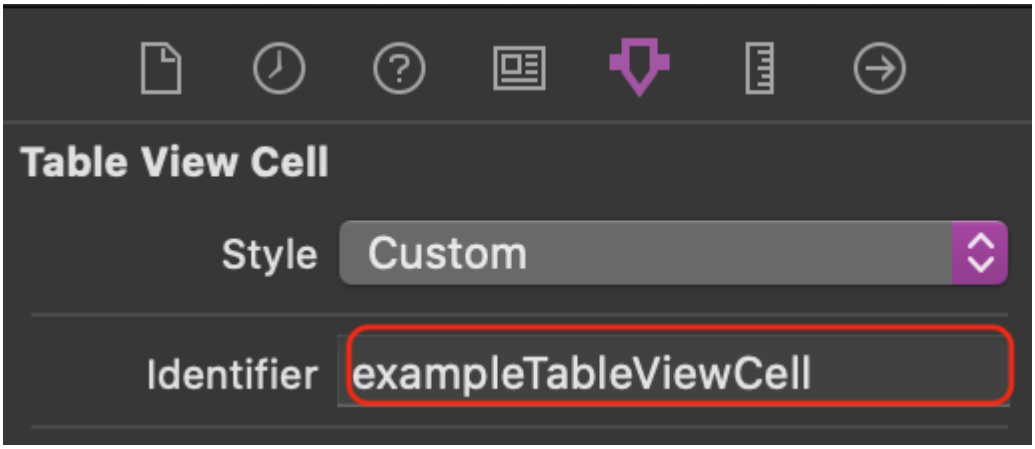


Select the TableView cell from the storyboard, set the Custom class for the cell that we just created.





Set an identifier for the cell. In my case, I named the identifier as the class name for simplicity.



That's it for your cell configuration.

Step 3: Show data: Now we will show data using TableView and its cell. We will use the following array as our data source.

```
var name = ["John", "Mike", "Adam", "Ricky", "Helen"]
```

Edit your "numberOfRowsInSection" method as following. This method actually returns the number of rows that TableView has.

```
func tableView(_ tableView: UITableView, numberOfRowsInSection
section: Int) -> Int {

return name.count

}
```

Configure your “cellForRowAt” function as following. This method will configure your every cell of the TableView using the class and identifier of the cell. We set the default label text of the TableView cell using each element of the array.

```
func tableView(_ tableView: UITableView, cellForRowAt indexPath:
IndexPath) -> UITableViewCell {

let cell = tableView.dequeueReusableCell(withIdentifier:
"exampleTableViewCell", for: indexPath) as! exampleTableViewCell

cell.textLabel?.text = name[indexPath.row]

return cell

}
```

Complete code snippet of your “ViewController” class should something like this.

```
import UIKit

class ViewController: UIViewController, UITableViewDelegate,
UITableViewDataSource {

@IBOutlet weak var exampleTableView: UITableView!

var name = ["John", "Mike", "Adam", "Ricky", "Helen"]

override func viewDidLoad() {

super.viewDidLoad()

exampleTableView.delegate = self

exampleTableView.dataSource = self

}

func tableView(_ tableView: UITableView, numberOfRowsInSection
section: Int) -> Int {

return name.count
```

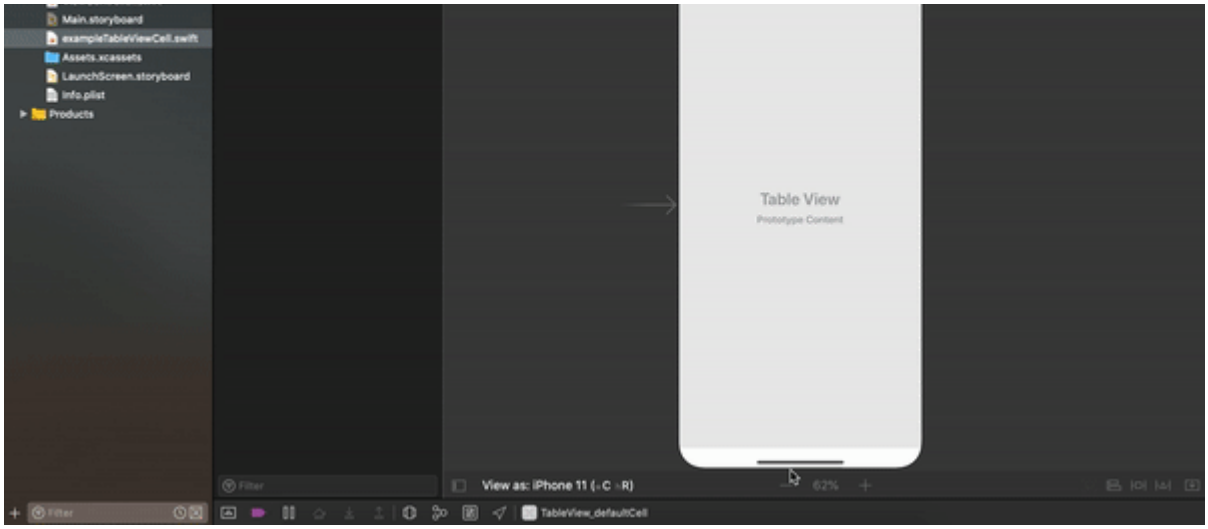
```
}

func tableView(_ tableView: UITableView, cellForRowAt indexPath:
IndexPath) -> UITableViewCell {

let cell = tableView.dequeueReusableCell(withIdentifier:
"exampleTableViewCell", for: indexPath) as! exampleTableViewCell

cell.textLabel?.text = name[indexPath.row]

return cell

}

}
```

Build and Run project on the simulator. You will definitely see something like this.

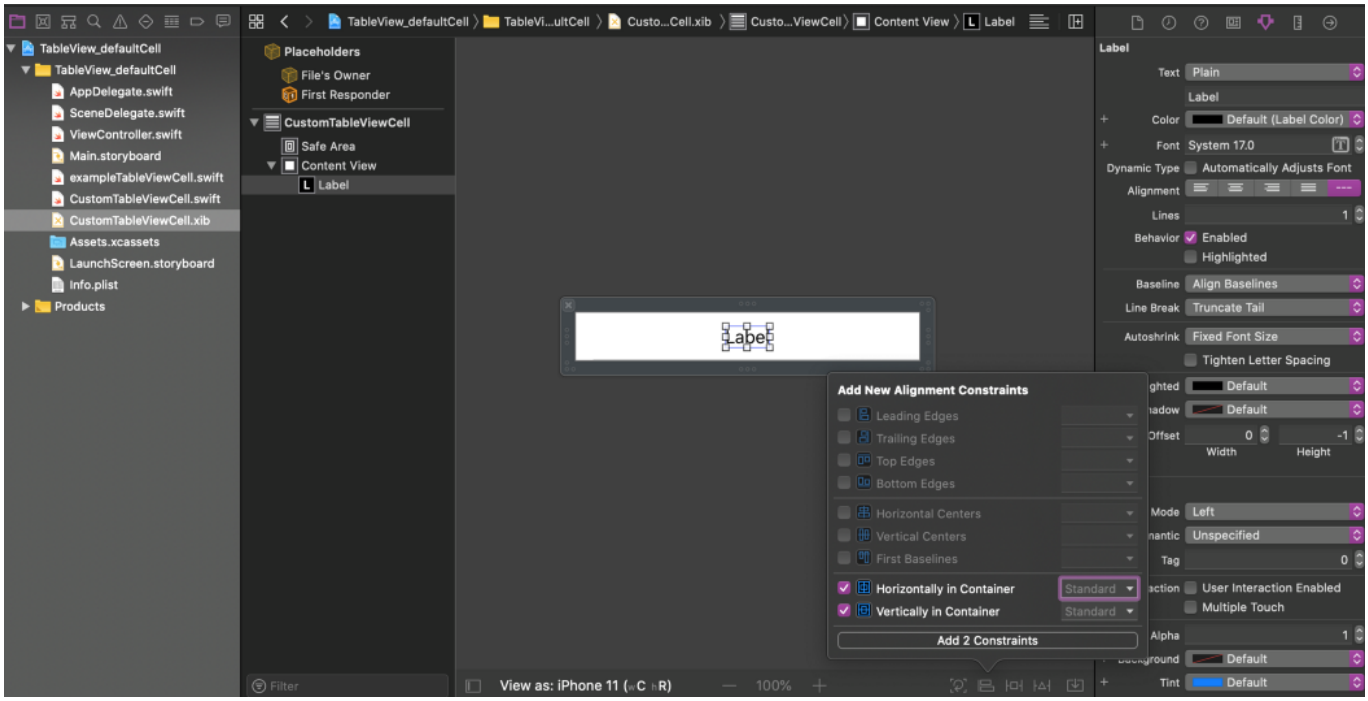
2. **Configure TableView using custom cell:** We can also create a cell using a XIB file and configure it with TableView. Let's see how can we make it.

Step 1: **Create an Xcode project & setup TableView**: We will use the same project and TableView in this tutorial. You can make a different project and TableView as I describe above.

Step 2: Setup TableView cell: Create a new Cocoa Touch file. Create a class subclassing "UITableViewCell". **Please make sure you are tick mark on the "also create XIB file" box**. Set an identifier for the cell.



Drag and drop a label into the custom cell and set constrain for the label as central horizontally and vertically in the center.



Take an outlet of this label to its cell class as nameLbl as we take the outlet of TableView above. Your cell class should have code shipped like this.

```
import UIKit

class CustomTableViewCell: UITableViewCell {

@IBOutlet weak var nameLbl: UILabel!
```

```
override func awakeFromNib() {

super.awakeFromNib()

}

override func setSelected(_ selected: Bool, animated: Bool) {

super.setSelected(selected, animated: animated)

}

}
```

We are using a XIB file as the cell for our TableView that's why we have to register cells with our TableView. We can register the cell using its name and identifier. Write this into the "viewDidLoad()" of your "ViewController" class.

```
exampleTableView.register(UINib(nibName: "CustomTableViewCell",
bundle: nil), forCellReuseIdentifier: "CustomTableViewCell")
```

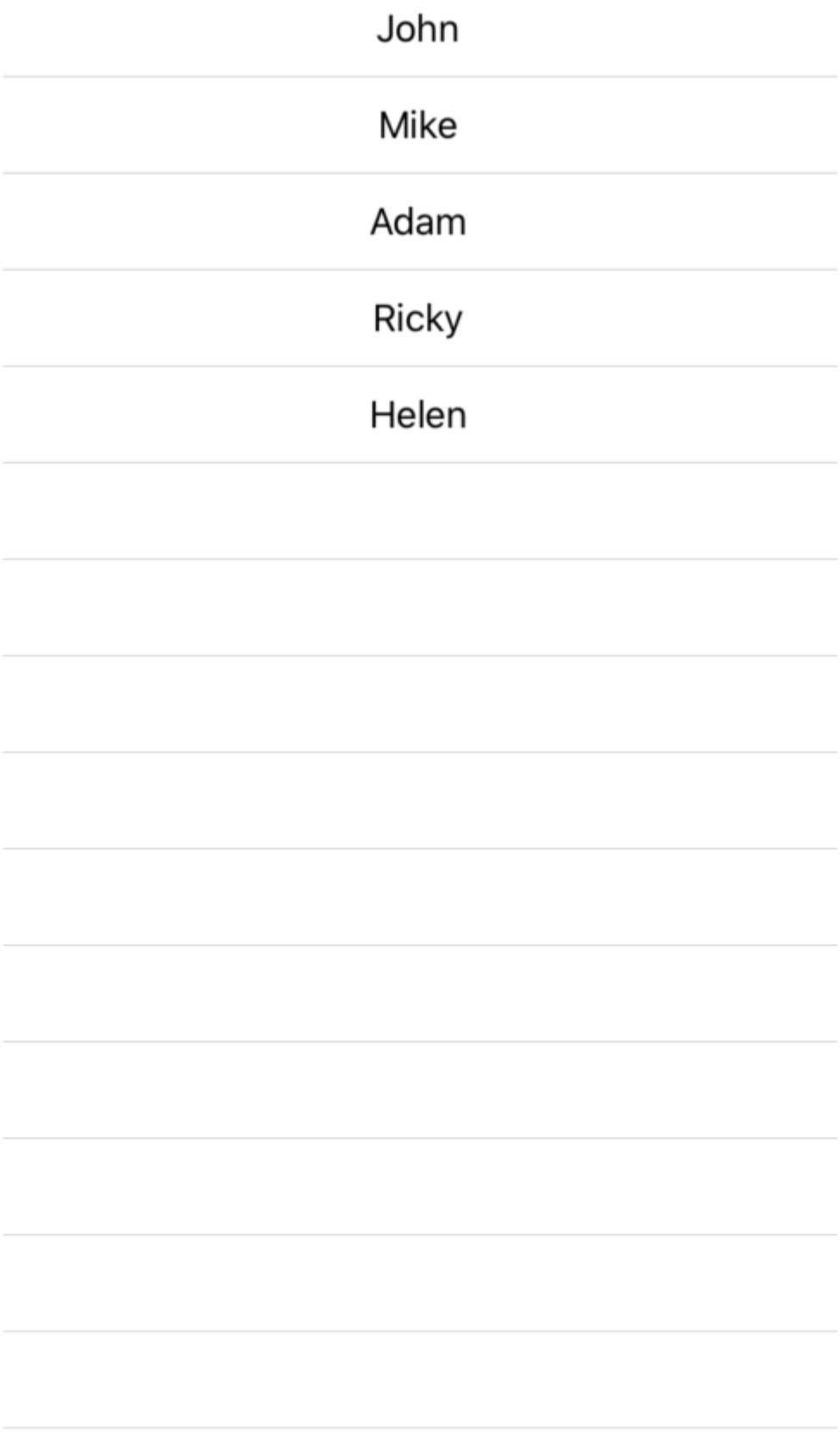
Change your TableView delegate and data source method as follow to show data using the custom cell.

```
func tableView(_ tableView: UITableView, numberOfRowsInSection
section: Int) -> Int {

return name.count

}

func tableView(_ tableView: UITableView, cellForRowAt indexPath:
IndexPath) -> UITableViewCell {

let cell = tableView.dequeueReusableCell(withIdentifier:
"CustomTableViewCell", for: indexPath) as! CustomTableViewCell

cell.nameLbl.text = name[indexPath.row]

return cell

}
```

Complete code snippet of your “ViewController” class should something like this.

```
import UIKit

class ViewController: UIViewController, UITableViewDelegate,
UITableViewDataSource {

@IBOutlet weak var exampleTableView: UITableView!

var name = ["John", "Mike", "Adam", "Ricky", "Helen"]

override func viewDidLoad() {

super.viewDidLoad()

exampleTableView.delegate = self

exampleTableView.dataSource = self

exampleTableView.register(UINib(nibName: "CustomTableViewCell",
bundle: nil), forCellReuseIdentifier: "CustomTableViewCell")

}

func tableView(_ tableView: UITableView, numberOfRowsInSection
section: Int) -> Int {

return name.count

}

func tableView(_ tableView: UITableView, cellForRowAt indexPath:
IndexPath) -> UITableViewCell {

let cell = tableView.dequeueReusableCell(withIdentifier:
"CustomTableViewCell", for: indexPath) as! CustomTableViewCell

cell.nameLbl.text = name[indexPath.row]

return cell

}

}
```

Build and Run again, you will see the data is loaded into your TableView using a custom XIB cell.



Congratulation 🎉 🎉 🎉 Now you know how to configure a TableView with a default and custom cell. You can do a lot of cool stuff using different kinds of TableView methods. Please check the Apple documentation for further classification.

If you found this article useful please share and give some clap 👏👏👏
Check my other articles on Medium and connect me on LinkedIn.

Thank you for reading & Happy coding

Swift iOS Storyboard Tableviews