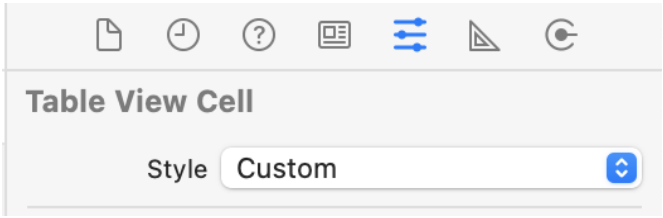
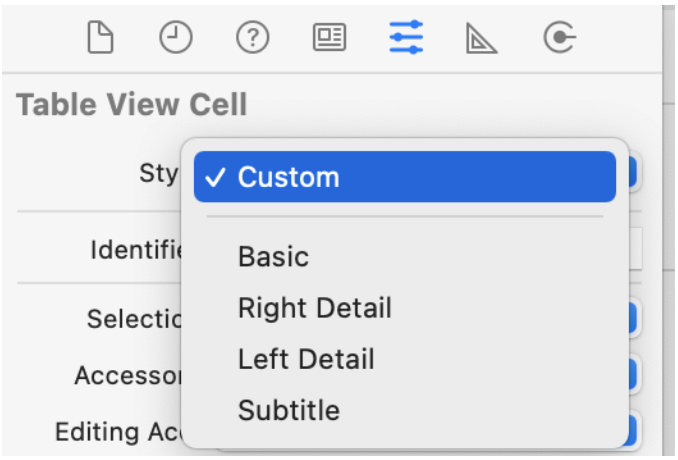
Designing a Table Row - Built in Layouts

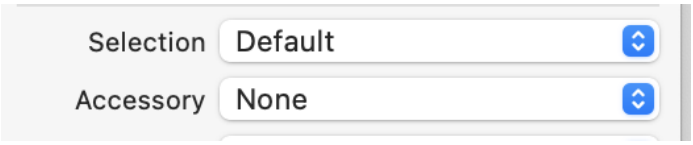
1/ Select the table view row

2/ In the attributes inspector, select the Style dropdown. Select a pre built layout from the dropdown menu.





You can also configure any selection colors and accessory icons:



Example: Using the Right Detail, Left Detail, Subtitle layouts

The Right Detail, Left Detail, and Subtitle layouts introduce a second Label to the row. The position of the second label depends on the chosen layout.

You can use this second label to add additional information.

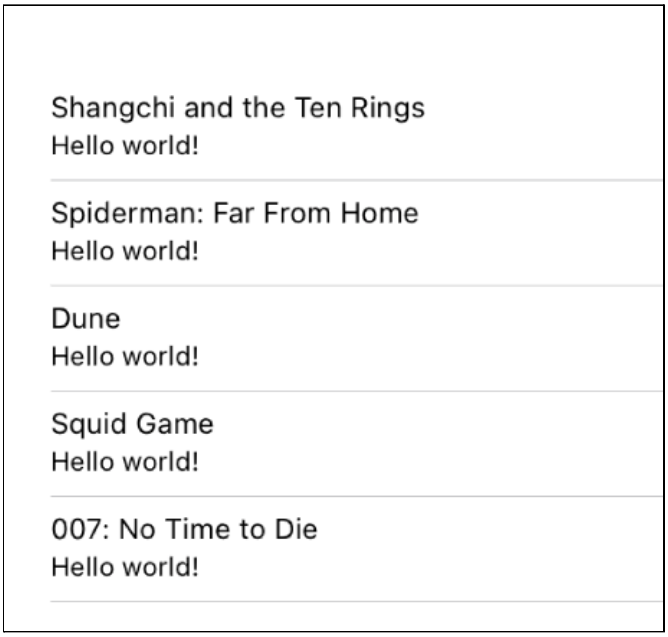
CODE SAMPLE: Specify the text for the second label in the tableView(:cellForRowAt) function. Use the cell's `detailTextLabel` property.

In sample below, "myCell" is the identifier of the table row. This information was provided in the storyboard.

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

        let cell = myTableView.dequeueReusableCell(withIdentifier:
"myCell", for: indexPath)
        cell.textLabel!.text = "Here is my first line of text"
        // @TODO: Put code here
      cell.detailTextLabel!.text = "Hello world!"

        return cell
    }
```

Expected result:

Subtitle layout



Right detail layout

Exercise: Customize the detail label to display the movie's genre

1/ Add another data source for the movie genres:

var moviesList = ["Shangchi and the Ten Rings", "Spiderman: Far From Home", "Dune", "Squid Game", "Love in the Time of Cholera"]
var genresList = ["Action Hero Movie", "Action Hero Movie", "Sci Fi", "Drama/Suspense", "Historical Drama"]

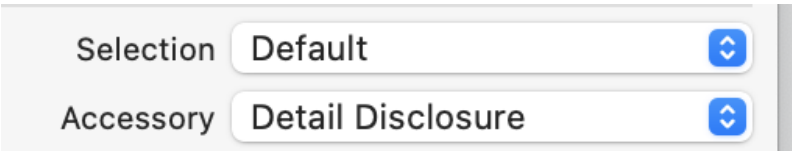
2/ In the cellForRowAt, use the genresList to output the movie genre

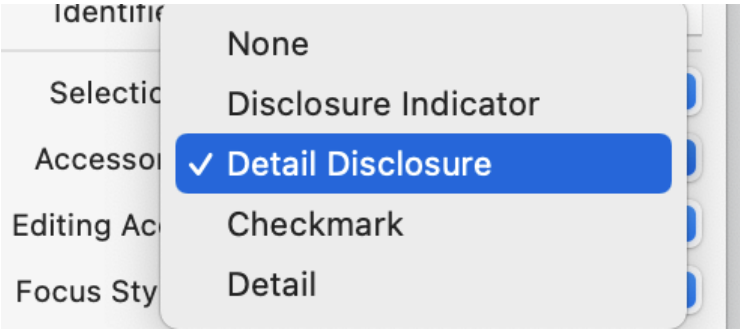
```
func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath)
-> UITableViewCell {

        // boilerplate code
        let cell = myTableView.dequeueReusableCell(withIdentifier: "myCell",
for: indexPath)

        // indexPath.row = the position of the row in the tableview
        // that is currently being rendered on the screen
        print("Drawing row #\(indexPath.row)")
        // Add whatever text / content you want to display in the row
        //cell.textLabel!.text = "Hello World!"
        cell.textLabel!.text = moviesList[indexPath.row]
        // @TODO: Put code here
        // cell.detailTextLabel!.text = "Hello world!"
        cell.detailTextLabel!.text = genresList[indexPath.row]
        return cell
    }
```

Shangchi and the Ten Rings	Action Hero Movie
Spiderman: Far From Home	Action Hero Movie
Dune	Sci Fi
Squid Game	Drama/Suspense
Love in the Time of Cholera	Historical Drama

Adding a row icon

You can add a row icon using the *Accessory* dropdown

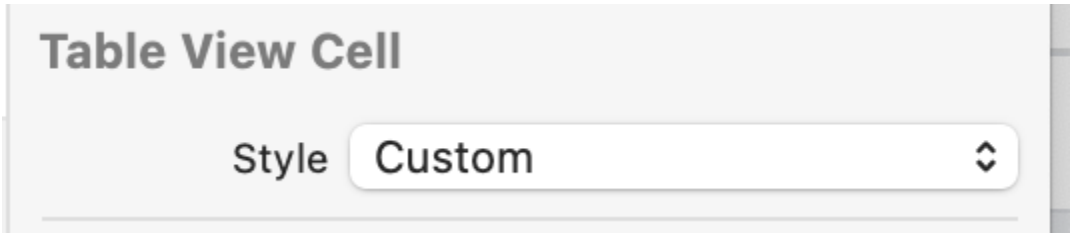




Expected result:

Designing a Row - Custom Design

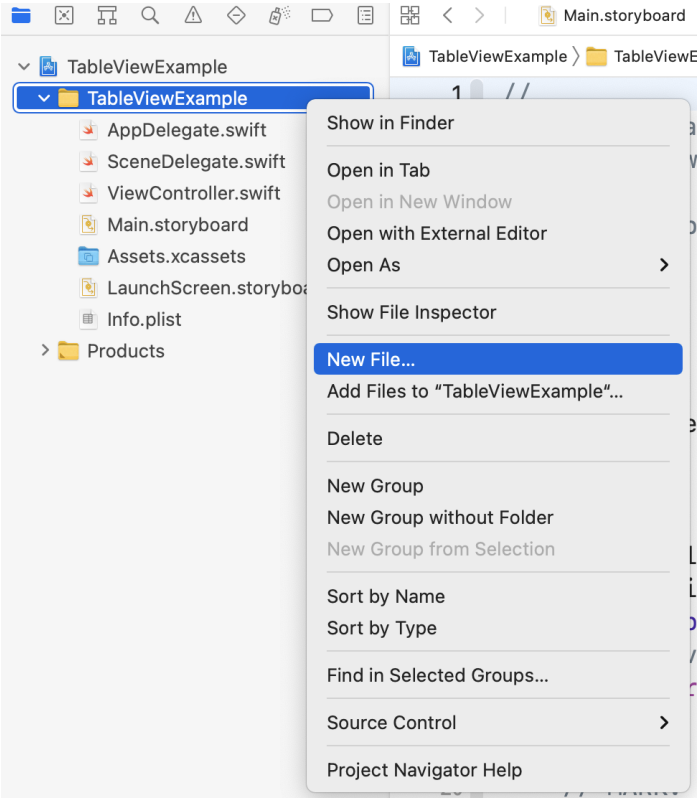
1/ In the storyboard, set the cell's style to Custom

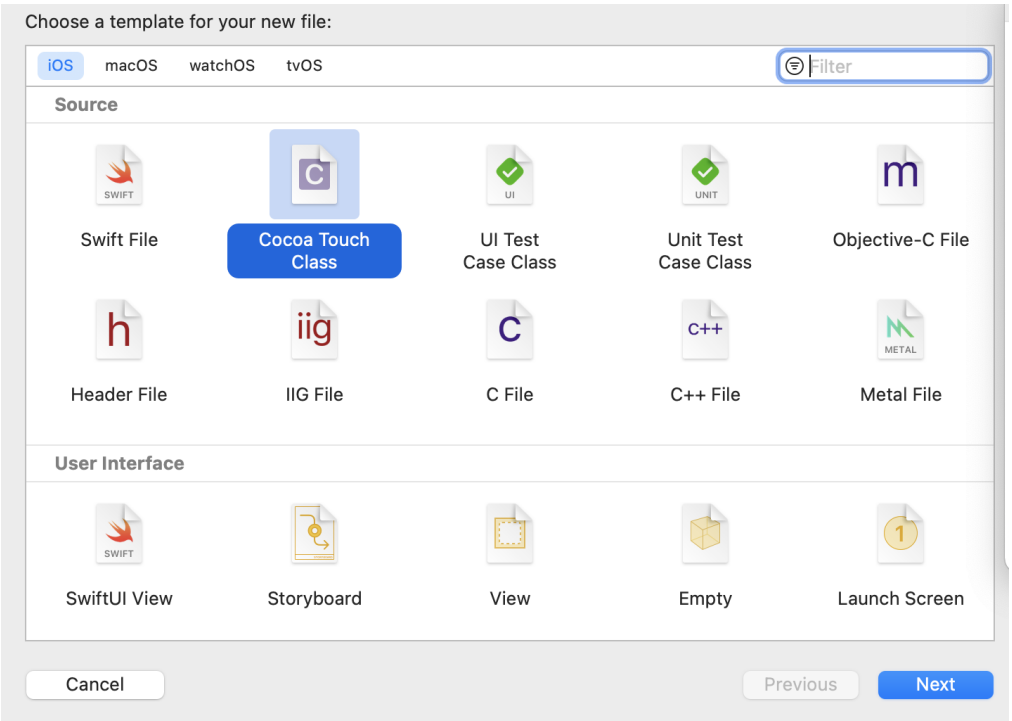


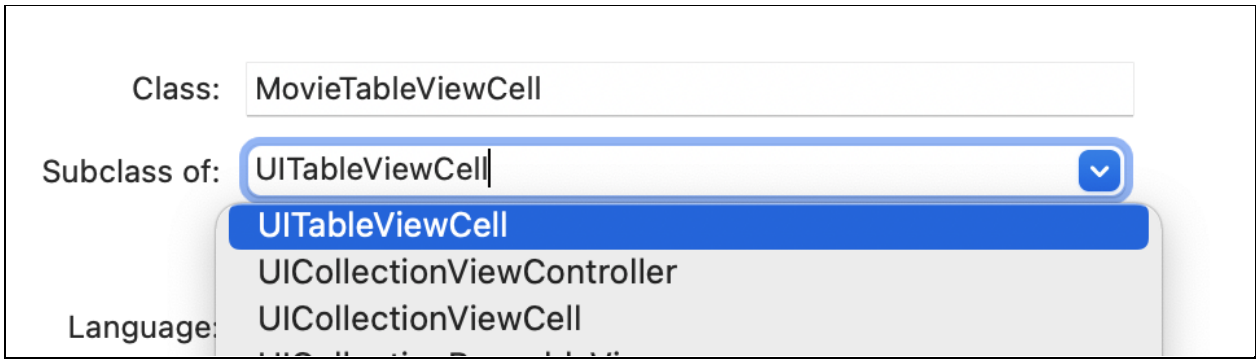
2/ Drag and drop ui elements into the cell

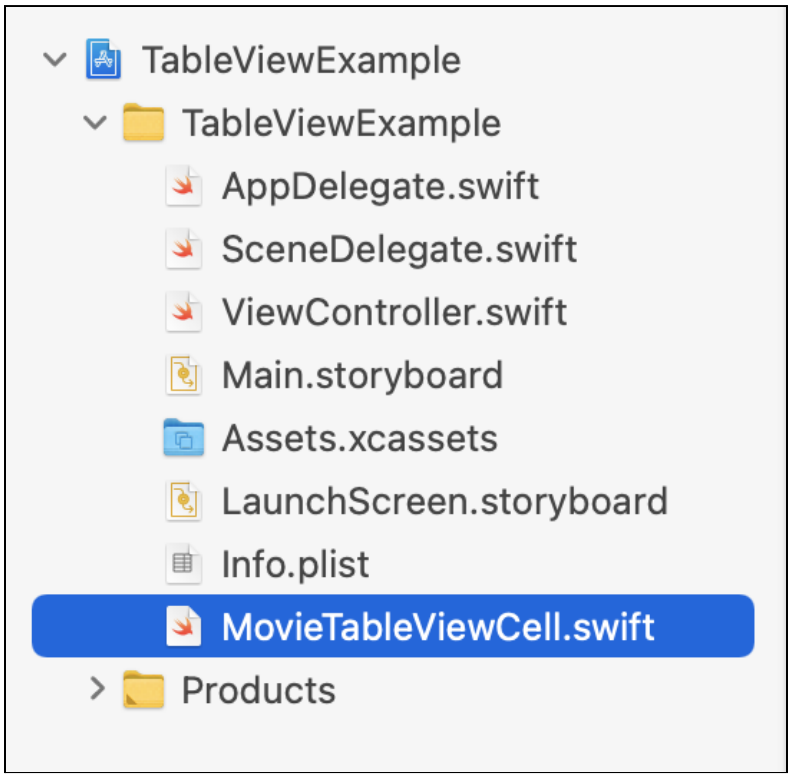
3/ Create a new Swift file that models the Cell

- Swift file should extend UITTableViewCell

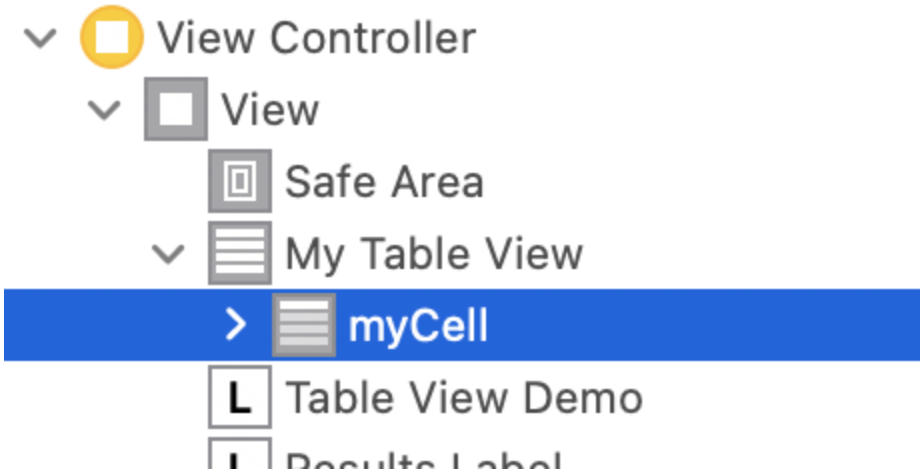

Main.storyboard
TableViewExample
TableViewExample
AppDelegate.swift
SceneDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboa
Info.plist
Products
Show in Finder
Open in Tab
Open in New Window
Open with External Editor
Open As
Show File Inspector
New File...
Add Files to "TableViewExample"...
Delete
New Group
New Group without Folder
New Group from Selection
Sort by Name
Sort by Type
Find in Selected Groups...
Source Control
Project Navigator Help

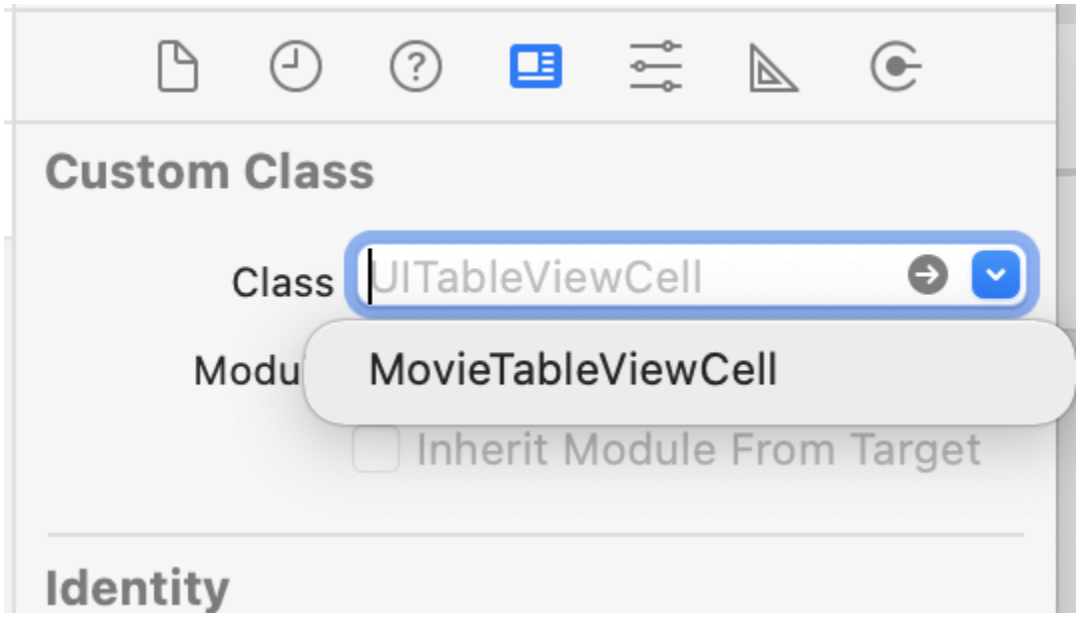

Choose a template for your new file:
iOS
macOS
watchOS
tvOS
Filter
Source
Swift File
Cocoa Touch Class
UI Test Case Class
Unit Test Case Class
Objective-C File
Header File
IIG File
C File
C++ File
Metal File
User Interface
SwiftUI View
Storyboard
View
Empty
Launch Screen
Cancel
Previous
Next

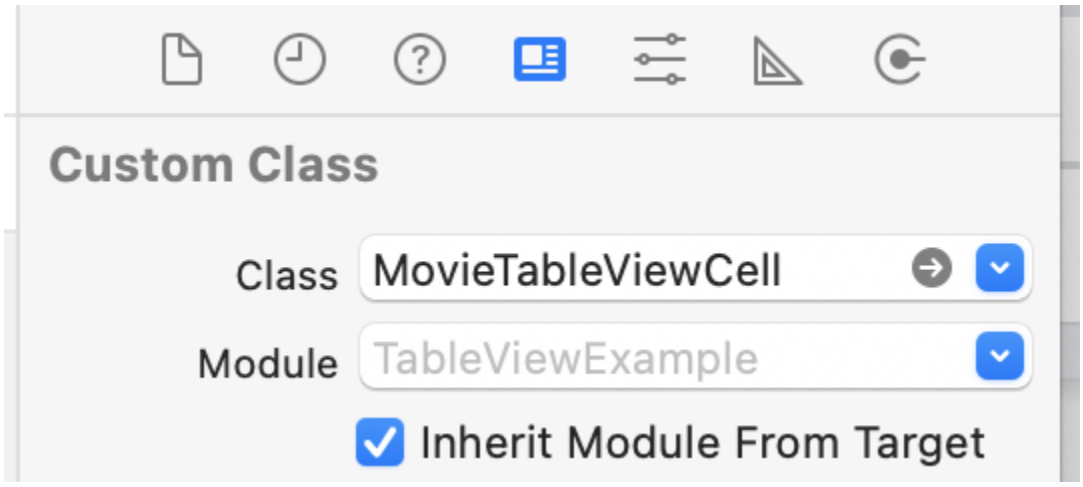




Associate this custom cell class with the cell in the storyboard

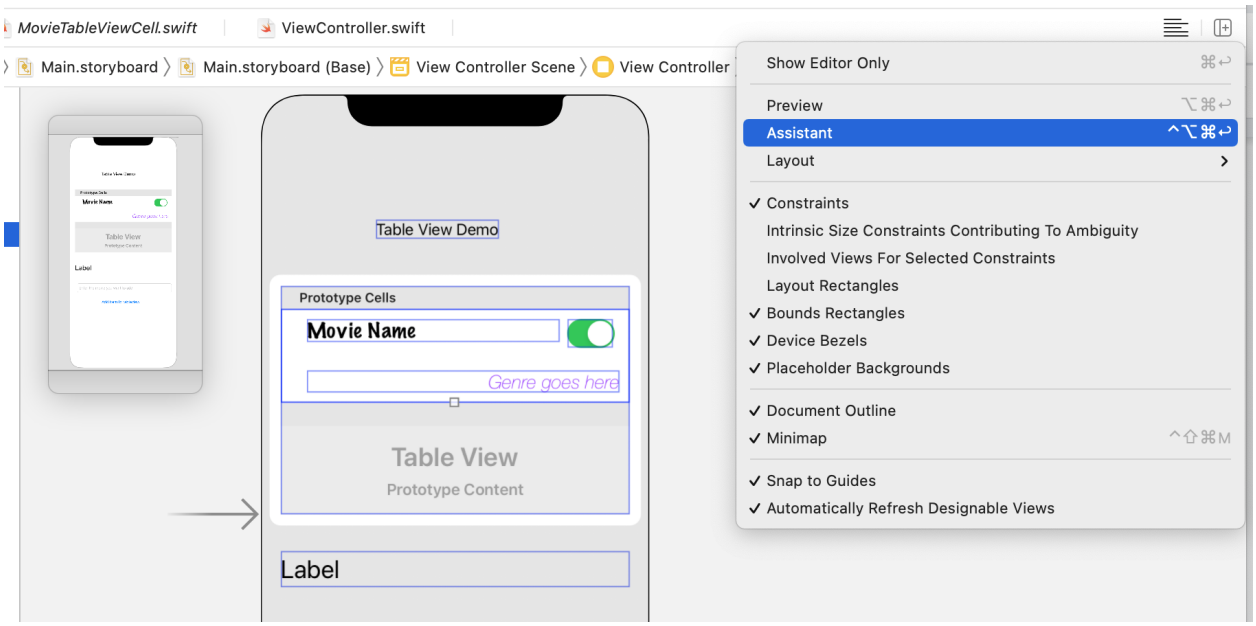

View Controller
View
Safe Area
My Table View
myCell
Table View Demo

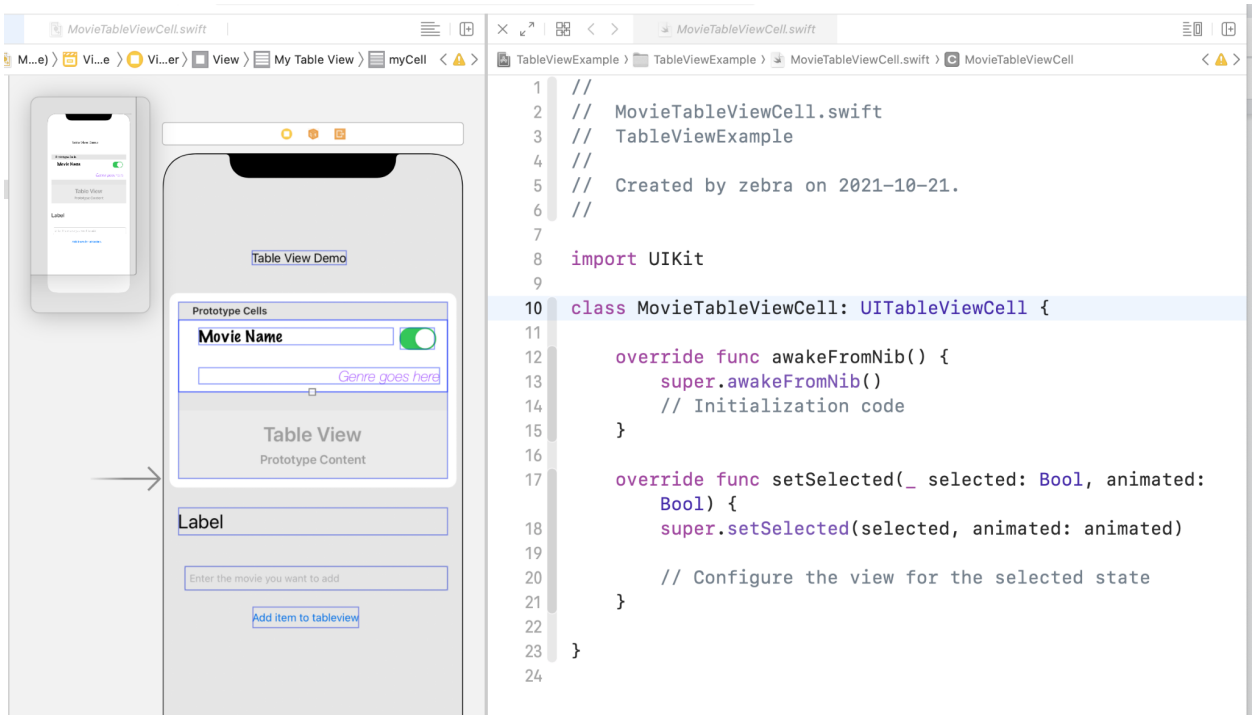

Custom Class
Class
UITableViewCell
Modu
MovieTableViewCell
Inherit Module From Target
Identity


Custom Class
Class
MovieTableViewCell
Module
TableViewExample
Inherit Module From Target

4/ In the assistant editor, create outlets for the cell's UI elements

- Add the outlets into the UITableViewCell Swift file you created in previous step





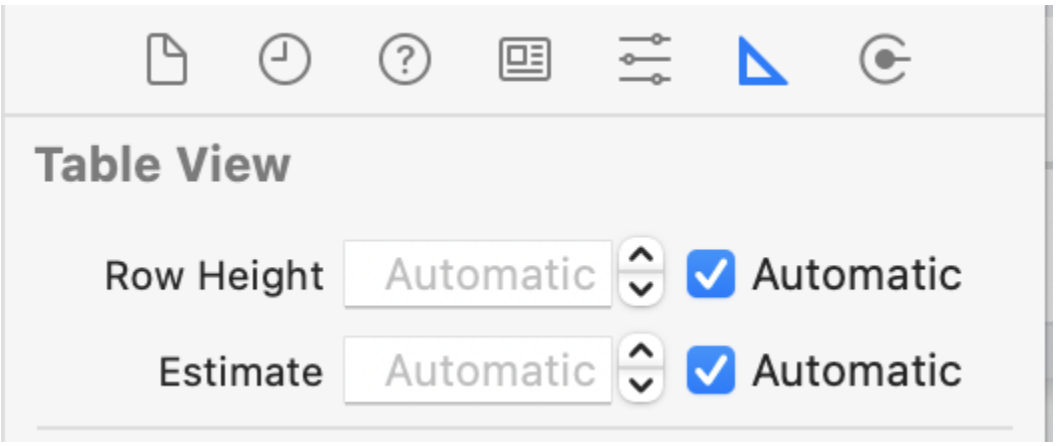
Code:

```
import UIKit

class MovieTableViewCell: UITableViewCell {
    // Outlets
    @IBOutlet weak var movieTitleLabel: UILabel!
    @IBOutlet weak var movieGenreLabel: UILabel!

    override func awakeFromNib() {
        super.awakeFromNib()
        // Initialization code
    }

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

        // Configure the view for the selected state
    }

}
```

ViewController.swift -> tableView(:cellForRowAt)

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

    // boilerplate code

    // 1. cast this cell to be of type MovieTableViewCell
    // - This is our custom class we just created
    let cell = myTableView.dequeueReusableCell(withIdentifier: "myCell", for: indexPath) as! MovieTableViewCell

    // 2. We use the cell to access the outlets in MovieTableViewCell
    cell.movieTitleLabel.text = moviesList[indexPath.row]
    cell.movieGenreLabel.text = genresList[indexPath.row]

    return cell
}
```

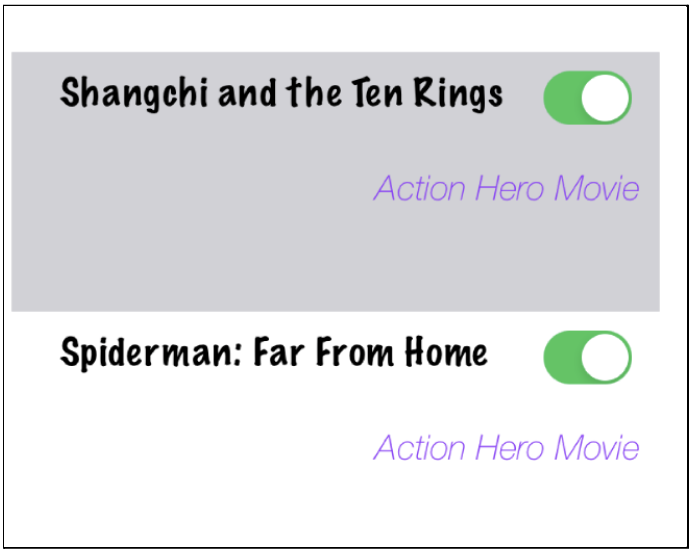
Adjust height of rows

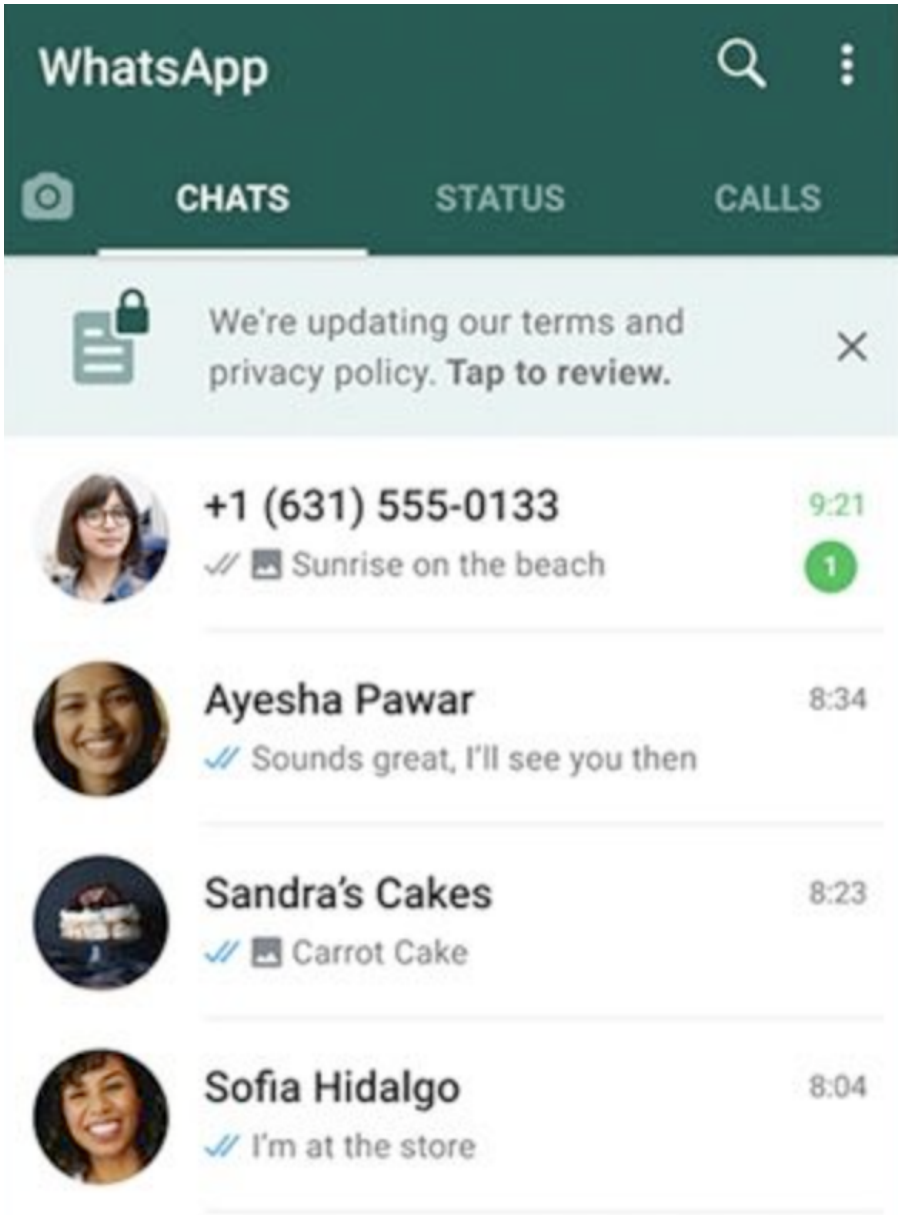


Set the row height to a value that will allow you to see the entire row

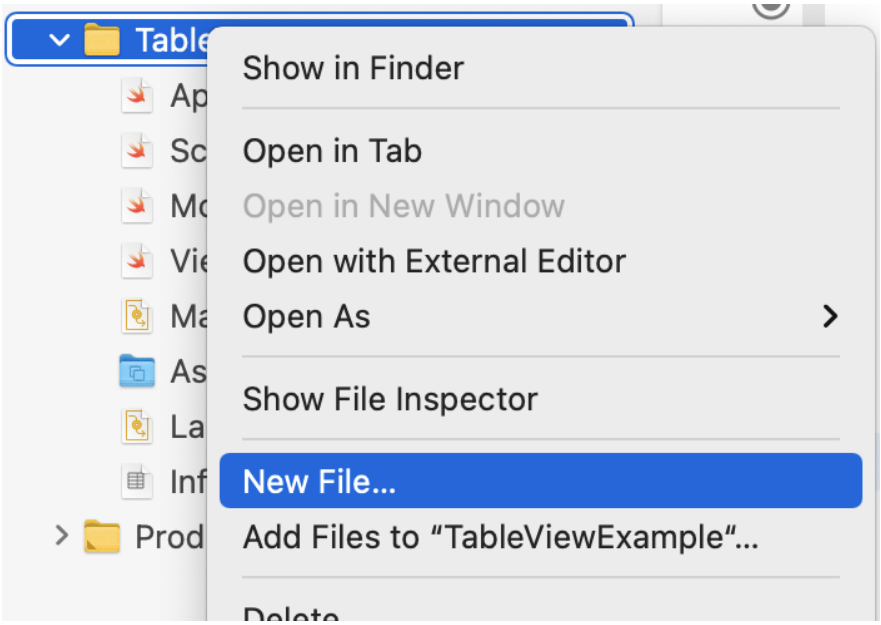
- Example: 120, 200, 300
- You may have to try several values before you find the correct height

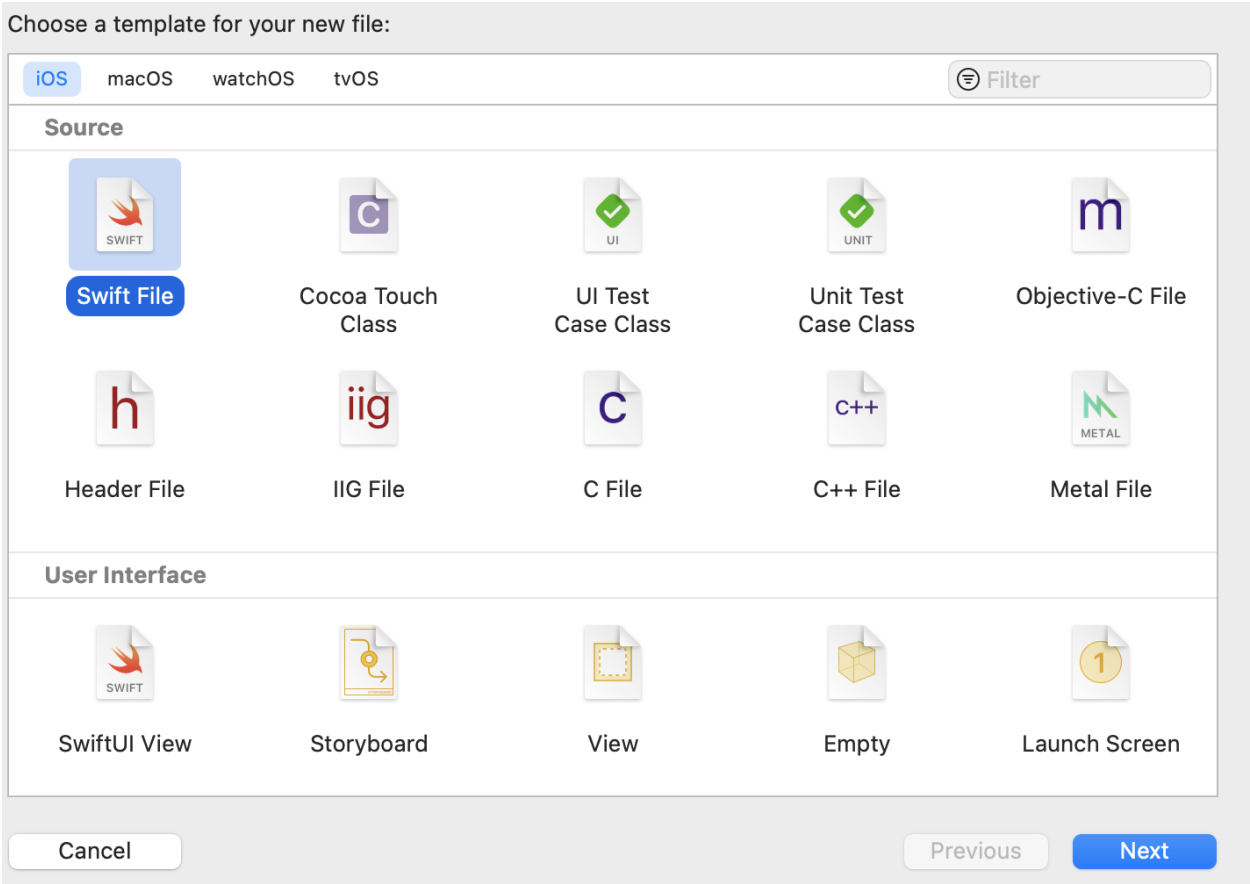
For my emulator, a row height of 150 produces this result:

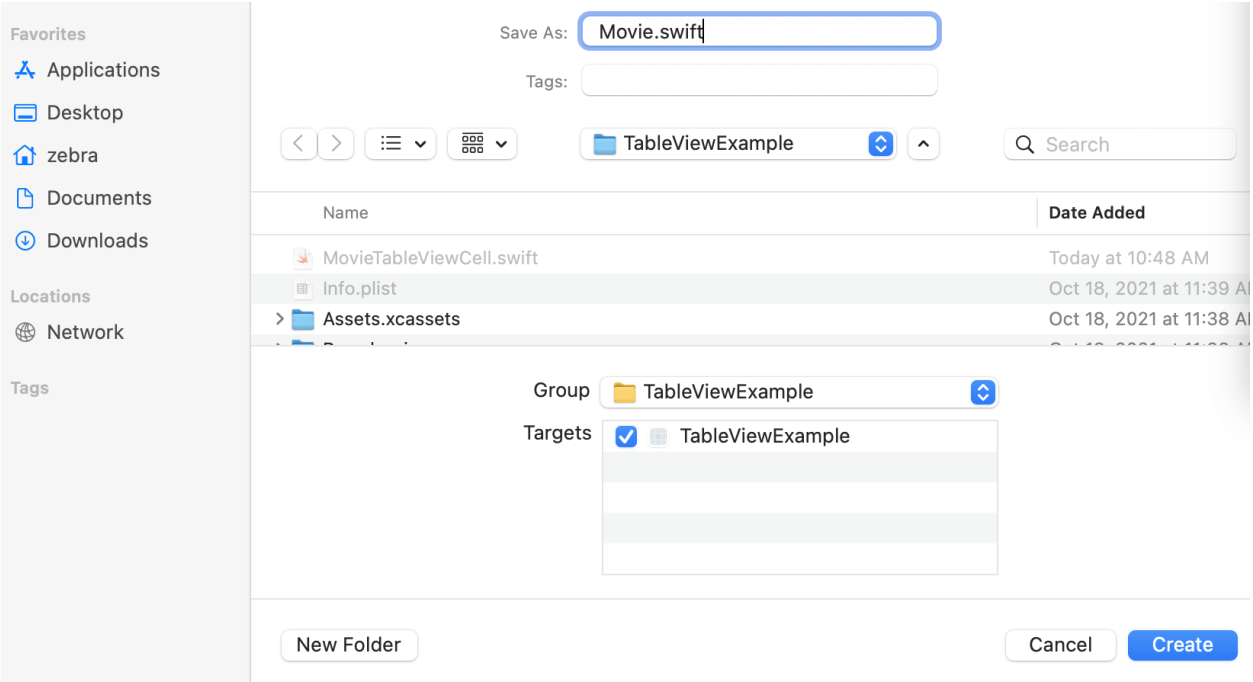


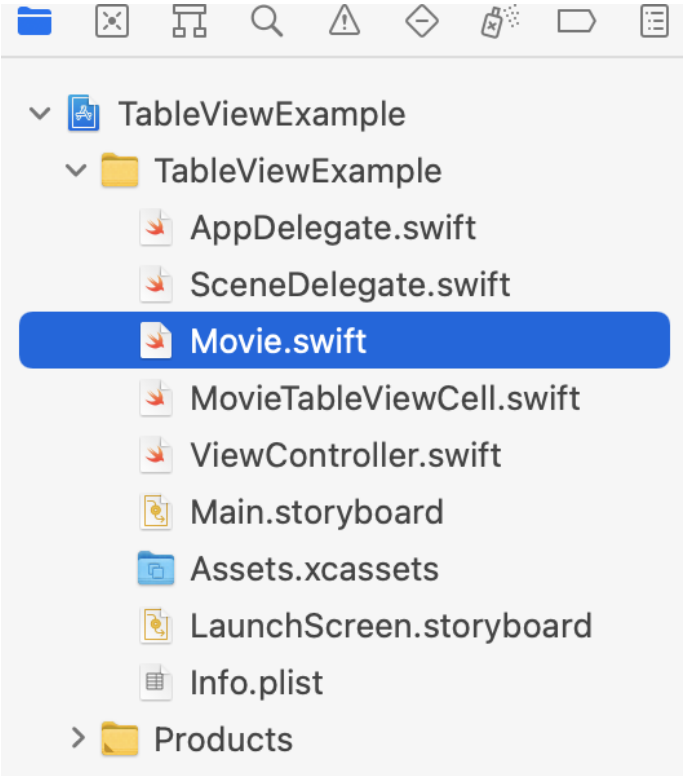

WhatsApp
CHATS
STATUS
CALLS
We're updating our terms and privacy policy. Tap to review.
+1 (631) 555-0133
9:21
Sunrise on the beach
1
Ayesha Pawar
8:34
Sounds great, I'll see you then
Sandra's Cakes
8:23
Carrot Cake
Sofia Hidalgo
8:04
I'm at the store

1. Create a new swift file called Movie.swift






Favorites
Applications
Desktop
zebra
Documents
Downloads
Locations
Network
Tags
Save As:
Movie.swift
Tags:
TableViewExample
Search
Name
Date Added
MovieTableViewCell.swift
Today at 10:48 AM
Info.plist
Assets.xcassets
Group
TableViewExample
Targets
TableViewExample
New Folder
Cancel
Create


TableViewExample
TableViewExample
AppDelegate.swift
SceneDelegate.swift
Movie.swift
MovieTableViewCell.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
Products

2/ Add the properties for a movie

```
class Movie {
    // properties
    let name:String
    let genre:String
    let runningTime:Int
    let year:Int

    // initializer
    init(name:String, genre:String, runningTime:Int, releaseYear:Int) {
        self.name = name
        self.genre = genre
        self.runningTime = runningTime
        self.year = releaseYear
    }
}
```

3/ View Controller

Create a list of movie objects

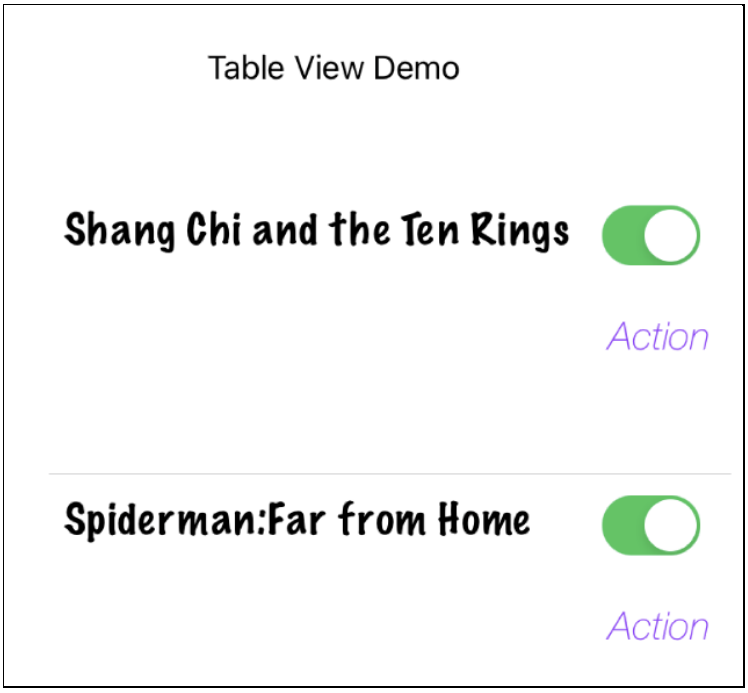
```
var moviesList:[Movie] = [
        Movie(name:"Shang Chi and the Ten Rings", genre: "Action",
runningTime: 120, releaseYear: 2021),
        Movie(name:"Spiderman:Far from Home", genre: "Action", runningTime:
180, releaseYear: 2022),
        Movie(name:"Dune", genre: "Sci Fi", runningTime: 100, releaseYear:
2021),
        Movie(name:"Love In the Time of Cholera", genre: "Historical
Fiction", runningTime: 90, releaseYear: 2010)
    ]
```

Update all of the functions

- Display of movie

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

        // boilerplate code

        // 1. cast this cell to be of type MovieTableViewCell
        // - This is our custom class we just created
        let cell = myTableView.dequeueReusableCell(withIdentifier: "myCell", for:
indexPath) as! MovieTableViewCell

        // 2. We use the cell to access the outlets in MovieTableViewCell
        let currMovie:Movie = moviesList[indexPath.row]
        cell.movieTitleLabel.text = currMovie.name
        cell.movieGenreLabel.text = currMovie.genre

        // option 2
       //  cell.movieTitleLabel.text = moviesList[indexPath.row].name
       //  cell.movieGenreLabel.text = moviesList[indexPath.row].genre

        return cell
    }
```

Expected result:



- Add movie

```
@IBAction func btnAddPressed(_ sender: Any) {
        // 1. Get the content the user entered in the textbox
        let movieToAdd = tbMovieName.text!

        // 2. Add that item to the datasource (moviesList)
        let m:Movie = Movie(name: movieToAdd, genre: "Unknown", runningTime: -1,
releaseYear: 2099)
        moviesList.append(m)
        print(moviesList)
        // 3. Update the user interface!!!!
        // 3a. Update the tableview
        myTableView.reloadData()

        // 3b. Clear the textbox and wait for new imput
        tbMovieName.text = ""
    }
```

- Delete movie

No changes need

- Select a movie

```
func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath)
{
        // indexPath will return us the row that is currently being clicked
on
        print("You clicked on a row: \(indexPath.row)")
        // moviesList[indexPath.row] is of data type Movie
        // so use the movie properties to populate the label
        resultsLabel.text = moviesList[indexPath.row].name
    }
```