

FIAP GRADUAÇÃO

Desenvolvimento Avançado iOS

Prof. Gustavo Calixto

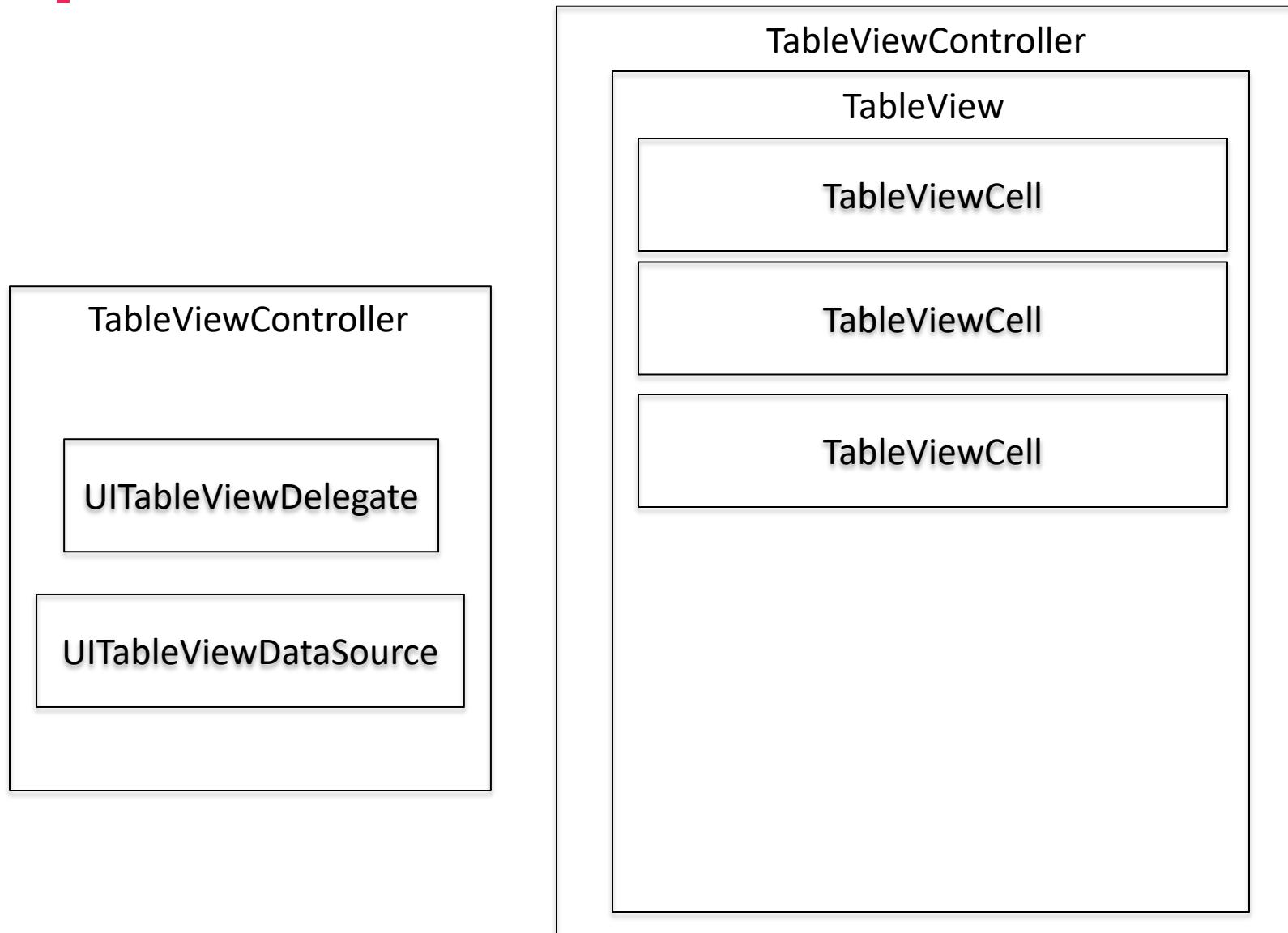
AULA DE HOJE

- Entendimento sobre o uso do
 - UITableViewController
 - UITableView
 - UITableViewCell
 - Navigation Controller

TABLE VIEW

- Uma TableView é um dos componentes mais utilizados no iOS, no qual permite empilhar widgets pré-definidos ou definidos pelo usuário.
- A melhor maneira de gerenciar um TableView é através do uso de uma UITableViewController
- A UITableViewController já herda os componentes necessários para controlar a tabela:
 - UITableViewDelegate
 - UITableViewDataSource

TABLE VIEW



| TABLE VIEW

- Vamos criar um projeto para entender como criar uma TableView e definir uma TableViewCell.

TABLE VIEW

- Crie um projeto em iOS, e arraste um UITableViewController da caixa de controles para a Storyboard.

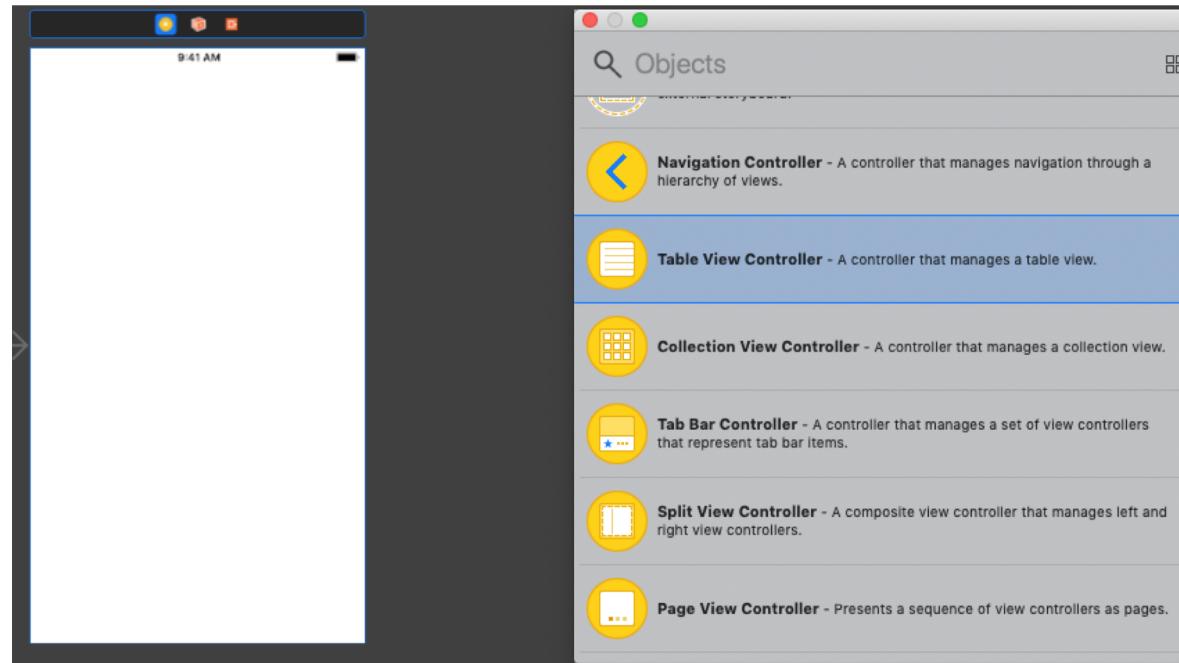


TABLE VIEW

- Componentes da UITableViewController

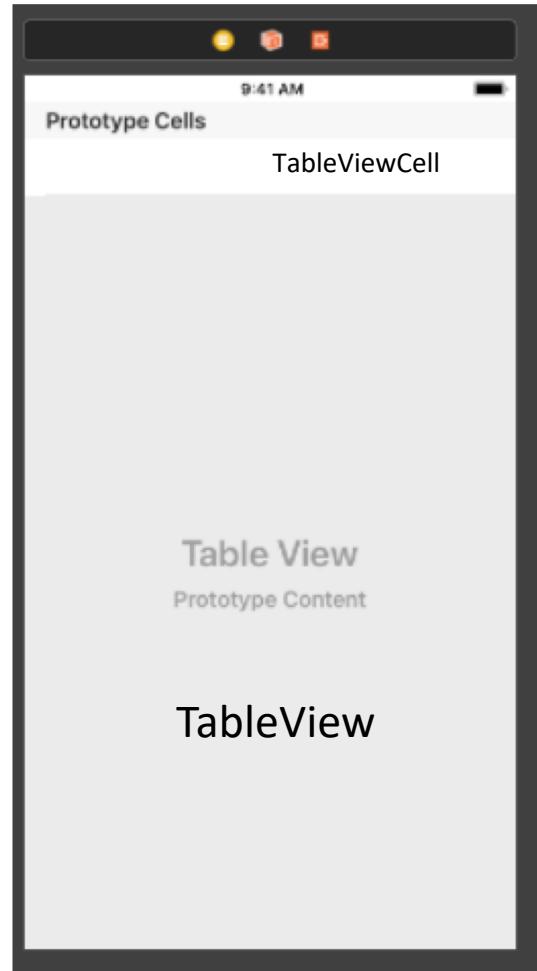
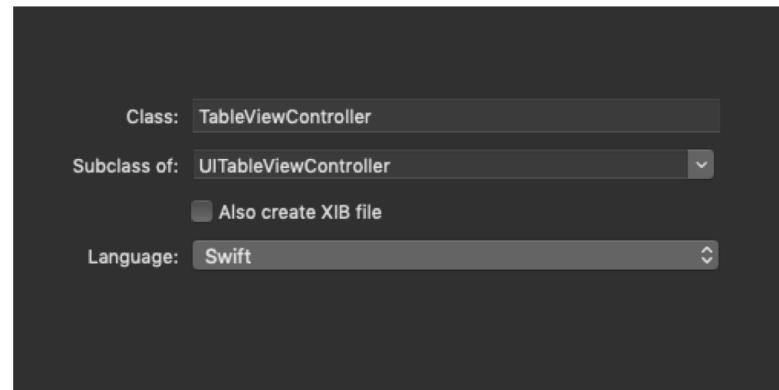


TABLE VIEW

- O próximo passo é definir uma classe customizada para a UITableViewController em NewFile -> CocoaTouchClass -> Subclass of UITableViewController.



```
import UIKit

class TableViewController: UITableViewController {
```

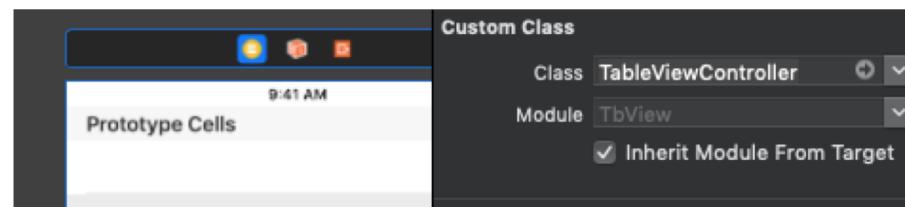


TABLE VIEW

- Depois, selecione a célula base na TableView (TableViewCell) e configure um Identifier para a mesma (será usado futuramente para configurar a impressão das células).
- No Attributes Inspector, configure o TableViewCell para atuar no estilo básico.

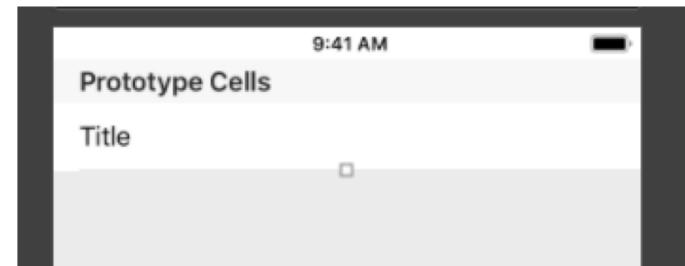
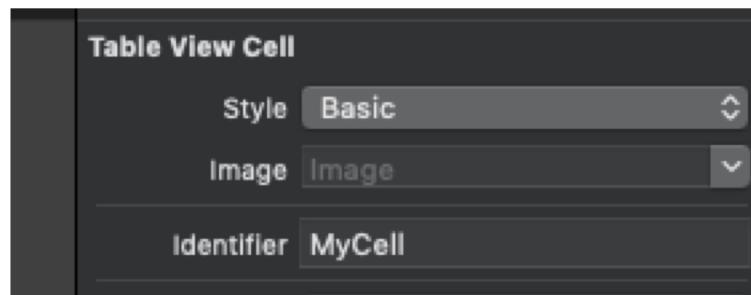


TABLE VIEW

- Com isso, agora é necessário definir.
 - A fonte de dados para a tabela (DataSource)
 - Quantas Sessões a tabela terá (no nosso caso, uma sessão).
 - O número de linhas será definido pelo número de elementos definidos no DataSource.
 - Como será a configuração de cada célula gerada.
 - Para isso, a classe customizada de TableViewController obriga a implementação de métodos que respondem às definições necessárias.

TABLE VIEW

- Definição da fonte de dados
 - Vamos criar um array de Strings com valores Fictícios para alimentar a tabela

```
class TableViewController: UITableViewController {

    var lista:[String] = []

    override func viewDidLoad() {
        super.viewDidLoad()

        //inicializando o array com valores de teste

        lista.append("Janeiro");
        lista.append("Fevereiro");
        lista.append("Março");
        lista.append("Abril");
        lista.append("Maio");
        lista.append("Junho");
        lista.append("Julho");
        lista.append("Agosto");
        lista.append("Setembro");
        lista.append("Outubro");
        lista.append("Novembro");
        lista.append("Dezembro");
```

TABLE VIEW

- Definição do número de sessões
 - Cada sessão define uma camada de exibição dos dados em tabela. No nosso caso será gerada somente uma sessão.

```
override func numberOfSections(in tableView: UITableView) -> Int {  
    // #warning Incomplete implementation, return the number of sections  
    return 1  
}
```

TABLE VIEW

- Definição do número de linhas
 - O número de linhas é proporcional ao gerado pelo Data Source (neste caso o vetor criado).

```
override func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) ->
    Int {
    // #warning Incomplete implementation, return the number of rows
    return lista.count
}
```

TABLE VIEW

- Configuração de cada célula gerada

```
override func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) ->  
    UITableViewCell {  
        let cell = tableView.dequeueReusableCell(withIdentifier: "MyCell", for: indexPath)  
  
        cell.textLabel?.text = lista[indexPath.row]  
  
        return cell  
    }
```

Propriedade de Label da Célula

Indexador da célula

Identificador da célula

TABLE VIEW

- Crie um Botão e uma Action Segue para transitar para a TableView.
- Faça o teste!

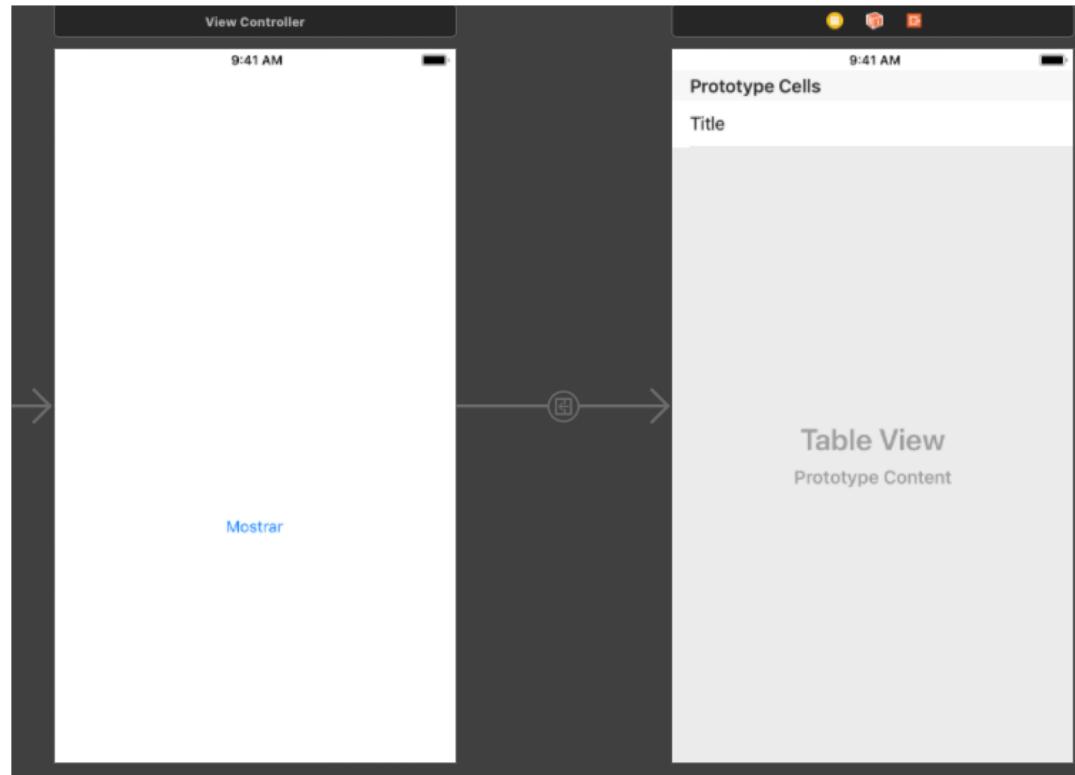


TABLE VIEW

- Resultado!

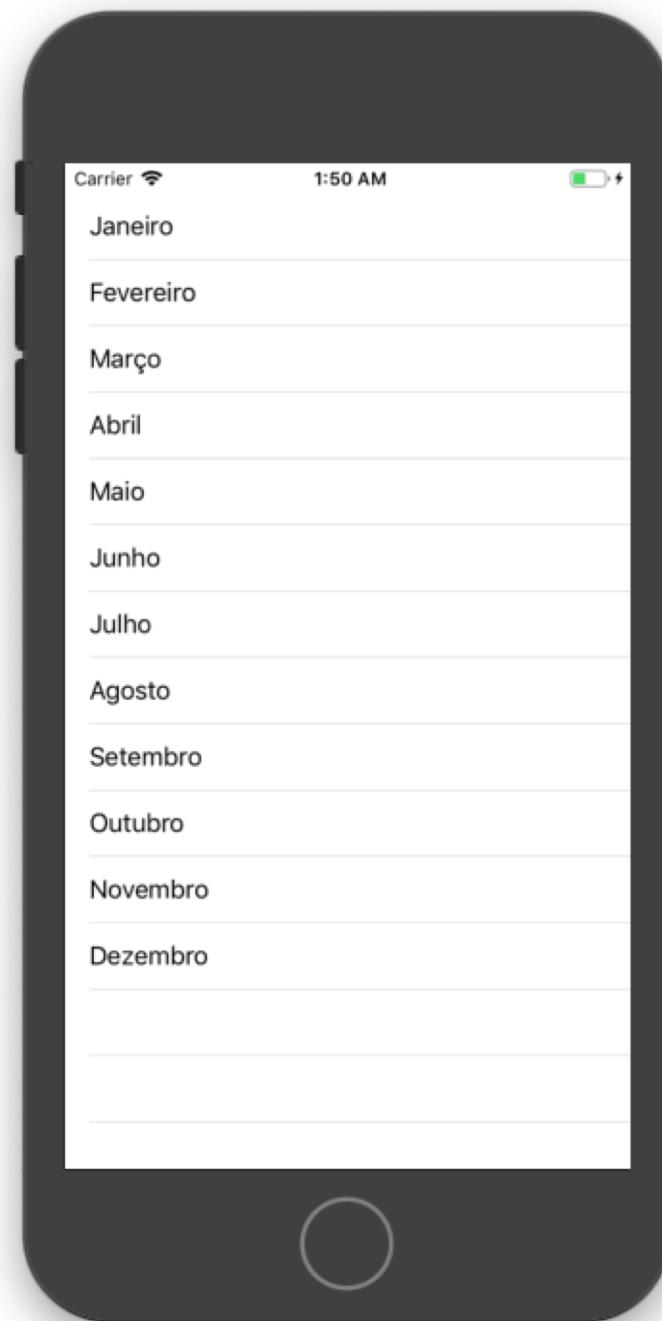
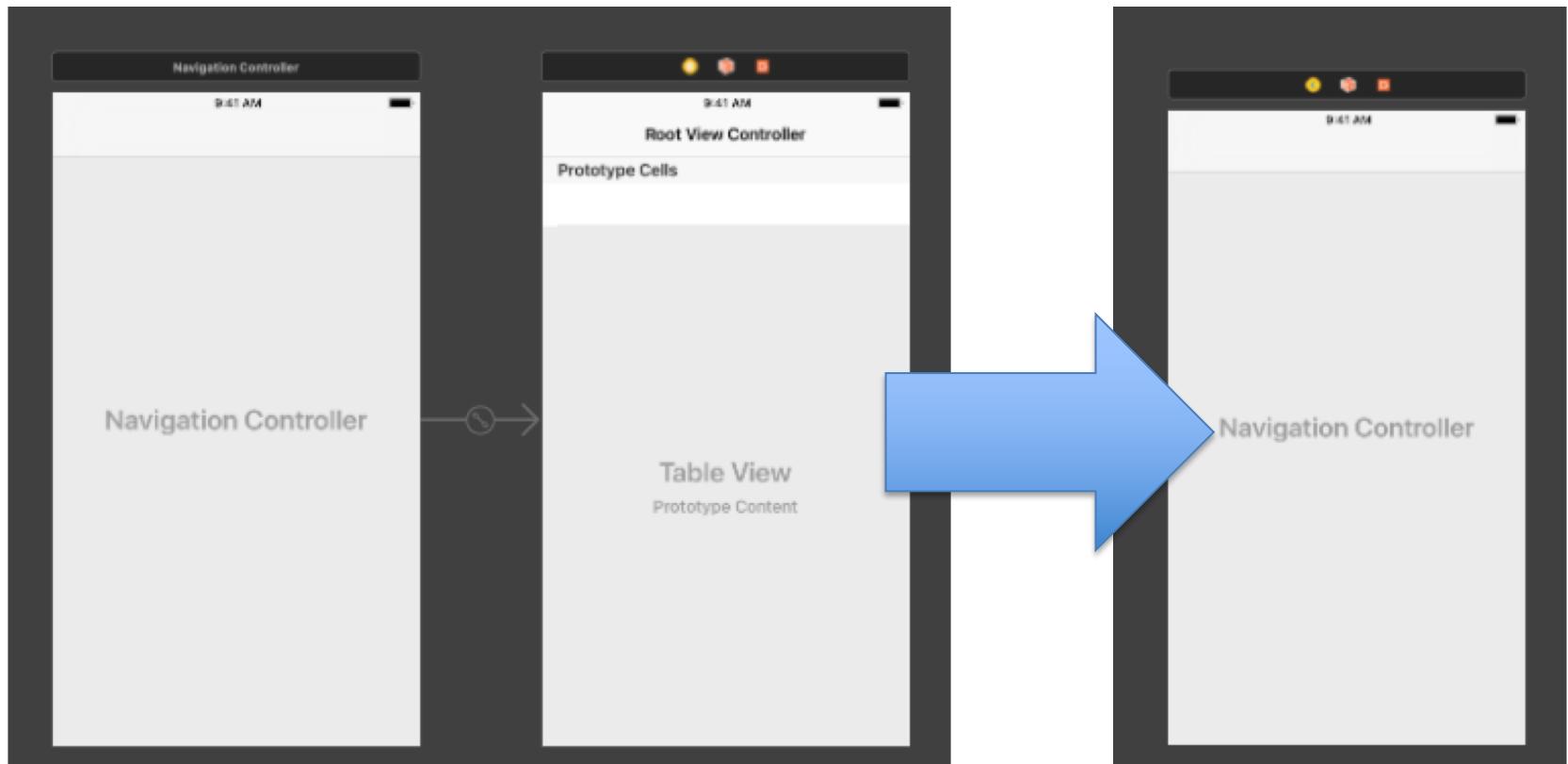


TABLE VIEW

- Vamos exercitar!
 - Explore a personalização da TableCellView adicionando.
 - Uma imagem em cada célula
 - Vamos criar um UINavigationController para permitir a navegação entre as telas.

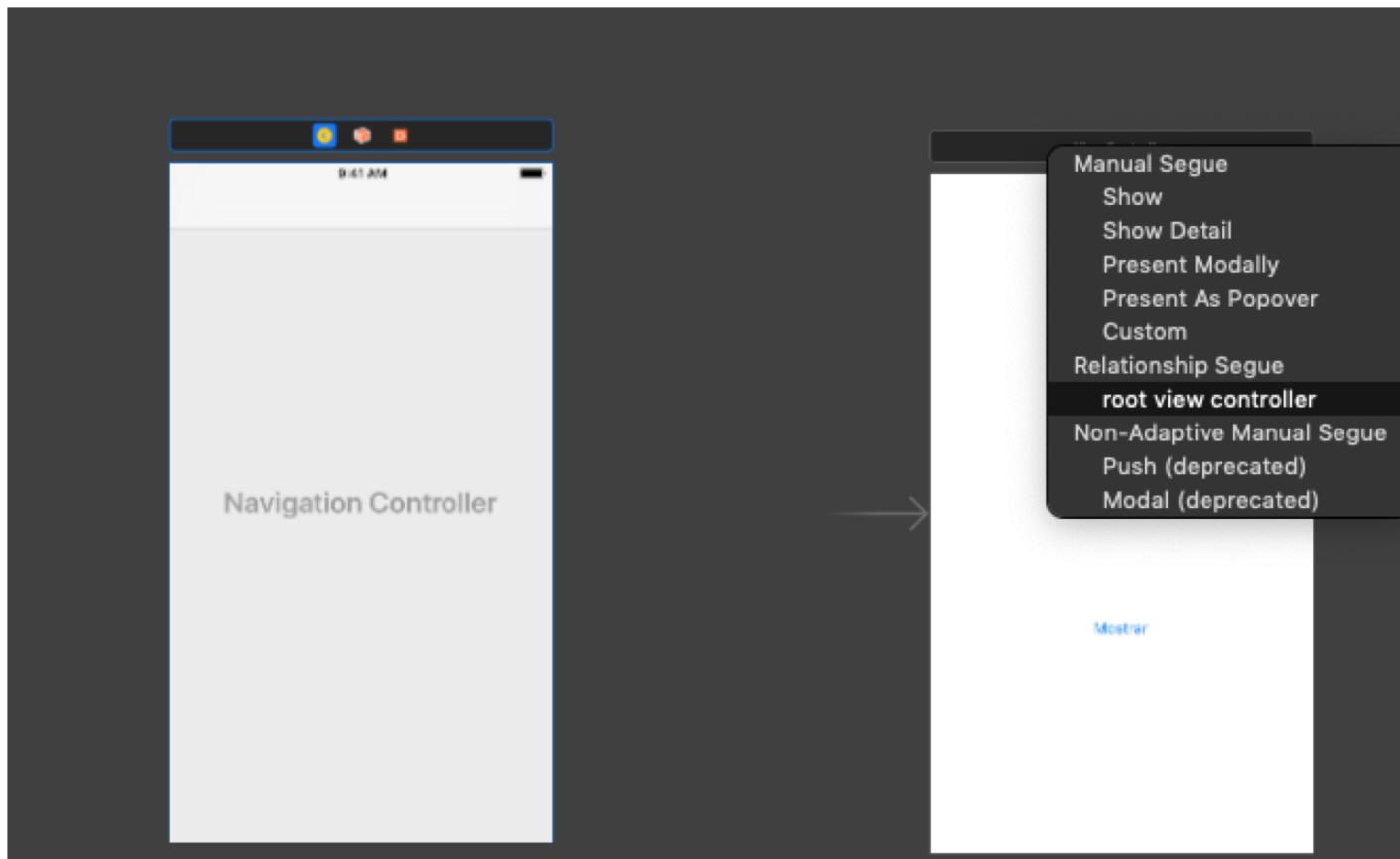
NAVIGATION CONTROLLER

- Faz a arbitragem de naveabilidade entre janelas conectadas por Segues.
- Para o exemplo desta aula, arraste uma Navigation Controller e apague a UITableViewController criada automaticamente



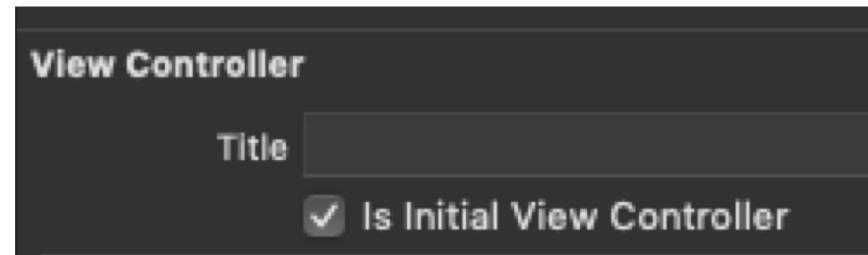
NAVIGATION CONTROLLER

- Depois, monte uma Segue do Tipo Relationship Segue -> Root View Controller



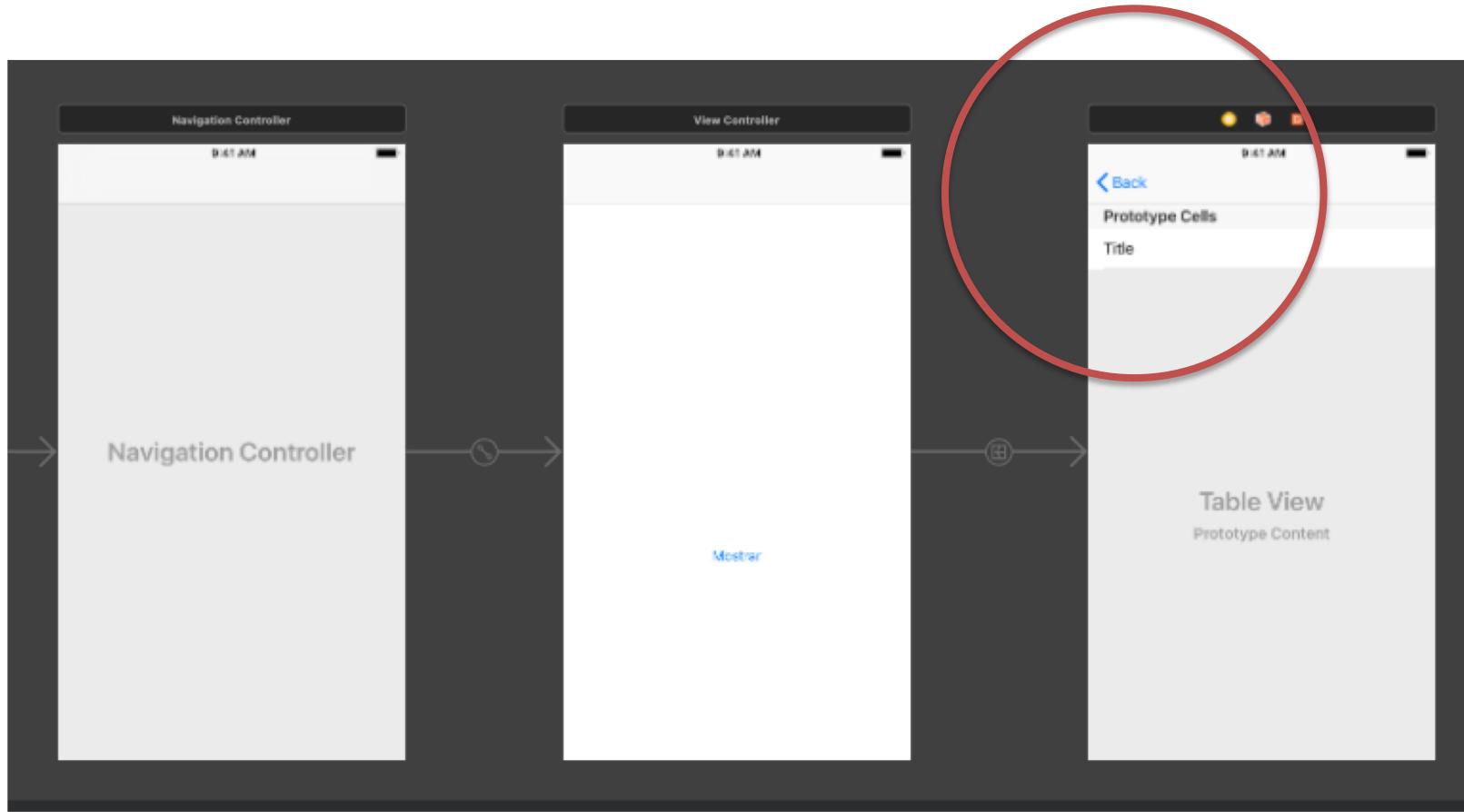
NAVIGATION CONTROLLER

- E então informe que a Navigation Controller é a View Controller Inicial



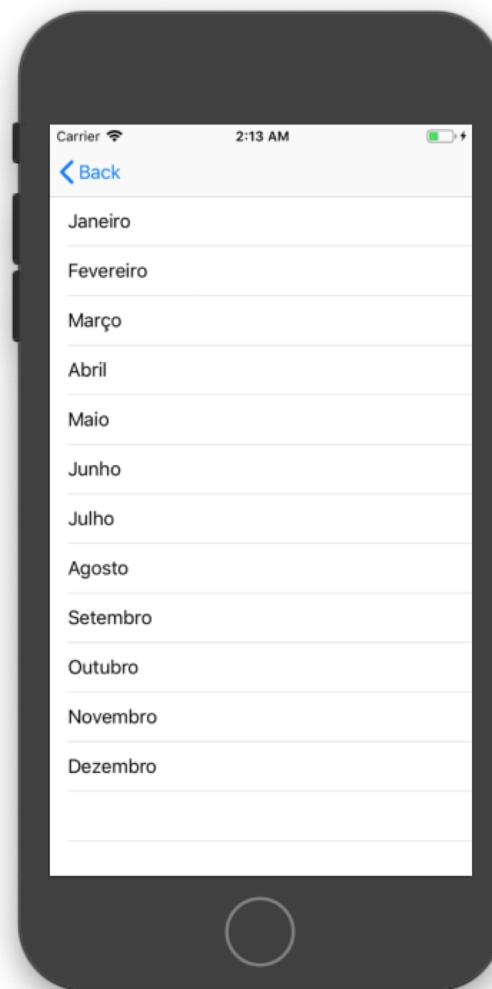
NAVIGATION CONTROLLER

- A Navigation Controller define automaticamente os ícones de naveabilidade para as outras Views.



NAVIGATION CONTROLLER

- Resultado!



Copyright © 2019 Prof. Gustavo Moreira Calixto

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).