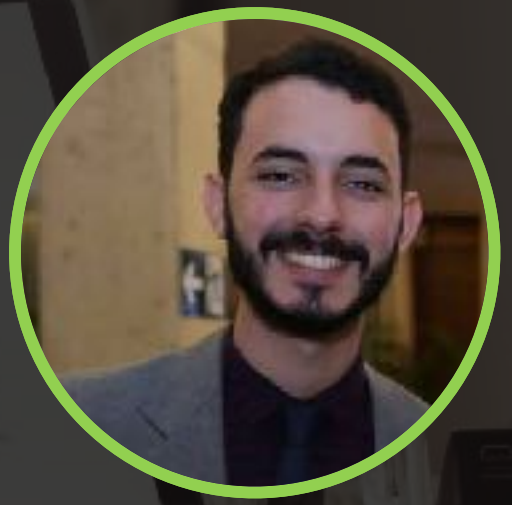

citi

Hilton **Pintor**

Desenvolvedor (iOS/tvOS/watchOS)



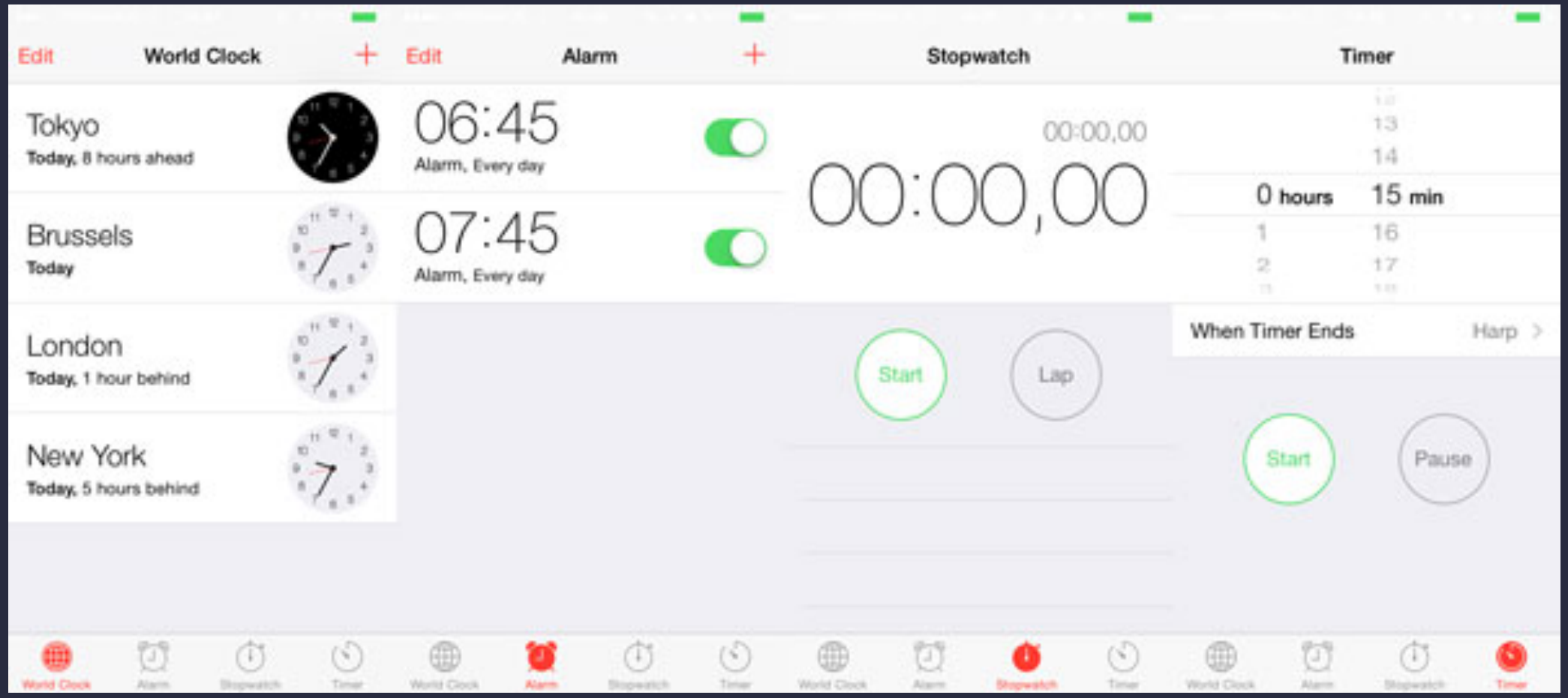
hiltonpintor@gmail.com

citi

// Aula 06

// Tab Bar

// tab bar



https://goo.gl/U9jLvQ

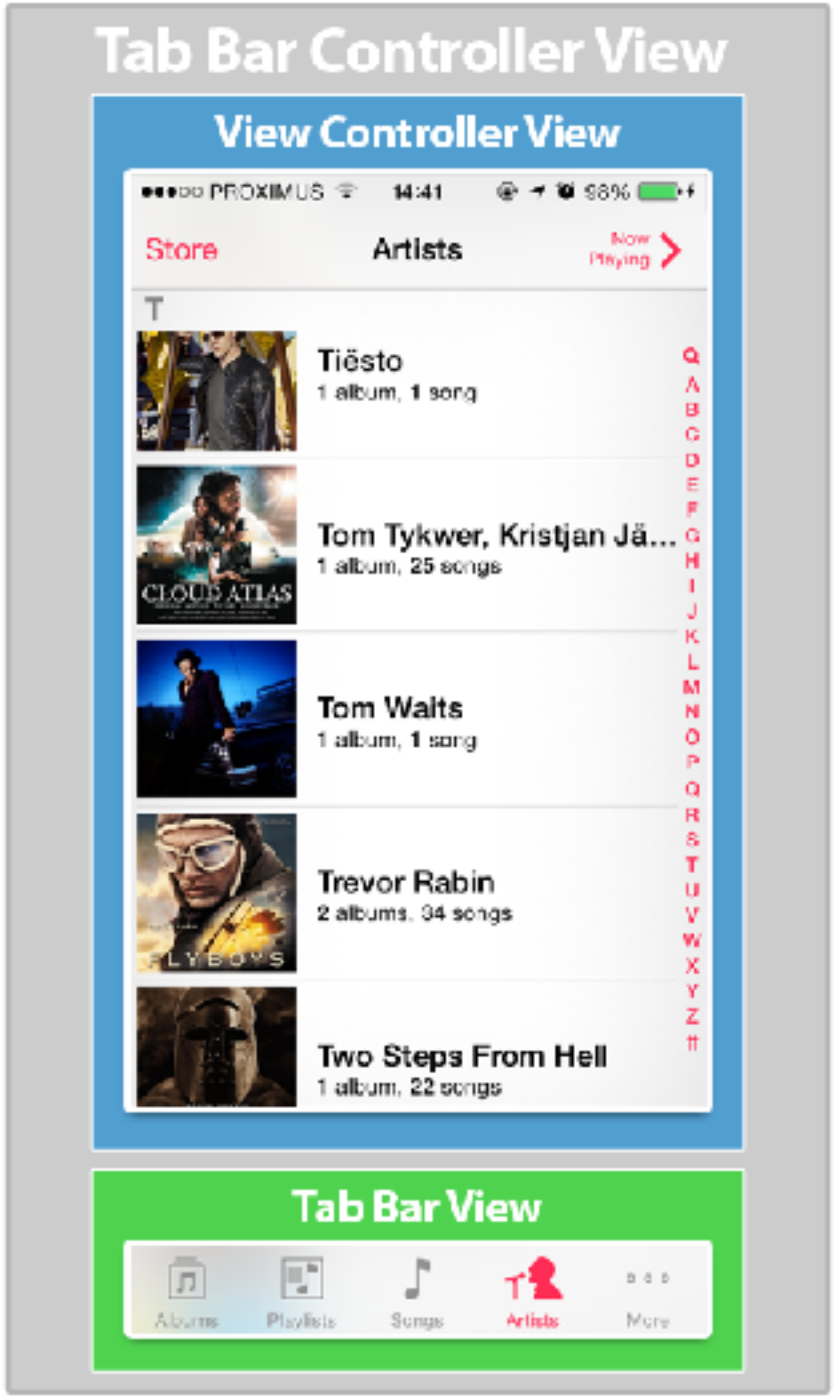
// tab bar

Gerencia navegação

View Controllers não relacionados

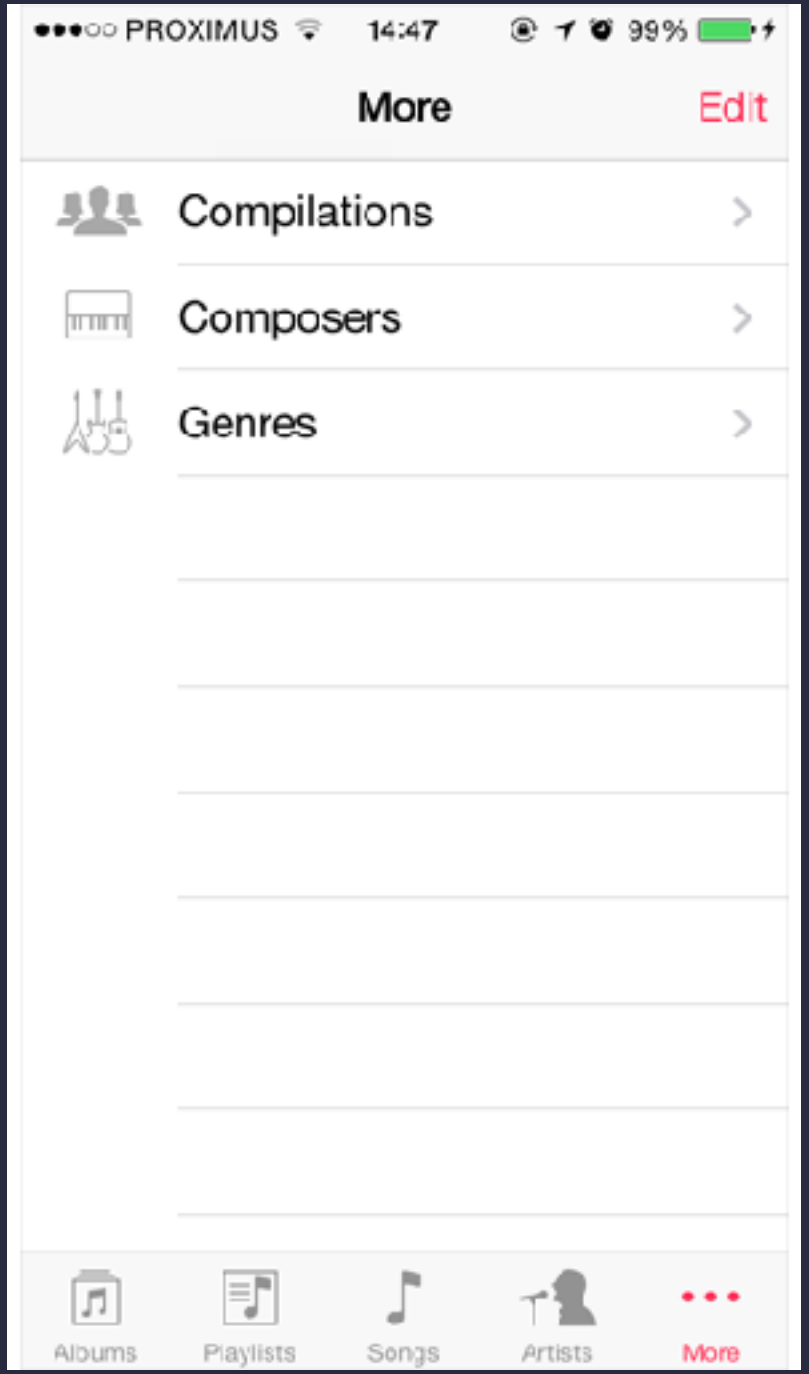
Tab bar vs Nav bar

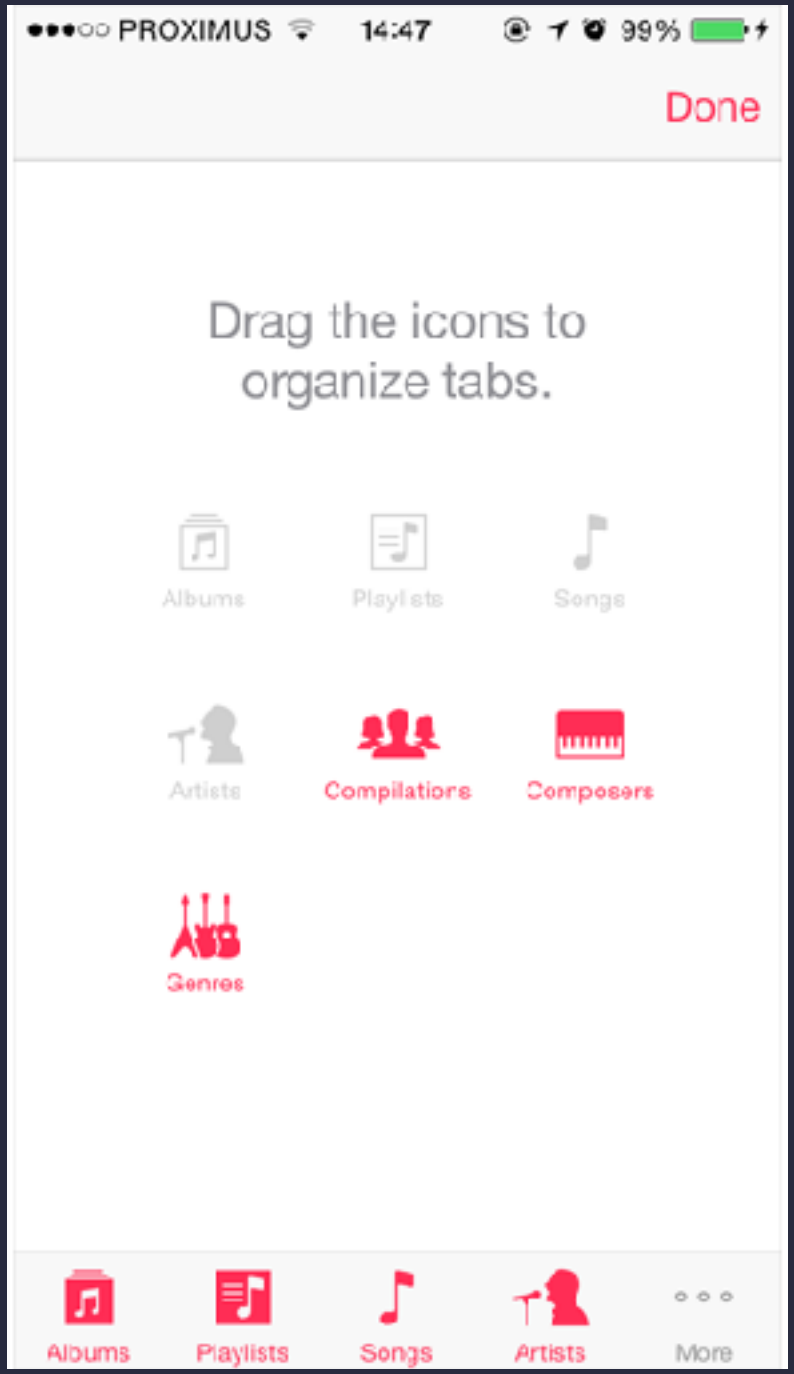
// tab bar



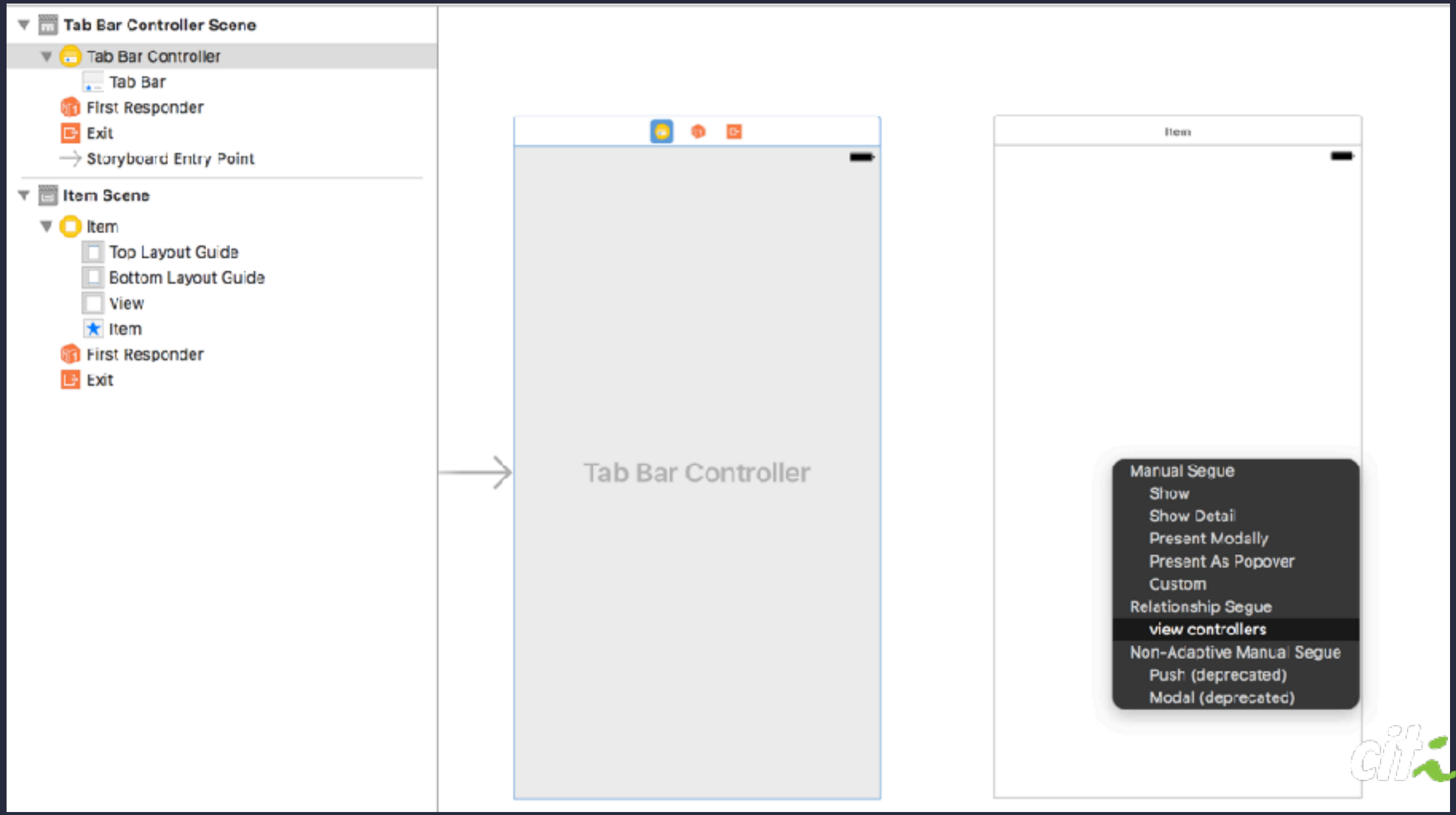
https://goo.gl/U9jLvQ

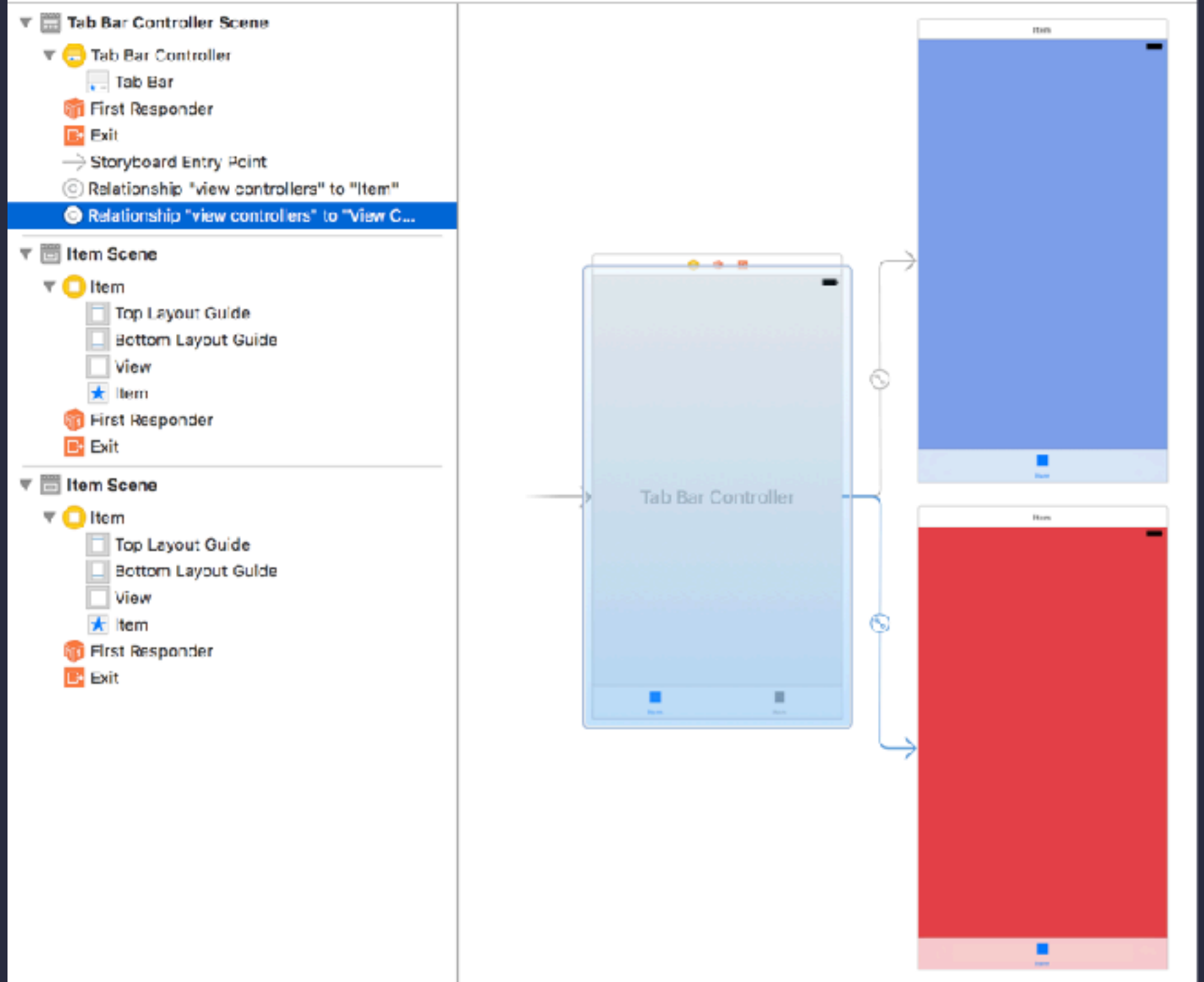
// tab bar





https://goo.gl/U9jLvQ


Tab Bar Controller Scene
Tab Bar Controller
Tab Bar
First Responder
Exit
Storyboard Entry Point
Item Scene
Item
Top Layout Guide
Bottom Layout Guide
View
Item
First Responder
Exit
Tab Bar Controller
Item
Manual Segue
Show
Show Detail
Present Modally
Present As Popover
Custom
Relationship Segue
view controllers
Non-Adaptive Manual Segue
Push (deprecated)
Modal (deprecated)


Tab Bar Controller Scene
Tab Bar Controller
Tab Bar
First Responder
Exit
Storyboard Entry Point
Relationship "view controllers" to "Item"
Relationship "view controllers" to "View C...
Item Scene
Item
Top Layout Guide
Bottom Layout Guide
View
Item
First Responder
Exit
Item Scene
Item
Top Layout Guide
Bottom Layout Guide
View
Item
First Responder
Exit
Tab Bar Controller

// tab bar item

```
class RedViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

        // Do any additional setup after loading the view.

        self.tabBarItem = UITabBarItem(title: "Fire", image: fire, tag: 2)
    }
```

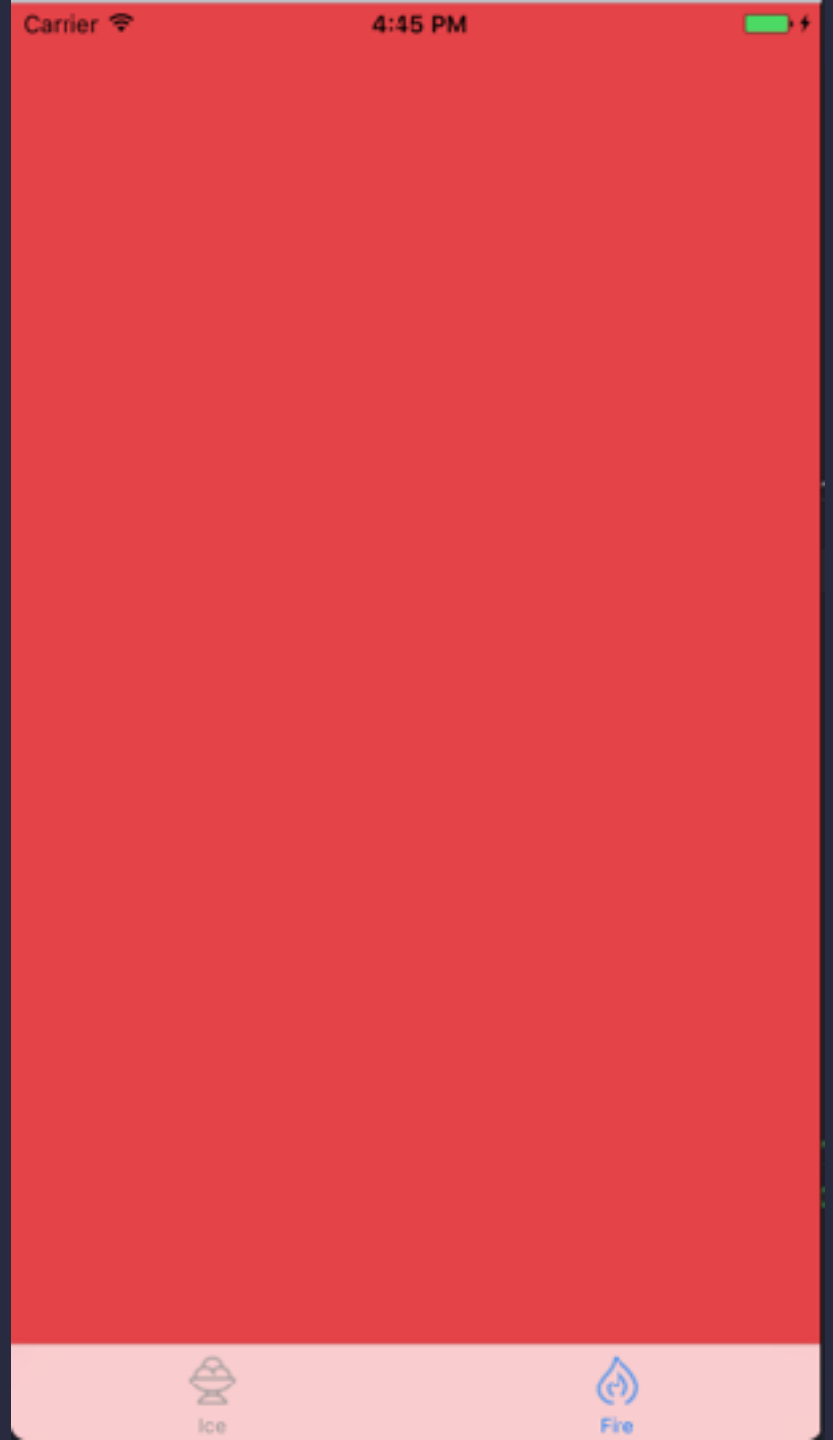
```
class BlueViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

        // Do any additional setup after loading the view.

        self.tabBarItem.title = "Ice"
        self.tabBarItem.image = ice
    }
```

icons by: https://icons8.com/

// tab bar item

icons by: https://icons8.com/

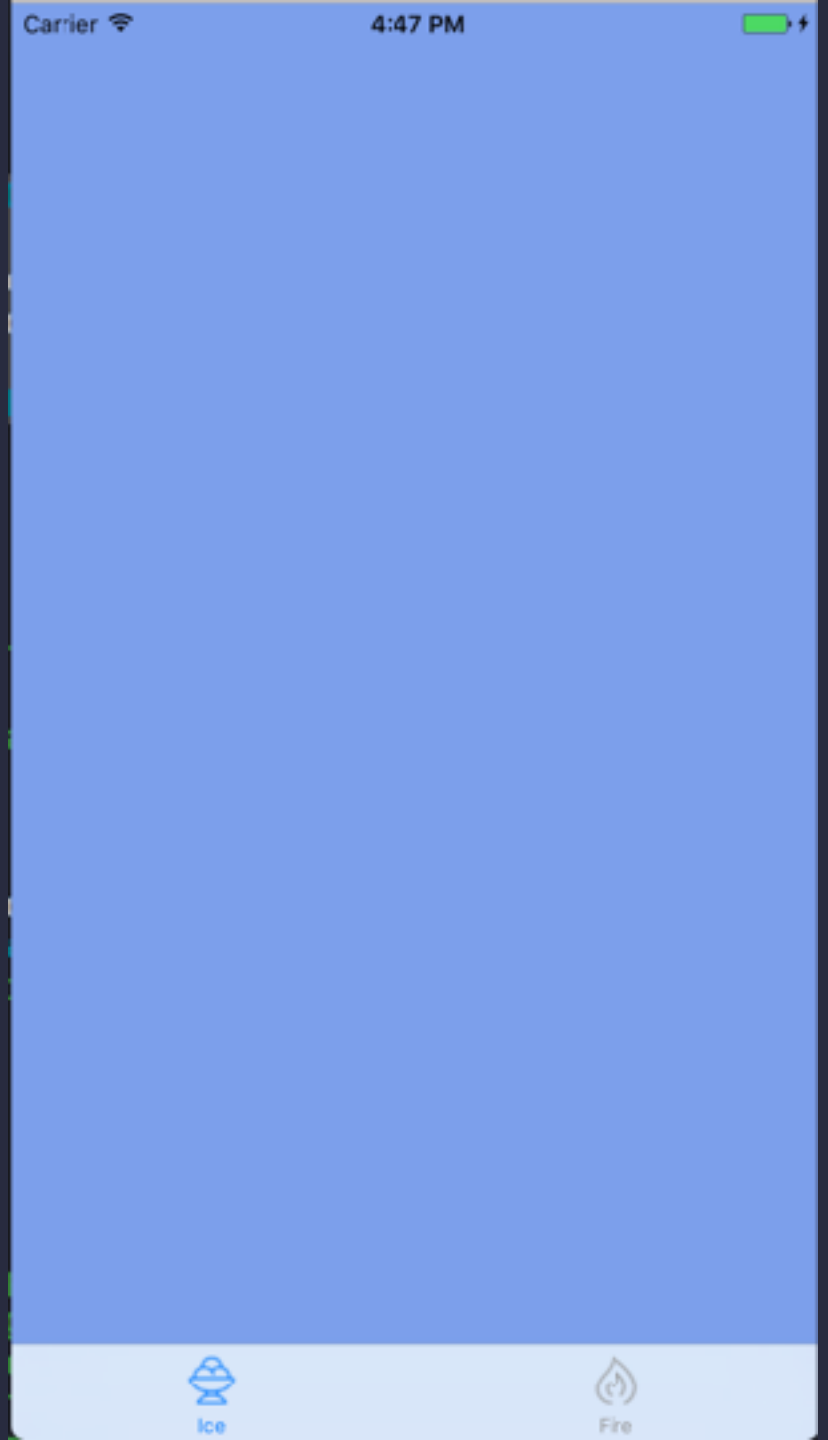
// tab bar item



icons by: https://icons8.com/

```
// tab bar item

class RedViewController: UIViewController {

    required init?(coder aDecoder: NSCoder) {
        super.init(coder: aDecoder)

        self.tabBarItem = UITabBarItem(title: “Fire",
                                       image: #imageLiteral(resourceName: “fire"),
                                       tag: 2)
    }
```

// tab bar item

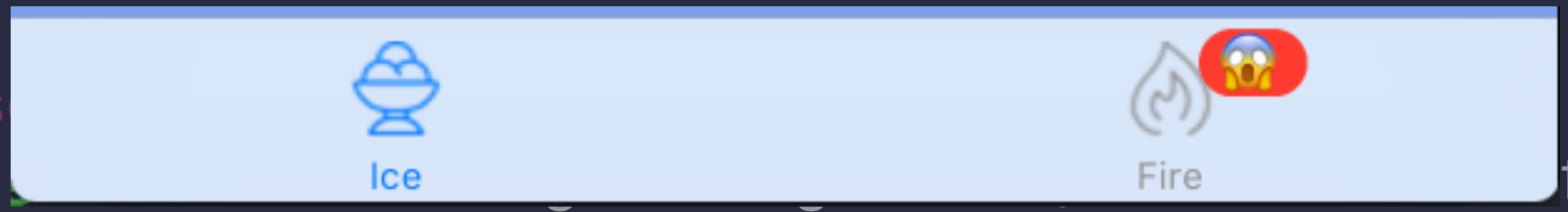


icons by: https://icons8.com/

// tab bar badge

```
class RedViewController: UIViewController {

    required init?(coder aDecoder: NSCoder) {
        super.init(coder: aDecoder)

        self.tabBarItem = UITabBarItem(title: “Fire",
                                       image: #imageLiteral(resourceName: “fire"),
                                       tag: 2)

         self.tabBarItem.badgeValue = “😱"
    }
```

```
// tab bar badge

class RedViewController: UIViewController {

    required init?(coder aDecoder: NSCoder) {
        super.init(coder: aDecoder)

        s
                                                                  fire"),
                                tag: 2)

         self.tabBarItem.badgeValue = "😱"
    }
```



// Exercício

// Exercício 12

Tab bar grande

Fazer uma aplicação com um **TabBarController**

Colocar os seguintes **ViewControllers**:

1. CarroViewController
2. MotoViewController
3. BarcoViewController
4. NavioViewController
5. TremViewController
6. OnibusViewController

Colocar um ícone e um título no **construtor** de cada VC

Colocar uma **ImageView** em cada ViewController com uma imagem correspondente

Tentar **reajustar** os itens mostrados na tab bar

// Desafio 01

Tab e Nav

Junte os últimos dois exercícios, para que sua aplicação final contenha uma **Tab Bar**, e uma **Nav Bar** ao mesmo tempo.

// Scroll View

// quando usar

- Conteúdo maior que a tela
- Zoom

// UIKit

- UIScrollView
- Table Views e Collection Views

// características

- A scroll view itself has **no appearance**, but does display transient scrolling indicators as people interact with it.
- **Don’t place a scroll view inside of another scroll view.** Doing so creates an unpredictable interface that’s difficult to control.

// implementação

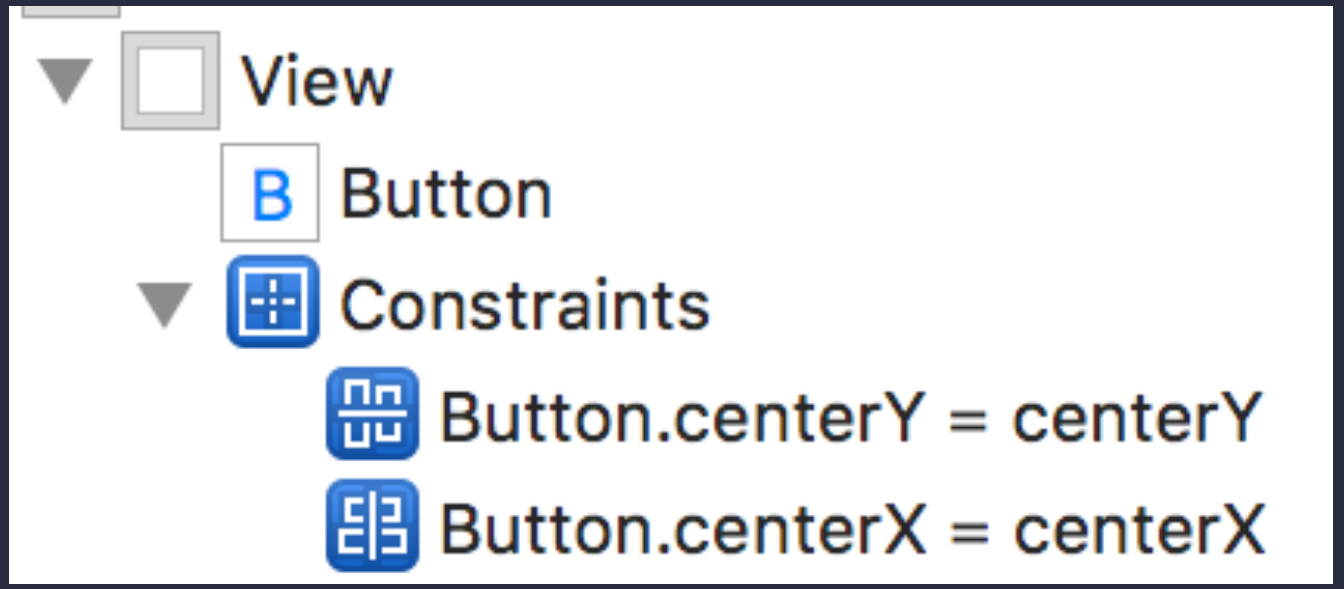
1. mudar tamanho do **ViewController**
2. adicionar **Scroll View**
    2.1. fazer com que cubra a tela toda
3. adicionar **View** de conteúdo
    3.1. fazer com que cubra a tela toda
    3.2. adicionar conteúdo

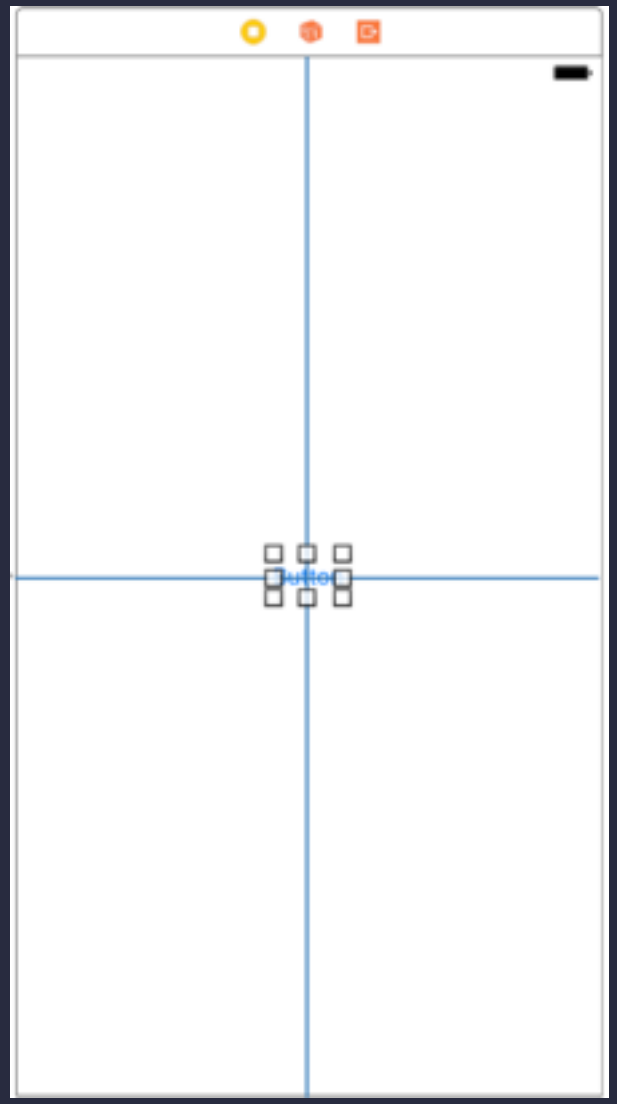
// Auto-Layout

// auto-layout

Auto Layout **dynamically** calculates the **size** and **position** of all the views in your view hierarchy, based on **constraints** placed on those views

// constraints





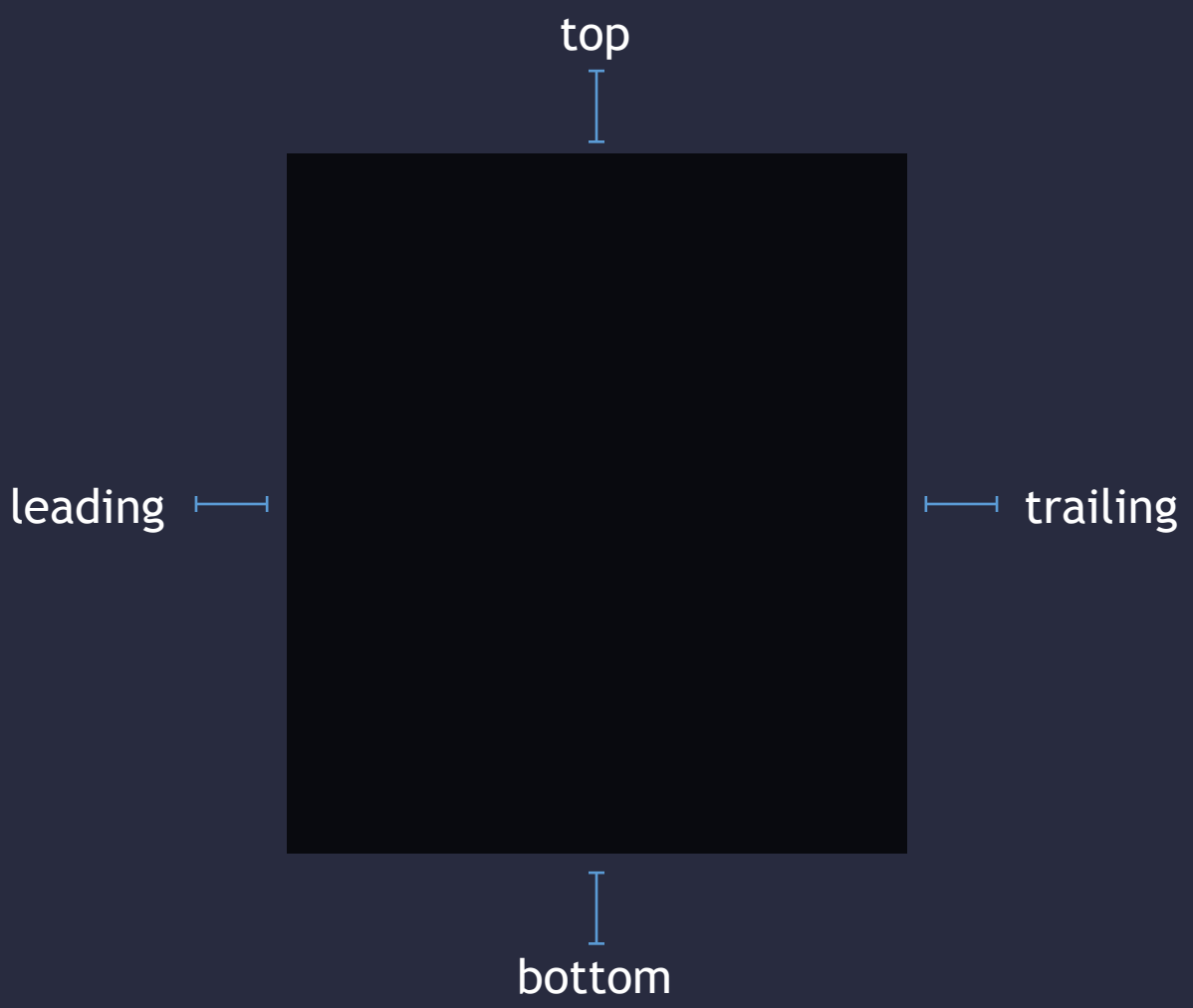
// mudanças externas

- Rotação do dispositivo
- Diferentes tamanhos de tela
- ...

// mudanças internas

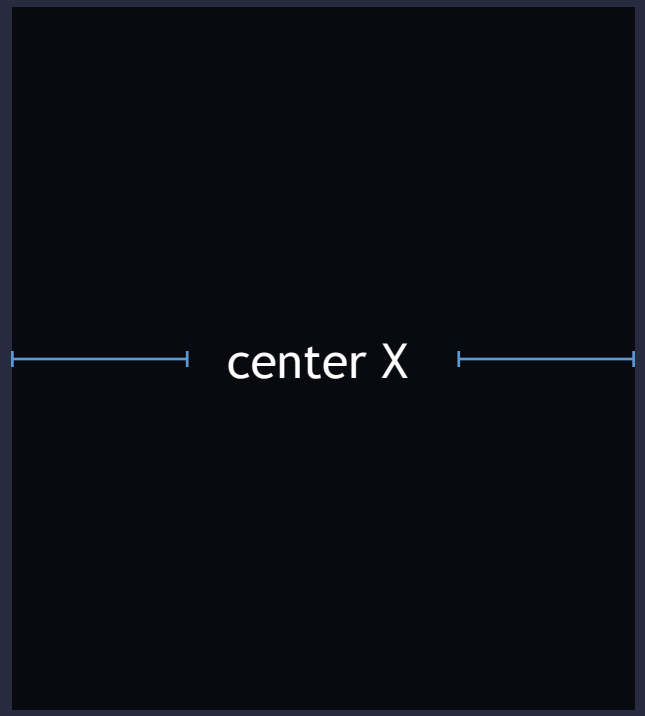
- Mudança no conteúdo do app
- Internacionalização
- Suporte a Dynamic Type

// atributos

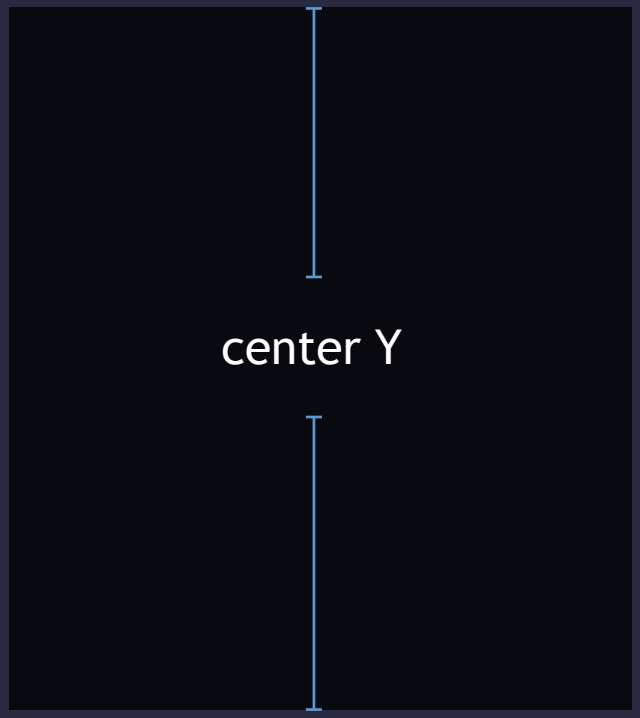


// atributos

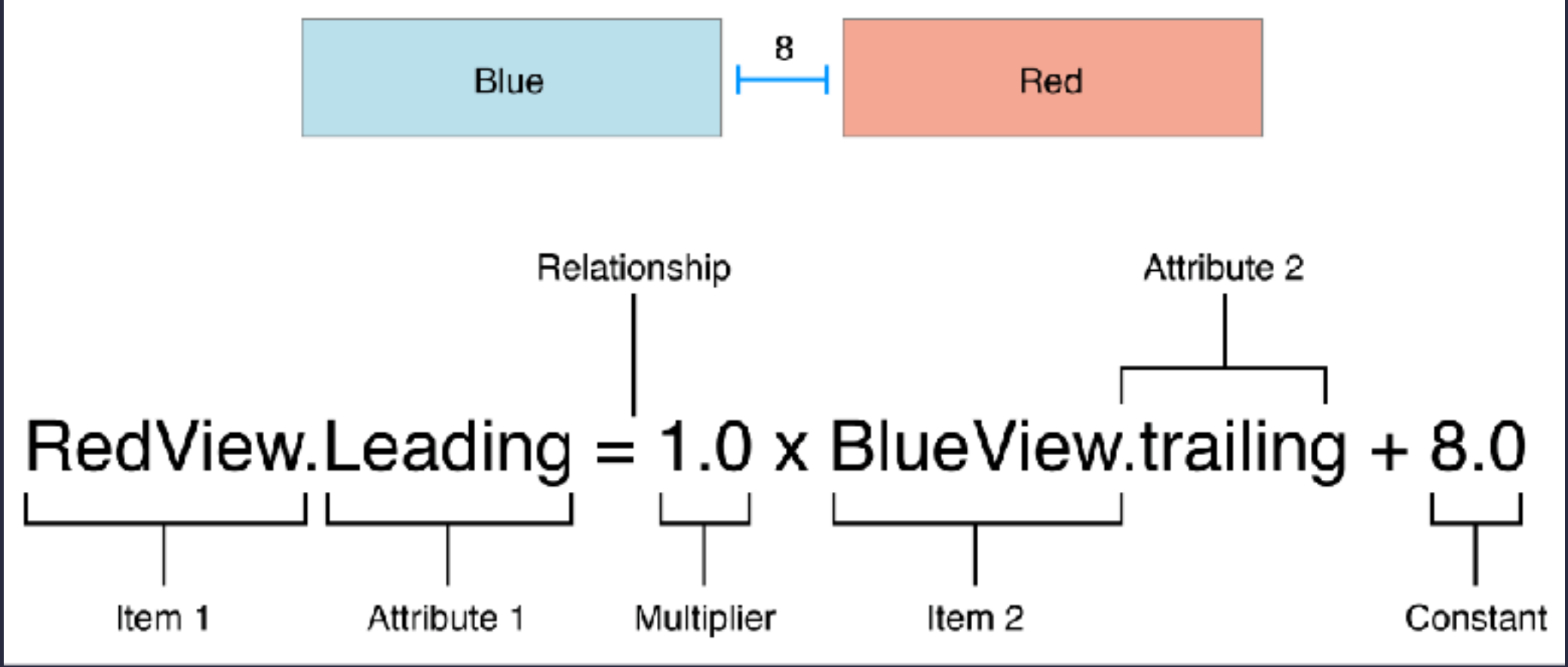
// atributos



// atributos



// constraint



https://goo.gl/2Rx9Jr

// Scroll View no Xcode

// Zoom

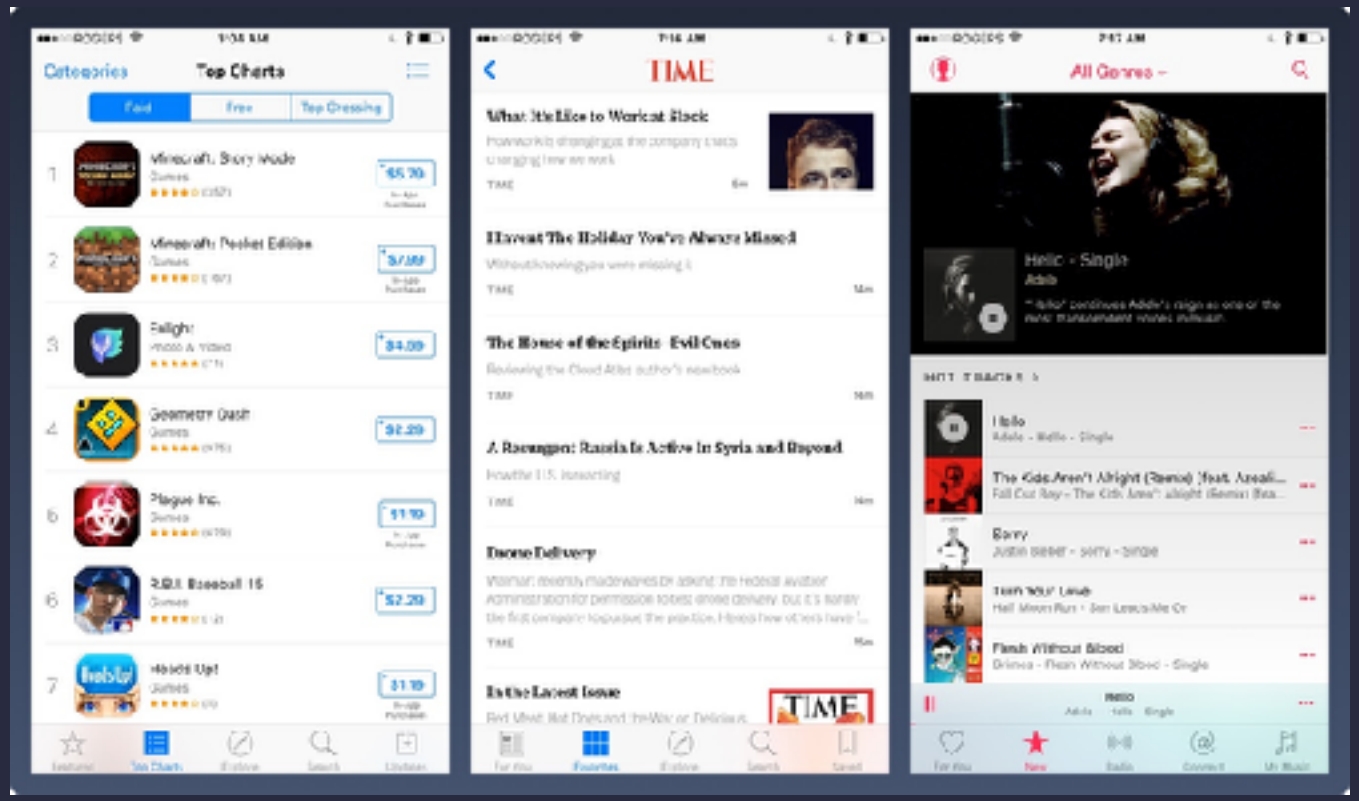
// implementação

1. adicionar **Scroll View**
2. adicionar **Image View**
3. criar **Outlets**
4. colocar **valores** do **zoom**
5. implementar **delegate**

// Table View

// table view

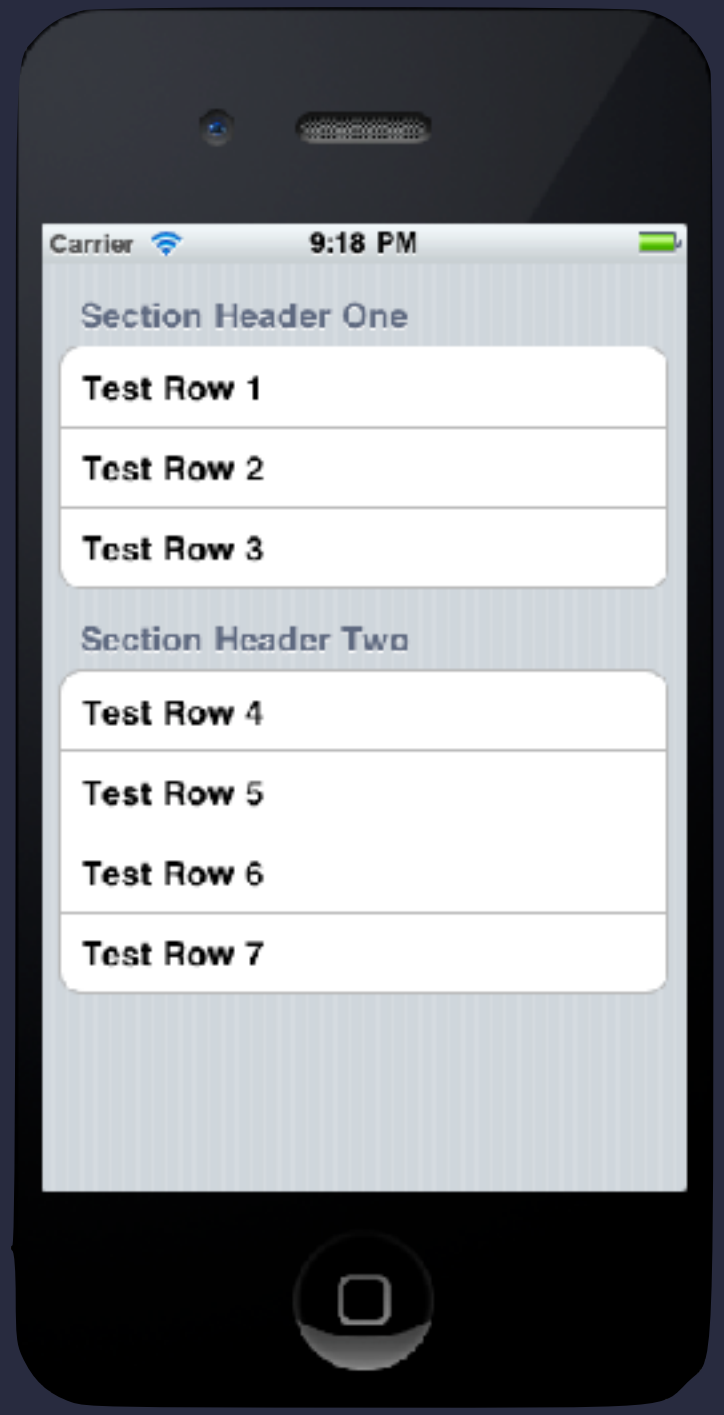
- Conteúdo em lista
- Customizável
- Subclasse de UIScrollView



https://designcode.io/cloud/chapter1/Design-TableExamples.jpg

// anatomia

- Seções
- Linhas
- Células



// storyBoard

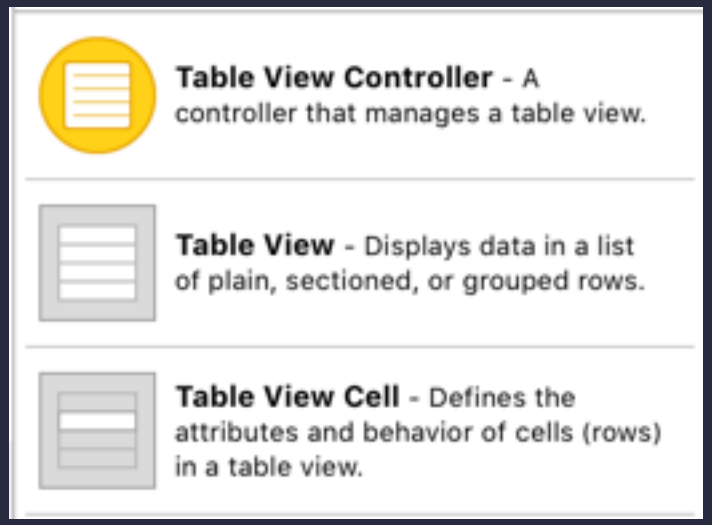


Table View Controller - A controller that manages a table view.
Table View - Displays data in a list of plain, sectioned, or grouped rows.
Table View Cell - Defines the attributes and behavior of cells (rows) in a table view.


Prototype Cells
Table View
Prototype Content

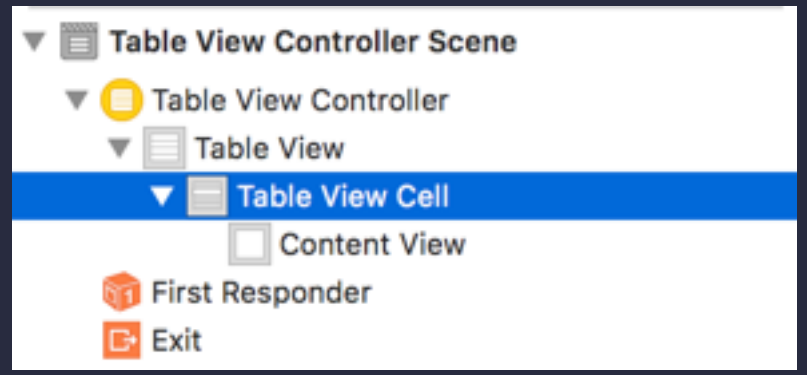


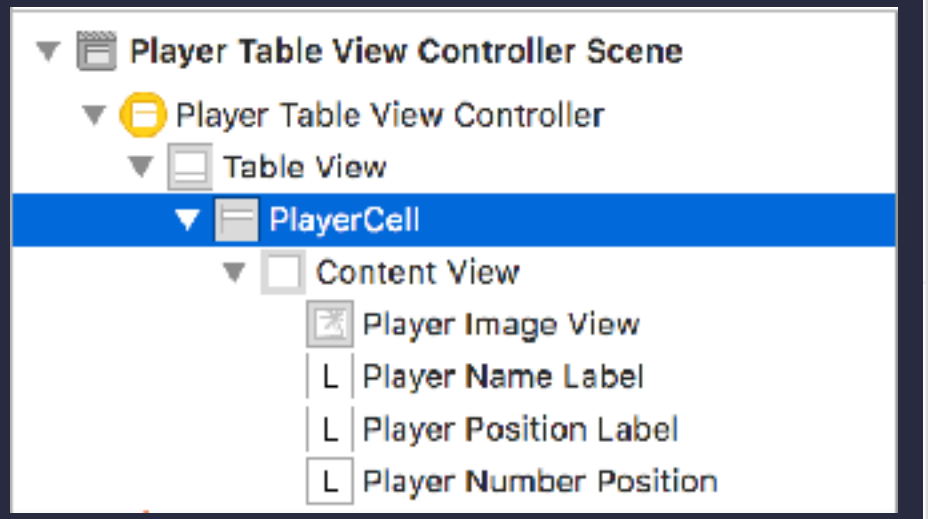
Table View Controller Scene
Table View Controller
Table View
Table View Cell
Content View
First Responder
Exit

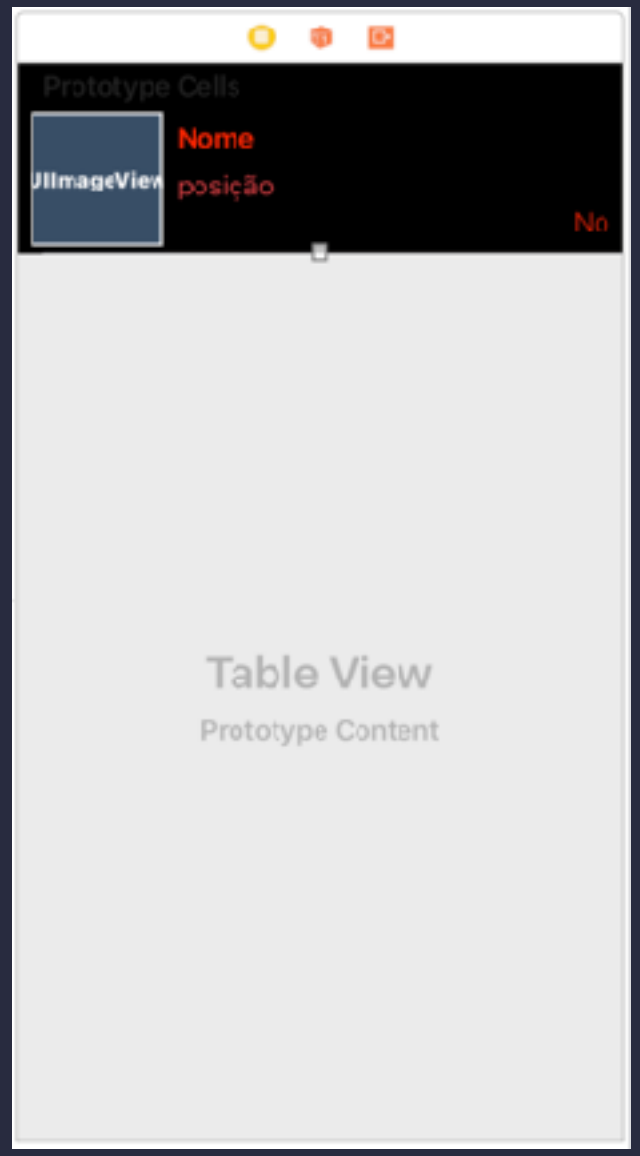
// Implementação

// tipo dos dados


```
class Player {
    var nome: String
    var posicao: String
    var numero: Int
    var foto: UIImage

    init(nome: String, posicao: String, numero: Int, foto: UIImage) {
        self.nome = nome
        self.posicao = posicao
        self.numero = numero
        self.foto = foto
    }
}
```

// células


Player Table View Controller Scene
Player Table View Controller
Table View
PlayerCell
Content View
Player Image View
Player Name Label
Player Position Label
Player Number Position


Prototype Cells
UIImageView
Nome
posição
No
Table View
Prototype Content

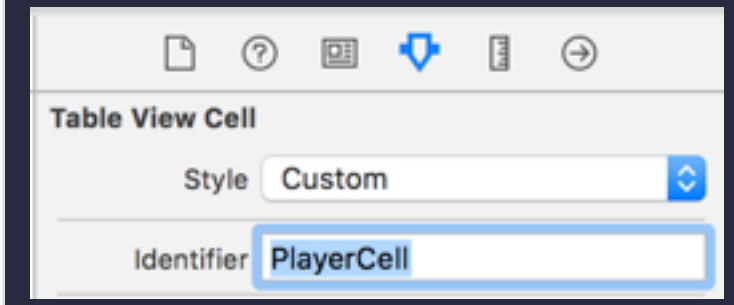
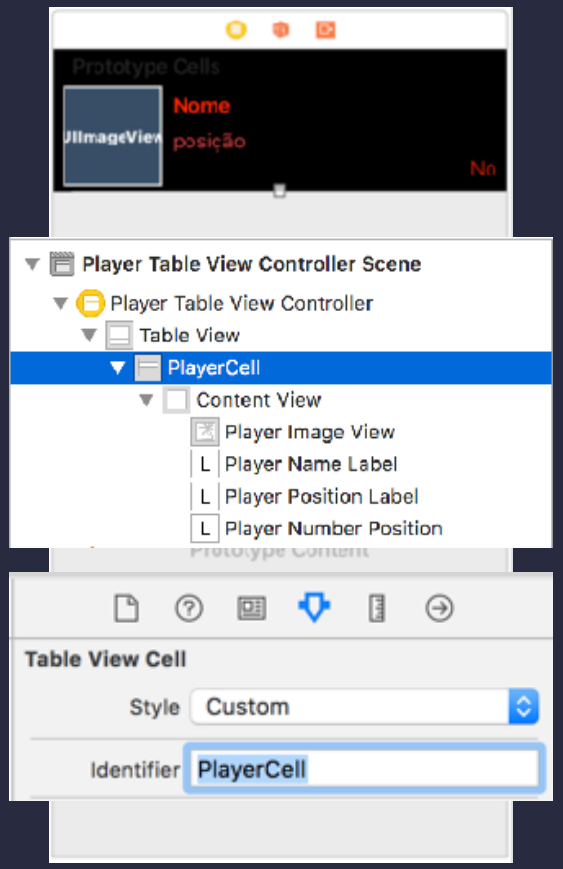


Table View Cell
Style
Custom
Identifier
PlayerCell

// células




```
class PlayerTableViewCell: UITableViewCell {
    @IBOutlet weak var playerImageView: UIImageView!
    @IBOutlet weak var playerNameLabel: UILabel!
    @IBOutlet weak var playerPositionLabel: UILabel!
    @IBOutlet weak var playerNumberPosition: UILabel!
        // ...
}
```

// controlador

```
class PlayerTableViewController: UITableViewController {
    var players: [Player] = []



}
```

// dados a serem mostrados

```
class PlayerTableViewController: UITableViewController {
    var players: [Player] = []



}
```

// criando/carregando dados

```
// MARK: - Cell Setup
private func loadPlayers() {
    // create player objects
    let diegoSouza = Player(nome: "Diego Souza", posicao: "Meia", numero: 87, foto: diego)
    let magrao = Player(nome: "Magrão", posicao: "Goleiro", numero: 1, foto: magrao)
    let durval = Player(nome: "Durval", posicao: "Zagueiro", numero: 4, foto: durval)

    // add them to players property
    players.append(contentsOf: [diegoSouza, magrao, durval])
}
```

// chamando função

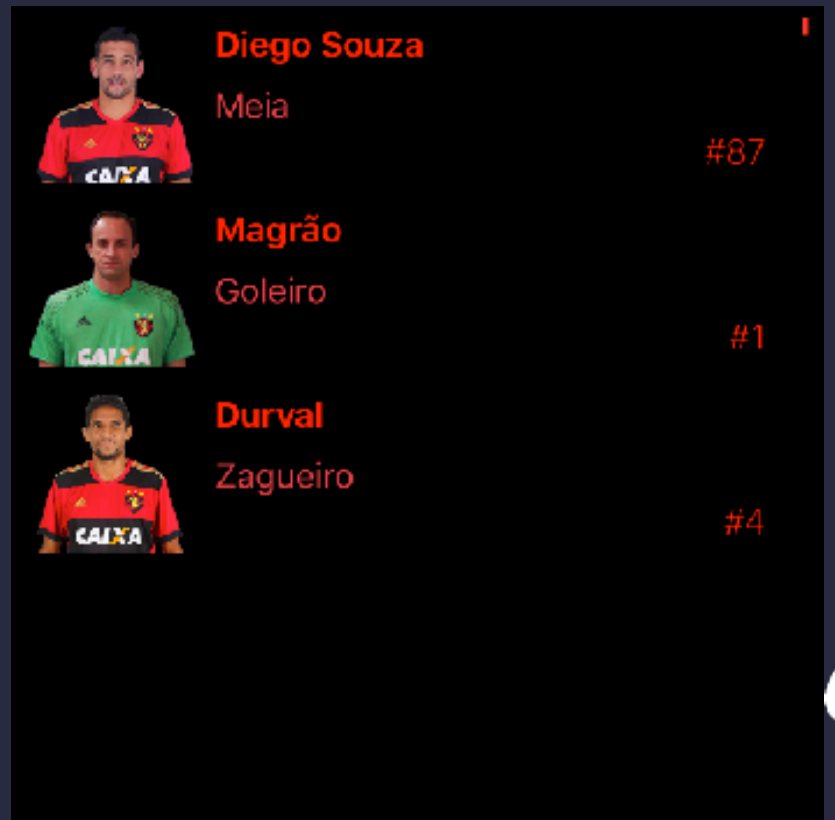
```
class PlayerTableViewController: UITableViewController {
    var players: [Player] = []

    override func viewDidLoad() {
        super.viewDidLoad()

        self.loadPlayers()
    }



}
```

// data source

- protocolo UITableViewDataSource
- provedor de dados
- implementado por UITableViewController

citi

```
// data source

class PlayerTableViewController: UITableViewController {

  override func numberOfSections(in tableView: UITableView) -> Int {
      // return the number of sections
      return 1
  }

}
```

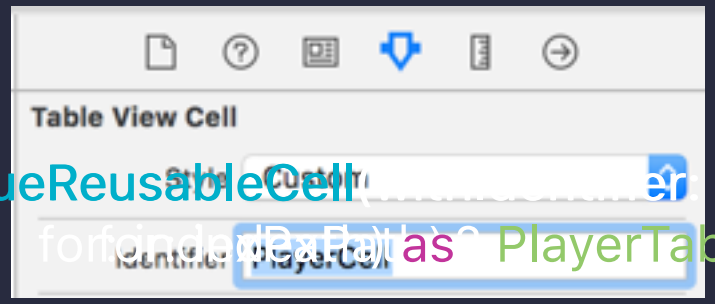


// data source

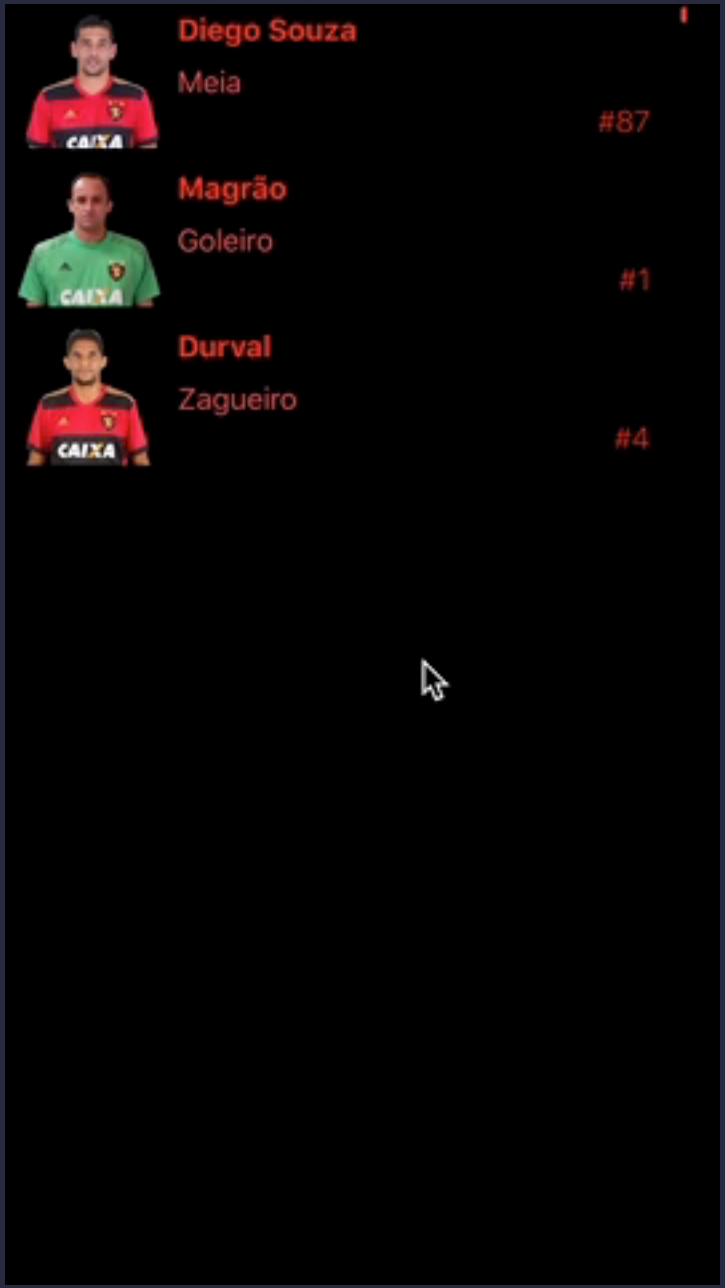
```
class PlayerTableViewController: UITableViewController {
  var players: [Player] = []

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

      // return the number of rows
      return self.players.count
  }
}
```

// data source

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

    let cellIdentifier = "PlayerCell"

    guard let cell = tableView.dequeueReusableCell(withIdentifier: cellIdentifier,
                                                   forIndexPath: indexPath) as? PlayerTableViewCell

      else {
       fatalError("Tipo de célula não é PlayerTableViewCell")
    }

    // Configure the cell...
    let currentPlayer = self.players[indexPath.row]

    cell.playerImageView.image = currentPlayer.foto
    cell.playerNameLabel.text = currentPlayer.nome
    cell.playerPositionLabel.text = currentPlayer.posicao
    cell.playerNumberPosition.text = "#" + String(currentPlayer.numero)

    return cell
}
```



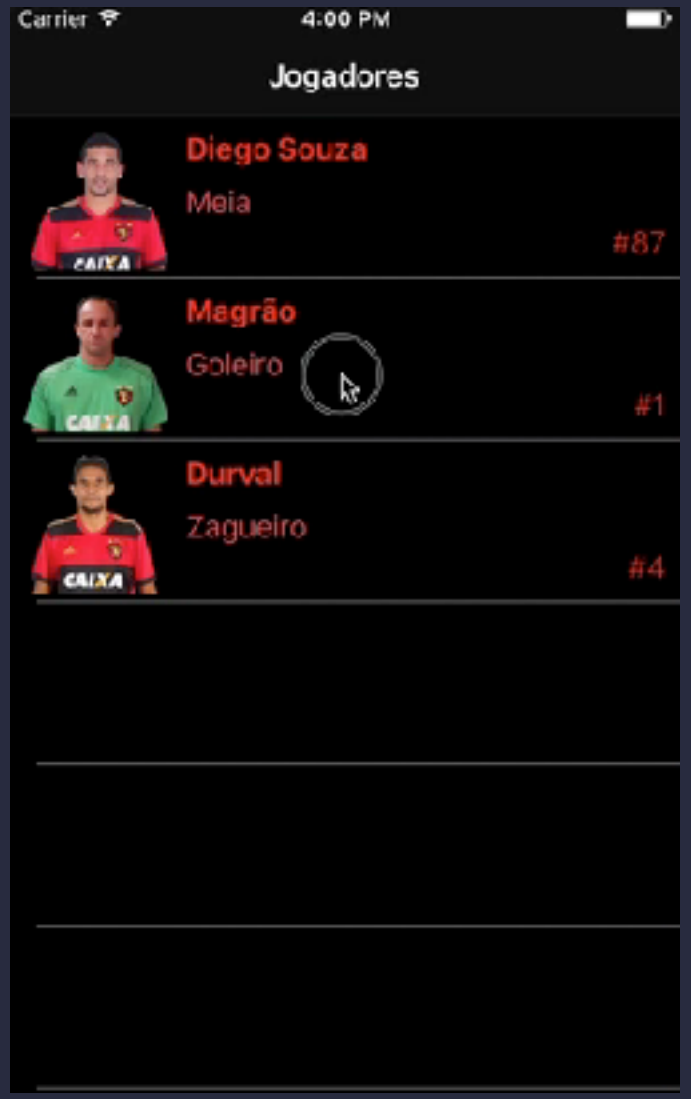
// resultado final



// Exercício

// Exercício 13

Lista de coisas



1. Faça uma Table View que mostre pelo menos **3 itens** de uma lista (contendo pelo menos foto e nome)

2. ao tocar num item, **direcione o usuário** para uma tela com os detalhes do item selecionado

    2.1. Permita que o usuário possa dar **zoom** na foto mostrada

cit

// Ementa

Swift:

- Por que desenvolver em swift?
- Variavéis, constantes e operadores
- Tipos, optionals
- Coleções (arrays, dicionários)
- Fluxo de controle
- Condicionais e loops
- Funções e closures
- Enums
- Classes e structs

Storyboard:

- Model
- View
- Controler

- Delegação
- UIKit
  - Labels
  - Botões
  - Textfields
  - Tableview
- Storyboard
- Navegação
- Persistência local

cit

Swift:

- ~~Por que desenvolver em swift?~~
- ~~Variavéis, constantes e operadores~~
- ~~Tipos, optionals~~
- ~~Coleções (arrays, dicionários)~~
- ~~Fluxo de controle~~
- ~~Condicionais e loops~~
- ~~Funções e closures~~
- ~~Enums~~
- ~~Classes e structs~~

Storyboard:

- ~~Model~~
- ~~View~~
- ~~Controler~~

- ~~Delegação~~
- ~~UIKit~~
  - ~~Labels~~
  - ~~Botões~~
  - ~~Textfields~~
  - ~~Tableview~~
- ~~Storyboard~~
- ~~Navegação~~
- Persistência local

// Proposta semana 02

Segunda	Terça	Quarta	Quinta	Sexta
WebView	Projeto	Projeto	Projeto	Projeto
Persistência		Extra	Extra	Extra

DÚVIDAS

?

citi