

*cit* 

# Hilton Pintor

Desenvolvedor (iOS/tvOS/watchOS)

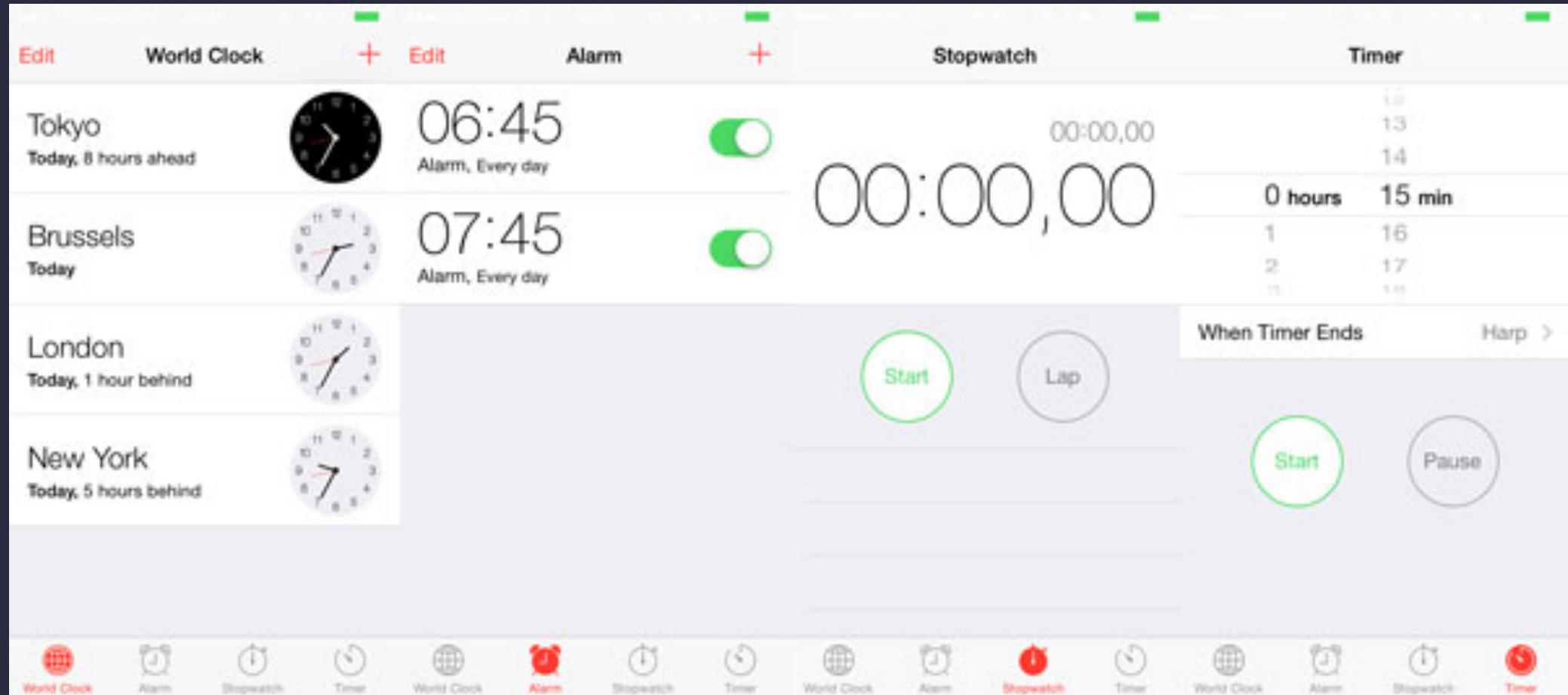


hiltonpintor@gmail.com

# // Aula 06

// Tab Bar

// tab bar



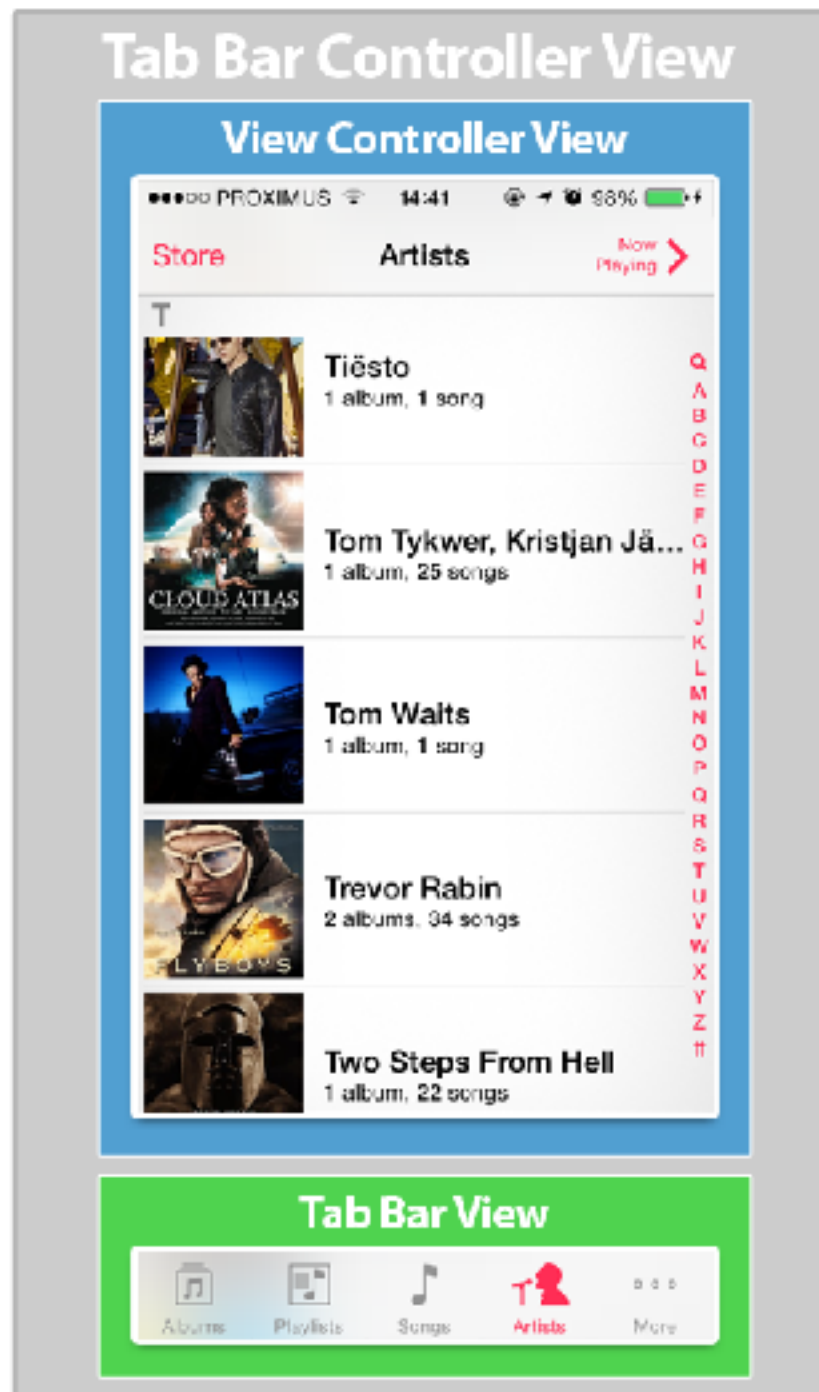
// tab bar

Gerencia navegação

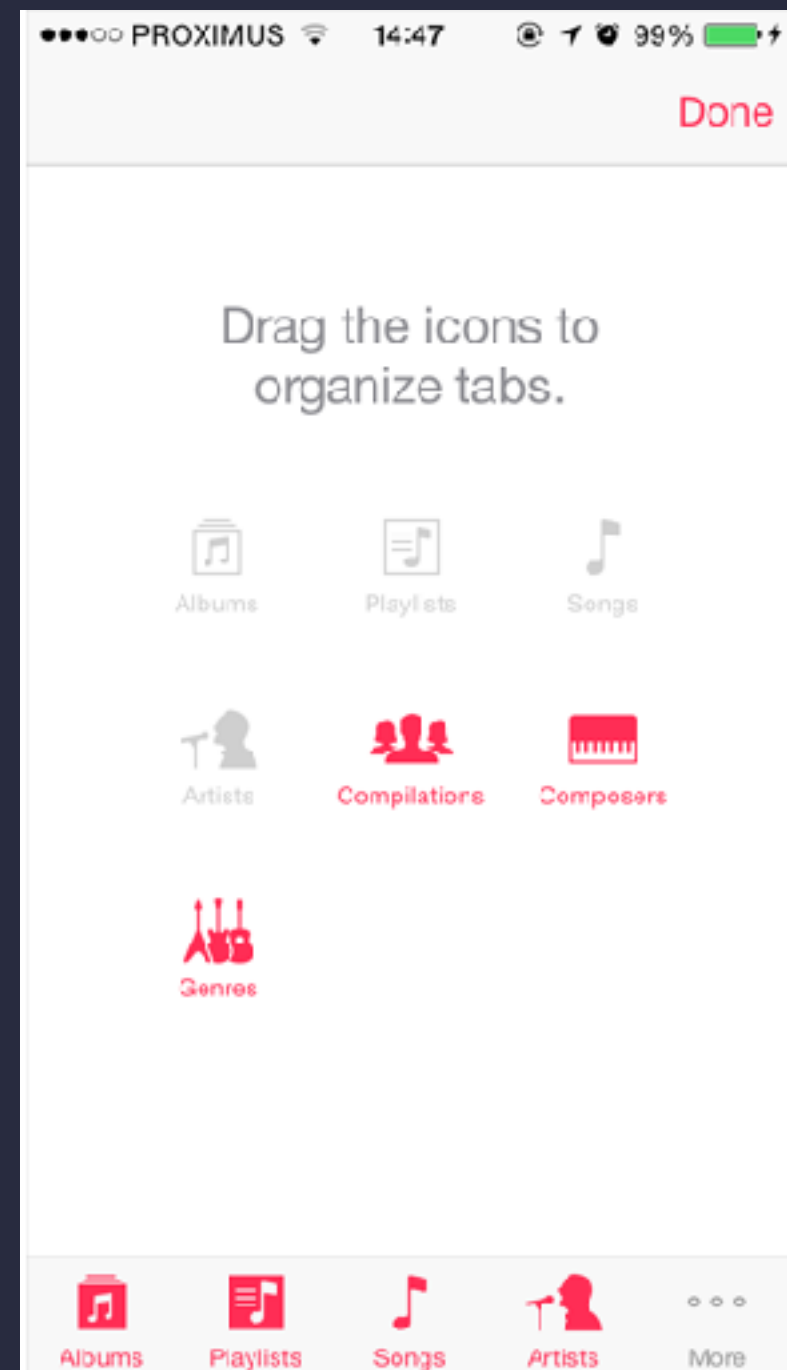
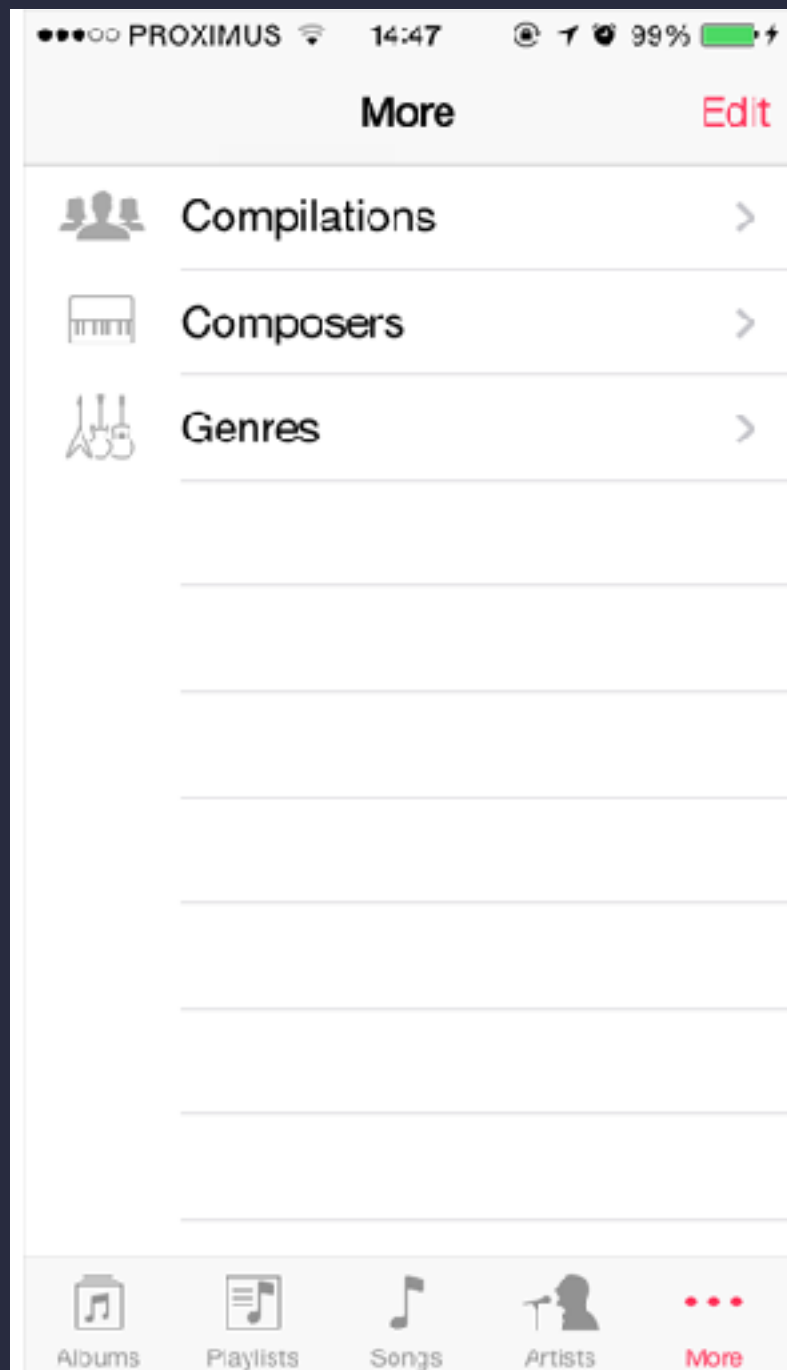
View Controllers não relacionados

Tab bar vs Nav bar

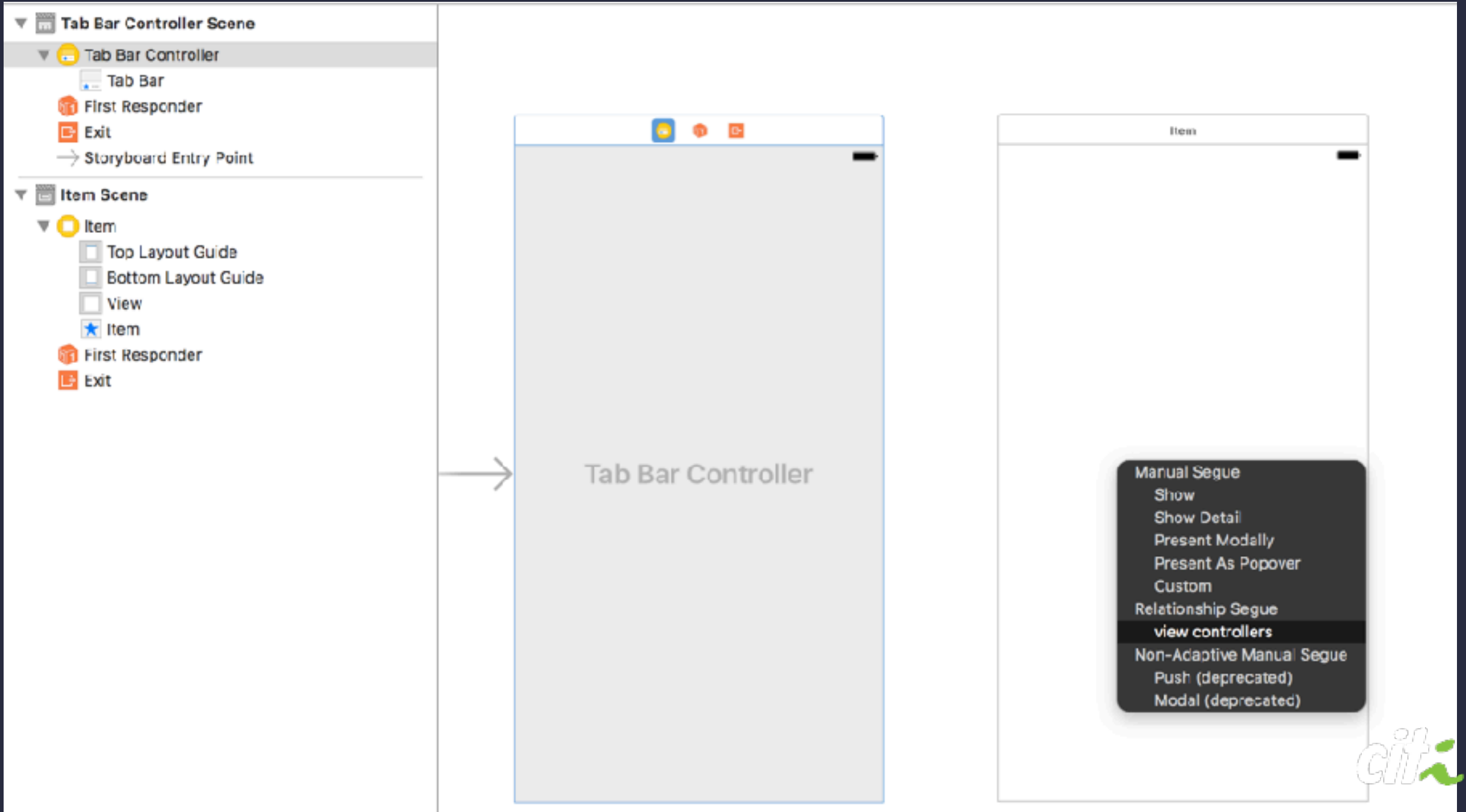
// tab bar

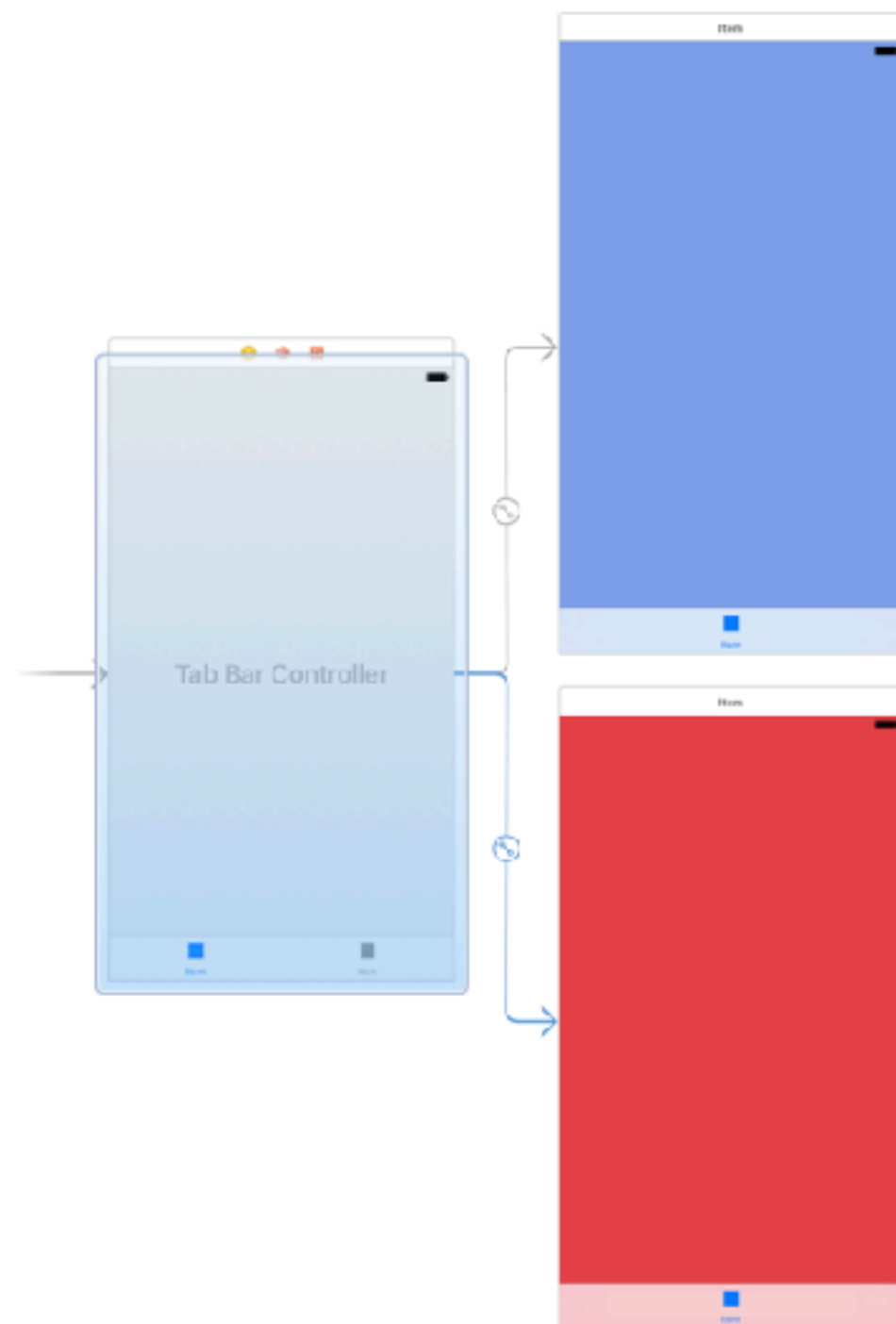
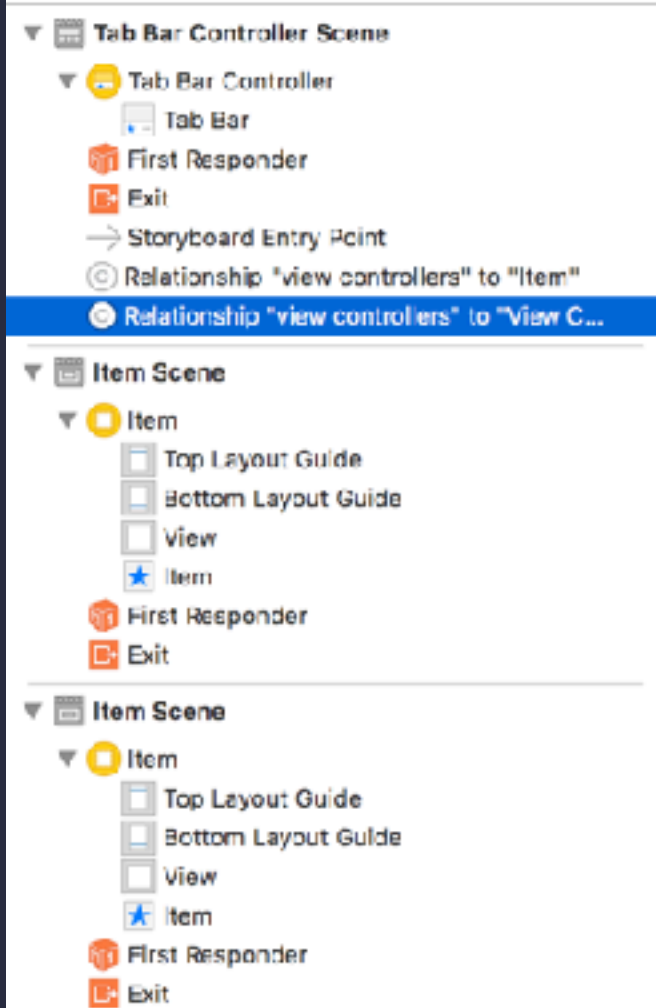


// tab bar









// 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  
    }  
}
```



// tab bar item



// tab bar item



```
// 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



// 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)
```



```
        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

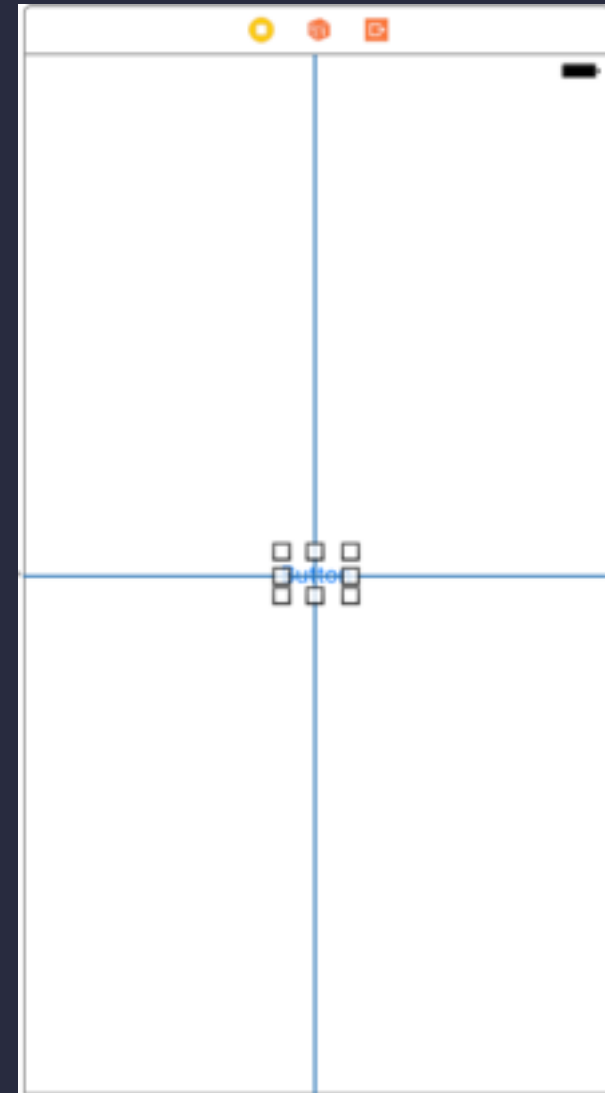
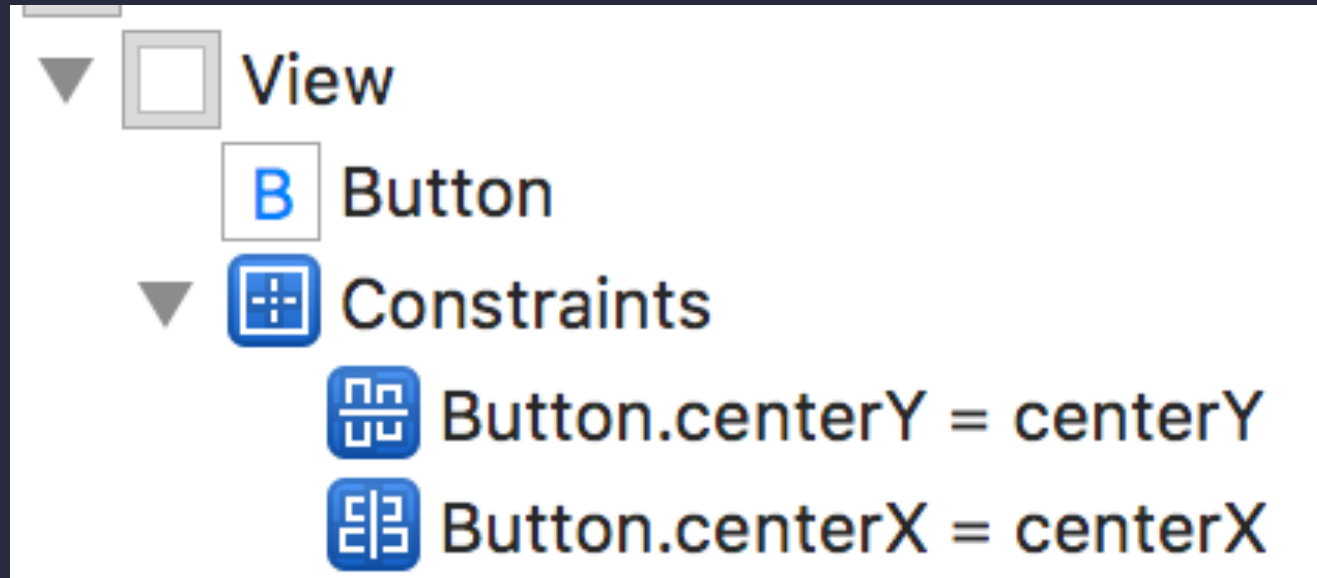
1. mudar tamanho do **ViewController**
2. adicionar **Scroll View**
  - 2.1. fazer com que cubra a tela toda
3. adicionar **View** de conteúdo
  - 3.1. fazer com que cubra a tela toda
  - 3.2. adicionar conteúdo

// 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



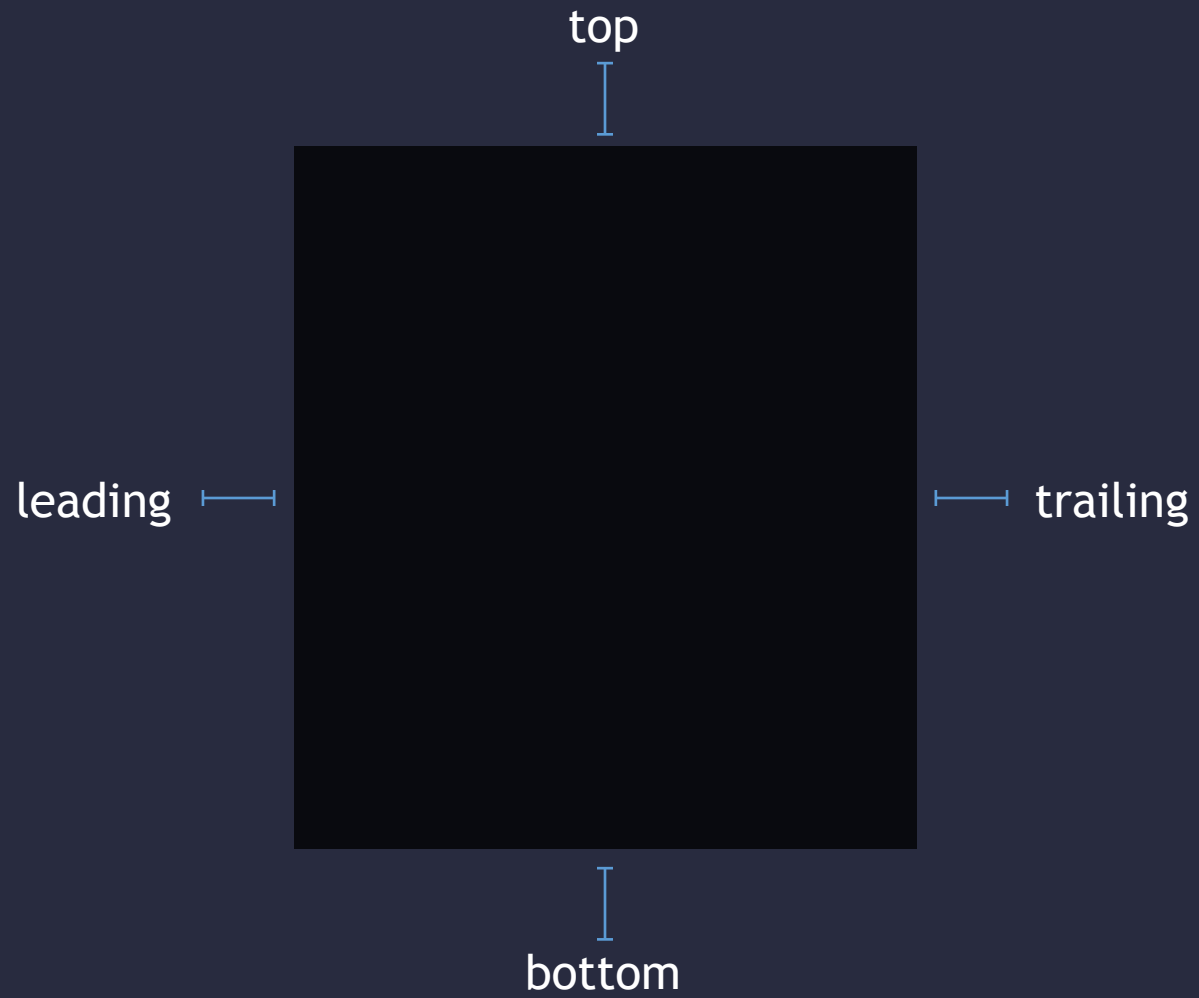
// mudanças externas

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

// mudanças internas

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

// atributos

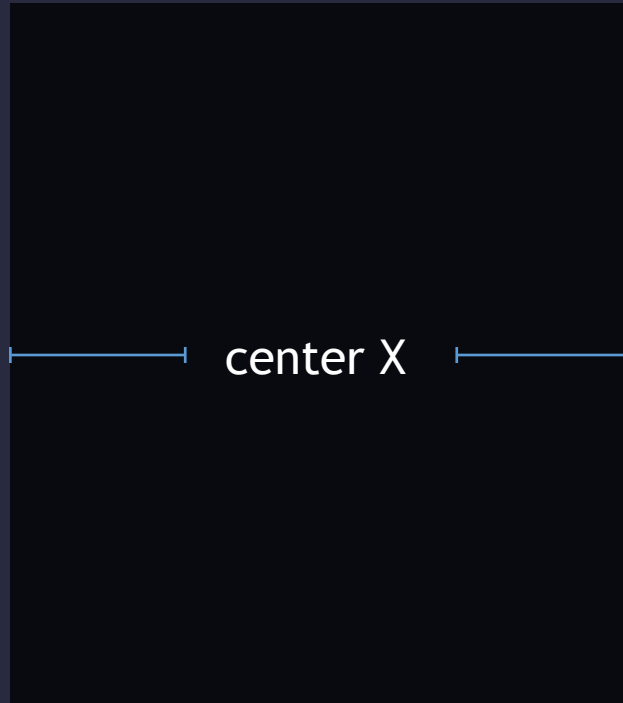


// atributos

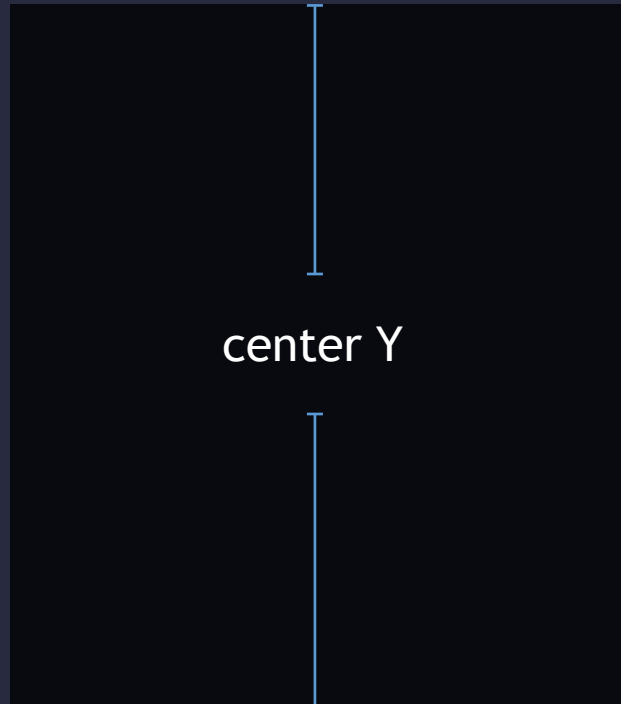




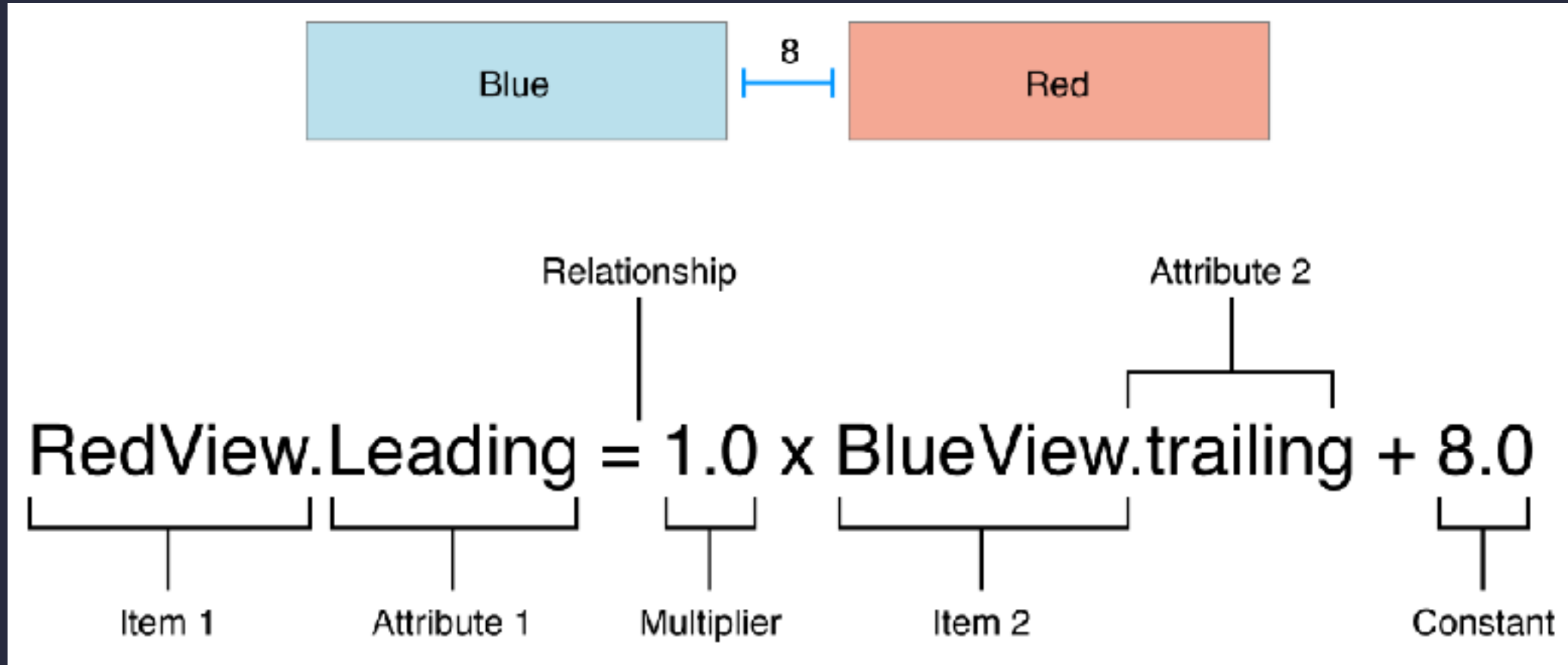
// atributos



// atributos



// constraint



// 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**



## // implementação

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

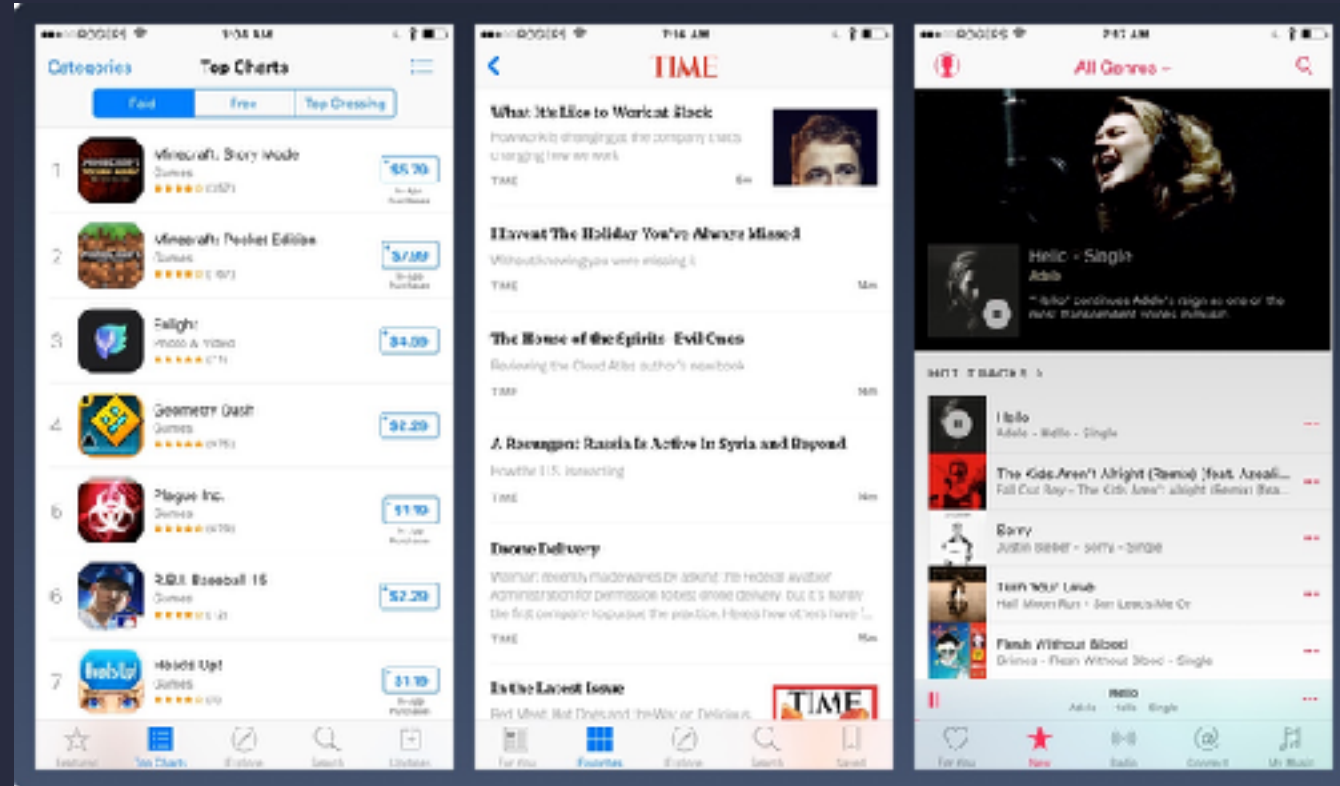


// Table View



// table view

- Conteúdo em lista
- Customizável
- Subclasse de UIScrollView





// anatomia


- Seções
- Linhas
- Células

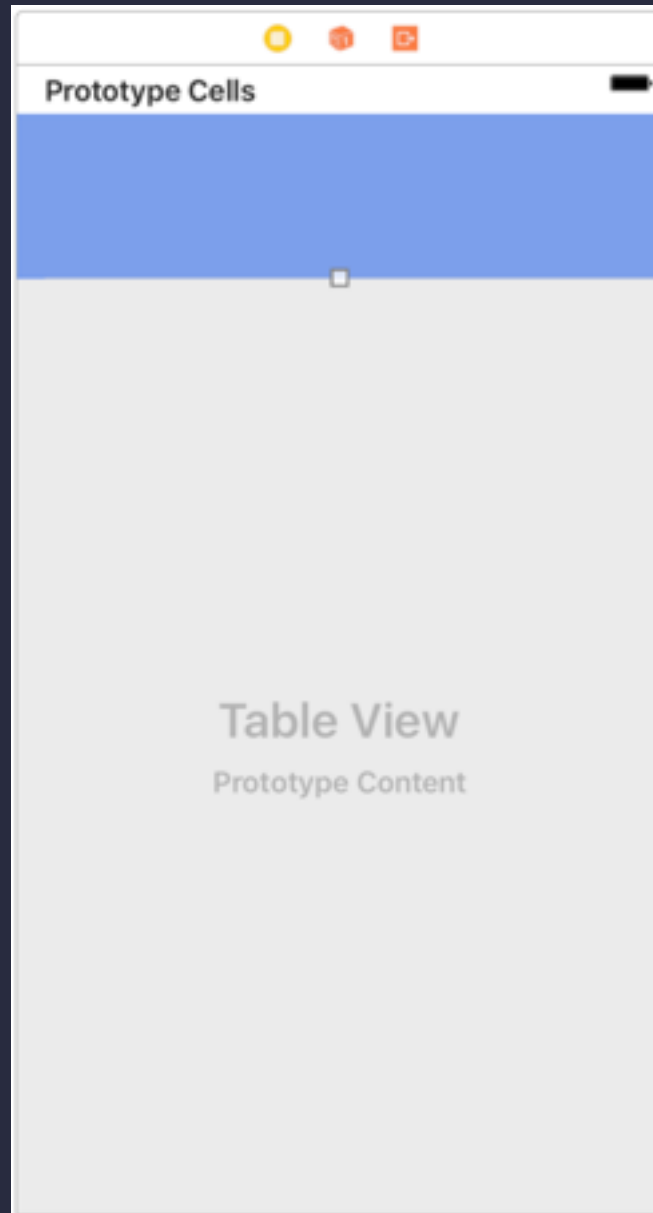



# // storyboard


**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.





▼  **Table View Controller Scene**


▼  **Table View Controller**

▼  **Table View**

▼  **Table View Cell**

 **Content View**

 **First Responder**

 **Exit**

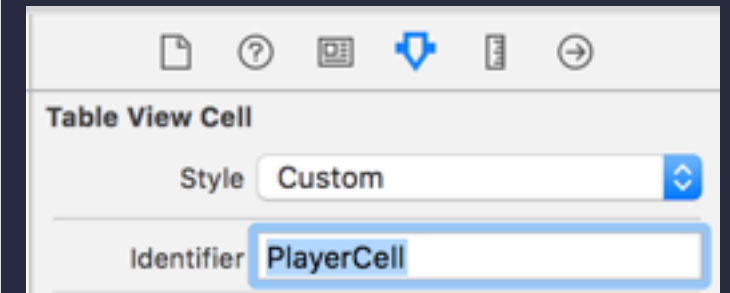
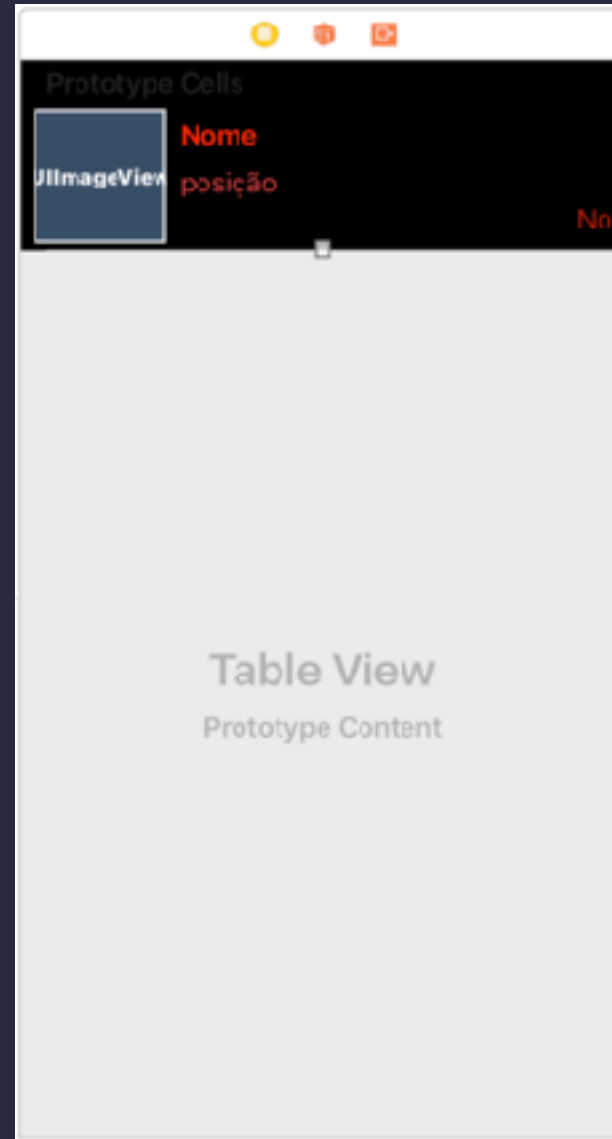
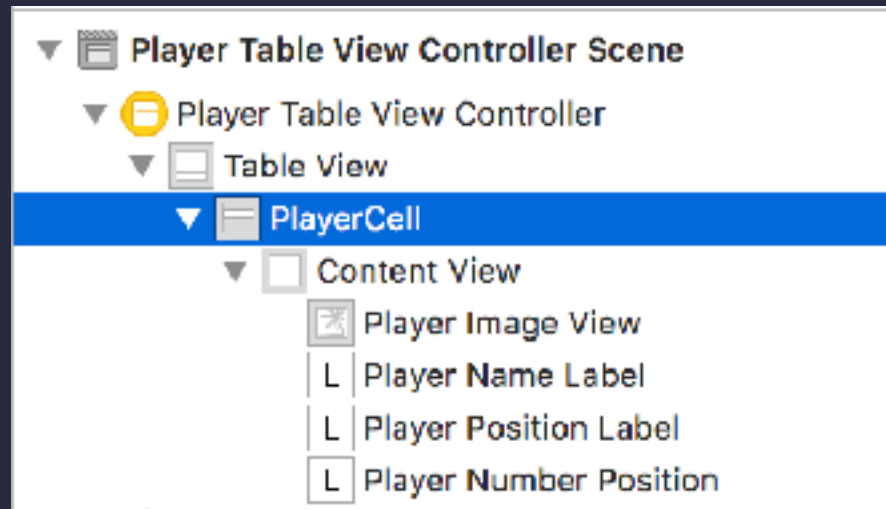
// Implementação

// tipo dos dados

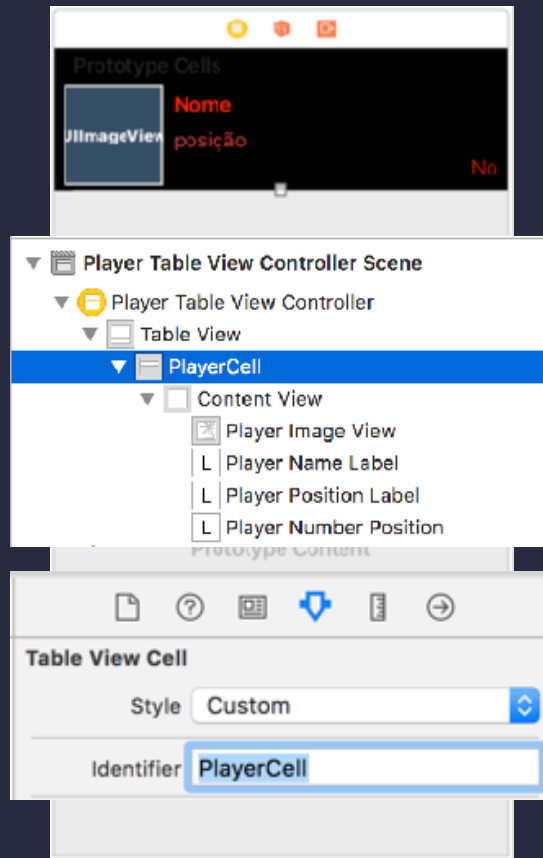
```
class Player {  
    var nome: String  
    var posicao: String  
    var numero: Int  
    var foto: UIImage  
  
    init(nome: String, posicao: String, numero: Int, foto: UIImage) {  
        self.nome = nome  
        self.posicao = posicao  
        self.numero = numero  
        self.foto =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



// células



// 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

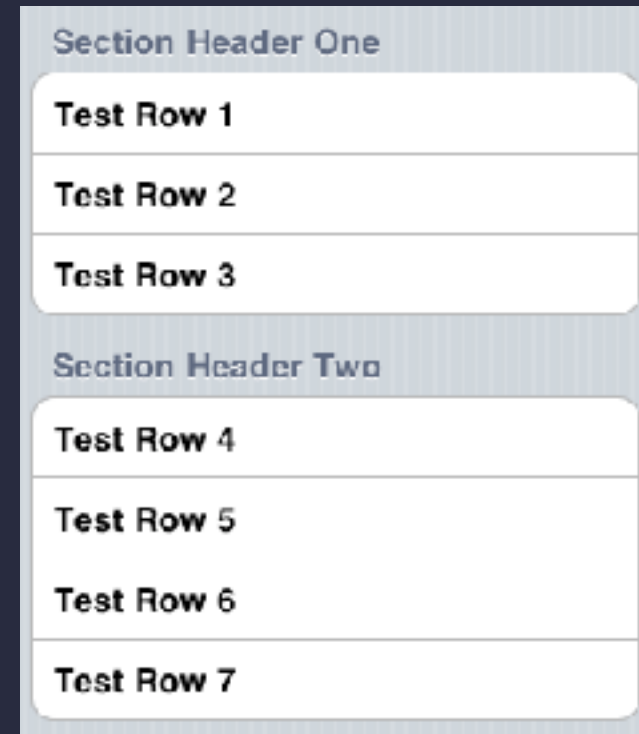
```
class PlayerTableViewController: UITableViewController {  
    var players: [Player] = []  
  
    override func viewDidLoad() {  
        super.viewDidLoad()  
  
        self.loadPlayers()  
    }  
  
}
```

// data source

- protocolo `UITableViewDataSource`
- proveedor de datos
- implementado por `UITableViewController`

// data source

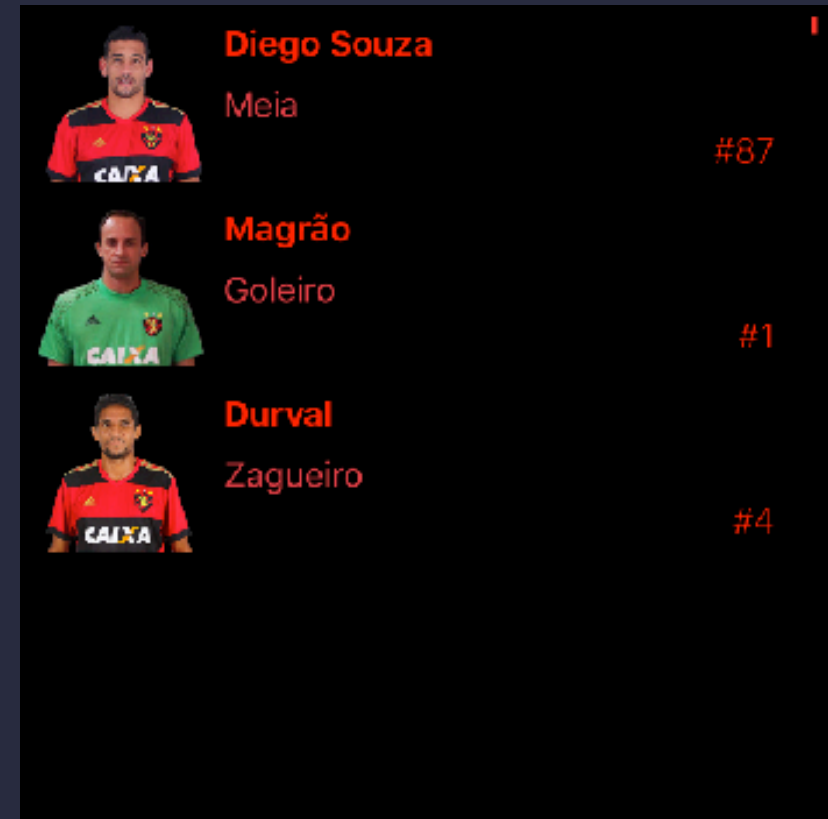
```
class PlayerTableViewController: UITableViewController {  
  
    override func numberOfSections(in tableView: UITableView) -> Int {  
        // return the number of sections  
        return 1  
    }  
}
```



Section Header One	
	Test Row 1
	Test Row 2
	Test Row 3
Section Header Two	
	Test Row 4
	Test Row 5
	Test Row 6
	Test Row 7

```
// 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
```

```
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 {
```

```
}
```

// data source

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

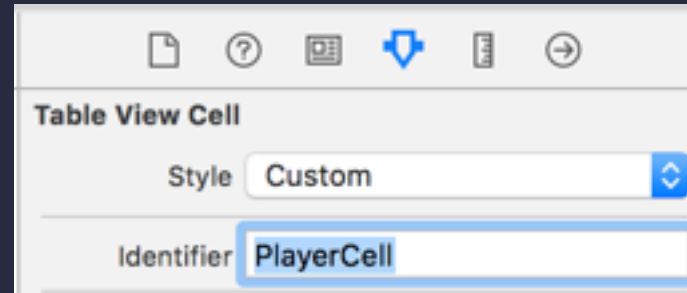
```
    let cellIdentifier = "PlayerCell"
```

```
}
```

// data source

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

```
    let cellIdentifier = "PlayerCell"
```



```
}
```

// data source

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

```
    let cellIdentifier = "PlayerCell"
```

```
}
```

# // data source

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

```
    let cellIdentifier = "PlayerCell"
```

```
    guard let cell = tableView.dequeueReusableCell(withIdentifier: cellIdentifier,  
                                                for: indexPath)
```

```
}
```

# // data source

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

```
    let cellIdentifier = "PlayerCell"
```

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

```
}
```

# // data source

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

    let cellIdentifier = "PlayerCell"

    guard let cell = tableView.dequeueReusableCell(withIdentifier: cellIdentifier,
                                                for: indexPath) as? PlayerTableViewCell
    else {
        fatalError("Tipo de célula não é PlayerTableViewCell")
    }

}
```

## // data source

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

    let cellIdentifier = "PlayerCell"

    guard let cell = tableView.dequeueReusableCell(withIdentifier: cellIdentifier,
                                                for: indexPath) as? PlayerTableViewCell
    else {
        fatalError("Tipo de célula não é PlayerTableViewCell")
    }

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

}
```



## // data source

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

    let cellIdentifier = "PlayerCell"

    guard let cell = tableView.dequeueReusableCell(withIdentifier: cellIdentifier,
                                                for: 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)

}
```

## // data source

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

    let cellIdentifier = "PlayerCell"




    guard let cell = tableView.dequeueReusableCell(withIdentifier: cellIdentifier,
                                                for: 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

	<b>Diego Souza</b> Meia	#87
	<b>Magrão</b> Goleiro	#1
	<b>Durval</b> Zagueiro	#4

// resultado final

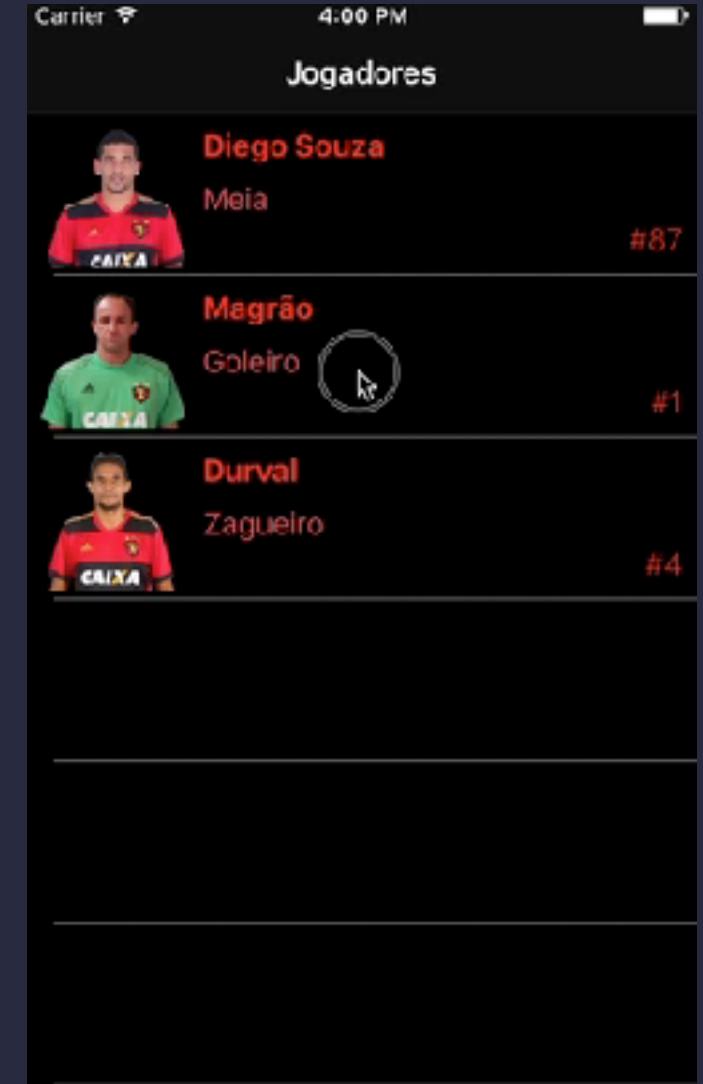
	<b>Diego Souza</b> Meia	#87
	<b>Magrão</b> Goleiro	#1
	<b>Durval</b> Zagueiro	#4

# // Exercício

## // Exercício 13

# Lista de coisas

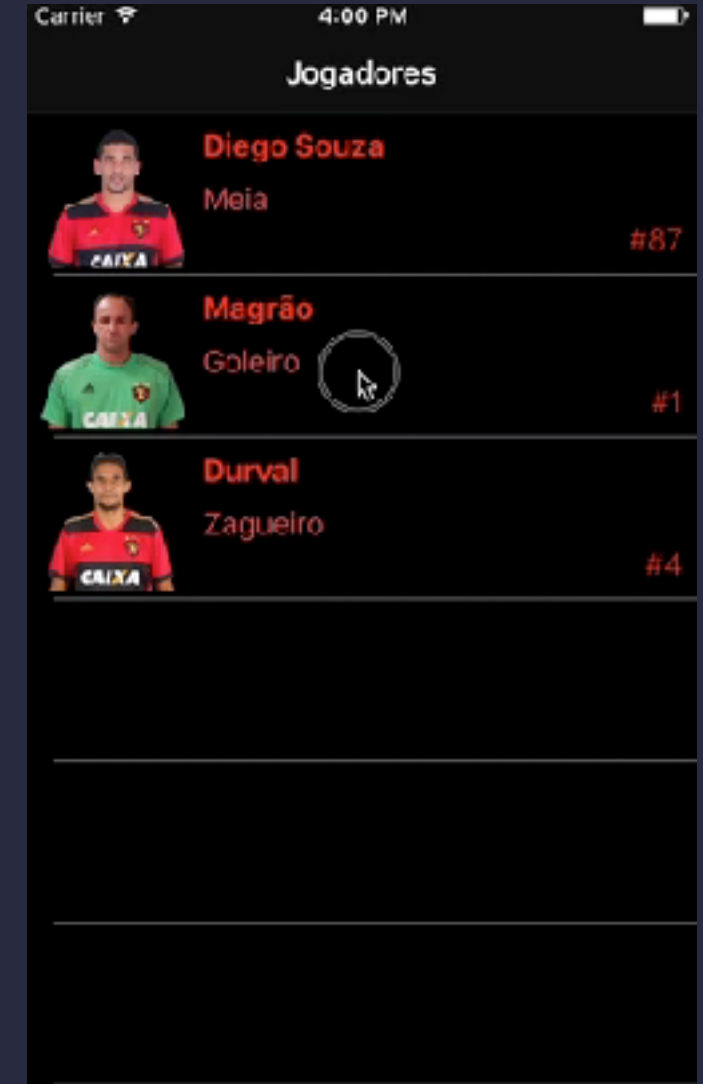
1. Faça uma **Table View** que mostre pelo menos 3 **itens** de uma lista (contendo pelo menos **foto** e **nome**)
2. ao tocar num item, **direcione o usuário** para uma tela com os detalhes do item selecionado
  - 2.1. Permita que o usuário possa dar **zoom** na foto mostrada



## // 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






// Edições



// remoção

Jogadores		
	<b>Diego Souza</b> Meia	#87
	<b>Magrão</b> Goleiro	#1
	<b>Durval</b> Zagueiro	#4




// remoção

Jogadores		
	<b>Diego Souza</b> Meia	#87
	<b>Magrão</b> Goleiro	#1
	<b>Durval</b> Zagueiro	#4




// remoção

```
override func tableView(_ tableView: UITableView,  
    commit editingStyle: UITableViewCellEditingStyle,  
    forRowAt indexPath: IndexPath) {  
  
    if editingStyle == .delete {  
        // Delete the row from the data source  
        self.players.remove(at: indexPath.row)  
        tableView.deleteRows(at: [indexPath], with: .fade)  
    }  
}
```

// inserção

Jogadores			+
	<b>Diego Souza</b> Meia	#87	
	<b>Magrão</b> Goleiro	#1	
	<b>Durval</b> Zagueiro	#4	

// inserção

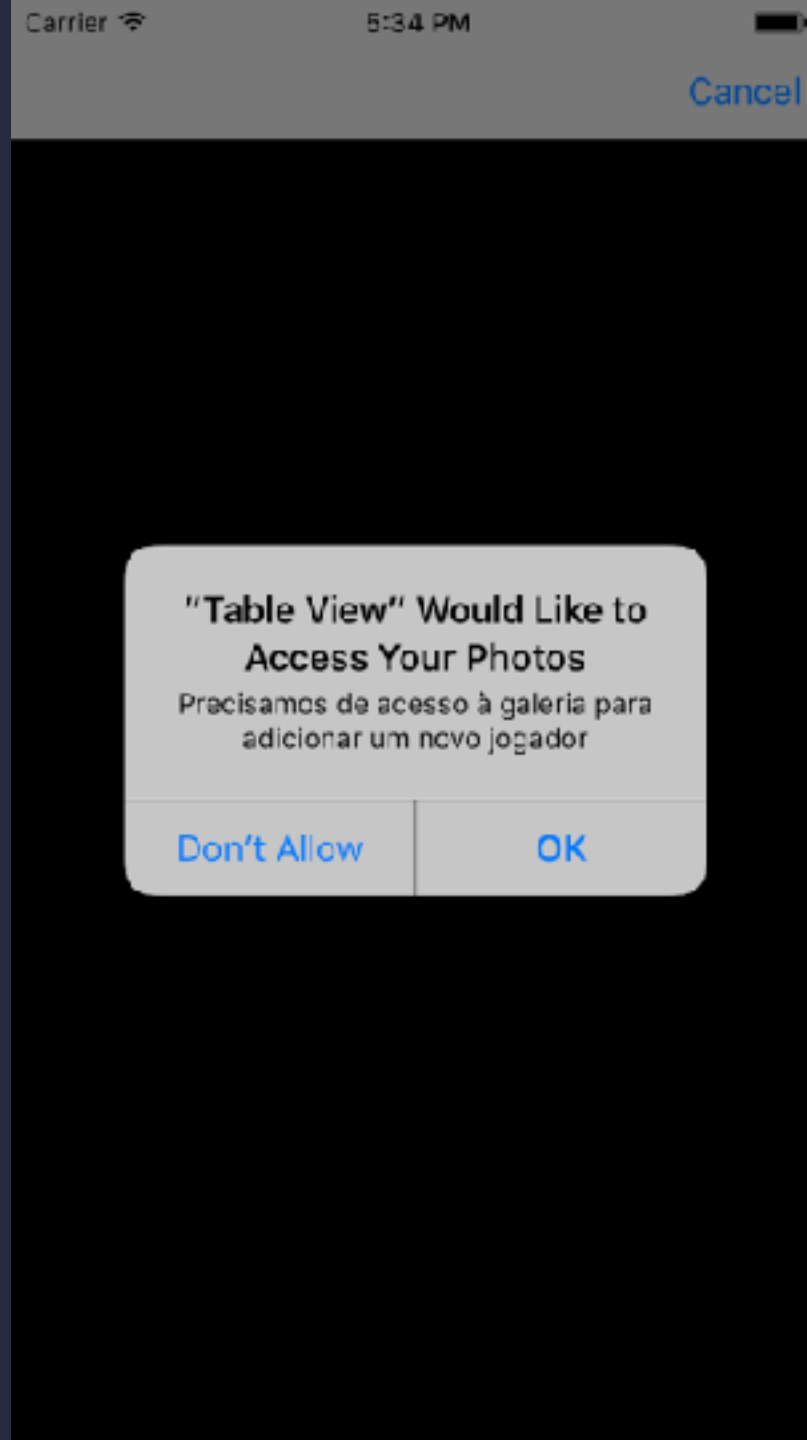
Jogadores			+
	<b>Diego Souza</b> Meia	#87	
	<b>Magrão</b> Goleiro	#1	
	<b>Durval</b> Zagueiro	#4	

// inserção

```
@IBAction func addPlayer(_ sender: Any) {  
    let newIndexPath = IndexPath(row: players.count, section: 0)  
  
    let andre = Player(nome: "André",  
                        posicao: "Atacante",  
                        numero: 90,  
                        foto: #imageLiteral(resourceName: "andre"))  
  
    players.append(andre)  
  
    self.tableView.insertRows(at: [newIndexPath], with: .automatic)  
}
```

*// Accesso à galleria*

// permissão





## // info.plist

Key		Type	Value
▼ Information Property List		Dictionary	(15 items)
Localization native development region	⬆	String	en ⬆
Executable file	⬆	String	\$(EXECUTABLE_NAME)
Bundle identifier	⬆	String	\$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	⬆	String	6.0
Bundle name	⬆	String	\$(PRODUCT_NAME)
Bundle OS Type code	⬆	String	APPL
Bundle versions string, short	⬆	String	1.0
Bundle version	⬆	String	1
Application requires iPhone environment	⬆	Boolean	YES ⬆
Launch screen interface file base name	⬆	String	LaunchScreen
Main storyboard file base name	⬆	String	Main
▶ Required device capabilities	⬆	Array	(1 item)
Status bar is initially hidden	⬆	Boolean	NO ⬆
Privacy - Photo Library Usage Description	⬆ + -	String	⬆ Precisamos de acesso à galeria para adicionar um novo jogador
▶ Supported interface orientations	⬆	Array	(3 items)

// image picker



// abrindo galeria

```
if UIImagePickerController.isSourceTypeAvailable(.photoLibrary) {  
    let imagePicker = UIImagePickerController()  
  
    imagePicker.delegate = self  
  
    imagePicker.sourceType = .photoLibrary  
  
    imagePicker.allowsEditing = true  
  
    self.present(imagePicker, animated: true, completion: nil)  
}
```

// capturando imagem selecionada

```
extension PlayerTableViewController: UIImagePickerControllerDelegate ,
                                   UINavigationControllerDelegate {

    func imagePickerController(_ picker: UIImagePickerController,
                               didFinishPickingMediaWithInfo info: [String : Any]) {

        if let pickedImage = info[UIImagePickerControllerOriginalImage] as? UIImage {

            let imagemSelecionada = pickedImage
            // adicionar na tabela
        }
        picker.dismiss(animated: true, completion: nil)
    }
}
```

# // Exercício

## // Exercício 14

# Lista de coisas II

1. ponha um botão de **adicionar**, que leva o usuário para uma tela onde ele escreverá um **nome**, e uma **foto** para um novo item a ser **inserido** na tabela tabela.
2. permita que o usuário possa **deletar** um item da tabela

## // Extra

3. permita que o usuário possa **editar** um item da tabela

// Ementa

## **Swift:**

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

## **Storyboard:**

- Model
- View
- Controller
  
- Delegação
- UIKit
  - Labels
  - Botões
  - Textfields
  - Tableview
- Storyboard
- Navegação
- Persistência local



## Swift:

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

## Storyboard:

- ~~— Model~~
- ~~— View~~
- ~~— Controller~~
- ~~— 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

