



### // Aula 05



## // Recapitulando





- 1. Definição do protocolo
- 2. Implementar protocolo em uma classe
- 3. adicionar propriedade delegate
- 4. chamar métodos do delegate



```
import UIKit

class ViewController: UIViewController {
    @IBOutlet weak var colorTextField: UITextField!
    @IBOutlet weak var colorButton: UIButton!

@IBAction func tapColorButton(_ sender: Any) {
    // ...
}
```



```
extension ViewController: UITextFieldDelegate {
  func textField(_ textField: UITextField,
                  shouldChangeCharactersIn range: NSRange,
                  replacementString string: String) -> Bool {
       let text = textField.text!
       let swiftRange = Range(range, in: text)!
       let newText = text.replacingCharacters(in: swiftRange, with: string)
       if newText != "" {
           self.colorButton.isEnabled = true
       } else {
           self.colorButton.isEnabled = false
       return true
```



```
class ViewController: UIViewController {
  @IBOutlet weak var colorTextField: UITextField!
  override func viewDidLoad() {
     super.viewDidLoad()
     self.colorTextField.delegate = self
```



- 1. UITextFieldDelegate
- 2. ViewController: UITextFieldDelegate
- 3. self.colorTextField.delegate = self
- 4. chamar métodos do delegate (responsabilidade do Text Field)



```
public protocol UITextFieldDelegate : NSObjectProtocol {
  @available(iOS 2.0, *)
  optional public func textFieldShouldBeginEditing( textField: UlTextField) -> Bool // return NO to disallow editing.
  @available(iOS 2.0, *)
  optional public func textFieldDidBeginEditing(_ textField: UITextField) // became first responder
  @available(iOS 2.0, *)
  optional public func textFieldShouldEndEditing(_ textField: UITextField) -> Bool // return YES to allow editing to stop and to resign
first responder status. NO to disallow the editing session to end
  @available(iOS 2.0, *)
  optional public func textFieldDidEndEditing(_ textField: UITextField) // may be called if forced even if shouldEndEditing returns NO
(e.g. view removed from window) or endEditing:YES called
  @available(iOS 10.0, *)
  optional public func textFieldDidEndEditing(_ textField: UlTextField, reason: UlTextFieldDidEndEditingReason) // if implemented,
called in place of textFieldDidEndEditing:
```

## // Navegação



Trocar de tela

Trocar de View

Trocar de View Controller

iOS Simulator - iPhone 6 - iPhone 6 / iOS 8.1 (12B411) View Controller #1

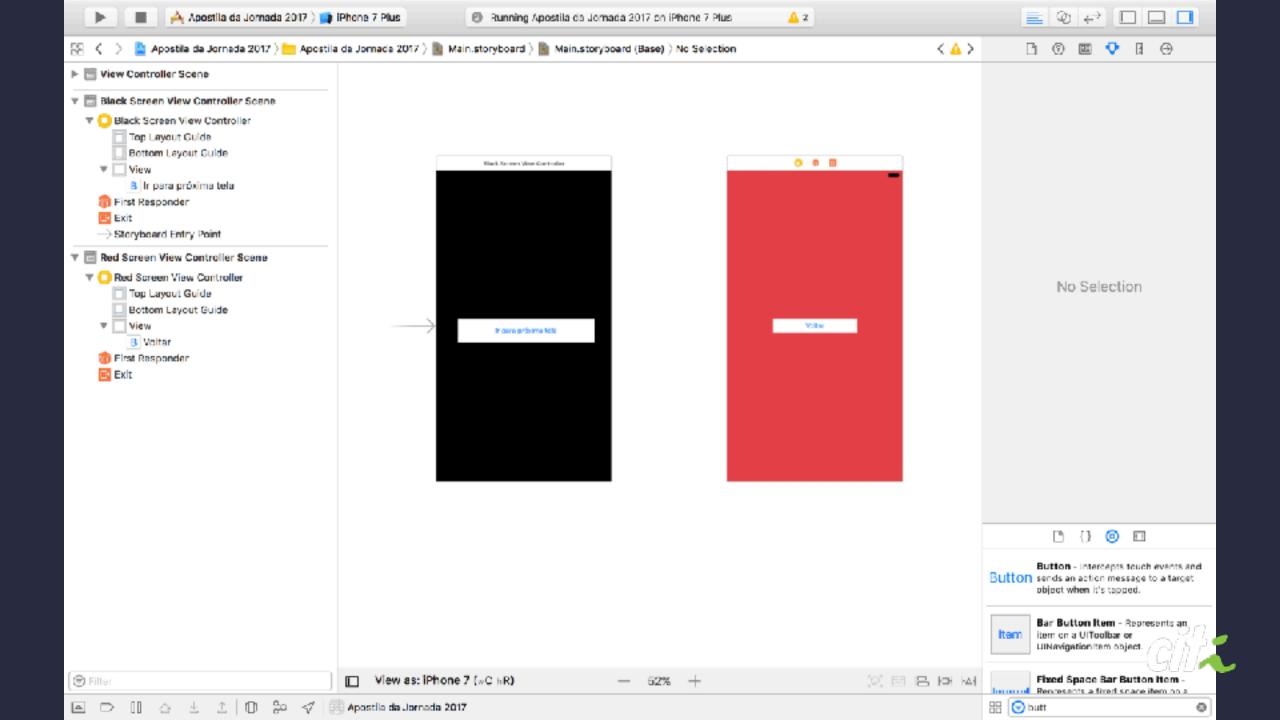


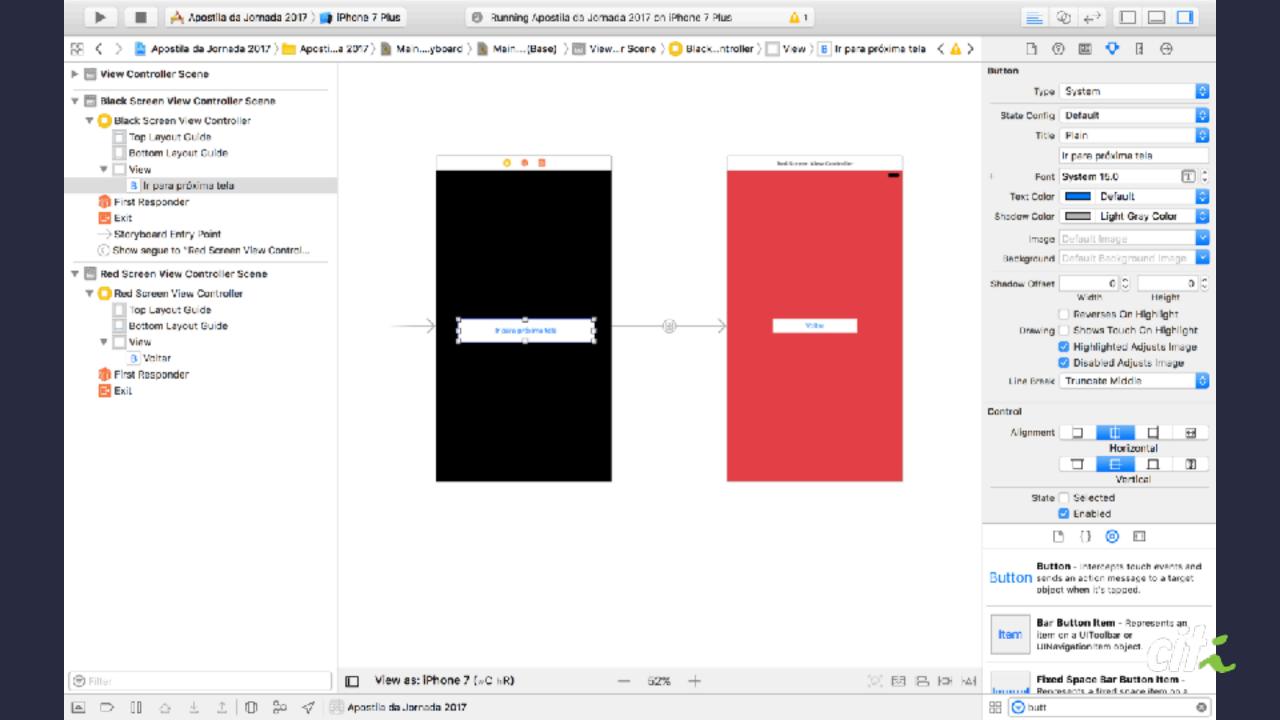




- Definem o fluxo de navegação do app
- Transição entre ViewControllers do StoryBoard
- Início: button, table row, or gesture recognizer
- **Destino**: ViewController a ser mostrado



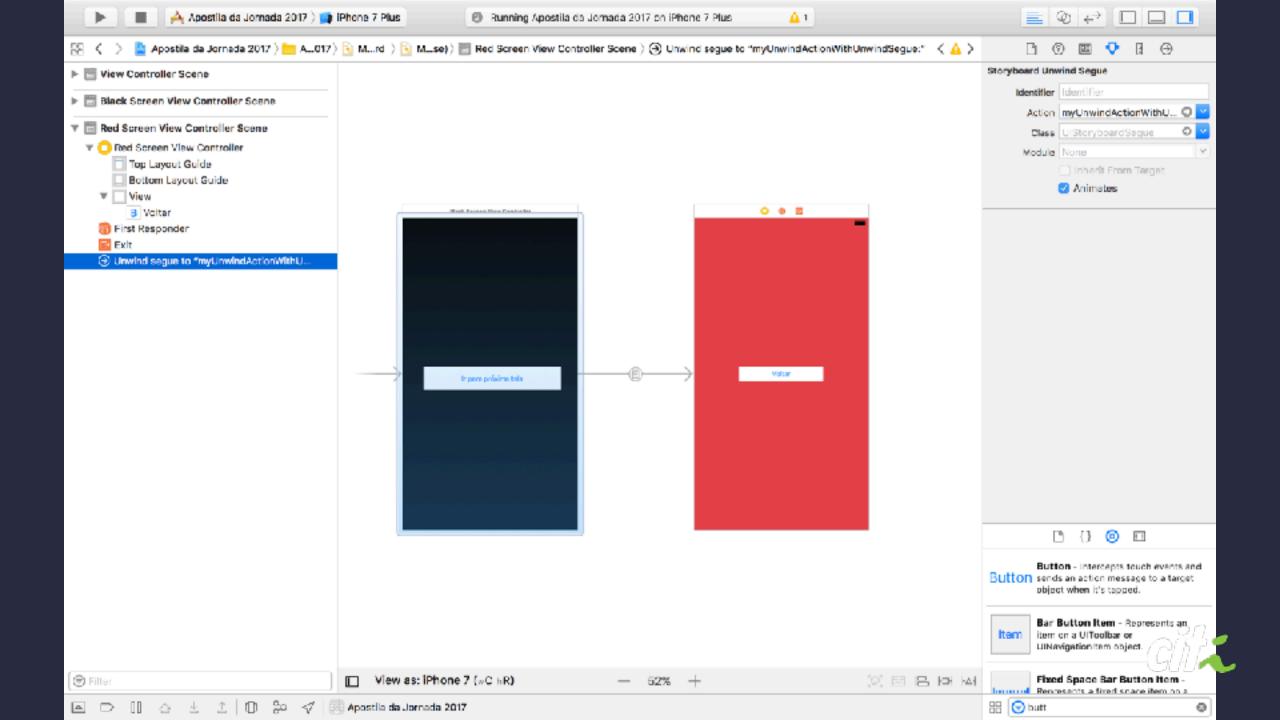




Apresentação Modal

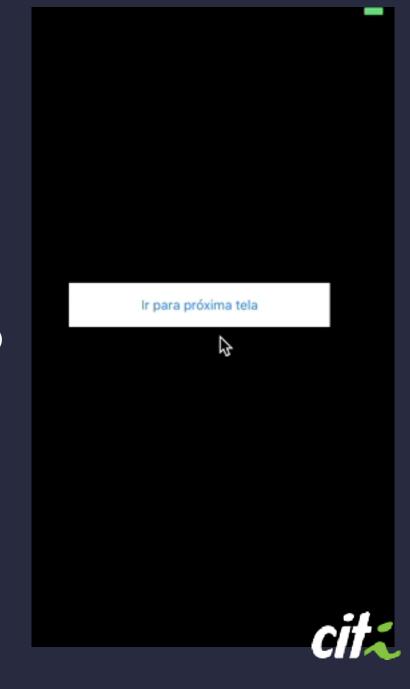






Unwind

@IBAction func myUnwindAction(unwindSegue: UIStoryboardSegue)



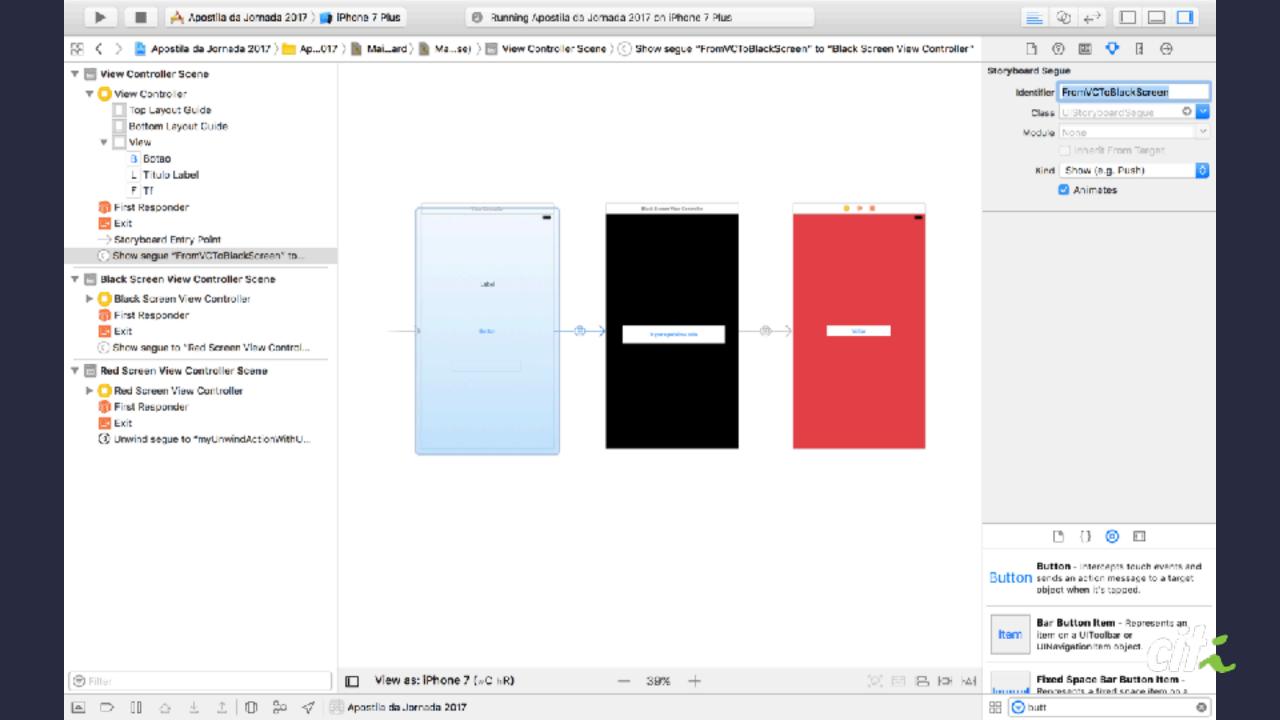
## // Chamando por Código

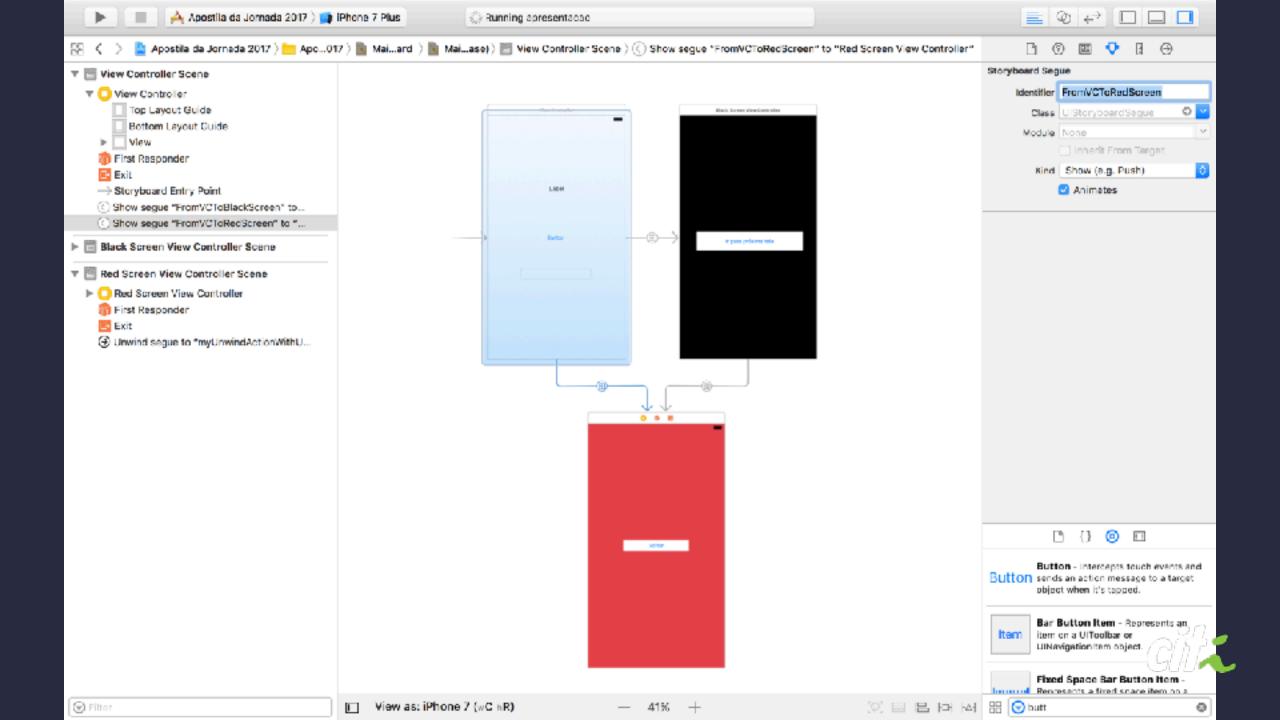


### // Segues programáticas

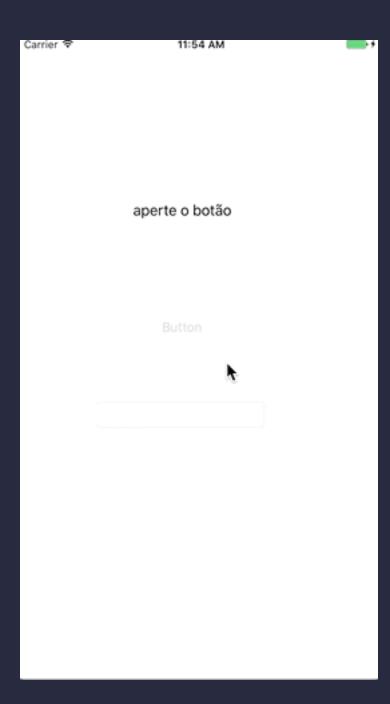
Às vezes é necessário **chamar** uma segue pelo código







### // chamando segue





### // chamando segue

```
if tituloLabel.text == "Black" {
    self.performSegue(withIdentifier: "FromVCToBlackScreen", sender: self)
} else if tituloLabel.text == "Red"{
    self.performSegue(withIdentifier: "FromVCToRedScreen", sender: self)
}
```



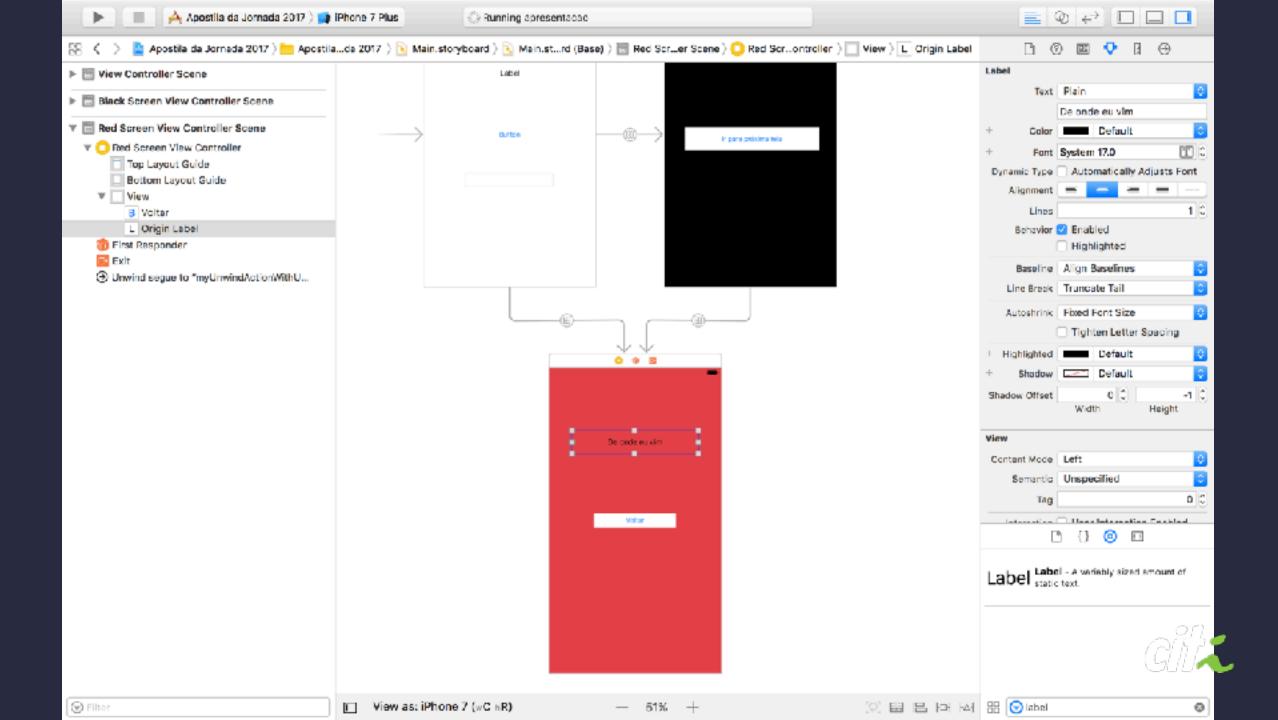
### // Passando Dados



### // passando dados

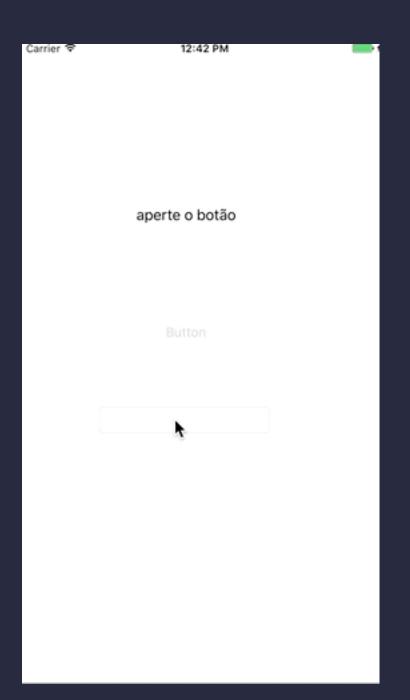
- Se precisarmos passar informações entre view controllers
- Podemos passá-las pela segue





// pass

12:40 PM aperte o botão





### // passando dados

```
class RedScreenViewController: UIViewController {
    var recievedData: String?
   @IBOutlet weak var originLabel: UILabel!
    override func viewDidLoad() {
        super.viewDidLoad()
        self.originLabel.text = self.recievedData
```



### // passando dados

#### No ViewController:

```
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
   if let redVC = segue.destination as? RedScreenViewController {
      redVC.recievedData = "Vim do VC"
   }
}
```

#### No BlackScreenViewController:

```
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
   if let redVC = segue.destination as? RedScreenViewController {
      redVC.recievedData = "Vim do BlackVC"
   }
}
```

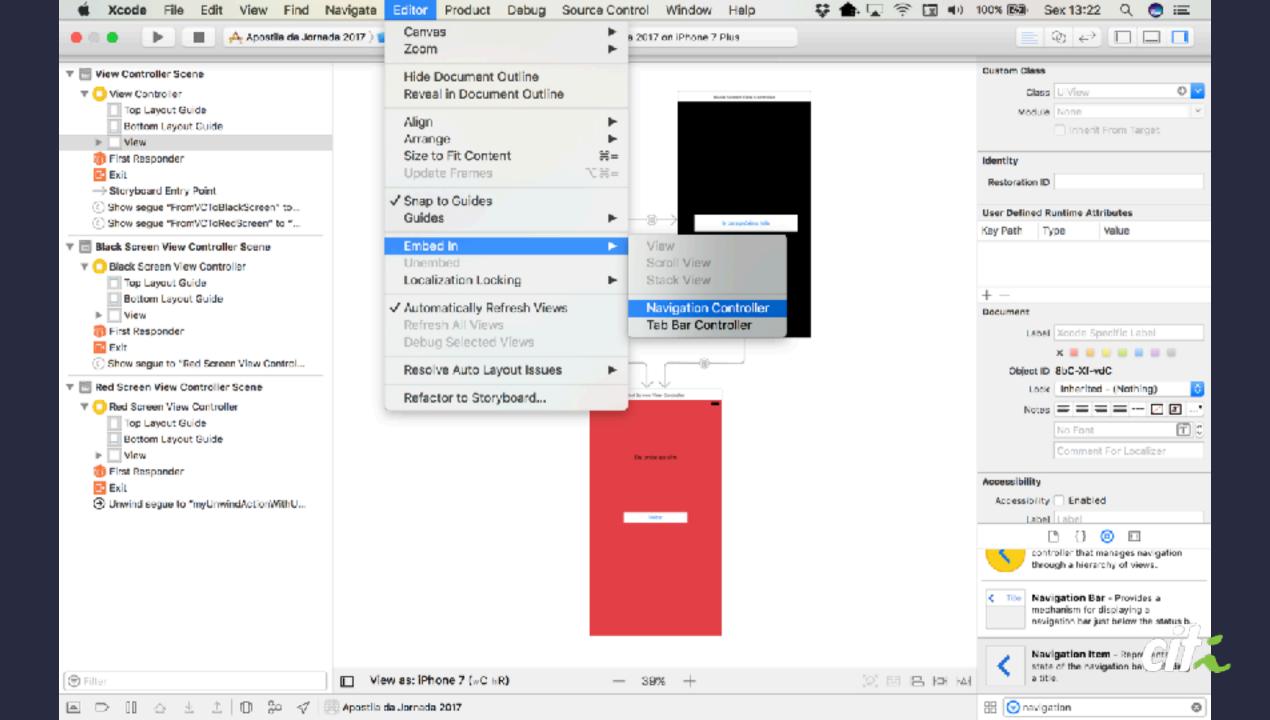


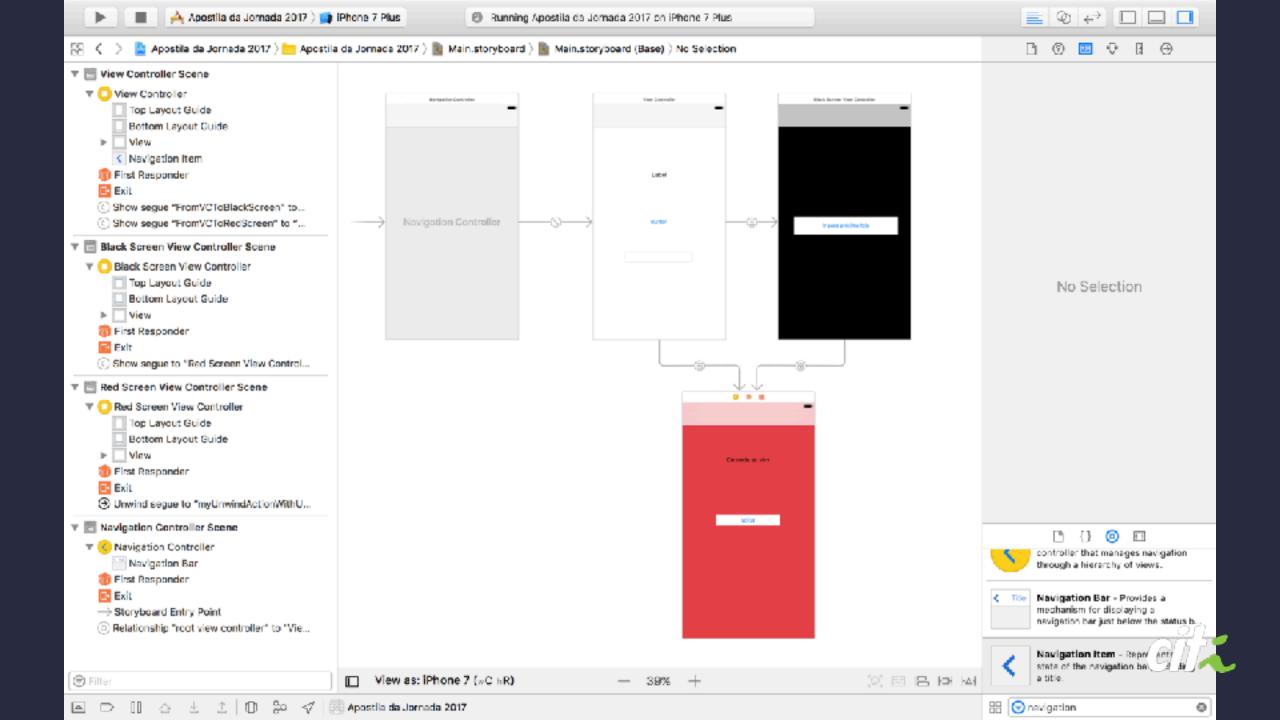
## // Navigation Controller

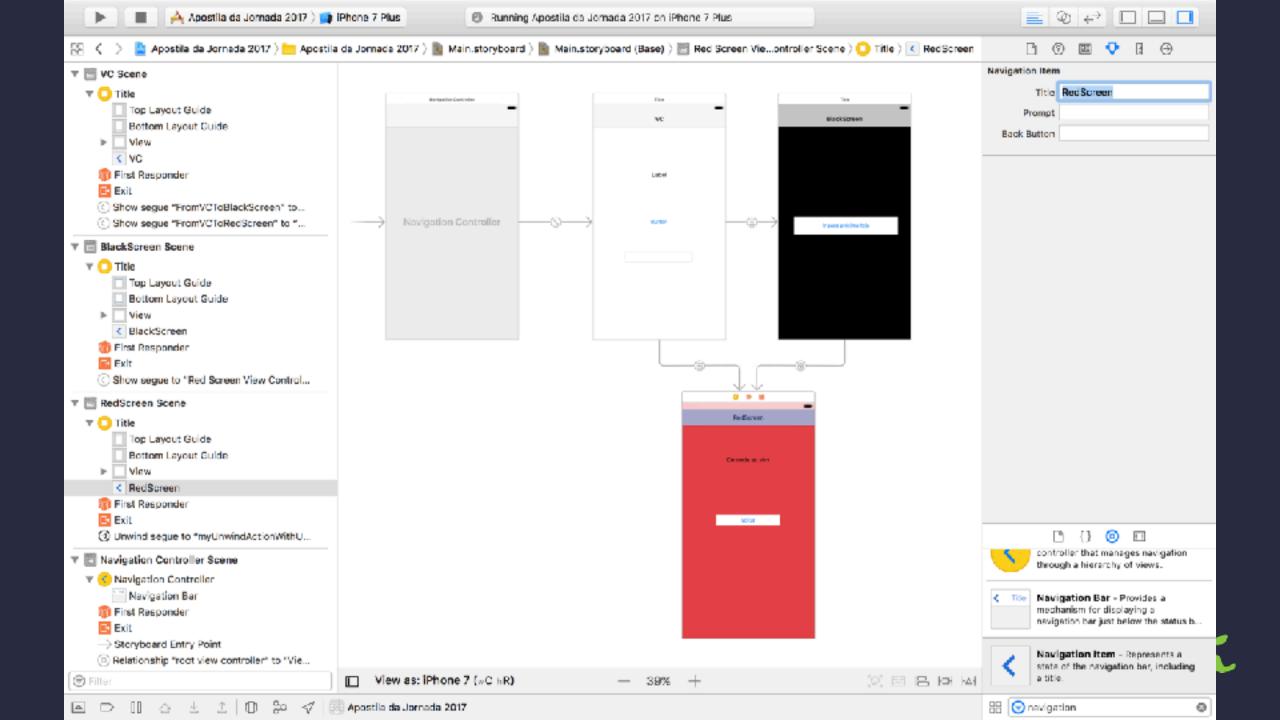






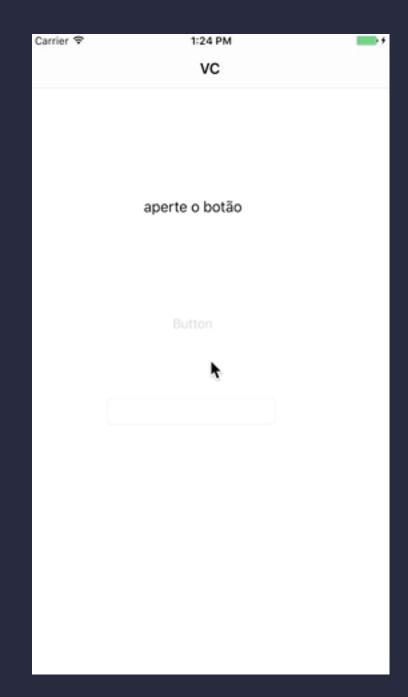




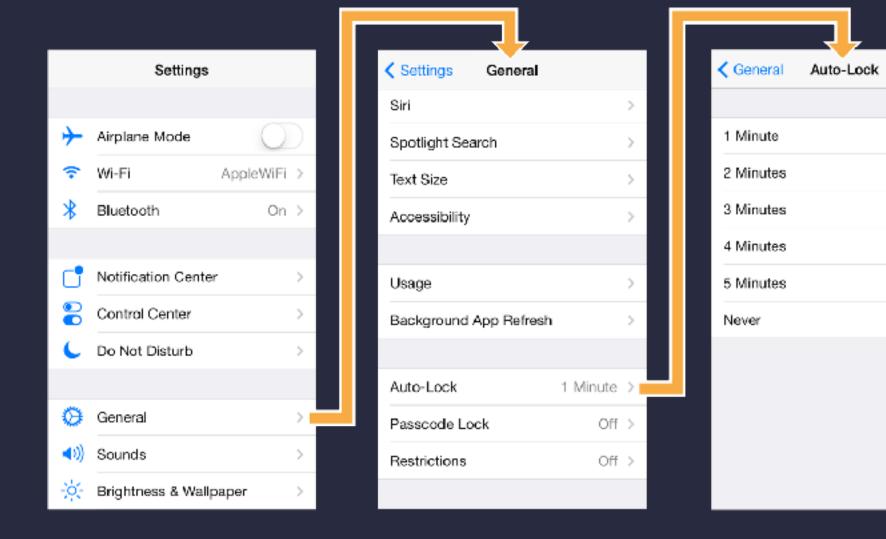


### // Navigation controller

- UINavigationController
- Navegação hierárquica
- Pilha de VCs

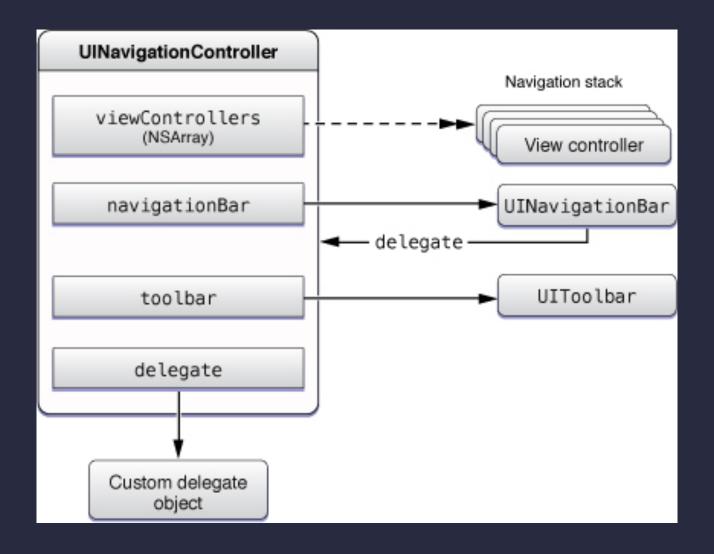


### // Navigation controller





### // Navigation controller





## // Exercício



#### // Exercício 11-a

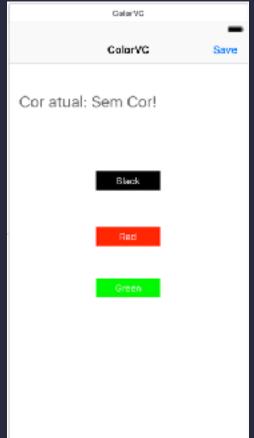
### Navegação

MainVC

Cor atual: Sem Cor!

Apertar em Escolher cor leva para ColorVC

O background de MainVC deve ser da cor escolhida, em ColorVC, e a label deve indicar o nome da cor.



Apertar em um botão de cor muda o conteúdo da label.

Apertar em save retorna para o MainVC, passando a **cor** escolhida



#### // Exercício 11-b

### Navegação com Delegate

MainVC

MainVC

Cor atual: Sem Cor!

Escolher cor

Implementa um protocolo ColorVCDelegate:

func chosen(color: UlColor,

by colorVC: ColorViewController)

retira o
ColorViewController da
pilha de navegação e
atualiza interface do
MainViewController



tem uma propriedade
delegate do tipo
ColorVCDelegate

ao apertar save, chama método chosen(color: by:) de seu **delegate** 



