Delegation

Mobile Computing - iOS

Objectives

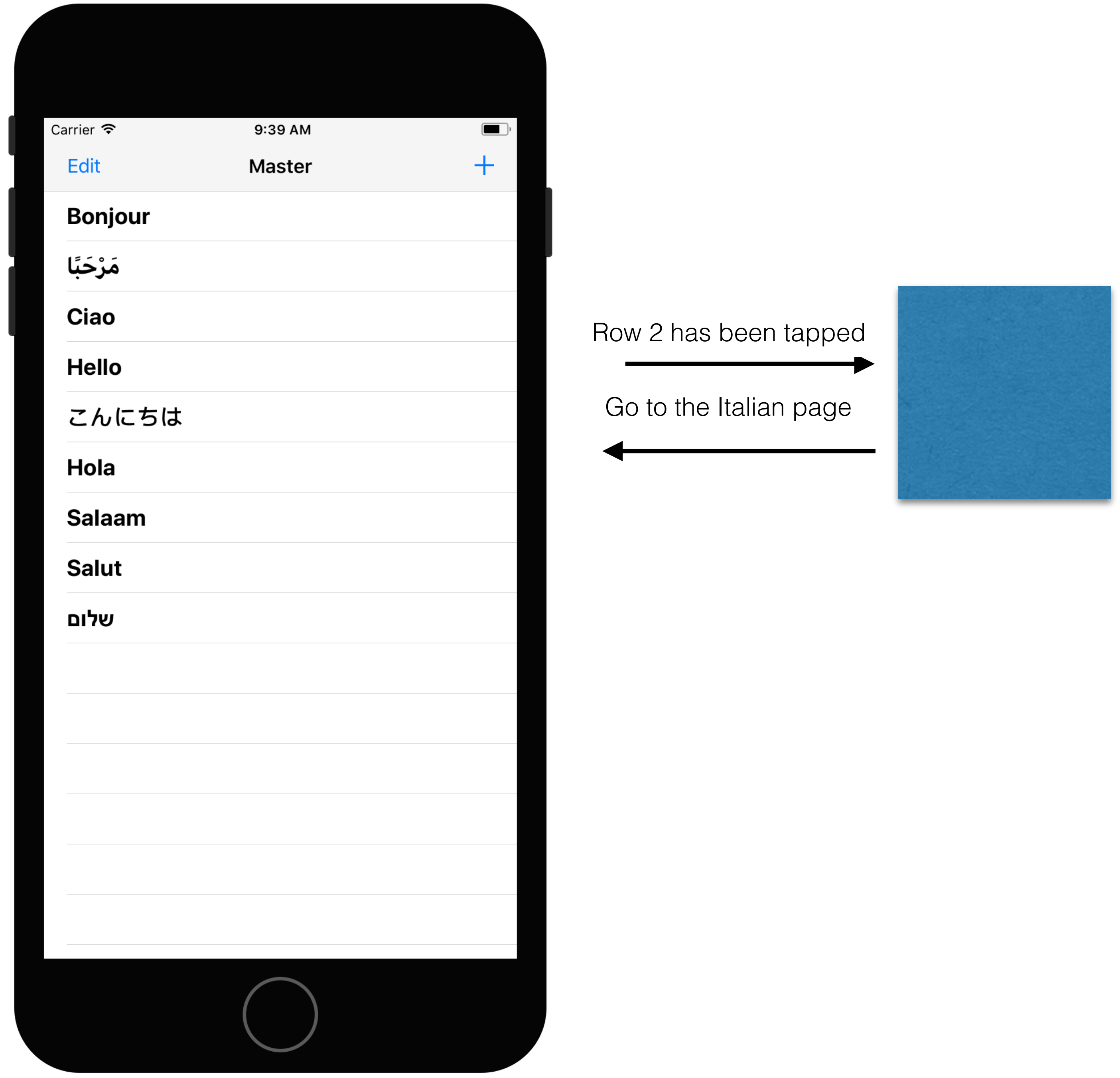
- Students will be able to:
    - explain what delegation is
    - explain the purpose of delegation
    - use protocols to create delegates

Delegation: Common Usage

- In English, a **delegate** is
    - somebody you send in your place (to a conference or meeting), and entrust to make decisions for you ("the UN delegate from China")
    - somebody you hand off a task to ("delegate responsibility"), to avoid being overwhelmed

Delegation: iOS Usage

- Delegation is technique for customizing the behavior of a class without resorting to subclassing. It is a design pattern* used by Apple in most of its frameworks.
- Instead of subclassing and overriding a method, identify a delegate — a separate object — and delegate authority to that object
- The delegating class is usually part of the UIKit frameworks; the delegate is a custom class that the developer writes to meet their needs
- A protocol is used to make sure everything works properly. The delegate adheres to that protocol, and the delegating class stores a reference to the delegate using the protocol as the type



*a design pattern is a solution to a frequently encountered problem in software development, expressed in a high level (as opposed to code)

A Lunchtime Example

- **FoodChooser Protocol**

  ```
  func whereShouldWeDine() -> String
  ```

- **Professor**

  ```
  var delegate:FoodChooser!

  func timeForLunch() // will ask its delegate for advice
  ```

- **Decider** — any object that implements the FoodChooser protocol: it will be the delegate for the professor

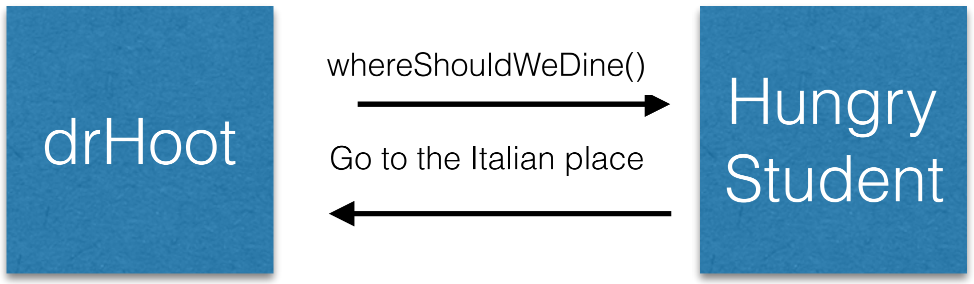
A Lunchtime Example

```
protocol FoodChooser{

    func whereShouldWeDine()->String
}

class Professor {
    var delegate:FoodChooser!

    func timeForLunch(){
        if delegate != nil {
          print("let's go eat at ... \(delegate.whereShouldWeDine())")
        }
    }

}
```

A Lunchtime Example, Cont'd

```
class HungryStudent : FoodChooser {
    func whereShouldWeDine() -> String{ return "Luigi's"}
}

class BuffetLover : FoodChooser {
    let names:[String] = ["Pizza Hut", "Paliais", "Pizza Ranch"]
    func whereShouldWeDine() -> String{
        return names[Int(arc4random()) % names.count]
    }
}

var drHoot:Professor = Professor()
drHoot.delegate = HungryStudent()       // Dr. Hoot will let HS decide

var drCase:Professor = Professor()
drCase.delegate = BuffetLover()         // Dr. Case will let BL decide

drHoot.timeForLunch()
drCase.timeForLunch()
```

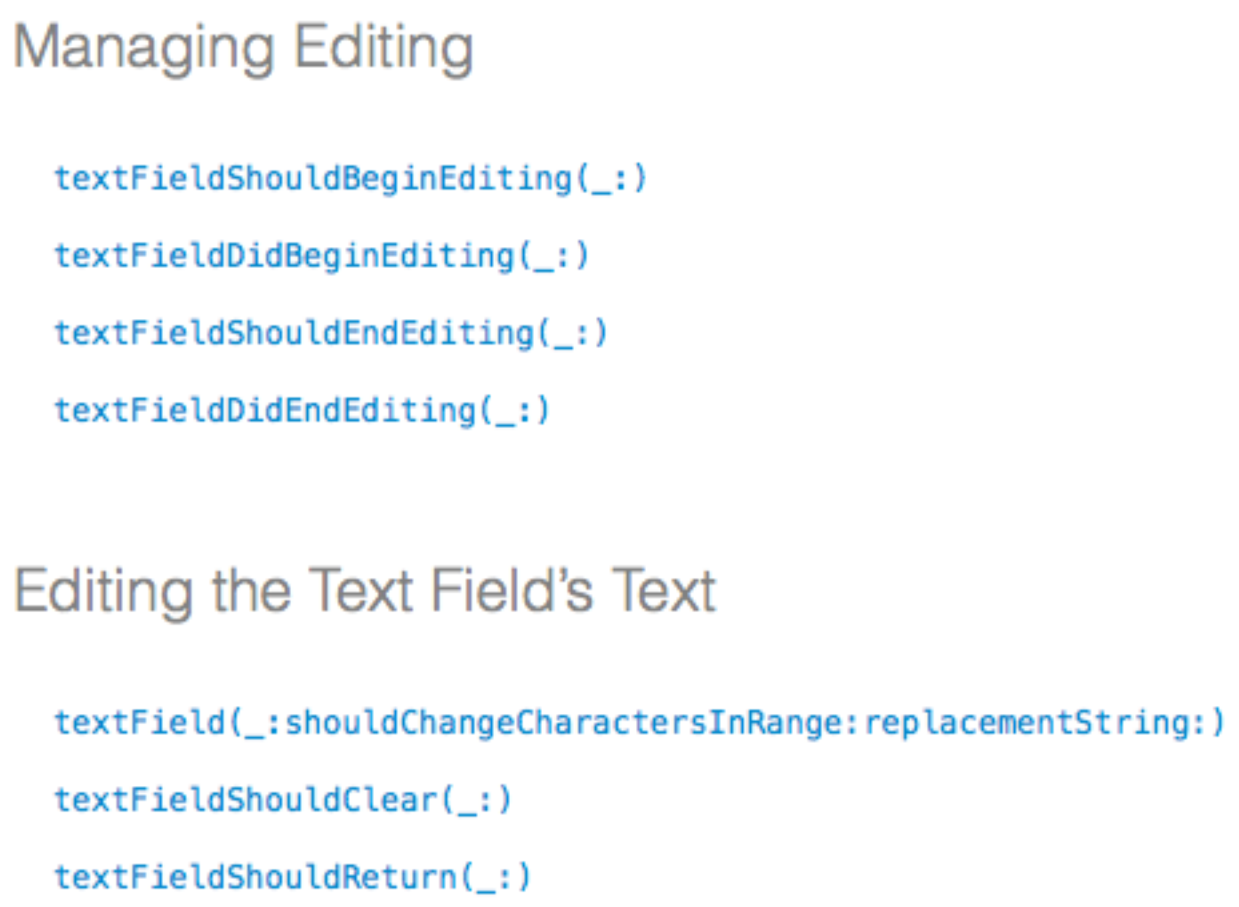
arc4random() returns an Int32, but names.count is an Int, hence the little inconvenience ...



A UIKit Example

- UITextFieldDelegate
- What happens when the user taps return after interacting with a UITextField?
- The UITextField doesn't decide — it asks its delegate

The UITextFieldDelegate Protocol



Managing Editing

`textFieldShouldBeginEditing(_:)`

`textFieldDidBeginEditing(_:)`

`textFieldShouldEndEditing(_:)`

`textFieldDidEndEditing(_:)`

Editing the Text Field's Text

`textField(_:shouldChangeCharactersInRange:replacementString:)`

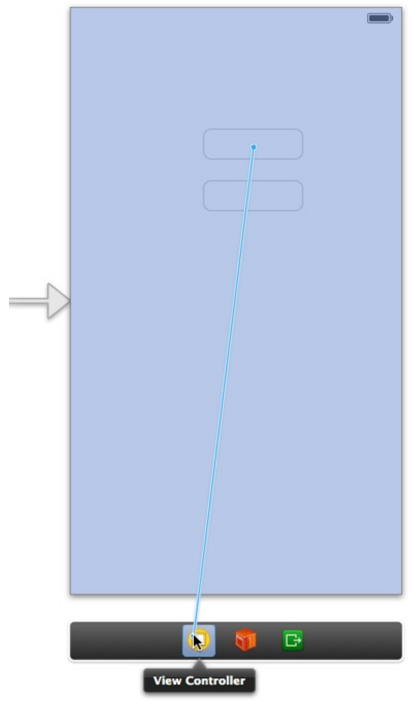
`textFieldShouldClear(_:)`

`textFieldShouldReturn(_:)`

Methods called at various points as the user interacts with the text field (see docs for details)

All methods are optional

How to Set up a UITextField Delegate in Storyboard



1. Ctrl-drag from UITextField to a UIViewController

2. Make the UIViewController adhere to UITextFieldDelegate protocol

```
class MyViewController : UIViewController, UITextFieldDelegate
```

3. Implement textFieldShouldReturn:

```
func textFieldShouldReturn(textField:UITextField) -> Bool{
    //if(textField == self.topTF) {
    textField.resignFirstResponder()
    return true
}
```

How to Setup a UITextFieldDelegate in Code

- Step 1 (connecting the UITextField to its delegate) changes from the previous slide -- steps 2 and 3 remain the same

1. In the viewDidLoad() method of MyViewController with a reference to a UITextField (say, topTF), include this line: `topTF.delegate = self`

2. Make the UIViewController adhere to UITextFieldDelegate protocol

```
class MyViewController : UIViewController, UITextFieldDelegate
```

3. Implement textFieldShouldReturn:

```
func textFieldShouldReturn(textField:UITextField) -> Bool{
   textField.resignFirstResponder()
    return true
}
```

Resources

- https://developer.apple.com/library/content/documentation/General/Conceptual/DevPedia-CocoaCore/Delegation.html