CS 329E
Elements of Mobile Computing

Spring 2018
University of Texas at Austin

Lecture 9

Agenda

- Segmented Control
- Swift Extensions
- Custom Protocols and Delegates
- Mockups Paper

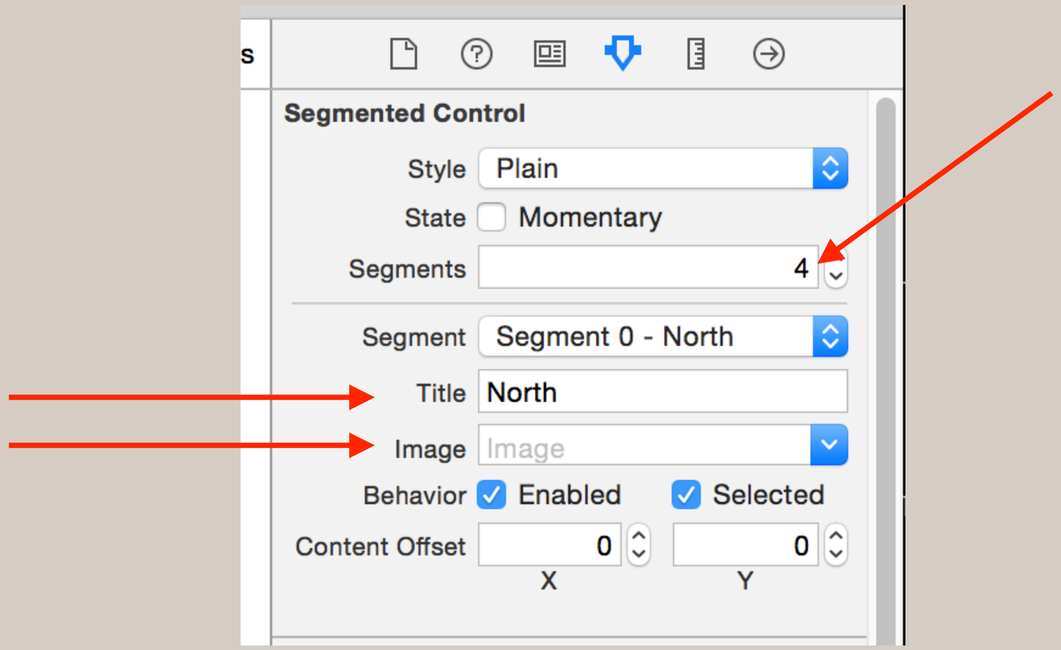
Segmented Control

Segmented Control

What is a segmented control?

- A horizontal control which consists of multiple segments where each segment functions as a discrete button
- Affords a compact means to group selection of one item
- A segmented control can display a title (an NSString object) or an image (UIImage object)

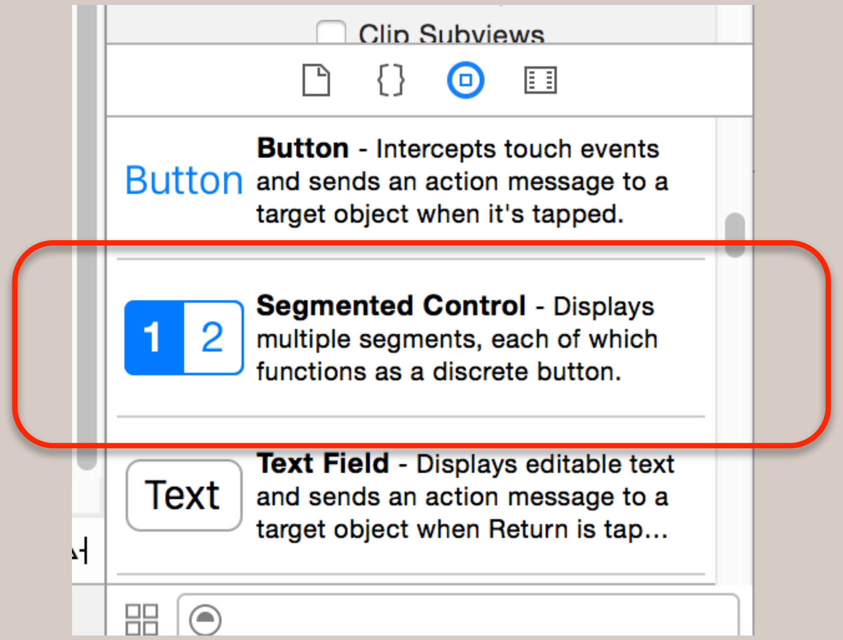
Attributes Inspector



Segmented Control

Style: Plain
State: ☐ Momentary
Segments: 4
Segment: Segment 0 - North
Title: North
Image: Image
Behavior: ✓ Enabled ✓ Selected
Content Offset: 0 (X) 0 (Y)

Object Library



Clip Subviews

Button - Intercepts touch events and sends an action message to a target object when it's tapped.

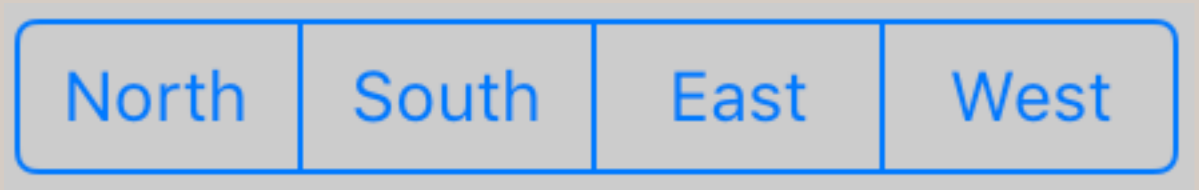
Segmented Control - Displays multiple segments, each of which functions as a discrete button.

Text Field - Displays editable text and sends an action message to a target object when Return is tap...

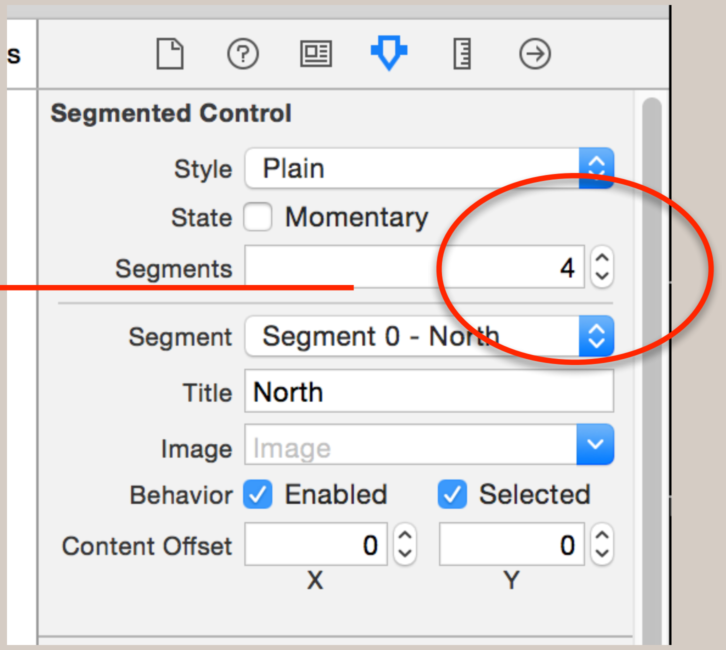
Segmented Control

You can have as many buttons in the control as you want

- It just divides the width of the control into the number of buttons you set
  - So, make sure it's wide enough for all the values



Attributes Inspector



Segmented Control

Style: Plain

State: ☐ Momentary

Segments: 4

Segment: Segment 0 - North

Title: North

Image: Image

Behavior: ✓ Enabled ✓ Selected

Content Offset: 0 (X) 0 (Y)

Segmented Control

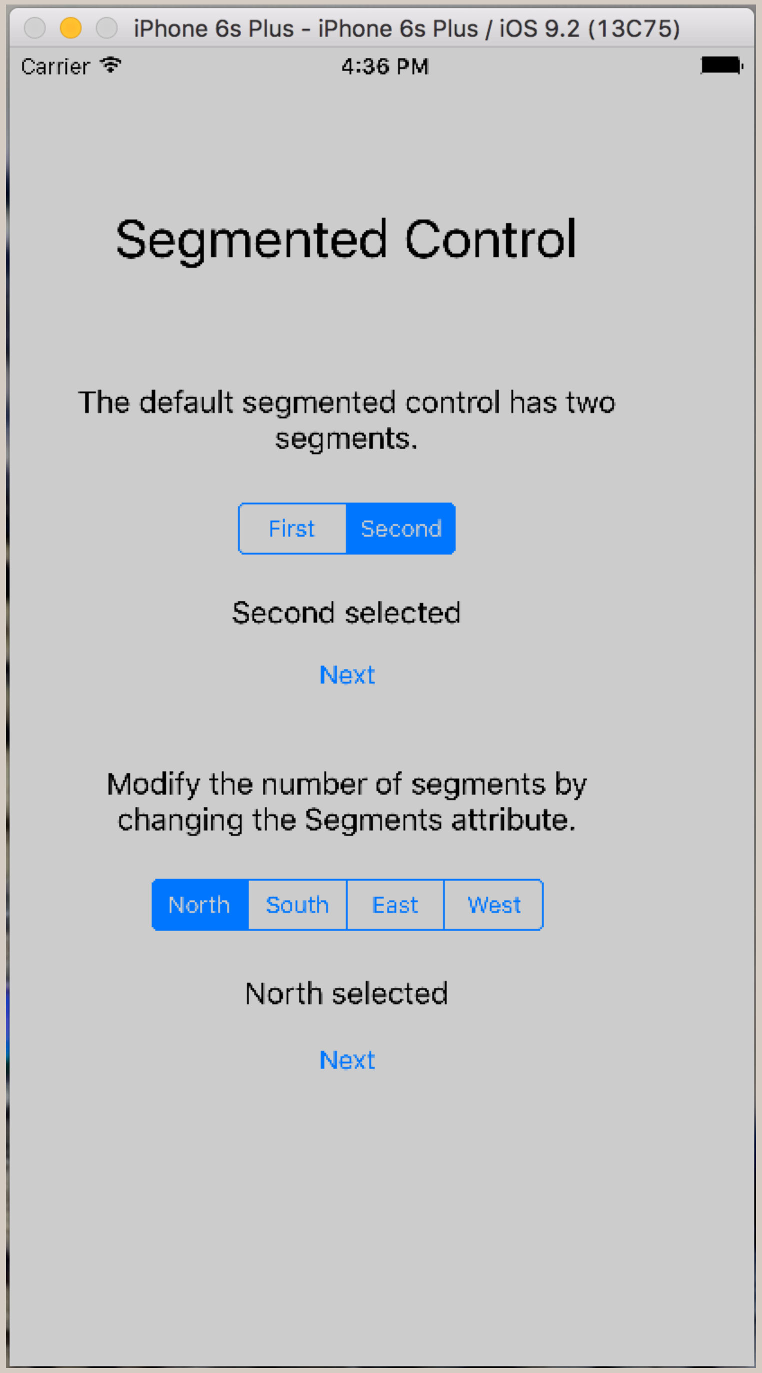
Demo:

- TestSegmentedControlSwift

Segmented control action handler. Called when button in segmented control is touched.

```
@IBAction func segControlTwoOptionsAction(sender: AnyObject) {

    switch self.segControlTwoOptions.selectedSegmentIndex
    {
        case 0:
            self.lblSegment1Message.text = "First selected"
        case 1:
            self.lblSegment1Message.text = "Second selected"
        default:
            break
    }
}
```


iPhone 6s Plus - iPhone 6s Plus / iOS 9.2 (13C75)

Carrier 4:36 PM

Segmented Control

The default segmented control has two segments.

Second selected

Modify the number of segments by changing the Segments attribute.

North selected

Swift Extensions

Swift Extensions

What are extensions?

- A Swift language capability.

- Extensions add *new functionality* to an *existing* class, structure, enumeration, or protocol type.
  - Includes the ability to extend types for which you <u>do not have access to the original source code</u>.

- Similar to categories in Objective-C but more capable.
  - Unlike Objective-C categories, Swift extensions do not have names.

Swift Extensions

Extensions in Swift can:

- Add instance and type (class-level) methods.
- Add *computed* instance and type properties.
- Provide new initializers.
- Define and use new nested types.
- Make an existing type conform to a protocol.

** However, you can <u>not</u> add regular properties.

Swift Extensions

Bottom line:

An excellent and easy way to add functionality/capability to an existing class/struct/enum/protocol, whether you have the source code or not.

Swift Extensions

Example: Extending the String class

- With this extension *every* string can now call this method!

Definition:

```
extension String
{
    func sayHi() -> String {
        return "Hi " + self + "  :)"
    }
}
```

Usage:

```
print(s.sayHi())
```

Swift Extensions

You can spread multiple extensions across N definitions, across N files.
- The build process coalesces the original class/struct/enum/protocol definition and all extensions into a single definition.

```
extension String
{
    func sayHi() -> String {
        return "Hi " + self + "  :)"
    }
}

extension String
{
    func sayBye() -> String {
        return "Bye " + self + "  :("
    }
}
```

Swift Extensions

See TestExtensions2 in Canvas.

In-Class Exercise

In-Class Exercise

Create an application with:

- Segmented Control
- An Extension

Custom Protocols and Delegates

Custom Protocols and Delegates

The custom protocol is the driving force when talking about protocols and delegates. Without a protocol, delegates are worthless. And without at least one delegate, protocols are superfluous, unnecessary.

In order for a class to make use of a protocol, it must set itself to be the *delegate* of that protocol, in addition to implementing all required elements of the protocol.

Custom Protocols and Delegates

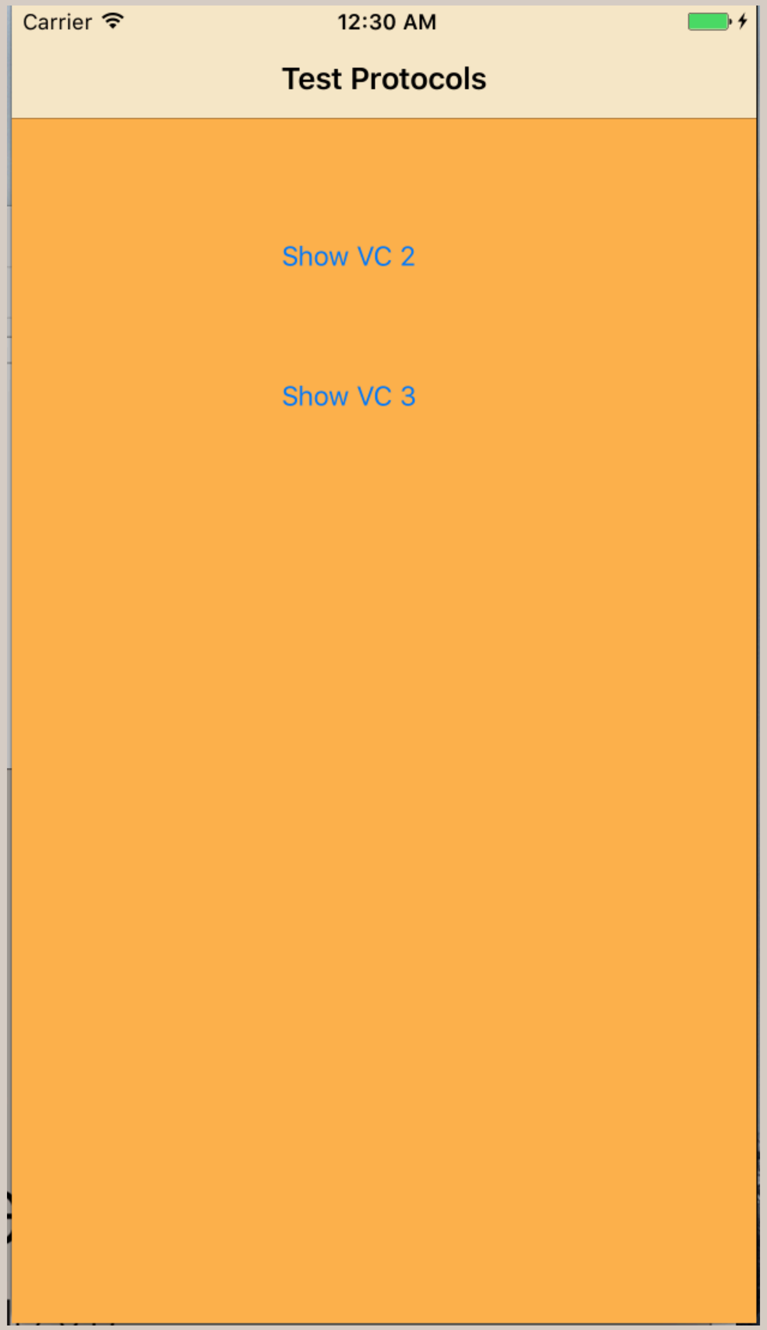
The purpose of defining a custom protocol is to define a *loosely-coupled* relationship between two objects.

Loosely-coupled means both objects know as little as possible about each other.

<u>Primary benefit:</u> The class that defines the protocol can interact with any other object, without knowing anything about the other object, other than it conforms to its protocol.

Custom Protocols and Delegates

TestProtocols



Mockups Paper

Mockups Paper

The next project deliverable.

The intent is to provide visual representations, with descriptions, of the intended app’s user interface.