Designing For The iPad

PROG31975 – Week 2 Part 2

Jawaad Sheikh

Jawaad.Sheikh@SheridanCollege.ca

Outline

- Introduction
- Designing iPad Apps
- Coding For Multiple Devices
- Auto Layout
- Text Entry Using UITextField
- Displaying a WebView With Feedback
- Exercises

Introduction

- When you build your app and layout your UI in your storyboard, it will layout nicely for 1 device.
- But the reality is, there are multiple iOS devices to support.
  - Multiple iPhone devices.
  - Multiple iPad devices.
  - Multiple Apple TV devices.
  - Multiple Apple Watch devices.

Introduction

- Where do you begin?
  - Start with the screen dimensions

Device	Retina	Portrait (px)	Landscape (px)
iPhone X	✓+	1125 x 2436	2436 x 1125
iPhone 6+, 6S+, 7+, 8+	✓+	1080 x 1920	1920 x 1080
iPhone 6, 6S, 7, 8	✓	750 x 1334	1334 x 750
iPhone 5, 6SE *5, 5S, 5C, 6SE*	✓	640 x 1136	1136 x 640
iPhone 4 *4, 4S*	✓	640 x 960	960 x 640
iPhone *1st, 2nd & 3rd Generation*		320 x 480	480 x 320
iPad Air / Retina iPad *1st & 2nd Generation / 3rd & 4th*	✓	1536 x 2048	2048 x 1536
iPad Pro	✓	2048 x 2732	2732 x 2048
iPad Mini *2nd, 3rd & 4th Generation*	✓	1536 x 2048	2048 x 1536
iPad *Mini, 1st & 2nd Generation*		768 x 1024	1024 x 768

Introduction

- Using these dimensions, you can develop graphics to ensure your look doesn't look old and pixelated

Device	Retina	Portrait (px)	Landscape (px)
iPhone X	✓+	1125 x 2436	2436 x 1125
iPhone 6+, 6S+, 7+, 8+	✓+	1080 x 1920	1920 x 1080
iPhone 6, 6S, 7, 8	✓	750 x 1334	1334 x 750
iPhone 5, 6SE *5, 5S, 5C, 6SE*	✓	640 x 1136	1136 x 640
iPhone 4 *4, 4S*	✓	640 x 960	960 x 640
iPhone *1st, 2nd & 3rd Generation*	✗	320 x 480	480 x 320
iPad Air / Retina iPad *1st & 2nd Generation / 3rd & 4th*	✓	1536 x 2048	2048 x 1536
iPad Pro	✓	2048 x 2732	2732 x 2048
iPad Mini *2nd, 3rd & 4th Generation*	✓	1536 x 2048	2048 x 1536
iPad *Mini, 1st & 2nd Generation*	✗	768 x 1024	1024 x 768

Designing For The iPad

- You will notice that the dimensions for the iPad are higher than the phone.
- This brings you more closer to a desktop feel.

iPad Air / Retina iPad *1st & 2nd Generation / 3rd & 4th*		1536 x 2048	2048 x 1536
iPad Pro		2048 x 2732	2732 x 2048
iPad Mini *2nd, 3rd & 4th Generation*		1536 x 2048	2048 x 1536
iPad *Mini, 1st & 2nd Generation*		768 x 1024	1024 x 768

Designing For The iPad

- With this larger screen comes additional widgets to play with.
  - CollectionView
  - SplitView

Designing For The iPad

- Though you have a ton of real estate, consider the following:
  - Don’t cram everything onto the screen.
  - Don’t use graphics with smaller resolutions.
  - Try and combine view’s using the UISplitView object.

Coding For Multiple Devices

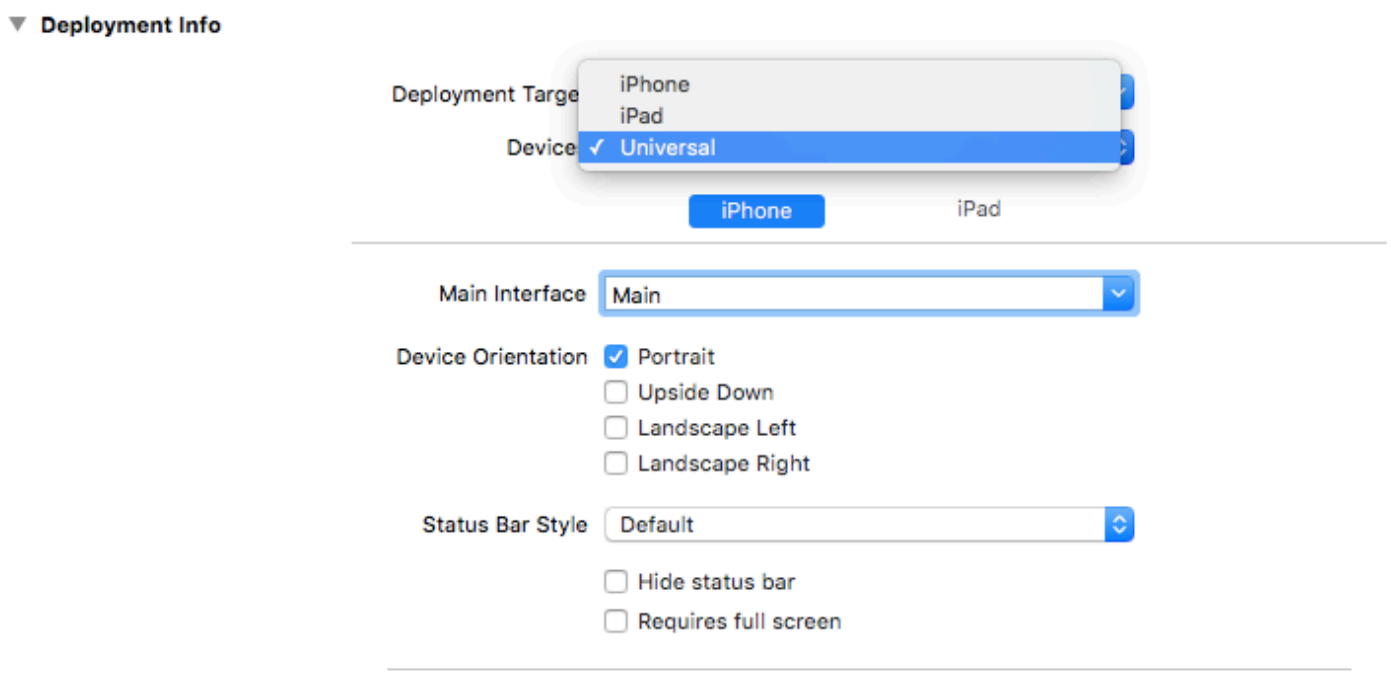
- Because there are multiple devices to build for your design approach changes.
  - Do you build for the iPhone only?
  - Do you build for the iPad only?
  - Do you build separately for iPhone and iPad?
  - Do you build universally for both iPhone and iPad?

Coding For Multiple Devices

- Coding for iPhone only – you lose out on iPad users.
- Coding for iPad only – you lose out on iPhone users.
- Coding separately for iPhone and iPad – you end up with smaller app size but end up maintaining 2 code bases.
- Coding universally – you end up with a larger app size but only 1 code base to maintain.

Coding For Multiple Devices

- You can choose which way to go in your project settings page:



Coding For Multiple Devices

- When building for a universal app there may be potions of your code that may only execute on the iPad or the iPhone.
- iOS has special code to allow you to selectively execute on the device you want.

Coding For Multiple Devices

- The following will check to see which device your app is executing on:
  - UIDevice.current.userInterfaceIdiom
- The following are possibilities:
  - .pad
  - .phone
  - .tv
  - .carplay

Coding For Multiple Devices

- Usage example:

```
if UIDevice.current.userInterfaceIdiom == .pad
{
        // execute iPad specific code
}
```

AutoLayout

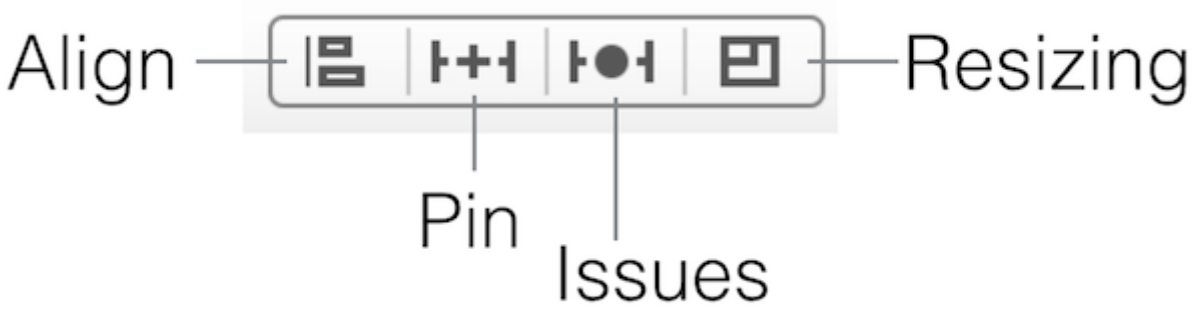
- Now that we know how to control our code, how do we adjust our storyboard to adapt to different screen sizes.
- This is where AutoLayout comes in.
  - Its an adaptive constraint based system that allows user to place constraints on how to display individual items on the screen.

Auto Layout

- It is not a trivial process to do and can add a lot of time to the development process.
- Whats the alternative? Create a new storyboard for each new iOS device?

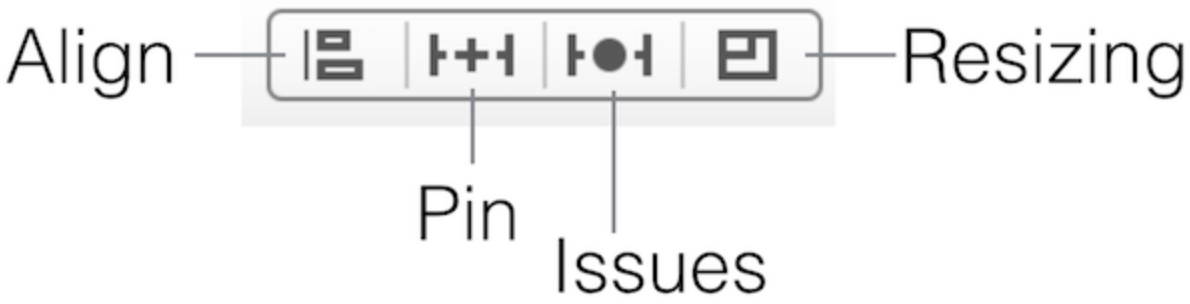
Auto Layout

- The Auto Layout system is below:



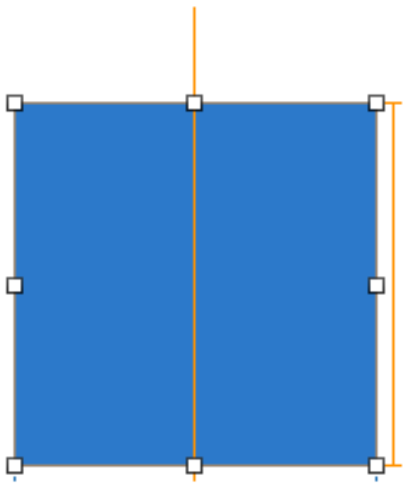
Auto Layout

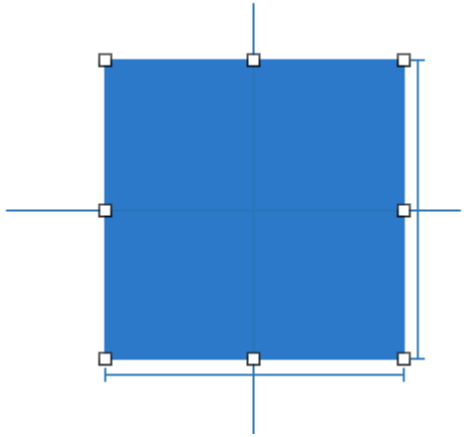
- Align – constraints for aligning 2 views.
- Pin – space constraints.
- Issues – layout issues.
- Resizing – resizing constraints.



Auto Layout

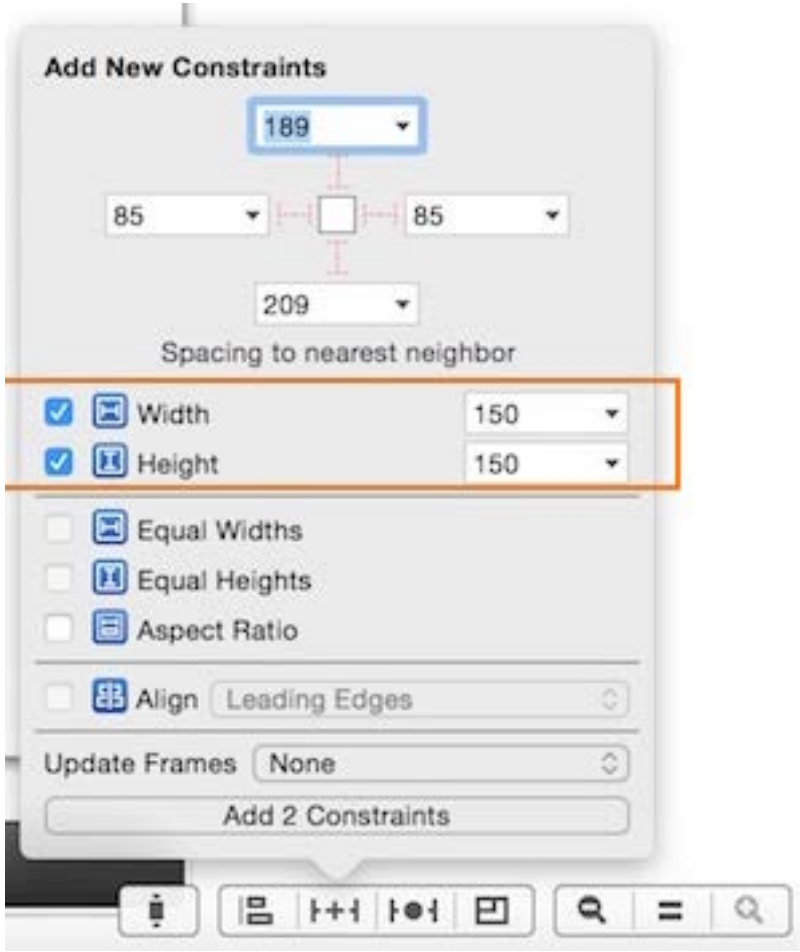
- Yellow constraint lines indicate insufficient constraints defined.
- Blue indicates proper constraints defined.





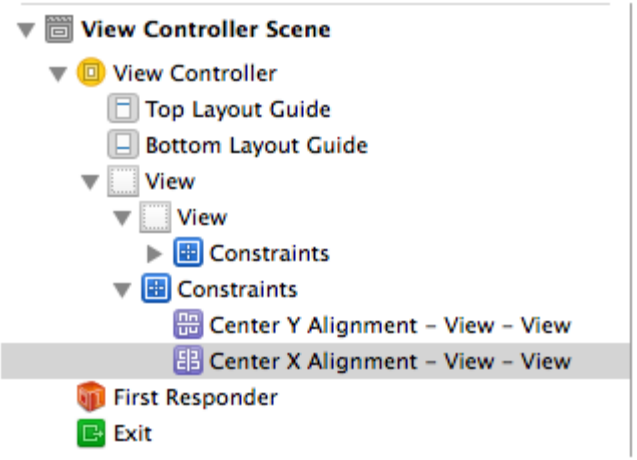
Auto Layout

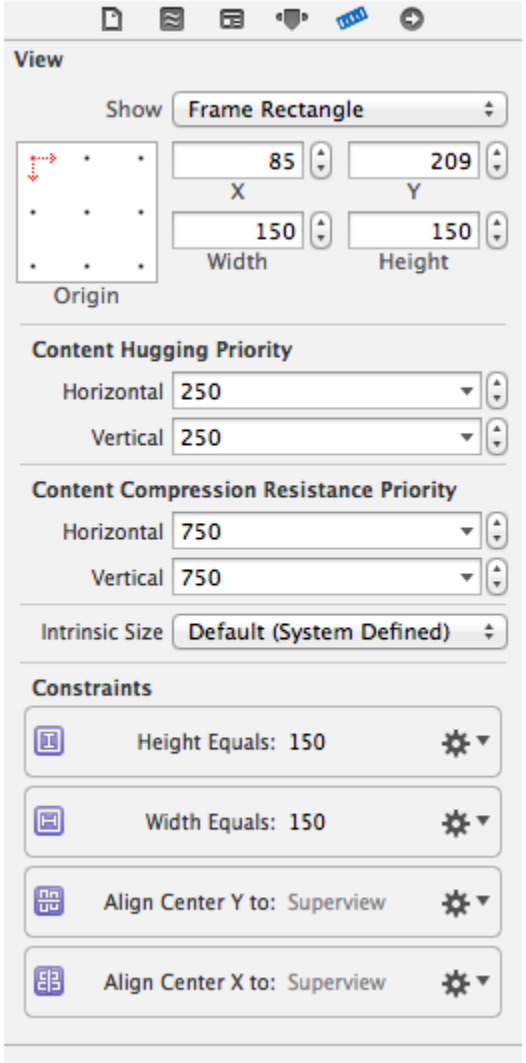
- This view will add new constraints to the highlighted object.



Auto Layout

- You can edit constraints in these views.





Text Entry Using UITextField

- Up until now you’ve displayed data using UILabel.
- But there will come a time where you will need user input.
- There are 2 objects to do this:
  - UITextField
  - UITextView

Text Entry Using UITextField

- UITextField is for single line entry.
- UITextView is for multi-line entry.
- Careful not to mix the two – you won’t be able to connect your IBOutlet to the wrong one.

Text Entry Using UITextField

- When adding text entry, an issue appears,
  - How to make the keyboard disappear?
  - It doesn’t disappear automatically.
- We need to bring in a Protocol (aka Interface in the Java world) called UITextFieldDelegate.

Text Entry Using UITextField

- Add a UITextFieldDelegate protocol as follows:

```
class myView : UIViewController, UITextFieldDelegate{
```

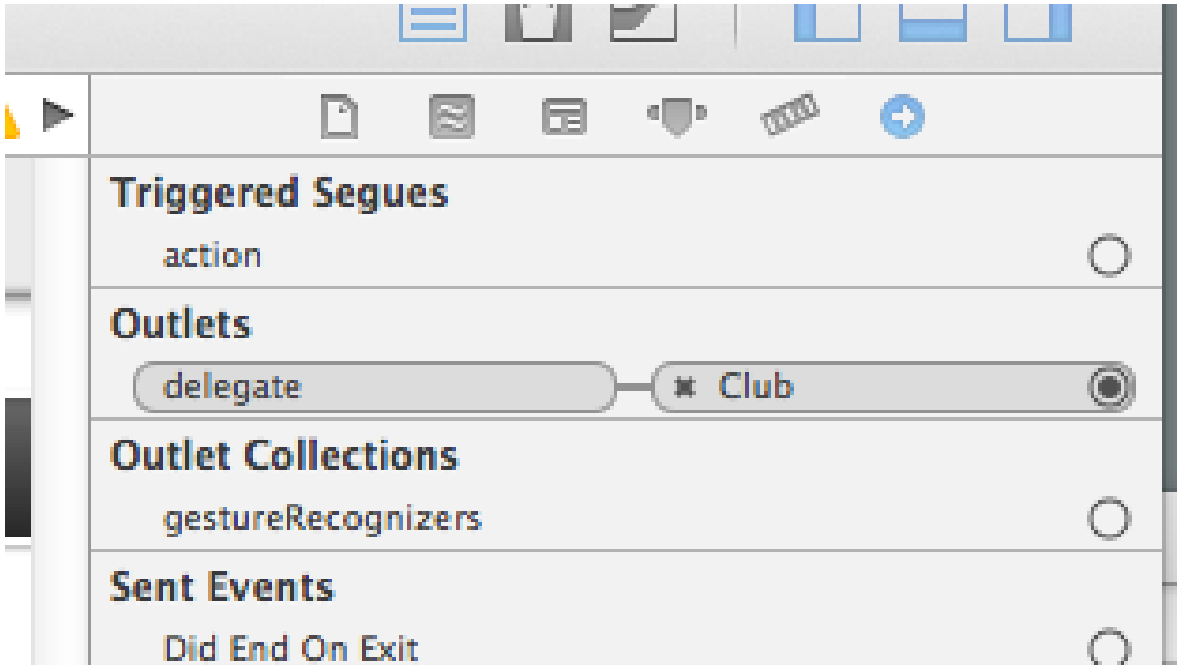
Text Entry Using UITextField

- Once added, you can add the following method to your View Controller:

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

Text Entry Using UITextField

- Once the method is added, you need to connect "delegate" to your "yellow circle" in your storyboard



Text Entry Using UITextField

- Other considerations
  - What happens when the keyboard appears above your textbox?
  - What happens when your user wants to click on the screen to get rid of your keyboard?

Displaying A UIWebView With Feedback

- Similar to the UITextFieldDelegate, there is a UIWebViewDelegate (being replaced with WKWebViewDelegate)
- It works by using a UIActivityIndicatorView as a loading indicator for your web page.

Displaying A UIWebView With Feedback

- Recall the code to get a webView to load a webpage placed in viewDidLoad()

```
// assume @IBOutlet var myWebView:UIWebView!
override func viewDidLoad()
{
        URL urlAddress = URL(string: “https://www.projectmkd.com”)
        URLRequest url = URLRequest(url: urlAddress)
        myWebView.loadRequest(url as URLRequest)
}
```

Displaying A UIWebView With Feedback

- To get an activity indicator view to hide/unhide during the page load, you will need the UIWebViewDelegate protocol

```
class myView : UIViewController, UIWebViewDelegate {
```

Displaying A UIWebView With Feedback

- Then you can add the following UIWebView support methods:
  - webViewDidStartLoad()
  - webViewDidFinishLoad()

Displaying A UIWebView With Feedback

- Usage example:

```
// assume @IBOutlet var activity:UIActivityIndicatorView!
func webViewDidStartLoad( webView : UIWebView)
{
        activity.isHidden = false
        activity.startAnimating()
}
```

Displaying A UIWebView With Feedback

- Usage example:

```
func webViewDidFinishLoad( webView : UIWebView)
{
        activity.isHidden = true
        activity.stopAnimating()
}
```

Displaying A UIWebView With Feedback

- Don’t forget to connect the delegate to View Controller in your storyboard.

Exercise

- Develop an iPad app that will have 2 text fields, 2 labels, a submit button and a webView such that:
  - The 2 textfields will display the user entry in labels below it upon selecting submit.
  - An alert box asking are you sure is displayed upon selecting submit to confirm updating the labels.
  - The webview displays a website with activity indicator.
  - Hide the webView for iPhones.
  - Adjust your view for iPhone and iPad.

Exercise 2

- Develop an iPad app will have 3 pages.
  - The home page will be a login screen with a link to register.
  - The registration page will ask for user information (name, email, address, etc)
  - The third page will be loaded upon successful login which will have a webview pointing to a website and have an activity indicator.
  - Use alert boxes for unsuccessful logins with 1 button being cancel and the other being register.