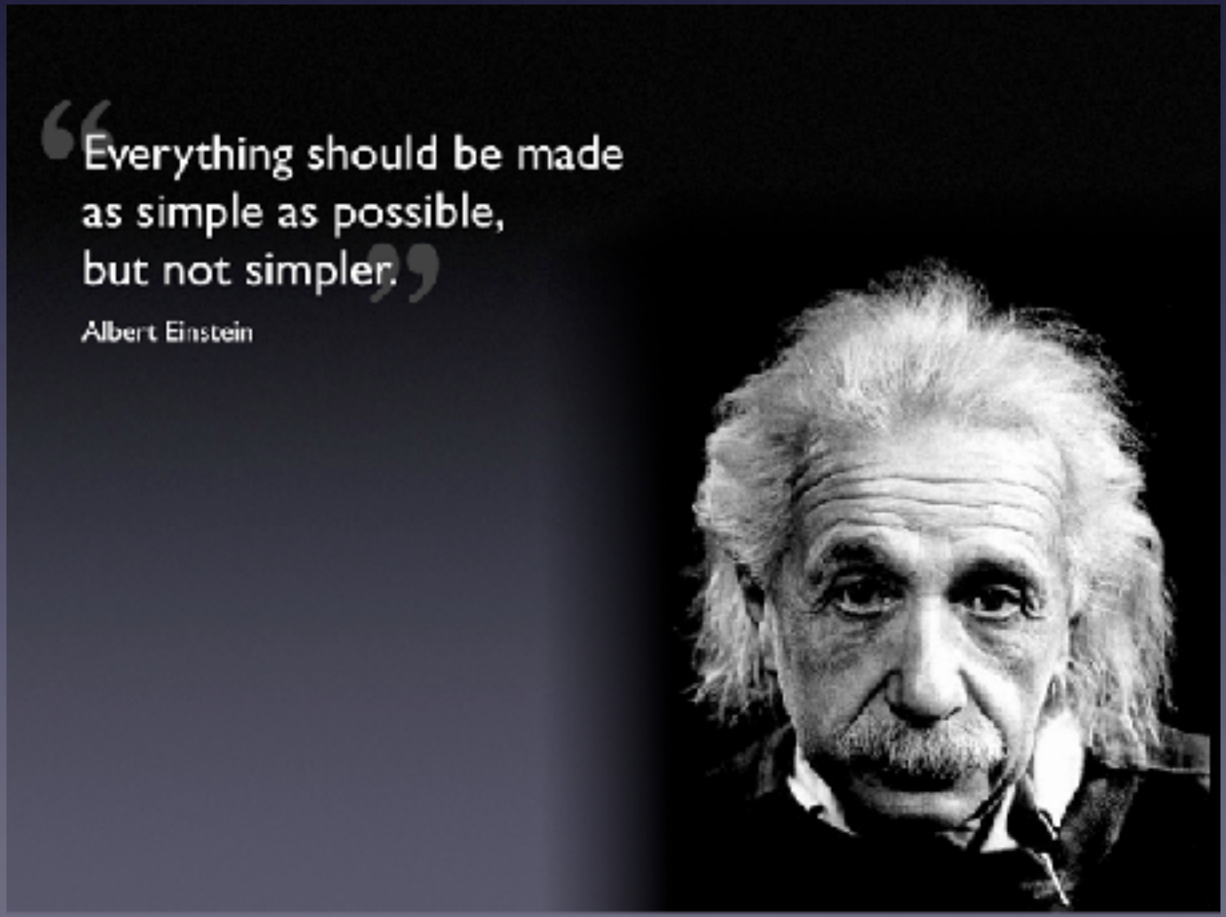
# Project Mongoose

Swift 3.0 & Xcode 8.2.1



# Topics Covered

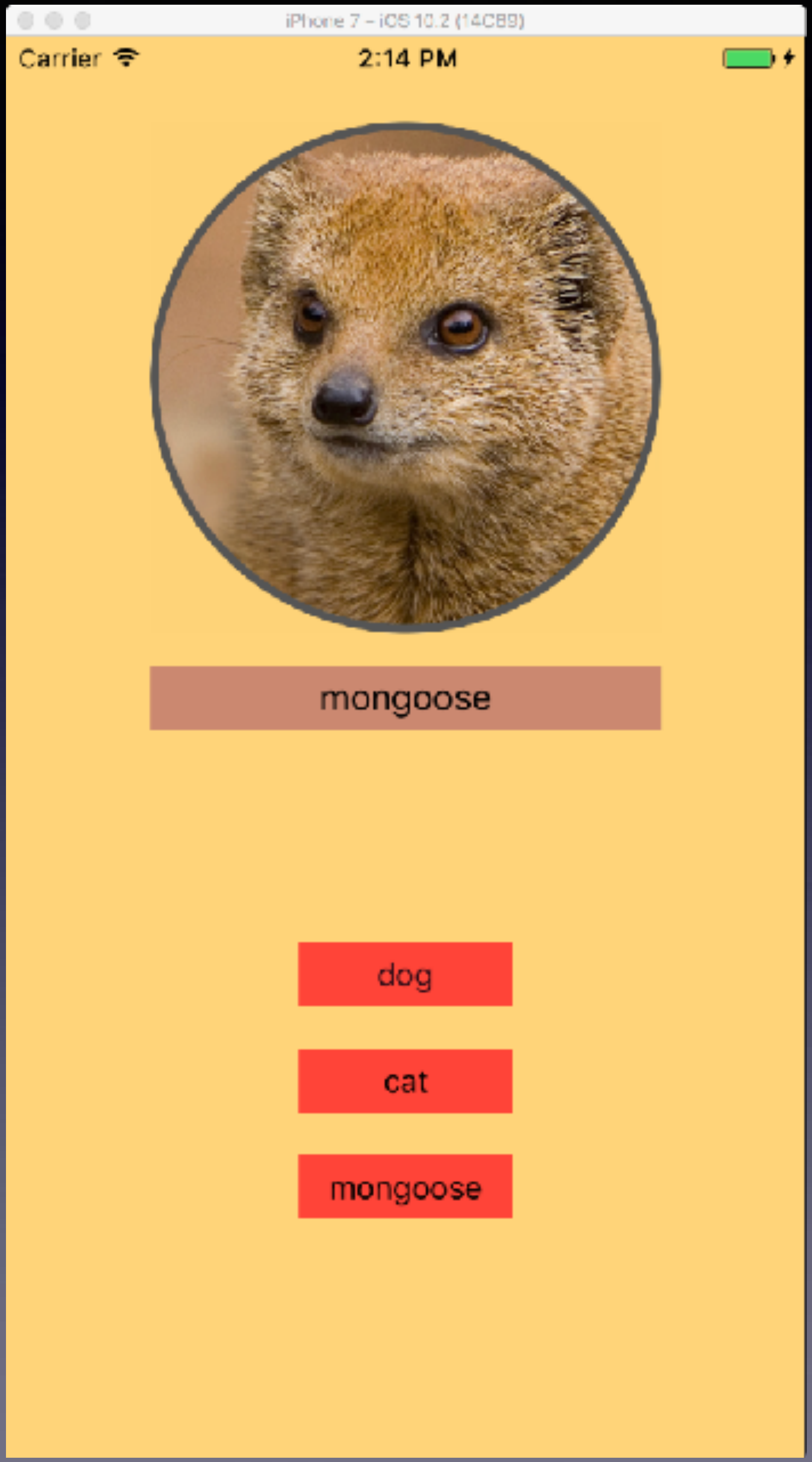
- Build a UIView which includes:
  UIImage, Label, Buttons, Stack Views
- Manage images and their display
- Use Stack Views and Constraints.
- Size Classes: portrait <—> landscape
- Connecting buttons, labels, and images
  Using Outlets and Actions to access UI from Controller

## Simple Project

This is a very simple project but it covers a number of important points.

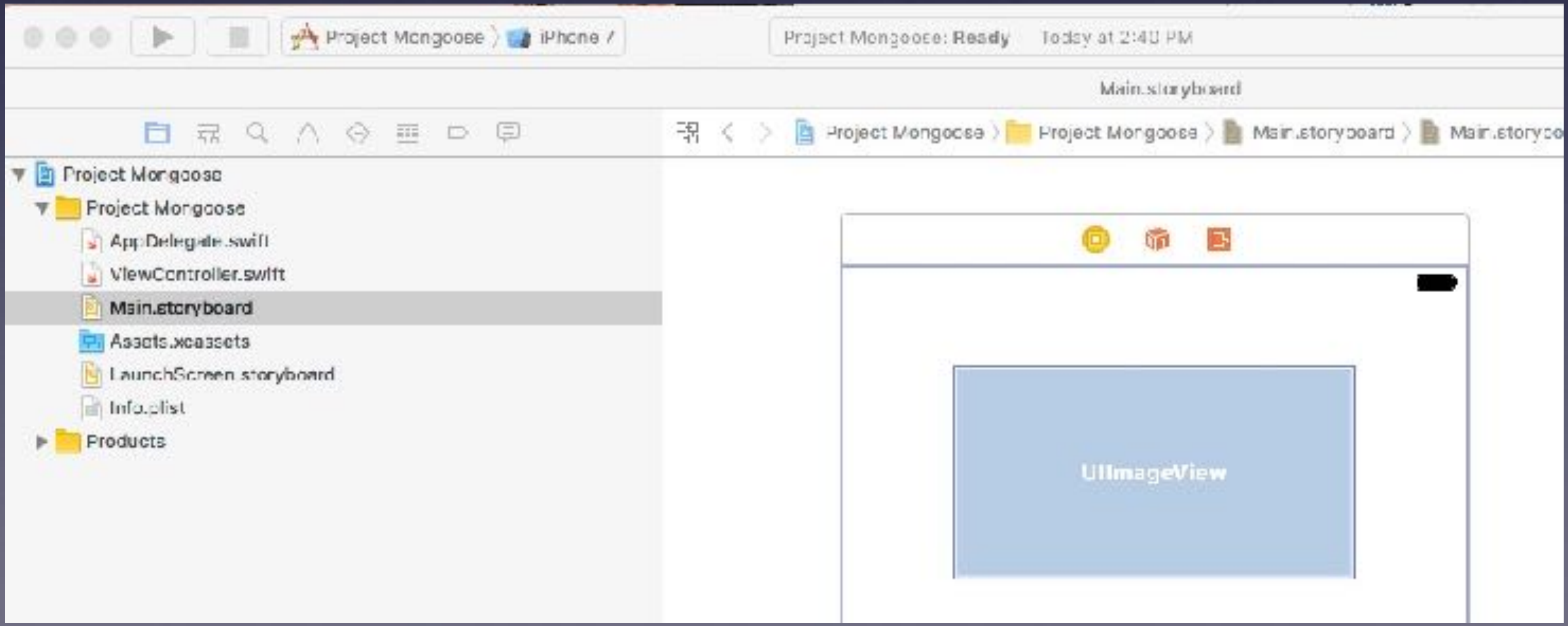
Pressing dog, cat, or mongoose updates the image and the text.

But we will cover many important features of Xcode and Swift along the way.



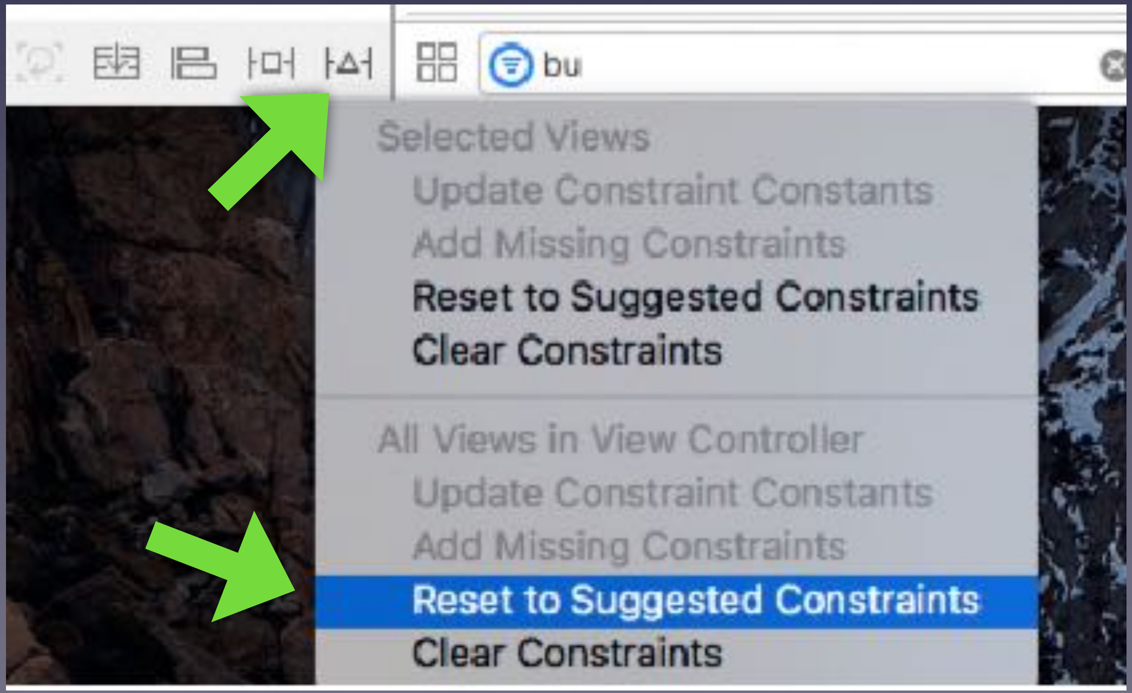
# Starting New Project

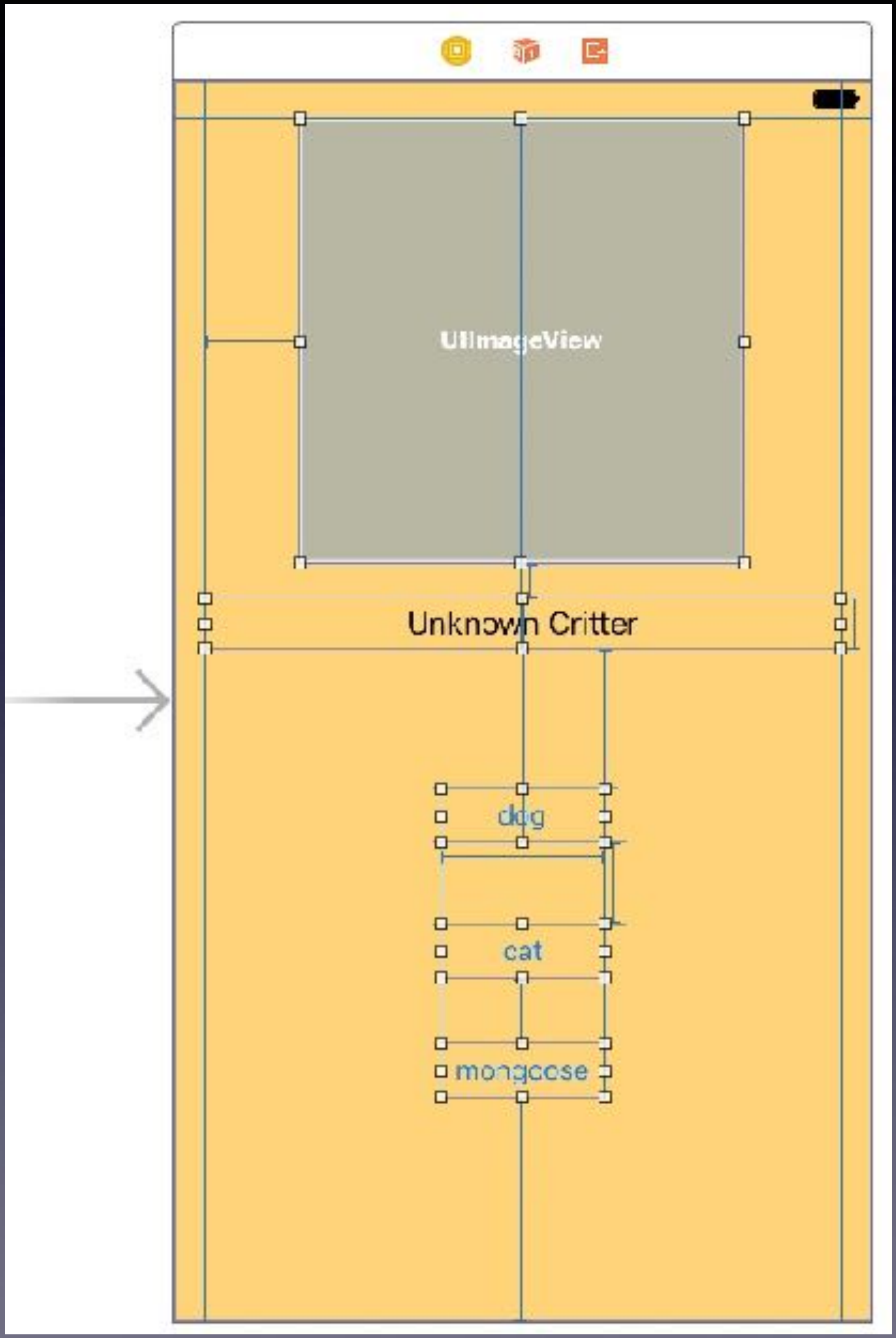
- Single View Application
- Product Name: **Project Mongoose**
- Add Image View, Label, and 3 Buttons



# Initial UI

- Add text to Label and Buttons.
- Arrange Image, Label and Buttons in the proximate layout
- Using pin control reset to suggested constraints.


bu
Selected Views
Update Constraint Constants
Add Missing Constraints
Reset to Suggested Constraints
Clear Constraints
All Views in View Controller
Update Constraint Constants
Add Missing Constraints
Reset to Suggested Constraints
Clear Constraints

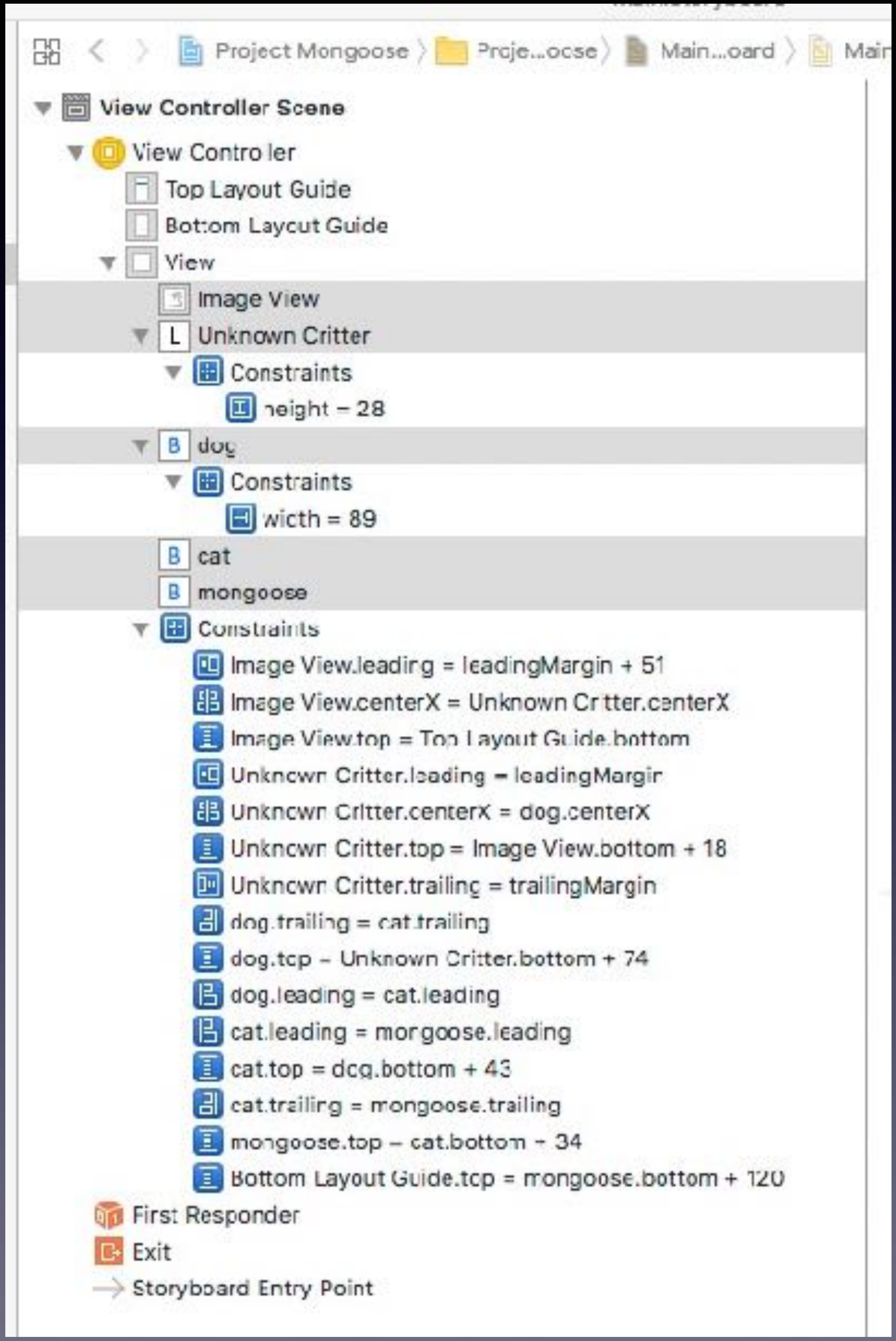


# Constraints

Select to display ->

Select constraint to edit.

This may be enough for some UI screens, but will not work for our project as we will see later.



```
View Controller Scene
  View Controller
    Top Layout Guide
    Bottom Layout Guide
    View
      Image View
      Unknown Critter
        Constraints
          height = 28
      dog
        Constraints
          width = 89
      cat
      mongoose
      Constraints
        Image View.leading = leadingMargin + 51
        Image View.centerX = Unknown Critter.centerX
        Image View.top = Top Layout Guide.bottom
        Unknown Critter.leading = leadingMargin
        Unknown Critter.centerX = dog.centerX
        Unknown Critter.top = Image View.bottom + 18
        Unknown Critter.trailing = trailingMargin
        dog.trailing = cat.trailing
        dog.top = Unknown Critter.bottom + 74
        dog.leading = cat.leading
        cat.leading = mongoose.leading
        cat.top = dog.bottom + 43
        cat.trailing = mongoose.trailing
        mongoose.top = cat.bottom + 34
        Bottom Layout Guide.top = mongoose.bottom + 120
  First Responder
  Exit
  Storyboard Entry Point
```

# Add Images

- Find images for the three critters and put them in a folder called ‘animals’.
- Include one image which will display when there is no image to display.

- Select Assets.xcassetts and drop the image folder ‘animals’ in the section with AppIcon.
- All your images will now be available to your project.

# Attach image to UIImageView

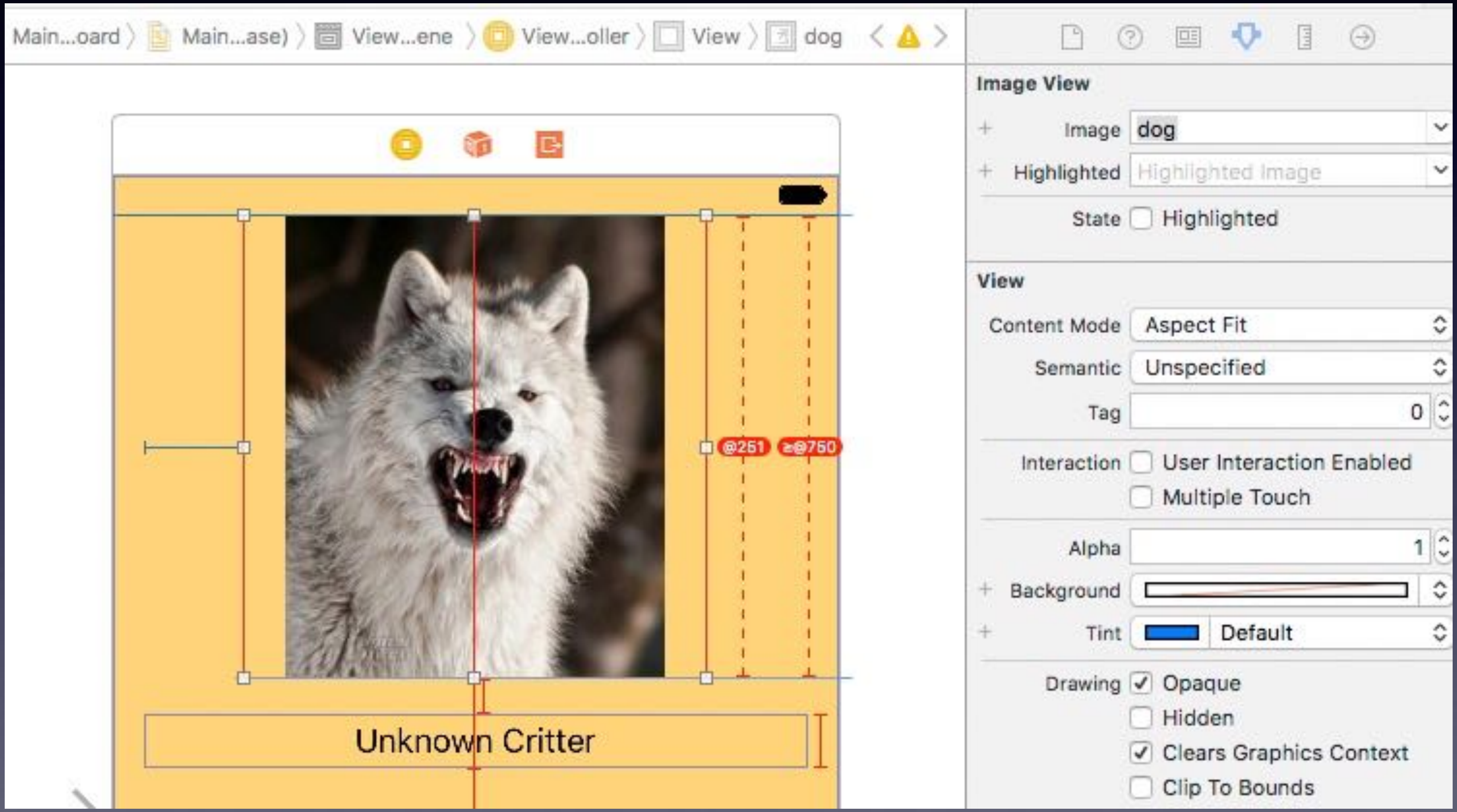


Image View

| | |
|---|---|
| Image | dog |
| Highlighted | Highlighted Image |
| State | ☐ Highlighted |

View

| | |
|---|---|
| Content Mode | Aspect Fit |
| Semantic | Unspecified |
| Tag | 0 |
| Interaction | ☐ User Interaction Enabled |
| | ☐ Multiple Touch |
| Alpha | 1 |
| Background | |
| Tint | Default |
| Drawing | ✓ Opaque |
| | ☐ Hidden |
| | ✓ Clears Graphics Context |
| | ☐ Clip To Bounds |

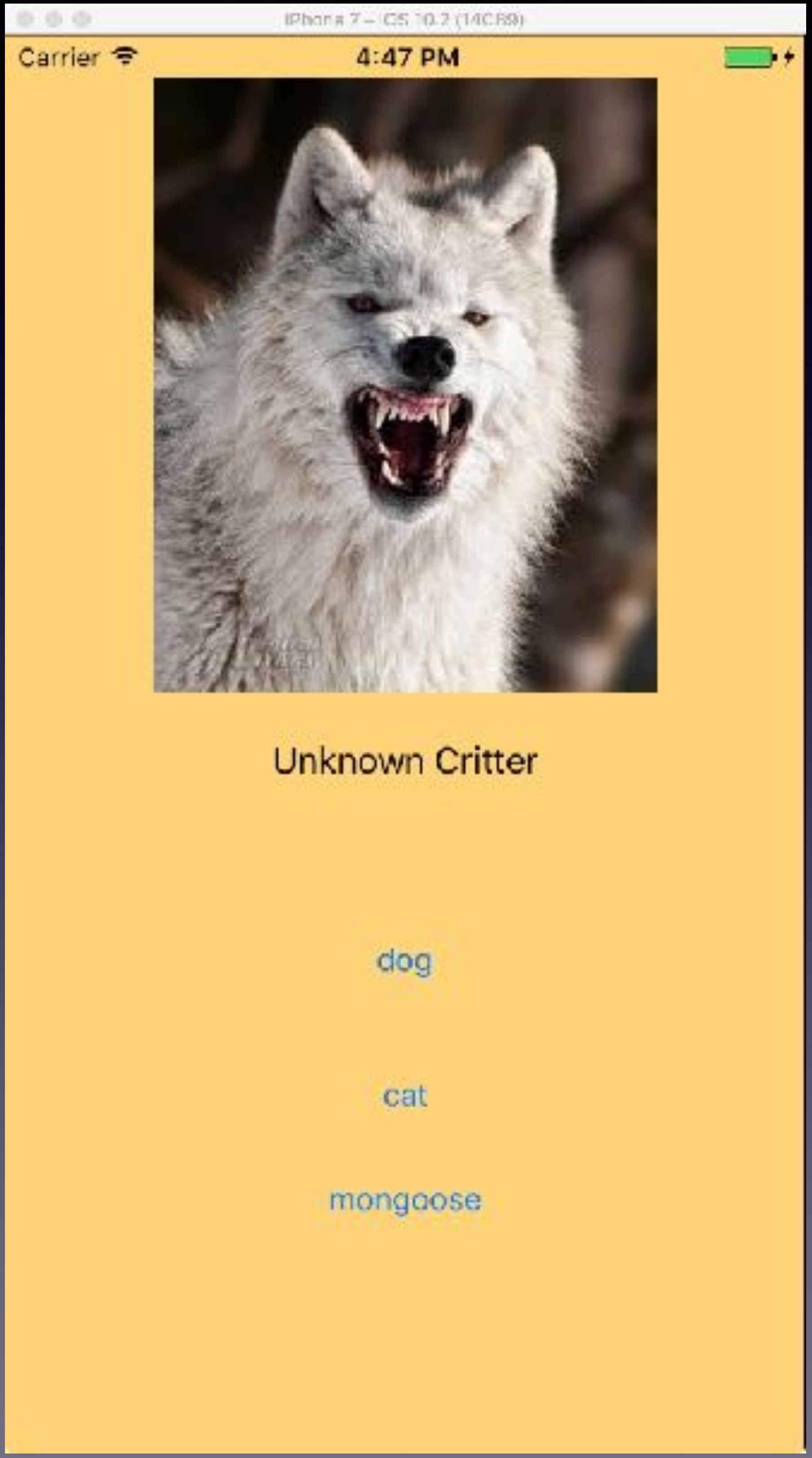
**Image**: select the image to display
**Content Mode:** select how image is fit into UIIMageView

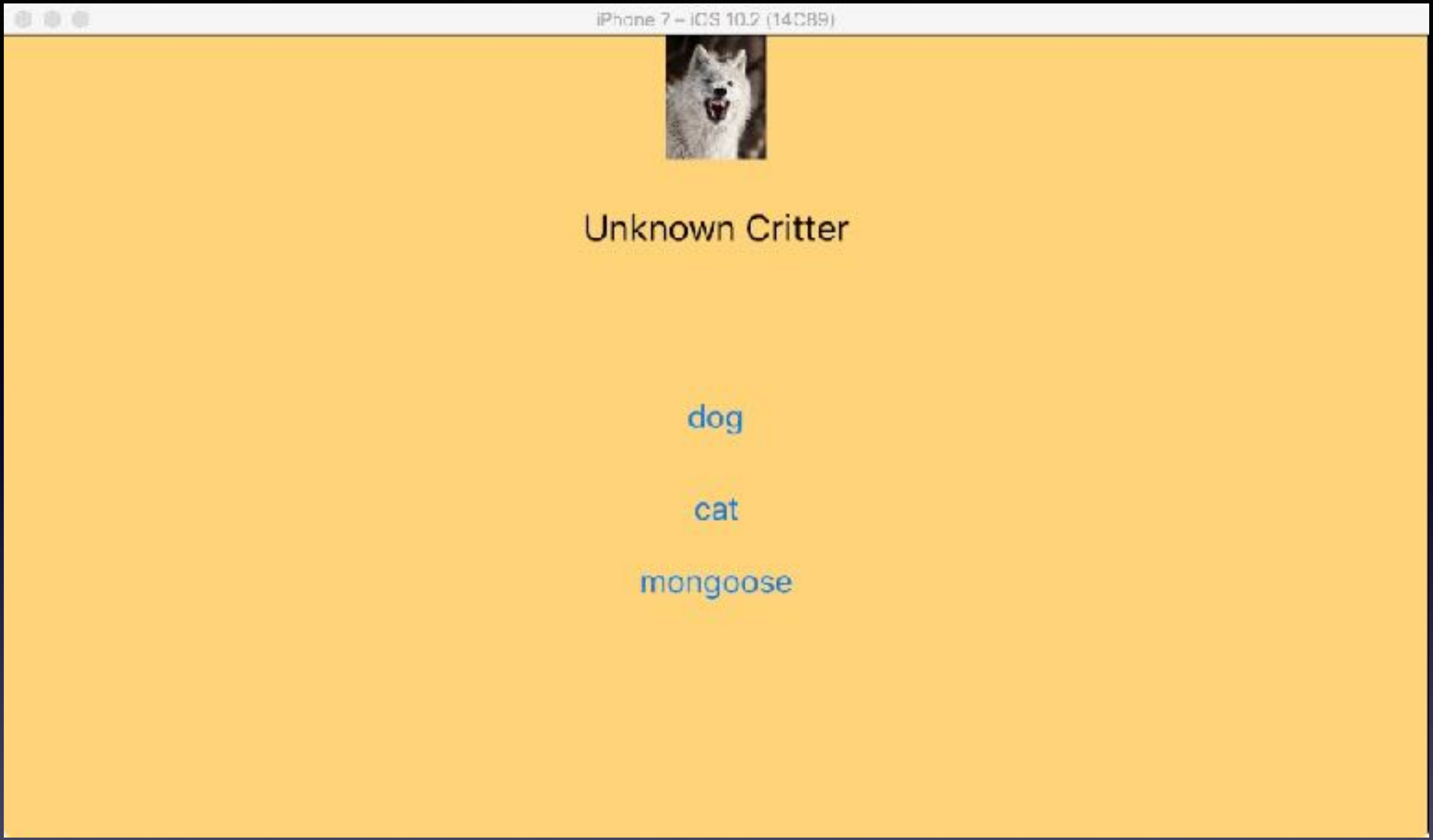
## Portrait

Not bad.

But remember we set the constraints in this mode.

How will it look in Landscape?



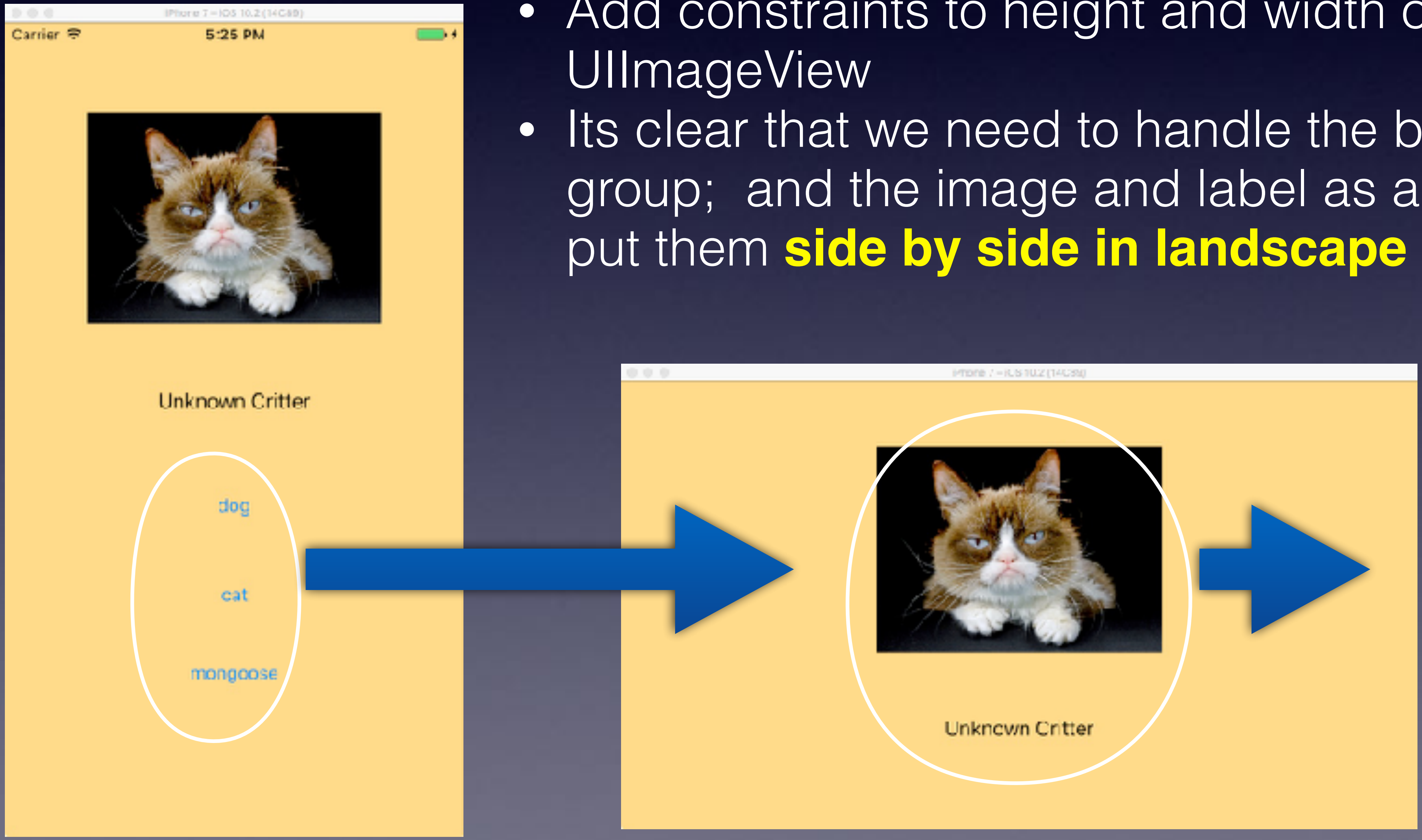


# Landscape

Not good.

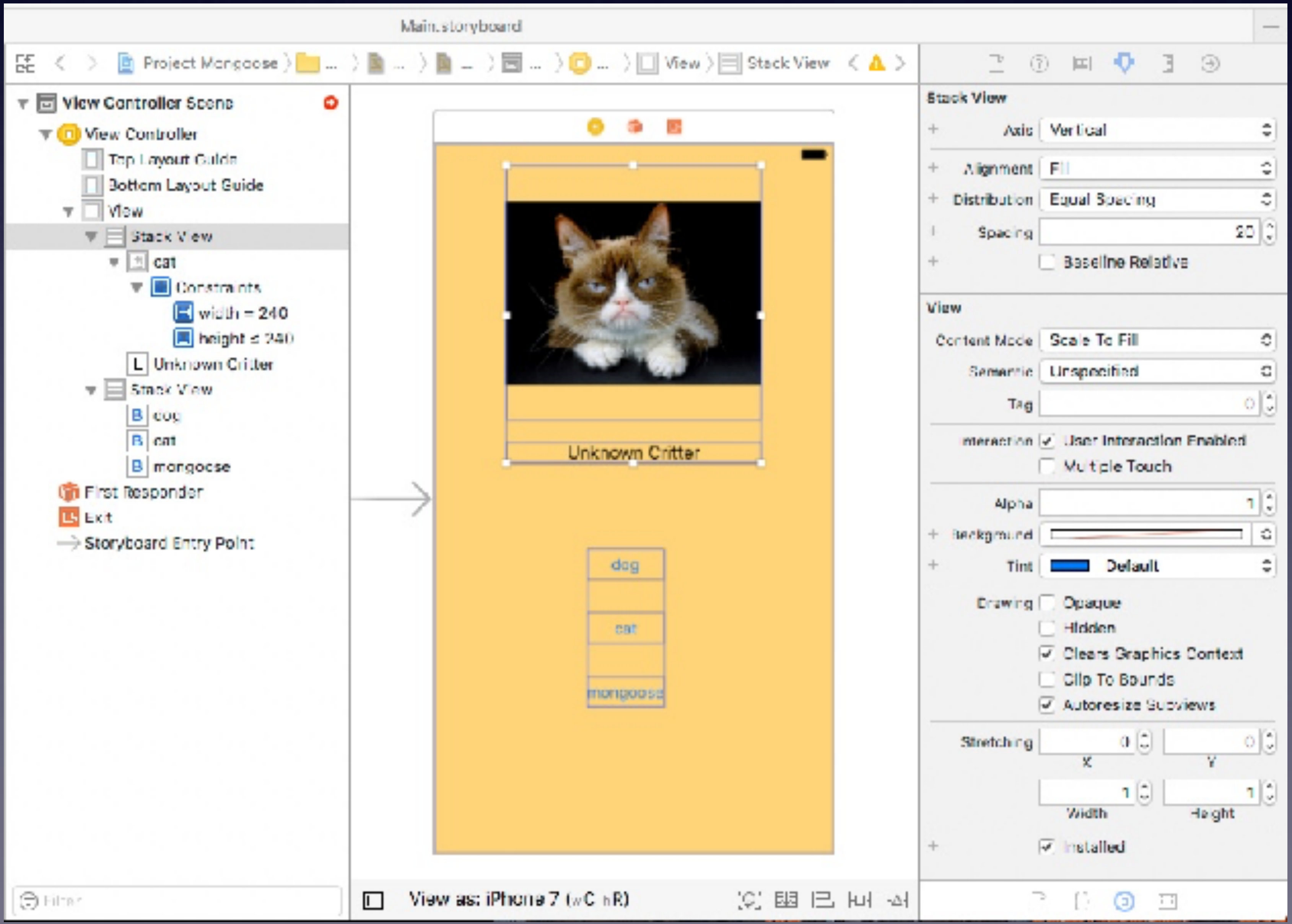
# Manipulate Constraints

- Clear constraints
- Select UIImageView and Label; align leading and trailing edges.
- Add constraints to height and width of UIImageView
- Its clear that we need to handle the buttons as a group; and the image and label as a group and put them **side by side in landscape**



# Stacking Views

- Select the three buttons and then put them into a stack.
- Select the image and the label and put it into another stack view.



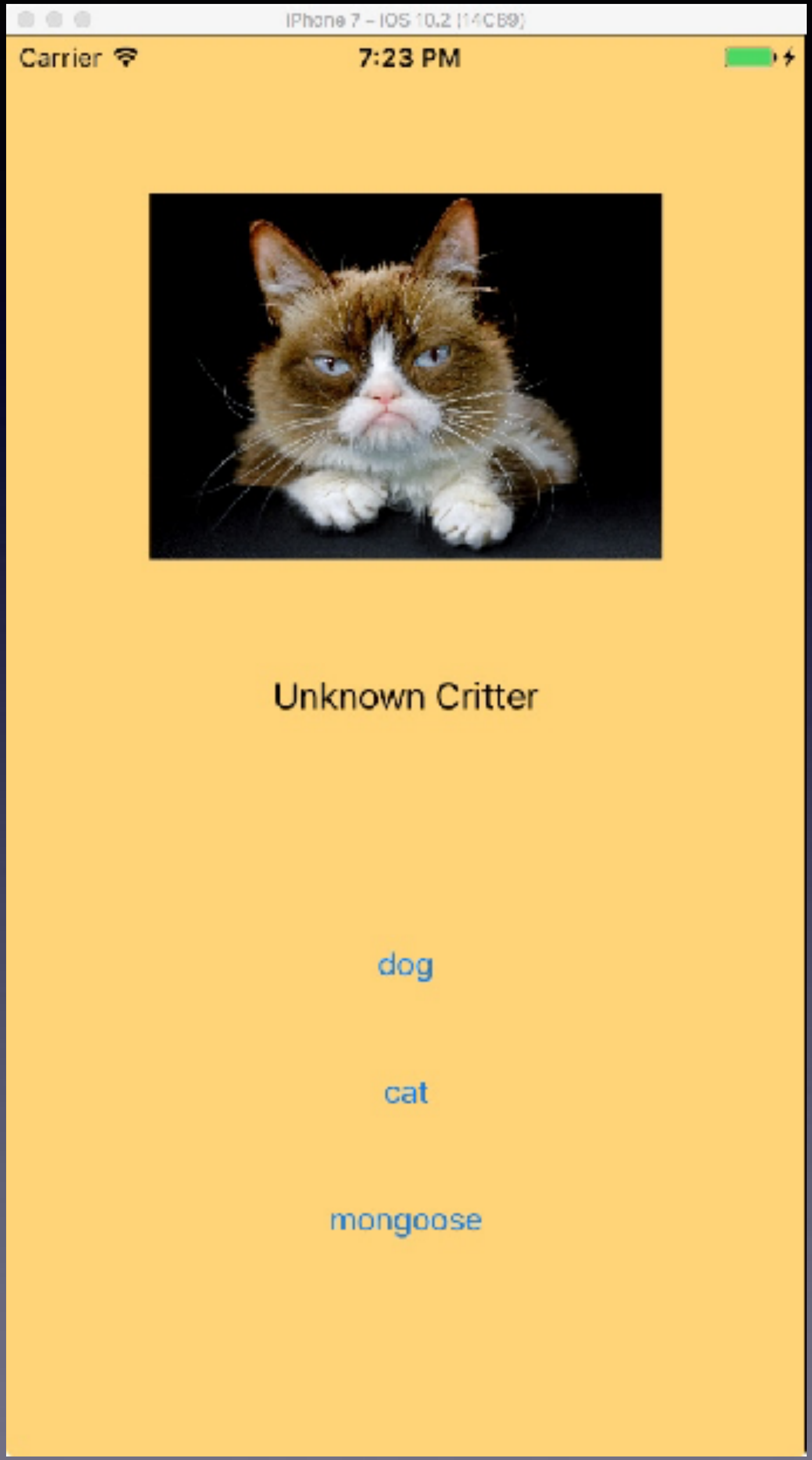
View Controller Scene
- View Controller
  - Top Layout Guide
  - Bottom Layout Guide
  - View
    - Stack View
      - cat
        - Constraints
          - width = 240
          - height ≤ 240
      - Unknown Critter
    - Stack View
      - dog
      - cat
      - mongoose
- First Responder
- Exit
- Storyboard Entry Point

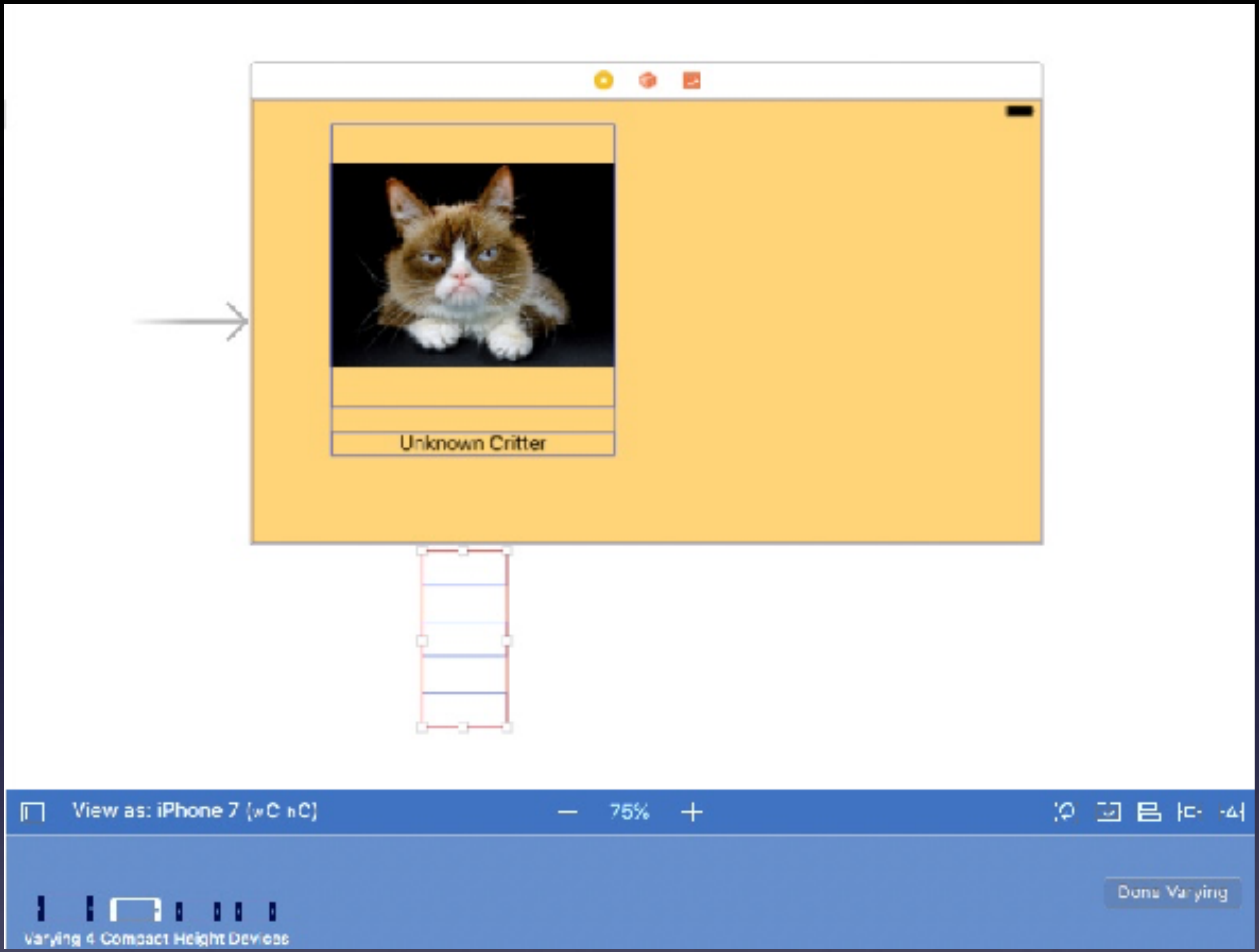
Stack View
- Axis: Vertical
- Alignment: Fill
- Distribution: Equal Spacing
- Spacing: 20
- ☐ Baseline Relative

View
- Content Mode: Scale To Fill
- Semantic: Unspecified
- Tag: 0
- Interaction: ✓ User Interaction Enabled; ☐ Multiple Touch
- Alpha: 1
- Background
- Tint: Default
- Drawing: ☐ Opaque; ☐ Hidden; ✓ Clears Graphics Context; ☐ Clip To Bounds; ✓ Autoresize Subviews
- Stretching: X 0, Y 0, Width 1, Height 1
- ✓ Installed

# Portrait:
# size class

- Set size view as: wC hR
- both stack views centerX
- constrain stack to top (20)
- constrain button stack to bottom of image stack (80)



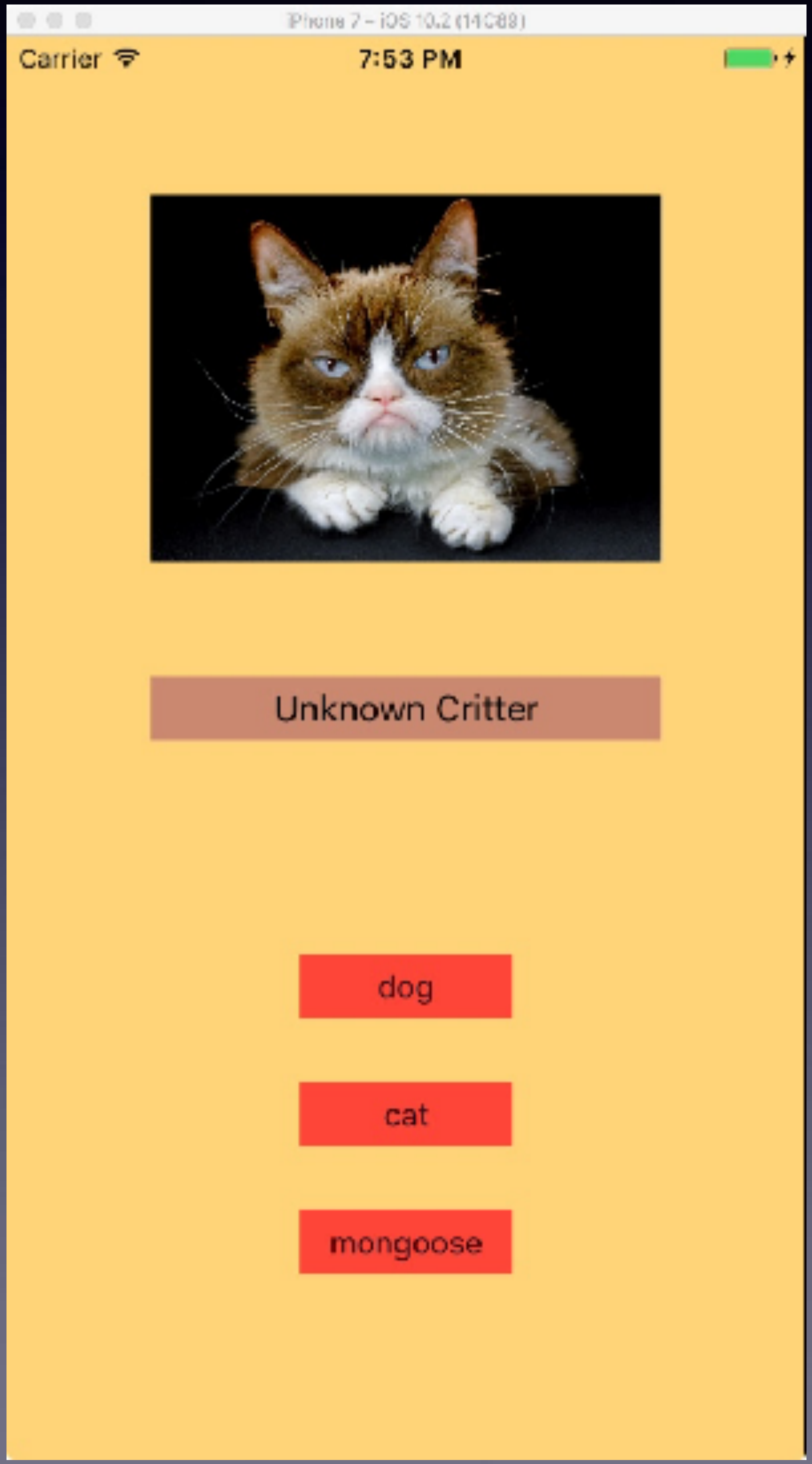


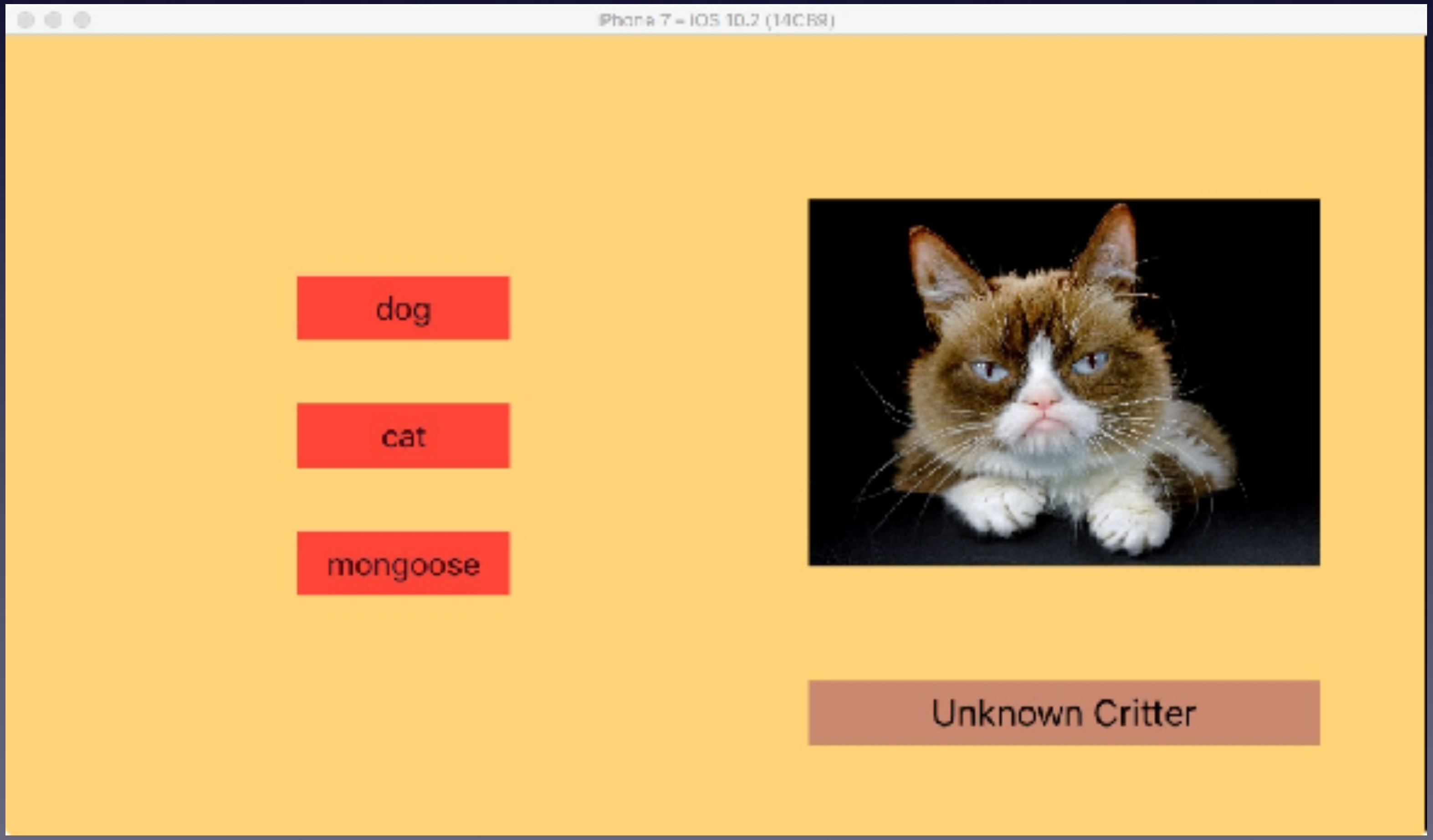
- Set size view as: wC hC
- both stack views centerY
- constrain stack to right (20)
- constrain button stack to right of image stack (150)

# Landscape: size class

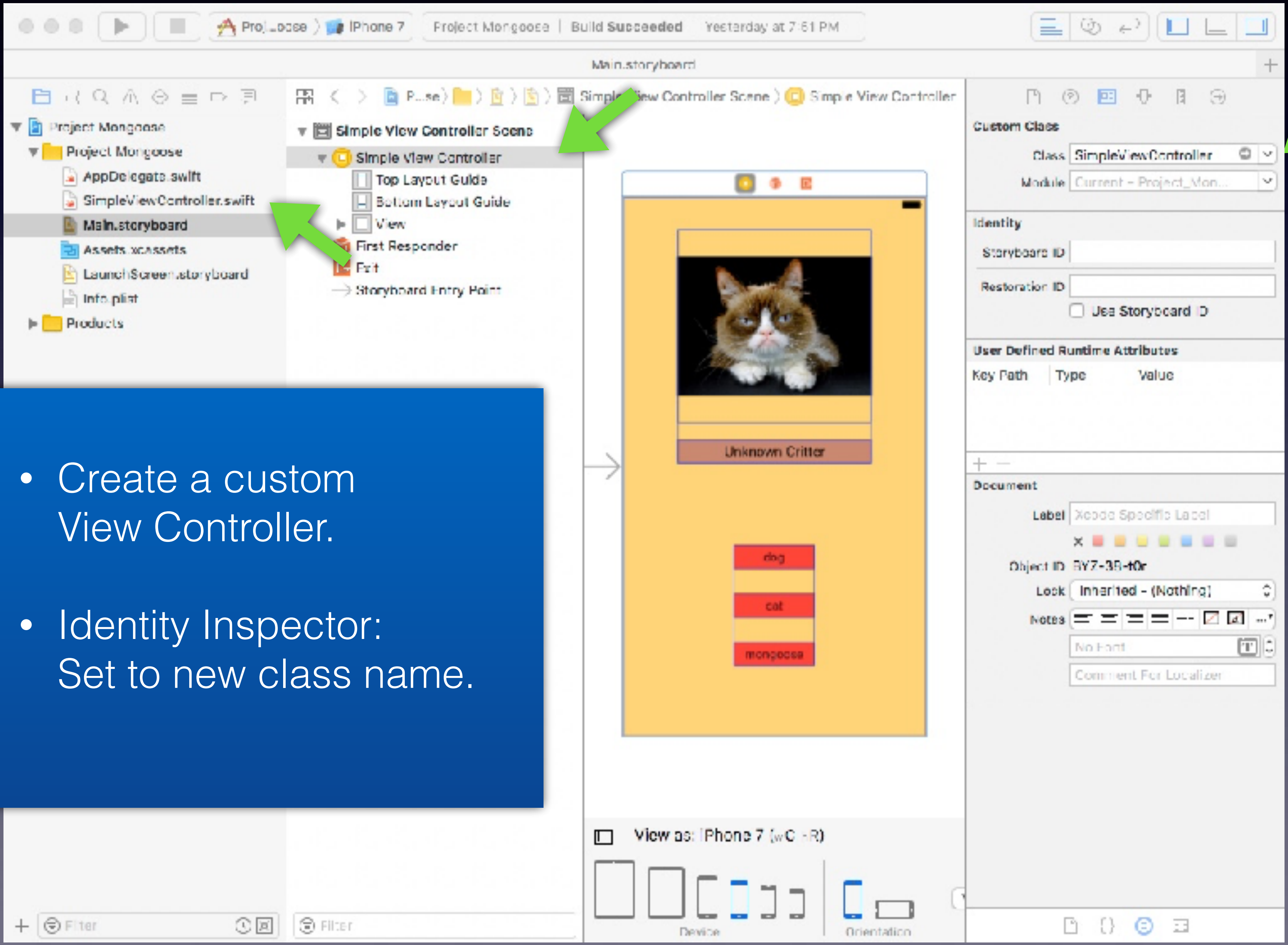
move stacks and set constraints for this view

# Layout Completed





# Coding Mongoose



Class: SimpleViewController

Module: Current – Project_Mon...

Storyboard ID:

Restoration ID:

Use Storyboard ID

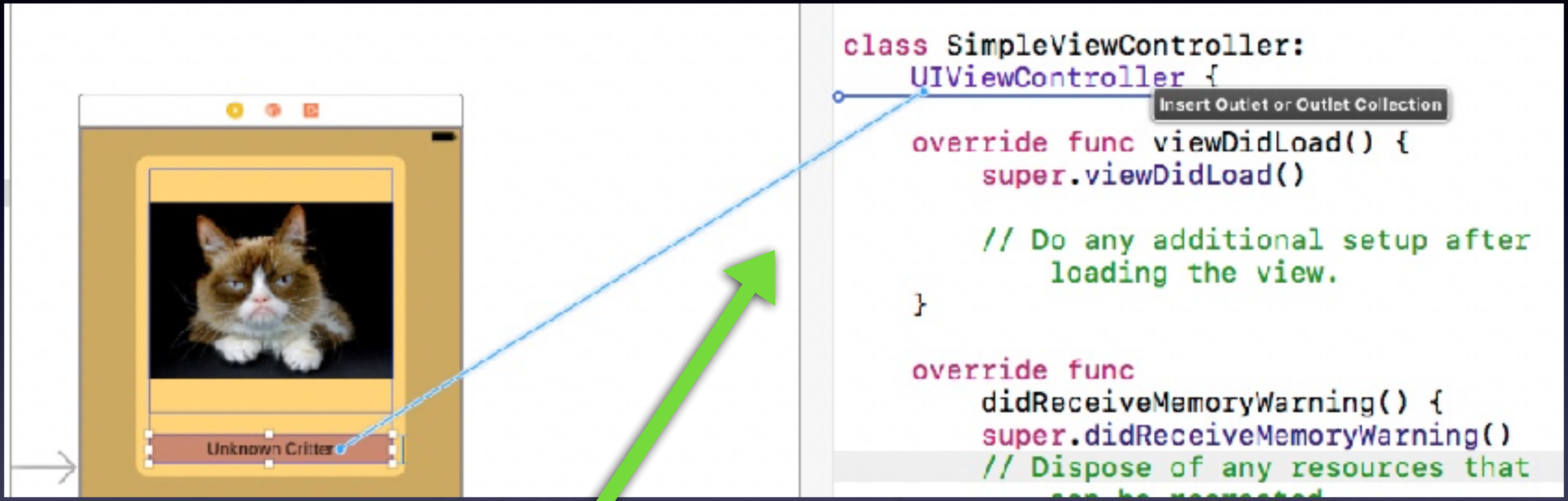
Label: Xcode Specific Label

Object ID: BYZ-38-t0r

Lock: Inherited - (Nothing)

- Create a custom View Controller.
- Identity Inspector: Set to new class name.

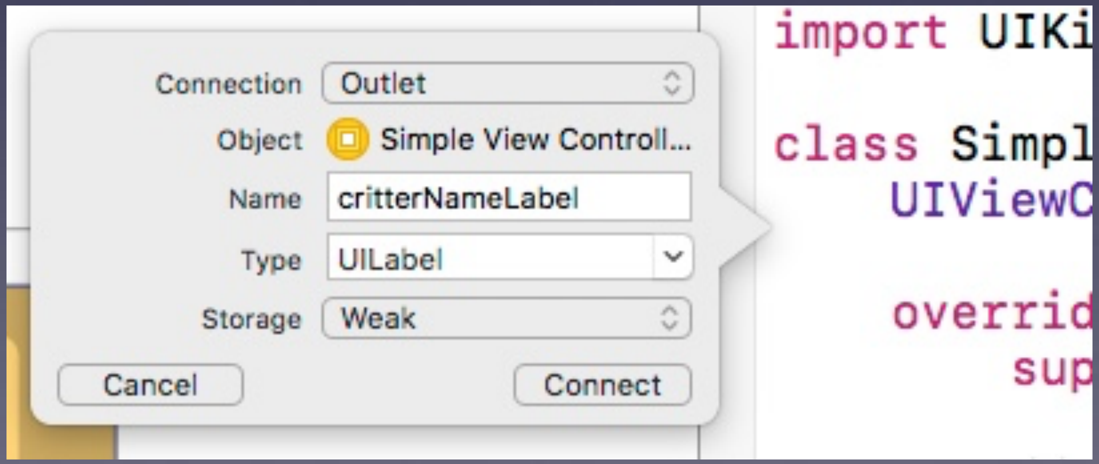
# Outlets



```
class SimpleViewController:
    UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

        // Do any additional setup after
            loading the view.
    }

    override func
        didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that
```

Select Label; Ctrl-drag

**Connection**: Outlet
**Name**: critterNameLabel
**Type**: UILabel
**Connect**

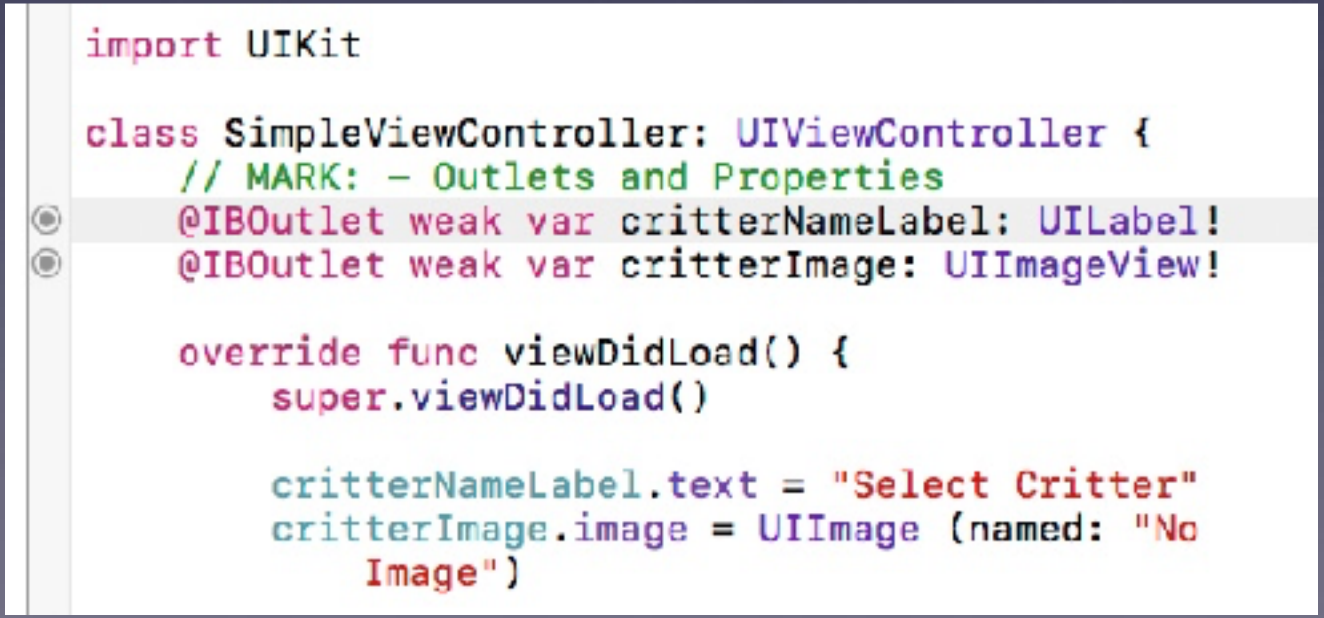


Connection: Outlet
Object: Simple View Controll...
Name: critterNameLabel
Type: UILabel
Storage: Weak

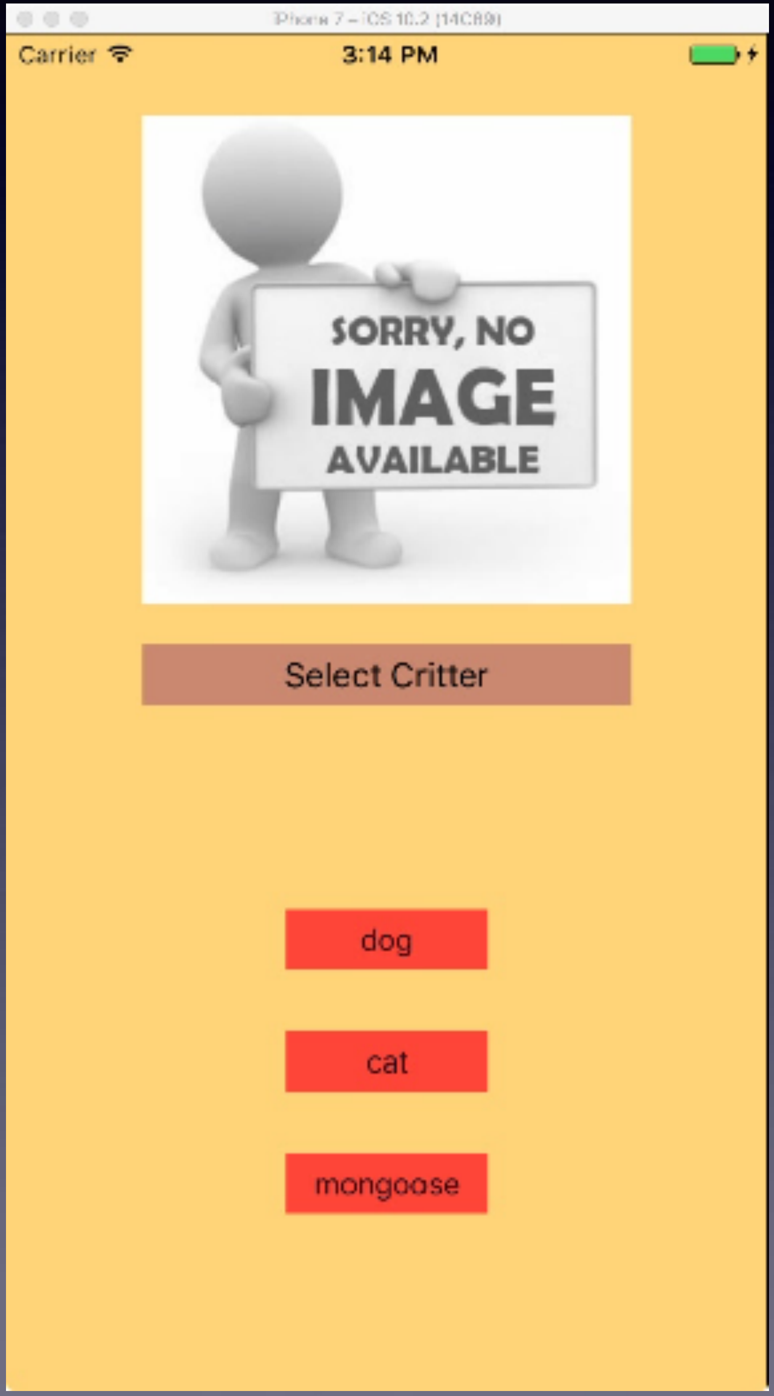
```
import UIKi

class Simpl
    UIViewC

    overrid
        sup
```

# Outlets - as code

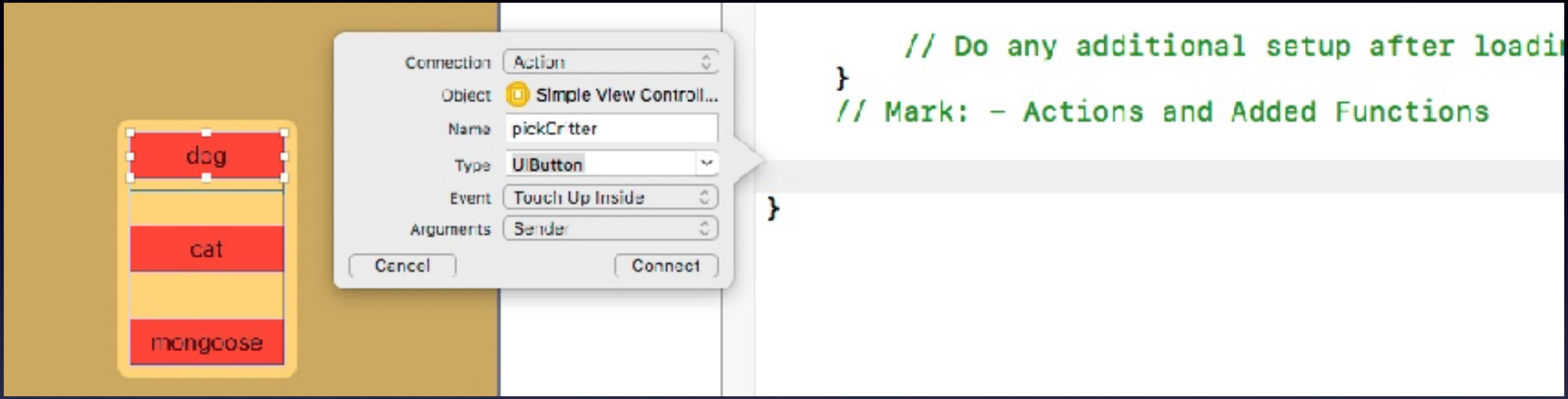
- The two outlets link code in the controller to the image and label in the UI.
- We want to populate both image and label with initial values - we do that in viewDidLoad.



```
import UIKit

class SimpleViewController: UIViewController {
    // MARK: - Outlets and Properties
    @IBOutlet weak var critterNameLabel: UILabel!
    @IBOutlet weak var critterImage: UIImageView!

    override func viewDidLoad() {
        super.viewDidLoad()

        critterNameLabel.text = "Select Critter"
        critterImage.image = UIImage (named: "No
            Image")
```



# Action

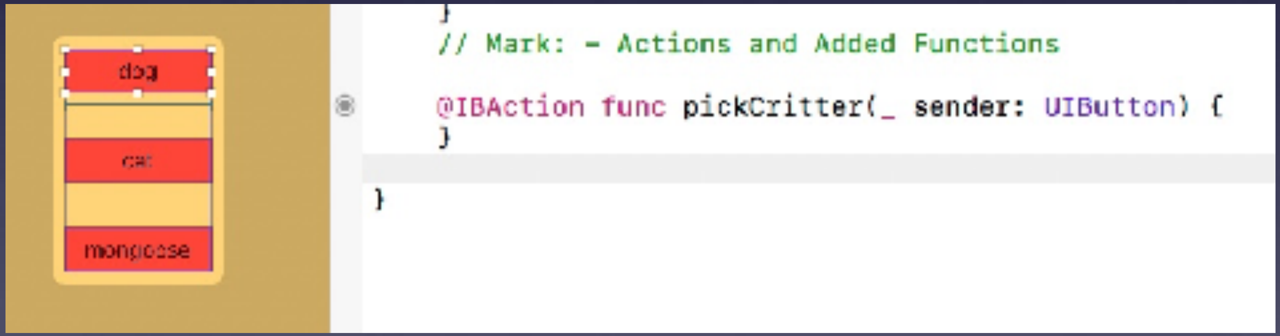


Connection: Action
Object: Simple View Controll...
Name: pickCritter
Type: UIButton
Event: Touch Up Inside
Arguments: Sender

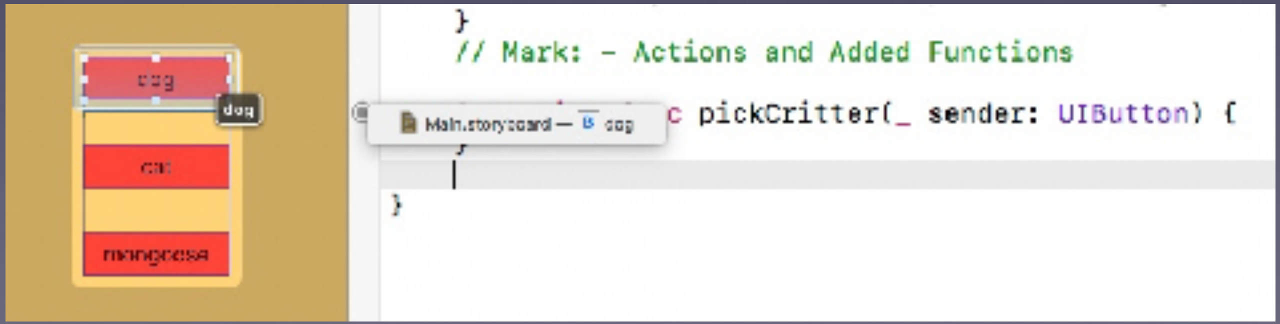
```
    // Do any additional setup after loadi
}
// Mark: - Actions and Added Functions

}
```

Select dog Button;  Ctrl-drag

**Connection**:  Action
**Name**:  name for Action “pickCritter”
**Type**:  UIButton
**Connect**

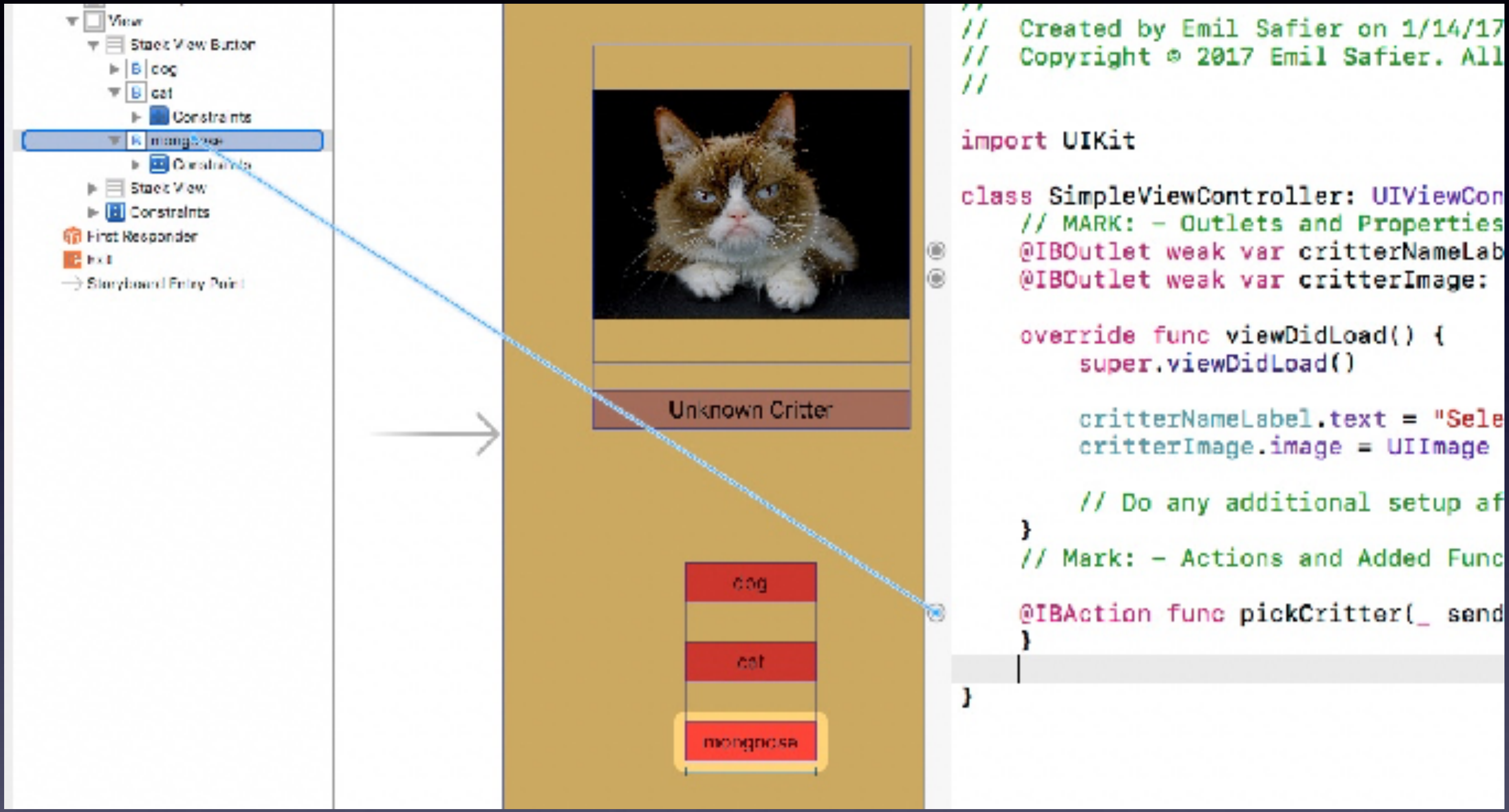


```
}
// Mark: - Actions and Added Functions

@IBAction func pickCritter(_ sender: UIButton) {
}

}
```

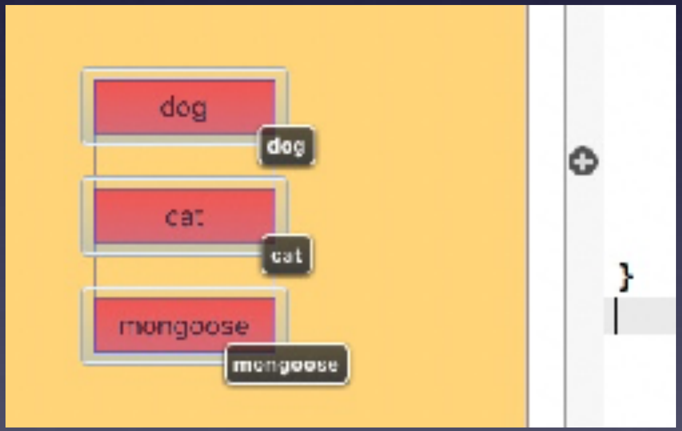


```
}
// Mark: - Actions and Added Functions

c pickCritter(_ sender: UIButton) {
}

}
```

# 3 Buttons:  1 Action



```
//  Created by Emil Safier on 1/14/17
//  Copyright © 2017 Emil Safier. All
//

import UIKit

class SimpleViewController: UIViewCon
    // MARK: - Outlets and Properties
    @IBOutlet weak var critterNameLab
    @IBOutlet weak var critterImage:

    override func viewDidLoad() {
        super.viewDidLoad()

        critterNameLabel.text = "Sele
        critterImage.image = UIImage

        // Do any additional setup af
    }
    // Mark: - Actions and Added Func

    @IBAction func pickCritter(_ send
    }

}
```

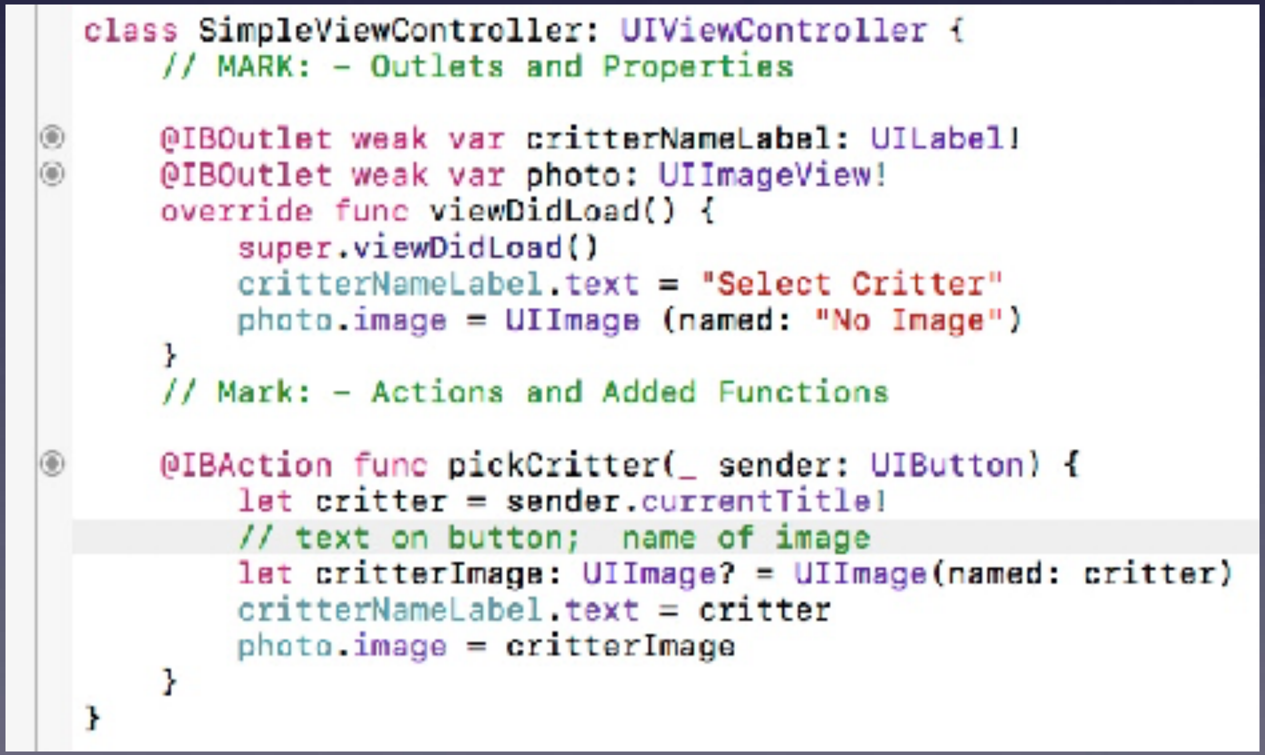


**Select** Action connection;
**Drag** to Mongoose Button in outline view because Buttons are inside Stack View and hard to connect.  Repeat for dog.

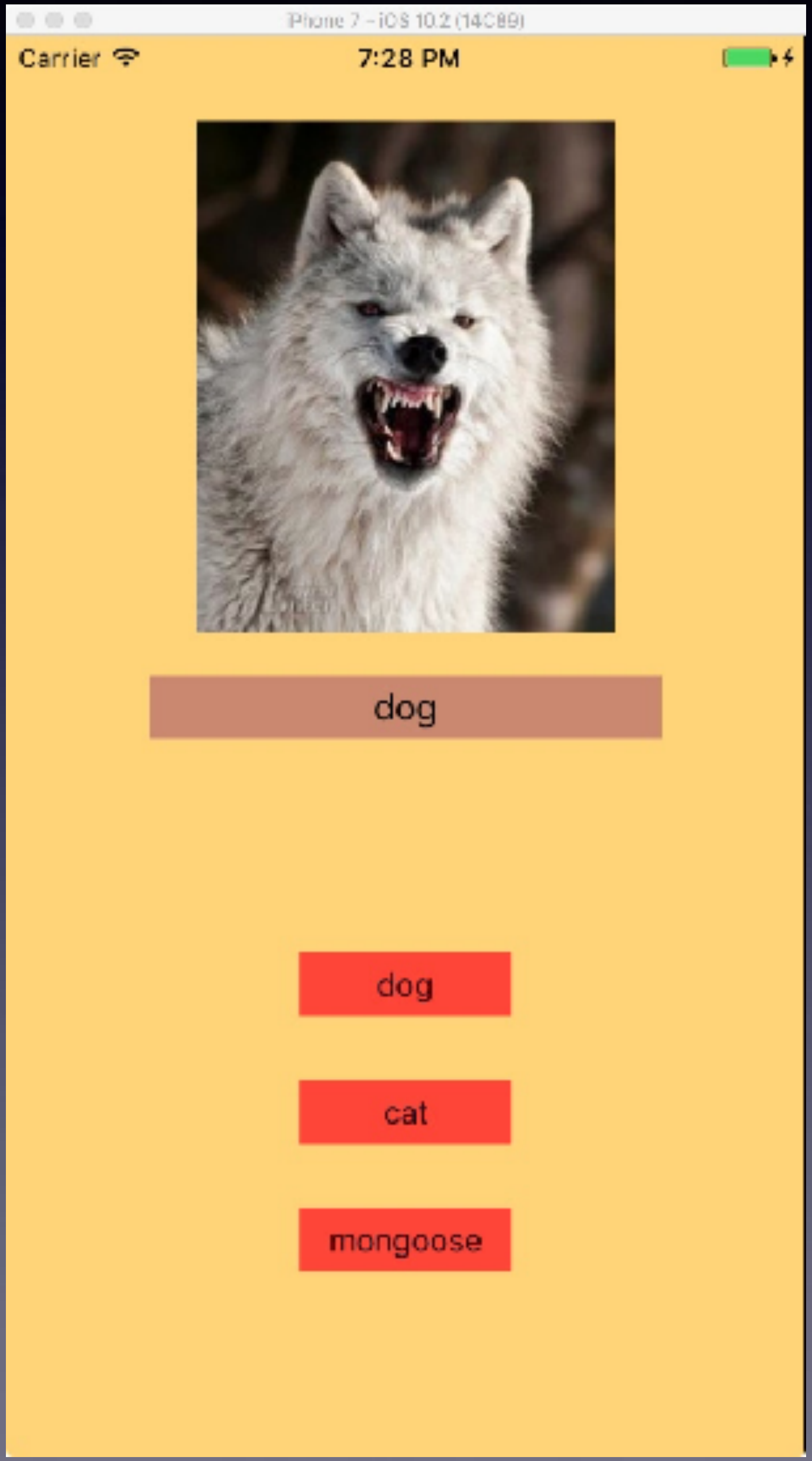
**Select** Action connection;

Now all three buttons are connected

## Add Swift code



```swift
class SimpleViewController: UIViewController {
    // MARK: - Outlets and Properties

    @IBOutlet weak var critterNameLabel: UILabel!
    @IBOutlet weak var photo: UIImageView!
    override func viewDidLoad() {
        super.viewDidLoad()
        critterNameLabel.text = "Select Critter"
        photo.image = UIImage (named: "No Image")
    }
    // Mark: - Actions and Added Functions

    @IBAction func pickCritter(_ sender: UIButton) {
        let critter = sender.currentTitle!
        // text on button;  name of image
        let critterImage: UIImage? = UIImage(named: critter)
        critterNameLabel.text = critter
        photo.image = critterImage
    }
}
```



# Quantum Films
s o f t w a r e

*presenter* Emil Safier

@EmilSafier
emil535@Gmail.com