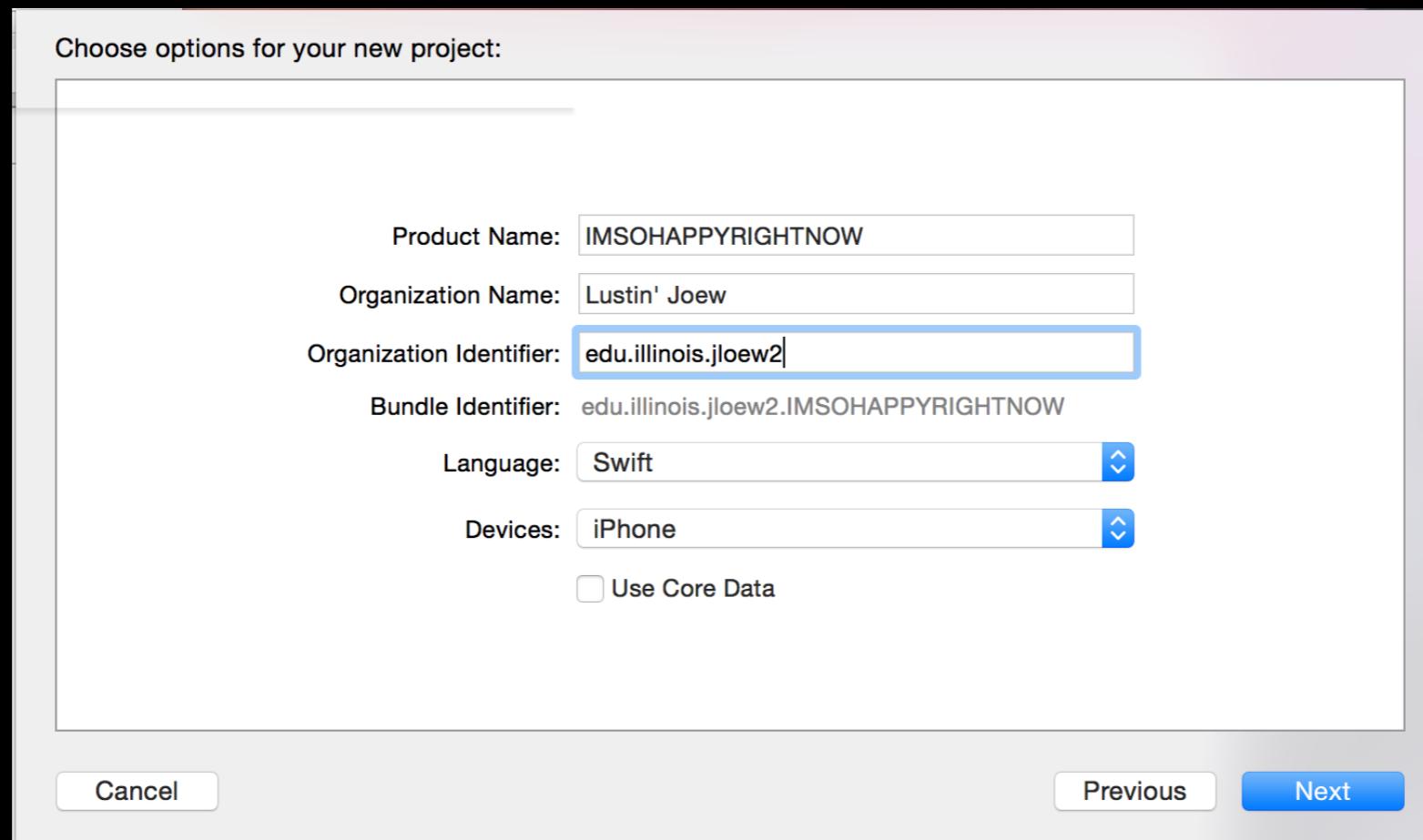


First, let's create a new Xcode project. A Single View Application. Language swift. Targeted toward iPhone.



New

Add Files to “IMSOHAPPYRIGHTNOW”... ⌘A

Open... ⌘O

Open Recent ►

Open Quickly... ⌘O

Close Window ⌘W

Close Tab

Close “Main.storyboard” ⌘W

Close Project ⌘W

Save ⌘S

Duplicate... ⌘S

Revert to Saved

Tab

⌘T

Window

⇧⌘T

File...

⌘N

Playground... ⌘⇧N

Target...

Project...

⇧⌘N

Workspace... ⌘N

^⌘N

Group ⌘N

Group from Selection

Choose a template for your new file:

iOS

Source

User Interface

Core Data

Resource

Other

OS X

Source

User Interface

Core Data

Resource

Other



Cocoa Class



Test Case Class



Playground



Swift File



Objective-C File



Header File



C File



C++ File

Swift File

An empty Swift file.

Cancel

Previous

Next

Save As: FaceView.swift



Tags:

Devices

- rmbp
- SSD256
- Remote Disc

Favorites

- Recents
- Dropbox
- iCloud Drive
- Applications
- Desktop
- Documents
- Downloads
- joloew

Shared

- imac

IMSOHAPPYRIGHTNOW

IMSOHAPPY...W.xcodeproj

IMSOHAPPY...TNOWTests

AppDelegate.swift

Base.lproj

Images.xcassets

Info.plist

ViewController.swift

13

RIGHTNOW

emo

file

Group IMSOHAPPYRIGHTNOW

Targets IMOHOAPPYRIGHTNOW
 IMOHOAPPYRIGHTNOWTests

New Folder

Cancel

Create

```
1 // FaceView.swift
2 // IMSOHAPPYRIGHTNOW
3 //
4 // Created by Justin Loew on 10/4/14.
5 // Copyright (c) 2014 Lustin' Joew. All rights reserved.
6 //
7
8 import Foundation
9
10 import UIKit
11
12
13
14 class FaceView: UIView {
15
16     var scale: Float = 0.90 {
17         // didSet is called every time scale is set (after it has the new value)
18         didSet {
19             // don't allow zero scale
20             if scale == 0 {
21                 scale = Float(0.90)
22             }
23             // any time our scale changes, call for redraw
24             setNeedsDisplay()
25         }
26     }
27 }
28
29 }
```

Add a 'scale' property to our FaceView class. Make sure FaceView inherits from UIView (FaceView: UIView) and you import UIKit.

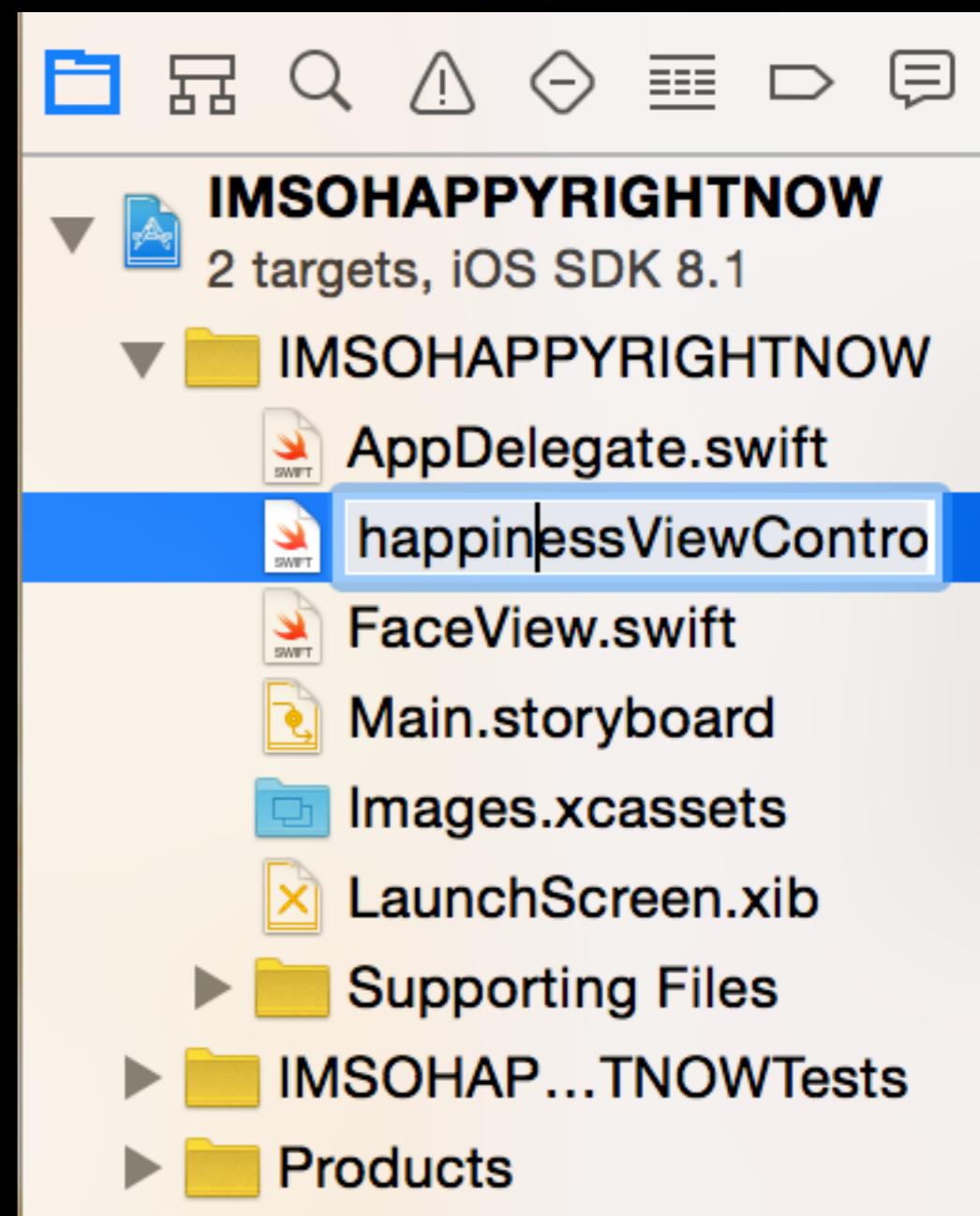
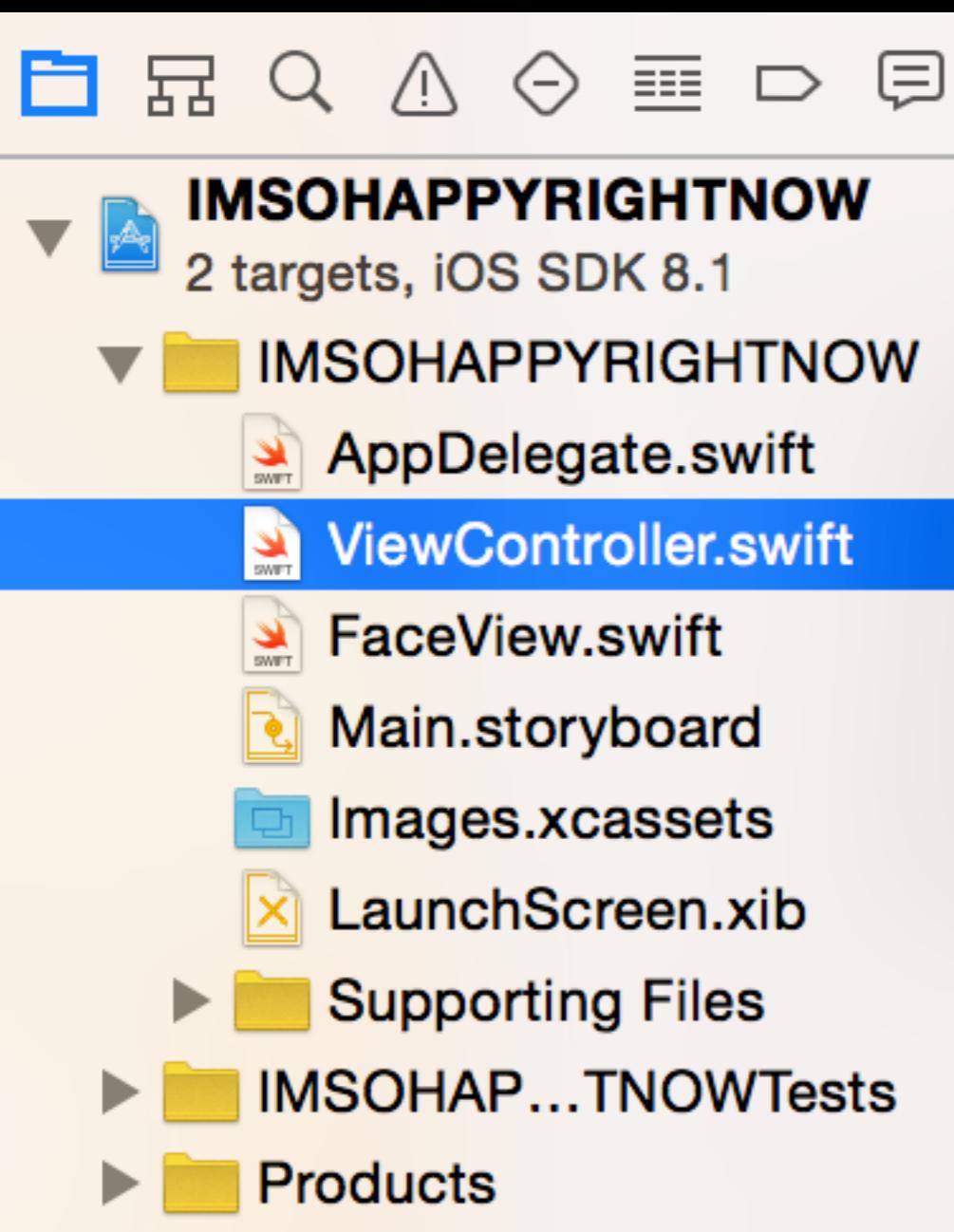
```
1 // FaceView.swift
2 // IMSOHAPPYRIGHTNOW
3 //
4 // Created by Justin Loew on 10/4/14.
5 // Copyright (c) 2014 Lustin' Joew. All rights reserved.
6 //
7 //
8 import Foundation
9
10
11 import UIKit
12
13
14 class FaceView: UIView {
15
16     var scale: Float = 0.90 {
17         // didSet is called every time scale is set (after it has the new value)
18         didSet {
19             // don't allow zero scale
20             if scale == 0 {
21                 scale = Float(0.90)
22             }
23             // any time our scale changes, call for redraw
24             setNeedsDisplay()
25         }
26     }
27
28
29     var dataSource: FaceViewDataSource? = nil
30
31     override init(frame: CGRect) {
32         super.init(frame: frame)
33     }
34
35     required init(coder aDecoder: NSCoder) {
36         super.init(coder: aDecoder)
37     }
38
39
40     func drawCircleAtPoint(p: CGPoint, withRadius r: CGFloat, inContext context: CGContextRef) {
41         UIGraphicsPushContext(context)
42
43         CGContextBeginPath(context)
44         CGContextAddArc(context, p.x, p.y, r, CGFloat(0.0), CGFloat(2*M_PI), 1)
45         CGContextStrokePath(context)
46
47         UIGraphicsPopContext()
48     }
49
50     func pinch(gesture: UIPinchGestureRecognizer) {
51         if (gesture.state == .Changed) || (gesture.state == .Ended) {
52             scale *= Float(gesture.scale) // adjust our scale
53             gesture.scale = 1 // reset gestures scale to 1 (so future changes are incremental, not cumulative)
54         }
55     }
56
57
58 protocol FaceViewDataSource {
59     func smileForFaceView(sender: FaceView)
60 }
```

At the bottom, there's a [FaceViewDataSource protocol](#). There are a few initializers, a `drawCircleAtPoint` method full of wizardry, and a `pinch` method.

IMSOHAPPYRIGHTNOW > IMSOHAPPYRIGHTNOW > FaceView.swift > drawRect(_)

```
55 }  
56  
57 override func drawRect(rect: CGRect) {  
58     var context = UIGraphicsGetCurrentContext()  
59  
60     var midpoint = CGPointMake(self.bounds.origin.x + self.bounds.size.width/2, self.bounds.origin.y + self.bounds.size.height/2)  
61  
62     var faceSize: CGFloat  
63     if self.bounds.size.width < self.bounds.size.height {  
64         faceSize = self.bounds.size.width / 2.0 * self.scale  
65     } else {  
66         faceSize = self.bounds.size.height / 2 * self.scale  
67     }  
68  
69     CGContextSetLineWidth(context, 5)  
70     UIColor.blueColor().setStroke()  
71  
72     drawCircleAtPoint(midpoint, withRadius: faceSize, inContext: context)  
73  
74     let EYE_H = CGFloat(0.35)  
75     let EYE_V = CGFloat(0.35)  
76     let EYE_RADIUS = CGFloat(0.10)  
77  
78     var eyePoint = CGPointMake(midpoint.x - faceSize * EYE_H, midpoint.y - faceSize * EYE_V)  
79  
80     drawCircleAtPoint(eyePoint, withRadius: faceSize * EYE_RADIUS, inContext: context) // left eye  
81     eyePoint.x += faceSize * EYE_H * 2  
82     drawCircleAtPoint(eyePoint, withRadius: faceSize * EYE_RADIUS, inContext: context) // right eye  
83  
84     let MOUTH_H = CGFloat(0.45)  
85     let MOUTH_V = CGFloat(0.40)  
86     let MOUTH_SMILE = CGFloat(0.25)  
87  
88     var mouthStart = CGPointMake(midpoint.x - MOUTH_H * faceSize, midpoint.y + MOUTH_V * faceSize)  
89     var mouthEnd: CGPoint = mouthStart  
90     mouthEnd.x += MOUTH_H * faceSize * 2  
91     var mouthCP1: CGPoint = mouthStart  
92     mouthCP1.x += MOUTH_H * faceSize * 2/3  
93     var mouthCP2: CGPoint = mouthEnd  
94     mouthCP2.x -= MOUTH_H * faceSize * 2/3  
95  
96     var smile = CGFloat(dataSource!.smileForFaceView(self))  
97     if smile < -1 {  
98         smile = -1;  
99     }  
100    if smile > 1 {  
101        smile = 1;  
102    }  
103  
104    var smileOffset: CGFloat = MOUTH_SMILE * faceSize * smile  
105    mouthCP1.y += smileOffset  
106    mouthCP2.y += smileOffset  
107  
108    CGContextBeginPath(context)  
109    CGContextMoveToPoint(context, mouthStart.x, mouthStart.y)  
110    CGContextAddCurveToPoint(context, mouthCP1.x, mouthCP1.y, mouthCP2.x, mouthCP2.y, mouthEnd.x, mouthEnd.y) // bezier curve  
111    CGContextStrokePath(context)  
112 }  
113  
114  
115  
116 protocol FaceViewDataSource {
```

Hoo boy, this one's a doozy.
It uses CoreGraphics to define lines, then draw them in order to draw the face.



Rename the ViewController to
happinessViewController.

IMSOHAPPYRIGHTNOW > iPhone 6 8.1 IMSOHAPPYRIGHTNOW | Build IMSOHAPPYRIGHTNOW: Succeeded | Today at 19:12

IMSOHAPPYRIGHTNOW
2 targets, iOS SDK 8.1

IMSOHAPPYRIGHTNOW

- AppDelegate.swift
- happines...roller.swift
- FaceView.swift
- Main.storyboard
- Images.xcassets
- LaunchScreen.xib
- Supporting Files
- IMSOHAP...TNOWTests
- Products

IMSOHAPPYRIGHTNOW > IMSOHAPPYRIGHTNOW > happinessViewController.swift > C happinessViewController

```
// happinessViewController.swift
// IMSOHAPPYRIGHTNOW
// Created by Justin Loew on 10/4/14.
// Copyright (c) 2014 Lustin' Joew. All rights reserved.

import UIKit

class happinessViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```

Identity and Type

Name: happinessViewController.swift
Type: Default - Swift Source

Location: Relative to Group
happinessViewController.swift
Full Path: /Users/jloloew/Dropbox/Projects/iOS/IMSOHAPPYRIGHTNOW/IMSOHAPPYRIGHTNOW/happinessViewController.swift

Target Membership

IMSOHAPPYRIGHTNOW
 IMSOHAPPYRIGHTNOWTests

Text Settings

Text Encoding: Default - Unicode (UTF-8)
Line Endings: Default - OS X / Unix (LF)
Indent Using: Tabs
Widths: 4 Tab 4 Indent
 Wrap lines

Source Control

Repository --
Type --
Current Branch --

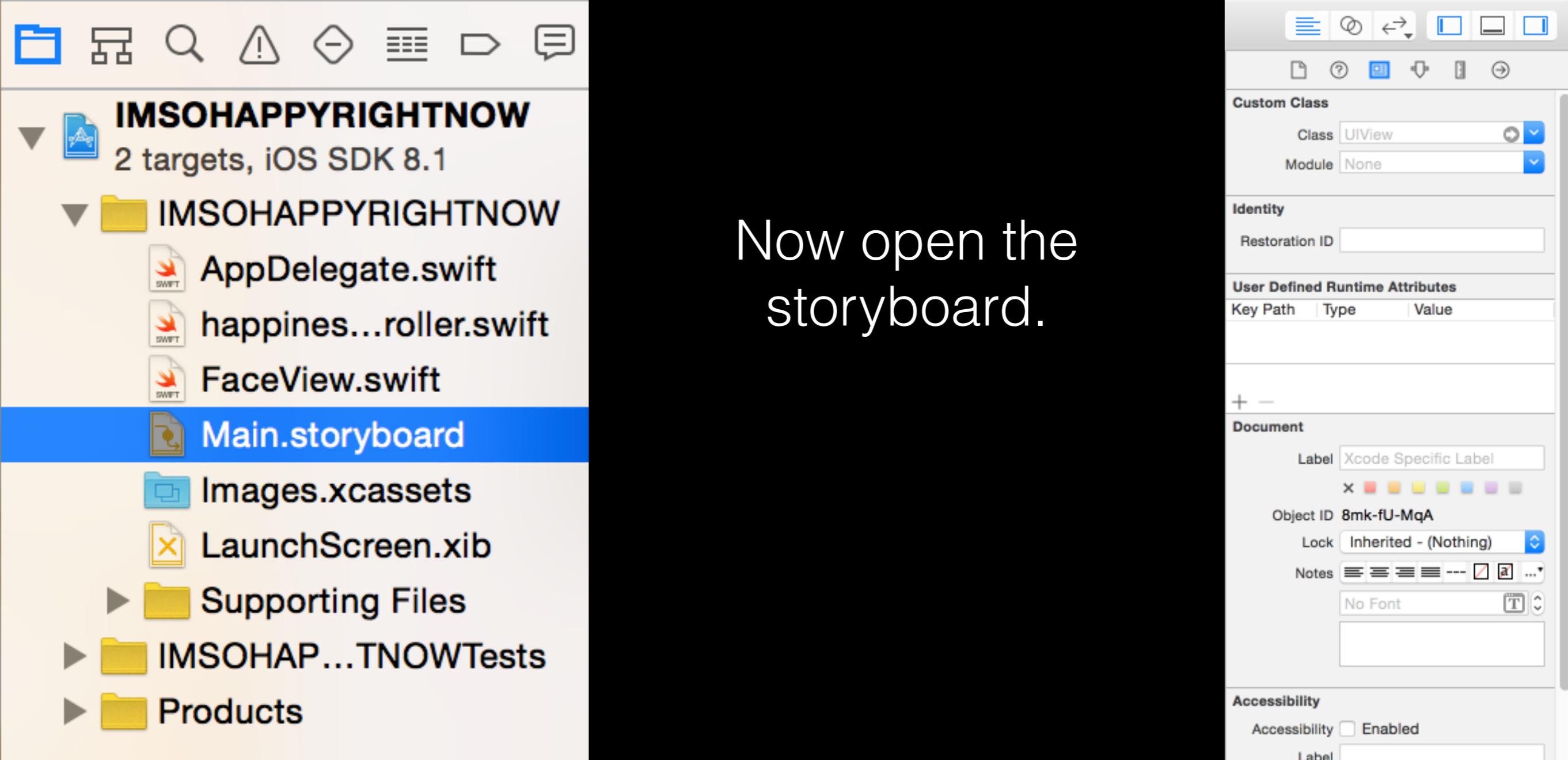
View Controller - A controller that supports the fundamental view-management model in iOS.

Navigation Controller - A controller that manages navigation through a hierarchy of views.

Table View Controller - A controller that manages a table view.

File Navigator Recent Files Search Editor Utilities

Rename the class inside the file, too.



Now open the storyboard.

Custom Class

Class	UIView	+	▼
Module	None	▼	

Identity

Restoration ID	
----------------	--

User Defined Runtime Attributes

Key Path	Type	Value
----------	------	-------

Document

Label Xcode Specific Label

X Y Z C G B R

Object ID 8mk-fU-MqA

Lock Inherited - (Nothing)

Notes

No Font

Accessibility

Enabled

Label

Hint

Traits

Button Link

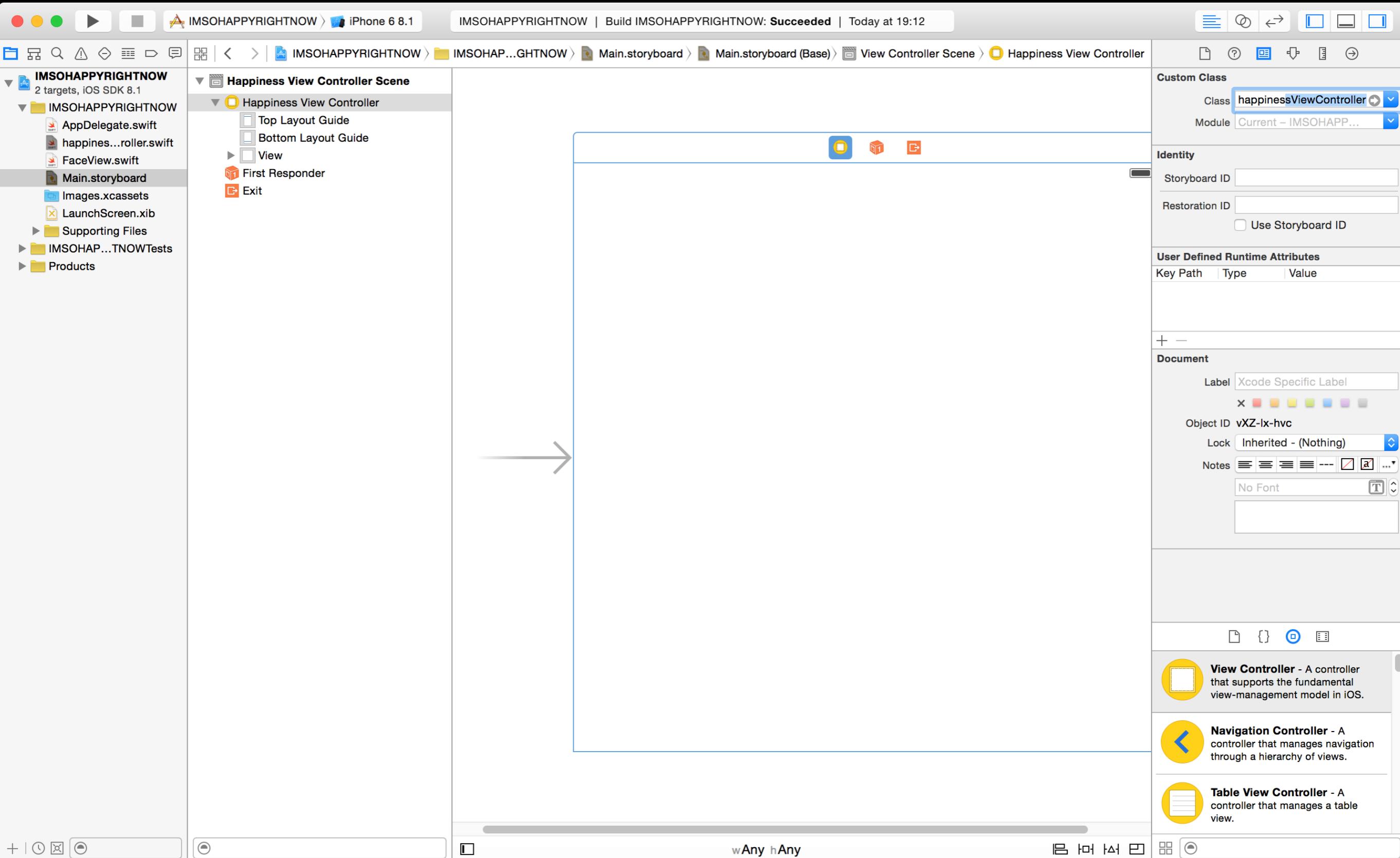
Image Selected

View Controller - A controller that supports the fundamental view-management model in iOS.

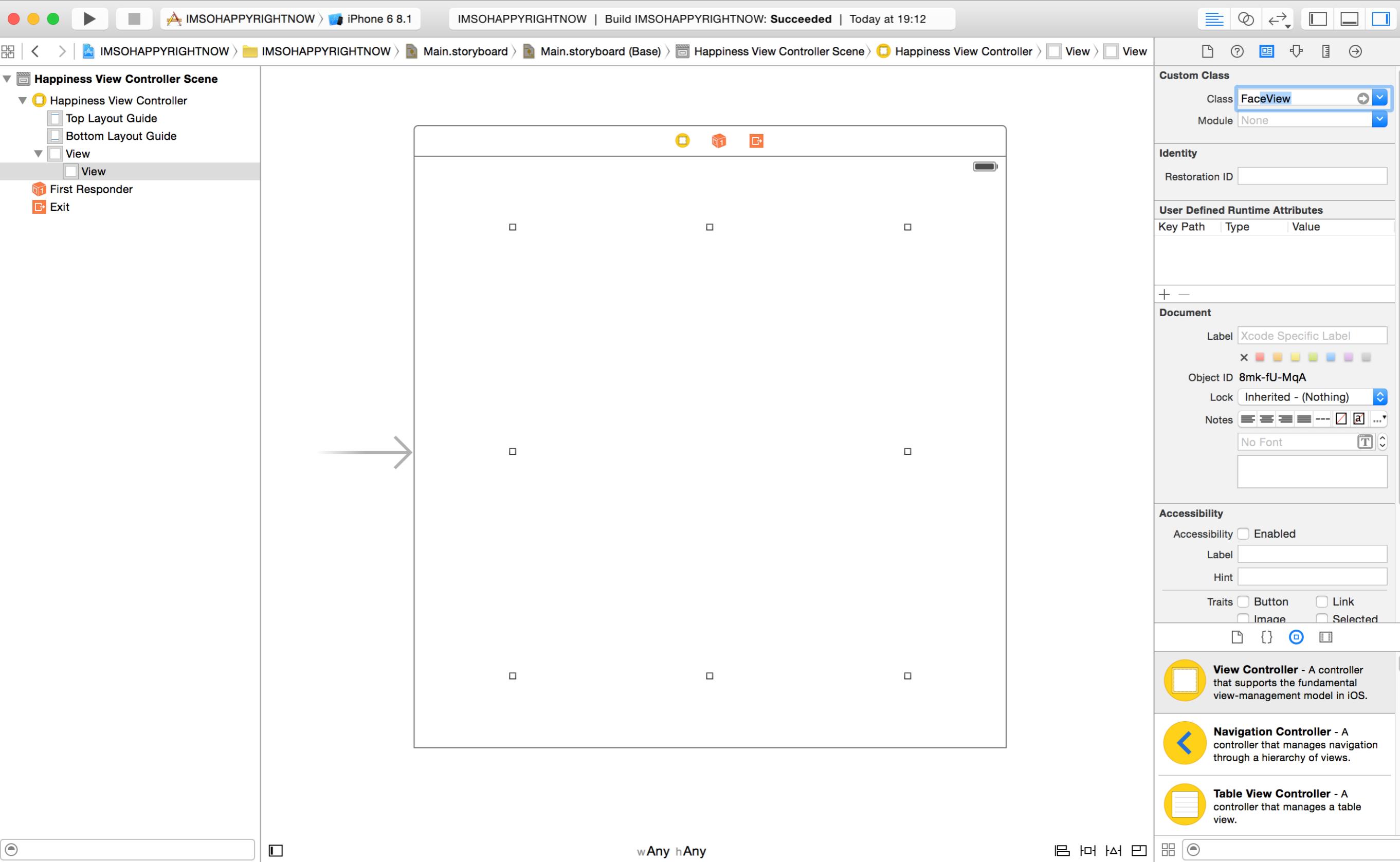
Navigation Controller - A controller that manages navigation through a hierarchy of views.

Table View Controller - A controller that manages a table view.

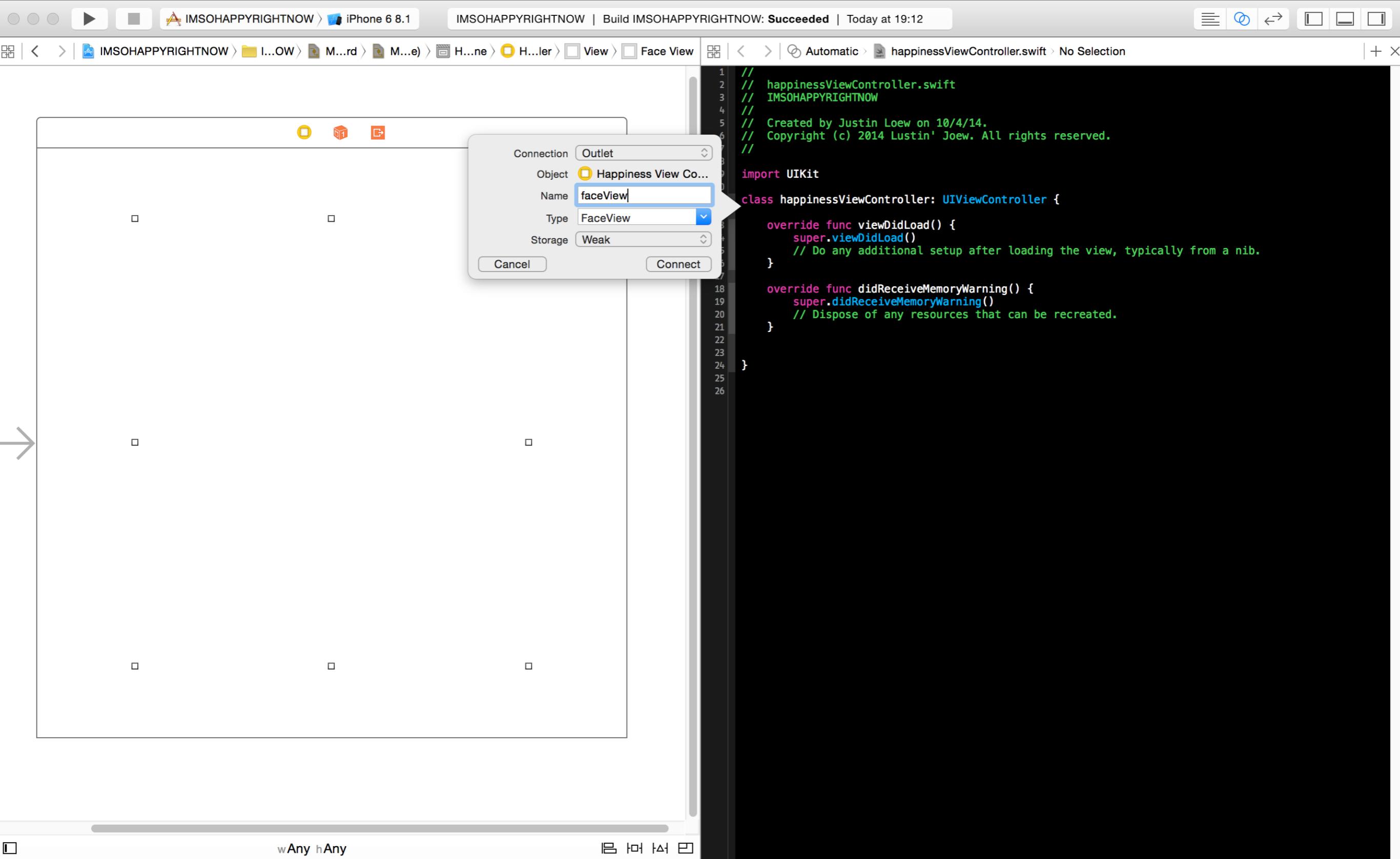
Select the ViewController, and in the Utilities pane on the right, set the class of the ViewController to happinessViewController.



In the bottom right, find a View and drag it into the middle of the happinessViewController. Set its class to FaceView.



Click the linked rings in the top right (where the butler used to be). Right click and drag from the View on the left to the code on the right as shown.



```
// happinessViewController.swift
// IMSOHAPPYRIGHTNOW
// Created by Justin Loew on 10/4/14.
// Copyright (c) 2014 Lustin' Joew. All rights reserved.

import UIKit

class happinessViewController: UIViewController {
    @IBOutlet weak var faceView: FaceView!

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```

Click the button left of the linked rings to switch back to normal mode, then switch to `happinessViewController.swift`.

The screenshot shows the Xcode interface with the following details:

- Top Bar:** Shows the project name "IMSOHAPPYRIGHTNOW", the target "iPhone 6 8.1", and the status "Build Succeeded".
- Left Navigator:** Shows the project structure:
 - Project: IMSOHAPPYRIGHTNOW (2 targets, iOS SDK 8.1)
 - IMSOHAPPYRIGHTNOW folder:
 - AppDelegate.swift
 - happines...roller.swift
 - FaceView.swift
 - Main.storyboard
 - Images.xcassets
 - LaunchScreen.xib
 - Supporting Files
 - IMSOHAP...TNOWTests
 - Products
- Central Editor:** Displays the content of the "happinessViewController.swift" file. The code is as follows:

```
// happinessViewController.swift
// IMSOHAPPYRIGHTNOW
//
// Created by Justin Loew on 10/4/14.
// Copyright (c) 2014 Lustin' Joew. All rights reserved.

import UIKit

class happinessViewController: UIViewController {

    @IBOutlet weak var faceView: FaceView!

    var happiness: Int = 0 { // 0 for sad, 100 for happy
        didSet {
            faceView.setNeedsDisplay()
        }
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```

A callout box highlights the `didSet` block of code.

Bottom Text: A large white text box contains the explanatory text: didSet is automatically called after `faceView` is set.

IMSOHAPPYRIGHTNOW happinessViewController.swift happinessViewController

```
1 // Make sure
2 // happinessViewController.swift
3 // IMSOHAPPYRIGHTNOW
4 // Created by Justin Loew on 10/4/14.
5 // Copyright (c) 2014 Lustin' Joew. All rights reserved.
6 //
7 import UIKit
8
9 class happinessViewController: UIViewController, FaceViewDataSource {
10
11     @IBOutlet weak var faceView: FaceView!
12     didSet {
13         faceView.dataSource = self
14
15         // enable pinch gestures in the FaceView using its pinch() handler
16         faceView.addGestureRecognizer(UIPinchGestureRecognizer(target: faceView, action: "pinch:"))
17
18         faceView.addGestureRecognizer(UIPanGestureRecognizer(target: self, action: "handleHappinessChange:"))
19     }
20
21 }
22
23 var happiness: Int = 0 {    // 0 for sad, 100 for happy
24     didSet {
25         faceView.setNeedsDisplay()
26     }
27 }
28 }
```

Make sure
happinessViewController
implements our
FaceViewDataSource protocol.

IMSOHAPPYRIGHTNOW
2 targets, iOS SDK 8.1

IMSOHAPPYRIGHTNOW

- AppDelegate.swift
- happines...roller.swift
- FaceView.swift
- Main.storyboard
- Images.xcassets
- LaunchScreen.xib

Supporting Files

IMSOHAP...TNOWTests

Products

```
1 // happinessViewController.swift
2 // IMSOHAPPYRIGHTNOW
3 //
4 // Created by Justin Loew on 10/4/14.
5 // Copyright (c) 2014 Lustin' Joew. All rights reserved.
6 //
7 //
8 import UIKit
9
10 class happinessViewController: UIViewController, FaceViewDataSource {
11     @IBOutlet weak var faceView: FaceView!
12     didSet {
13         faceView.dataSource = self
14
15         // enable pinch gestures in the FaceView using its pinch() handler
16         faceView.addGestureRecognizer(UIPinchGestureRecognizer(target: faceView, action: "pinch"))
17
18         faceView.addGestureRecognizer(UIPanGestureRecognizer(target: self, action: "handleHappinessChange"))
19     }
20 }
21
22 var happiness: Int = 0 {    // 0 for sad, 100 for happy
23     didSet {
24         faceView.setNeedsDisplay()
25     }
26 }
27
28
29
30 func handleHappinessChange(gesture: UIPanGestureRecognizer) {
31     if gesture.state == .Changed || gesture.state == .Ended {
32         let translationAmount: CGPoint = gesture.translationInView(faceView)
33         happiness += Int(translationAmount.y / 2)
34         if happiness < 0 {
35             happiness = 0
36         }
37         if happiness > 100 {
38             happiness = 100
39         }
40
41         gesture.setTranslation(CGPointZero, inView: faceView)
42     }
43 }
44
45 override func viewDidLoad() {
46     super.viewDidLoad()
47     // Do any additional setup after loading the view, typically from a nib.
48 }
49
50 override func didReceiveMemoryWarning() {
51     super.didReceiveMemoryWarning()
52     // Dispose of any resources that can be recreated.
53 }
54
55
56 }
```

IMSOHAPPYRIGHTNOW > iPhone 6 8.1 IMSOHAPPYRIGHTNOW | Build IMSOHAPPYRIGHTNOW: Succeeded | Today at 19:12

IMSOHAPPYRIGHTNOW 2 targets, iOS SDK 8.1

IMSOHAPPYRIGHTNOW

- AppDelegate.swift
- happines...roller.swift
- FaceView.swift
- Main.storyboard
- Images.xcassets
- LaunchScreen.xib

Supporting Files

IMSOHAP...TNOWTests

Products

IMSOHAPPYRIGHTNOW > IMSOHAPPYRIGHTNOW > happinessViewController.swift > smileForFaceView(_:)

```
// happinessViewController.swift
// IMSOHAPPYRIGHTNOW
//
// Created by Justin Loew on 10/4/14.
// Copyright (c) 2014 Lustin' Joew. All rights reserved.

import UIKit

class happinessViewController: UIViewController, FaceViewDataSource {

    @IBOutlet weak var faceView: FaceView!
    didSet {
        faceView.dataSource = self
        // enable pinch gestures in the FaceView using its pinch() handler
        faceView.addGestureRecognizer(UIPinchGestureRecognizer(target: faceView, action: "pinch"))

        faceView.addGestureRecognizer(UIPanGestureRecognizer(target: self, action: "handleHappinessChange"))
    }

    var happiness: Int = 0 { // 0 for sad, 100 for happy
        didSet {
            faceView.setNeedsDisplay()
        }
    }

    func handleHappinessChange(gesture: UIPanGestureRecognizer) {
        if gesture.state == .Changed || gesture.state == .Ended {
            let translationAmount: CGPoint = gesture.translationInView(faceView)
            happiness += Int(translationAmount.y / 2)
            if happiness < 0 {
                happiness = 0
            }
            if happiness > 100 {
                happiness = 100
            }

            gesture.setTranslation(CGPointZero, inView: faceView)
        }
    }

    func smileForFaceView(sender: FaceView) -> Float {
        // happiness is 0-100. smile range is -1 to 1.
        return Float(happiness - 50) / Float(50)
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```

The screenshot shows the Xcode interface with the project 'IMSOHAPPYRIGHTNOW' open. The left sidebar displays the project structure with files like 'AppDelegate.swift', 'FaceView.swift', and 'Main.storyboard'. The main editor area shows the 'happinessViewController.swift' code. The code defines a class 'happinessViewController' that implements 'UIViewController' and 'FaceViewDataSource'. It contains methods for handling gestures and calculating a smile value based on happiness. A specific line of code, 'override func shouldAutorotate() -> Bool { return true // support all orientations }', is highlighted in gray.

```
// happinessViewController.swift
// IMSOHAPPYRIGHTNOW
//
// Created by Justin Loew on 10/4/14.
// Copyright (c) 2014 Lustin' Joew. All rights reserved.

import UIKit

class happinessViewController: UIViewController, FaceViewDataSource {

    @IBOutlet weak var faceView: FaceView!
    didSet {
        faceView.dataSource = self

        // enable pinch gestures in the FaceView using its pinch() handler
        faceView.addGestureRecognizer(UIPinchGestureRecognizer(target: faceView, action: "pinch"))

        faceView.addGestureRecognizer(UIPanGestureRecognizer(target: self, action: "handleHappinessChange"))
    }

    var happiness: Int = 0 { // 0 for sad, 100 for happy
        didSet {
            faceView.setNeedsDisplay()
        }
    }

    func handleHappinessChange(gesture: UIPanGestureRecognizer) {
        if gesture.state == .Changed || gesture.state == .Ended {
            let translationAmount: CGPoint = gesture.translationInView(faceView)
            happiness += Int(translationAmount.y / 2)
            if happiness < 0 {
                happiness = 0
            }
            if happiness > 100 {
                happiness = 100
            }
            gesture.setTranslation(CGPointZero, inView: faceView)
        }
    }

    func smileForFaceView(sender: FaceView) -> Float {
        // happiness is 0-100. smile range is -1 to 1.
        return Float(happiness - 50) / Float(50)
    }

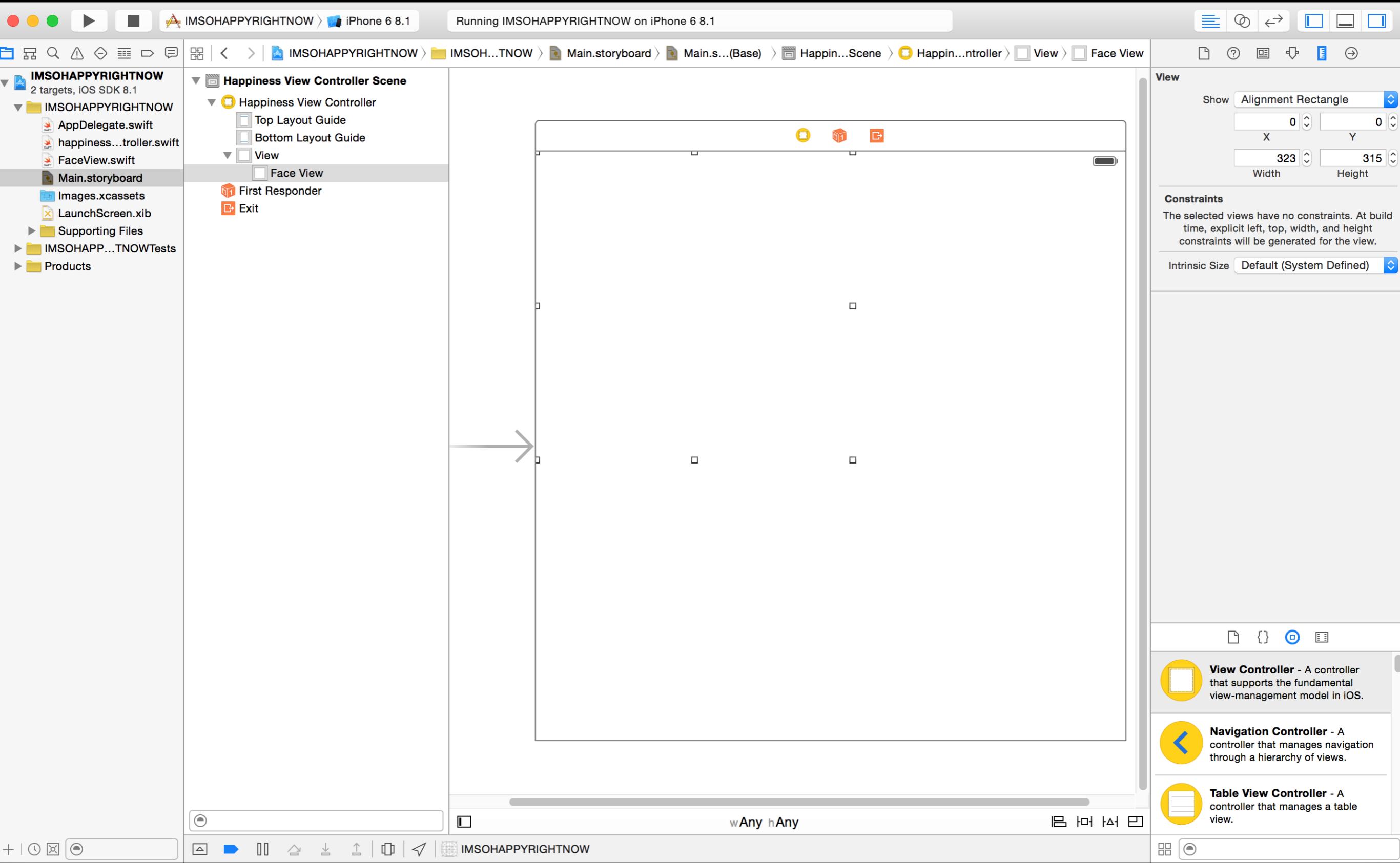
    override func shouldAutorotate() -> Bool {
        return true // support all orientations
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a nib.
    }

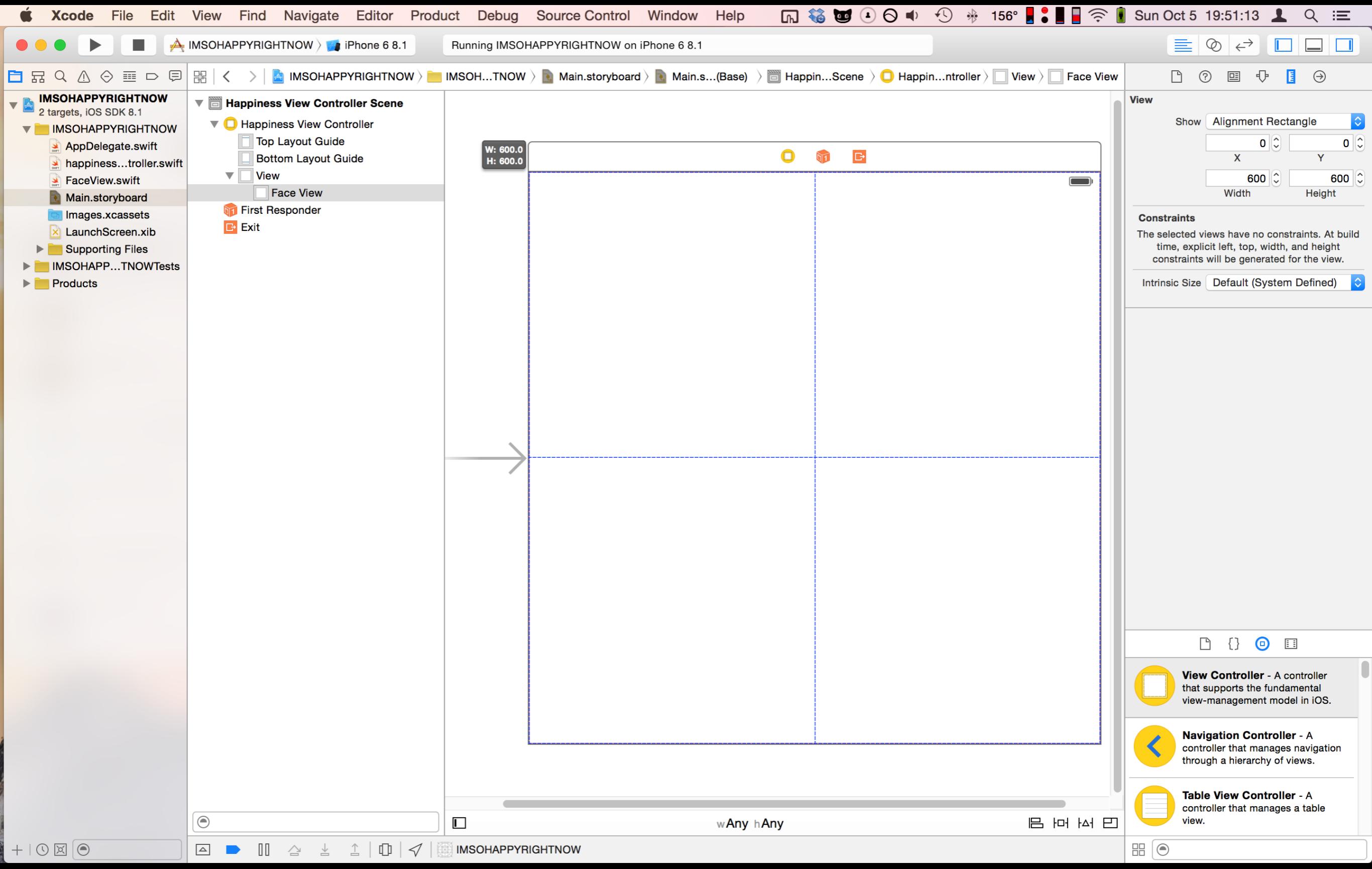
    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```

shouldAutorotate is called to see if the app should turn to landscape mode when the device is tilted.

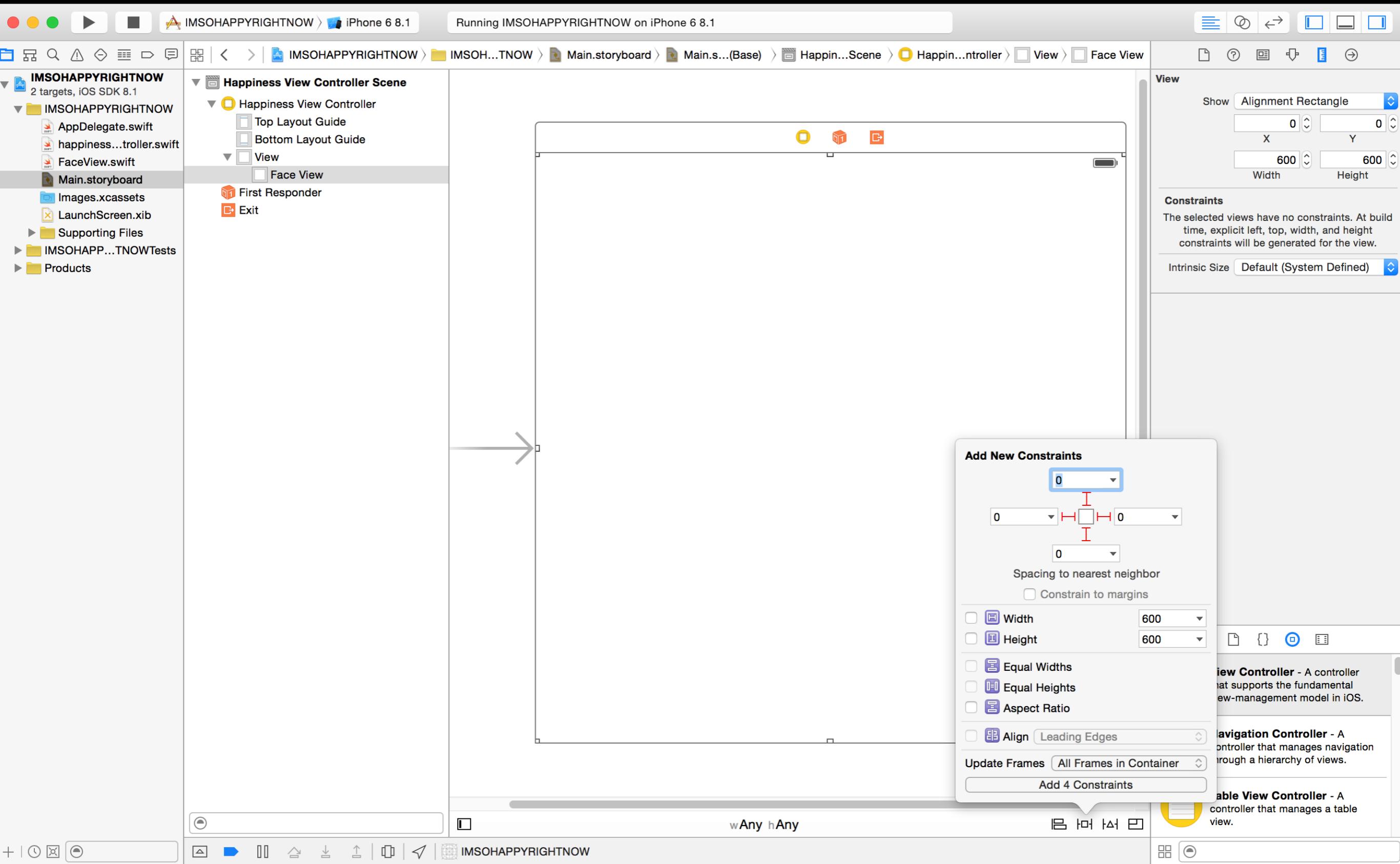
Move the view to the top left corner



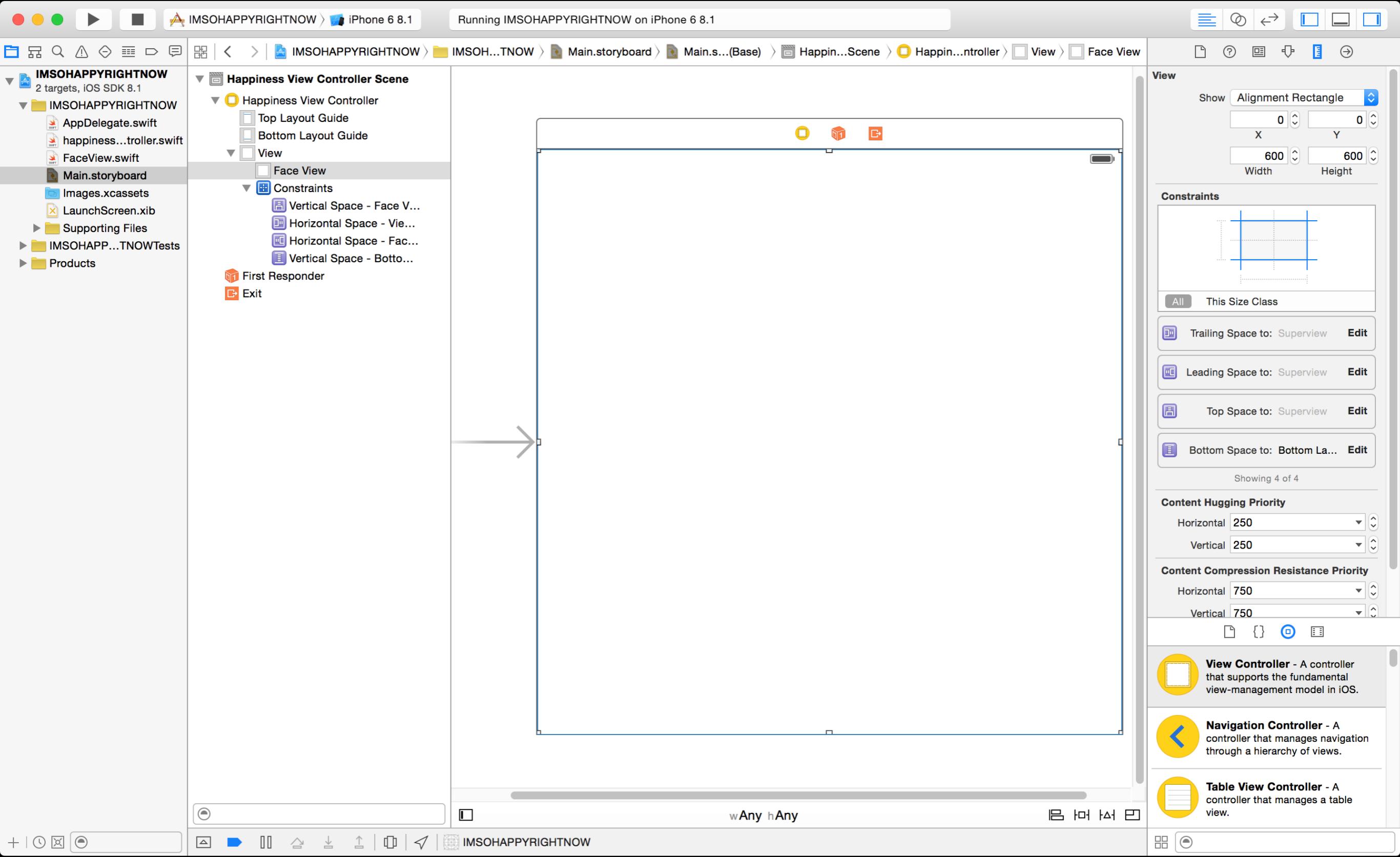
...And expand it to fill the screen.



Use this button to add constraints, so the smiley can autoresize itself for different device sizes. Make sure you click the I-shaped things at the top so they're solid, not dashed.



They should show up in the bar on the left if you did it right.



Great success! Build and Run, and play around with it. Swipe up and down to change the smile to a frown, and pinch to zoom (hold option to get two touches in the simulator).

