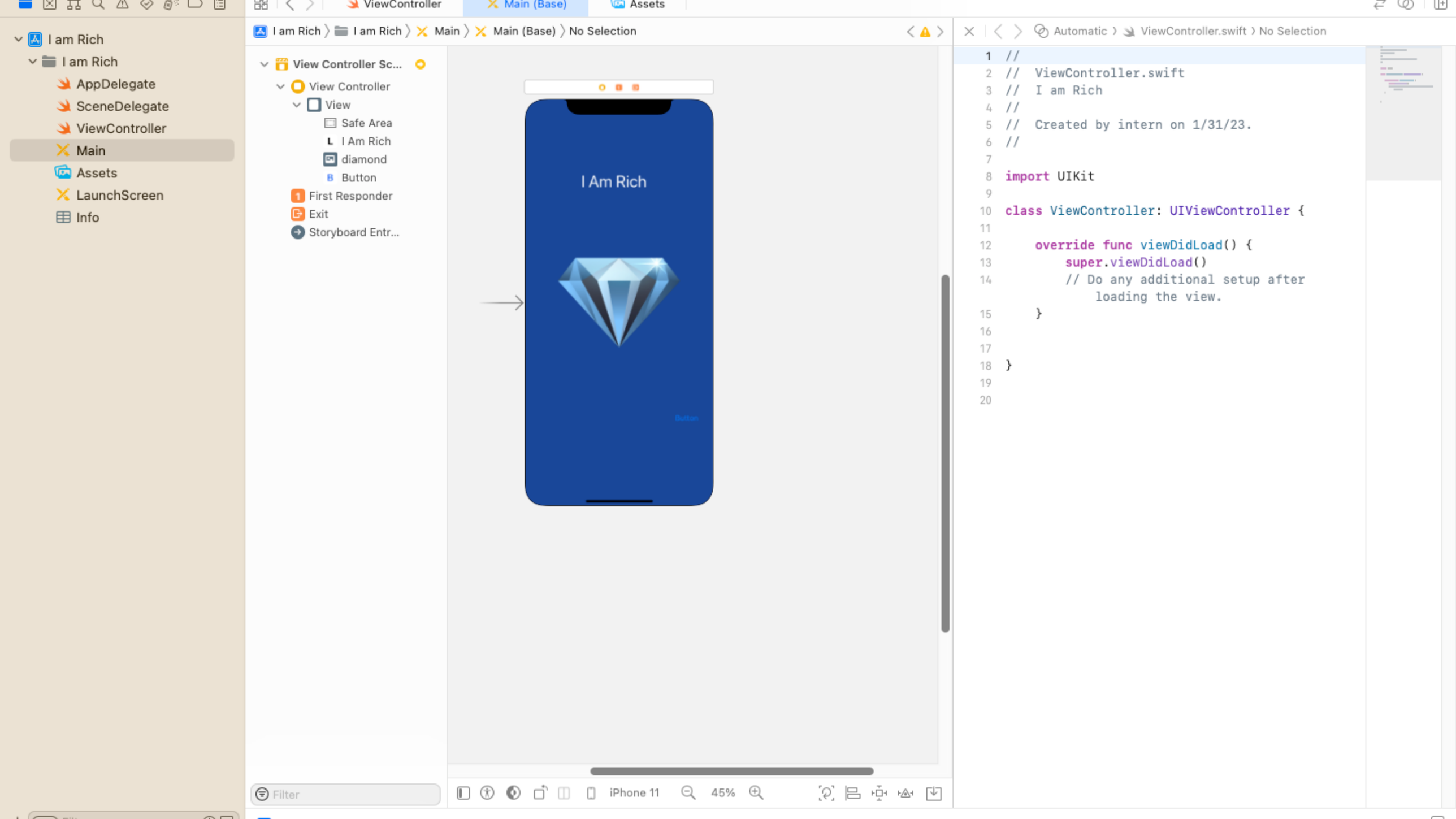
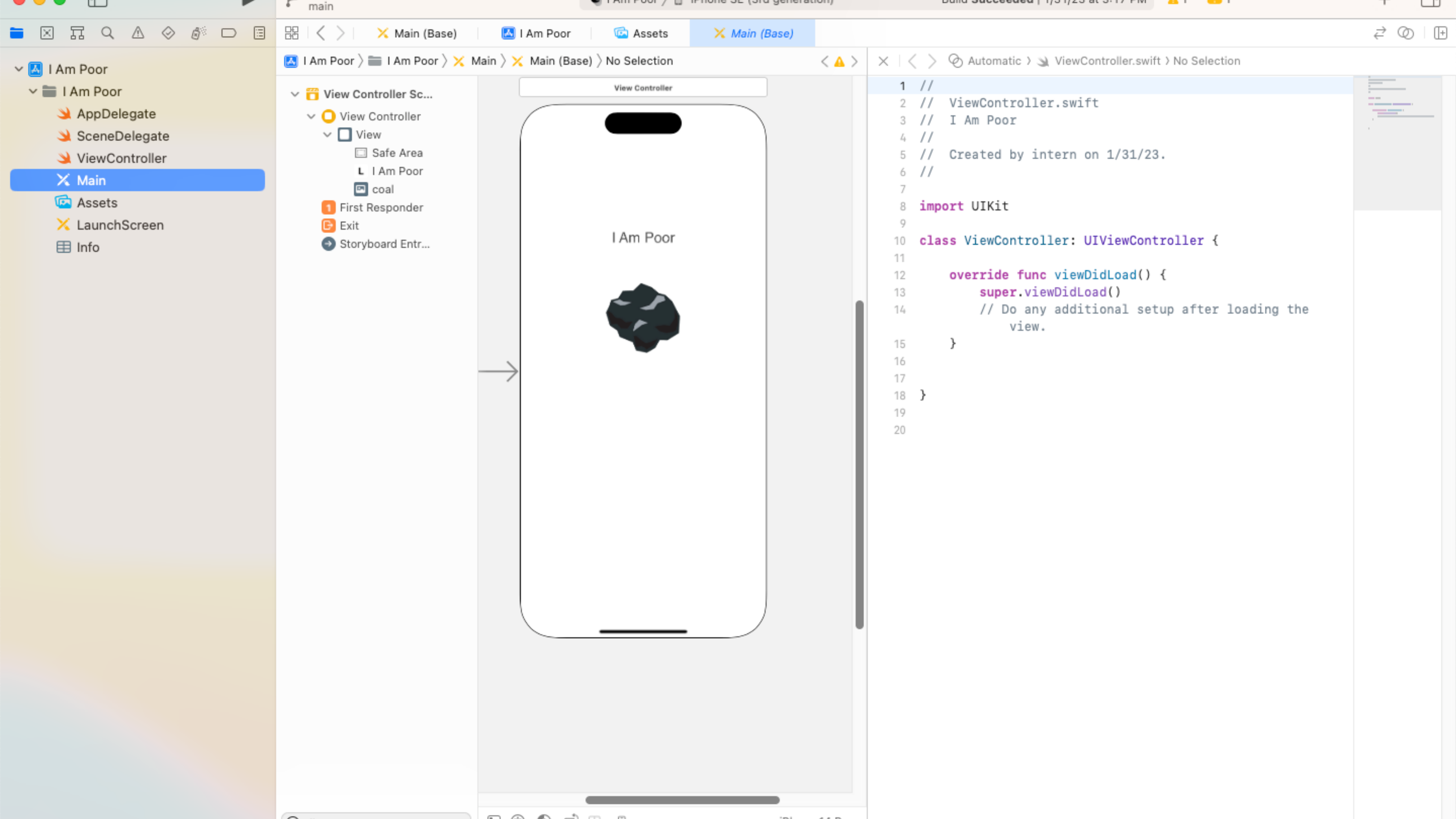
SUMMARY OF LEARNINGS

JANUARY 26 - FEBRUARY 6, 2023

Concepts discussed in iOS and Swift for the past week:

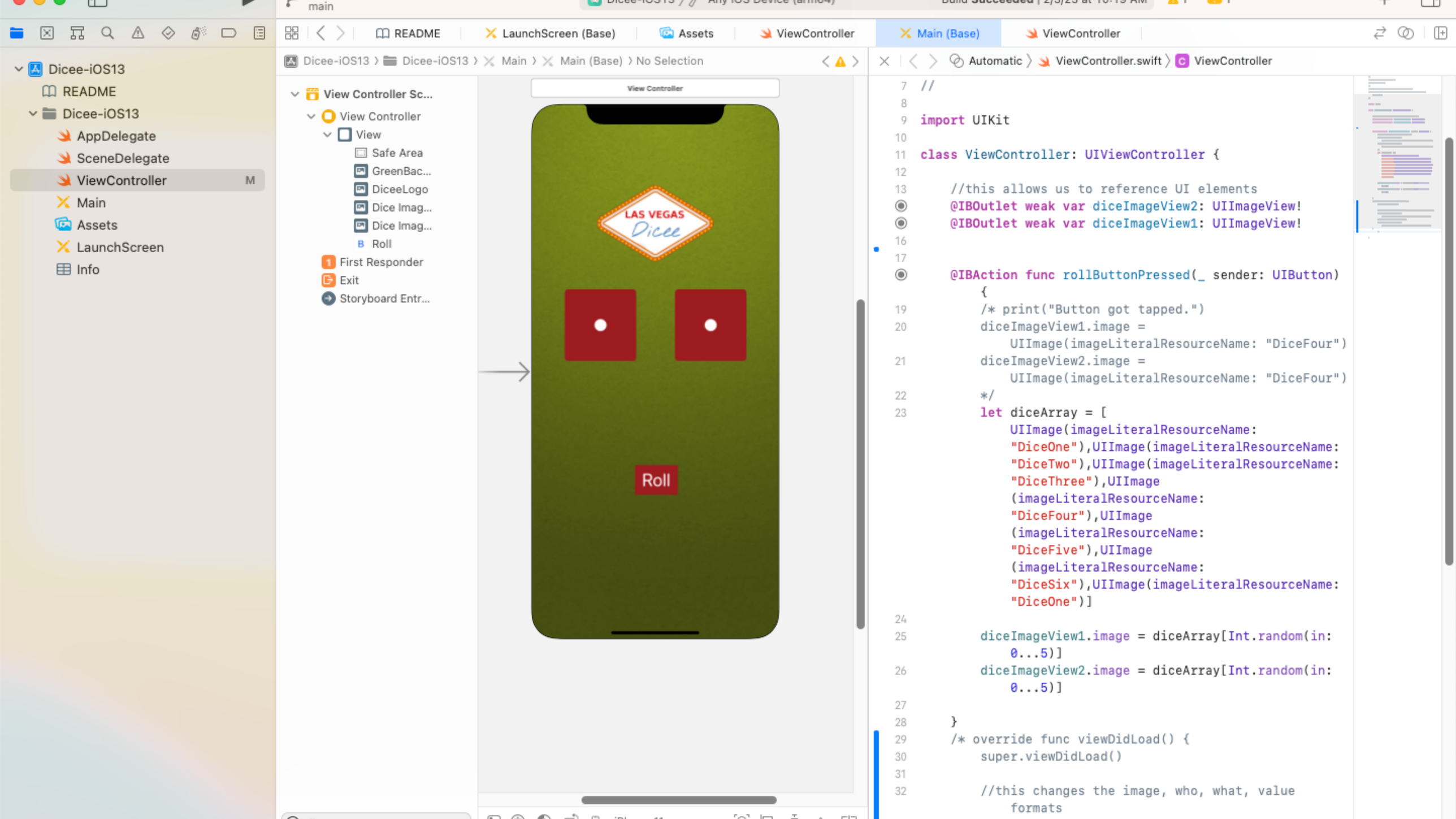
- Introduction to App Development (iOS and Swift)
- Setting up new project on Xcode
- Familiarizing iOS and Xcode Interface Builder
- I Am Rich App
  - Creating and designing new project
  - Incorporating own image assets
  - Design custom app icon
  - Running app on a simulator and iPhone
- I Am Poor App
  - Adding elements from Object Library
  - Adding app icon

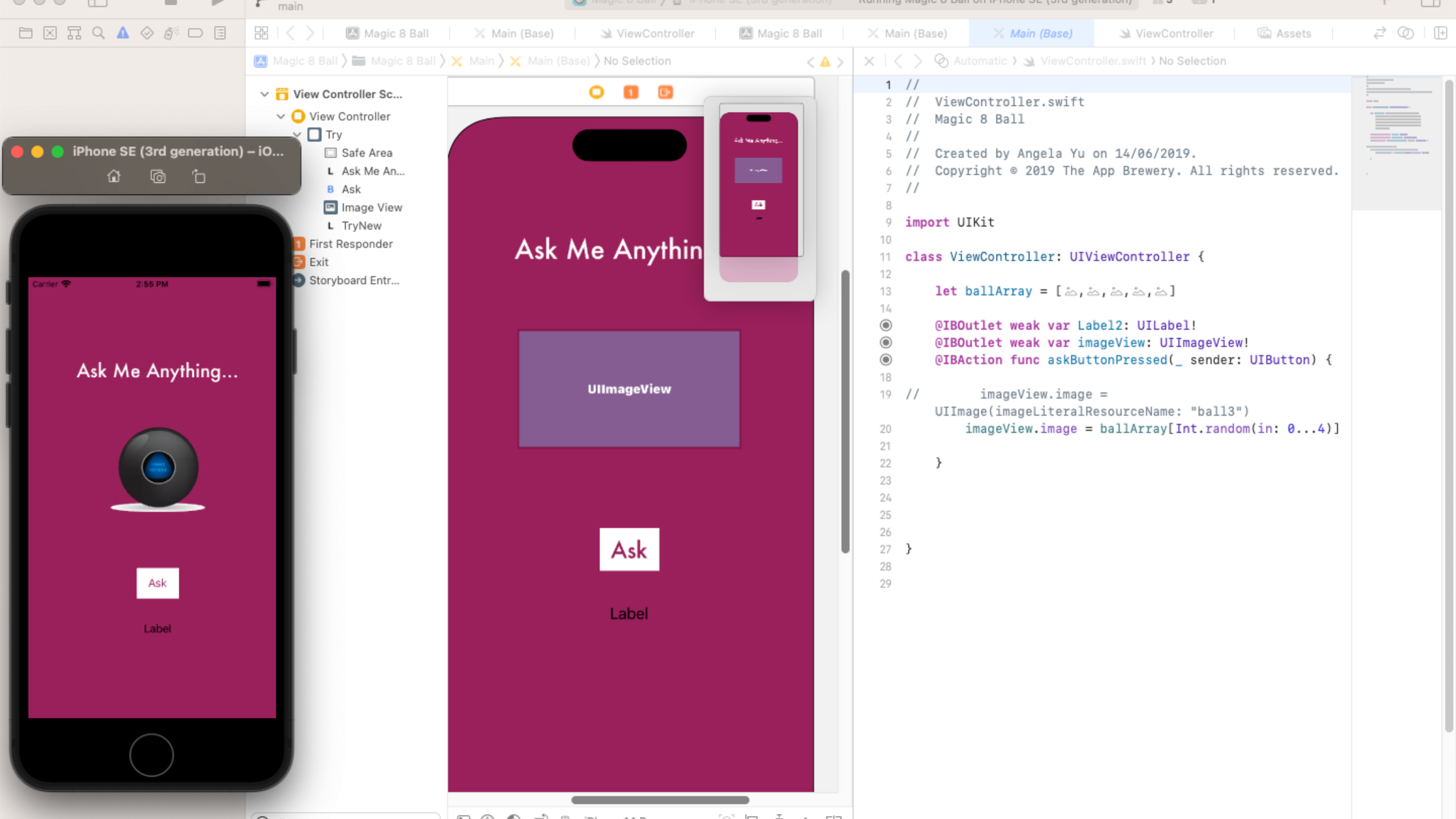

I am Rich
I am Rich
AppDelegate
SceneDelegate
ViewController
Main
Assets
LaunchScreen
Info
ViewController
Main (Base)
Assets
I am Rich
I am Rich
Main
Main (Base)
No Selection
View Controller Sc...
View Controller
View
Safe Area
I Am Rich
diamond
Button
First Responder
Exit
Storyboard Entr...
I Am Rich
Button
Filter
iPhone 11
45%
Automatic
ViewController.swift
No Selection
1 //
2 // ViewController.swift
3 // I am Rich
4 //
5 // Created by intern on 1/31/23.
6 //
7
8 import UIKit
9
10 class ViewController: UIViewController {
11
12 override func viewDidLoad() {
13 super.viewDidLoad()
14 // Do any additional setup after
loading the view.
15 }
16
17
18 }
19
20

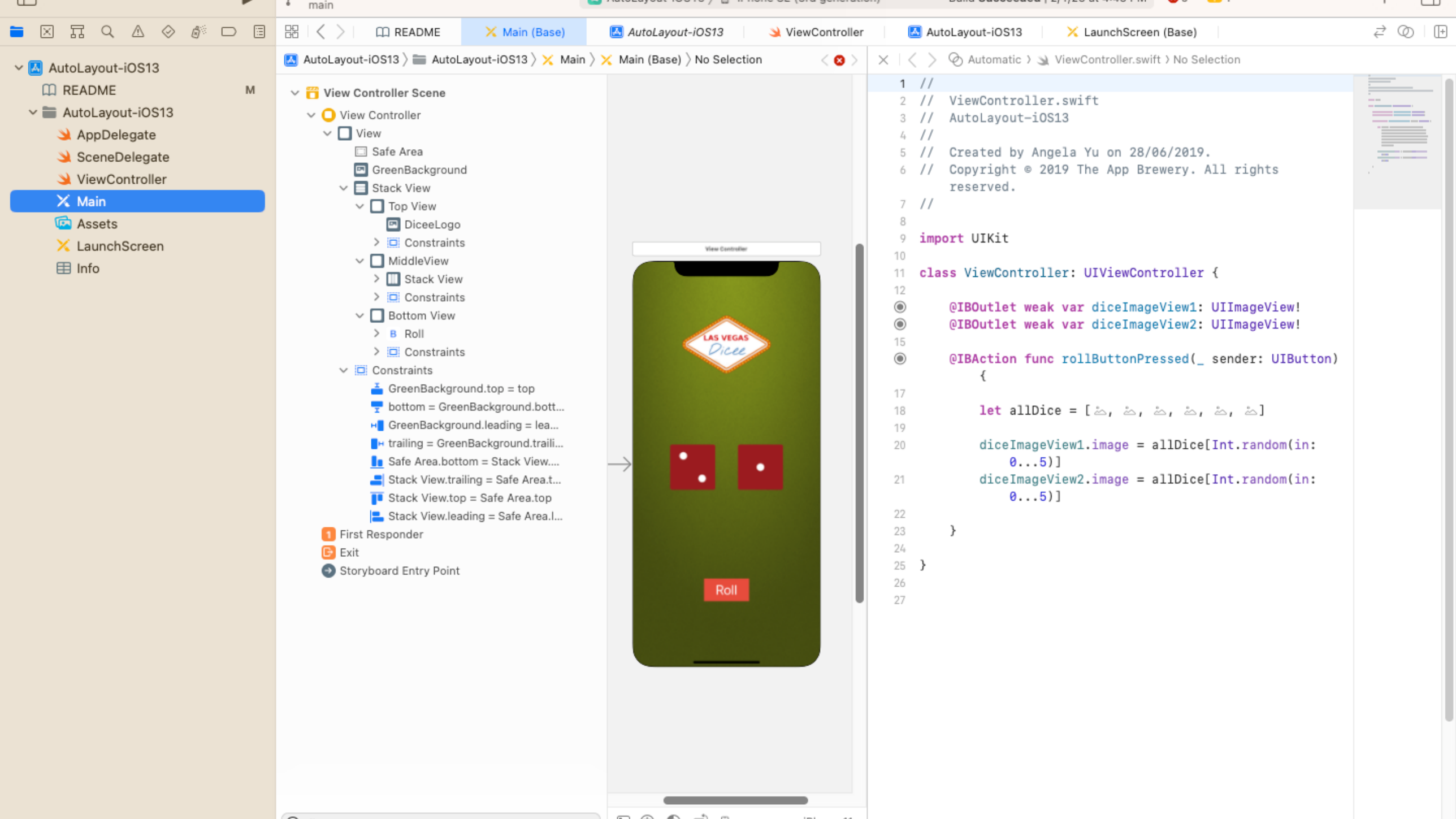

I Am Poor
I Am Poor
AppDelegate
SceneDelegate
ViewController
Main
Assets
LaunchScreen
Info
Main (Base)
I Am Poor
Assets
Main (Base)
I Am Poor > I Am Poor > Main > Main (Base) > No Selection
View Controller Sc...
View Controller
View
Safe Area
I Am Poor
coal
First Responder
Exit
Storyboard Entr...
View Controller
I Am Poor
Automatic > ViewController.swift > No Selection
1 //
2 // ViewController.swift
3 // I Am Poor
4 //
5 // Created by intern on 1/31/23.
6 //
7
8 import UIKit
9
10 class ViewController: UIViewController {
11
12 override func viewDidLoad() {
13 super.viewDidLoad()
14 // Do any additional setup after loading the
view.
15 }
16
17
18 }
19
20

Concepts discussed in iOS and Swift for the past week:

- Dicee App
  - Cloning an existing Xcode project from Github
  - Design user interface
  - Changing UI elements
  - Detect user interaction and respond
  - Use Swift variables, arrays, and randomization
- Magic 8 Ball App
- Auto Layout
  - Size classes and orientation
  - Constraints
  - Alignment and Pinning
  - Containers
  - StackViews

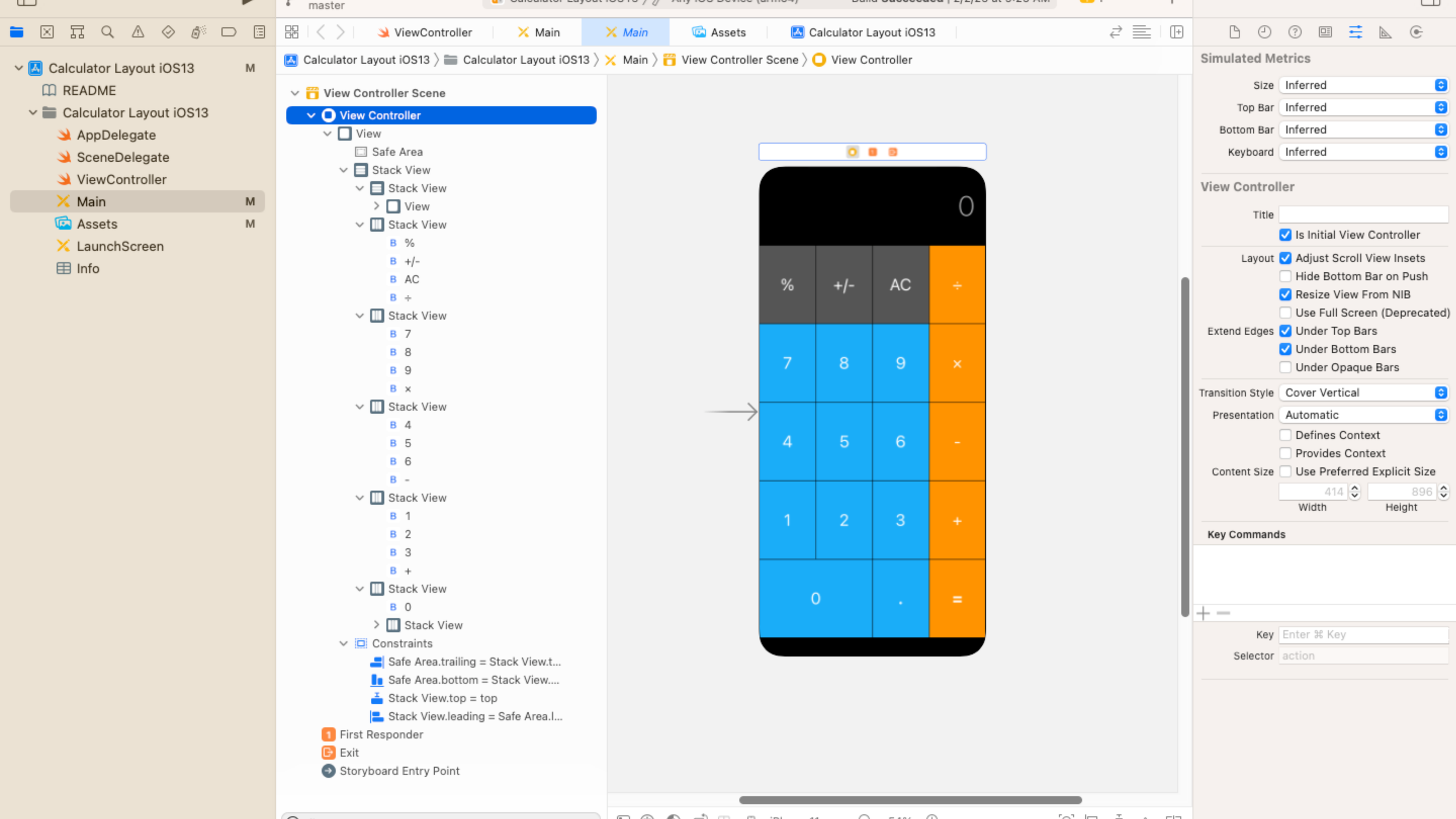

Dicee-iOS13
README
Dicee-iOS13
AppDelegate
SceneDelegate
ViewController
Main
Assets
LaunchScreen
Info
README
LaunchScreen (Base)
Assets
ViewController
Main (Base)
ViewController
Dicee-iOS13 › Dicee-iOS13 › Main › Main (Base) › No Selection
View Controller Sc...
View Controller
View
Safe Area
GreenBac...
DiceeLogo
Dice Imag...
Dice Imag...
Roll
First Responder
Exit
Storyboard Entr...
View Controller
LAS VEGAS
Dicee
Roll
Automatic › ViewController.swift › ViewController
//
import UIKit
class ViewController: UIViewController {
//this allows us to reference UI elements
@IBOutlet weak var diceImageView2: UIImageView!
@IBOutlet weak var diceImageView1: UIImageView!
@IBAction func rollButtonPressed(_ sender: UIButton)
{
/* print("Button got tapped.")
diceImageView1.image =
UIImage(imageLiteralResourceName: "DiceFour")
diceImageView2.image =
UIImage(imageLiteralResourceName: "DiceFour")
*/
let diceArray = [
UIImage(imageLiteralResourceName:
"DiceOne"),UIImage(imageLiteralResourceName:
"DiceTwo"),UIImage(imageLiteralResourceName:
"DiceThree"),UIImage
(imageLiteralResourceName:
"DiceFour"),UIImage
(imageLiteralResourceName:
"DiceFive"),UIImage
(imageLiteralResourceName:
"DiceSix"),UIImage(imageLiteralResourceName:
"DiceOne")]
diceImageView1.image = diceArray[Int.random(in:
0...5)]
diceImageView2.image = diceArray[Int.random(in:
0...5)]
}
/* override func viewDidLoad() {
super.viewDidLoad()
//this changes the image, who, what, value
formats

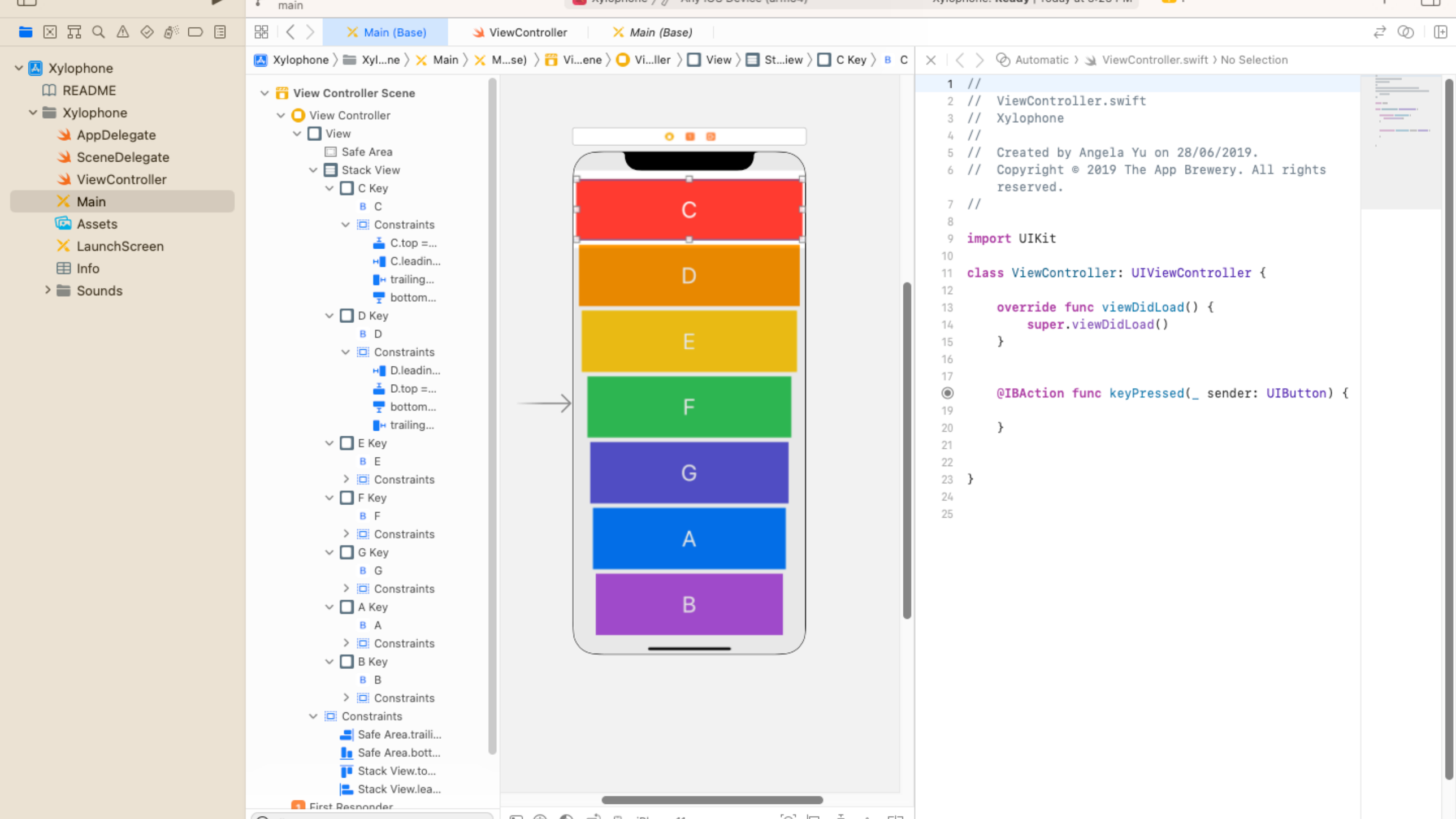

iPhone SE (3rd generation) – iO...
Carrier
2:55 PM
Ask Me Anything...
Ask
Label
View Controller Sc...
View Controller
Try
Safe Area
Ask Me An...
Ask
Image View
TryNew
First Responder
Exit
Storyboard Entr...
Ask Me Anythin
UIImageView
Ask
Label
//
// ViewController.swift
// Magic 8 Ball
//
// Created by Angela Yu on 14/06/2019.
// Copyright © 2019 The App Brewery. All rights reserved.
//
import UIKit
class ViewController: UIViewController {
let ballArray = [,,,,]
@IBOutlet weak var Label2: UILabel!
@IBOutlet weak var imageView: UIImageView!
@IBAction func askButtonPressed(_ sender: UIButton) {
// imageView.image =
UIImage(imageLiteralResourceName: "ball3")
imageView.image = ballArray[Int.random(in: 0...4)]
}
}


AutoLayout-iOS13
README
AutoLayout-iOS13
AppDelegate
SceneDelegate
ViewController
Main
Assets
LaunchScreen
Info
README
Main (Base)
AutoLayout-iOS13
ViewController
AutoLayout-iOS13
LaunchScreen (Base)
AutoLayout-iOS13 〉 AutoLayout-iOS13 〉 Main 〉 Main (Base) 〉 No Selection
View Controller Scene
View Controller
View
Safe Area
GreenBackground
Stack View
Top View
DiceeLogo
Constraints
MiddleView
Stack View
Constraints
Bottom View
Roll
Constraints
Constraints
GreenBackground.top = top
bottom = GreenBackground.bott...
GreenBackground.leading = lea...
trailing = GreenBackground.traili...
Safe Area.bottom = Stack View....
Stack View.trailing = Safe Area.t...
Stack View.top = Safe Area.top
Stack View.leading = Safe Area.l...
First Responder
Exit
Storyboard Entry Point
LAS VEGAS
Dicee
Roll
Automatic 〉 ViewController.swift 〉 No Selection
1 //
2 // ViewController.swift
3 // AutoLayout-iOS13
4 //
5 // Created by Angela Yu on 28/06/2019.
6 // Copyright © 2019 The App Brewery. All rights reserved.
7 //
8
9 import UIKit
10
11 class ViewController: UIViewController {
12
@IBOutlet weak var diceImageView1: UIImageView!
@IBOutlet weak var diceImageView2: UIImageView!
15
@IBAction func rollButtonPressed(_ sender: UIButton) {
17
18 let allDice = [⚀, ⚁, ⚂, ⚃, ⚄, ⚅]
19
20 diceImageView1.image = allDice[Int.random(in: 0...5)]
21 diceImageView2.image = allDice[Int.random(in: 0...5)]
22
23 }
24
25 }
26
27

Concepts discussed in iOS and Swift for the past week:

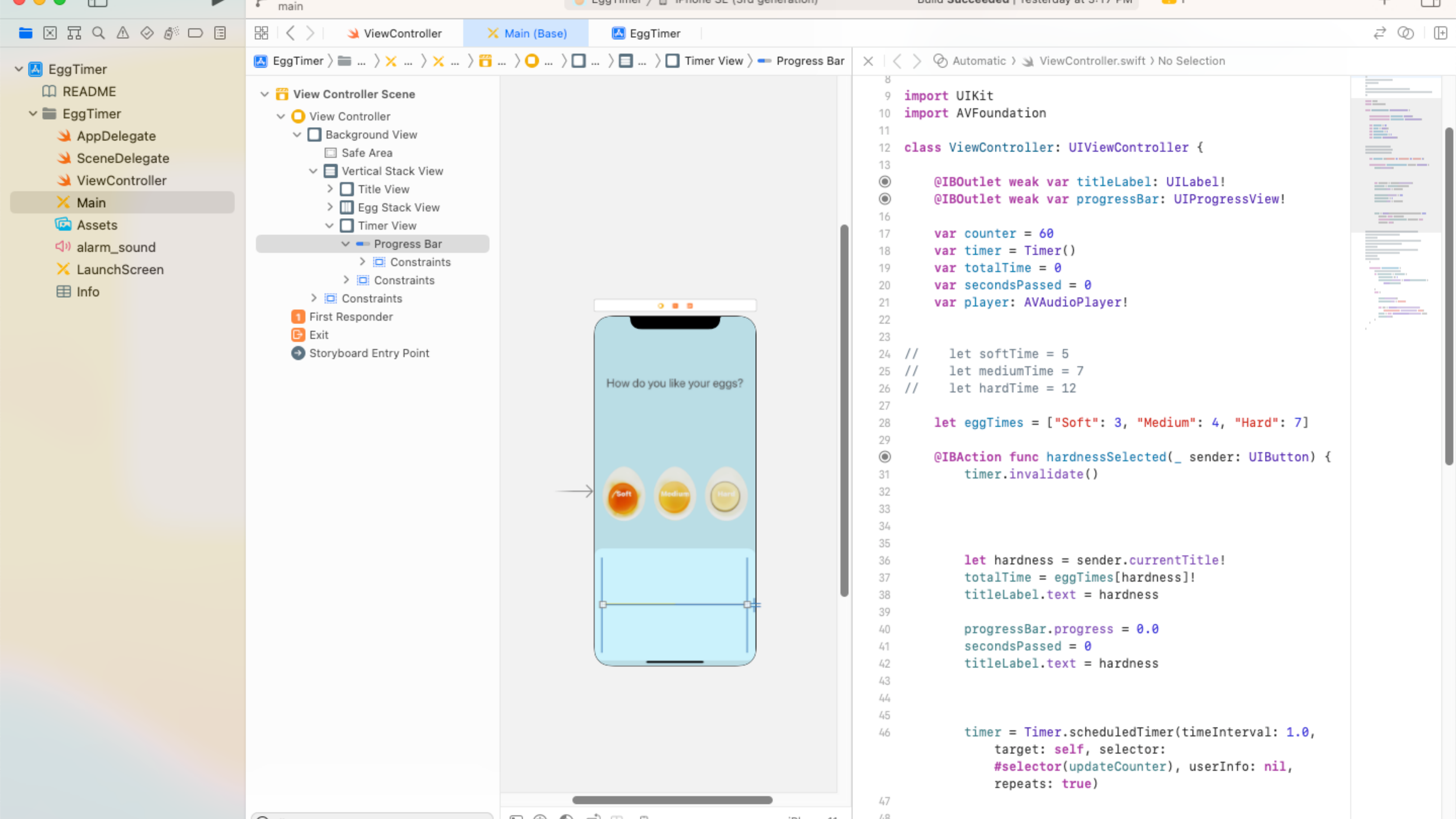
- Auto Layout Calculator App
  - Horizontal and Vertical stacks
- Xylophone App
  - Play sound using Apple Documentation and StackOverflow
  - Use the 5 Step Approach in Solving Programming Problem (Google, StackOverflow, Implement, Docs, Customize)
  - Swift Functions
  - Linking multiple elements to one IBAction
  - Functions with inputs and Type Inference
  - Play different sounds for different buttons


Calculator Layout iOS13
README
Calculator Layout iOS13
AppDelegate
SceneDelegate
ViewController
Main
Assets
LaunchScreen
Info
ViewController
Main
Assets
Calculator Layout iOS13
Calculator Layout iOS13 〉 Calculator Layout iOS13 〉 Main 〉 View Controller Scene 〉 View Controller
View Controller Scene
View Controller
View
Safe Area
Stack View
Stack View
View
Stack View
%
+/-
AC
÷
Stack View
7
8
9
×
Stack View
4
5
6
-
Stack View
1
2
3
+
Stack View
0
Stack View
Constraints
Safe Area.trailing = Stack View.t...
Safe Area.bottom = Stack View....
Stack View.top = top
Stack View.leading = Safe Area.l...
First Responder
Exit
Storyboard Entry Point
0
%
+/-
AC
÷
7
8
9
×
4
5
6
-
1
2
3
+
0
.
=
Simulated Metrics
Size
Inferred
Top Bar
Inferred
Bottom Bar
Inferred
Keyboard
Inferred
View Controller
Title
Is Initial View Controller
Layout
Adjust Scroll View Insets
Hide Bottom Bar on Push
Resize View From NIB
Use Full Screen (Deprecated)
Extend Edges
Under Top Bars
Under Bottom Bars
Under Opaque Bars
Transition Style
Cover Vertical
Presentation
Automatic
Defines Context
Provides Context
Content Size
Use Preferred Explicit Size
414
Width
896
Height
Key Commands
Key
Enter ⌘ Key
Selector
action


Xylophone
README
Xylophone
AppDelegate
SceneDelegate
ViewController
Main
Assets
LaunchScreen
Info
Sounds
Main (Base)
ViewController
Main (Base)
View Controller Scene
View Controller
View
Safe Area
Stack View
C Key
C
Constraints
C.top =...
C.leadin...
trailing...
bottom...
D Key
D
Constraints
D.leadin...
D.top =...
bottom...
trailing...
E Key
E
Constraints
F Key
F
Constraints
G Key
G
Constraints
A Key
A
Constraints
B Key
B
Constraints
Constraints
Safe Area.traili...
Safe Area.bott...
Stack View.to...
Stack View.lea...
First Responder
C
D
E
F
G
A
B
Automatic
ViewController.swift
No Selection
//
// ViewController.swift
// Xylophone
//
// Created by Angela Yu on 28/06/2019.
// Copyright © 2019 The App Brewery. All rights reserved.
//
import UIKit
class ViewController: UIViewController {
override func viewDidLoad() {
super.viewDidLoad()
}
@IBAction func keyPressed(_ sender: UIButton) {
}
}

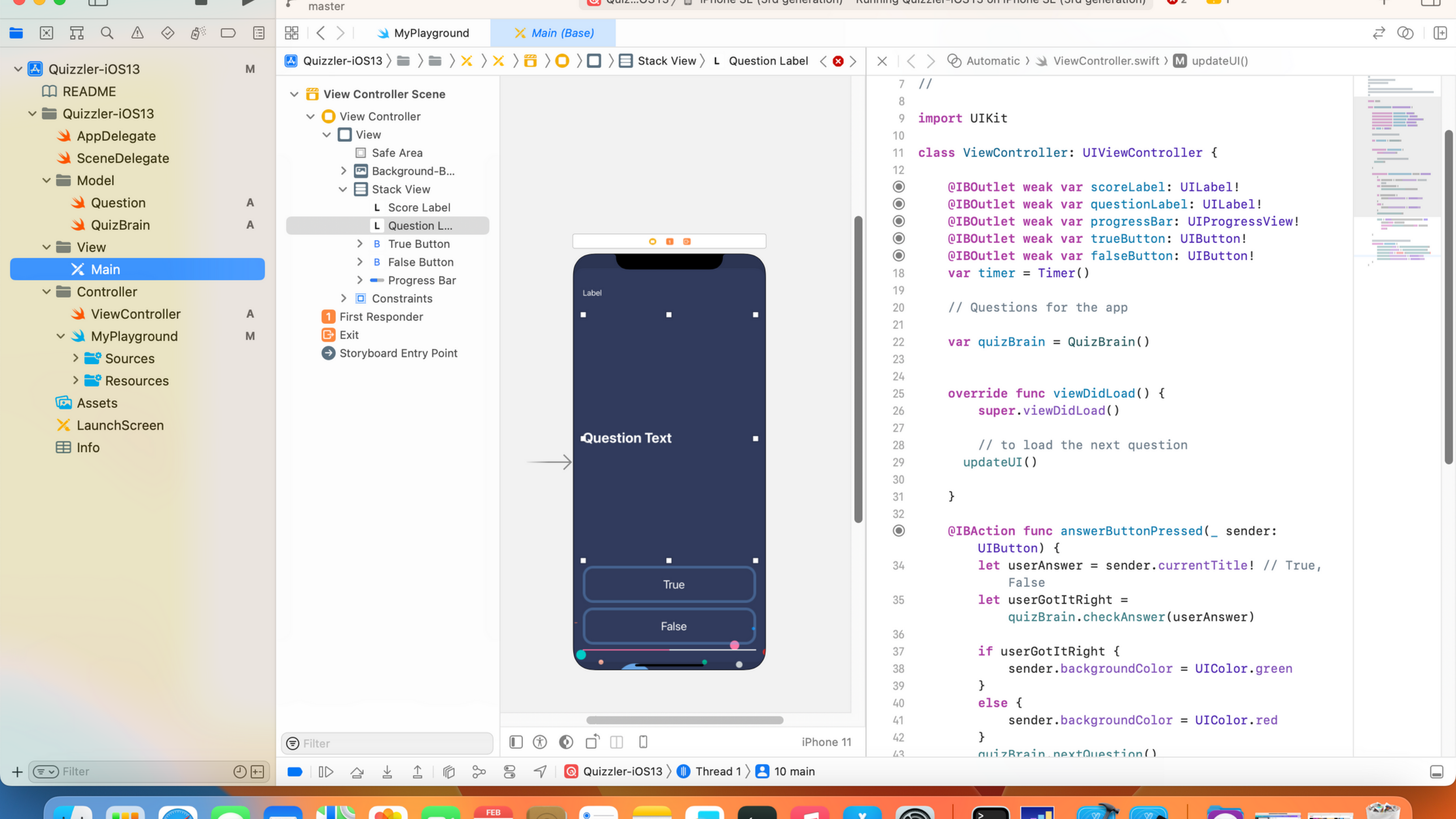
Concepts discussed in iOS and Swift for the past week:

- Egg Timer App
  - If/Else statements
  - Switch statements, Range Operators (a...b, a..<b,...b)
  - Dictionaries
  - Optionals
  - UIProgressView
  - Debugging App ("What did you expect your code to do?", "What happened instead?", "What does your expectation depend on?", "How can we test the things our expectations depend on?", "Fix our code to make reality match expectations.")


EggTimer
README
EggTimer
AppDelegate
SceneDelegate
ViewController
Main
Assets
alarm_sound
LaunchScreen
Info
ViewController
Main (Base)
EggTimer
Timer View
Progress Bar
View Controller Scene
View Controller
Background View
Safe Area
Vertical Stack View
Title View
Egg Stack View
Timer View
Progress Bar
Constraints
Constraints
Constraints
First Responder
Exit
Storyboard Entry Point
How do you like your eggs?
Soft
Medium
Hard
Automatic
ViewController.swift
No Selection
import UIKit
import AVFoundation
class ViewController: UIViewController {
@IBOutlet weak var titleLabel: UILabel!
@IBOutlet weak var progressBar: UIProgressView!
var counter = 60
var timer = Timer()
var totalTime = 0
var secondsPassed = 0
var player: AVAudioPlayer!
// let softTime = 5
// let mediumTime = 7
// let hardTime = 12
let eggTimes = ["Soft": 3, "Medium": 4, "Hard": 7]
@IBAction func hardnessSelected(_ sender: UIButton) {
timer.invalidate()
let hardness = sender.currentTitle!
totalTime = eggTimes[hardness]!
titleLabel.text = hardness
progressBar.progress = 0.0
secondsPassed = 0
titleLabel.text = hardness
timer = Timer.scheduledTimer(timeInterval: 1.0,
target: self, selector:
#selector(updateCounter), userInfo: nil,
repeats: true)

Concepts discussed in iOS and Swift for the past week:

- Quizzler App
  - Swift Structures
  - Design patterns, and use the Model View Controller pattern
  - Swift Functions with outputs
  - Immutability




```swift
//

import UIKit

class ViewController: UIViewController {

    @IBOutlet weak var scoreLabel: UILabel!
    @IBOutlet weak var questionLabel: UILabel!
    @IBOutlet weak var progressBar: UIProgressView!
    @IBOutlet weak var trueButton: UIButton!
    @IBOutlet weak var falseButton: UIButton!
    var timer = Timer()

    // Questions for the app

    var quizBrain = QuizBrain()


    override func viewDidLoad() {
        super.viewDidLoad()

        // to load the next question
      updateUI()

    }

    @IBAction func answerButtonPressed(_ sender:
        UIButton) {
        let userAnswer = sender.currentTitle! // True,
            False
        let userGotItRight =
            quizBrain.checkAnswer(userAnswer)

        if userGotItRight {
            sender.backgroundColor = UIColor.green
        }
        else {
            sender.backgroundColor = UIColor.red
        }
        quizBrain.nextQuestion()
```

Classes and Objects

```swift
// Class
class Dog {

    let dogBreed : String
    let dogName : String

    init(dogBreed : String, dogName : String){
        self.dogBreed = dogBreed
        self.dogName = dogName
    }

}

// Object

let dog1 = Dog (dogBreed : "Aspin", dogName : "Browny")
print("This is my dog \(dog1.dogName), my dog is a \(dog1.dogBreed)")
```

Properties and Methods

```
27  // Properties and Methods
28  class Animal {
29      var name: String
30
31      init (name: String) {
32          self.name = name
33      }
34
35      func eat() {
36          print("I can eat.")
37      }
38  }
39
40  class Cat : Animal {
41      func displayName () {
42          print("I am \(name)")
43      }
44  }
45
46  var object1 = Cat(name: "Bryan the Cat")
47  object1.displayName()
48  object1.eat()
```

Pass by Reference

```
62 // We declare a class, Employee, and create an instance, employee1/ We set the name
       property of employee1 to Tom.
63 // We then declare another variable, employee2, and assign employee1 to it. We set the
       name property of employee2 to Fred and print the names of both Employee instances.
64 // Because instances of classes are passed by reference, the valie of the name
       property of both instances equal to Fred.
65 class Employee{
66     var name: String
67 
68     init (name: String) {
69         self.name = name
70     }
71 }
72 
73 // object
74 var employee1 = Employee(name: "Tom")
75 print(employee1.name)
76 
77 // another object
78 var employee2 = employee1
79 employee2.name = "Fred"
80 
81 print(employee1.name)
82 print(employee2.name)
```

Pass by Value

```
98  // The moment movie1 is assigned to the movie2 variable, a copy of movie1 is made and
        assigned to movie2.
99  // The values of movie1 and movie2 have no relation to one another apart from the fact
        that they are copies
100 struct Movie{
101     var movieName : String
102     init(movieName : String)
103     {
104         self.movieName = movieName
105     }
106 }
107
108 var movie1 = Movie(movieName : "Avengers")
109
110 var movie2 = movie1
111 movie2.movieName = "Justice League"
112
113 print(movie1.movieName)
114 print(movie2.movieName)
115 print(movie1.movieName)
116 print(movie2.movieName)
```

Access Controls (Public)

```
130 class Food{
131     public var foodType : String //= "ChickenJoy"
132 
133     init(foodType : String)
134     {
135         self.foodType = foodType
136     }
137     public func display()
138     {
139         print("I am  a \(foodType) \n Please eat me ")
140     }
141 }
142 
143 //created an object
144 // access and assign value to public property
145 var burger = Food(foodType : "Cheese Burger")
146 
147 // access the public method
148 burger.display()
```

Access Controls (Private)

```
155 class Student{
156     private var name : String = ""
157 
158     //initializer
159     private func display(){
160         print("Hello I'm a student")
161     }
162 
163     public func getName() -> String{
164         display()
165         return name
166     }
167 
168     public func setName(pName: String)
169     {
170         self.name = pName
171     }
172 }
173 
174 var student1 = Student()
175 student1.setName(pName : "Kath")
176 print(student1.getName())
```

Method Overriding

```
184 class Vehicle {
185
186     // method in parent class
187     func displayInfo () {
188         print ("Four Wheeler")
189     }
190 }
191
192 // motorcycle inherits vehicle
193 class Motorcycle: Vehicle {
194
195     // override displayInfo method
196     override func displayInfo () {
197         print ("Two Wheeler \n")
198     }
199 }
200
201 // create an object
202 var vehicle1 = Motorcycle ()
203 vehicle1.displayInfo()
```

Method Overloading

```
212 // function with Int parameter
213 func displayValues (n: Int) {
214     print ("This is an integer: \(n)")
215 }
216
217 // function with String parameter
218 func displayValues (n: String) {
219     print ("This is a string: \(n)")
220 }
221
222 // same function name, but different argument
223 displayValues(n:2)
224 displayValues(n:"Hello World")
```

Thank You!