GENERAL ASSEMBLY

IOS DEVELOPMENT WORKSHOP

Slides and Code Samples on Github

http://bit.ly/2e75RU2

Jeffrey Bergier
UX Designer, Riverbed Technology

LEARNING OBJECTIVES

- Explain relevant history and trends in iOS development.
- Identify key skills leveraged by iOS developers.
- Survey the common tools used within the iOS ecosystem.
- Apply key concepts and skills to build your own basic iOS application.
- Create a custom learning plan to help you continue to build fundamental iOS development skills after this workshop.

IOS DEVELOPMENT

PRE-WORK

PRE-WORK REVIEW

- Bring a Mac laptop with Xcode installed. Macs are required to create apps for the iOS ecosystem.
- Please note: you may need to update your OS in order to install the latest version of Xcode.

OPENING

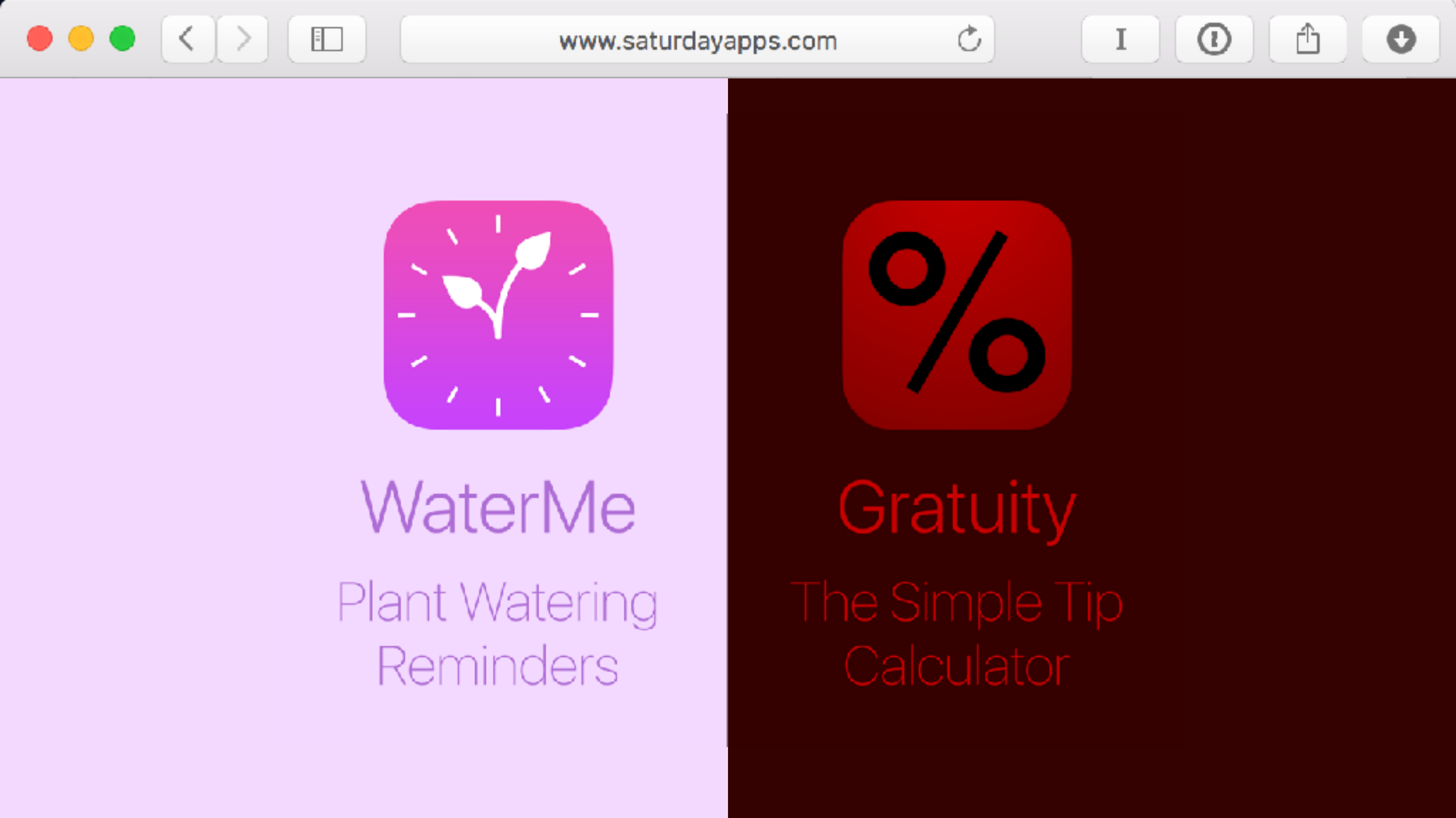
GENERAL ASSEMBLY

JEFFREY BERGIER

UX Designer @ Riverbed
TA @ General Assembly

Teacher @ MobileBridge
Addicted iOS Dev @ Home

🕊 @jeffburg
🕸 jeffburg.com



www.saturdayapps.com

WaterMe

Plant Watering Reminders

Gratuity

The Simple Tip Calculator

ABOUT YOU

- Before we dive in, let’s talk a bit about you!

- Name:
- What brings you to GA?
  - Current activities:
  - Goals:
- Fun fact?

AGENDA

- Mobile Intro
- Learn basics of Xcode IDE
- stretch break
- Learn programming basics with Swift
- stretch break
- Make a basic iOS application
- stretch break
- Dive a little deeper into Swift
- Resources

MOBILE INTRO

MOBILE INTRO

- 2 Primary Avenues
    - Web
    - iOS App
- Always default to web
    - Can still have dedicated app icon on home screen
    - Supports offline use
    - No “Disney” filter app review
    - Instant updates
    - No installation necessary
    - Potentially cross-platform

MOBILE INTRO

- Why Go Native?
    - Performance
    - Device specific capabilities
        - Sensors, Camera, Location, Backgrounding
    - 3D / OpenGL / Metal
    - Notifications
- Note that many of the above items are now do-able on web
    - Camera, Pictures
    - Location
    - Notifications (Desktop Safari only right now)

THAT BEING SAID, I LOVE NATIVE!

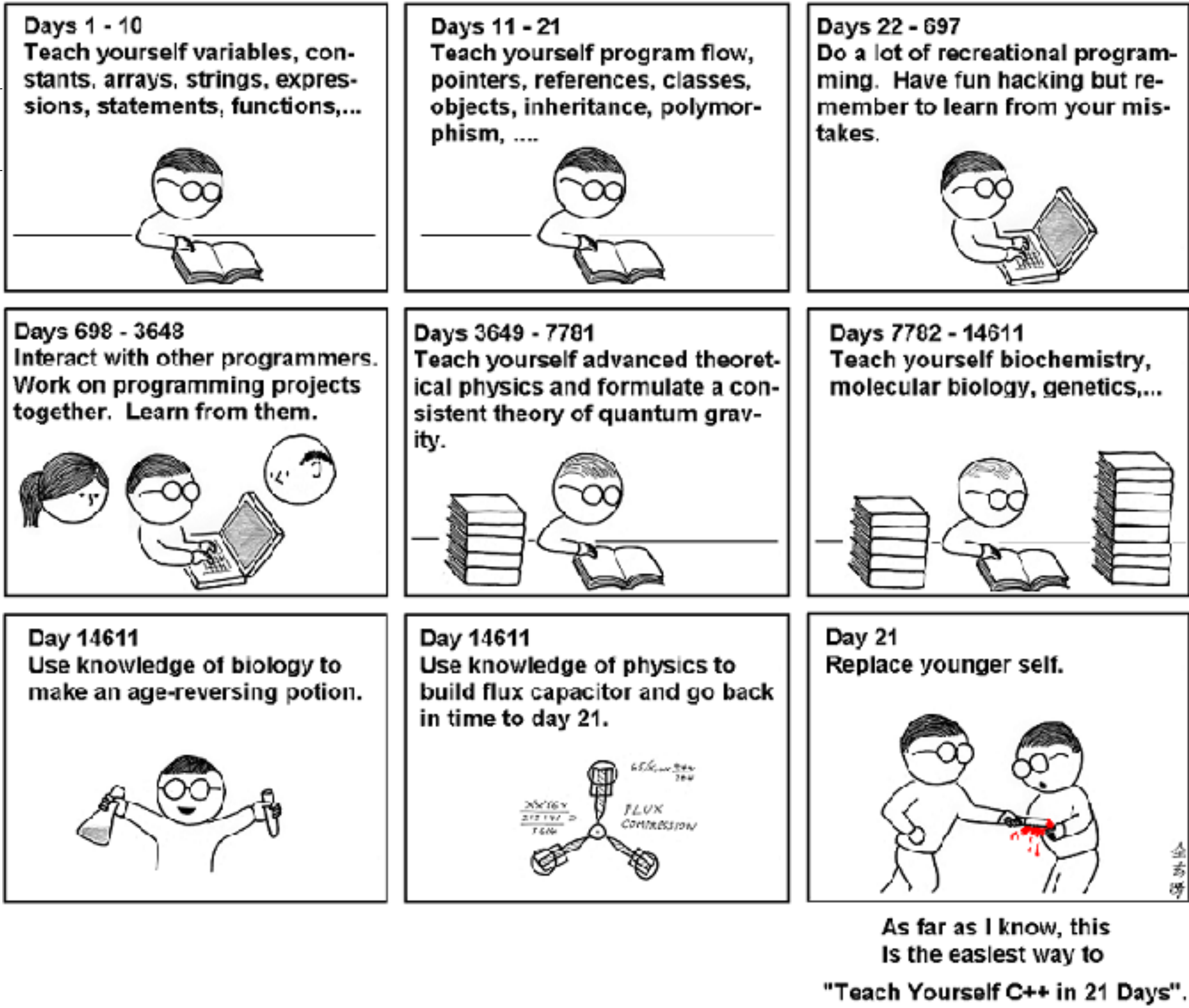
- I like learning 1 language and being able to do everything
    - As opposed to HTML/CSS/Javascript as 3 languages
- I like that the developer ecosystem is contained
    - Apple maintains huge influence over how things “should” work
    - The web is a wild west of frameworks and approaches
- I can’t stand CSS. I find Auto Layout much easier

MOBILE INTRO

- Just remember to think critically about your project and whether the user experience will be better on web vs native
  - Is performance stretched on Web?
  - Is this something a user will only use 1 time and be hesitant to install permanently on their device?
  - Does this use unique features of native?
- e.g Amazon Shopping (great on web, terrible native)
- e.g. Instagram (great on native, questionable on web)

SET EXPECTATIONS

- Learning iOS
  - in 21 days


Days 1 - 10
Teach yourself variables, constants, arrays, strings, expressions, statements, functions,...
Days 11 - 21
Teach yourself program flow, pointers, references, classes, objects, inheritance, polymorphism, ...
Days 22 - 697
Do a lot of recreational programming. Have fun hacking but remember to learn from your mistakes.
Days 698 - 3648
Interact with other programmers. Work on programming projects together. Learn from them.
Days 3649 - 7781
Teach yourself advanced theoretical physics and formulate a consistent theory of quantum gravity.
Days 7782 - 14611
Teach yourself biochemistry, molecular biology, genetics,...
Day 14611
Use knowledge of biology to make an age-reversing potion.
Day 14611
Use knowledge of physics to build flux capacitor and go back in time to day 21.
FLUX COMPRESSION
Day 21
Replace younger self.
As far as I know, this is the easiest way to "Teach Yourself C++ in 21 Days".

MOBILE INTRO

- We are going to barely touch the surface
- Basics of Swift
- Basics of iOS
    - From here on out referred to as Cocoa or Cocoa Touch

- Leave you with resources so you can combine tonight’s lesson with online resources so you can continue learning.

2017/05/23 – START BUILDING MOBILE APPS

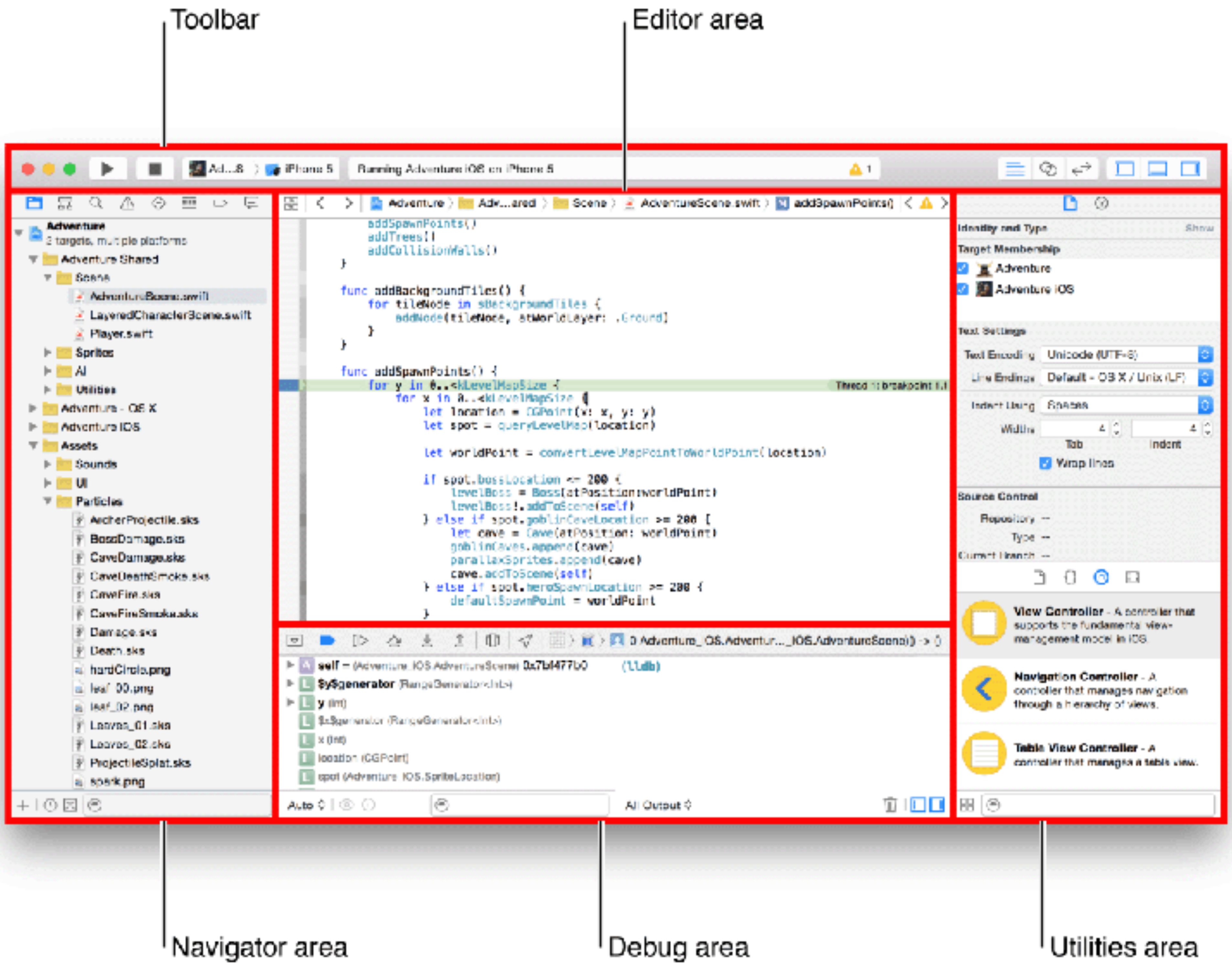
AGENDA

- ~~Mobile Intro~~
- Learn basics of Xcode IDE
- stretch break
- Learn programming basics with Swift
- stretch break
- Make a basic iOS application
- stretch break
- Dive a little deeper into Swift
- Resources

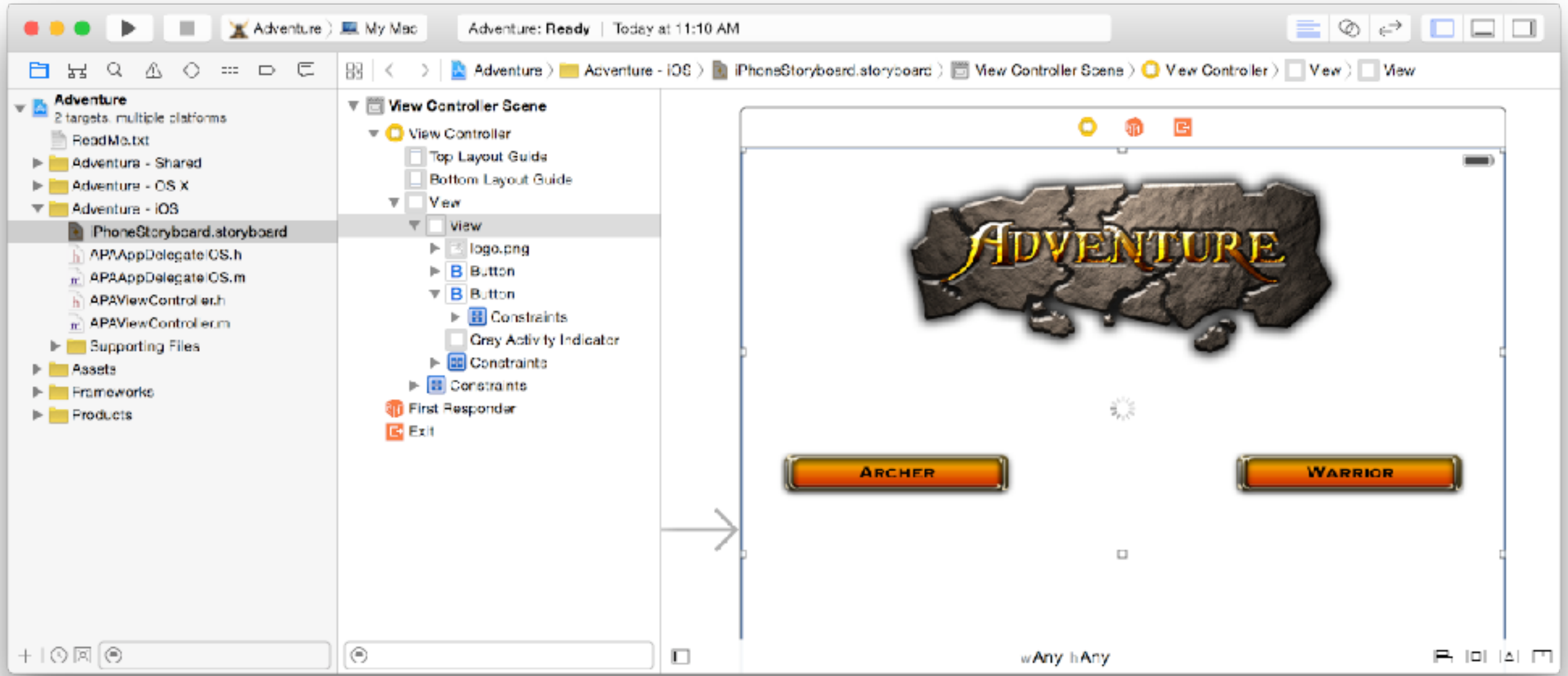
XCODE

XCODE

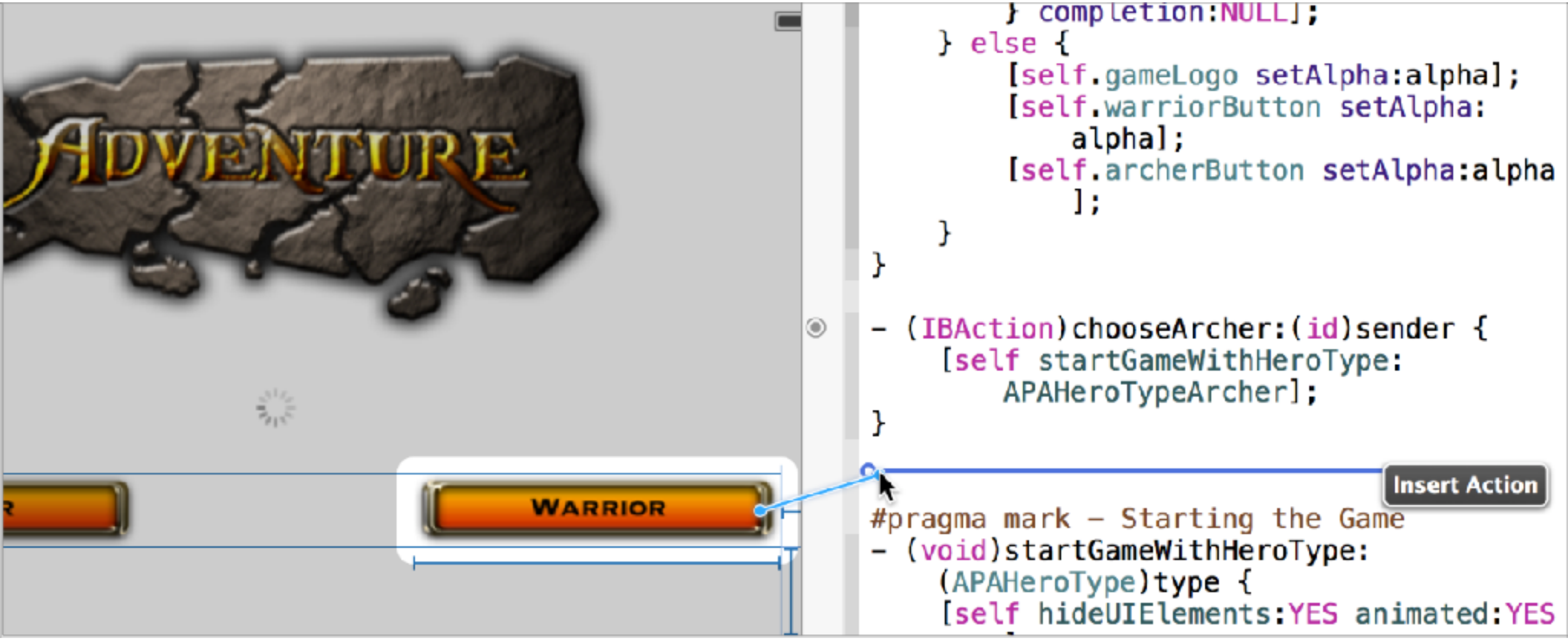
- Apple’s primary IDE for the iOS/watchOS/tvOS/macOS platforms
- Available Free from Mac App Store and http://developer.apple.com
- It does everything:
    - Code editor with auto complete and warnings for common mistakes
    - Interface Builder
    - Compiler
    - Debugger
    - Unit Testing
    - Submitting to App Store



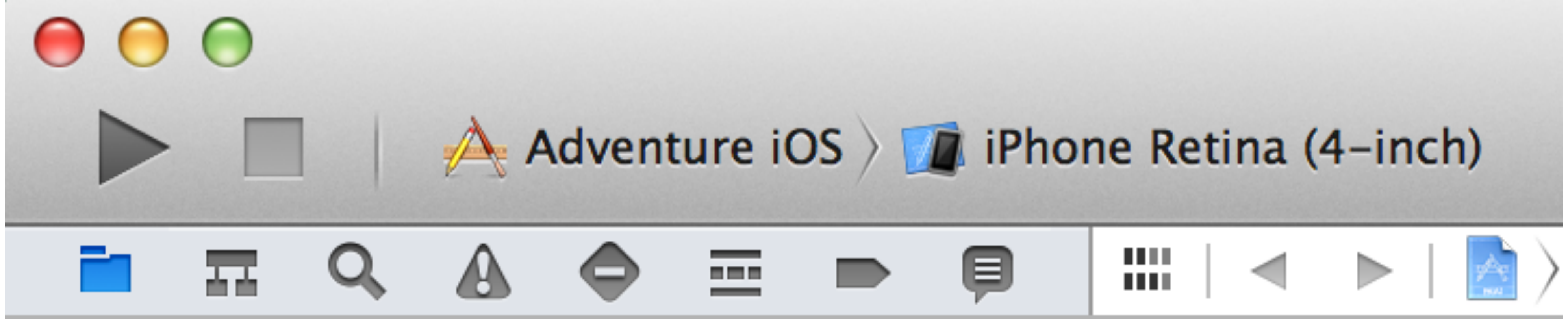
```
        addSpawnPoints()
        addTrees()
        addCollisionWalls()
    }

    func addBackgroundTiles() {
        for tileNode in sBackgroundTiles {
            addNode(tileNode, atWorldLayer: .Ground)
        }
    }

    func addSpawnPoints() {
        for y in 0..<kLevelMapSize {
            for x in 0..<kLevelMapSize {
                let location = CGPoint(x: x, y: y)
                let spot = queryLevelMap(location)

                let worldPoint = convertLevelMapPointToWorldPoint(location)

                if spot.bossLocation <= 200 {
                    levelBoss = Boss(atPosition:worldPoint)
                    levelBoss!.addToScene(self)
                } else if spot.goblinCaveLocation >= 200 {
                    let cave = Cave(atPosition: worldPoint)
                    goblinCaves.append(cave)
                    parallaxSprites.append(cave)
                    cave.addToScene(self)
                } else if spot.heroSpawnLocation >= 200 {
                    defaultSpawnPoint = worldPoint
                }
```

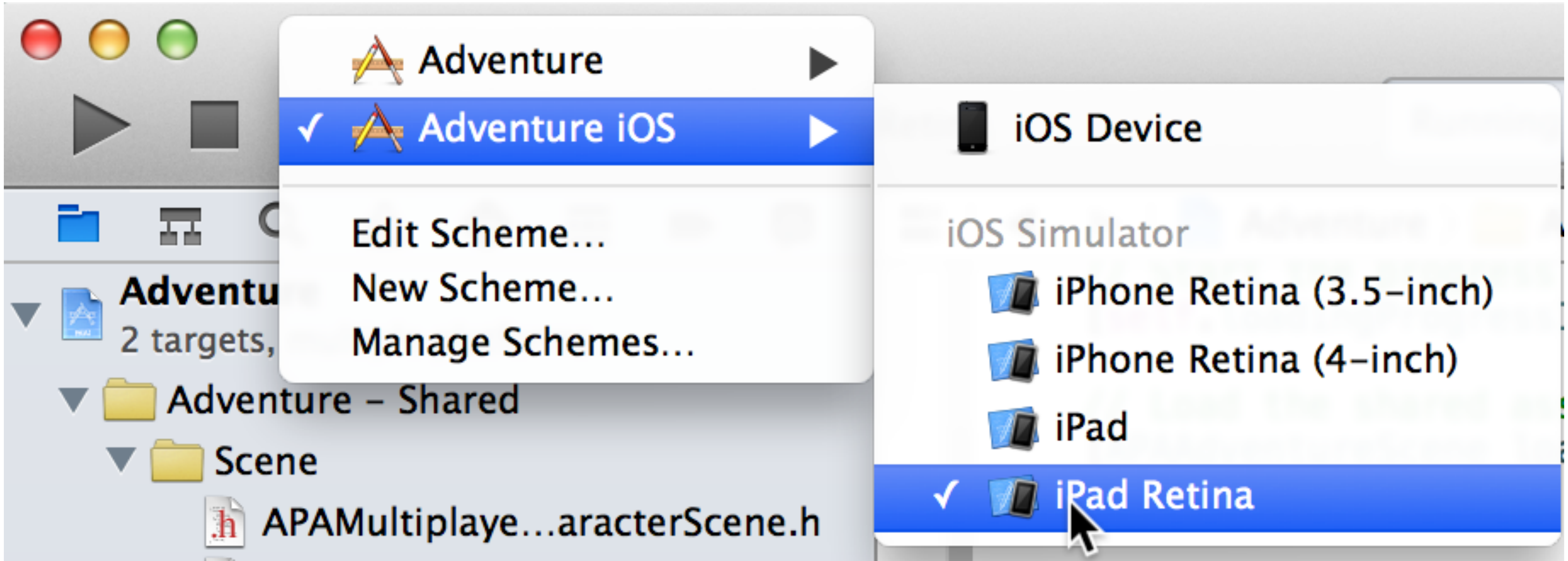


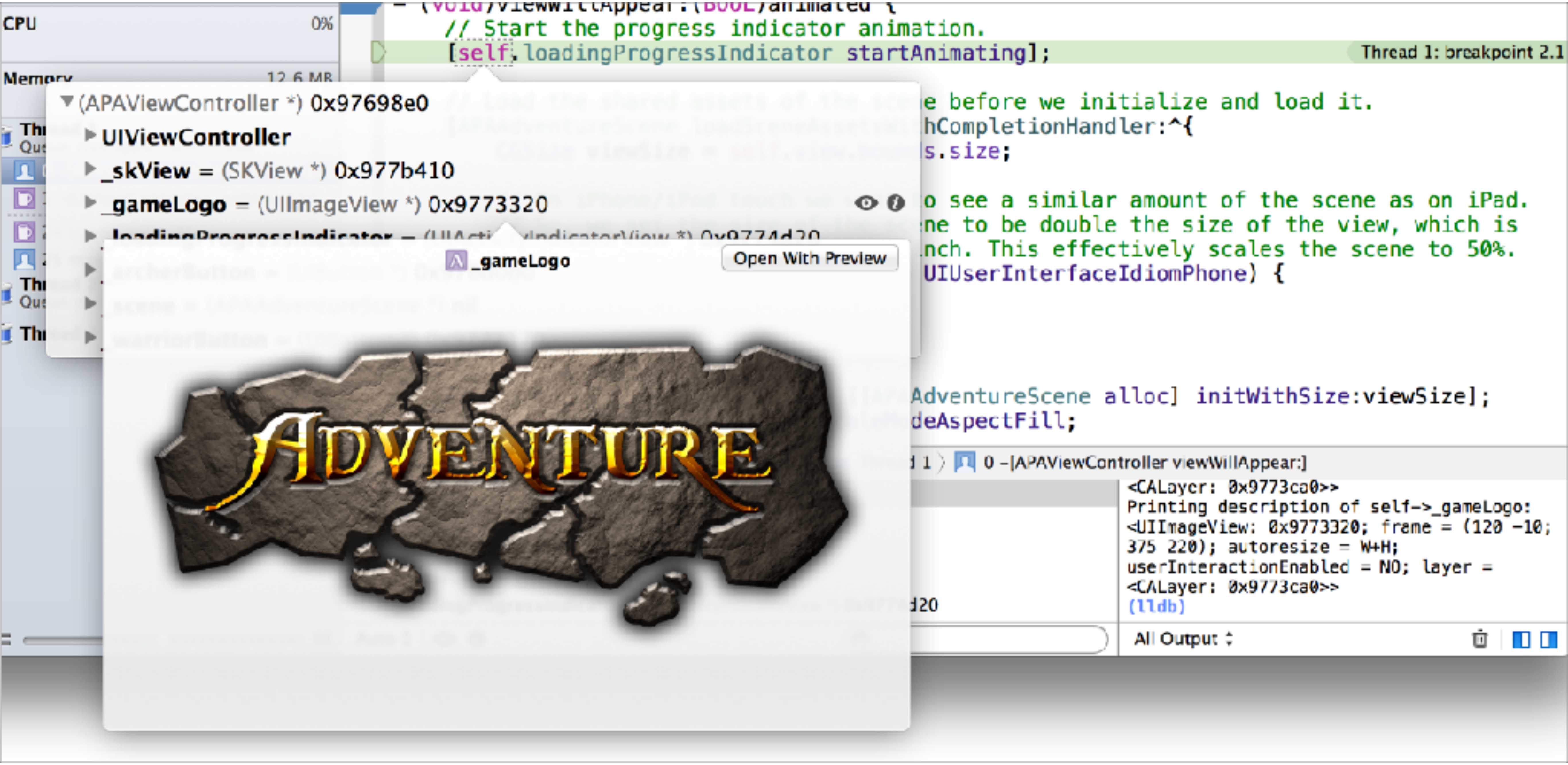
Adventure
My Mac
Adventure: Ready | Today at 11:10 AM
Adventure
Adventure - iOS
iPhoneStoryboard.storyboard
View Controller Scene
View Controller
View
View
Adventure
2 targets, multiple platforms
ReadMe.txt
Adventure - Shared
Adventure - OS X
Adventure - iOS
iPhoneStoryboard.storyboard
APAAppDelegateiOS.h
APAAppDelegateiOS.m
APAViewController.h
APAViewController.m
Supporting Files
Assets
Frameworks
Products
View Controller Scene
View Controller
Top Layout Guide
Bottom Layout Guide
View
View
logo.png
Button
Button
Constraints
Grey Activity Indicator
Constraints
Constraints
First Responder
Exit
ADVENTURE
ARCHER
WARRIOR
wAny hAny



```
            } completion:NULL];
    } else {
        [self.gameLogo setAlpha:alpha];
        [self.warriorButton setAlpha:
            alpha];
        [self.archerButton setAlpha:alpha
            ];
    }
}

- (IBAction)chooseArcher:(id)sender {
    [self startGameWithHeroType:
        APAHeroTypeArcher];
}

#pragma mark - Starting the Game
- (void)startGameWithHeroType:
    (APAHeroType)type {
    [self hideUIElements:YES animated:YES
```


Adventure iOS
iPhone Retina (4-inch)

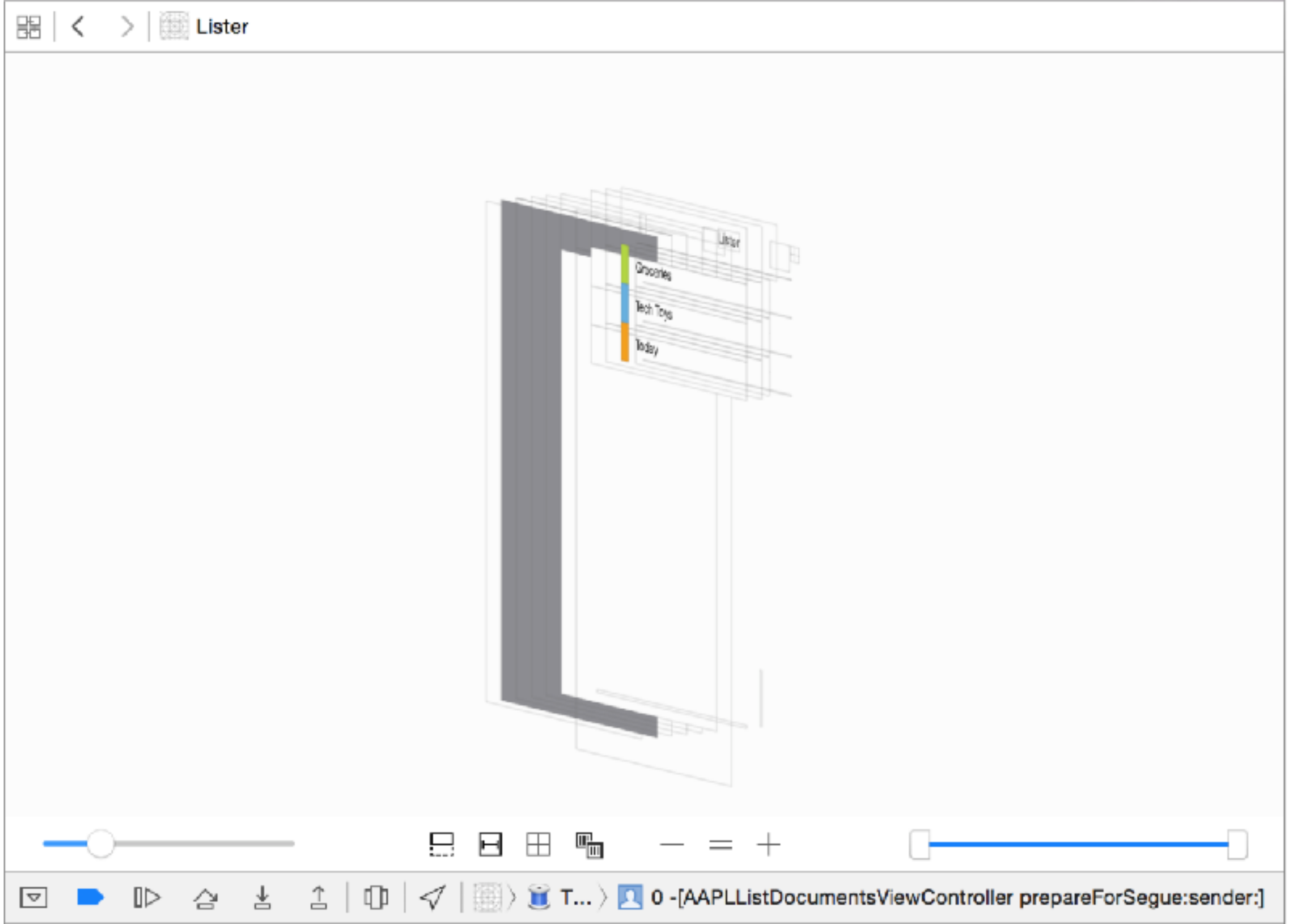

Adventure
✓ Adventure iOS
Edit Scheme…
New Scheme…
Manage Schemes…
iOS Device
iOS Simulator
iPhone Retina (3.5-inch)
iPhone Retina (4-inch)
iPad
✓ iPad Retina
Adventu
2 targets,
Adventure – Shared
Scene
APAMultiplaye…aracterScene.h

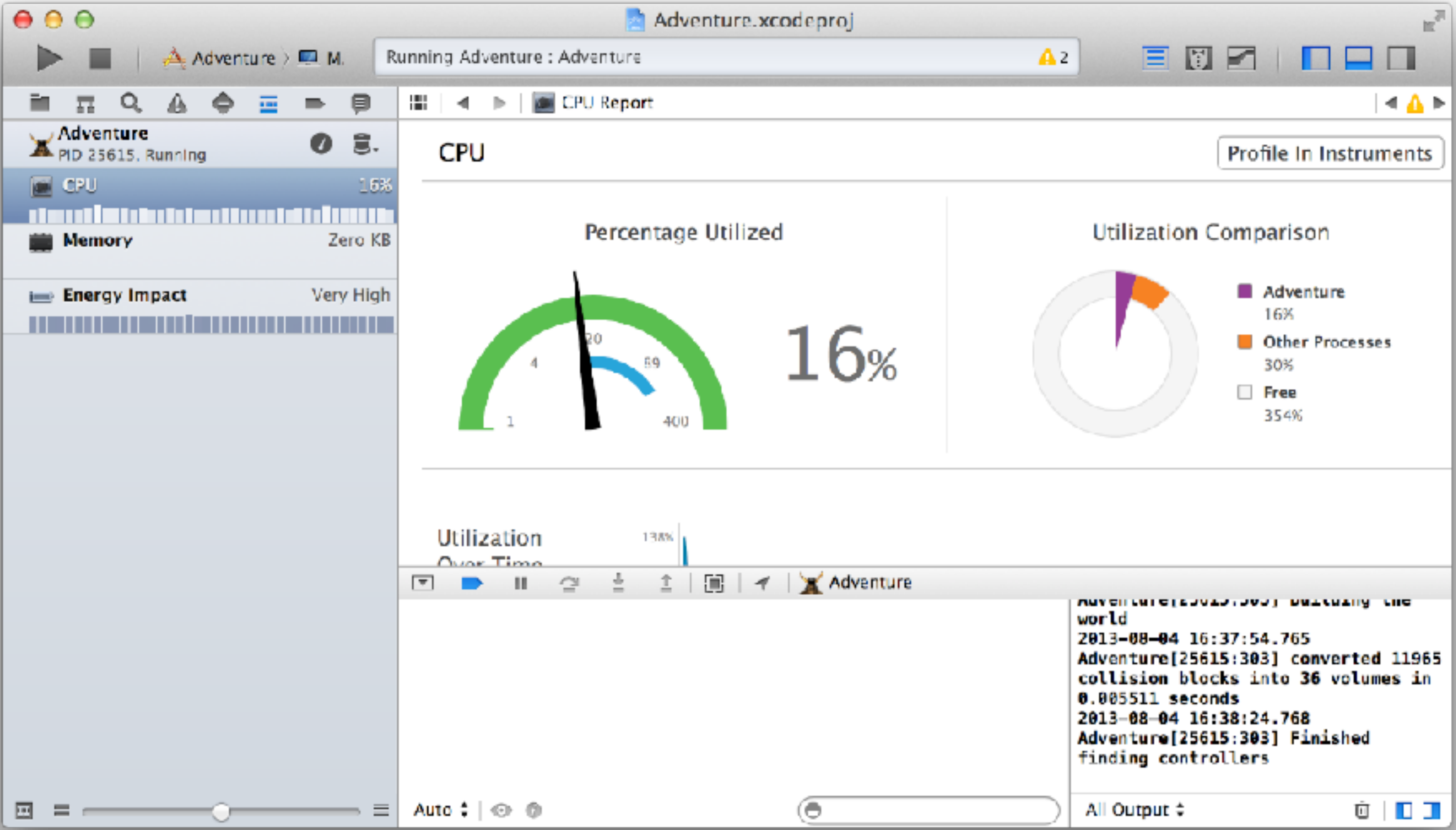


```
- (void)viewWillAppear:(BOOL)animated {
    // Start the progress indicator animation.
    [self.loadingProgressIndicator startAnimating];

                                   e before we initialize and load it.
                                   hCompletionHandler:^{
                                   s.size;

                                   o see a similar amount of the scene as on iPad.
                                   ne to be double the size of the view, which is
                                   nch. This effectively scales the scene to 50%.
                                   UIUserInterfaceIdiomPhone) {

                                   AdventureScene alloc] initWithSize:viewSize];
                                   deAspectFill;
```

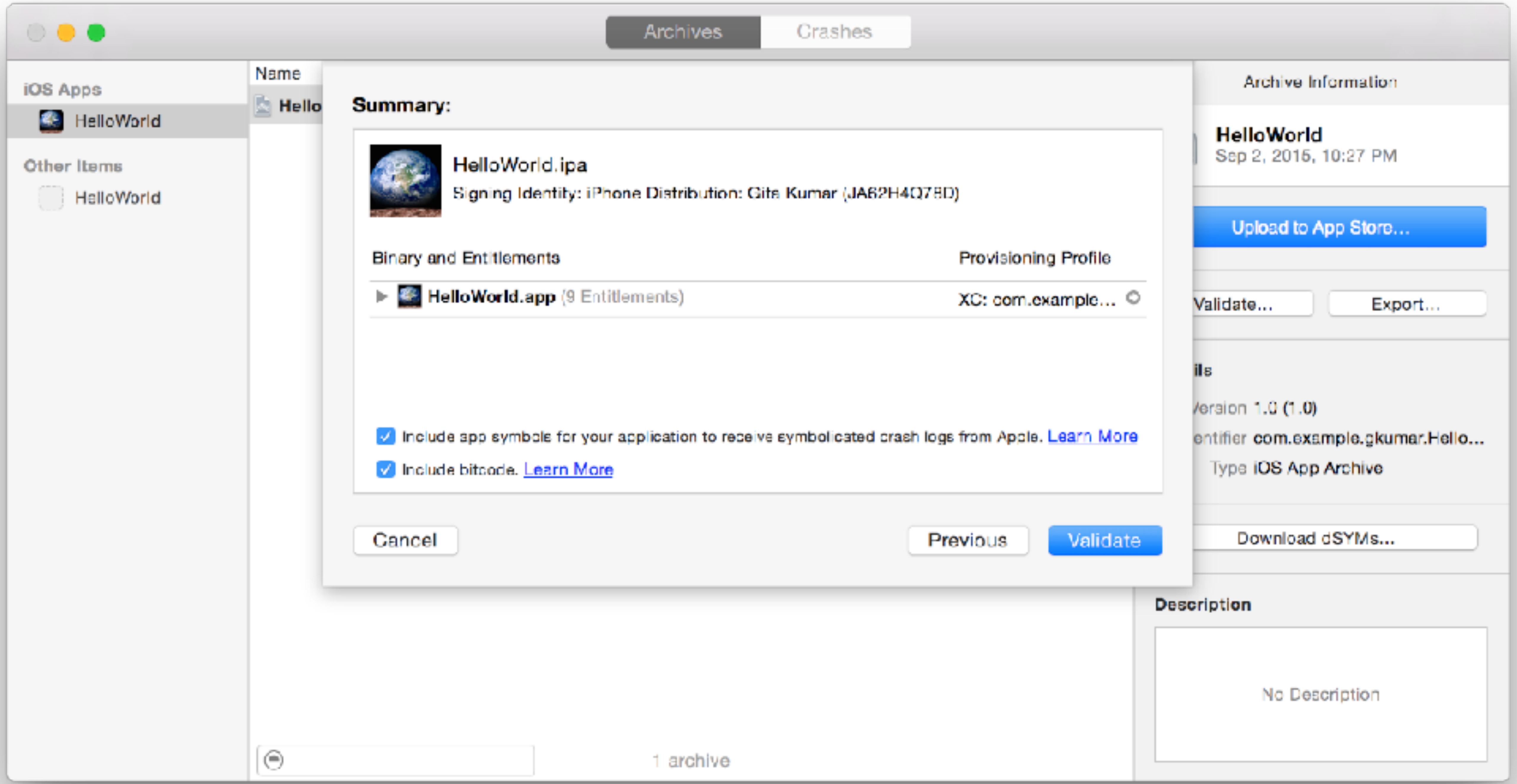
```
▼ (APAViewController *) 0x97698e0
  ▶ UIViewController
  ▶ _skView = (SKView *) 0x977b410
  ▶ _gameLogo = (UIImageView *) 0x9773320
  ▶ _loadingProgressIndicator = (UIActivityIndicatorView *) 0x9774d20
```

```
<CALayer: 0x9773ca0>>
Printing description of self->_gameLogo:
<UIImageView: 0x9773320; frame = (120 -10;
375 220); autoresize = W+H;
userInteractionEnabled = NO; layer =
<CALayer: 0x9773ca0>>
(lldb)
```


Lister
Lister
Groceries
Tech Toys
Today
T...
0 -[AAPLListDocumentsViewController prepareForSegue:sender:]



```
Adventure[25615:303] building the
world
2013-08-04 16:37:54.765
Adventure[25615:303] converted 11965
collision blocks into 36 volumes in
0.005511 seconds
2013-08-04 16:38:24.768
Adventure[25615:303] Finished
finding controllers
```



Summary:

HelloWorld.ipa
Signing Identity: iPhone Distribution: Gita Kumar (JA62H4Q75D)

Binary and Entitlements | Provisioning Profile
HelloWorld.app (9 Entitlements) | XC: com.example...

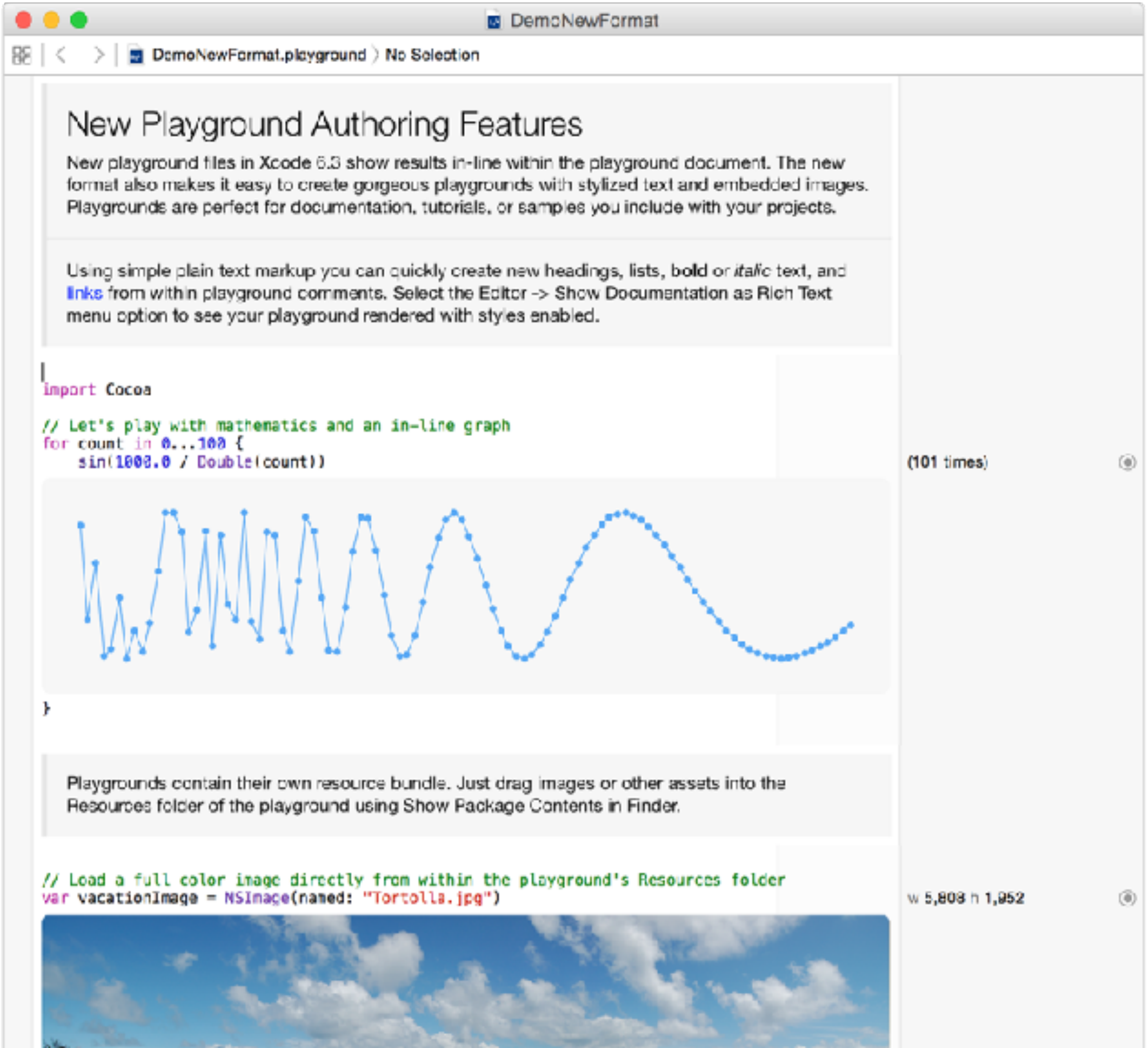
☑ Include app symbols for your application to receive symbolicated crash logs from Apple. Learn More
☑ Include bitcode. Learn More

HelloWorld
Sep 2, 2015, 10:27 PM

ils
Version 1.0 (1.0)
entifier com.example.gkumar.Hello...
Type iOS App Archive

Description
No Description

1 archive



New Playground Authoring Features

New playground files in Xcode 6.3 show results in-line within the playground document. The new format also makes it easy to create gorgeous playgrounds with stylized text and embedded images. Playgrounds are perfect for documentation, tutorials, or samples you include with your projects.

Using simple plain text markup you can quickly create new headings, lists, **bold** or *italic* text, and links from within playground comments. Select the Editor -> Show Documentation as Rich Text menu option to see your playground rendered with styles enabled.

```
import Cocoa

// Let's play with mathematics and an in-line graph
for count in 0...100 {
    sin(1000.0 / Double(count))                                    (101 times)
}
```

Playgrounds contain their own resource bundle. Just drag images or other assets into the Resources folder of the playground using Show Package Contents in Finder.

```
// Load a full color image directly from within the playground's Resources folder
var vacationImage = NSImage(named: "Tortolla.jpg")                 w 5,808 h 1,952
```

XCODE

- Create New Project
- Add Button and View to Storyboard
- Change the text in the button and the color of the view
- Run in the simulator
- Zip (01)

2017/05/23 – START BUILDING MOBILE APPS

AGENDA

- ~~Mobile Intro~~
- ~~Learn basics of Xcode IDE~~
- stretch break
- Learn programming basics with Swift
- stretch break
- Make a basic iOS application
- stretch break
- Dive a little deeper into Swift
- Resources

BREAK

2017/05/23 – START BUILDING MOBILE APPS

AGENDA

- ~~Mobile Intro~~
- ~~Learn basics of Xcode IDE~~
- ~~stretch break~~
- Learn programming basics with Swift
- stretch break
- Make a basic iOS application
- stretch break
- Dive a little deeper into Swift
- Resources

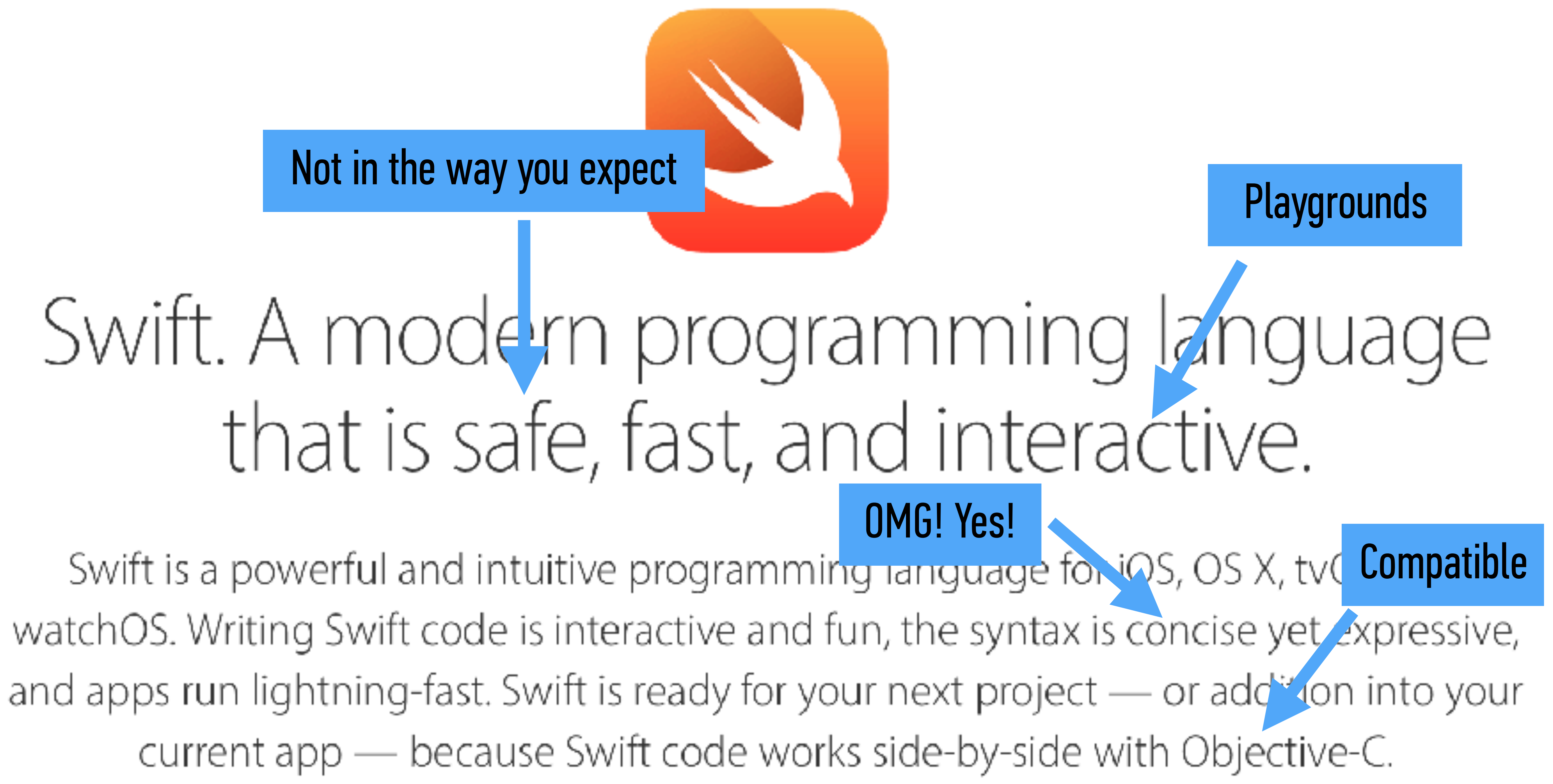
SWIFT BASICS

SWIFT

Swift. A modern programming language that is safe, fast, and interactive.

Swift is a powerful and intuitive programming language for iOS, OS X, tvOS, and watchOS. Writing Swift code is interactive and fun, the syntax is concise yet expressive, and apps run lightning-fast. Swift is ready for your next project — or addition into your current app — because Swift code works side-by-side with Objective-C.

SWIFT – THE GOOD



Not in the way you expect

Playgrounds

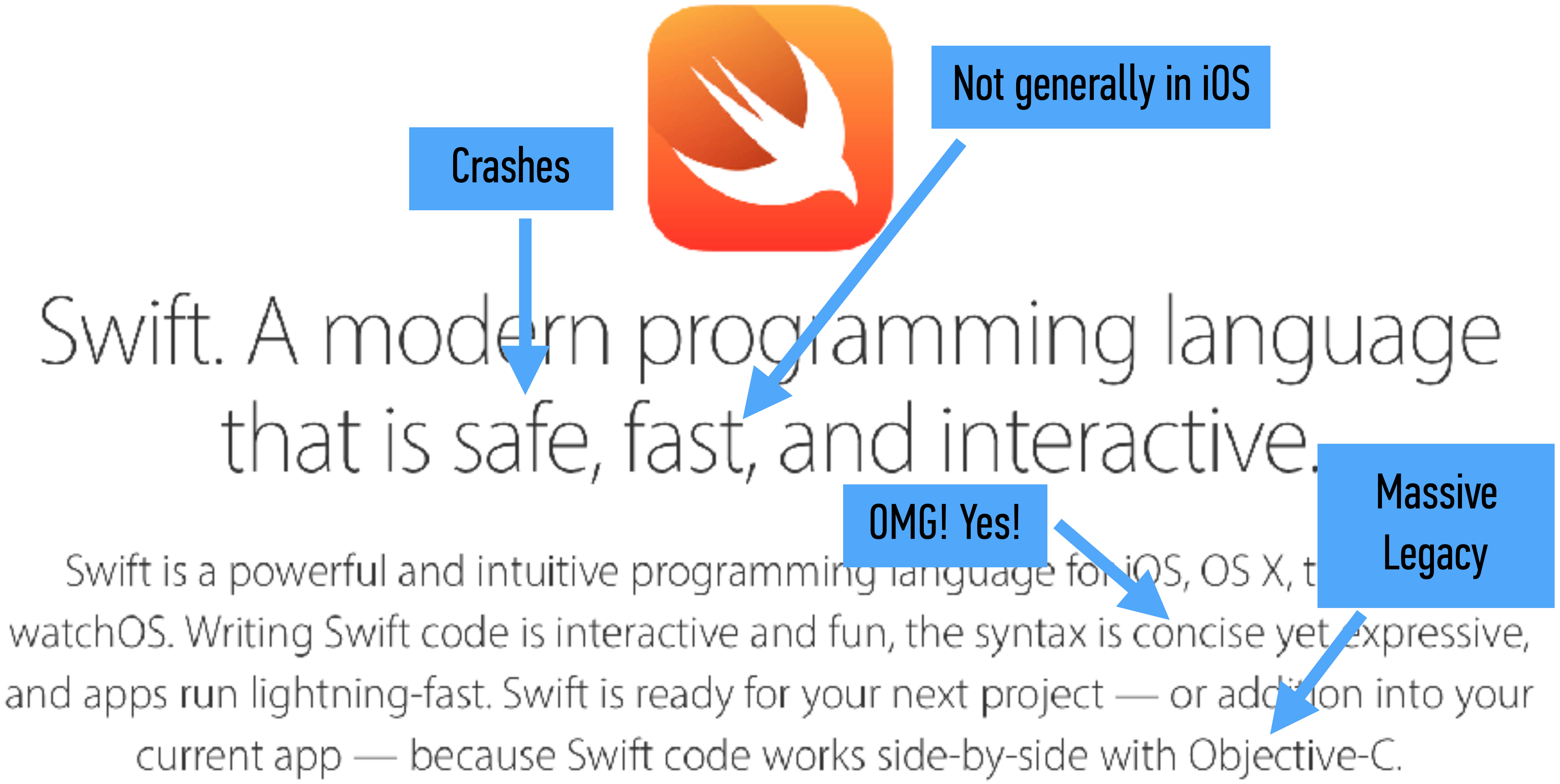
OMG! Yes!

Compatible

Swift. A modern programming language that is safe, fast, and interactive.

Swift is a powerful and intuitive programming language for iOS, OS X, tv watchOS. Writing Swift code is interactive and fun, the syntax is concise yet expressive, and apps run lightning-fast. Swift is ready for your next project — or addition into your current app — because Swift code works side-by-side with Objective-C.

SWIFT – THE BAD



Swift. A modern programming language that is safe, fast, and interactive.

Swift is a powerful and intuitive programming language for iOS, OS X, t watchOS. Writing Swift code is interactive and fun, the syntax is concise yet expressive, and apps run lightning-fast. Swift is ready for your next project — or addition into your current app — because Swift code works side-by-side with Objective-C.

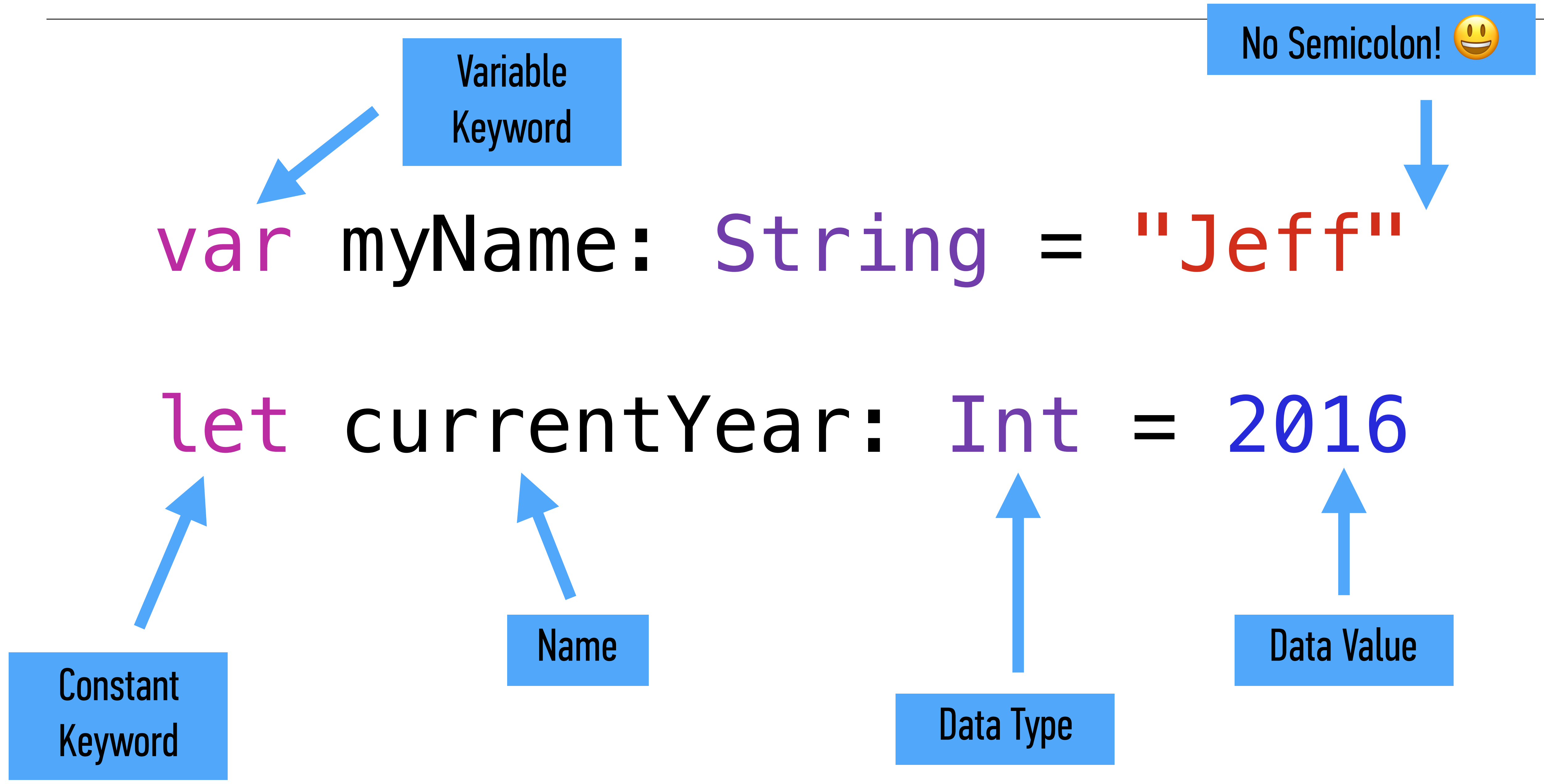
BASIC SWIFT TYPES

```
String
Int
Double
Bool
Optional
Array
Dictionary
```

DECLARING VARIABLES AND CONSTANTS

```swift
var myName: String = "Jeff"

let currentYear: Int = 2016
```

DECLARING VARIABLES AND CONSTANTS



```
var myName: String = "Jeff"

let currentYear: Int = 2016
```

DECLARING VARIABLES AND CONSTANTS

Type Inference

```
var myName = "Jeff"

let currentYear = 2016
```

BOOL

```
let isNervous = true
```

BOOL

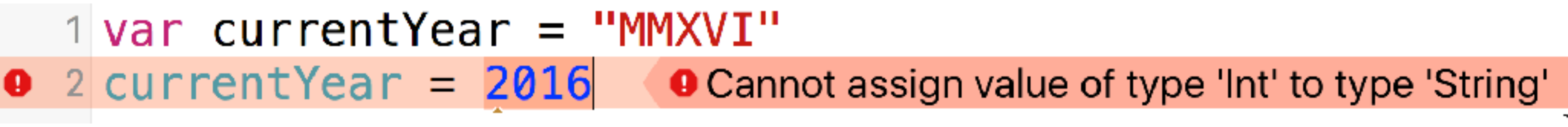
```swift
let isNervous = true
```

Objective-C used YES / NO

STRONGLY TYPED

- Once a variable is declared, its type cannot change
    - This is for both explicit and inferred types
- This makes code easier to reason about
- But it makes conversion from one type to another a PITA
- This is where Swift differs most from “easy” languages
    - Javascript, Python, Ruby, etc

STRONGLY TYPED


```
var currentYear = "MMXVI"
currentYear = 2016
```
Cannot assign value of type 'Int' to type 'String'

BASIC SWIFT TYPES

```
String
Int
Double
Bool
Optional
Array
Dictionary
```

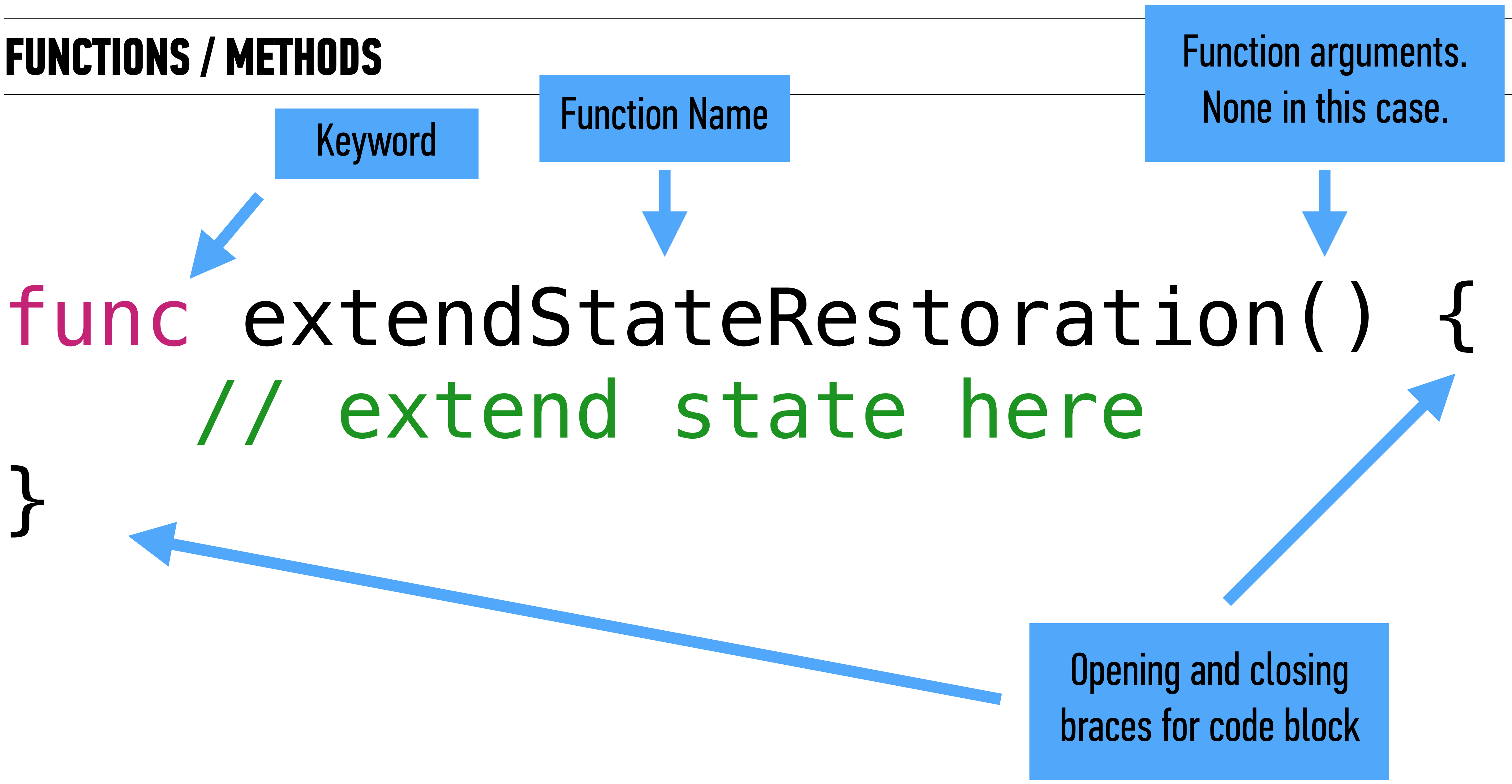
XCODE PLAYGROUND

- Create a new iOS playground
- Declare constants of type: String, Int, Double, Bool
- Show how to check type
- Try to change constant
- Zip (02)

FUNCTIONS / METHODS

```swift
func extendStateRestoration() {
    // extend state here
}
```

FUNCTIONS / METHODS



```swift
func extendStateRestoration() {
    // extend state here
}
```

Keyword
Function Name
Function arguments. None in this case.
Opening and closing braces for code block

FUNCTIONS / METHODS WITH ARGUMENTS

Declaration Site

```
func openURL(_ url: URL) {
    // Log the URL to the console
    NSLog(“The URL is: \(url)”)
}
```

Call Site

```
openURL(myURL)
```

FUNCTIONS / METHODS WITH ARGUMENTS

Function Name

External Name

Internal Name

Type

Declaration Site

```
func openURL(_ url: URL) {
    // Log the URL to the console
    NSLog("The URL is: \(url)")
}
```

Call Site

```
openURL(myURL)
```

FUNCTIONS / METHODS WITH ARGUMENTS

Declaration Site

```
func open(url: URL) {
    // Log the URL to the console
    NSLog(“The URL is: \(url)”)
}
```

Call Site

```
open(url: myURL)
```

FUNCTIONS / METHODS WITH ARGUMENTS

Function Name

Internal and external name

Type

Declaration Site

```
func open(url: URL) {
    // Log the URL to the console
    NSLog(“The URL is: \(url)”)
}
```

Call Site

```
open(url: myURL)
```

SOME DETAILS

```
func open(url: URL) {
    // Log the URL to the console
    NSLog(“The URL is: \(url)”)
}
```

Print / Log command

String “Interpolation”

FUNCTIONS / METHODS WITH RETURN VALUES

```
func canOpenURL(_ url: URL) -> Bool {
    // I can totally open this URL
    return true
}
```

FUNCTIONS / METHODS WITH RETURN VALUES

Swift ASCII Art Indicating there is a return value

```
func canOpenURL(_ url: URL) -> Bool {
    // I can totally open this URL
    return true
}
```

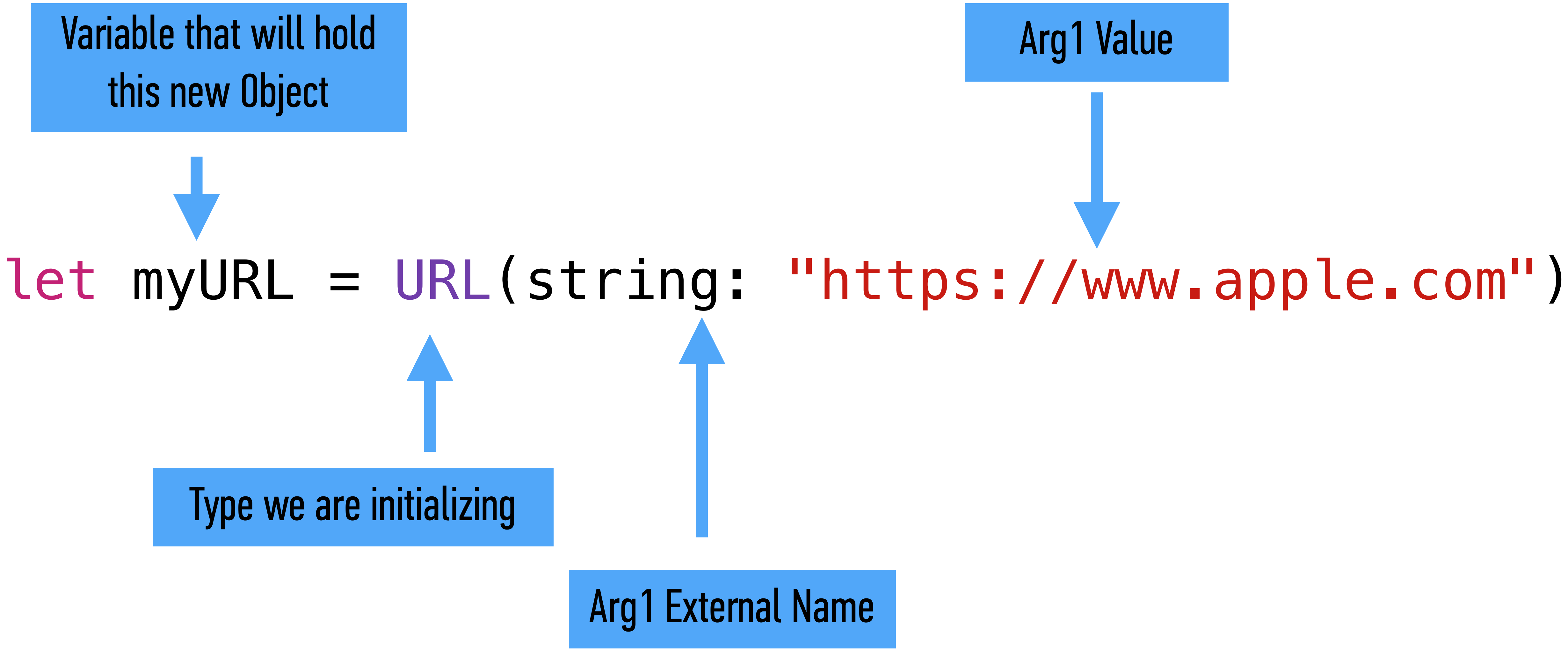
Return Type

Required: Any function that has a return type must call return before the end

USING INITIALIZERS

```
let myURL = URL(string: "https://www.apple.com")
```

USING INITIALIZERS



```
let myURL = URL(string: "https://www.apple.com")
```

XCODE PLAYGROUND

- Create a no argument function
    - Call it
- Create a 1 argument function
    - Call it
- Create a function that returns a value
    - Call it
- Zip (03)

FUNCTIONS / METHODS WITH MULTIPLE ARGUMENTS

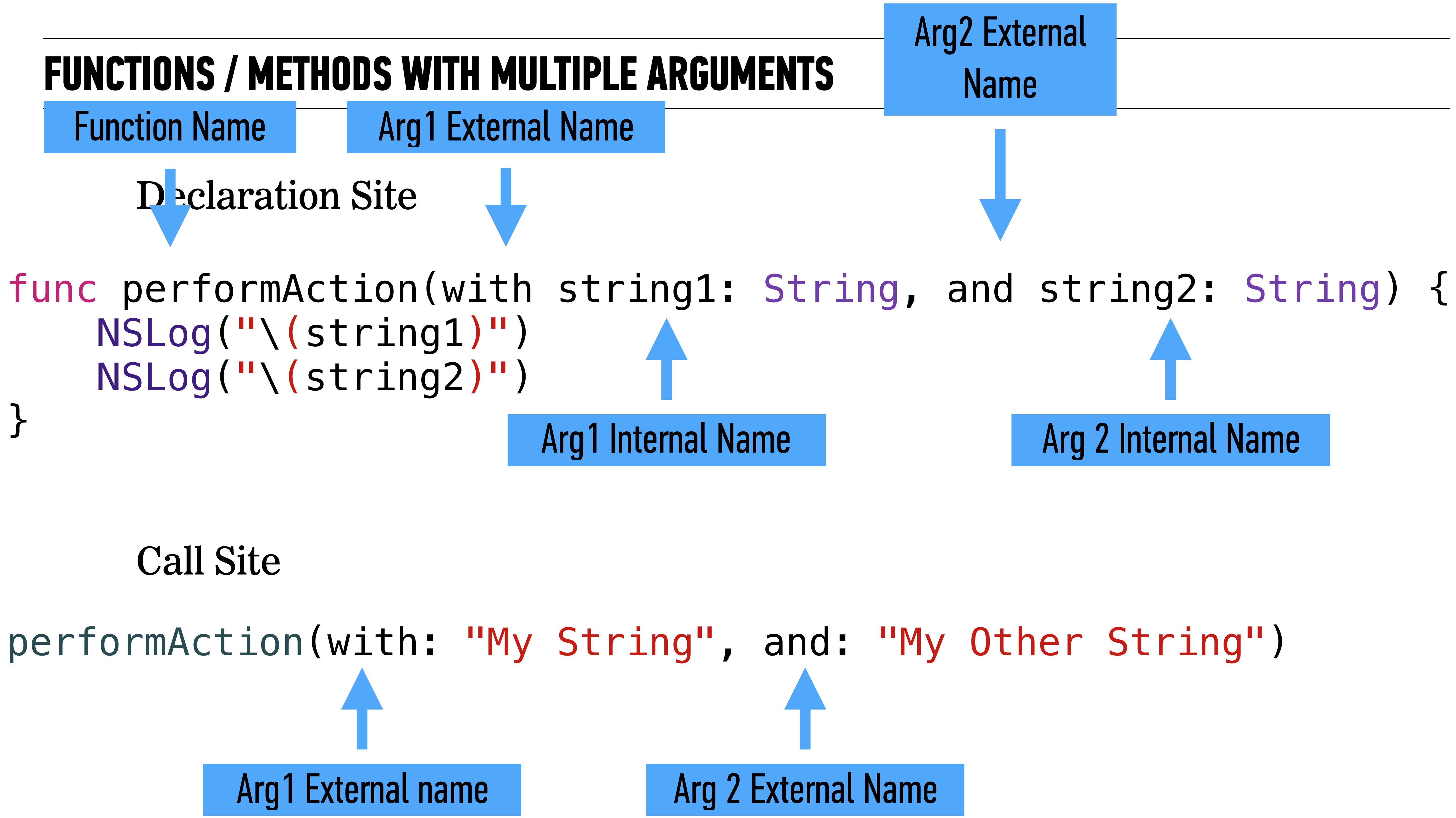
Declaration Site

```
func performAction(with string1: String, and string2: String) {
    NSLog("\(string1)")
    NSLog("\(string2)")
}
```

Call Site

```
performAction(with: "My String", and: "My Other String")
```

FUNCTIONS / METHODS WITH MULTIPLE ARGUMENTS



Declaration Site

```
func performAction(with string1: String, and string2: String) {
    NSLog("\(string1)")
    NSLog("\(string2)")
}
```

Call Site

```
performAction(with: "My String", and: "My Other String")
```

METHODS SHOULD SOUND LIKE PROSE

```
+ (void)animateWithDuration:(NSTimeInterval)duration
                      delay:(NSTimeInterval)delay
     usingSpringWithDamping:(CGFloat)dampingRatio
      initialSpringVelocity:(CGFloat)velocity
                    options:(UIViewAnimationOptions)options
                 animations:(void)animations
                 completion:(void)finished
```

METHODS SHOULD SOUND LIKE PROSE

```
func animate(withDuration duration: TimeInterval,
             delay: TimeInterval,
             usingSpringWithDamping dampingRatio: CGFloat,
             initialSpringVelocity velocity: CGFloat,
             options: UIViewAnimationOptions,
             animations: (void),
             completion: (void))
```

2017/05/23 – START BUILDING MOBILE APPS

AGENDA

- ~~Mobile Intro~~
- ~~Learn basics of Xcode IDE~~
- ~~stretch break~~
- ~~Learn programming basics with Swift~~
- stretch break
- Make a basic iOS application
- stretch break
- Dive a little deeper into Swift
- Resources

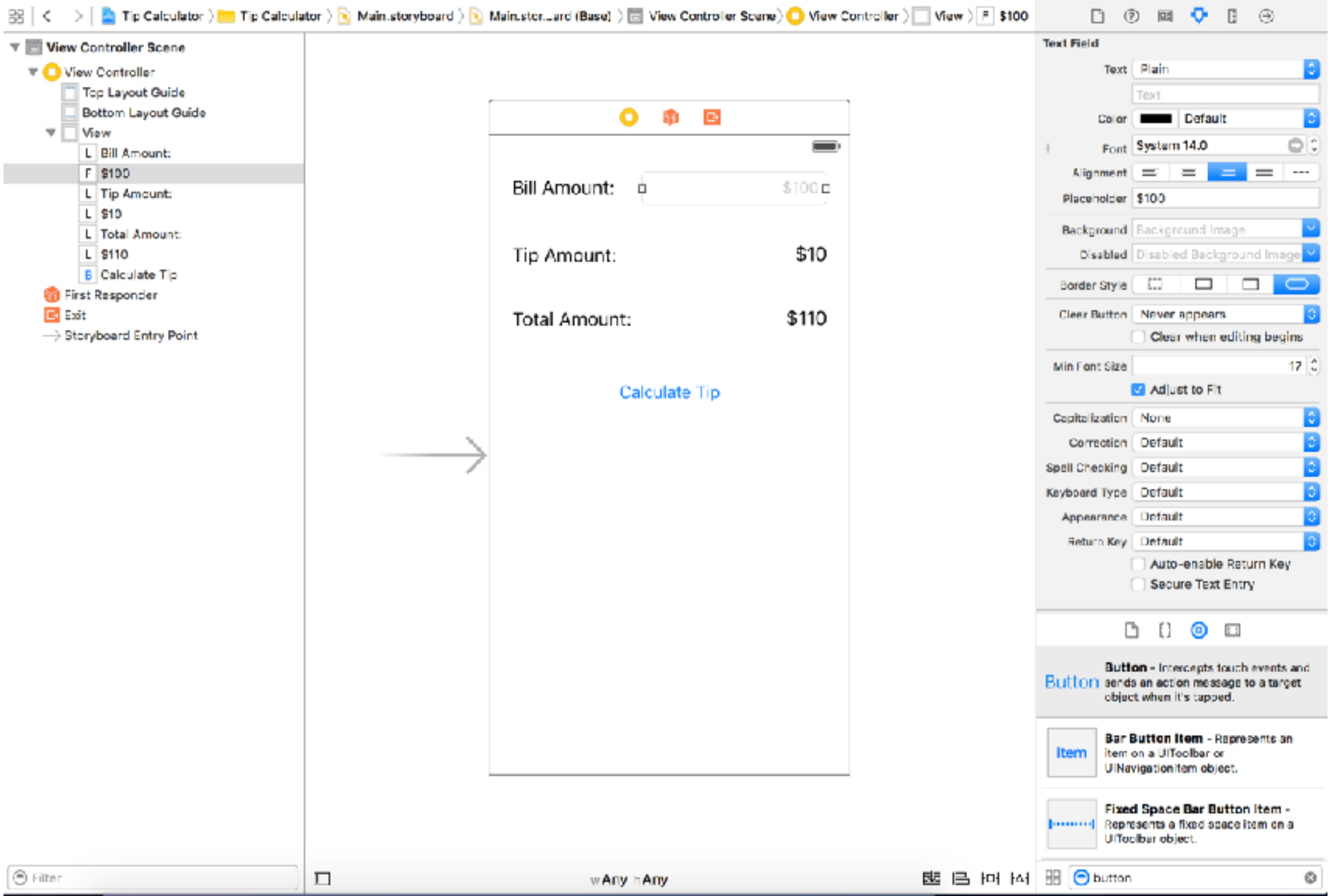
BREAK

2017/05/23 – START BUILDING MOBILE APPS

AGENDA

- ~~Mobile Intro~~
- ~~Learn basics of Xcode IDE~~
- ~~stretch break~~
- ~~Learn programming basics with Swift~~
- ~~stretch break~~
- Make a basic iOS application
- stretch break
- Dive a little deeper into Swift
- Resources

IOS APP


Tip Calculator 〉 Tip Calculator 〉 Main.storyboard 〉 Main.stor...ard (Base) 〉 View Controller Scene 〉 View Controller 〉 View 〉 $100

- View Controller Scene
  - View Controller
    - Top Layout Guide
    - Bottom Layout Guide
    - View
      - Bill Amount:
      - $100
      - Tip Amount:
      - $10
      - Total Amount:
      - $110
      - Calculate Tip
  - First Responder
  - Exit
  - Storyboard Entry Point

Bill Amount: $100

Tip Amount: $10

Total Amount: $110

Calculate Tip

Text Field

Text	Plain
	Text
Color	Default
Font	System 14.0
Alignment	
Placeholder	$100
Background	Background Image
Disabled	Disabled Background Image
Border Style	
Clear Button	Never appears
	Clear when editing begins
Min Font Size	17
	✓ Adjust to Fit
Capitalization	None
Correction	Default
Spell Checking	Default
Keyboard Type	Default
Appearance	Default
Return Key	Default
	Auto-enable Return Key
	Secure Text Entry

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

Bar Button Item - Represents an item on a UIToolbar or UINavigationItem object.

Fixed Space Bar Button Item - Represents a fixed space item on a UIToolbar object.

Filter

wAny hAny

button

PROPERTIES AND METHODS - COCOA OBJECTS

- Properties - Describe an object
    - eg. A physical car has:
        - Make
        - Model
        - Color
- Methods - Functions that let an object do stuff
    - eg. A car can do:
        - Start engine
        - Drive forward
- The difference between these and our playgrounds is these live at the top level of our custom objects in Cocoa.

IBOUTLETS AND IBACTIONS - COCOA OBJECTS

- IBOutlet - Property that lets our code communicate with the interface
- IBAction - Function that lets the interface communicate with our code

TIP CALCULATE SNEAK PEAK

```swift
class ViewController: UIViewController {

    @IBOutlet weak var totalAmountLabel: UILabel!
    @IBOutlet weak var tipAmountLabel: UILabel!
    @IBOutlet weak var billAmountTextField: UITextField!

    @IBAction func calculateTip(_ sender: UIButton) {
        // get the double value of the string in the text field
        let billAmount = Double(self.billAmountTextField.text ?? "") ?? 0

        // hard code our tip percentage
        let tipPercentage = 0.2

        // calculate the tip amount and update the UI
        let tipAmount = billAmount * tipPercentage
        self.tipAmountLabel.text = "$\(tipAmount)"

        // calculate the total amount and update the UI
        let total = billAmount + tipAmount
        self.totalAmountLabel.text = "$\(total)"
    }
}
```

XCODE

- Create a tip calculator
- Layout the interface
- Create an IBAction for the button
- Create outlets for the labels
- Do the math
- Zip (04)

2017/05/23 – START BUILDING MOBILE APPS

AGENDA

- ~~Mobile Intro~~
- ~~Learn basics of Xcode IDE~~
- ~~stretch break~~
- ~~Learn programming basics with Swift~~
- ~~stretch break~~
- ~~Make a basic iOS application~~
- stretch break
- Dive a little deeper into Swift
- Resources

BREAK

2017/05/23 – START BUILDING MOBILE APPS

AGENDA

- ~~Mobile Intro~~
- ~~Learn basics of Xcode IDE~~
- ~~stretch break~~
- ~~Learn programming basics with Swift~~
- ~~stretch break~~
- ~~Make a basic iOS application~~
- ~~stretch break~~
- Dive a little deeper into Swift
- Resources

SWIFT OPTIONALS

BASIC SWIFT TYPES

```
String
Int
Double
Bool
Optional
Array
Dictionary
```

BASIC SWIFT TYPES

~~String~~
~~Int~~
~~Double~~
~~Bool~~
Optional ← What the heck is this thing?
Array
Dictionary

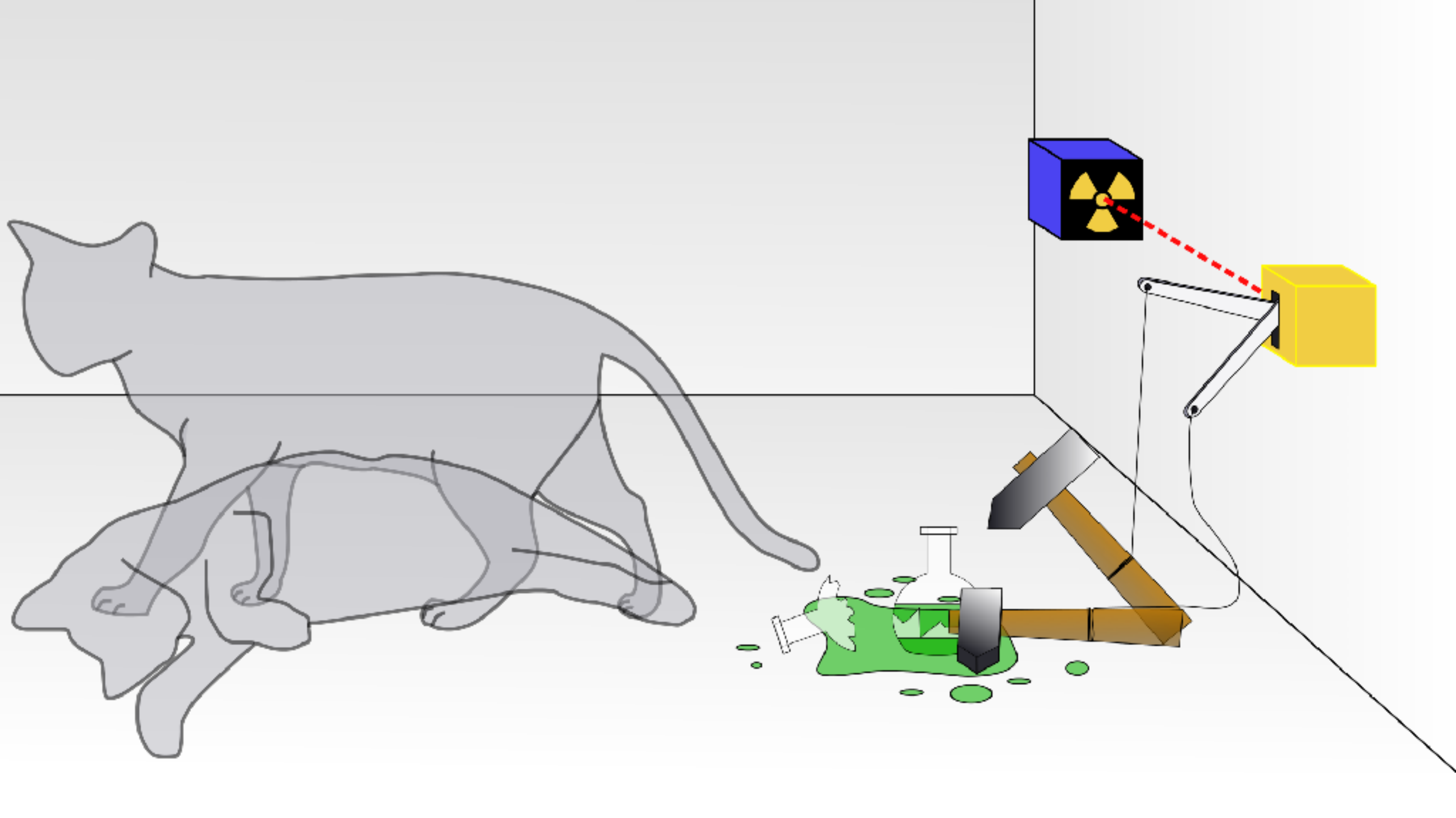
BASIC SWIFT TYPES

~~String~~
~~Int~~
~~Double~~
~~Bool~~
Optional ←
Array
Dictionary

Swift has its own

Schrödinger Cat Type

😸 🙀



OPTIONALS

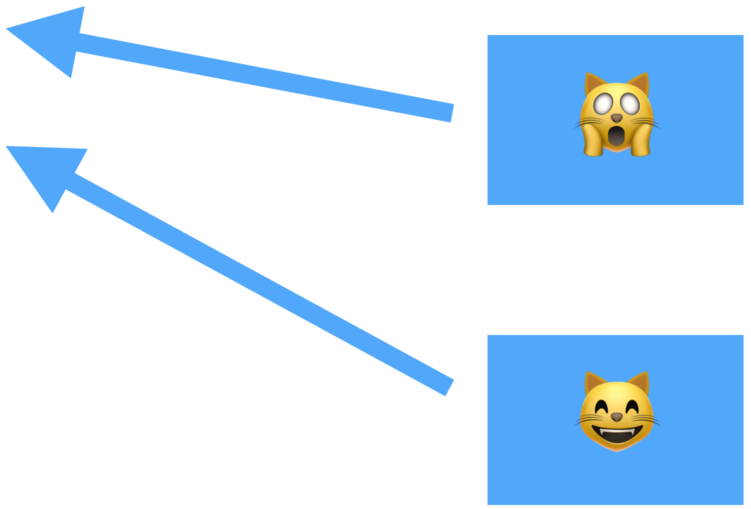
- Optionals are actually very similar to Booleans
- Booleans are special kind of type called an ENUM(eration)
- Enums contain a set list of possible options
- For example:
    - True / False
    - Logged In / Logged Out
    - Not Downloading / Downloading / Downloaded
    - Not Downloading / Downloading / Downloaded / Error

OPTIONALS

- Optionals are just this same Enum concept but the options are
    - None
    - Some

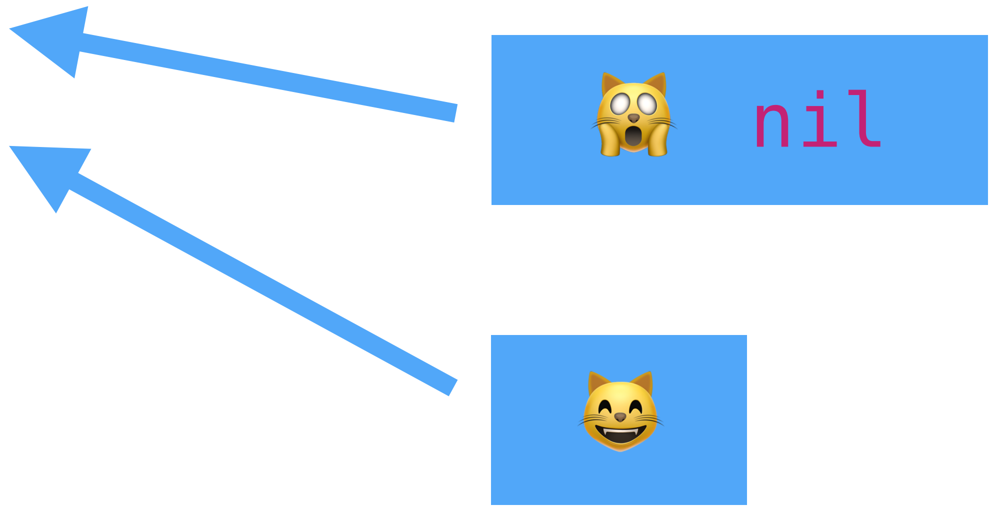
OPTIONALS

- Optionals are just this same Enum concept but the options are
  - None
  - Some



OPTIONALS

- Optionals are just this same Enum concept but the options are
  - None
  - Some



OPTIONALS

- Optionals allow Swift developers to explicitly specify what is known and when.
  - “compile time” knowledge
    - vs
  - “runtime” knowledge
- For example. Downloading an image:
  - You have a URL. You tell the code to download the URL
  - But there is no guarantee the server will actually send you an image
  - It could send you a 404 error which is an HTML file… or nothing…
  - Either way, its not an image and you don’t know this at “compile time”

EXPRESSING OPTIONALITY

```
String?
Int?
Double?
Bool?
Anything?
```

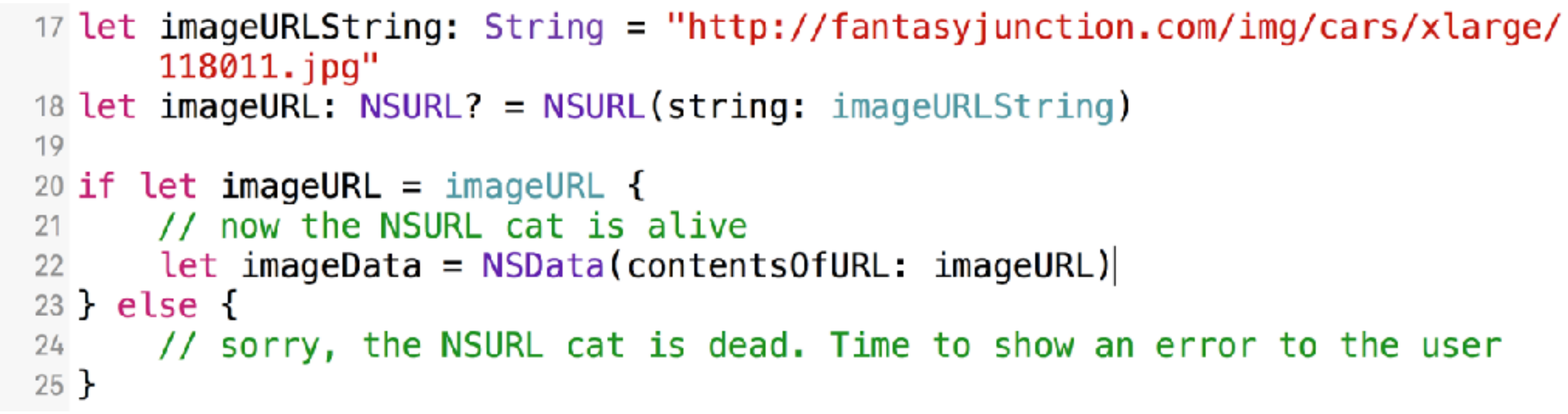
? - Indicates Optional

```swift
let imageURLString: String = "http://fantasyjunction.com/img/
    cars/xlarge/118011.jpg"
let imageURL: NSURL? = NSURL(string: imageURLString)
let imageData: NSData? = NSData(contentsOfURL: imageURL!)
let image: UIImage? = UIImage(data: imageData!)
```

```
"http://fantasyjunction.co...

http://fantasyjunction.co...
<ffd8ffe0 00104a46 494...
w 800 h 533
```

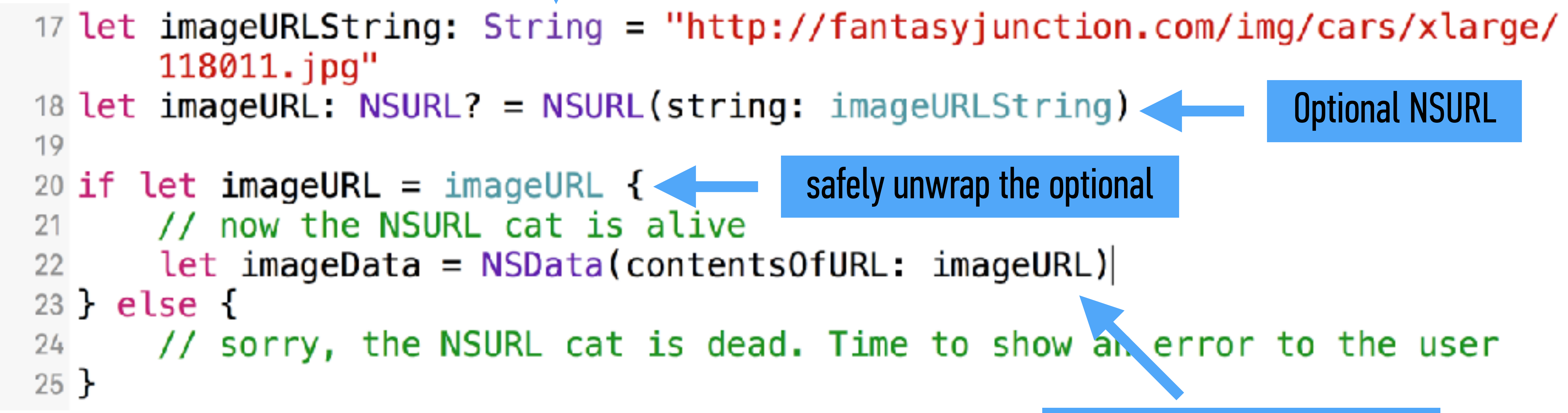
OPTIONALS

- ? = The type we're dealing with is optional
- ! = I'm super confident that the cat is alive
    - If I'm wrong, I accept that my app will crash for my users if the cat is dead
- But, there is a way to deal with optionals in a safe way
    - So that you can present an error to the user if the cat is dead
    - This is never an easy conversation :-/



```swift
let imageURLString: String = "http://fantasyjunction.com/img/cars/xlarge/118011.jpg"
let imageURL: NSURL? = NSURL(string: imageURLString)

if let imageURL = imageURL {
    // now the NSURL cat is alive
    let imageData = NSData(contentsOfURL: imageURL)
} else {
    // sorry, the NSURL cat is dead. Time to show an error to the user
}
```

Confirmed String (non-optional)



```swift
let imageURLString: String = "http://fantasyjunction.com/img/cars/xlarge/118011.jpg"
let imageURL: NSURL? = NSURL(string: imageURLString)

if let imageURL = imageURL {
    // now the NSURL cat is alive
    let imageData = NSData(contentsOfURL: imageURL)
} else {
    // sorry, the NSURL cat is dead. Time to show an error to the user
}
```

Optional NSURL

safely unwrap the optional

Now the imageURL constant can be used safely

XCODE PLAYGROUNDS

- Create an optional string
- Set it to NIL
- Experiment with Printing it
- Safely Unwrap it
- Zip (05)

SWIFT COLLECTION TYPES

BASIC SWIFT TYPES

~~`String`~~
~~`Int`~~
~~`Double`~~
~~`Bool`~~
~~`Optional`~~
`Array`
`Dictionary`

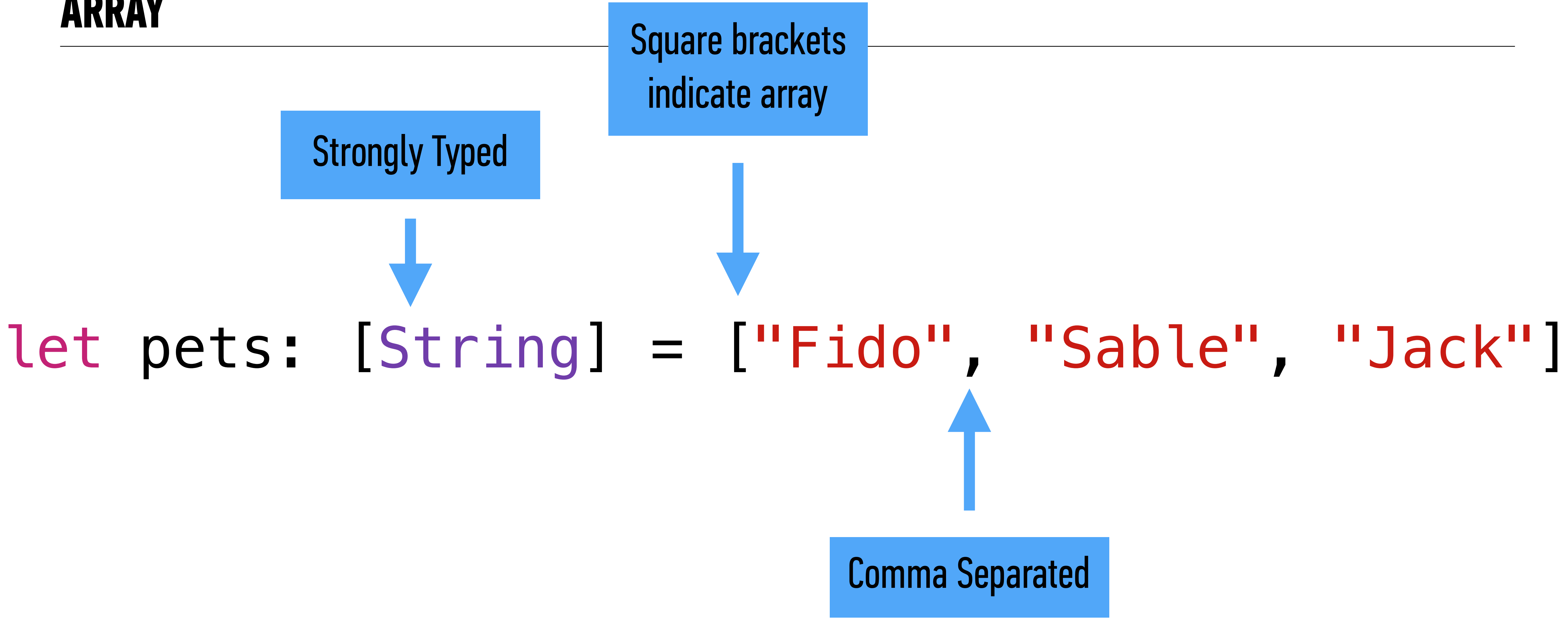
ARRAY

- Ordered list of items
- Its #1 job in life is to keep items in order
- Strongly Typed
- Mixed Type arrays are allowed but not recommended
- Get items out by asking the array for the item at an Integer index
- Arrays are 0-indexed
- Runtime crash caused by asking the Array for an item that doesn’t exist.

ARRAY

```
let pets: [String] = ["Fido", "Sable", "Jack"]
```

ARRAY



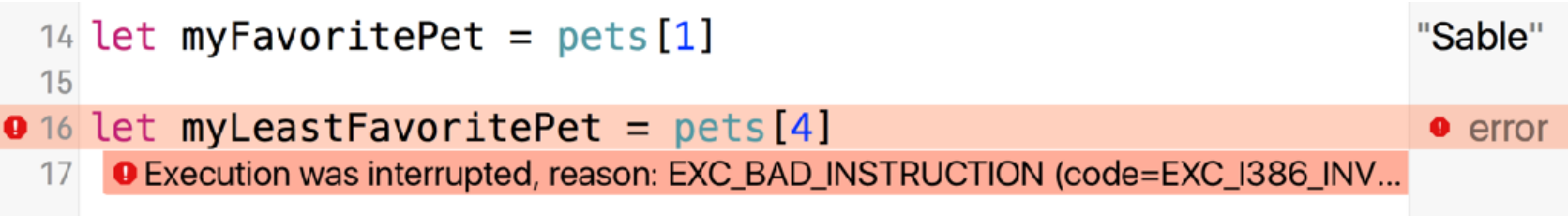
```
let pets: [String] = ["Fido", "Sable", "Jack"]
```

ARRAY

Type can be inferred

```
let pets = ["Fido", "Sable", "Jack"]
```

ARRAY



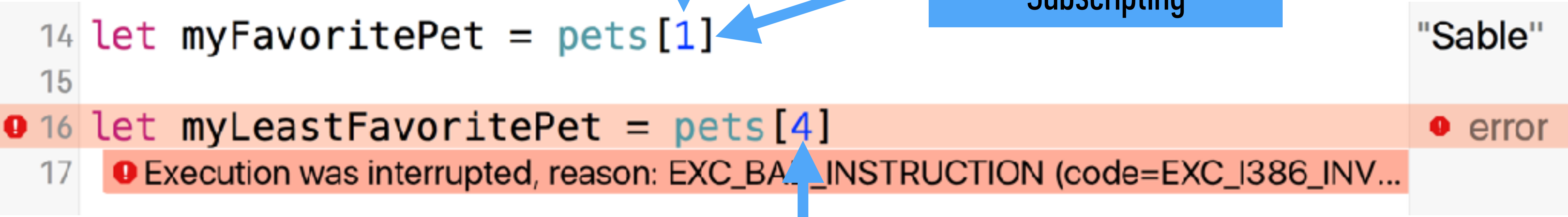
```
let myFavoritePet = pets[1]                                           "Sable"

let myLeastFavoritePet = pets[4]                                      error
  Execution was interrupted, reason: EXC_BAD_INSTRUCTION (code=EXC_I386_INV...
```

ARRAY

Pick your item via its number in the list

List is 0 indexed. So 1 is the second item.

Special square brackets syntax is called "Subscripting"



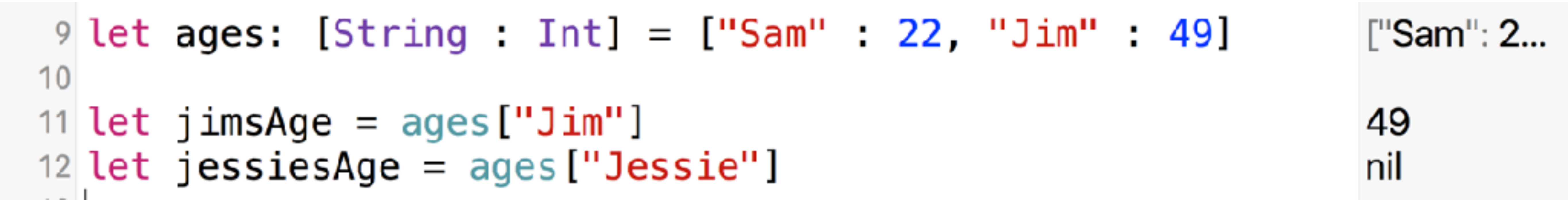
```
let myFavoritePet = pets[1]                                   "Sable"

let myLeastFavoritePet = pets[4]                              error
Execution was interrupted, reason: EXC_BAD_INSTRUCTION (code=EXC_I386_INV...
```

If you ask for an entry that does not exist, you get a runtime crash.

DICTIONARY

- Key / value pairs
- Unordered
- The Key is strongly typed and the Value is strongly typed
  - But they do not need to be the same type.
- Access the value by asking for the dictionary for it via the key.
- Its ok to ask the dictionary for an item with a key that does not exist.
  - It returns nothing. No crash.

DICTIONARY



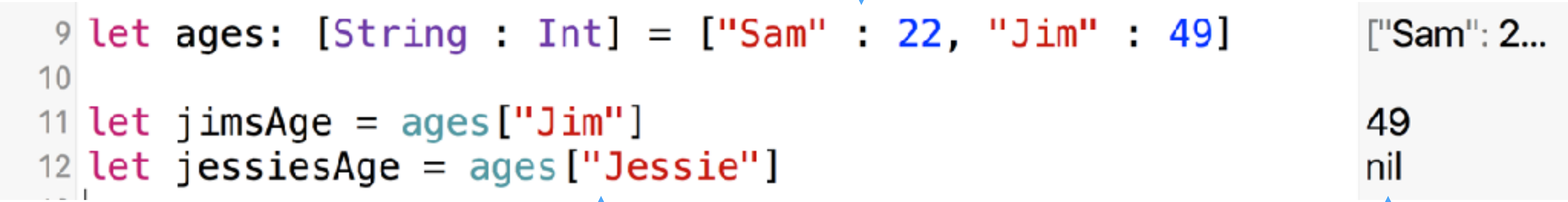
```
let ages: [String : Int] = ["Sam" : 22, "Jim" : 49]

let jimsAge = ages["Jim"]
let jessiesAge = ages["Jessie"]
```

```
["Sam": 2...

49
nil
```

DICTIONARY

Strongly Typed

Square bracket with Colon indicates Dictionary



```
let ages: [String : Int] = ["Sam" : 22, "Jim" : 49]        ["Sam": 2...

let jimsAge = ages["Jim"]                                   49
let jessiesAge = ages["Jessie"]                             nil
```

Subscripting to get the value out

When you ask for something that doesn't exist, you get NIL

XCODE PLAYGROUNDS

- Create an Array
- Get an item out of the array
- Create a dictionary
- Get an item out of the dictionary
- Zip (06)

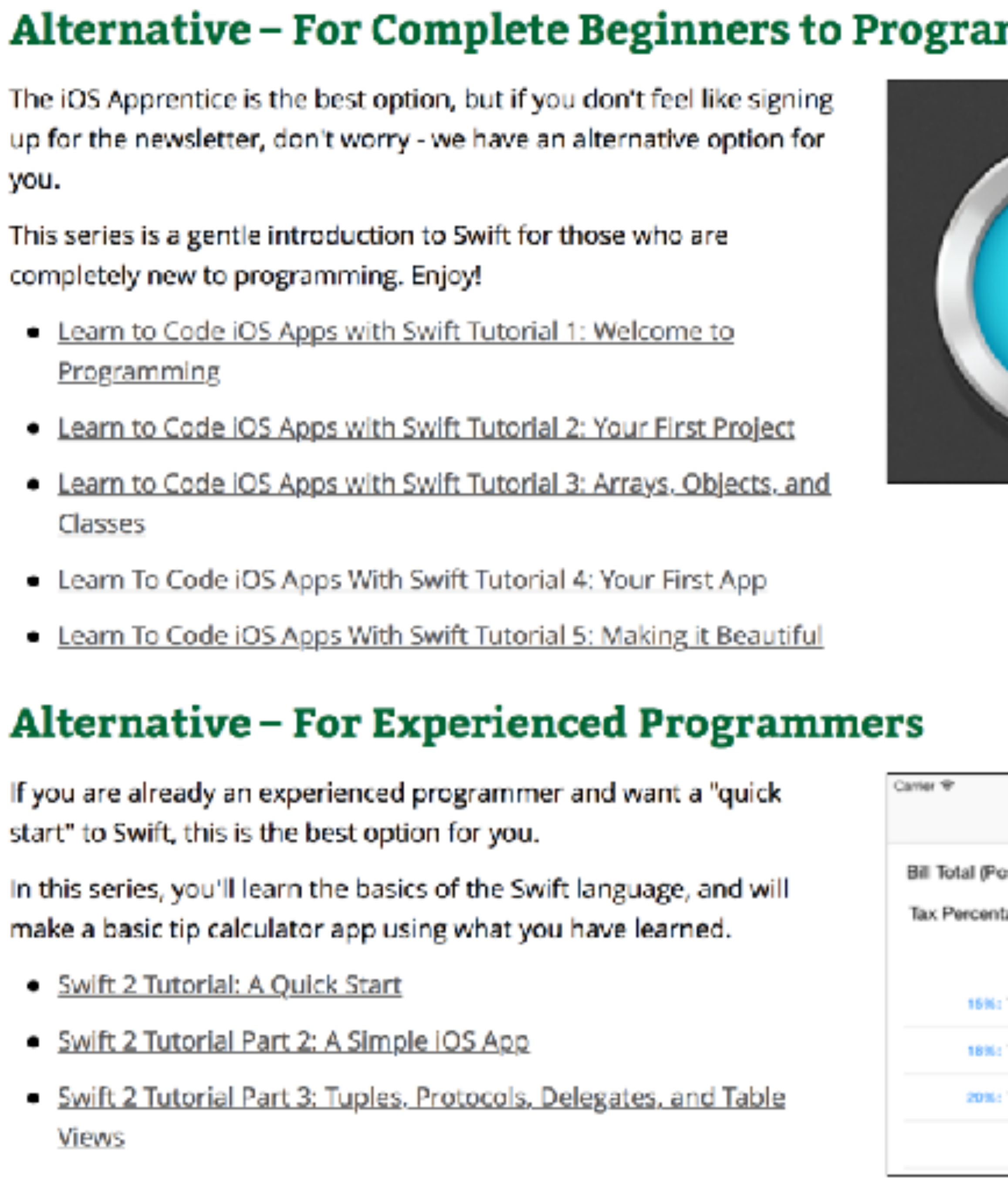
AGENDA

- ~~Mobile Intro~~
- ~~Learn basics of Xcode IDE~~
- ~~stretch break~~
- ~~Learn programming basics with Swift~~
- ~~stretch break~~
- ~~Make a basic iOS application~~
- ~~stretch break~~
- ~~Dive a little deeper into Swift~~
- Resources

SWIFT AND COCOA TOUCH RESOURCES

RESOURCES

https://www.raywenderlich.com/category/swift



Alternative – For Complete Beginners to Progra

The iOS Apprentice is the best option, but if you don't feel like signing up for the newsletter, don't worry - we have an alternative option for you.

This series is a gentle introduction to Swift for those who are completely new to programming. Enjoy!

- Learn to Code iOS Apps with Swift Tutorial 1: Welcome to Programming
- Learn to Code iOS Apps with Swift Tutorial 2: Your First Project
- Learn to Code iOS Apps with Swift Tutorial 3: Arrays, Objects, and Classes
- Learn To Code iOS Apps With Swift Tutorial 4: Your First App
- Learn To Code iOS Apps With Swift Tutorial 5: Making it Beautiful

Alternative – For Experienced Programmers

If you are already an experienced programmer and want a "quick start" to Swift, this is the best option for you.

In this series, you'll learn the basics of the Swift language, and will make a basic tip calculator app using what you have learned.

- Swift 2 Tutorial: A Quick Start
- Swift 2 Tutorial Part 2: A Simple iOS App
- Swift 2 Tutorial Part 3: Tuples, Protocols, Delegates, and Table Views

2017/05/23 – START BUILDING MOBILE APPS

AGENDA

- ~~Mobile Intro~~
- ~~Learn basics of Xcode IDE~~
- ~~stretch break~~
- ~~Learn programming basics with Swift~~
- ~~stretch break~~
- ~~Make a basic iOS application~~
- ~~stretch break~~
- ~~Dive a little deeper into Swift~~
- ~~Resources~~

Q&A

IOS DEVELOPMENT 101

EXIT TICKETS

DON'T FORGET TO FILL OUT YOUR EXIT TICKET