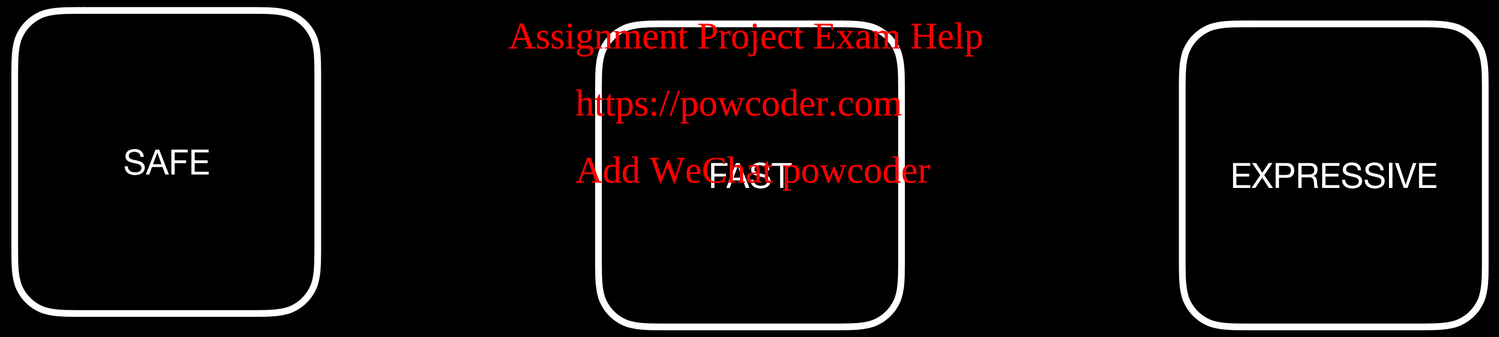
Unit 1 — Lesson 1: Introduction to Swift and Playgrounds

A little history

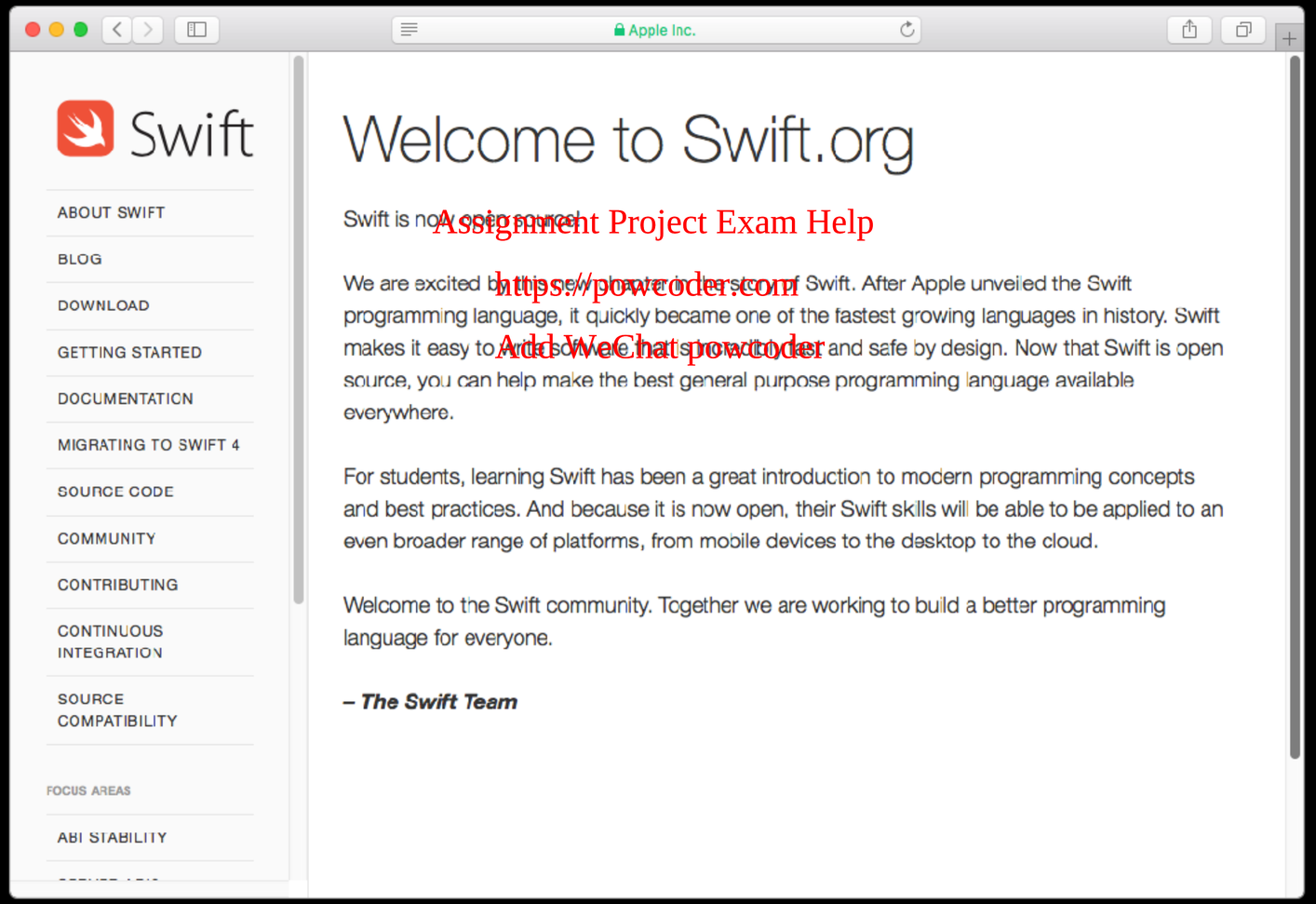
A modern language



A safe language

- Explicit object "types"
- Type inference
- Optionals
- Error handling

Open Source



Hello, world

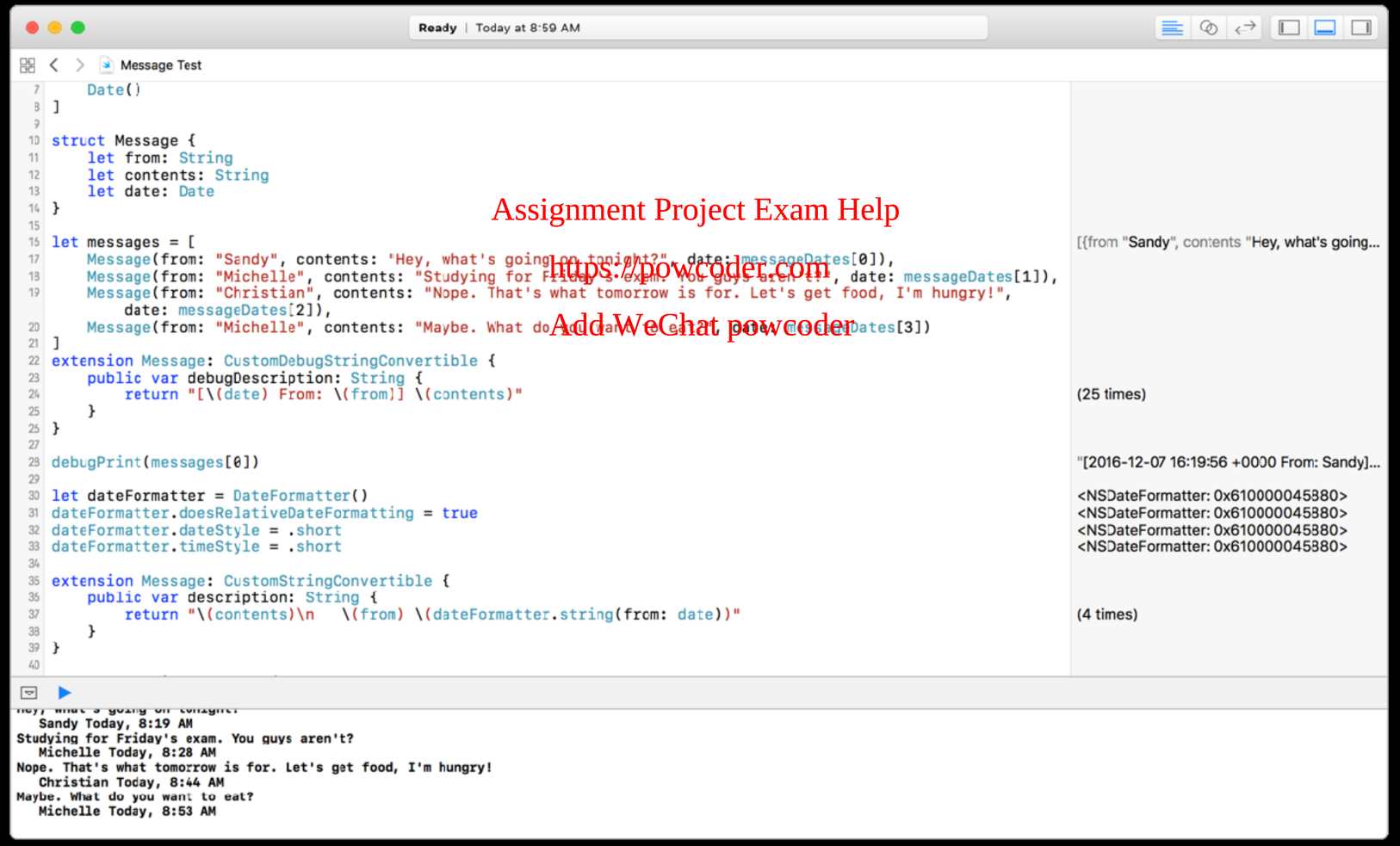
```
print("Hello, world!")
```


main.swift

Hello, world

1. Open Terminal
2. Type `swift` and press Enter
3. Type `print("Hello, world!")` and press Return
4. Type `:quit` and press Return
5. Quit Terminal

Playgrounds



Hello, world

1. Open Xcode
2. Choose File > New > Playground
3. Select iOS, select the Blank template and click Next
4. Name the playground "Hello, world!"
5. Click Create to save the playground
6. Add `print("Hello, world!”)`
7. Replace `"Hello, world!"` with `str`

Unit 1 — Lesson 1

Open and complete the exercises in `Lab - Introduction.playground`

Unit 1 — Lesson 2: Constants, Variables, and Data Types

Constants and variables

Associate a name with a value

Defining a constant or variable

- Allocates storage for the value in memory
- Associate the constant name with the assigned value

Constants

Defined using the `let` keyword

```
let name = "John"
```


Defined using the let keyword

```
let pi = 3.14159
```

Can’t assign a constant a new value

```
let name = "John"
name = "James"
```

Variables

Defined using the `var` keyword

```
var age = 29
```

Can assign a new value to a variable

```
var age = 29

age = 30
```

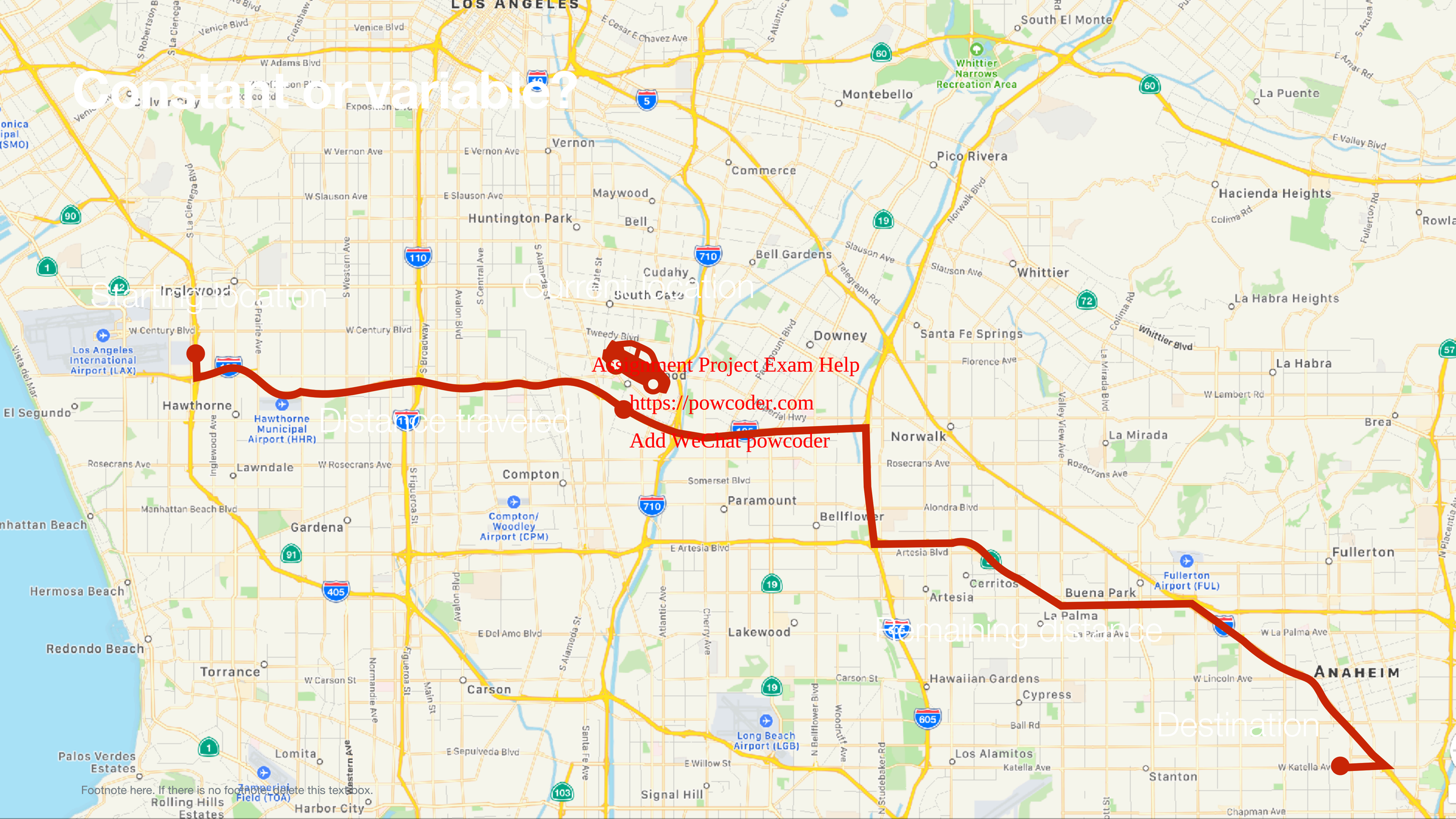
```swift
let defaultScore = 100
var playerOneScore = defaultScore
var playerTwoScore = defaultScore

print(playerOneScore)
print(playerTwoScore)

playerOneScore = 200
print(playerOneScore)
```



```
100
100
200
```


Constant or variable?
Starting location
Current location
Distance traveled
Remaining distance
Destination
Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder
Footnote here. If there is no footnote, delete this textbox.

Naming constants and variables

Rules

No mathematical symbols

No spaces

Can’t begin with a number


```
let π = 3.14159
let 一百 = 100
let 🎲 = 6
let mañana = "Tomorrow"
let anzahlDerBücher = 15 //numberOfBooks
```

Naming constants and variables

Best practices

1. Be clear and descriptive

   ✗ `n` ✓ `firstName`

2. Use camel case when multiple words in a name

   ✗ `firstname` ✓ `firstName`

Comments

```
// Setting pi to a rough estimate
let π = 3.14
```

```
/* The digits of pi are infinite,
so instead I chose a close approximation.*/
let π = 3.14
```

Types

```
struct Person {
  let firstName: String
  let lastName: String

  func sayHello() {
    print("Hello there! My name is \(firstName) \(lastName).")
  }
}
```

```
struct Person {
  let firstName: String
  let lastName: String

  func sayHello() {
    print("Hello there! My name is \(firstName) \(lastName).")
  }
}

let aPerson = Person(firstName: "Jacob", lastName: "Edwards")
let anotherPerson = Person(firstName: "Candace", lastName: "Salinas")

aPerson.sayHello()
anotherPerson.sayHello()
```



```
Hello there! My name is Jacob Edwards.
Hello there! My name is Candace Salinas.
```

Most common types

	Symbol	Purpose	Example
Integer	`Int`	Represents whole numbers	`4`
Double	`Double`	Represents numbers requiring decimal points	`13.45`
Boolean	`Bool`	Represents true or false values	`true`
String	`String`	Represents text	`"Once upon a time…"`

Type safety

```
let playerName = "Julian"
var playerScore = 1000
var gameOver = false
playerScore = playerName
```


Cannot assign value of type ‘String’ to type ‘Int’

```
var wholeNumber = 30
var numberWithDecimals = 17.5
wholeNumber = numberWithDecimals
```

Cannot assign value of type ‘Double’ to type ‘Int’

Type inference

```
let cityName = "San Francisco"
let pi = 3.1415927
```

Type annotation

```
let cityName: String = "San Francisco"
let pi: Double = 3.1415927
```



```
let number: Double = 3
print(number)
```

```
3.0
```

Type annotation

Three common cases

1. When you create a constant or variable before assigning it a value

```
let firstName: String
//...
firstName = "Layne"
```

Type annotation

Three common cases

2. When you create a constant or variable that could be inferred as two or more different types

```
let middleInitial: Character = "J"
var remainingDistance: Float = 30
```

Type annotation

Three common cases

3. When you add properties to a type definition

```
struct Car {
    let make: String
    let model: String
    let year: Int
}
```

Required values

```
var x
```

Required values

```
var x: Int
```

Required values

```
var x: Int
print(x)
```

Variable 'x' used before being initialized

Required values

```
var x: Int
x = 10
print(x)
```



```
10
```

Numeric literal formatting

```
var largeUglyNumber = 1000000000
var largePrettyNumber = 1_000_000_000
```

Unit 1 — Lesson 2

Lab: Constants and Variables.playground

Open and complete the exercises in `Lab - Constants and Variables.playground`

Unit 1 — Lesson 3: Operators

Assign a value

Use the = operator to assign a value

```
var favoritePerson = "Luke"
```


Use the = operator to modify or reassign a value

```
var shoeSize = 8
shoeSize = 9
```

Basic arithmetic

You can use the `+`, `−`, `*`, and `/` operators to perform basic math functions

```
var opponentScore = 3 * 8
var myScore = 100 / 4
```


You can also use the value of other variables

```
var totalScore = opponentScore + myScore
```

Or you can use the current variable you're updating

```
myScore = myScore + 3
```

Basic arithmetic

Use Double values for decimal point precision

```
let totalDistance = 3.9
var distanceTravelled = 1.2
var remainingDistance = totalDistance - distanceTravelled
print(remainingDistance)
```



```
2.7
```

Basic arithmetic

```
let x = 51
let y = 4
let z = x / y
print(z)
```



```
12
```

Basic arithmetic

Using Double values

```
let x: Double = 51
let y: Double = 4
let z = x / y
print(z)
```



```
12.75
```

Compound assignment

```
var myScore = 10
myScore = myScore + 3
```

```
myScore += 3
myScore -= 5
myScore *= 2
myScore /= 2
```

Order of operations

1. `(` `)`
2. `*` `/`
3. `+` `-`


```
var x = 2
var y = 3
var z = 5
print(x + y * z)
print((x + y) * z)
```

```
17
25
```

Numeric type conversion

```
let x = 3
let y = 0.1415927
let pi = x + y
```

Binary operator ‘+’ cannot be applied to operands of type ‘Int’ and ‘Double’

Numeric type conversion

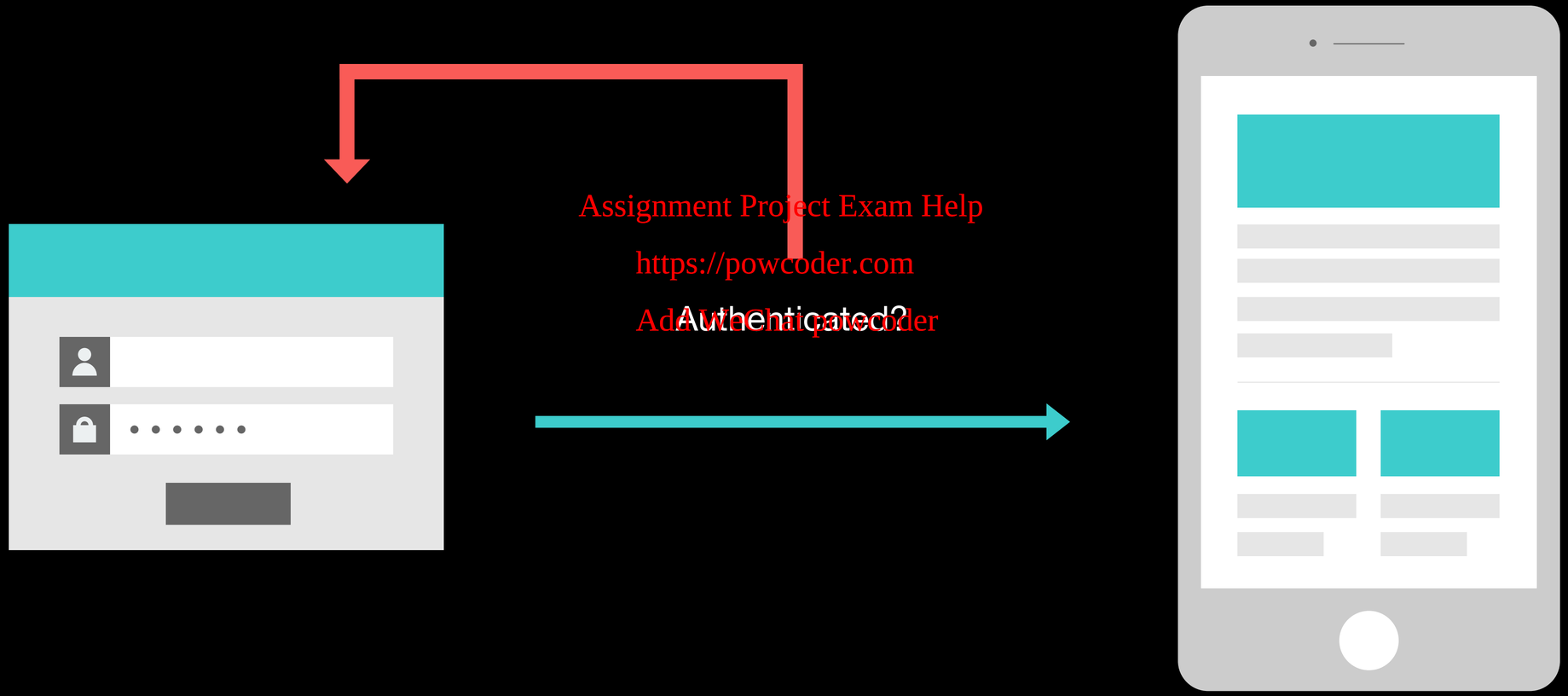
```
let x = 3
let y = 0.1415927
let pi = Double(x) + y
```

Unit 1 — Lesson 3
Lab: Operators

Open and complete the exercises in `Lab-Operators.playground`

Unit 1 — Lesson 4: Control Flow

Conditional flow



Logical operators

Operator	Description
`==`	Two items must be equal
`!=`	The values must not be equal to each other
`>`	Value on the left must be greater than the value on the right
`>=`	Value on the left must be greater than or equal to the value on the right
`<`	Value on the left must be less than the value on the right
`<=`	Value on the left must be less than or equal to the value on the right
`&&`	AND—The conditional statement on the left and right must be true
`||`	OR—The conditional statement on the left or right must be true
`!`	Returns the opposite of the conditional statement immediately following the operator

if statements

```
if condition {
    code
}
```



```
let temperature = 100
if temperature >= 100 {
  print("The water is boiling.")
}
```

```
The water is boiling
```

if-else statements

```
if condition {
    code
} else {
    code
}
```



```
let temperature = 100
if temperature >= 100 {
  print("The water is boiling.")
} else {
  print("The water is not boiling.")
}
```

Boolean values

```
let number = 1000
let isSmallNumber = number < 10
```

```
let speedLimit = 65
let currentSpeed = 72
let isSpeeding = currentSpeed > speedLimit
```

Boolean values

NOT

```
var isSnowing = false
if !isSnowing {
  print("It is not snowing.")
}
```



```
It is not snowing.
```

Boolean values

AND

```
let temperature = 70
if temperature >= 65 && temperature <= 75 {
  print("The temperature is just right.")
} else if temperature < 65 {
  print("It's too cold.")
} else {
  print("It's too hot.")
}
```



```
The temperature is just right.
```

Boolean values

OR

```
var isPluggedIn = false
var hasBatteryPower = true
if isPluggedIn || hasBatteryPower {
  print("You can use your laptop.")
} else {
  print("😱")
}
```

switch statement

```
switch value {
case n:
   code
case n:
   code
case n:
   code
default:
   code
}
```

```
let numberOfWheels = 2
switch numberOfWheels {
case 1:
    print("Unicycle")
case 2:
    print("Bicycle")
case 3:
    print("Tricycle")
case 4:
    print("Quadcycle")
default:
    print("That's a lot of wheels!")
}
```

switch statement

Multiple conditions

```
let character = "z"

switch character {
case "a", "e", "i", "o", "u" :
    print("This character is a vowel.")
default:
    print("This character is not a vowel.")
}
```

switch statement

Ranges

```
switch distance {
case 0...9:
    print("Your destination is close.")
case 10...99:
    print("Your destination is a medium distance from here.")
case 100...999:
    print("Your destination is far from here.")
default:
    print("Are you sure you want to travel this far?")
}
```

switch challenge

Rewrite the following using a switch statement:

```
let temperature = 70
if temperature >= 65 && temperature <= 75 {
  print("The temperature is just right.")
} else if temperature < 65 {
  print("It's too cold.")
} else {
  print("It's too hot.")
}
```


Hint: The smallest possible value for an integer is `Int.min`

switch challenge

Solution

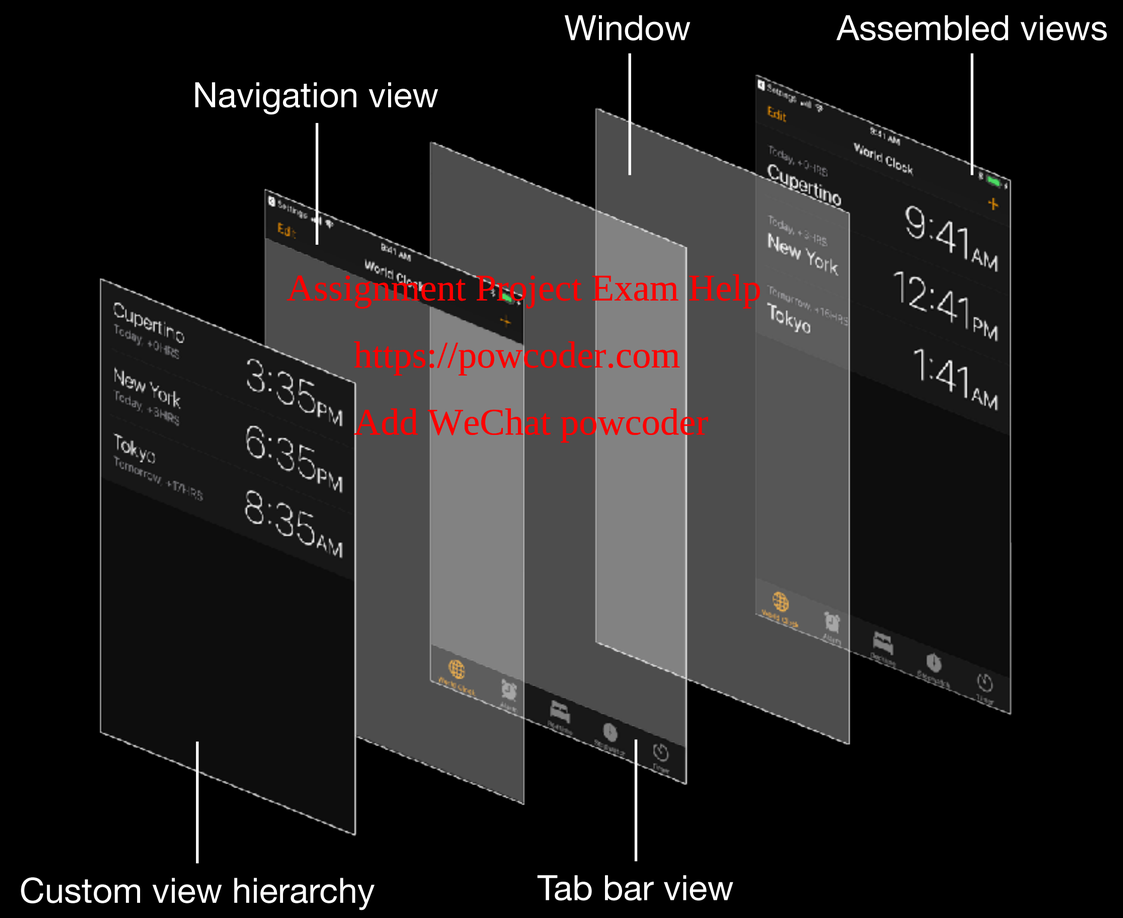
```
let temperature = 76
switch temperature {
case Int.min...64:
    print("It's too cold.")
case 65...75:
    print("The temperature is just right.")
default:
    print("It's too hot.")
}
```

Unit 1 — Lesson 4
Lab: Control Flow

Open and complete the exercises in `Lab - Control Flow.playground`

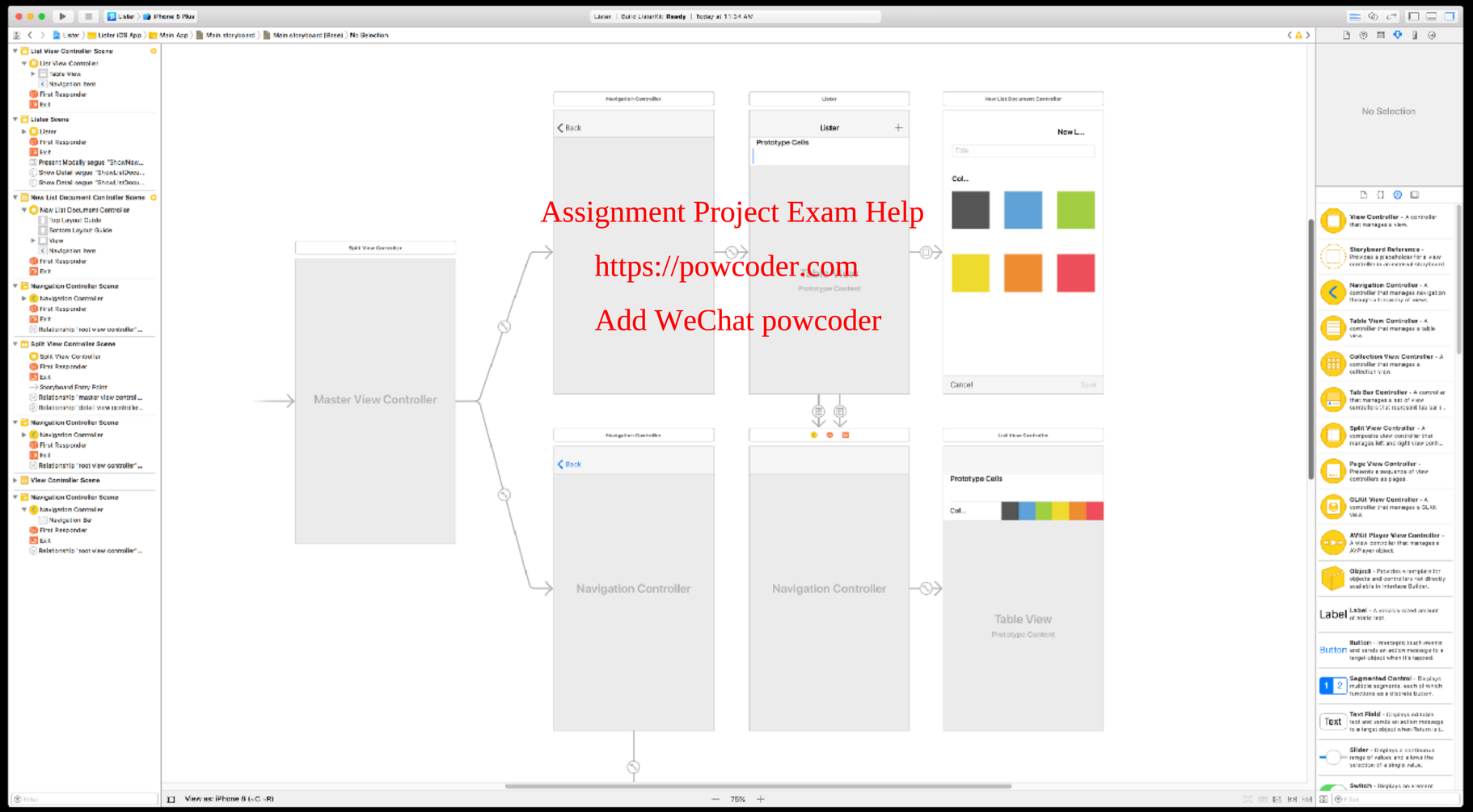
Unit 1 — Lesson 8: Interface Builder Basics

Common system views



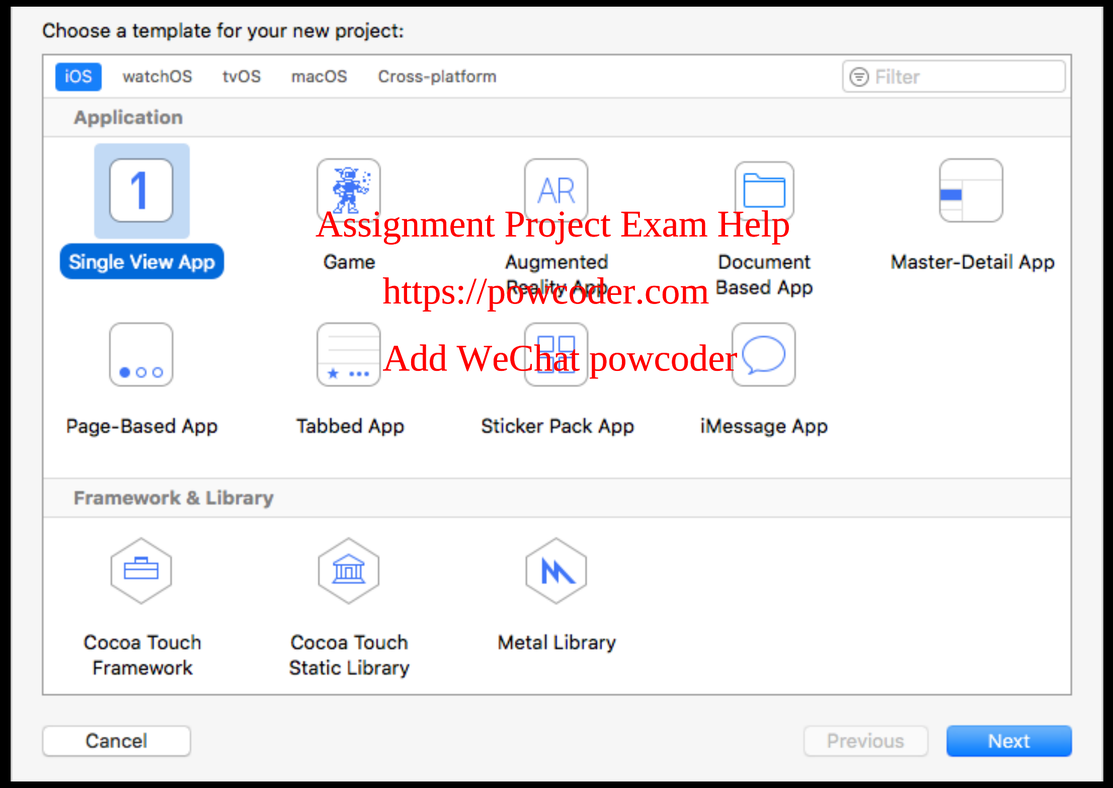
Interface Builder

Storyboards



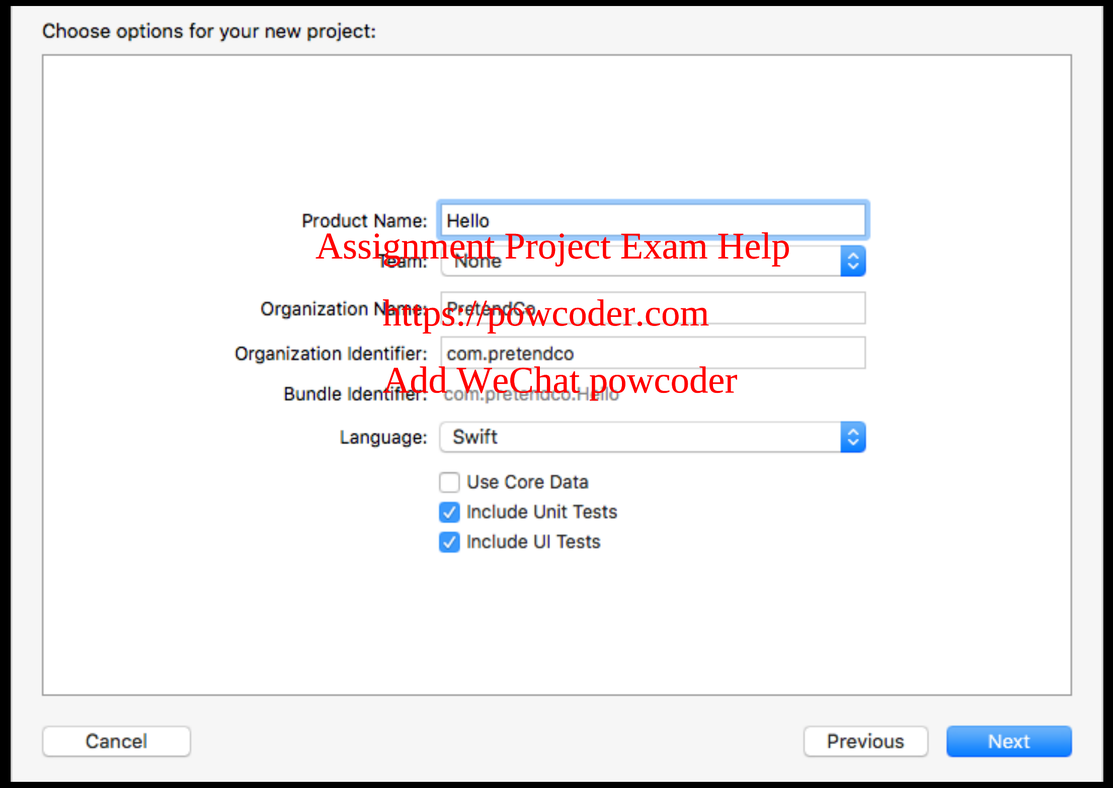
Hello

Create a new project



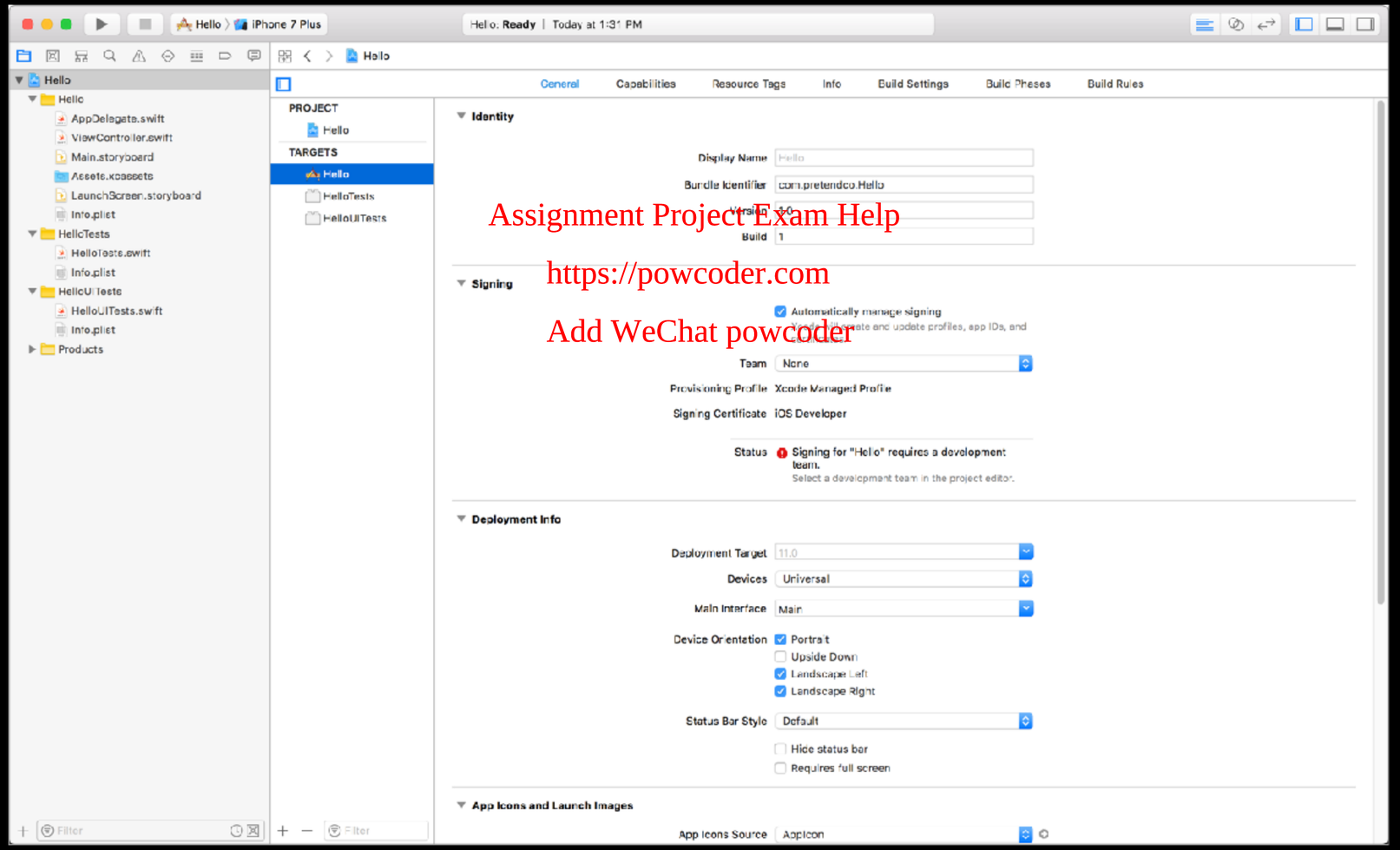
Hello

Project options

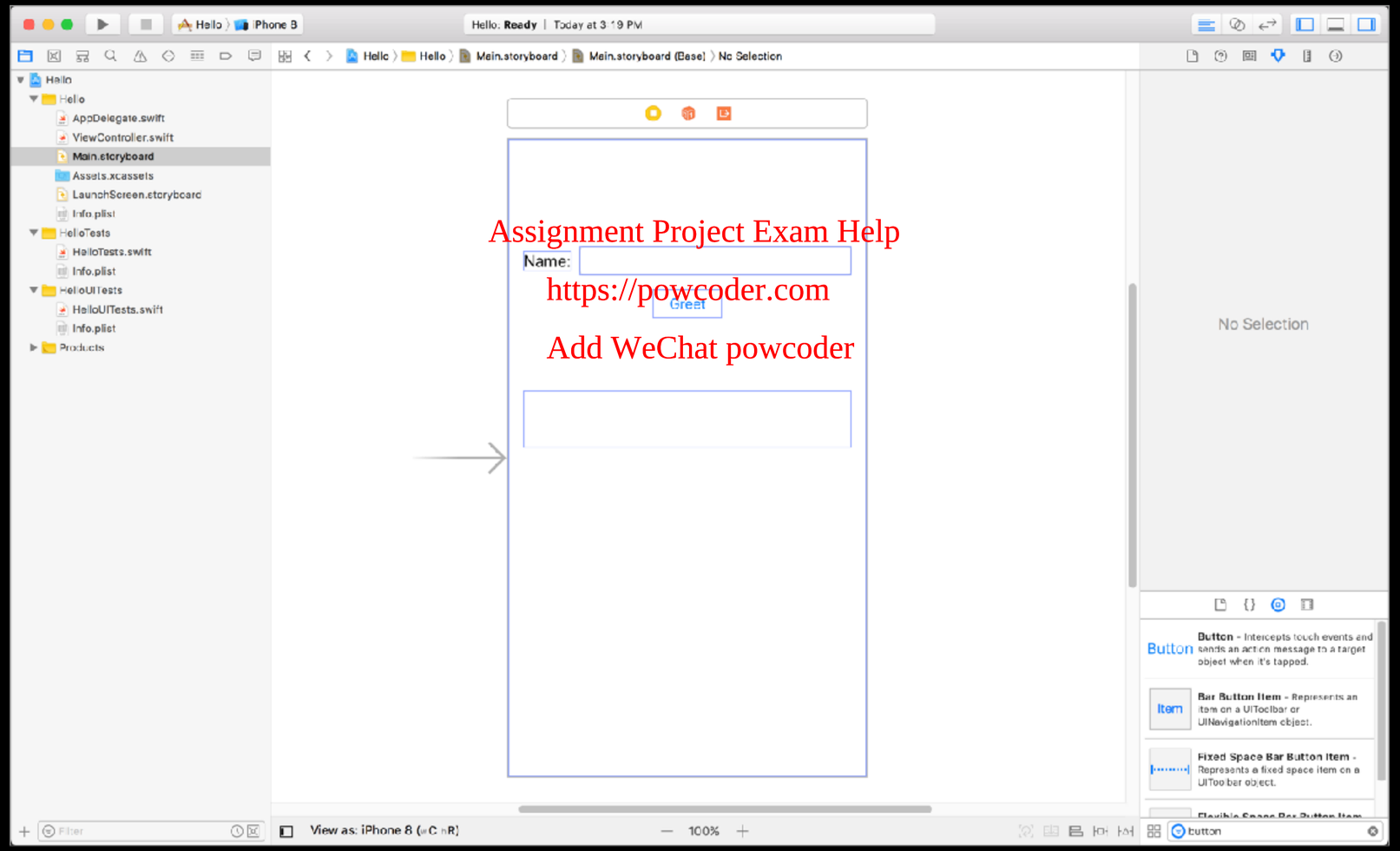


Hello

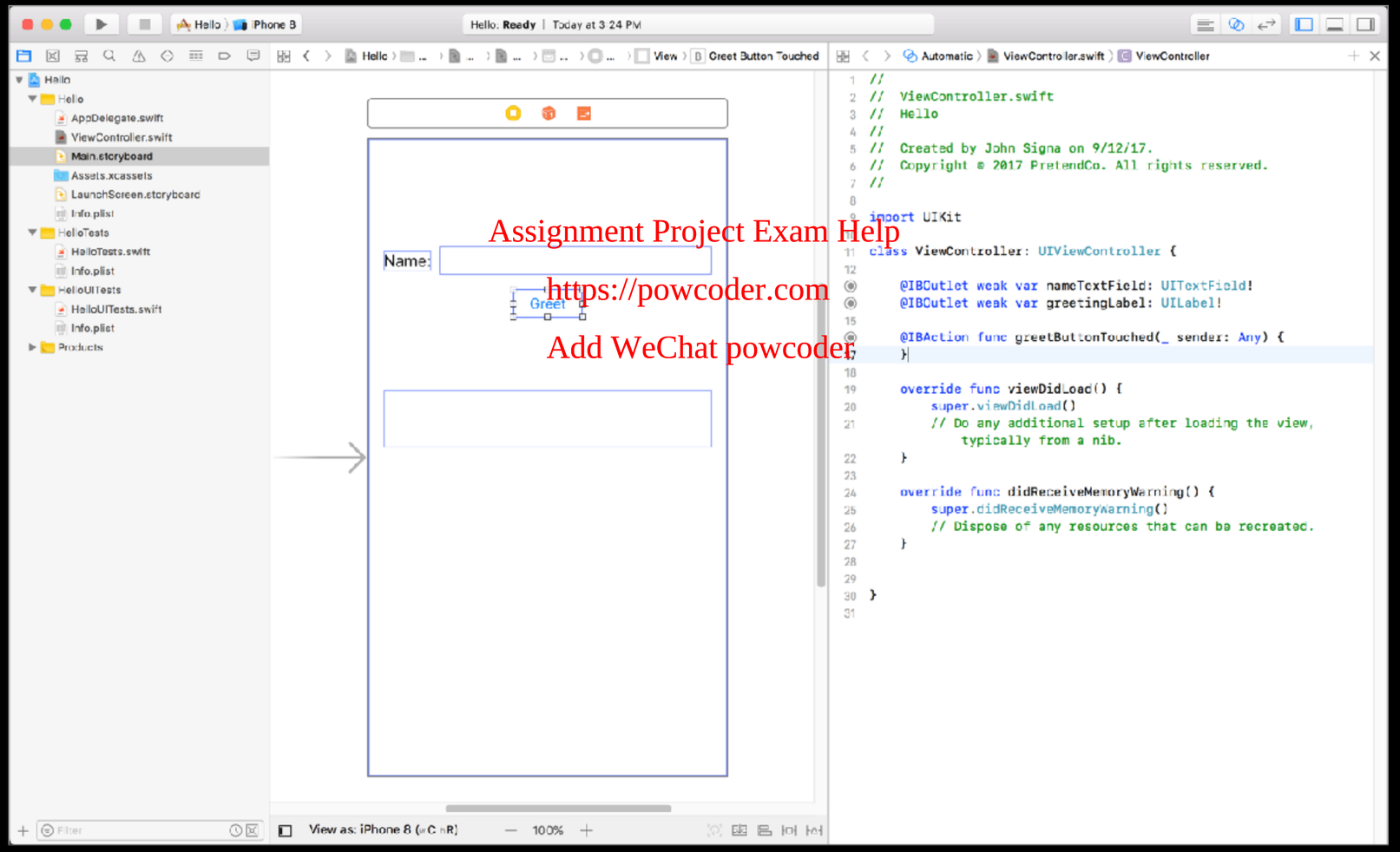
Default project



Hello


Assignment Project Exam Help
Name:
https://powcoder.com
Add WeChat powcoder
No Selection

Hello



Hello

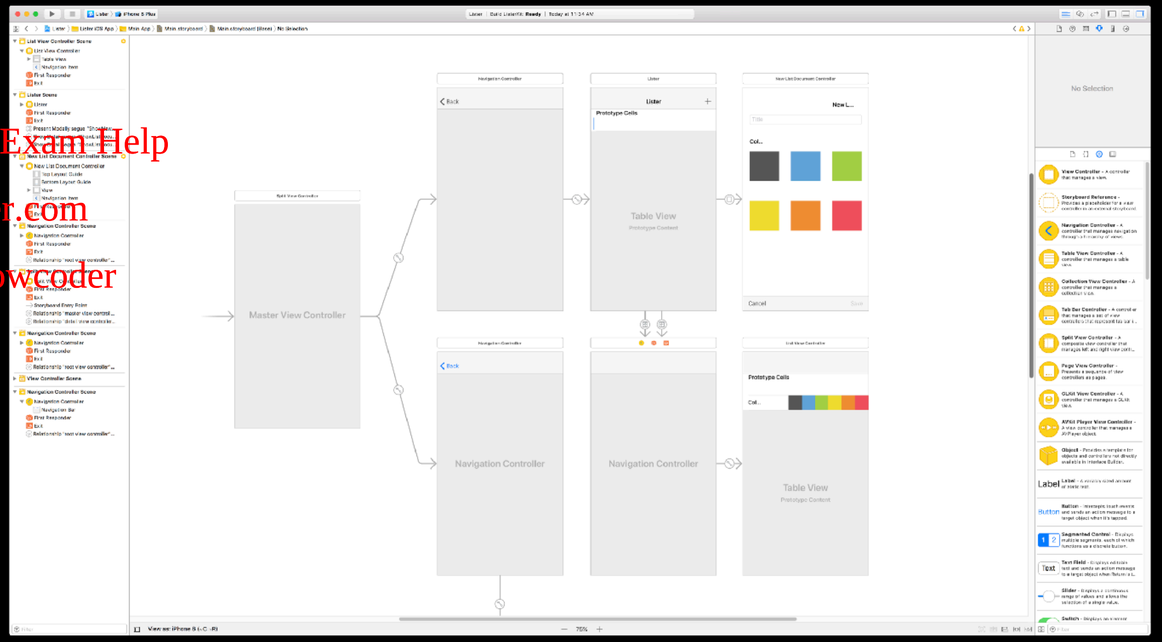
Complete the greet function

```
@IBAction func greetButtonTouched(_ sender: Any) {
    greetingLabel.text = "Hello, " + nameTextField.text!
}
```

Unit 1 — Lesson 8

Interface Builder Basics

Learn how to navigate through Interface Builder, add elements onto the canvas, and interact with those elements in code.



Unit 1—Lesson 8

Lab: Use Interface Builder

1. Create an Xcode project
2. Create a simple view with Interface Builder
3. Use the Assistant Editor to connect your view