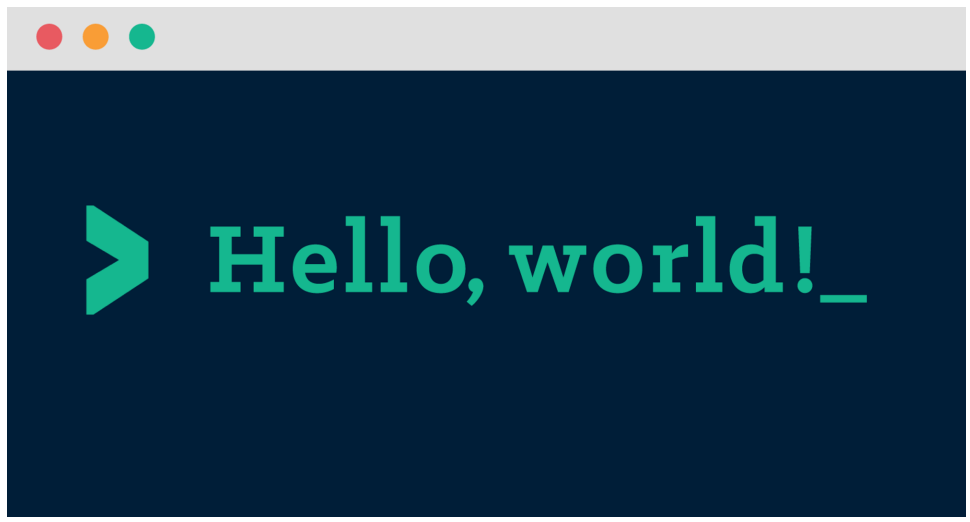
CHAPTER FOUR : USER INTERFACE OF THE HELLO WORLD APP



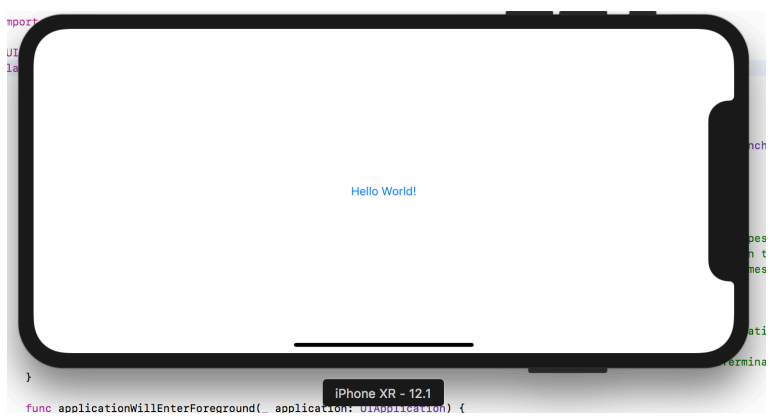
In this chapter, we will modify our user interface and make it functional. With the knowledge of the project structure and the file organization from Chapter Three, we are going to modify the Main.storyboard and ViewController.swift in order to achieve the specifications.

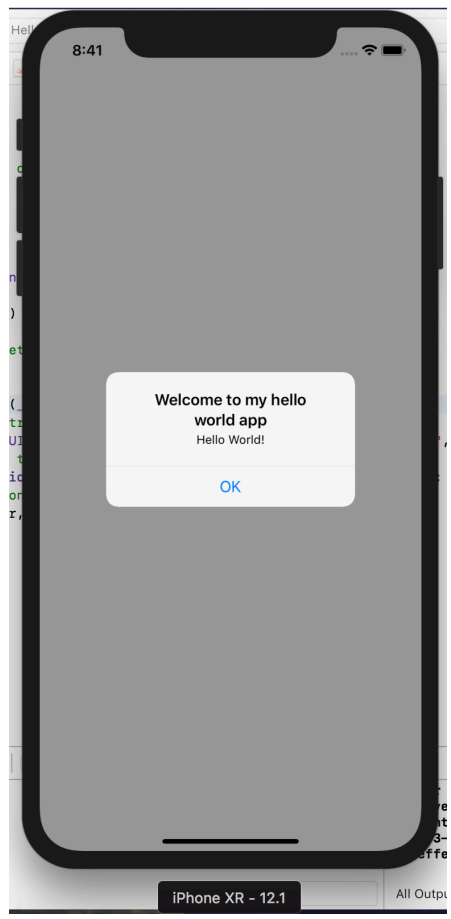
I. OVERVIEW OF THE FUNCTION

In this chapter, the feature we need to add is a button that displays a “Hello World”message box. Before moving on to the next section, try to think about how can you solve this problem on your own. The natural thinking process is:

- Add a button to the UI
- Design the MessageBox that contains our “Hello World!” message
- “Tell” the app to display MessageBox when a button is clicked.

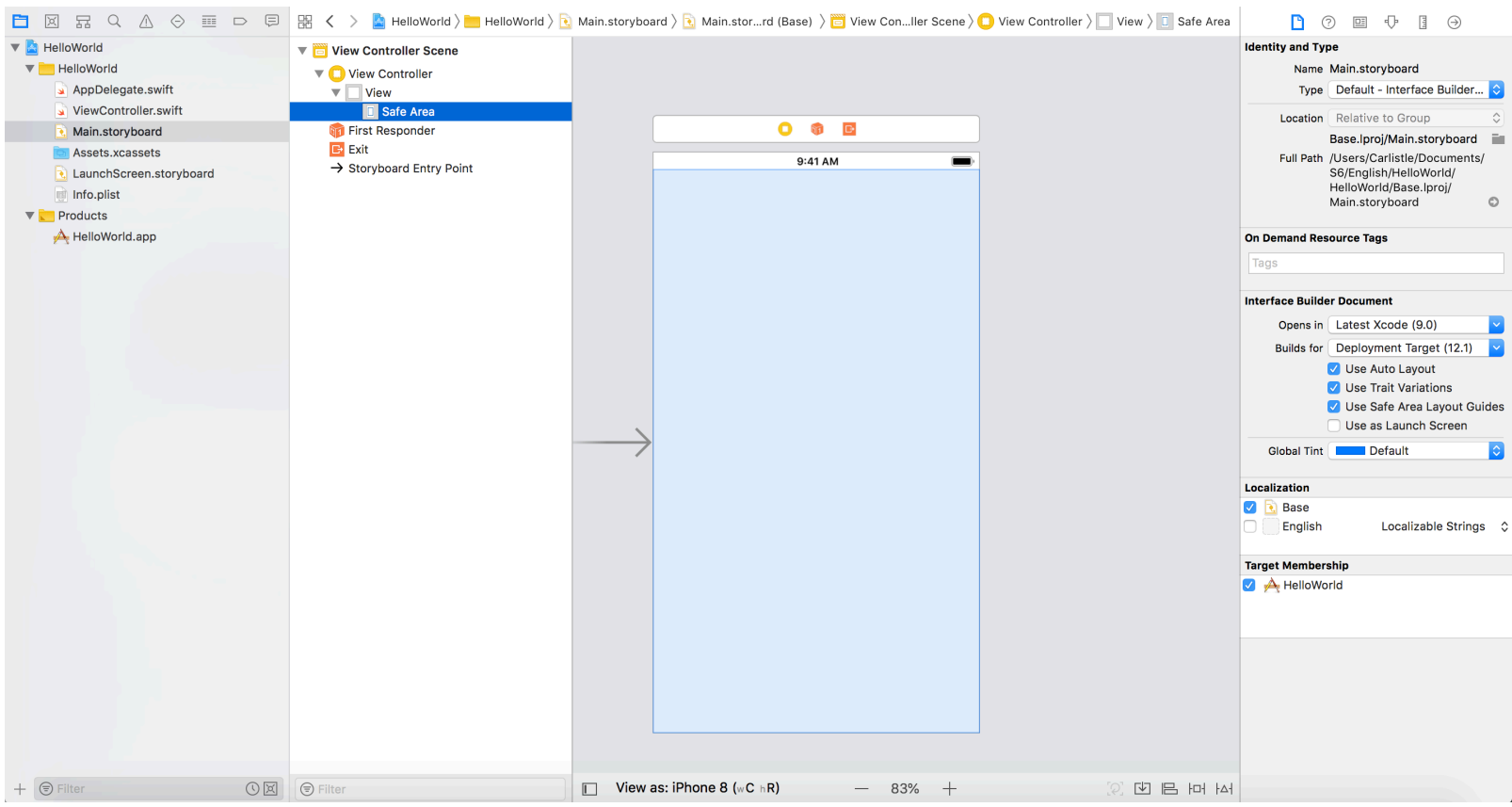
These are exactly the steps to take to add a feature to our UI. And we almost always follow this procedure from now on.



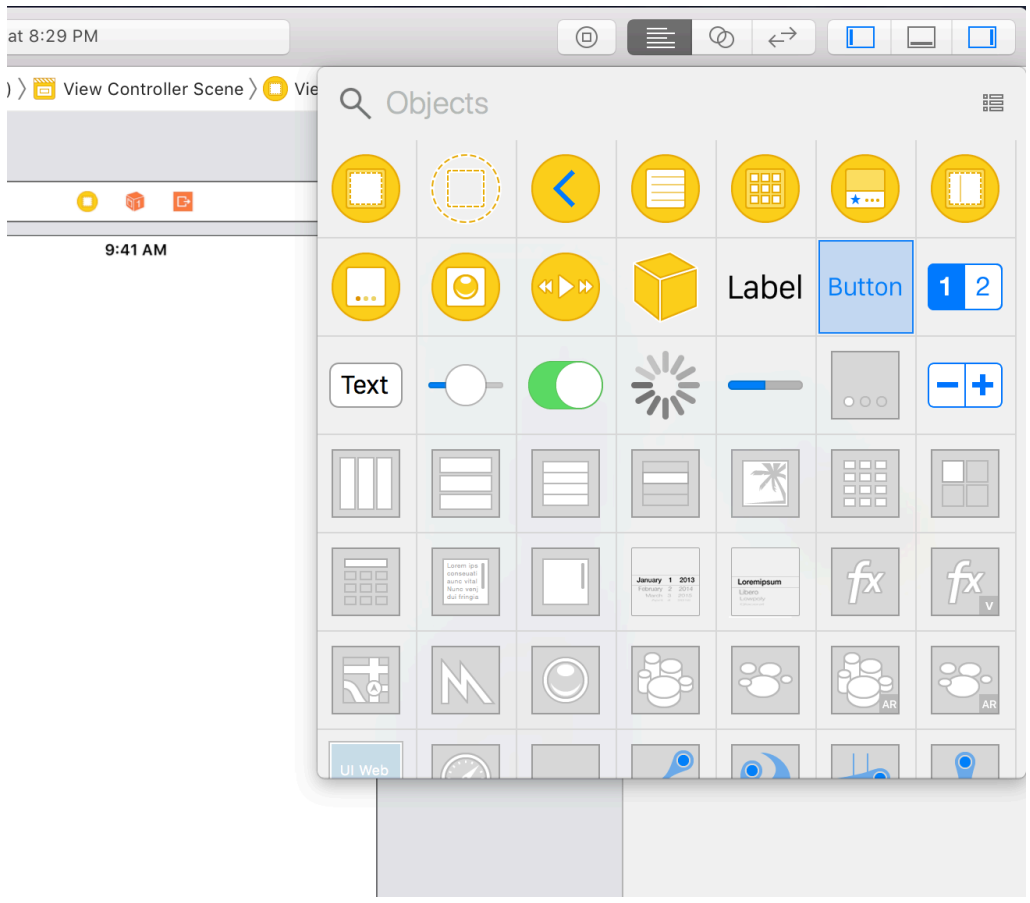


II. MODIFY MAIN.STORYBOARD

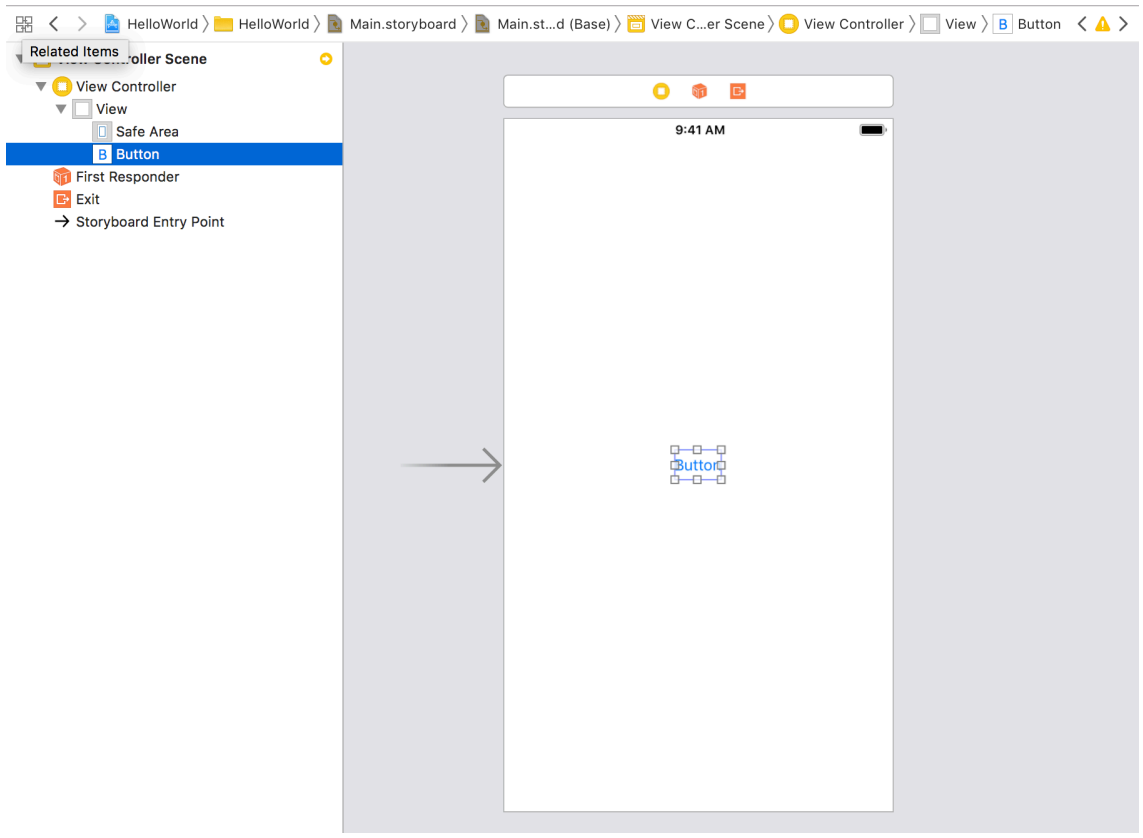
In the **Navigator**(use cmd+0 to toggle if you don't see it), select `HelloWorld> HelloWorld>Main.storyboard` in the project tree. In **Document Outline**, we need to put a button in the safe area.



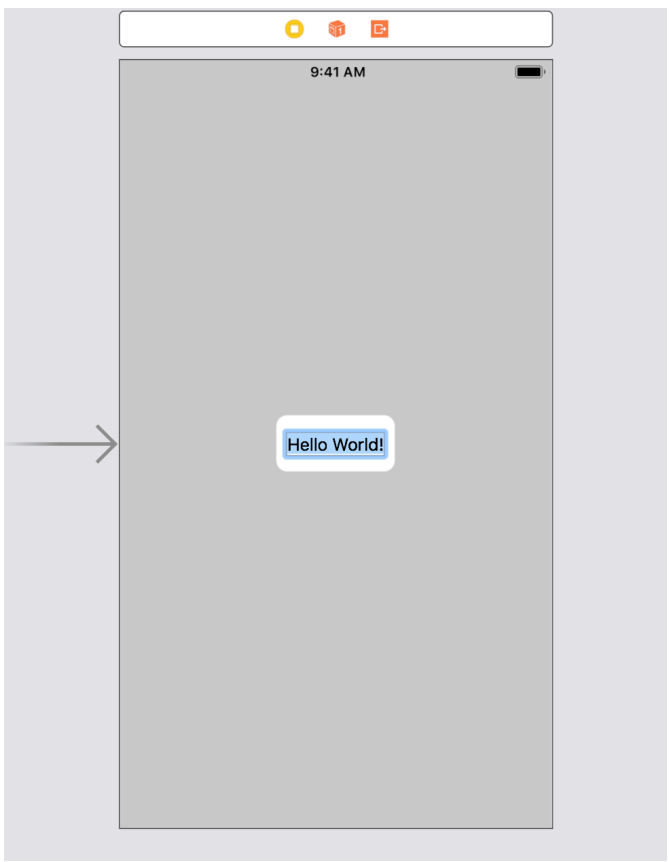
On the upper right corner, click **library** and choose **button** from the object library. Drag it to the middle of the safe area.

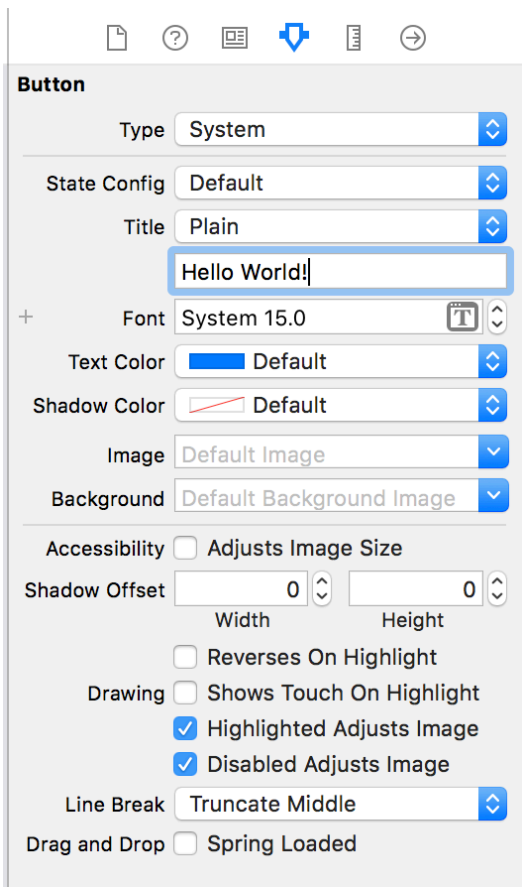


In the document outline, the button should be **inside the view** like this. If you have trouble dragging the item in to the correct container in the storyboard, you can drag it directly to the document outline.



Now we have a button that says `Button`, next we are going to change the text to `Hello World!`. Double click the text and type `Hello World!` in the text box. Or go to **Attributes Inspector>Title** and change the text.

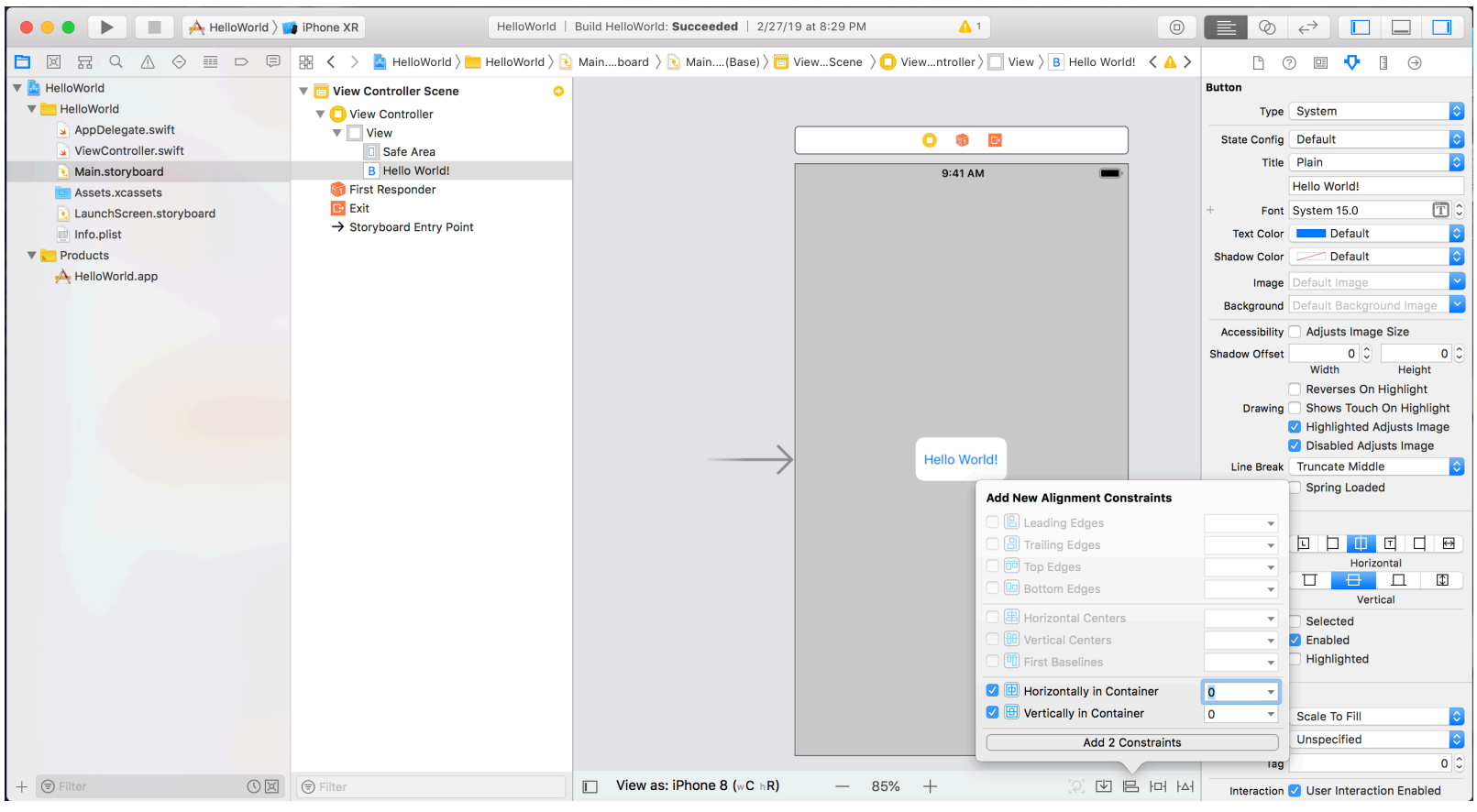




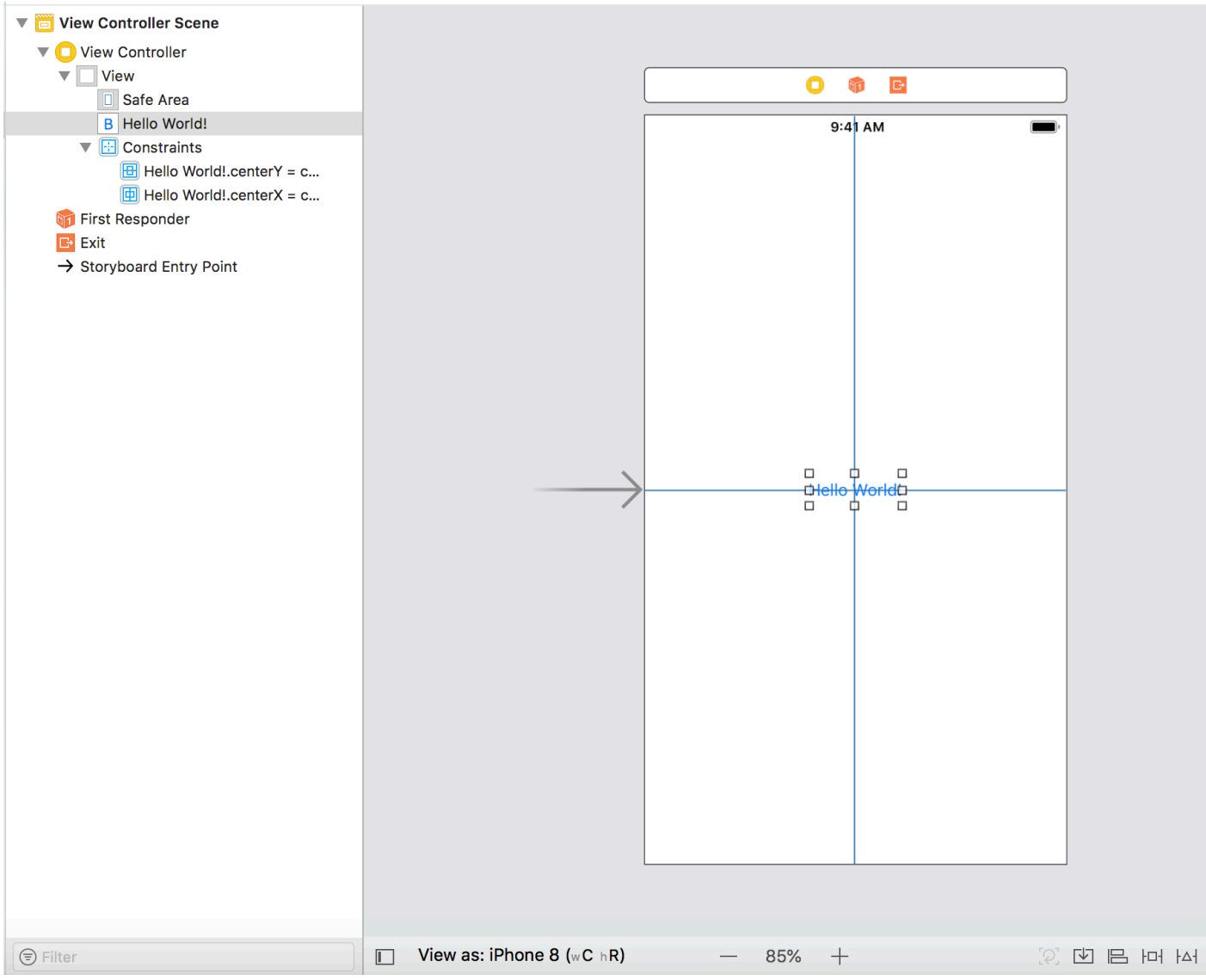
Layout and constraints

If you have tried the Exercise 2 of the last chapter, you might have noticed that there are many sizes and many resolutions for different iOS devices. This is the reason why we need to "settle" our layout correctly in our storyboard. Here is just a preview of layout and we will work on auto layout in later chapters.

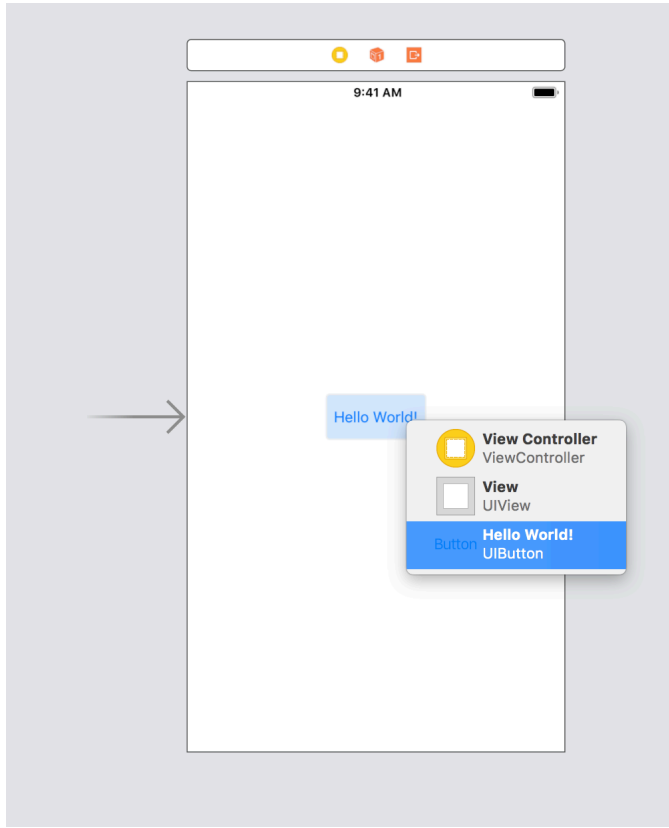
Choose `Hello World` button in the outline or click the button object in the storyboard. In the bottom bar of the storyboard, click `Align`. Choose `Horizontally in Container` and `Vertically in Container`. Finally, confirm the change by clicking `Add 2 Constraints`.



After adding these constraints, you will see that the Hello World! button has blue reference lines with respect to the it's superview. You can also verify these constraints in **Document Outline**. There are two new constraints in View, if this is not the case, delete your constraints and redo from the start of this section.

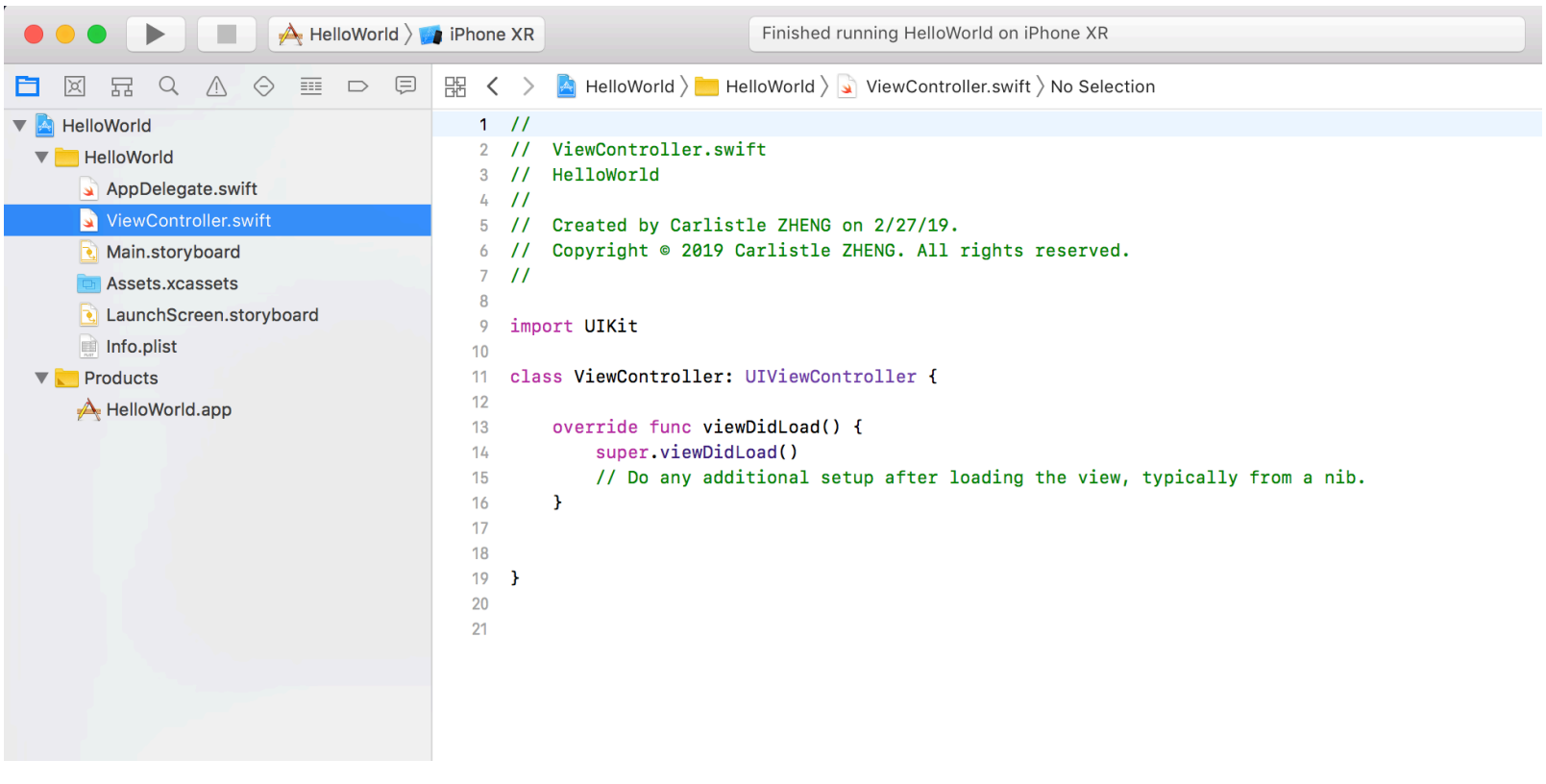


If you have trouble selecting the object you want, hover your mouse over the object. Press **shift** and **right click**, you will see a list of the overlapping objects and you can choose the object you want.

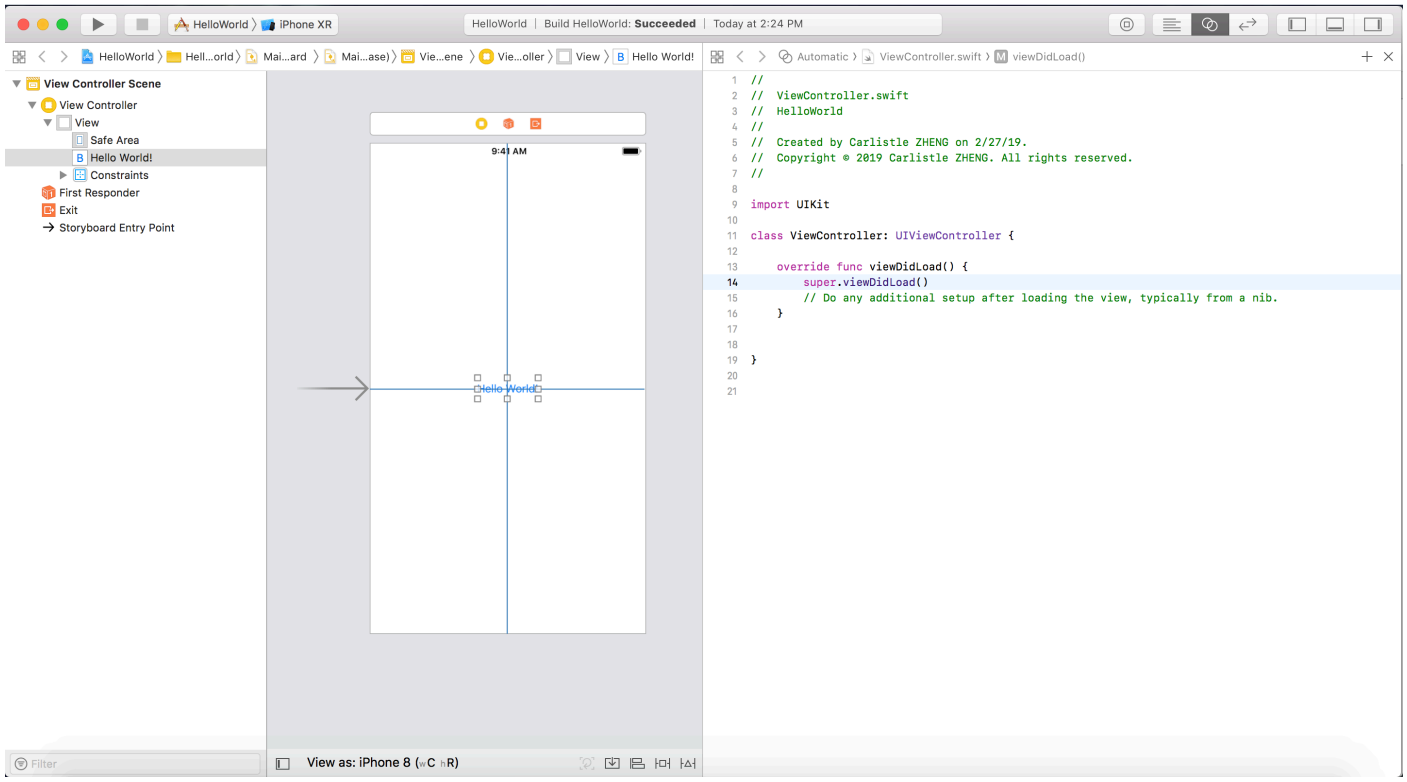


III. LINK THE BUTTON TO VIEWCONTROLLER

In the Chapter Three, we learnt about different components of an iOS app, and one of the most important components is view controllers. In this Hello World app, the controller controls what happens when the button is clicked. In **Navigator**, choose `ViewController.swift` and you will see the following default code.



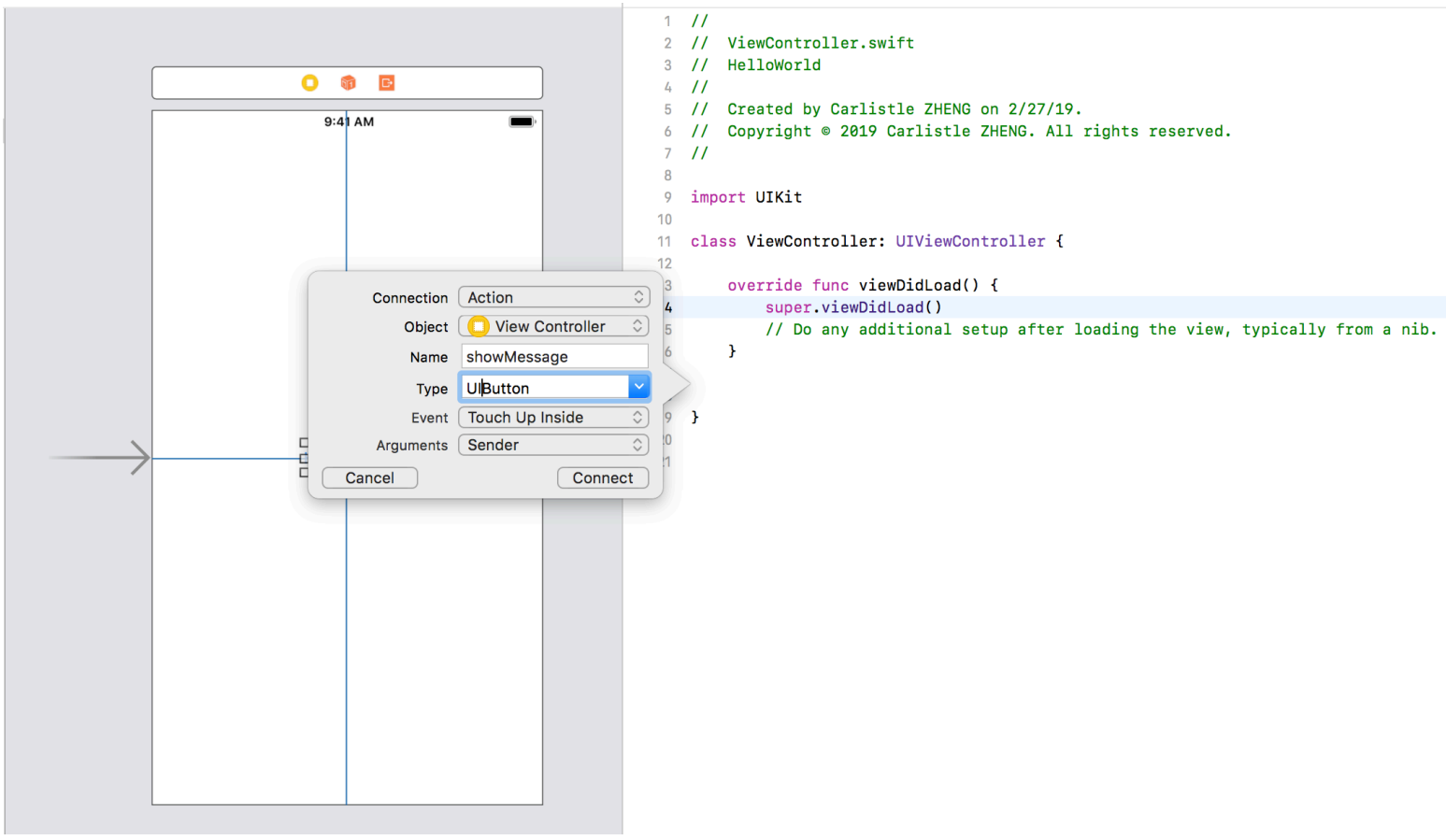
Click **Show Assistant Editor** in the menu bar or use shortcut **option+cmd+return**. Then choose `Main.storyboard` on the left and `ViewController.swift` on the right. Your Xcode window will look like this.



This layout will make editing two files at the same time a lot easier. If you have a small display, close the Navigator and the Inspector(**cmd+0** and **cmd+option+0**) to make enough space for the main editing area. As you might have imagined, we need to make a connection between Main.storyboard and the ViewController.swift.

Control Drag

Press command and drag from the button in the storyboard to line 18 of the view controller (or anywhere inside the `ViewController` class, outside the method `ViewController`). This is called **Control Drag** and you will use this feature quite often. Add the name `showMessage` and select the type **UIButton**. This method(showMessage) is called with the argument UIButton when the `Hello World!` button in the storyboard is touched up inside.



Then add the following code to the method we just created.

```
    @IBAction func showMessage(_ sender: UIButton) {
        // Create an alert controller object to display an alert
        let alertController = UIAlertController(title: "Welcome to my hello world app", message:
"Hello World!", preferredStyle: .alert)
        // Add an action OK to the alert controller
        alertController.addAction(UIAlertAction(title: "OK", style: .default, handler: nil))
        // Present the alert controller with animation
        present(alertController, animated: true, completion: nil)
    }
```

If you don't understand what this code does, don't worry. We will go through this in the next chapter. If it really troubles you, here's a short version of explanation:

- Line 19, 21, 23:

```
// Create an alert controller object to display an alert

// Add an action OK to the alert controller

// Present the alert controller with animation
```

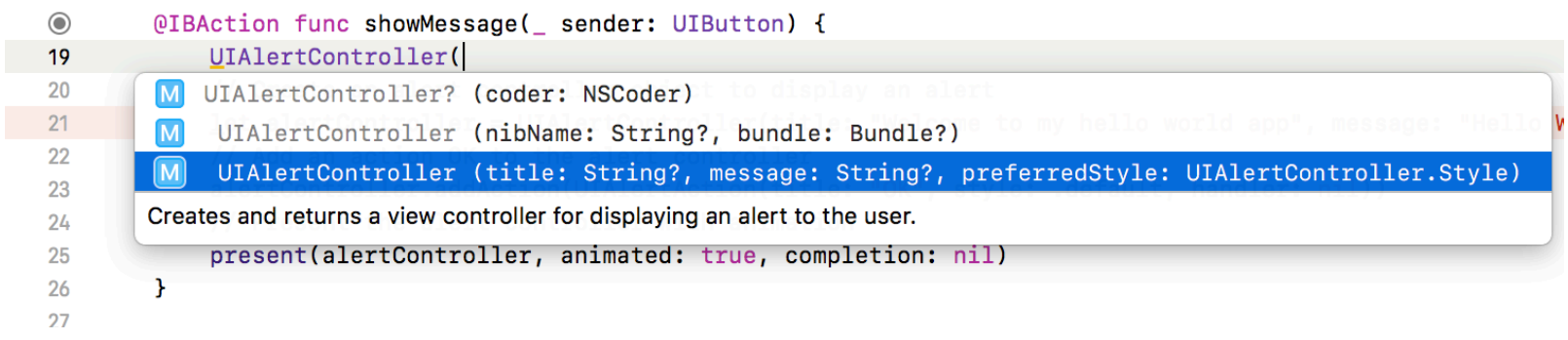
line comment in swift starts with two forward slashes `//` Writing comments in your code is an awesome habit and you will have a clearer mind while reading your own code and others can understand your code easily by reading your comment.

- Line 20:

```
    let alertController = UIAlertController(title: "Welcome to my hello world app", message: "Hello World!", preferredStyle: .alert)
```

creates an UIAlertController that displays the title "Welcome to my hello world app" and the message "Hello World!". If you have experience with Object Oriented Programming(OOP) languages like Java, Python, and C++, is is the special method of a class called a **Constructor**. A constructor creates a new instance of its class, in this case, a UI alert controller.

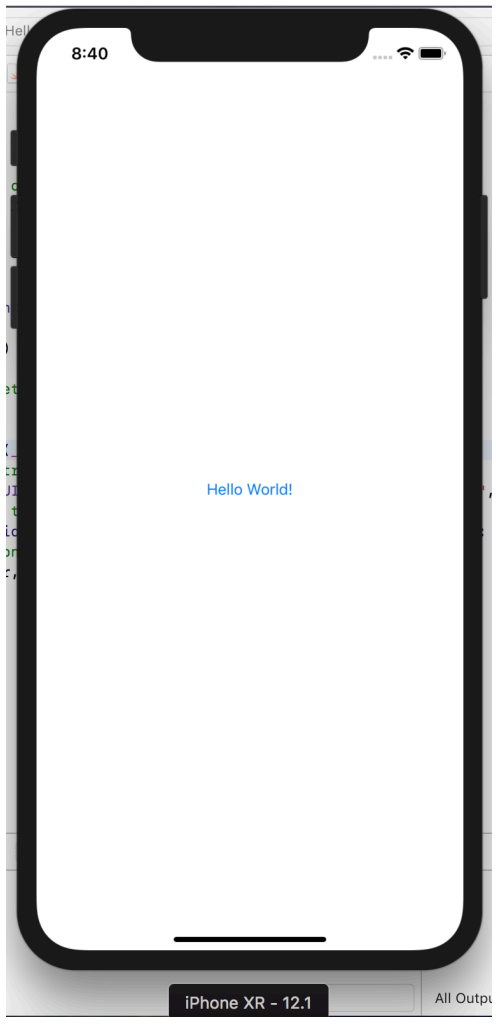
There are three constructors of the class UIAlertController and here we used the last constructor of the class UIAlertController. The auto completion of Xcode will indicate all the options.

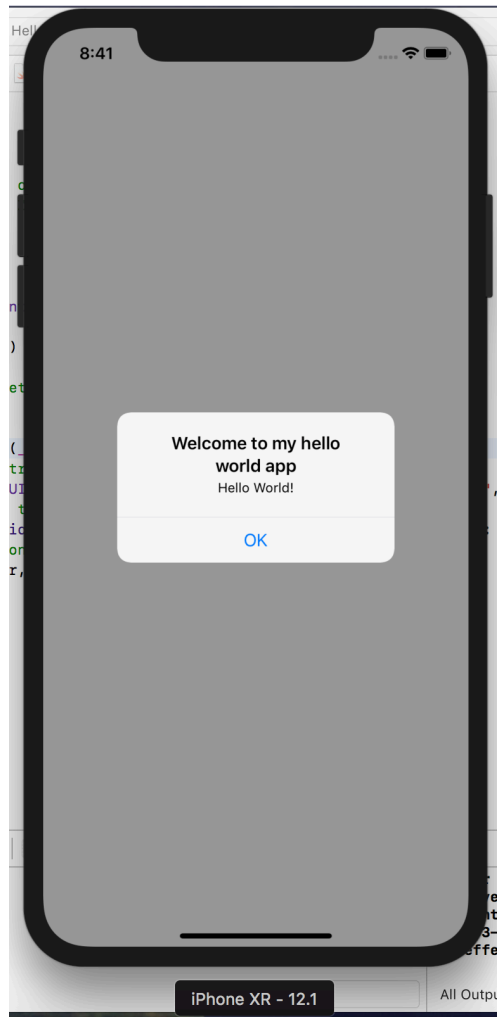


When your project becomes more and more complex, you will use autocompletion all the time. Using autocompletion also helps you to have consistent parameters and to avoid typos.

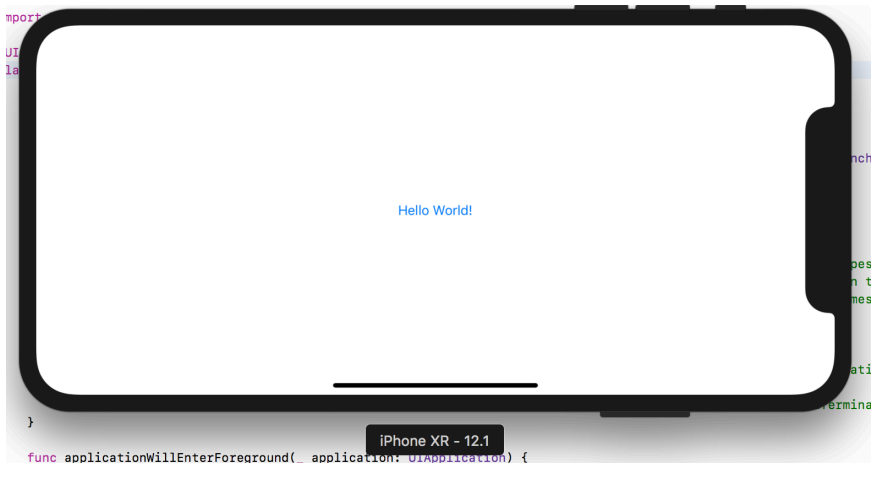
IV. TEST THE MODIFICATIONS

Save all your modified files and click run to test it out in a simulator or a real iOS device. I'm running in an iPhone XR simulator and here is how it looks when I click the button.





If you have some doubts on why we added constraints to the hello world button in the previous section, choose **Device>Rotate Device.** If the constraints work fine, the button should be centered in all orientations like this:



EXERCISE #1

Try to change the style of the button to something like this:

Hints:

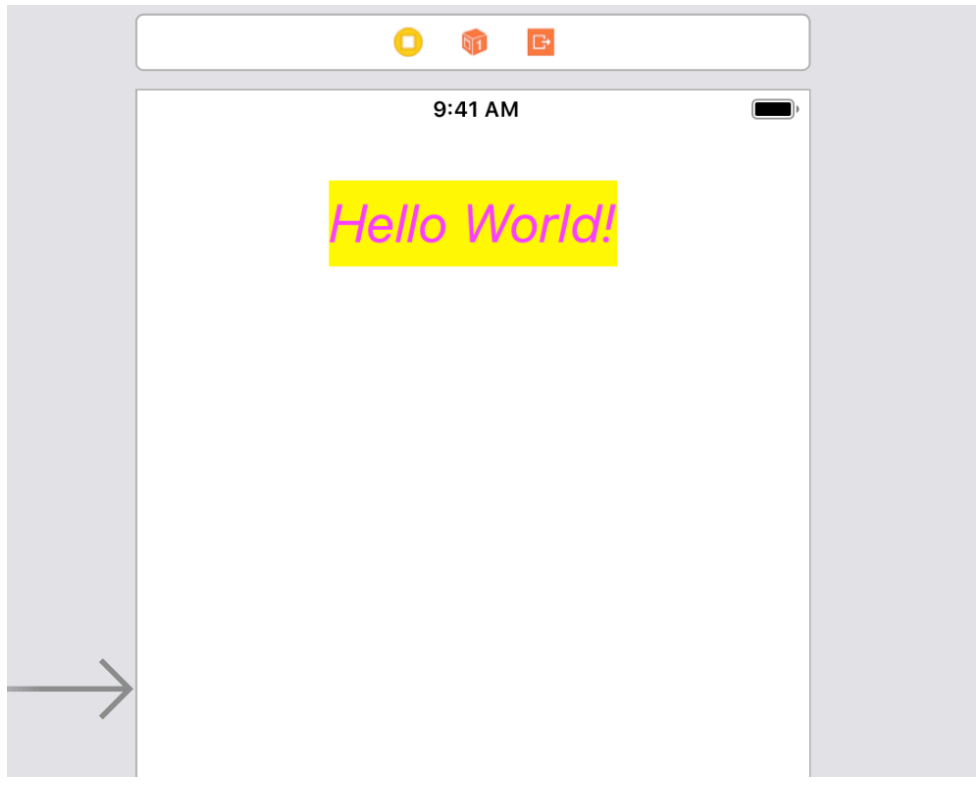
Button: Text color: pink; font: System Italic 30.0

View: background: yellow

EXERCISE #2

Change the position of the button to:

> **Horizontally centered and vertically (top) 50 points to the top of its superview**



Hint: If you read the instruction carefully, you will notice that you only need to modify one constraint: **HelloWorld!.centerX = Superview.centerX.**

Now let's transform the requirement into a more mathematic way:

HelloWorld!.top = Superview.top + 50

Go to the Attributes Inspector and make the changes to the old vertical constraints. Test out your changes in different devices and different orientations.

SUMMARY

In this chapter, we have worked on these topics:

- translated user requirements into technical specifications
- added UI objects to storyboards
- used basic layout constraints
- connected storyboard events to view controllers
- basic Swift syntax and OOP

If any of these is unclear to you, please make sure to go back and read the related part or parts before moving on the the next chapter.

For reference, you can download the complete Xcode project from

https://github.com/CarlistleZ/MyiOSTutorial/blob/master/ChapterFour_HelloWorld.zip