

# Practical: 8

**AIM- Create your first Hello World iOS app  
in SwiftUI**

Submitted By: Pael Krish Pankajbhai  
Enrollment number: 23012011062



**Ganpat  
University**

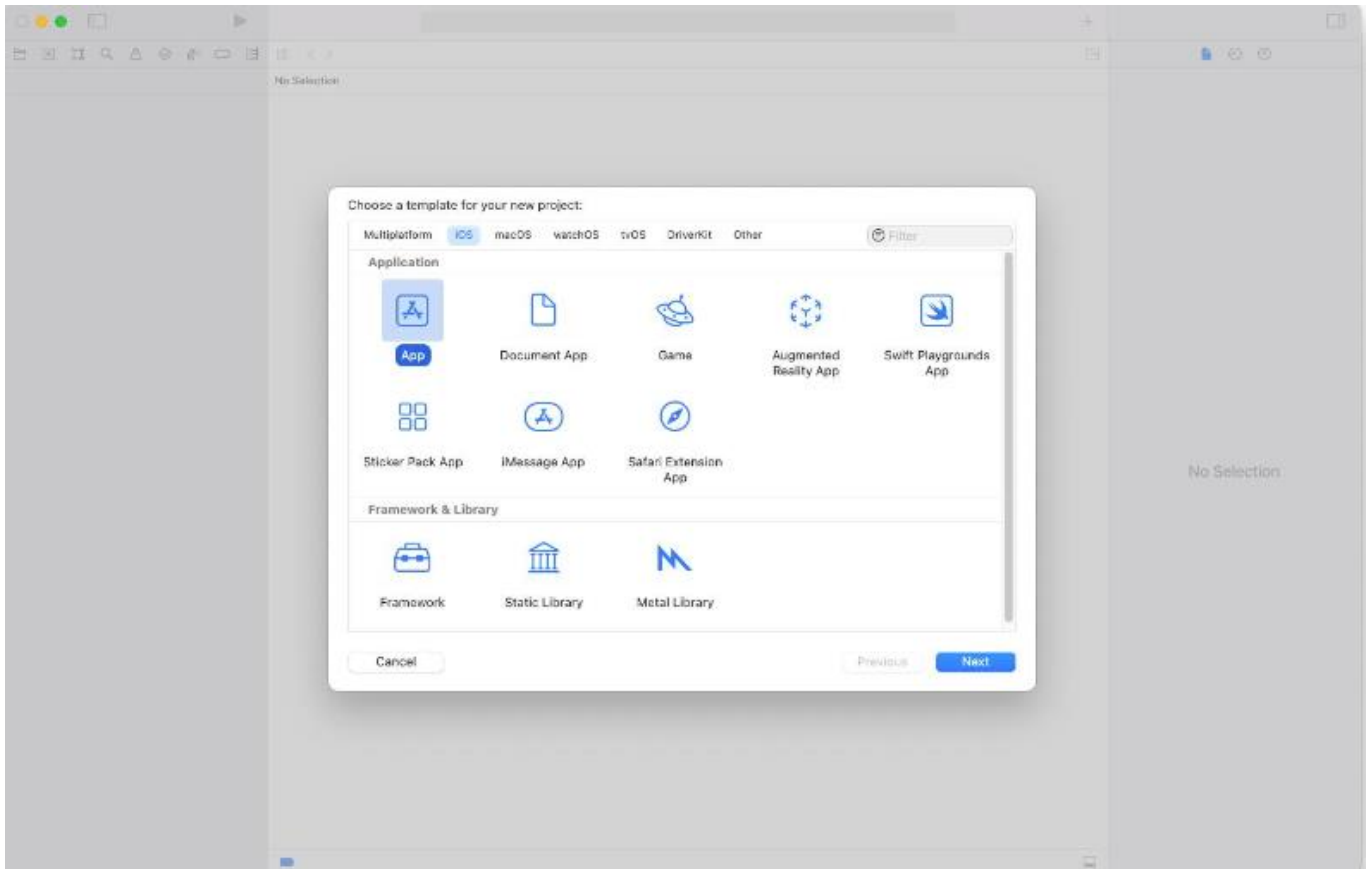
॥ विद्यया समाजोत्कर्षः ॥

**U.V. Patel  
College of  
Engineering**

**Department of Computer Engineering**

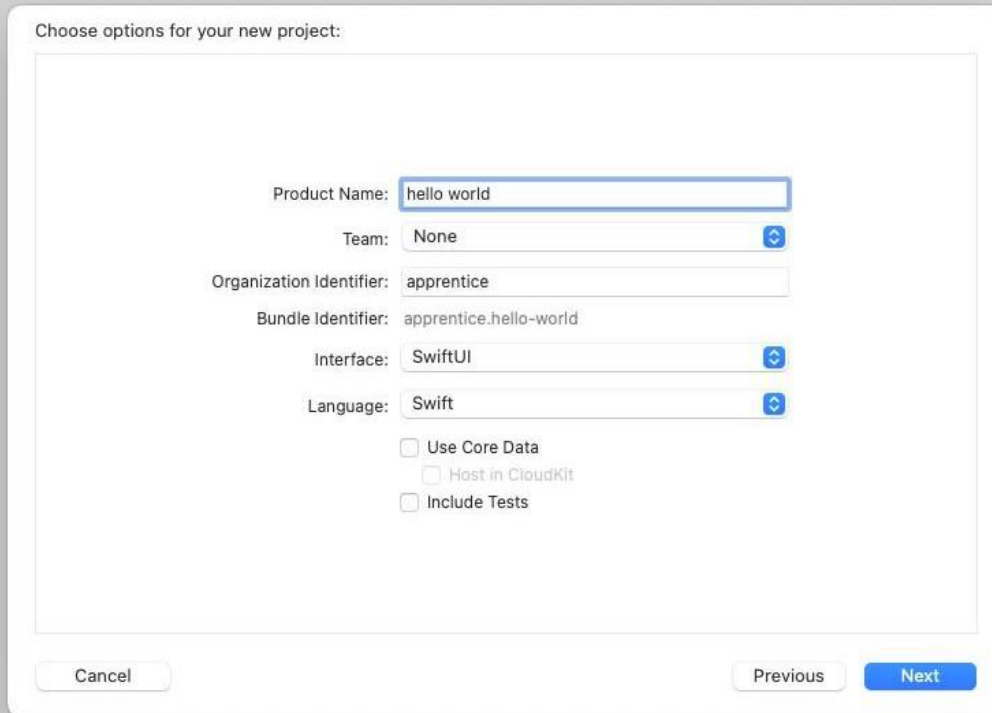
### Practical-8

1. Create a new iOS app by selecting File > New > Project... from the Xcode menu. Choose the App template from the Application section in the iOS tab.

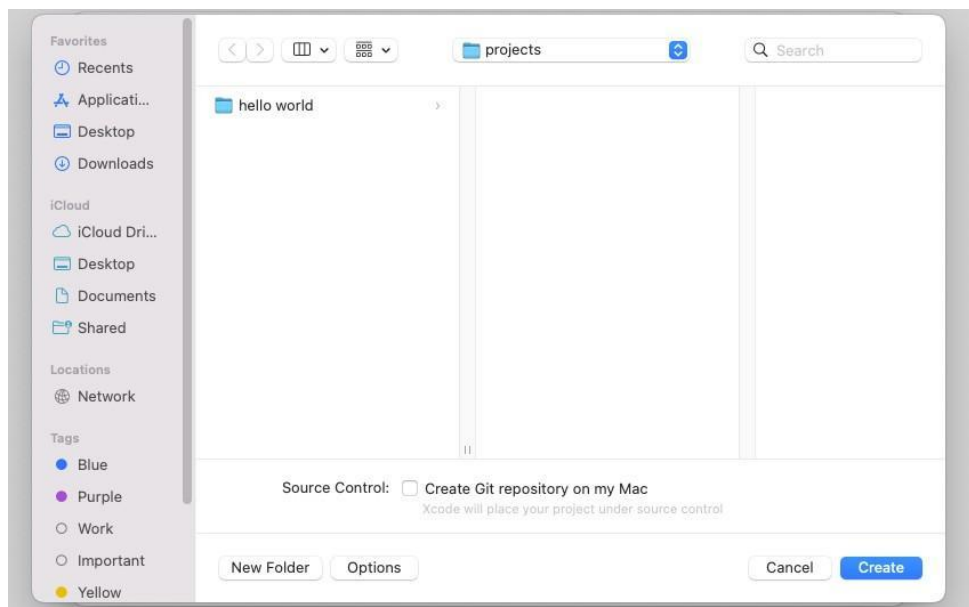


- (a) Name the app hello world.
- (b) Xcode uses name to set the bundle identifier for the app.
- (c) Select SwiftUI for the technology used to build the app's user interface.
- (d) Set Swift for the programming language used to code the app.
- (e) Ensure that the Core Data and unit testing options are not selected. You don't need local databases to store the app's data since it's elementary. There is also no need to test the code of such a basic app.

## Practical-8



(f) Finally, check the Create Git repository on my Mac for your project and save it. It's useful to initialize a Git repository for any project you create since you will likely need it



## Practical-8

### 2. Understanding the structure of a basic SwiftUI iOS app

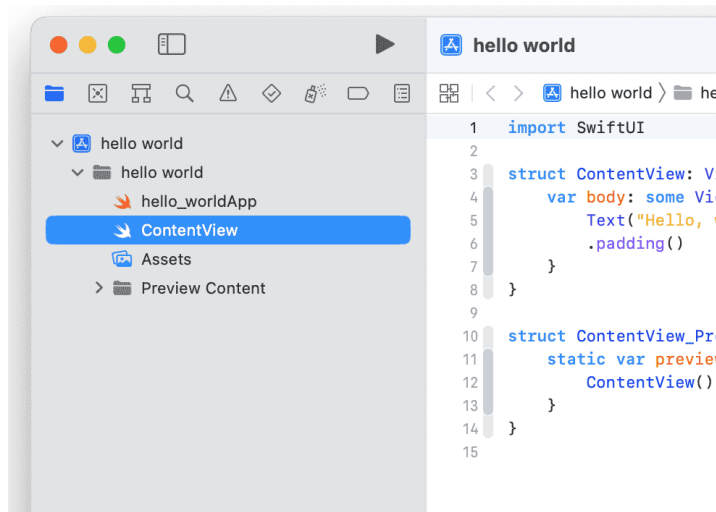
(a) The hello world project has two source code files. You can choose either one of them from the project navigator on the left side of the Xcode editor.

(b) Open the *hello\_worldApp.swift* file and take a look at the code.

```
import SwiftUI
@main
struct hello_worldApp: App {
    var body: some Scene {
        WindowGroup {
            ContentView()
        }
    }
}
```

(c) The most important thing you should be aware of is that you are using a *structure*, which is a custom Swift data type, to define the main scene of the app. The scene handles a group of views of the app. There

is only one view, in this case, declared in the *ContentView.swift* file.



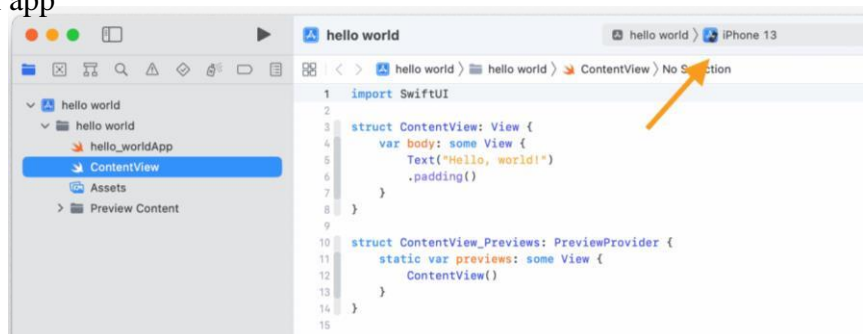
Open the *ContentView.swift* file. And take a look the code:

```
struct ContentView: View {
    var body: some View {
        Text("Hello, world!")
        .padding()
    }
}
```

### (3) Run your first iOS app in the simulator.

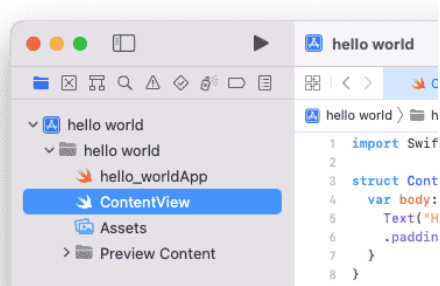
You don't need an iPhone to run basic apps. A *simulator* has all of the essential features of an actual device, and it is much easier to run an app using this.

- (a) The first step is to choose the iOS simulator that you want to run the app on. A default simulator is already selected at the top of your Xcode project window.

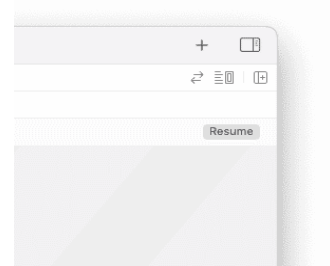
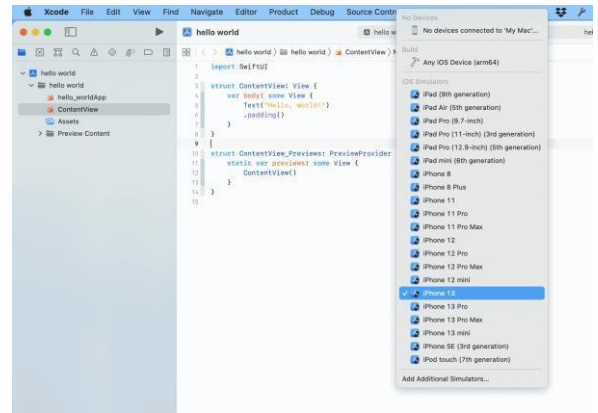


## Practical-8

- (b) You can click on the default simulator and choose a custom one from the simulator menu that appears.

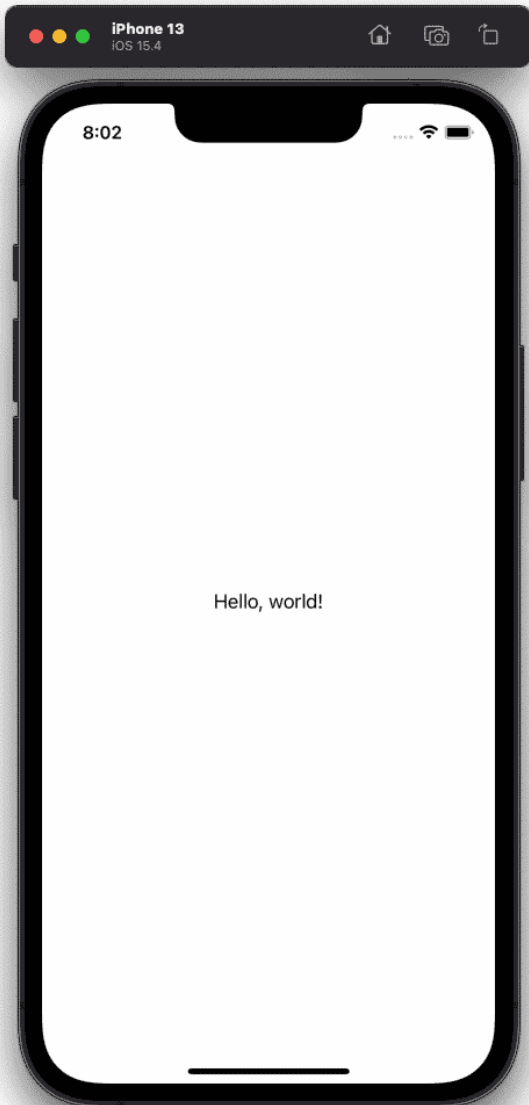


- (c) Press the play button at the top left of the project window to run the app.



## Practical-8

---



The project builds itself without errors and then runs the app on the chosen simulator

---

SwiftUI also allows you to preview the app's appearance without running the app on a simulator. To do this, switch to the *ContentView.swift* file and look at the preview code.

```
struct ContentView_Previews:
PreviewProvider { static var previews:
some View {
                                ContentView()
                                }
}
```

The code creates a structure that handles the updates of the app's main view to preview them. You can press the resume button at the top right of the

The preview simulator is perfectly synchronized with the code of the view. This means that any changes you make to the code in the left panel can be seen in the simulator in the right panel immediately

## Practical-8

