# Introduction of Xcode

- Xcode is Apple’s IDE (integrated development environment), which you use to create iOS applications.

- The word “integrated” refers to the fact that Xcode brings together several different tools into a single application.

- Xcode contains several tools, but the ones you’ll use most of the time are the source code editor, debugger, and the Interface Builder.

# Why Xcode is important?

- Xcode is the heart of iOS development because:
  - It provides everything in one place.
  - It supports swift and objective-c programming.
  - It includes many built-in tools like simulator, debugger, UI designer etc.

# Key features of Xcode

- Code editor
- Interface builder
- IOS simulator
- Debugger
- Project navigator
- Playground

# What can students build using Xcode?

- Mobile apps (iPhone / iPad)
- Games
- Apple watch apps
- Mac desktop apps
- UI / UX prototype
- Learning experiments (swift basic, animations, API calls).

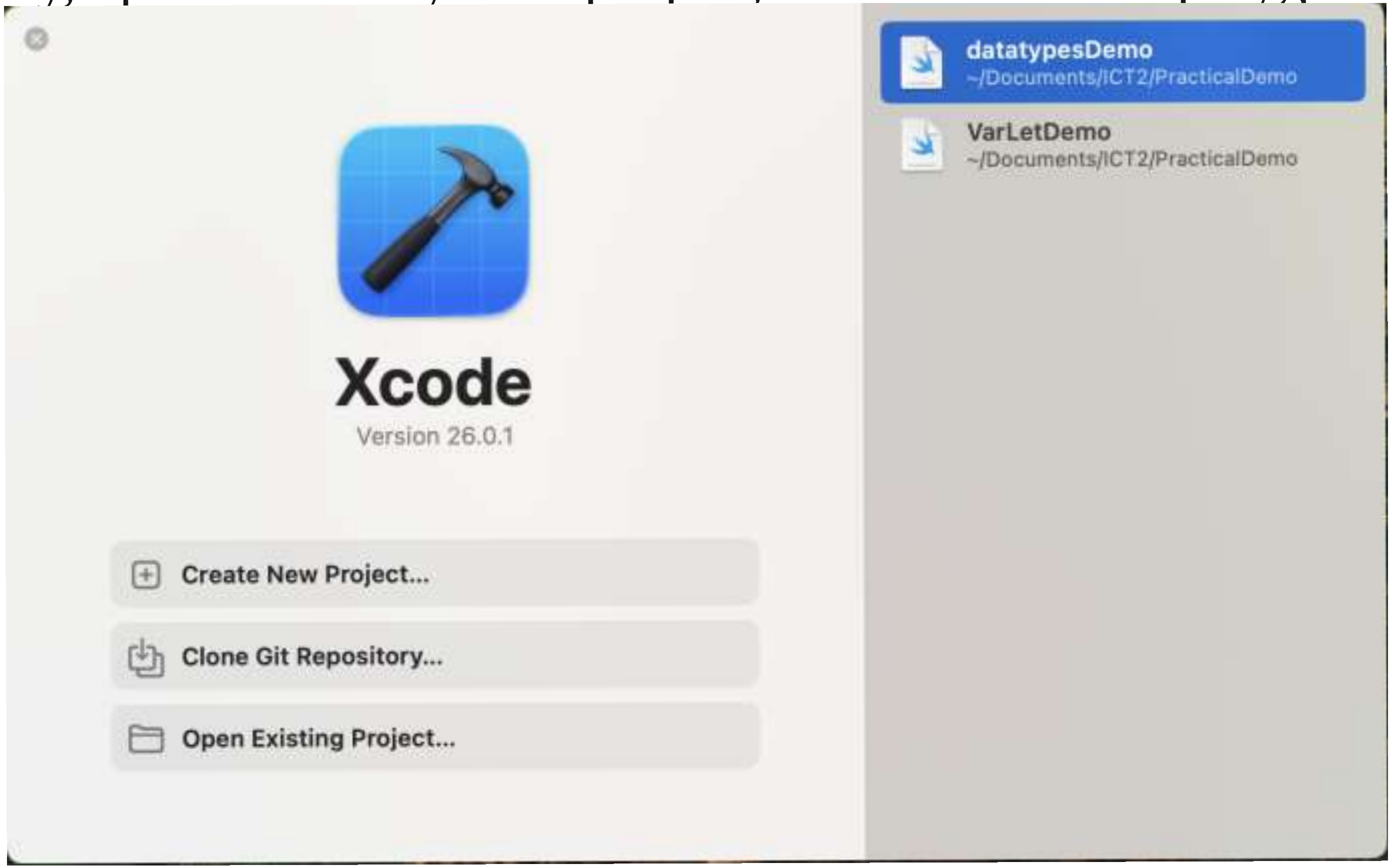
# Why should students learn Xcode?

- Industry demands for iOS developer high.
- Easy to learn once students learn basic.
- Helps to improve logic building and UI/UX thinking.
- Students can publish their own apps on the app store.
- Enhance resume and career opportunities.

# System requirements

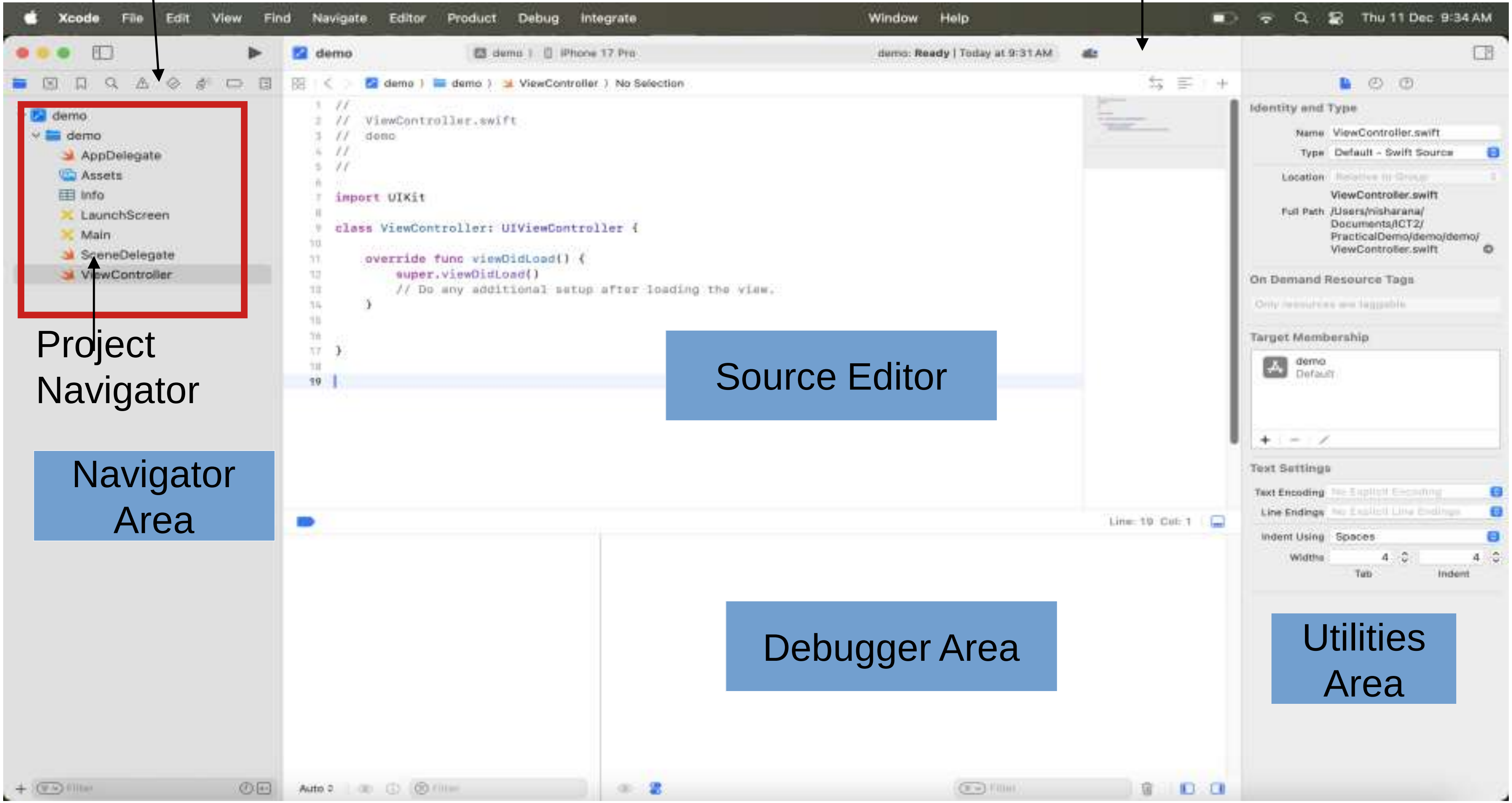
- To run Xcode, students need:
  - Mac computer with macOS
  - Latest version of Xcode(free from mac app store)

- When you launch Xcode, you are presented with the welcome dialog box. You can use the welcome dialog box to quickly create a new project, connect to a source code repository, open a recently used project, or create a Swift playground.



Symbol Navigator

Toolbar



```
//
//  ViewController.swift
//  demo
//
//

import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.
    }


}

```

Identity and Type

Name: ViewController.swift

Type: Default - Swift Source

Location: Relative to Group

ViewController.swift

Full Path: /Users/nisharana/Documents/ICT2/PracticalDemo/demo/demo/ViewController.swift

On Demand Resource Tags

Only resources are taggable

Target Membership

demo
Default

Text Settings

Text Encoding: No Explicit Encoding

Line Endings: No Explicit Line Endings

Indent Using: Spaces

Widths: 4 Tab, 4 Indent

Project Navigator

Navigator Area

Source Editor

Debugger Area

Utilities Area

# Navigator Area :

- The left side of the workspace window is the navigator area.
- The Project Navigator :
  - The project navigator shows the contents of your project.
  - Individual files are organized within groups that are represented as folders in a tree structure.
- The Symbol Navigator :
  - The symbol navigator shows the classes in your project along with their methods and member variables.
  - A top-level node in a tree-like structure represents each class.

# Navigator Area :

- Find navigator :
- The find navigator  lets you find all occurrences of some text, across all files of the project.

- The Issue Navigator :
- The issue navigator  lists all compile-time errors and warnings in your project.

# Navigator Area :

- The Test Navigator :
  - The test navigator gives you a snapshot of all the unit tests created with the project.
- The Debug Navigator :
  - The debug navigator is used during an active debugging session and lists the call stack for each running thread.

# Navigator Area :

- The Breakpoint Navigator :
  - The breakpoint navigator lists all breakpoints in your code and allows you to manage them.
- The Report Navigator :
  - The report navigator shows you a history of build logs and console debug sessions.

# Editor Area :

- The right side of the workspace window is the editor area. Xcode includes editors for many file types, including source code, user interface files, XML files, and project settings, to name a few.
- The content of the editor area depends on the current selection in the navigator area.

# Debugger Area :

- The debugger area  also supplements the editor area. You can access it by selecting View ➪ Show Debug Area or by clicking the debugger button in the toolbar.
- The debugger area is used while debugging an application and to access the debug console window.
- You can use this area to examine the values of variables in your programs.

# Utilities Area :

- The utilities area supplements the editor area. You can display it by selecting View ➪ Utilities ➪ Show Utilities or by clicking the utility button in the toolbar.
- The Inspector Area :
    - The top portion of the utilities area contains the inspector area  The number of tabs available depends on the currently selected item in the project navigator.
    - The file inspector provides access to the properties of the current file.
    - The quick help inspector provides a short description of the current file.

# Utilities Area :

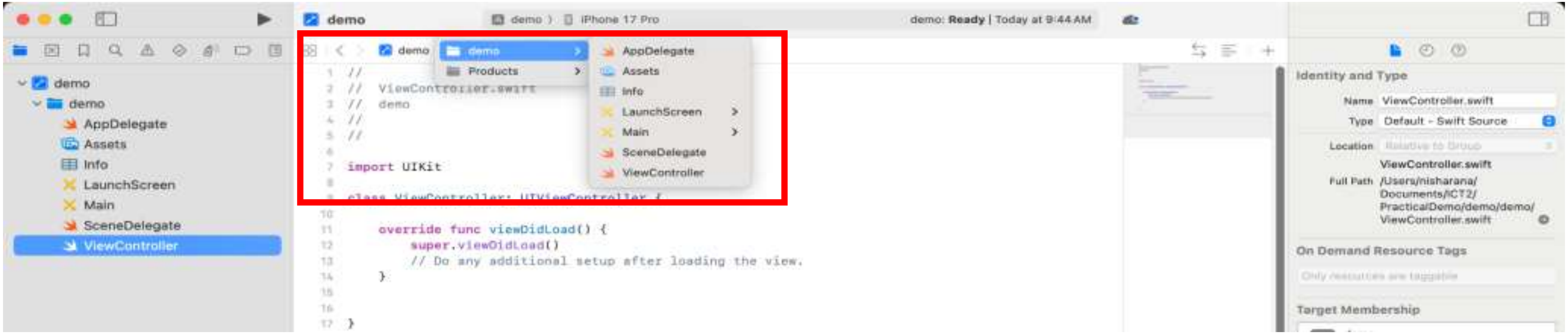
- The Library Area:
  - The bottom portion of the utilities area contains the library area.
  - This area contains a library of file templates, user interface objects, and code snippets that you can use in your applications.

# The Toolbar:

- The Xcode toolbar is located at the top of the workspace window.
- Use the first two buttons on the left side to run/stop the active build scheme. Immediately following the stop button is the Scheme/Target multi-selector.
- When you create an iOS project, Xcode creates a scheme with the same name as the project and several build targets.

- The build targets that are typically generated for a project include:
- iOS Device
    - iPad 2 (if it is an iPad or Universal project)
    - iPad Air (if it is an iPad or Universal project)
    - iPad Retina (if it is an iPad or Universal project)
    - iPhone 4S (if it is an iPhone or Universal project)
    - iPhone 5 (if it is an iPhone or Universal project)
    - iPhone 5S (if it is an iPhone or Universal project)
    - iPhone 6 Plus (if it is an iPhone or Universal project)
    - iPhone 6 (if it is an iPhone or Universal project)
- You can use the Scheme/Target multi-selector to switch build targets and create/edit schemes.

# Jump Bars:



```
//
//  ViewController.swift
//  demo
//
//

import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.
    }

}
```

Identity and Type

Name: ViewController.swift

Type: Default - Swift Source

Location: Relative to Group

ViewController.swift

Full Path: /Users/nisharana/Documents/ICT2/PracticalDemo/demo/demo/ViewController.swift

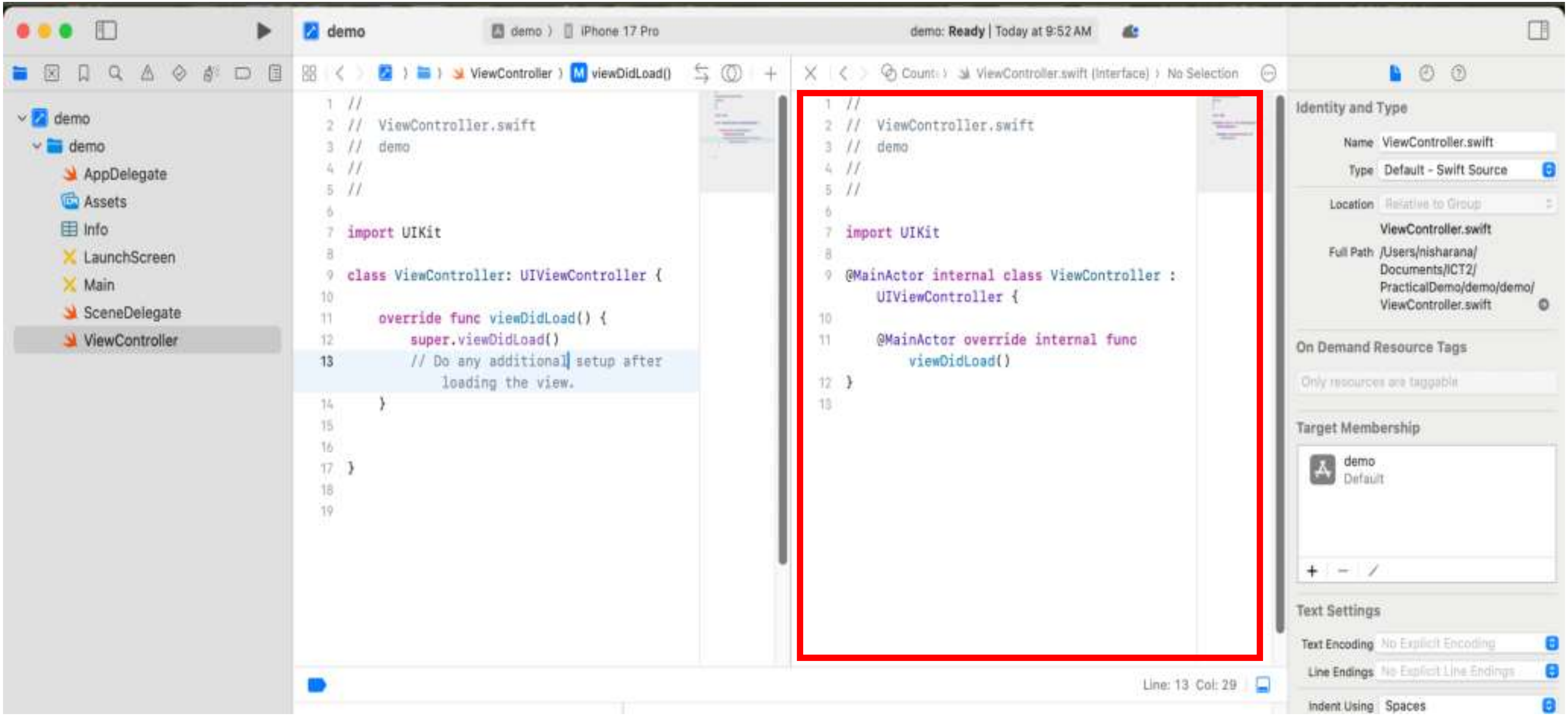
On Demand Resource Tags

Only resources are taggable

Target Membership

- At the top of the editor area is the jump bar. The jump bar displays the path to the current file being edited and can be used to quickly select another file in the workspace.
- The jump bar also has back and forward buttons to move through a history of files edited.

# The Assistant Editor:


demo
demo 〉 iPhone 17 Pro
demo: Ready | Today at 9:52 AM
ViewController 〉 viewDidLoad()
Counts 〉 ViewController.swift (Interface) 〉 No Selection
demo
demo
AppDelegate
Assets
Info
LaunchScreen
Main
SceneDelegate
ViewController
Line: 13 Col: 29

```
//
//  ViewController.swift
//  demo
//
//

import UIKit

class ViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view.
    }


}
```

```
//
//  ViewController.swift
//  demo
//
//

import UIKit

@MainActor internal class ViewController : UIViewController {

    @MainActor override internal func viewDidLoad()
}
```

Identity and Type

Name: ViewController.swift

Type: Default - Swift Source

Location: Relative to Group

ViewController.swift

Full Path: /Users/nisharana/Documents/ICT2/PracticalDemo/demo/demo/ViewController.swift

On Demand Resource Tags

Only resources are taggable

Target Membership

demo
Default

Text Settings

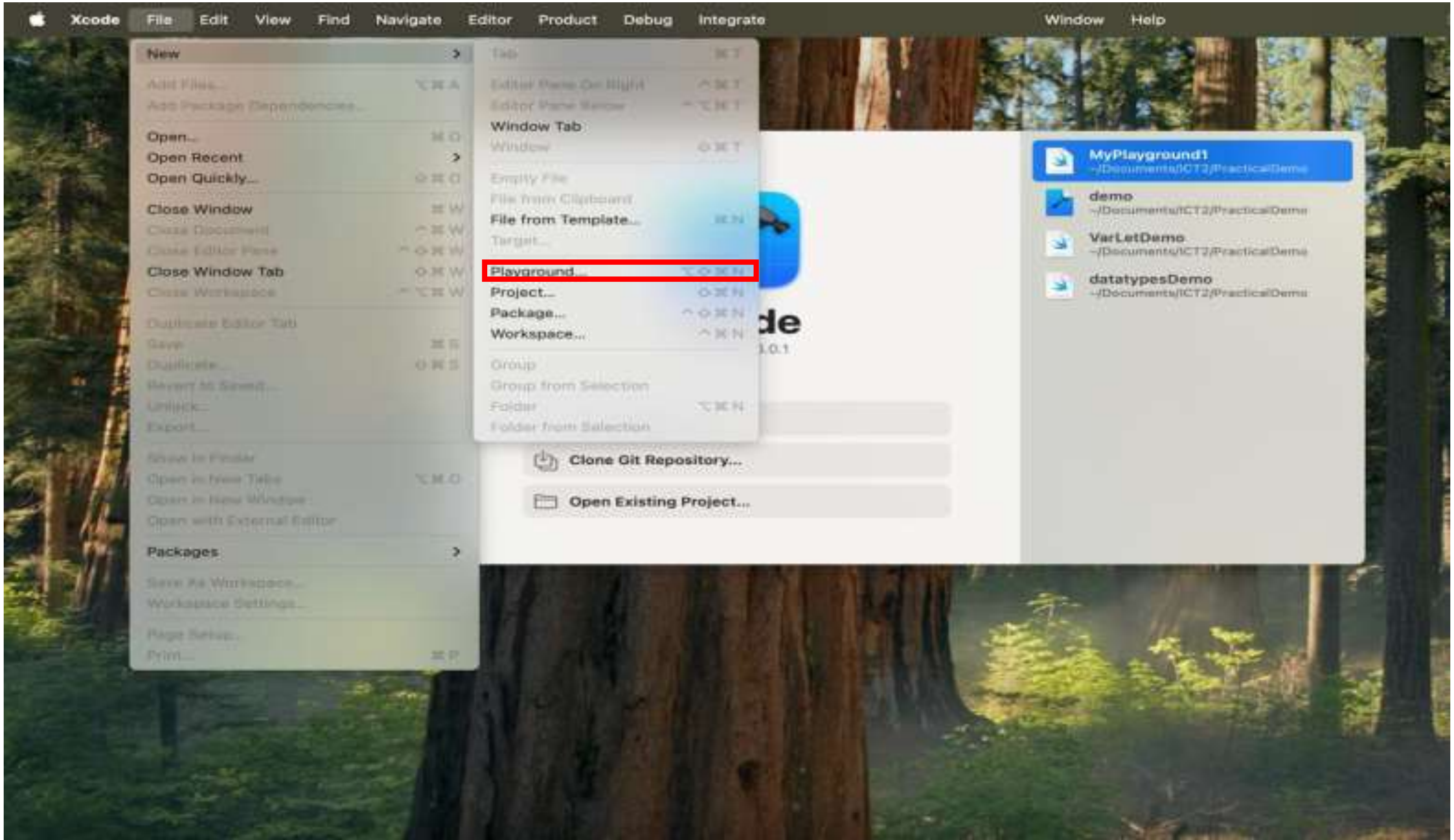
Text Encoding: No Explicit Encoding

Line Endings: No Explicit Line Endings

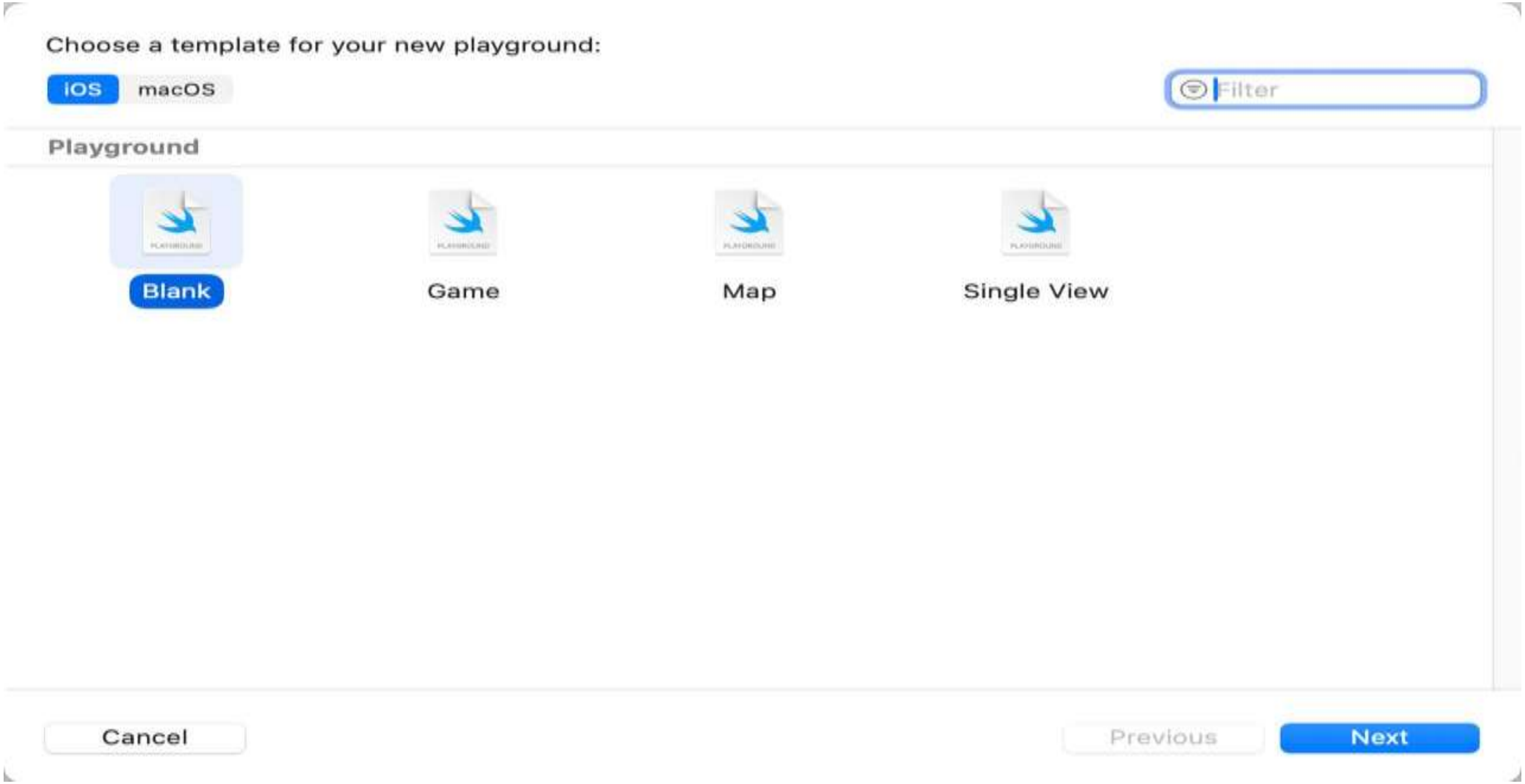
Indent Using: Spaces

- The assistant editor was introduced in Xcode 4 and enables you to view multiple files side-by-side.
- The assistant editor is not visible by default and can be accessed by using the editor selector buttons in the Xcode toolbar or by selecting View ⇨ Assistant Editor ⇨ Show Assistant Editor.
- You can create additional assistant editor panes by using the + button in the jump bar of the assistant editor.

# Playground :

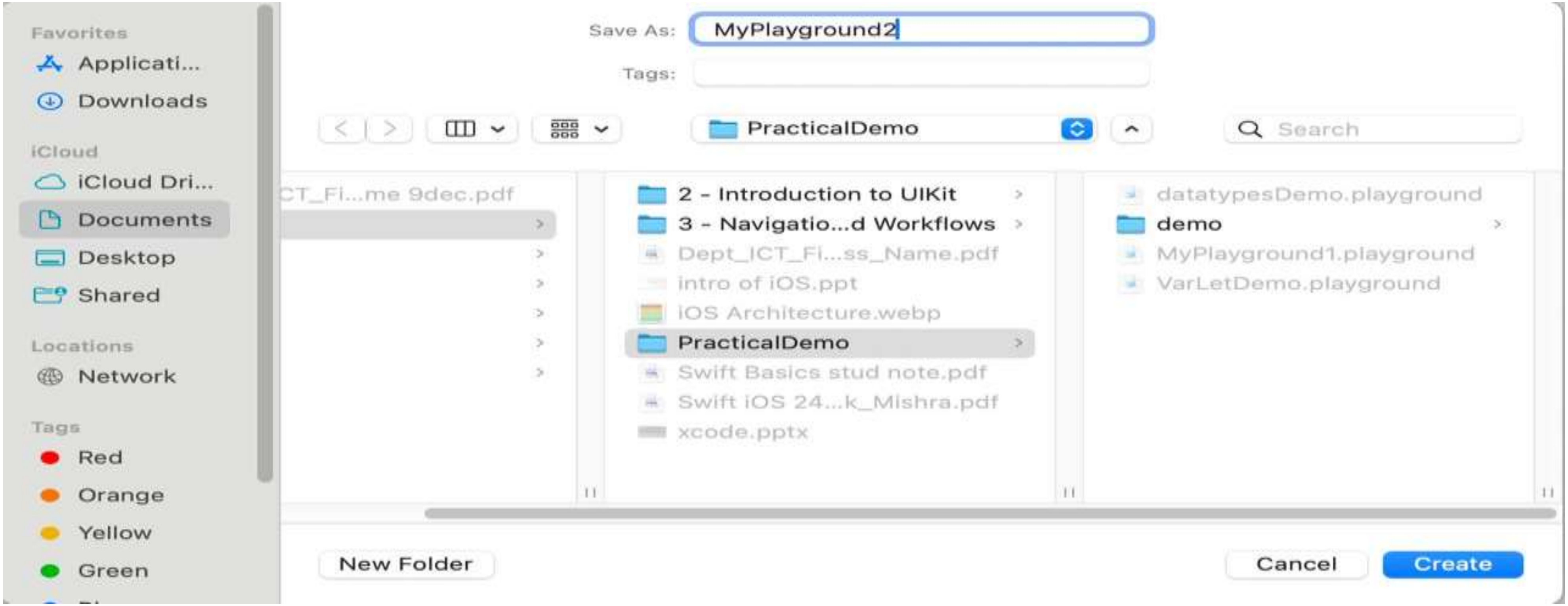

Xcode
File
Edit
View
Find
Navigate
Editor
Product
Debug
Integrate
Window
Help
New
Add Files...
Add Package Dependencies...
Open...
Open Recent
Open Quickly...
Close Window
Close Document
Close Editor Pane
Close Window Tab
Close Workspace
Duplicate Editor Tab
Save
Duplicate...
Revert to Saved...
Unlock...
Export...
Show in Finder
Open in New Tabs
Open in New Window
Open with External Editor
Packages
Save As Workspace...
Workspace Settings...
Page Setup...
Print...
Tab
Editor Pane On Right
Editor Pane Below
Window Tab
Window
Empty File
File from Clipboard
File from Template...
Target...
Playground...
Project...
Package...
Workspace...
Group
Group from Selection
Folder
Folder from Selection
Clone Git Repository...
Open Existing Project...
MyPlayground1
~/Documents/ICT2/PracticalDemo
demo
~/Documents/ICT2/PracticalDemo
VarLetDemo
~/Documents/ICT2/PracticalDemo
datatypesDemo
~/Documents/ICT2/PracticalDemo

# Playground :



Choose a template for your new playground:

# Playground :

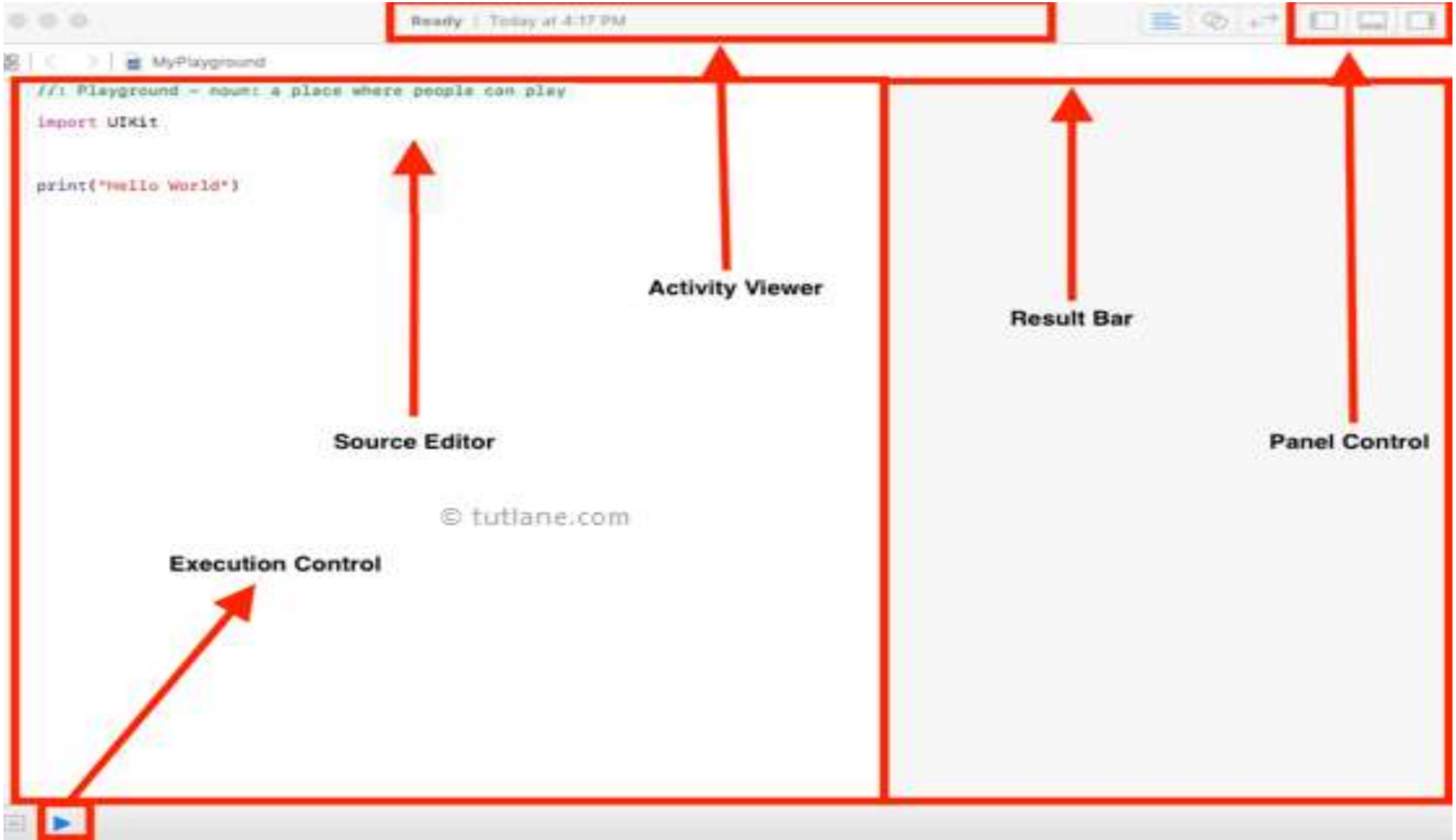


Save As: MyPlayground2

Tags:

PracticalDemo

# Playground :



```
//: Playground - noun: a place where people can play

import UIKit

print("Hello World")
```

# Playground :