

**Mobile Application Development**

**MAD400 – iOS Development**

**Assignment 1 Part 2**

**Submitted By:**

Rohan Patel

**Student ID:**

A00115524

**Submitted to:**

Prof. Manhar Kapoor

**Date:**

14-7-2021

**What is XCode?**

Developers use Xcode to create apps for Apple's numerous platforms, including the iPhone, iPad, Macs, AppleTV, and Apple Watch.

**XCode Requirements**

Although Xcode is exclusively available for Macs, there are other solutions for PC users. I'm also frequently asked if you can run Xcode on an iPad, and the answer is no.

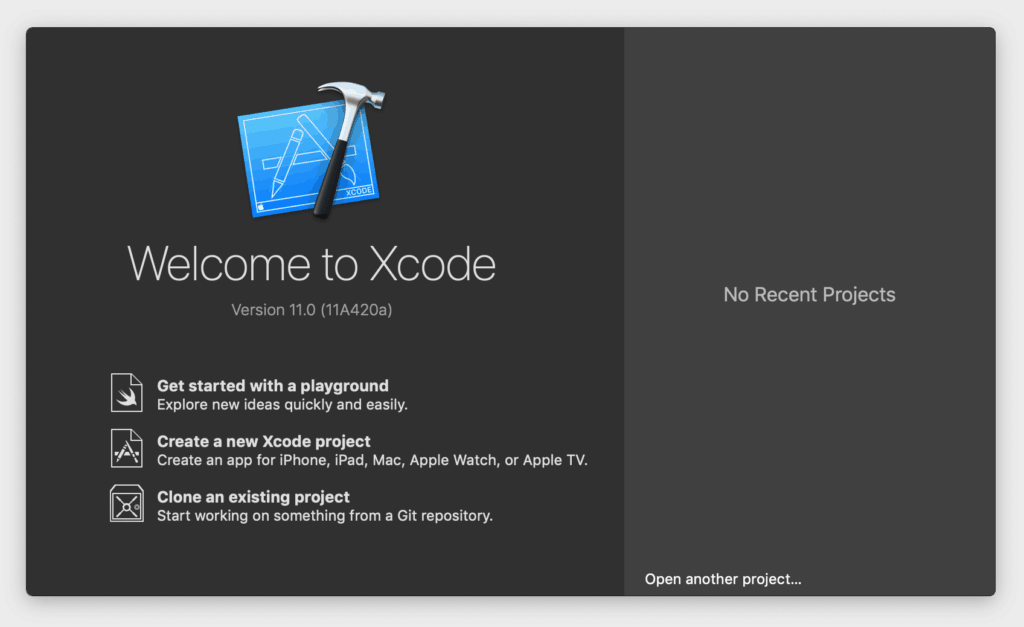
It requires macOS 10.14.4 or later, as well as 7.6 GB of free space on your hard disc. If you don't satisfy the minimal requirements, you can use this approach to upgrade Xcode (at your own risk).

Xcode 11 is the most recent version. If you want to see what version of Xcode you have, simply run it and look at the welcome screen, where it will be clearly displayed.



**What is new in XCode 11?**

Support for Swift 5.1, Apple's latest edition of the Swift programming language, as well as SDKs for iOS 13, tvOS 13, watchOS 6, and macOS Catalina 10.15 are all included in Xcode 11.



Each of those platforms has evolved since the last version, and Xcode 11 allows you to create apps that take advantage of the new capabilities.

In terms of how Xcode has evolved as an IDE (integrated development environment), there are a slew of new capabilities to make developing applications easier.

**XCode on windows**

Unfortunately, Xcode is only available for Macs, but there are ways to run Xcode on Windows if you have a PC.

Alternatives of Xcode:

AppCode

Visual Studio Code

React Native

Flutter

Xamarin

PhoneGap

Appcelerator

**Xcode IDE**

Source Editor

Write code using a professional editor that includes powerful code completion, code folding, syntax highlighting, and message bubbles that show warnings, errors, and other context-sensitive information right alongside your code.

Assistant Editor

The Assistant button divides the editor into two panes, one of which automatically displays files that are most useful to you based on the code you're now editing. It can display the counterpart of the header, the superclass, callers, callees, and other useful files.

Version Editor

With full support for Subversion and Git source control (SCM) systems, Xcode's Version editor displays a running timeline of commits, helps you determine blame, and graphically goes back in time to compare source files.

Built-in Interface Builder

Without writing a single line of code, you can design and test your user interface in minutes, then graphically connect it to the source in the Xcode editor.

Simulator

Xcode can build, instal, run, and debug Cocoa Touch apps in a Mac-based Simulator using the iOS SDK, allowing for a more efficient development process.

Integrated Build System

Handles the most sophisticated builds, scaling to take advantage of multi-core Macs' capabilities, and will sign, provision, and instal iPad and iPhone apps on a device effortlessly.

Compilers

Xcode includes the sophisticated open-source LLVM compiler for C, C++, and Objective-C, which can be accessed via Terminal. Your code compiles rapidly using it, and Apple optimises it to deliver bl

Graphic Debugger

You can debug your programme right in the Xcode editor. Drill into the contents of any variable by hovering over it, using Quick Look to see the data it contains, or right-clicking to add the variable to the watch list.

Continuous Integration

Xcode Server manages server-side bots that build, analyse, test, and archive your Xcode projects in real time. The Xcode IDE sets up these bots, analyses nightly build and test results, and can pinpoint which check-in caused the build to fail.

Asset Catalog

The asset catalogue editor in Xcode organises your app's photos by grouping them together in different resolutions. Xcode builds the asset catalogue into the most efficient bundle for ultimate distribution when constructing it.

Open Quickly

To open any file in your workspace using the primary editor, simply press Cmd-Shift-O, or hold the Option key when selecting a file to open it in the Assistant editor. In every keyboard-driven workflow, Open Quickly is a must-have tool.

OpenGL Frame Capture

To record a complete representation of the current OpenGL frame from an iOS device, simply press a single button. Within the Xcode debugger, Xcode exposes the shader information and may visually reconstruct how the frame was put together.

Complete Documentation

Within Help, the Documentation and API Reference viewer, you can easily search and locate anything.

Live issues

Live Issues displays common coding issues in the same way that a word processor does, without the requirement to first select "build."

Fix-it

Xcode does a lot more than merely report issues. When you make a coding error, Xcode will notify you right away, and a simple keyboard shortcut will instantly correct the problem, ensuring that you don't miss a beat when coding.

Quick Help

While you're programming, a condensed version of the API documentation is displayed, along with any comments you add to your code. During code completion, a brief overview is displayed, with more links and references available in the Utility area.

The XCTest Framework

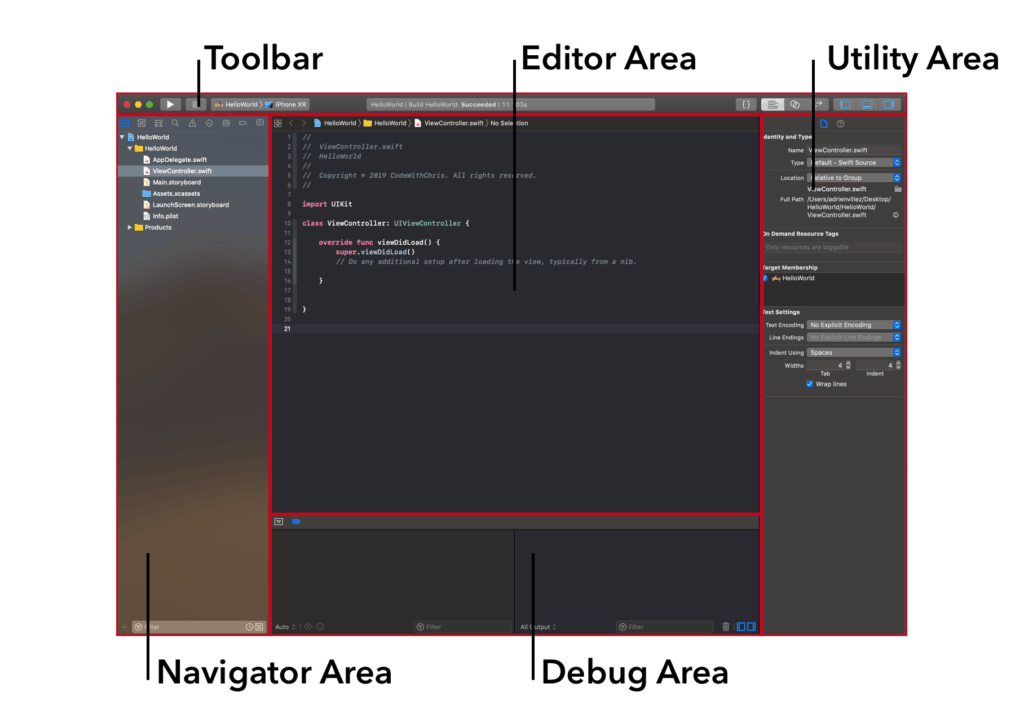
The XCTest APIs enable it simple to create unit tests that exercise app functionality on the Mac, iPad, iPhone, or Simulator.

Static Analysis

Allow the built-in static analyzer to test thousands of possible code paths in a matter of seconds to find issues in your code before the programme is even launched. You'll receive a report detailing any potential bugs that may have stayed undetected or are nearly impossible to reproduce.

**How to use Xcode**

Let's have a look at the illustration below. Make sure you're using Xcode 11 and not an earlier version if your interface looks different.

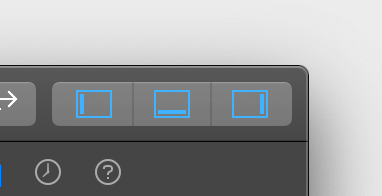


This is an excellent diagram since it allows me to refer to these different portions of the interface while also allowing you to return back to it.

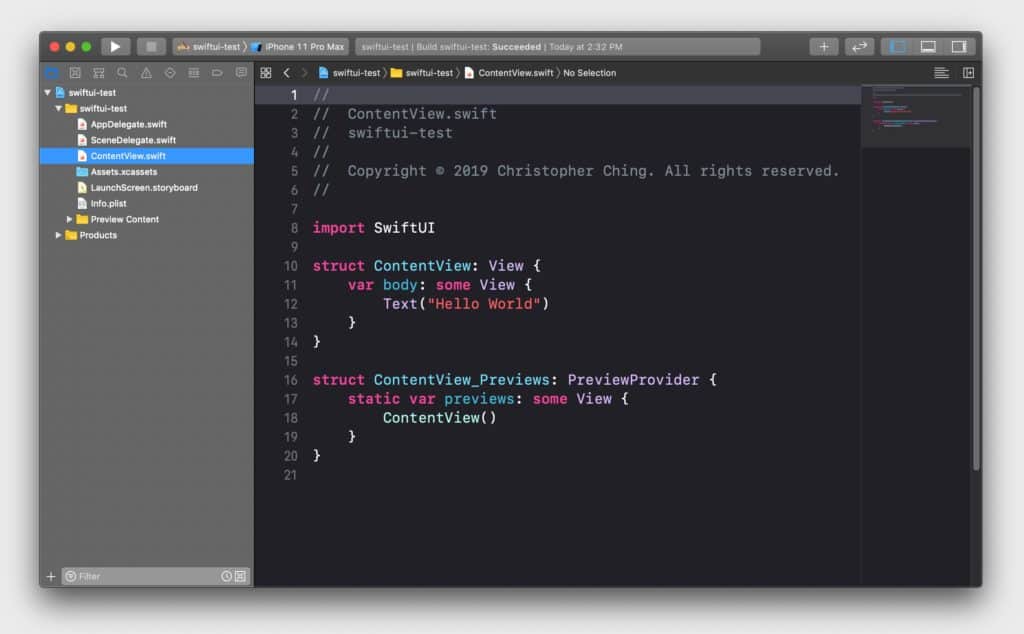
The Navigator, Editor, Utility Area, Toolbar, and Debug Area are the five key regions shown in the diagram. We'll go over each section in detail later, but for now, let's speak about Xcode's overall navigation.

Keep in mind that you may change the size of each of those windows by dragging the edges of each section with your cursor.

You can also use the "View" buttons in the upper right corner to reveal and conceal the various regions as needed:

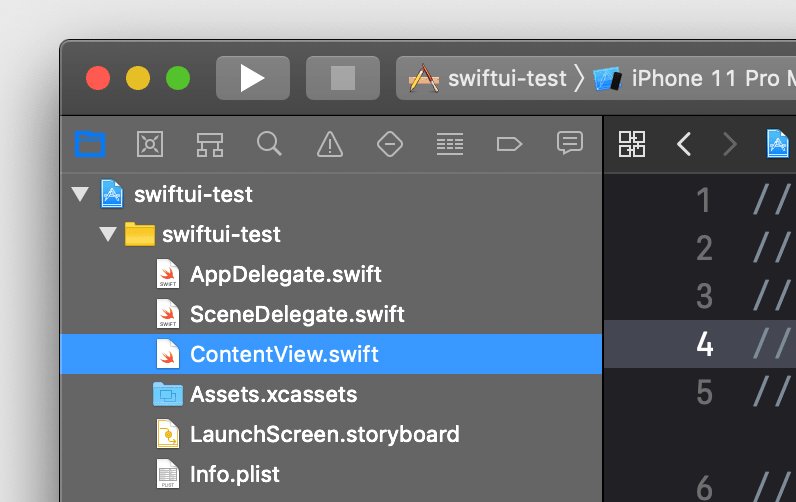


When you're writing code, for example, and you don't need the debugger or utility areas, this can be useful. Then, using the View buttons, you may hide those two windows to provide more room for your editor.



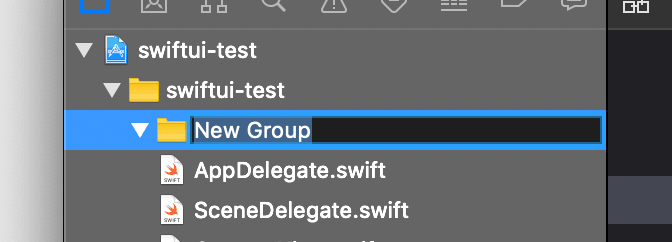
**The Xcode project navigator**

All of the files related with your project are displayed in the Project Navigator. When you create a new Xcode project, this is the default tab you'll see.



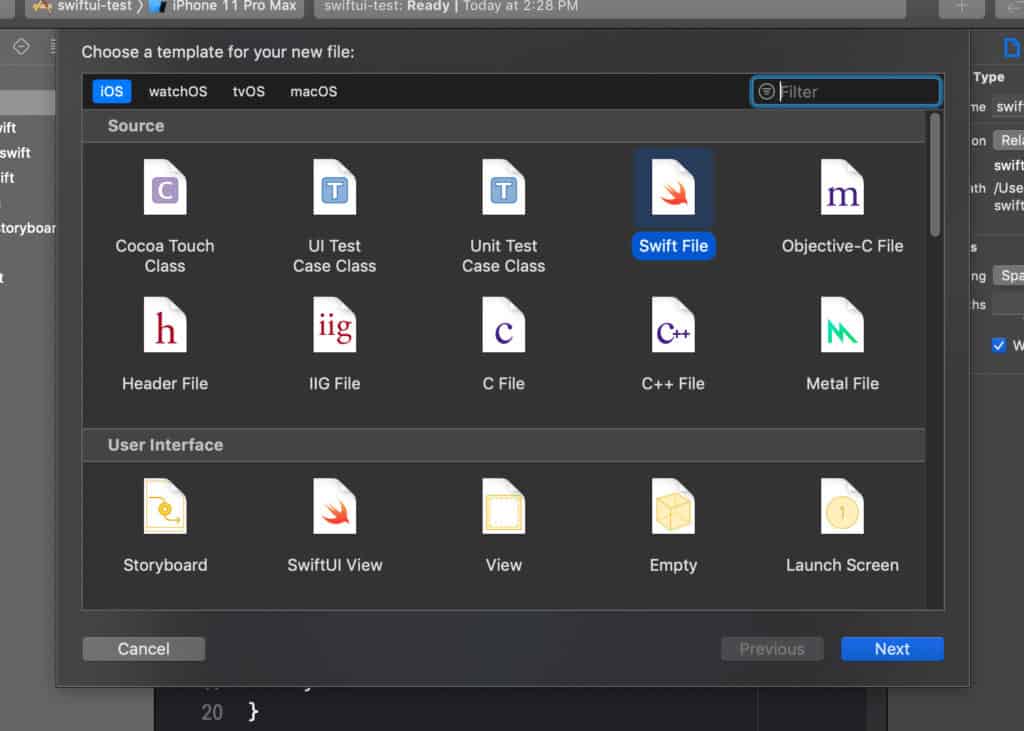
Organizing your files

You may also use the Project Navigator to arrange your files by creating Groups. Consider them to be folders. You can arrange your project by creating a new Group and then dragging your files into it.



Creating and adding new file

You can also create new files or add existing files to your project by right-clicking on the Project Navigator. You can also drag folders or files directly from your computer into the Project Navigator, which will prompt you to choose how you wish to add the data.

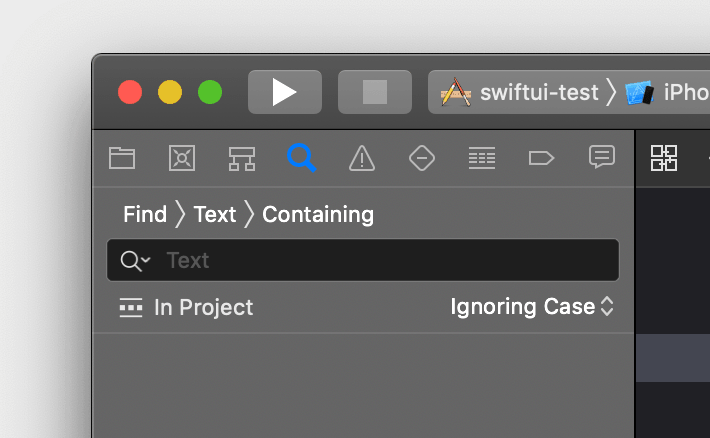


Editing Files

When you click on a file in the Project Navigator, the contents of that file will appear in the Editor Area, allowing you to modify it.

**The Xcode search navigator**

You can quickly search for text in your project using the search navigation tab.

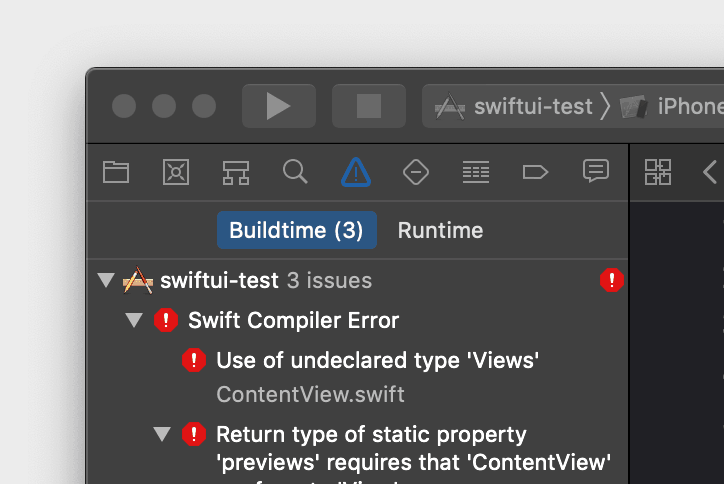


When you're looking for a specific property, method, or variable and can't remember where it is, you'll find yourself utilising this a lot.

Cmd+shift+F is one of my favourite Xcode shortcuts. By pressing these keys, you'll be taken to the Search Navigator, where you can do a project-wide search. When you need to make a change in several places within a project, this is quite useful.

**The Xcode issue navigator**

The issue navigator displays all of your app's issues.



When you build and compile your programme for testing, if there are any issues that prevent Xcode from building it, it will halt and display red coloured errors in the issue navigator. You can click on an error to see which file and line it's failing at in the editing area.

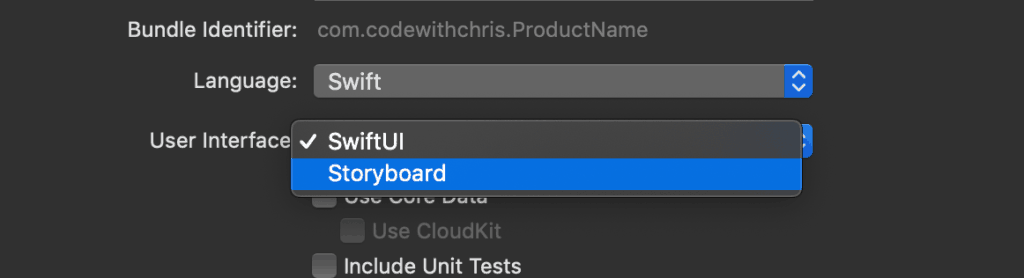
Yellow Warnings will also appear in the issue navigator. Although resolving warnings isn't required for constructing your app, they are indicators that there could be problems.

Similarly, any mistakes that occur when developing your user interface in Xcode will appear in the Issue Navigator.

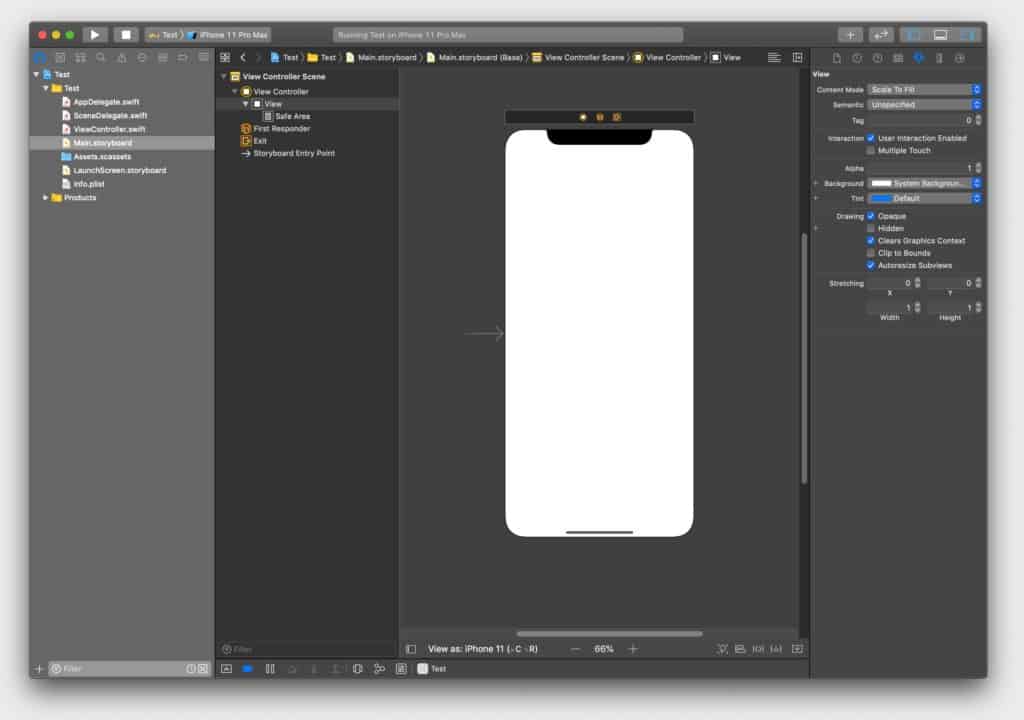
**Build your user interface (Storyboards or SwiftUI)**

Storyboards and SwiftUI are two distinct ways to create the user interface (UI) for your project in Xcode 11.

When you start a new Xcode project, you'll get a dropdown menu where you can select one of the following two options:



Storyboards are visual drag-and-drop interfaces created in the Editor Area with Interface Builder.

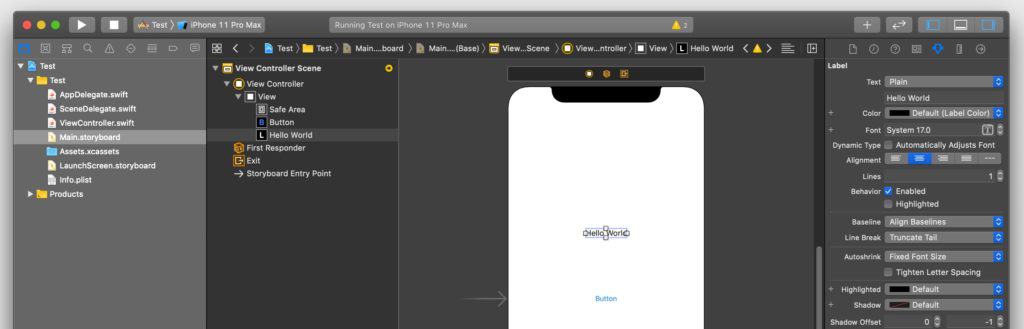


SwiftUI was published in late September 2019 as a new approach of writing code to create user interfaces. In Xcode 11, you can utilise the Preview Canvas window to see how your user interface changes in real time as you write the code to edit it.

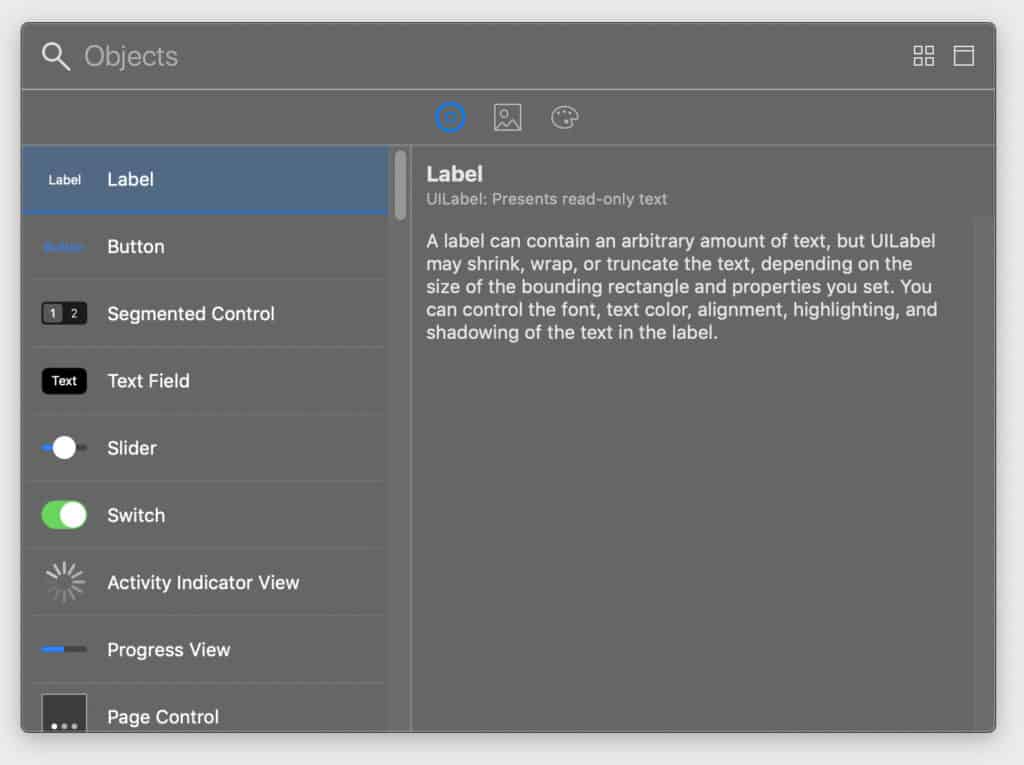
**How to use Storyboard and interface Builder**

If you chose Storyboards as your UI construction choice when creating your Xcode project, you'll have a file called Main.Storyboard in your project.

When you double-click this file, the Editor Area changes to Interface Builder, a visual UI designer.



The object Library



The Object Library is a pane that allows you to search for UIKit user interface elements. Buttons, sliders, labels, and textfields are examples of these elements.

You can drag and drop items onto the screen, but you'll need to use the Auto Layout system to size and position them.

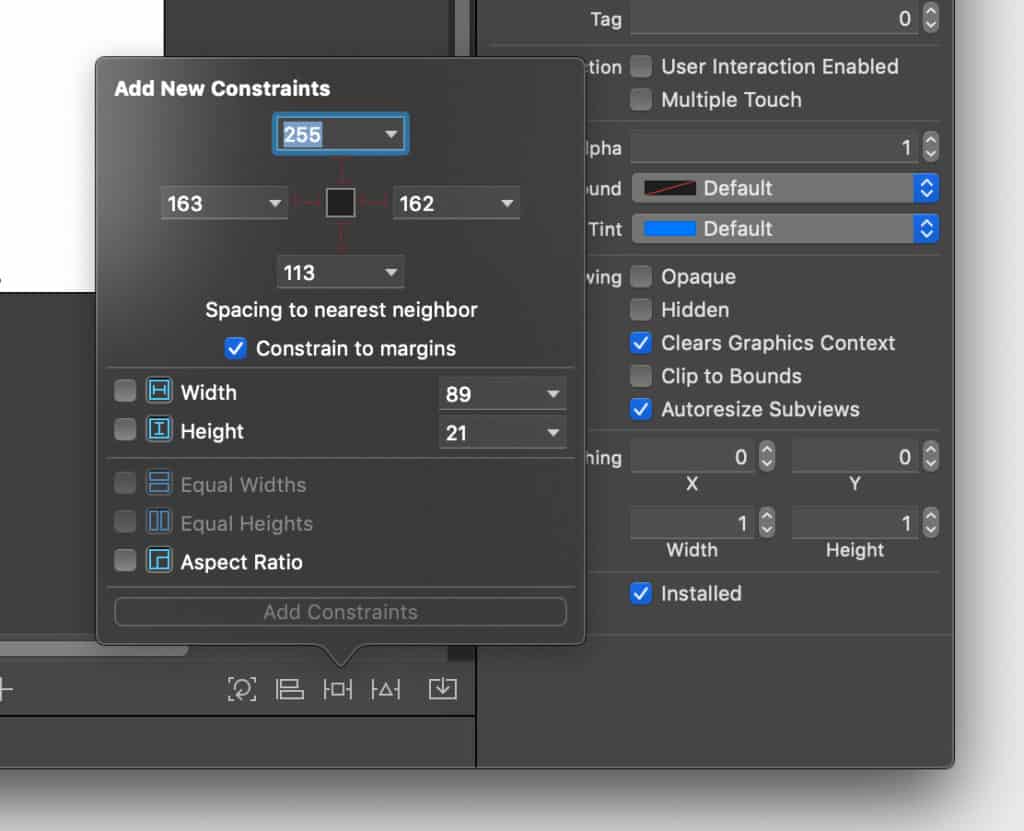
Adding auto layout constraints

Auto Layout is a mechanism for arranging and sizing elements on a display based on the rules (also known as "Constraints") you apply to them.

You could, for example, establish a constraint that says an element must be 20pt below the element above it.

The Auto Layout system has enough information to set out the user interface with enough limits in place. Having too many restrictions, on the other hand, might lead to conflicts when two or more constraints have opposing demands. In this situation, Xcode will alert you to the problem, which you must correct.

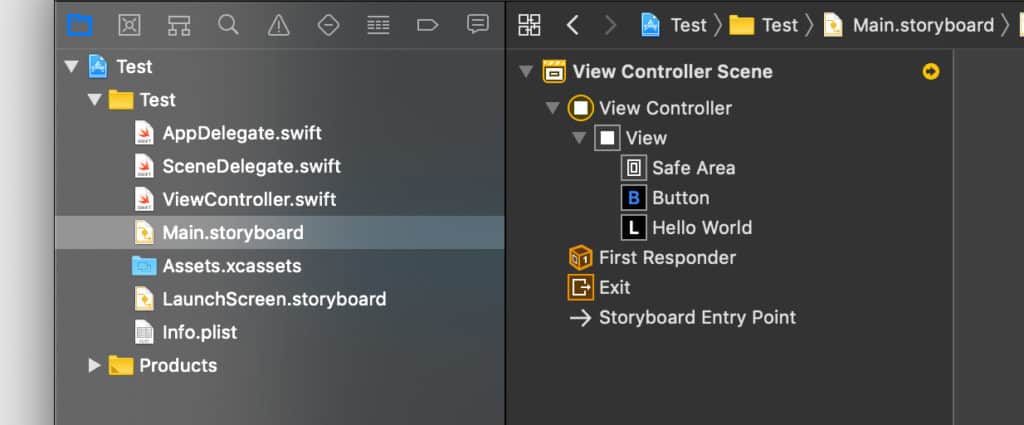
Use the buttons in the lower right corner of Interface Builder to add constraints:



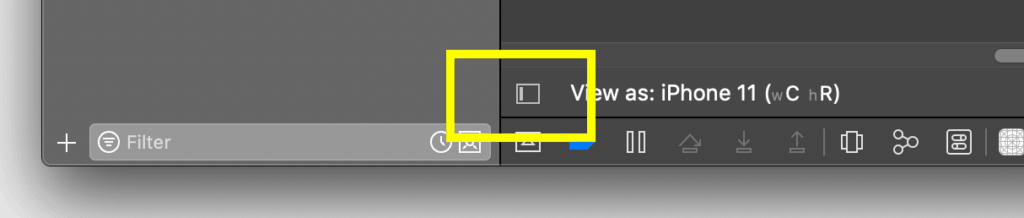
The Document Outline

Between the Navigator Area and the Editor Area is the Document Outline, which is a thin panel. It only appears while viewing an Interface Builder file, such as the Storyboard.

It's a crucial pane since it displays the visual hierarchy of all the objects you've placed in the view. It also shows you a list of all the Auto Layout limitations you've set up.



This small button in the lower left corner of Interface Builder can also be used to toggle the visibility of the Document Outline:

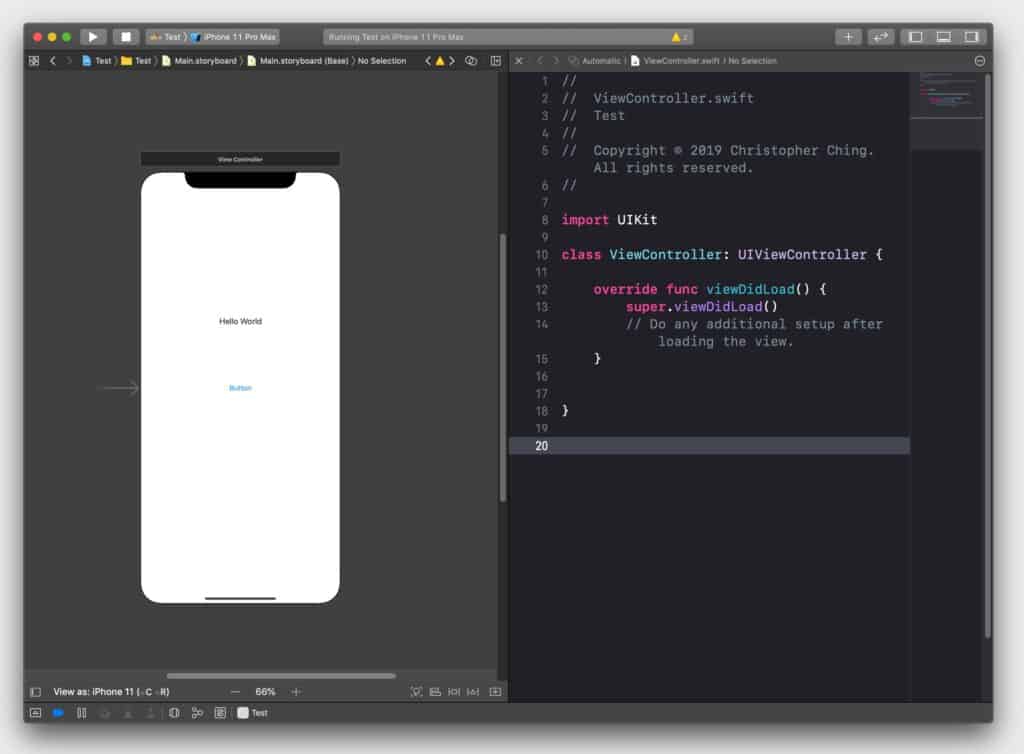


Assistant Editor

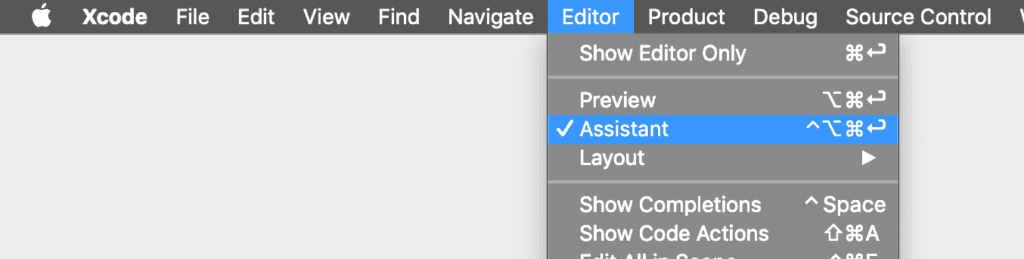
The Assistant Editor view, which displays the file that goes with the one you're now looking at. It appears to be a two-pane view.

When working with Storyboards, the Assistant Editor can help you connect user interface components from the Storyboard view to the code file so you may edit them using code.

If you're looking at a view in the Storyboard, for example, the right pane will display you the view's related class file. Each window also has its own jump bar, which you may use to alter which file is displayed in each pane.



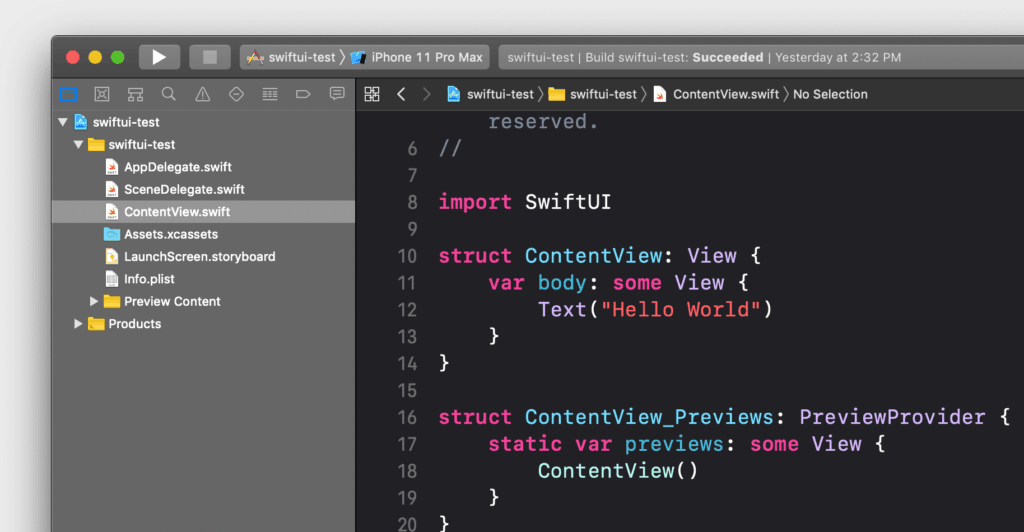
You can access the Assistant Editor by going to the Editor menu and selecting Assistant.



**How to use SwiftUI and the preview canvas**

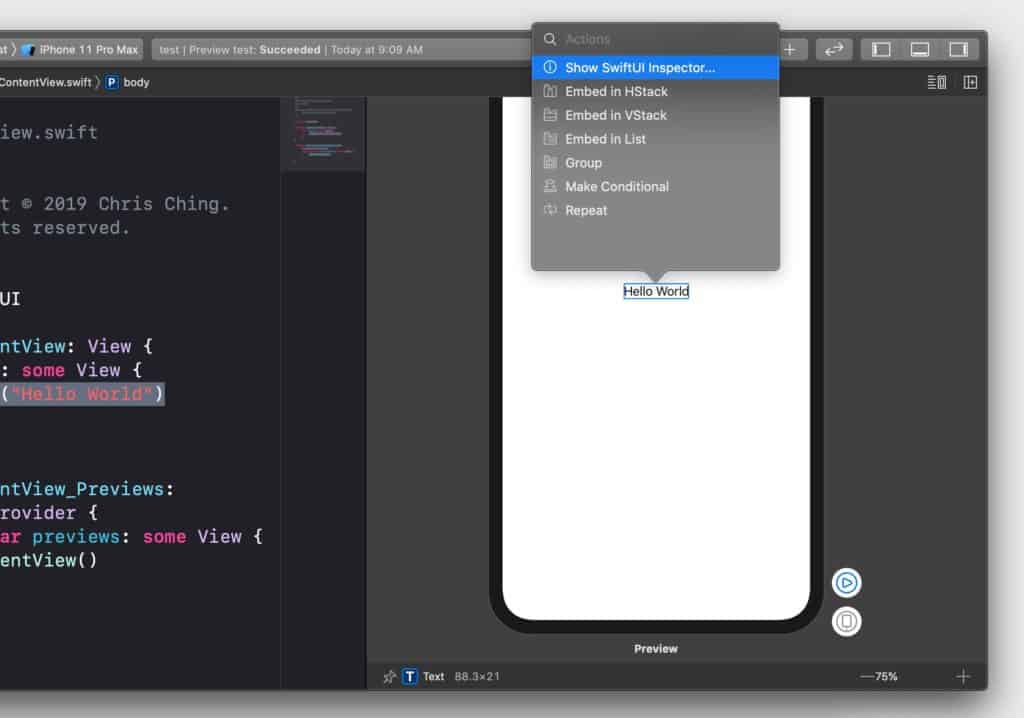
If you chose SwiftUI as your user interface option when creating your Xcode project, instead of a Main.Storyboard file, you'll get a ContentView.

swift file. You'll create your first SwiftUI user interface in this file.

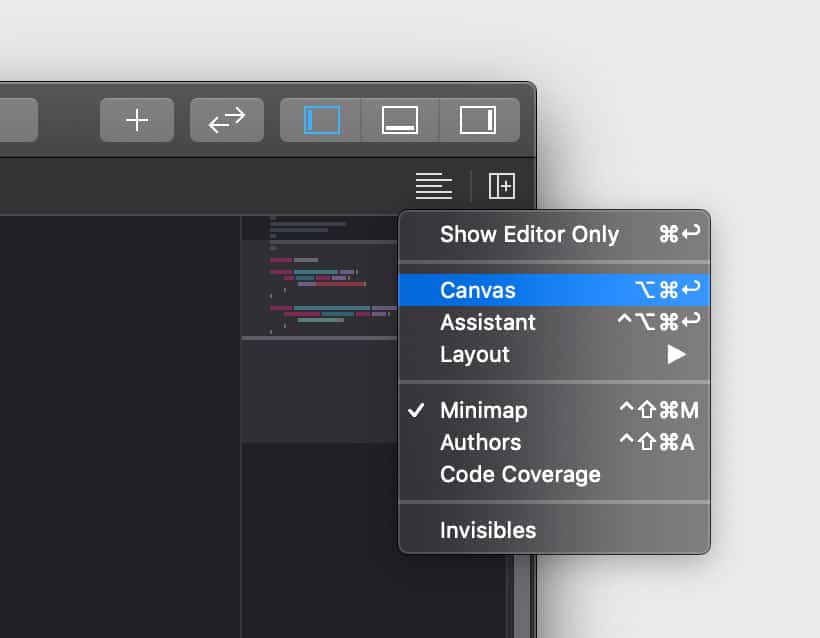


**The Preview Canvas**

The canvas is a pane that sits next to your code editor and displays a live preview of your user interface.

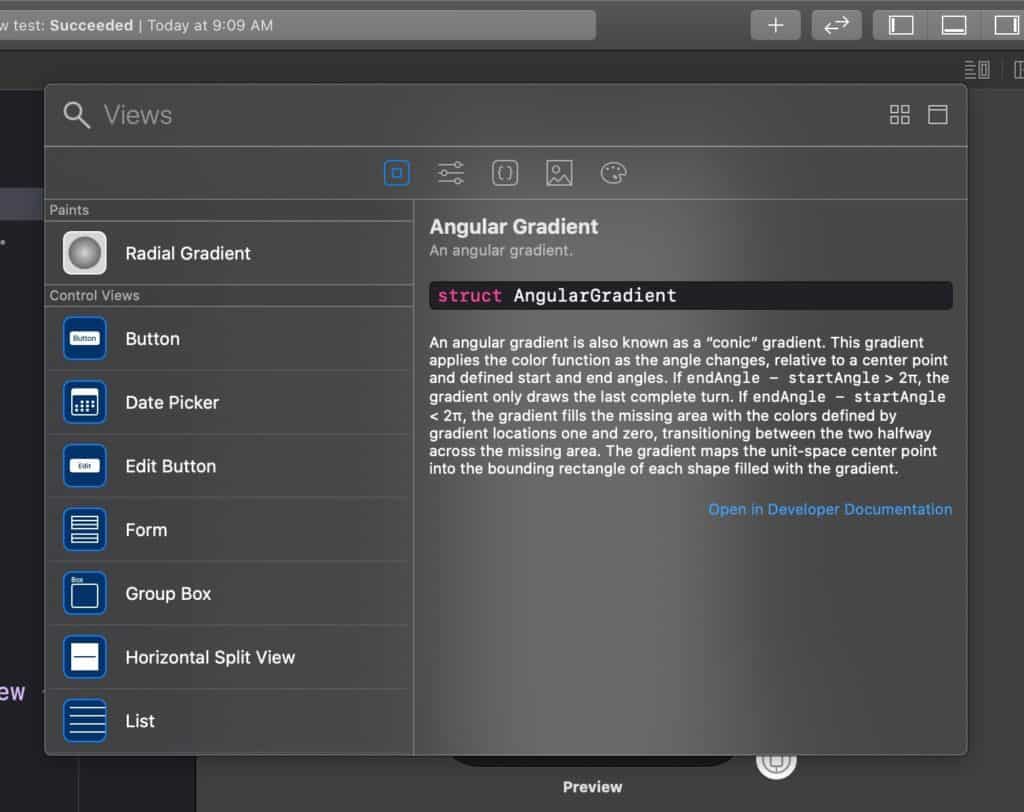


If you don't see it, select Canvas from the Editor menu. It requires macOS 10.15 or later to run. If you don't have that, you may still use the iOS Simulator to test your app's user interface.



**Using the object Library for SwiftUI**

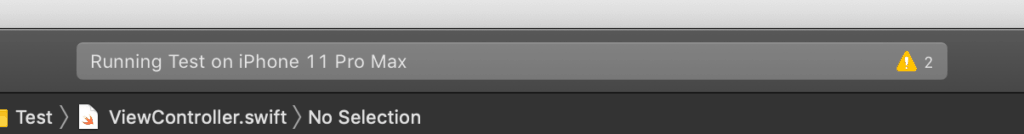
The Object Library displays a list of SwiftUI items that you can use in your user interface. You can search for what you need, then drag and drop it into the code editor to have the code generated for you.



We won't go into detail about how to utilise SwiftUI to create UIs because it requires a lot of coding, and this is an Xcode tutorial.

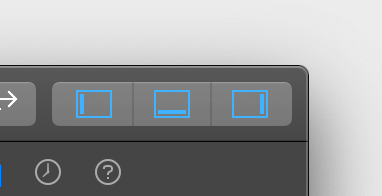
**The XCode Status Bar**

The status bar displays what Xcode is doing right now. If your project contains issues or warnings, you'll notice small signs for them.



**The objective Library and hiding/showing panes**

Finally, we have the library button (which we discussed in the user interface portion of this book), the Code Review button (which is used to review code changes if you're using source control), and buttons to toggle other panels on and off in the far-right side of the toolbar.

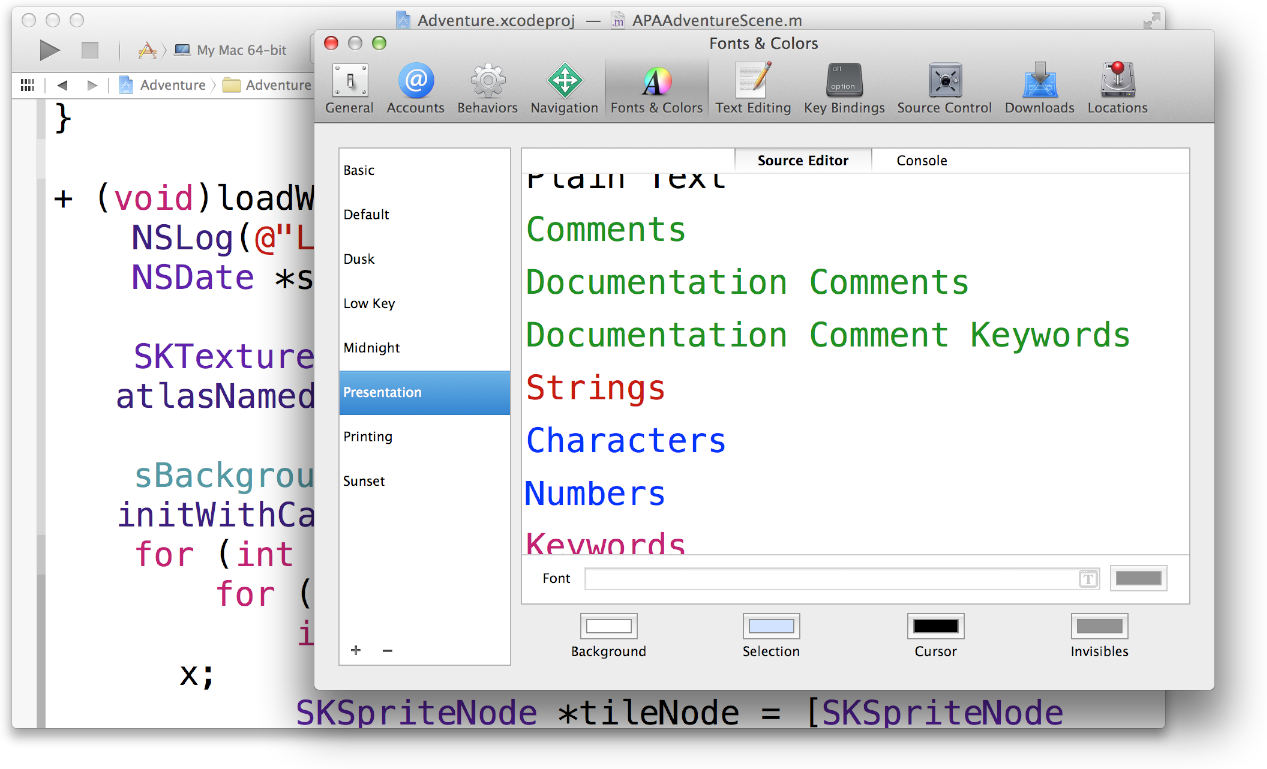


**Customizing the Editor**

Choosing syntax-aware fonts and Text Colors

Each token or string—for example, each comment, keyword, and class name declared in the project—is assigned a syntactic label by Xcode, which parses code based on the language. To make it easier to comprehend the code, Xcode gives a colour and font to each syntactic type.

By going to Xcode > Preferences and then Fonts & Colors, you may choose from a variety of font and colour themes. The Presentation theme, for example, boosts font sizes to make text easier to read when projected on a screen. You may even make your own fonts and colour schemes.



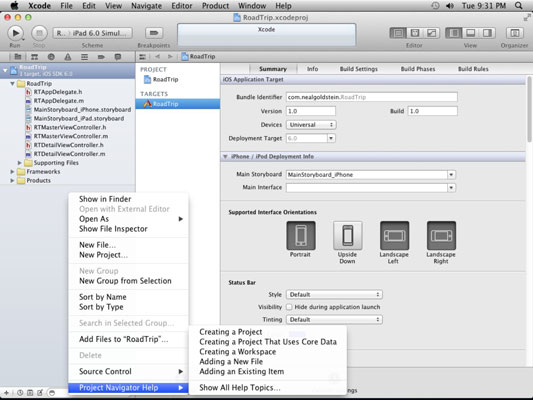
Customizing Editing and Indenting Options

You can customise the source editing and indenting settings to your liking. To change options like these, go to Xcode > Preferences and pick Text Editing:

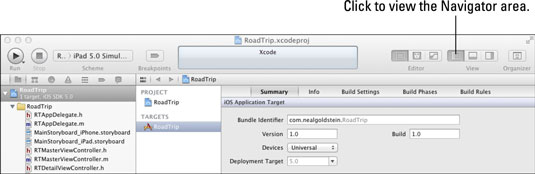
* In the source editor gutter, show line numbers.
* As you type, closing braces will appear automatically.
* While entering code, suggest code completions.
* To indent, use spaces or tabs.
* Lines with a soft wrap.
* Indent using syntax-aware indenting.

**Project Navigator in Xcode**

To create an iOS app, you must operate inside the context of an Xcode project. The Project navigator appears in the Xcode workspace when you've finished creating your project in Xcode.

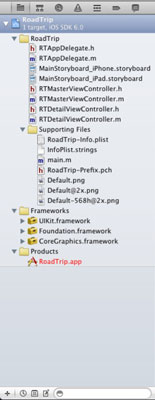


The Navigator area is an optional section on the left side of the Workspace window where you can use the Navigator picker to load several navigators, including the Project navigator. Click the left View selector button in the workspace toolbar, as shown in this image, to conceal or expose the Navigator area.



The Navigator selecting bar, the Content area, and the Filter bar are all found in the Navigator section. Other features particular to the chosen navigator can also be included. You may use the Project navigator to add, delete, group, and otherwise manage files in your project, as well as select a file to read or edit in the Editor area. (The proper editor will appear depending on whatever file you select.)

In this figure, for example, all the disclosure triangles are open so that the Project navigator displays all the files in the project.



You can use the Filter bar to limit what is displayed, such as recently changed files, unsaved files, or filenames.

**Xcode Development Tools**

Xcode is a set of tools used by developers to create programmes for Apple's platforms. Xcode can help you manage the whole development process, from app creation to testing, optimization, and submission to the App Store.

You can use Xcode to launch its development tools, or you can use the Xcode > Open Developer Tool menu to launch them independently:

When a real device isn't available, use Simulator to quickly prototype and test your app in a virtual environment. Simulator creates distinct settings, files, and operating system versions for iPhone, iPad, Apple Watch, and Apple TV devices.

Instruments may be used to profile and analyse your programme, as well as to optimise speed and identify memory issues. Instruments is a tool that takes data and delivers the findings using a variety of techniques.

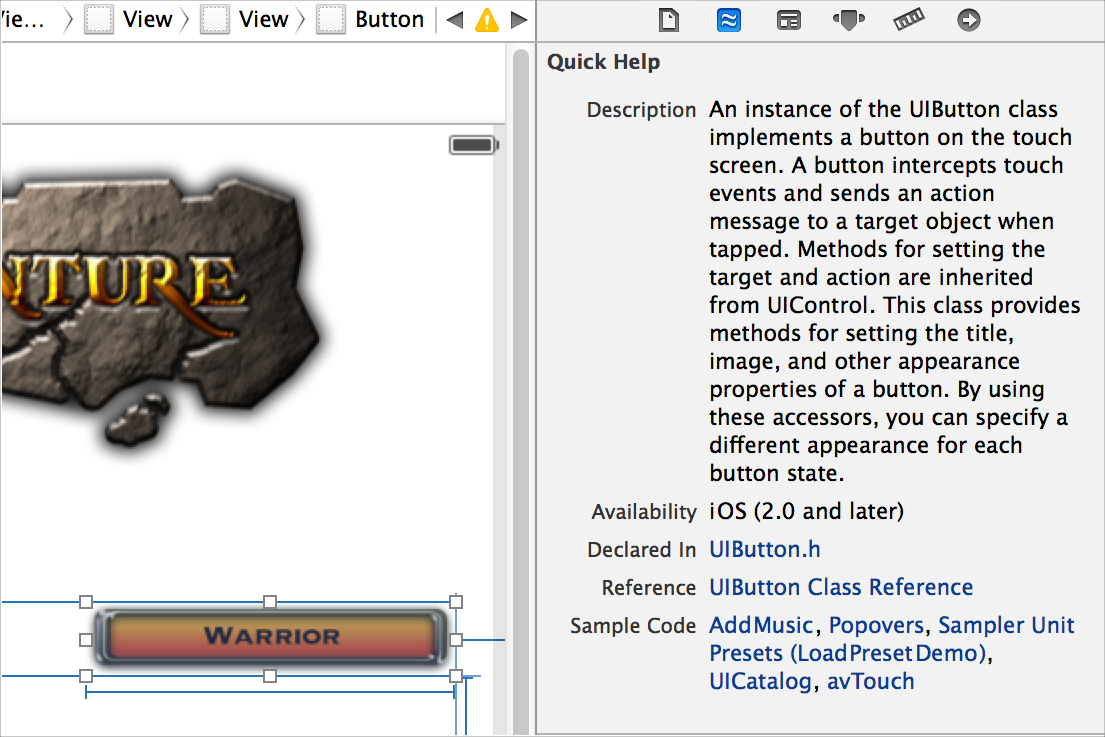
Create and train bespoke machine learning models for your app using Create ML.

Create 3D compositions and augmented reality (AR) experiences using Reality Composer.

**Quick Help in Xcode**

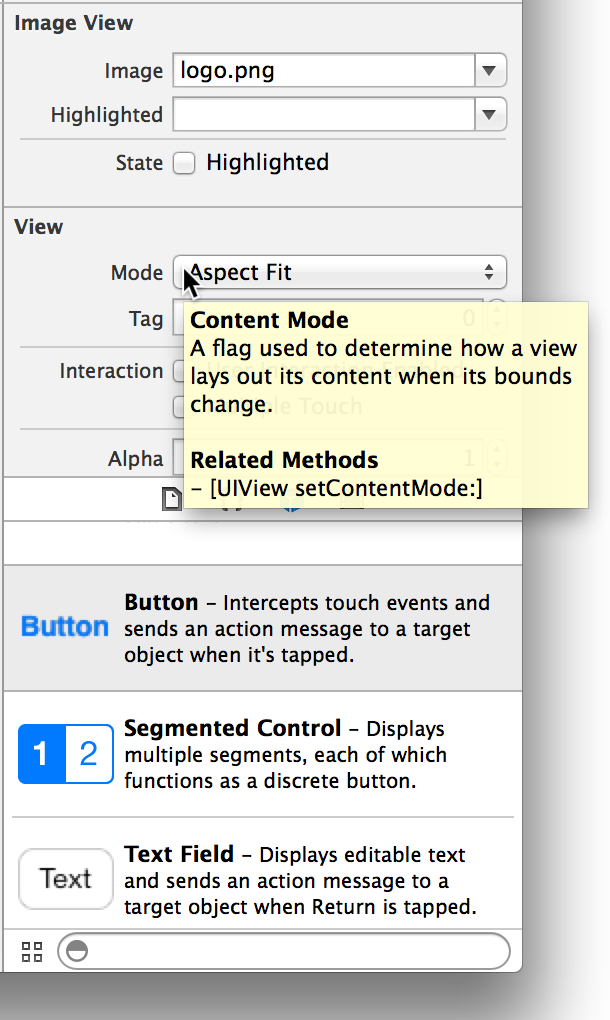
Without leaving Interface Builder, you may get concise class reference documentation for a user interface object. By pressing the right button in the view selection in the toolbar when a file is open in Interface Builder, you can access the utilities area.

In the inspector bar, click the Quick Help button. Click the object you wish to learn more about in Interface Builder. The utilities area's inspector pane displays documentation.



Click the title of the reference document listed in Quick Help for detailed reference information about the object. In the documentation viewer window, the reference document appears. By clicking the titles of relevant programming manuals, sample code, and other related resources in Quick Help, you can open them.

Move the pointer over a control in an inspector to get more information about the settings you configure there. A help tag is displayed.



Bibliography:

1. <https://codewithchris.com/xcode-tutorial/#1-what-is-xcode>

2. <https://developer.apple.com/xcode/features/>

3. <https://developer.apple.com/library/archive/documentation/ToolsLanguages/Conceptual/Xcode_Overview/CustomizingtheEditor.html>

4. <https://www.dummies.com/web-design-development/mobile-apps/overview-of-the-project-navigator-in-xcode/>

5. <https://developer.apple.com/documentation/xcode/>

6. <https://developer.apple.com/library/archive/documentation/ToolsLanguages/Conceptual/Xcode_Overview/LookingupObjectDocumentation.html>