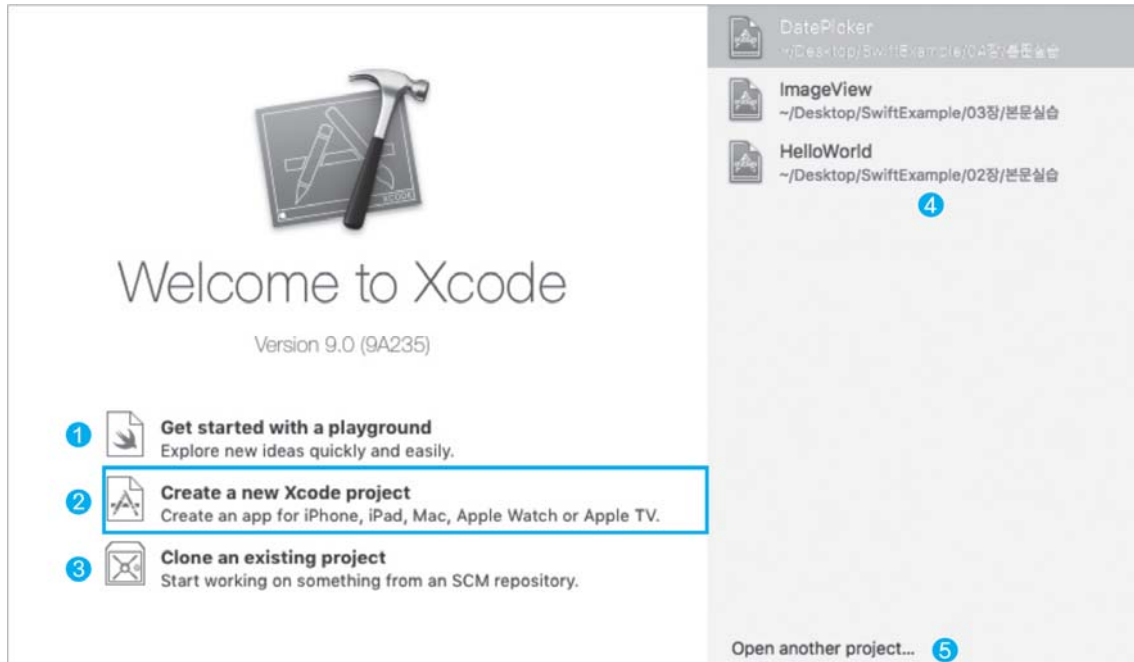


## 02. Hello World 앱 만들며 Xcode 에 완벽 적응하기

### 1. Xcode 실행하기



1) 스위프트 언어를 간단하게 연습할 수 있는 플레이그라운드를 실행한다. 플레이그라운드는 코딩하는 즉시 해당 코드에 대한 결과를 오른쪽 결과창에 표시해주므로 결과를 바로 확인하면서 코딩할 수 있다.

2) 아이폰 아이패드 맥앱을 만들기 위한 새로운 Xcode 프로젝트를 생성할 수 있다. 일반적으로 가장 많이 사용한다.

3) SVN 이나 git 과 같은 버전 관리 도구로 연결하여 기존소스를 가져올 수 있다.

4) 최근에 사용한 프로젝트 중에 선택해서 불러올 수 있다.

5) 최근 프로젝트에 포함되지 않은 다른 프로젝트를 불러올 수 있다.

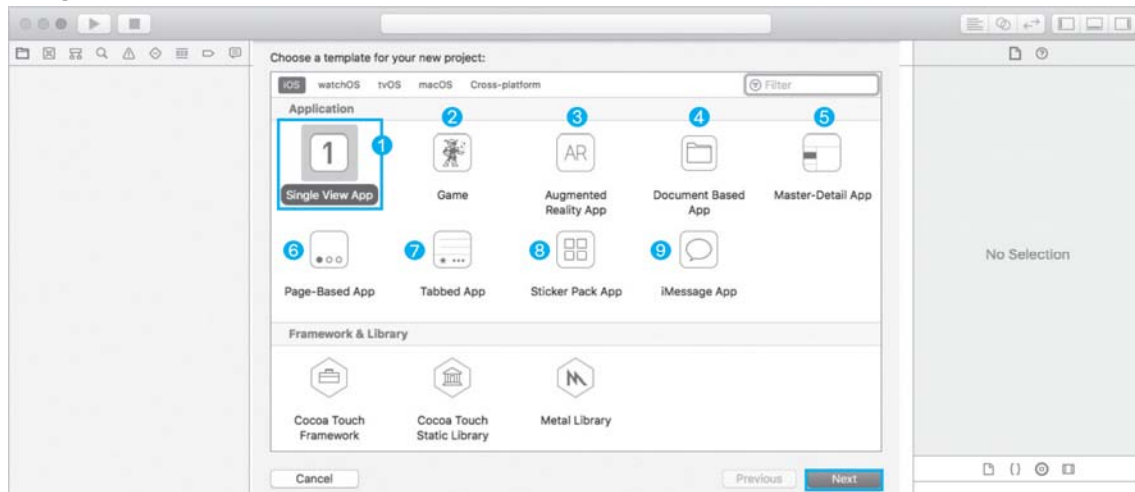
## 2. 템플릿 선택하기

새로운 프로젝트를 시작할 때 템플릿을 선택하는 창이 나타난다.

각 템플릿에는 용도에 맞는 앱을 개발하는 데 필요한 기본 틀이 설정되어 있다.

개발자는 이 기본 틀에 여러가지 기능을 추가하여 앱을 만들면 된다.

[Single View Application]을 선택한 후 [Next] 버튼을 클릭합니다.



1) 뷰를 사용하는 앱을 개발할 때 사용하는 템플릿이다,  
기본적으로 하나의 뷰가 나타나며 필요에 따라 새로운 뷰를 추가하여 만들 수 있다.  
일반적으로 가장 많이 사용하는 템플릿이다.

2) 게임 앱을 개발할 때 사용하는 템플릿이다. 그래픽처리를 위한 OpenGL 게임뷰를 생성해준다.

3) 증감현실 앱을 개발할 때 사용하는 템플릿이다.

4) 데이터를 저장할 수 있는 문서를 기반으로 개발할 때 사용하는 템플릿

5) 목록 기반의 앱을 개발할 때 사용하는 템플릿이다.

아이폰의 메모 앱처럼 목록을 보여주고 목록 중 하나를 선택하면 해당 목록의 상세 내용을 볼수 있는 앱을 만들 수 있다.

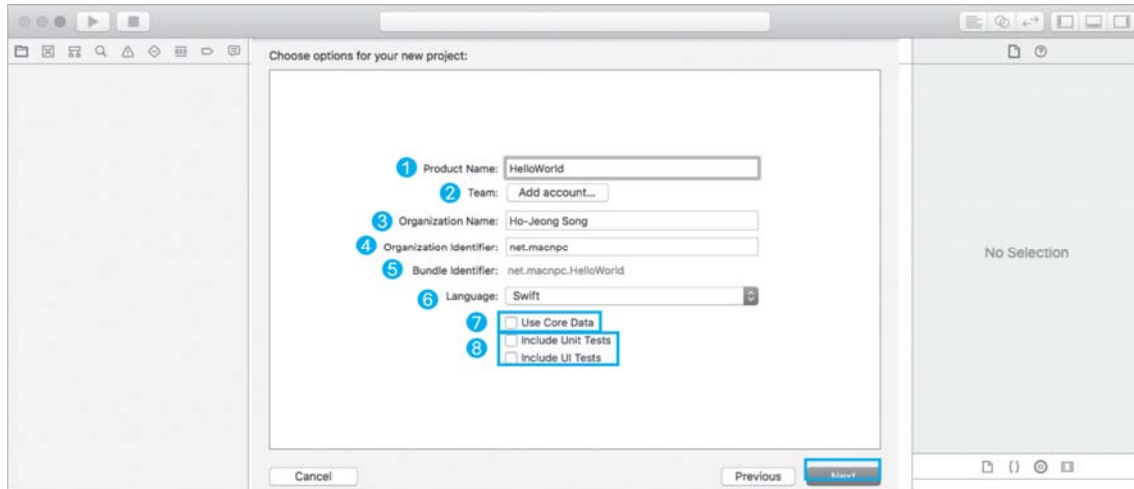
6) 여러개의 페이지로 이루어진 앱을 개발할 때 사용하는 템플릿이다. 페이지를 넘기는 효과를 지원하므로 앨범 앱이나 E-Book 앱을 만들 때 적절한 템플릿이다,

7) 아랫부분의 탭바를 사용하여 뷰를 이동하는 앱을 개발할 때 사용하는 템플릿이다. 전화 앱이나 시계 앱처럼 탭을 사용하여 뷰를 이동하는 앱을 개발할 때 적합하다,

8) 스티커 팩 앱을 개발할 때 사용하는 템플릿이다.

9) 아이메세지 앱을 개발할 때 사용하는 템플릿이다,

### 3. 프로젝트 정보 설정



1) 개발하려고 하는 앱의 이름을 입력한다.

여기서는 'HelloWorld'를 입력한다.

2) 개발자 프로그램에 등록된 ID 또는 팀을 입력한다.

개발자 인증서가 등록되어 있으면 여기서 선택할 수 있다.

처음 시작할 때는 시뮬레이션을 사용할 것이므로 입력하지 않아도 된다.

3) 프로그램을 관리하는 사람의 이름을 입력한다. 일반적으로 개발자의 이름을 입력한다,

4) 조직의 식별자를 입력한다, 일반적으로 개인이나 조직의 도메인주소(URL)를 역순으로 입력한다. 조직 식별자는 앱 식별자를 만드는 데 사용되므로 공부를 하는 동안에는 'com.yourcompany'등의 아무 URL 을 입력해도 무관하지만 앱을 앱스토어에 등록하려면 개인이나 조직이 소유하고 있는 유일한 URL 이 있어야 한다.

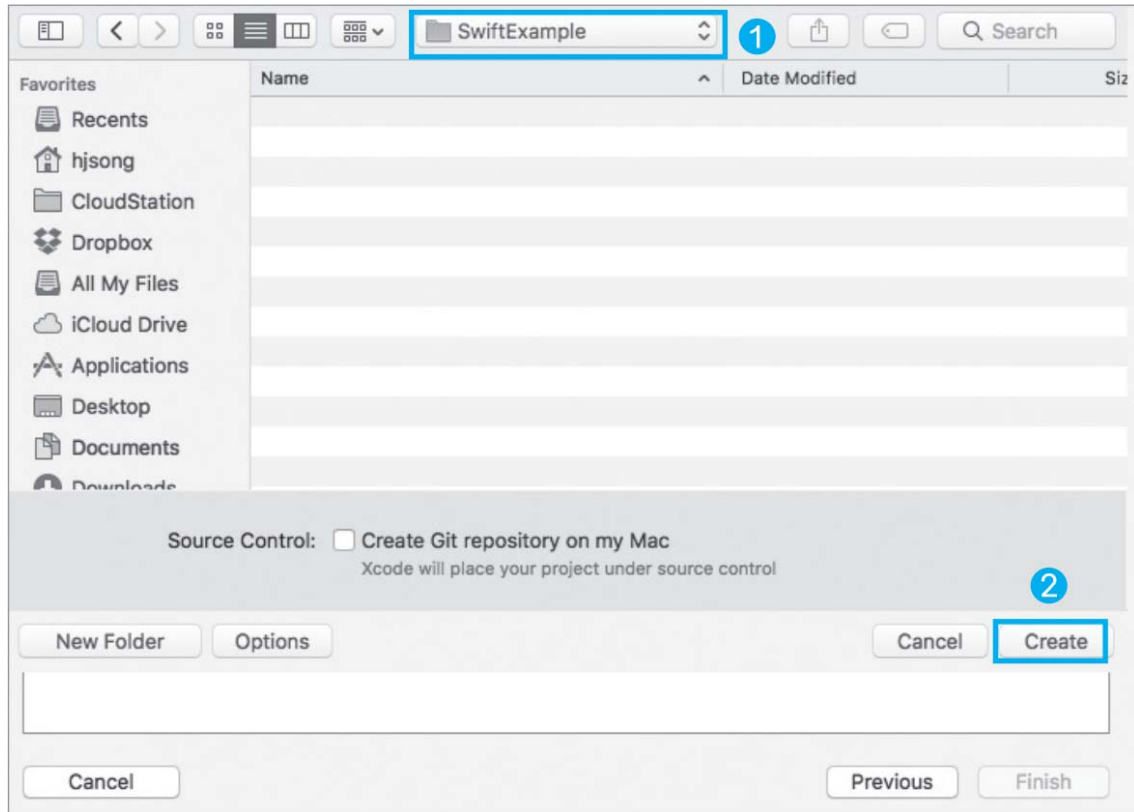
5) 식별자 이다. 'Company Identifier.Product Name'으로 자동으로 생성된다. 앱 식별자는 앱을 앱스토어에 등록할 때 다른 앱들과 구분하는 용도로 사용하므로 유일한 식별자를 사용하여 앱을 등록해야 한다.

6) 앱개발에 사용할 언어를 선택한다. 스위프트와 오브젝트-C 중에 하나를 선택할 수 있다. 여기서는 [Swift]를 선택한다.

7) iOS 에서 제공하는 데이터 관리 툴킷의 사용여부를 선택한다.

8) 앱의 동작 등을 자동으로 테스트할 때 사용한다.

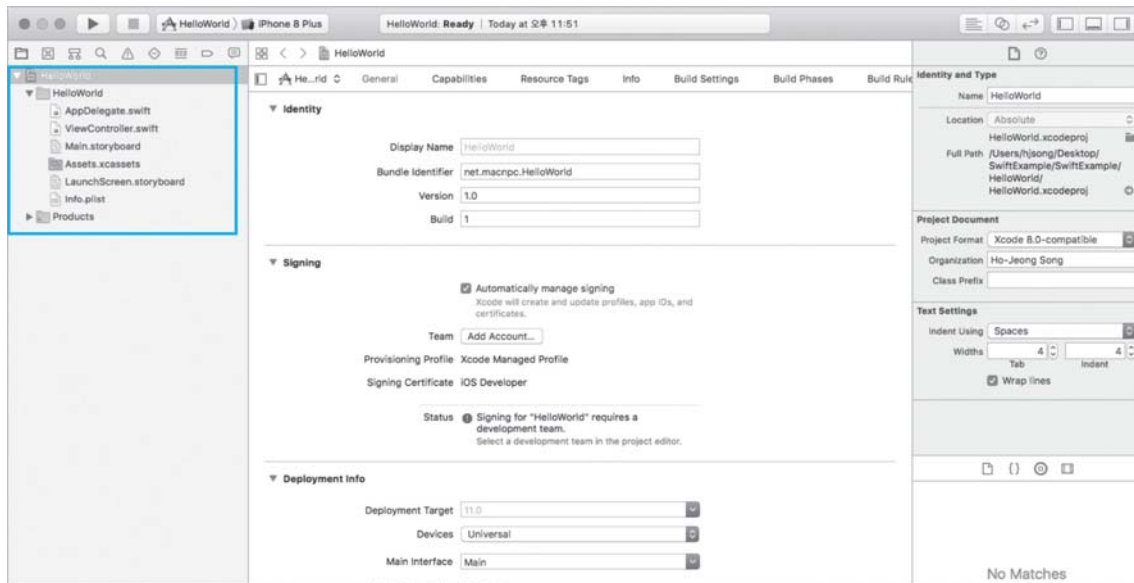
4. 프로젝트를 저장할 작업 폴더를 선택한 후 [Create]를 클릭한다.



1)에서 프로젝트를 저장할 폴더를 지정해주면 된다.

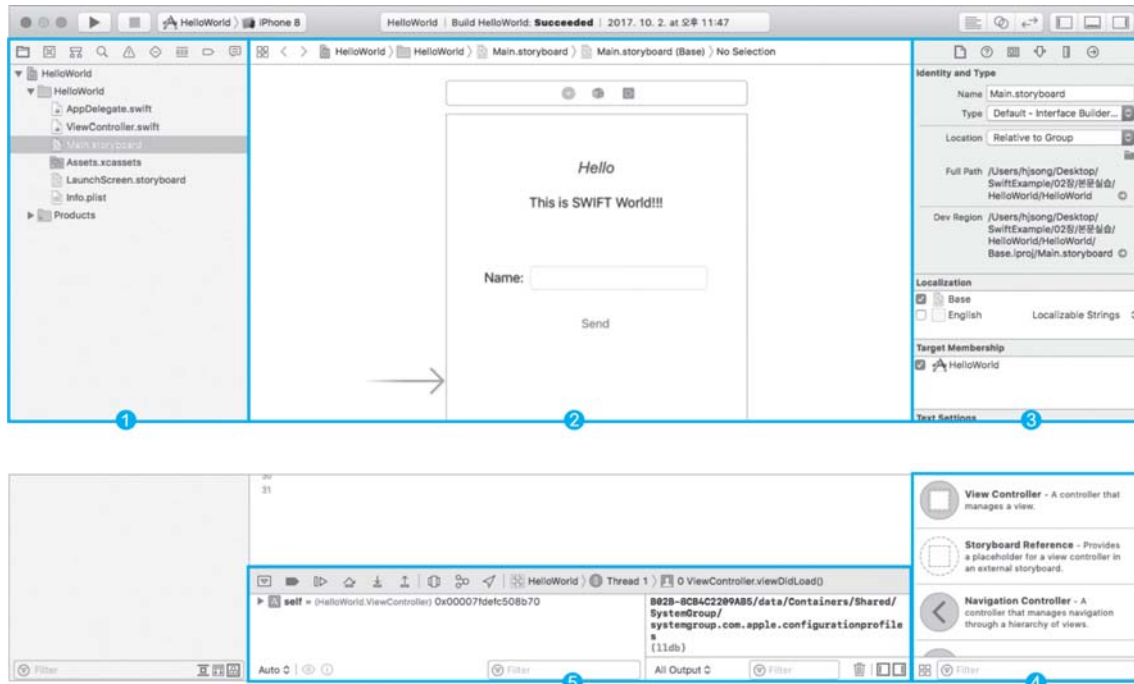
소스 컨트롤을 사용할 지에 대한 선택항목이 나타날 것이다, 소스컨트롤은 Git 이라는 소스 버전 관리도구를 사용하여 프로젝트를 관리하거나 다른 사람들과 협업하고 싶을 때 사용하는 기능이다.

5. 이렇게 새 프로젝트가 만들어졌다. 화면 왼쪽의 내비게이터 영역에 프로젝트 파일들이 있는 것을 확인 할 수 있다.



- 1) AppDelegate.swift : 앱의 실행주기(Life Cycle)를 관리하는 내용의 스위프트 소스코드가 들어 있는 클래스 파일이다. 앱을 실행하거나 종료 또는 백그라운드 실행할 때 하는 일들을 관리한다. 일반적으로 초보 단계일 때는 프로그래머가 직접 코딩하지 않아도 된다.
- 2) ViewController.swift : 화면에 보이는 뷰에서 처리하는 내용의 스위프트 소스 코드를 담고 있는 클래스 파일이다. 일반적으로 프로그래머는 이 파일에서 코딩을 하게 되며 뷰 하나당 클래스 하나가 대응된다, 그러므로 스토리보드에서 여러 개의 뷰를 추가하면 뷰의 개수만큼 뷰 컨트롤러 클래스 파일이 필요하다.
- 3) Main.storyboard : 앱의 내용을 시각적으로 쉽게 이해하고 프로그래밍할 수 있도록 그림으로 표현한 파일이다. 이 스토리보드를 통해 화면에 보이는 내용 및 뷰와 뷰간의 연결 관계등을 표현할 수 있다.
- 4) Assets.xcassets : 앱의 아이콘을 보관하는 저장소이다. 이곳에서 앱 아이콘을 설정해야 원하는 앱 아이콘으로 표시할 수 있다.
- 5) LaunchScreen.storyboard : 앱이 실행될 때 잠시 나타나는 스플래시 화면을 만드는 스토리보드이다.
- 6) info.plist : 앱이 실행되는 데 필요한 정보를 저장하고 있는 파일이다.

## Xcode의 화면 구성 살펴보기



1) 내비게이션 영역 : 프로젝트 내비게이터, 심볼 내비게이터, 심볼 내비게이터 검색 내비게이터, 이슈 내비게이터, 테스트 내비게이터, 디버그 내비게이터, 브레이크 포인트 내비게이터, 리포트 내비게이터 등의 정보를 나타내 주는 영역이다.

각 항목들은 한번 클릭하면 가운데의 편집기 영역에 나타나고, 더블 클릭하면 새로운 창이 열리면서 나타난다.

2) 편집기 영역 : 소스 파일을 열어 소스를 직접 입력하거나 스토리보드를 이용하여 화면을 디자인할 수 있는 영역이다.

3) 인스펙터 영역 : 스토리보드를 편집할 때 버튼, 컨트롤러, 뷰 등 모든 객체의 속성을 편집할 수 있는 영역이다.

4) 라이브러리 영역 : 스토리보드에서 사용할 수 있는 버튼, 컨트롤러, 뷰 등의 모든 객체를 볼 수 있고 사용할 수 있는 영역이다.

5) 디버그 영역 : 버그를 찾아 수정하는 과정인 디버깅을 진행할 때 원하는 변수의 값을 확인하거나 테스트할 목적으로 사용한 입출력 내용이 출력되는 영역이다.

디버그 창은 왼쪽의 변수 영역과 오른쪽의 콘솔영역으로 구성되어 있다.

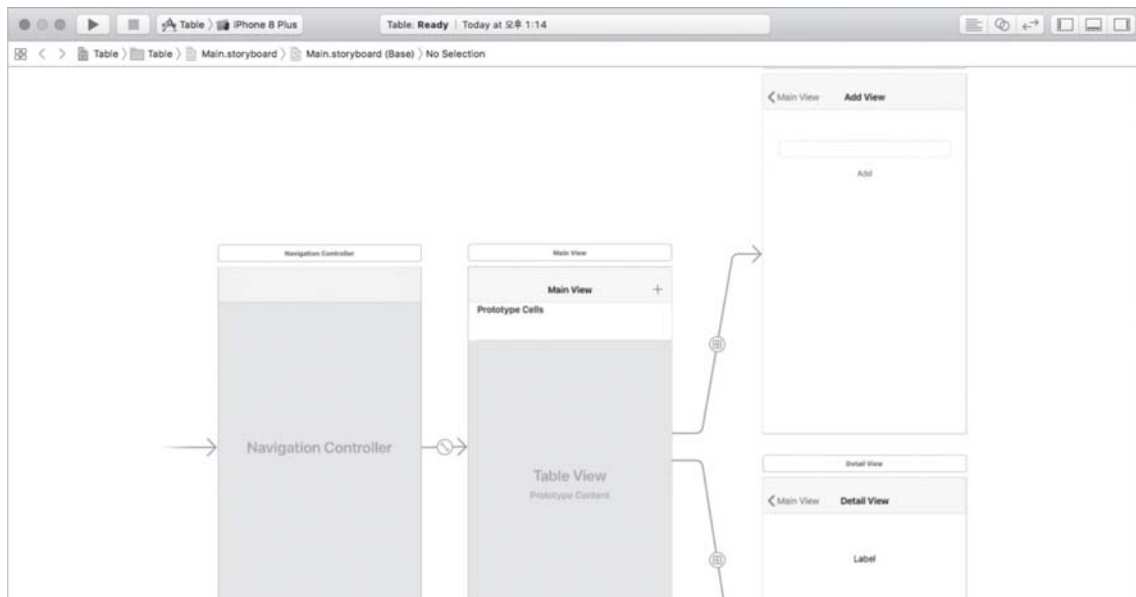
## 02-2 스토리보드로 작업하기 위한 기본 환경 구성하기

스토리보드란?

앱의 화면구성을 시각적이고 직관적으로 구성할 수 있게 지원하는 기능입니다.

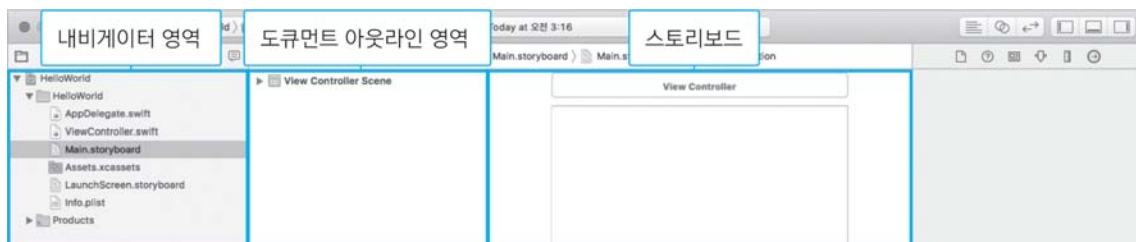
즉, Xcode 에서 만들고자 하는 앱이 어떤 모양으로 화면에 구성되어 있고, 버튼을 누르거나 화면을 스와이프하는 등의 특정액션을 취했을 때 어떤 방식으로 화면 간 전환이 이루어지는 지를 보여 준다.

화면간의 흐름 및 전체적인 모양을 시각적인 방식으로 연결하고 표현해 줌으로써 직관적으로 앱의 흐름을 확인할 수 있게 만든 기능이다.



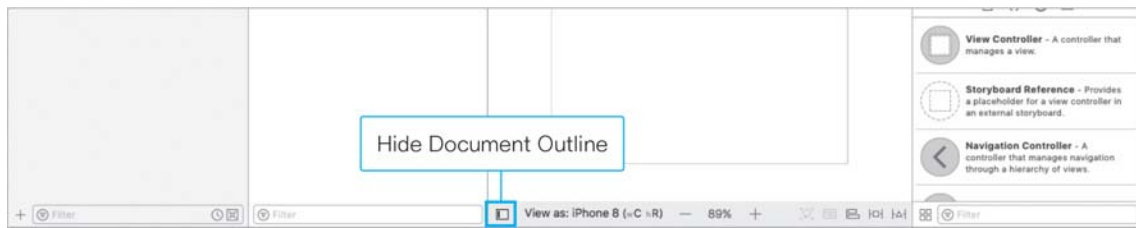
스토리보드 작업화면 조정하기

1. Xcode 화면이 처음 열렸을 때 화면 왼쪽의 내비게이터 영역에서 [Main.storyboard]를 선택하면 스토리보드가 화면에 나타난다.

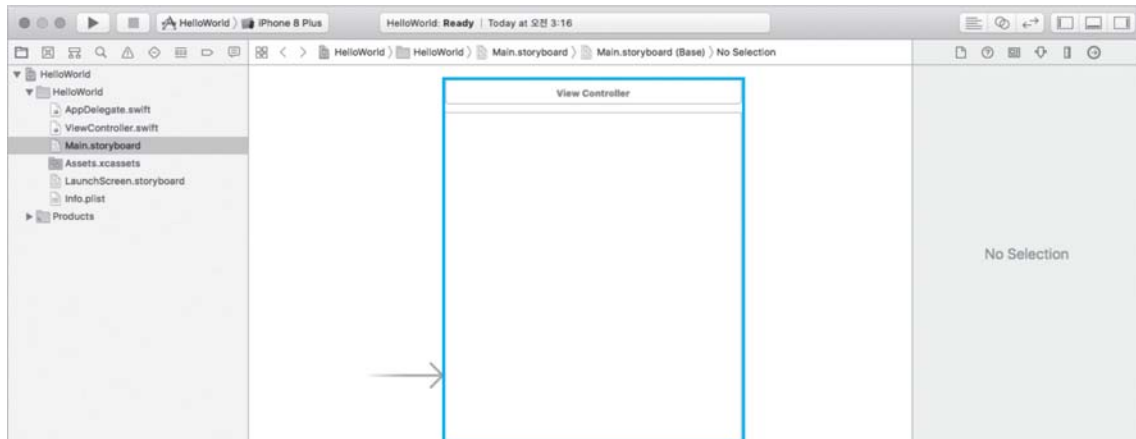


2. 내비게이터 영역과 도큐먼트영역이 꽤 많은 부분을 차지하다 보니 스토리보드 영역이 상대적으로 작게 보인다, 이럴 때 '내비게이터' 영역 혹은 '도큐먼트 아웃라인' 영역을 줄이거나 닫아서 스토리보드가 잘 보이게 할 수 있다.

도큐먼트 아웃라인 영역은 [Hide Document Outline]버튼을 클릭하여 닫을 수 있다.



도큐먼트 아웃라인 영역이 사라지면서 스토리보드가 크게 보입니다

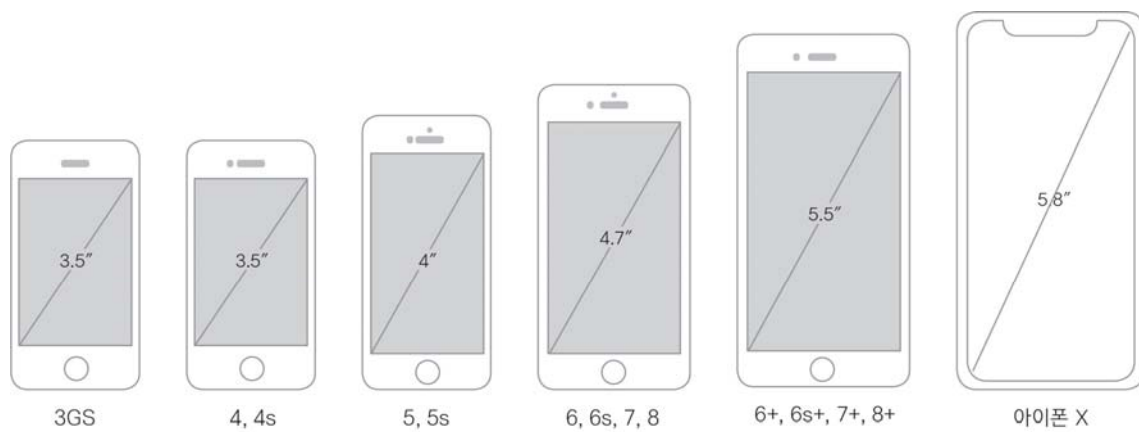


- 1) 기본편집기 (Standard editor) 버튼 – 편집기 영역에 기본 편집기를 연다.
- 2) 보조편집기 (Assistant editor) 버튼 – 편집기 영역에 보조 편집기를 연다.
- 3) 버전편집기 (Version editor) 버튼 – 버전 편집기를 연다. 버전 편집기에서는 Source Control 기능을 이용할 경우 이전 버전과 현재 버전의 차이점을 확인 할 수 있다.
- 4) 내비게이터 영역 감춤/보임 (Hide or show the Navigator) 버튼 – 왼쪽의 내비게이터 영역을 감추거나 보여 준다.
- 5) 디버그 영역 감춤/보임 (Hide or show the Debug area) 버튼 – 아래쪽의 디버그 영역을 감추거나 보여준다.
- 6) 유틸리티 영역 감춤/보임 (Hide or show the Utilities) 버튼 – 오른쪽의 유틸리티 영역을 감추거나 보여준다.

어떤 기종에 맞춰 앱을 만들까? – 아이폰 버전별 화면 크기를 알아보자

아이폰은 3GS 부터 10+까지 다양한 크기의 화면으로 구성되어 있으며 각 버전별 화면 크기는 다음과 같다.



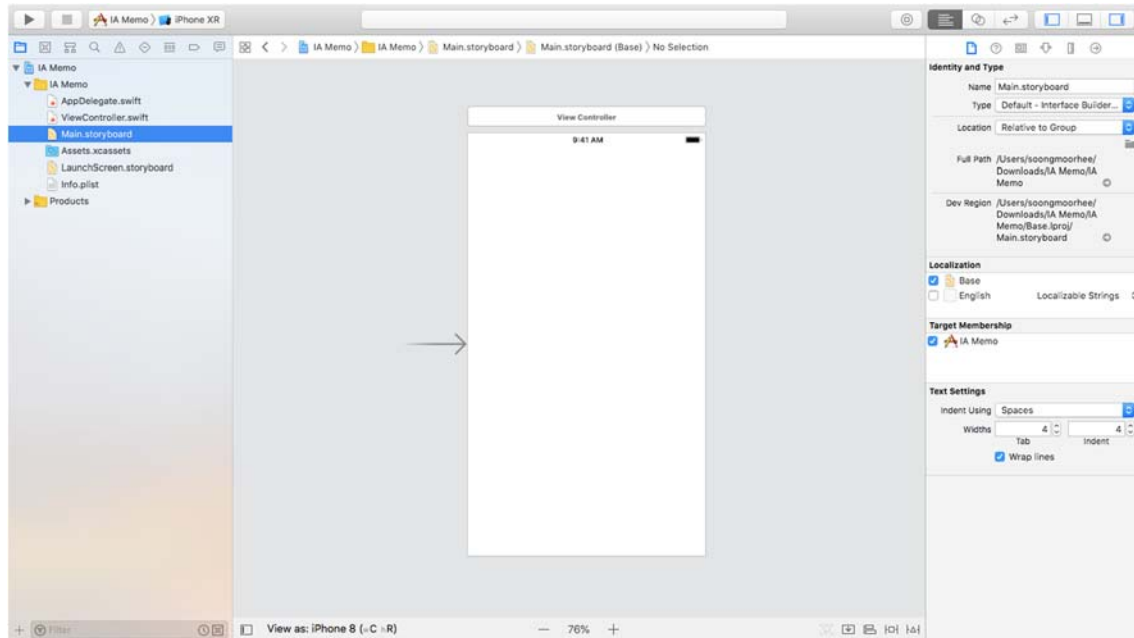


버전	화면 크기	버전	화면 크기
아이폰 3GS	3.5인치(320×480px)	아이폰 4, 4s	3.5인치(640×960px)
아이폰 5, 5s	4인치(640×1136px)	아이폰 6, 6s, 7, 8	4.7인치(750×1334px)
아이폰 6+, 6s+, 7+, 8+	5.5인치(1080×1920px)	아이폰 X	5.8인치(1125×2436px)
아이패드 에어 2	9.7인치(1536×2048px)	아이패드 프로	9.7인치(1536×2048px) 12.9인치(2732×2048px)
아이패드 미니 4	9.7인치(1536×2048px)		

## 02-3 스토리보드로 앱 화면 꾸미기


1) 네비게이터 메뉴에서 Main.storyboard 를 클릭하면 중앙에 화면이 뜬다.

스토리보드에서 간단한 외관을 꾸며 줄 수 있다.

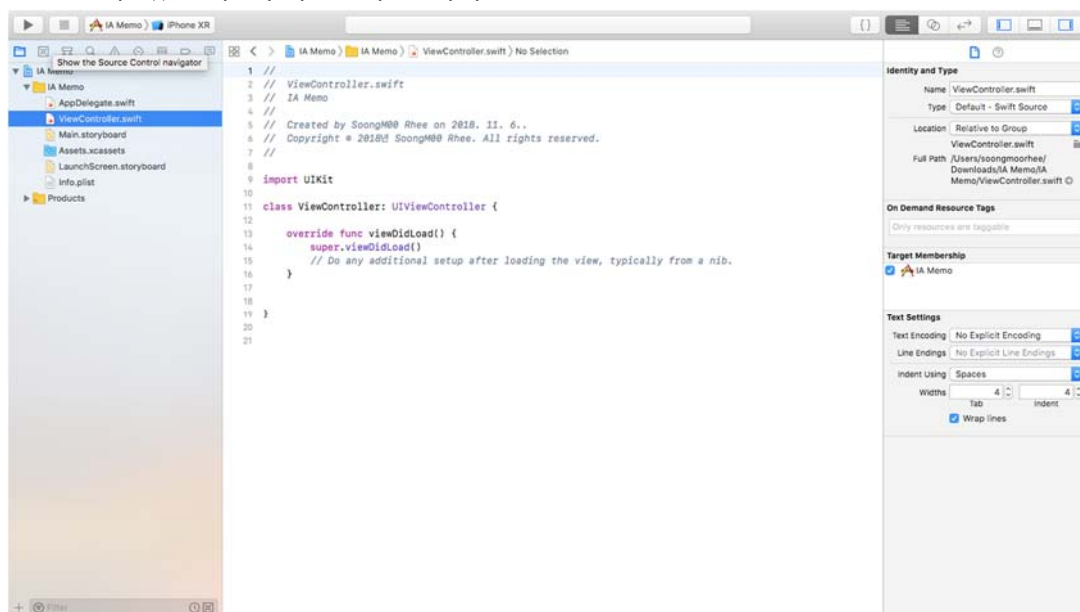


스토리보드에 가운데 있는 화살표(→)가 처음 시작을 알리는 것이다,

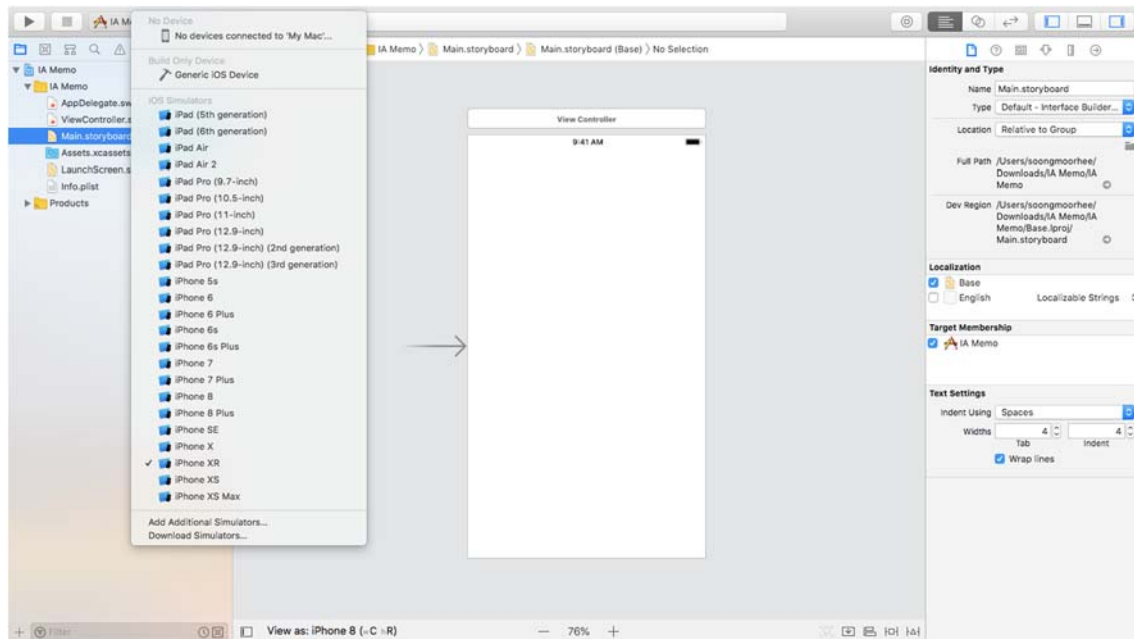
앱을 실행시키게 되면 현재 화면 부터 시작이 된다는 것을 알려 주는 것이다.

상단 오른쪽에 있는 여러가지 기능들을 가지고 있는 아이템을 선택할 수 있는  버튼을 클릭하여 아이템을 선택할 수 있다.

네비게이터 메뉴에 있는 .swift 라는 파일은 스토리보드에서 구현하지 못하는 내용들을 코딩할 수 있도록 해 주는 화면 이다.




상단에 있는 드롭 다운 메뉴는 어느 아이폰에서 테스트를 할 것인지를 선택할 수 있게 해준다.

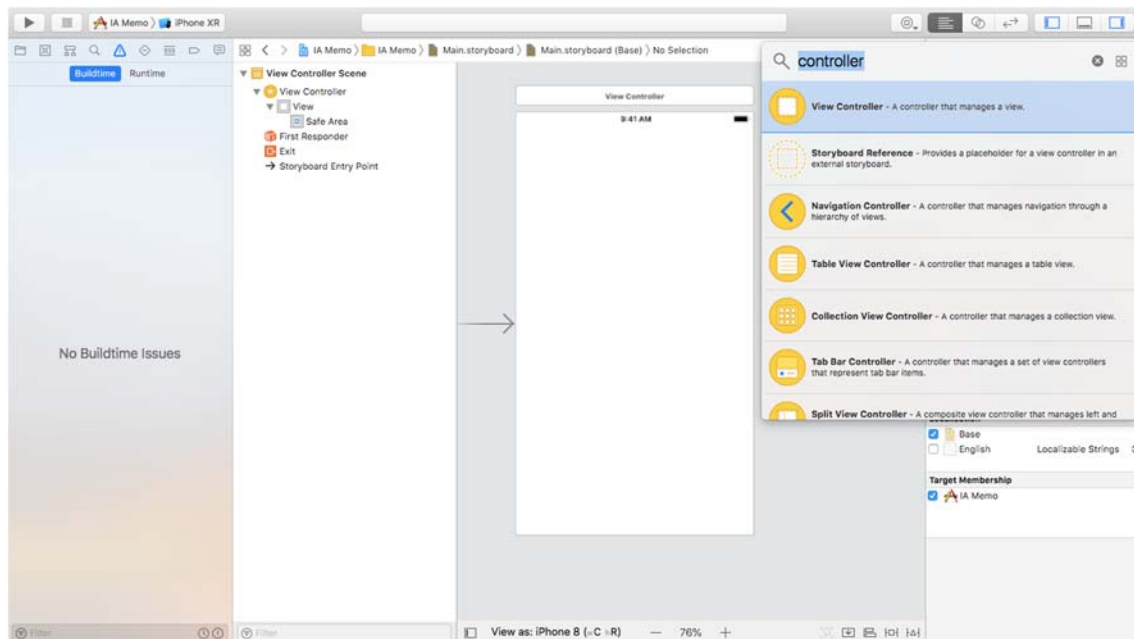


왼쪽에 있는 재생 버튼(▶)은 빌드 버튼으로서 지금까지 만든 앱을 테스트하기 위해서 실행하는 버튼이다. Simulator 라고 생성된 창으로 볼 수 있다.

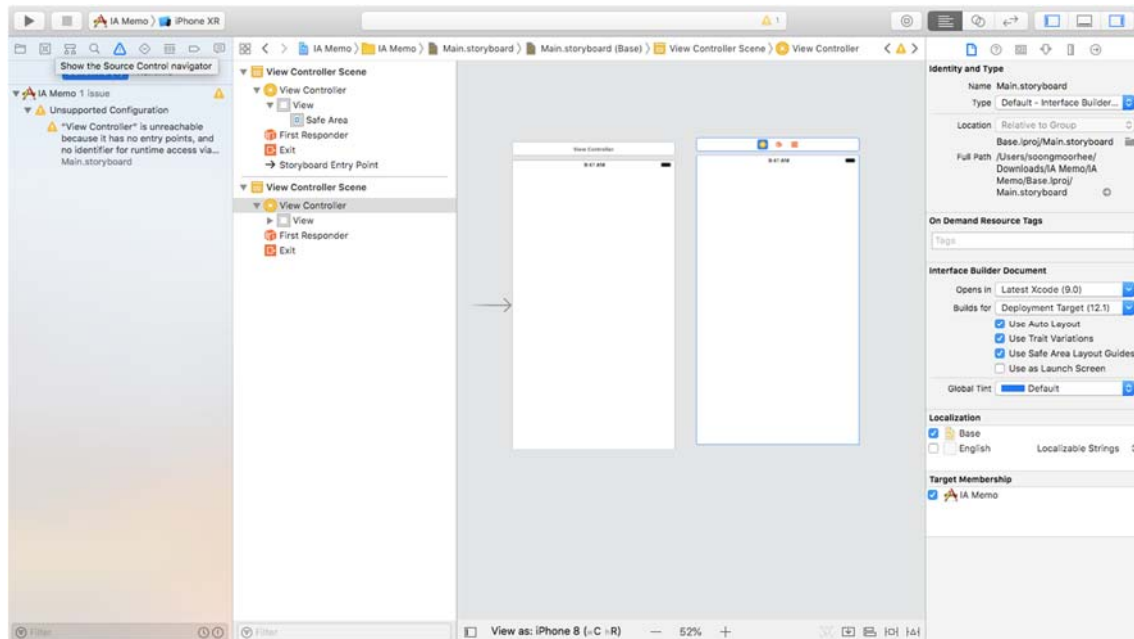
## 1. 스토리 보드로 화면 꾸며 보기

1) 먼저 스토리보드에 View Controller 를 하나 추가해보도록 한다.

아이템을 선택할 수 있는  버튼을 눌러서 검색 창에 Controller 라고 쳐서 아무기능도 하지 않고 화면만 보여 주는 View Controller 를 선택하여 드래그드롭을 한다.



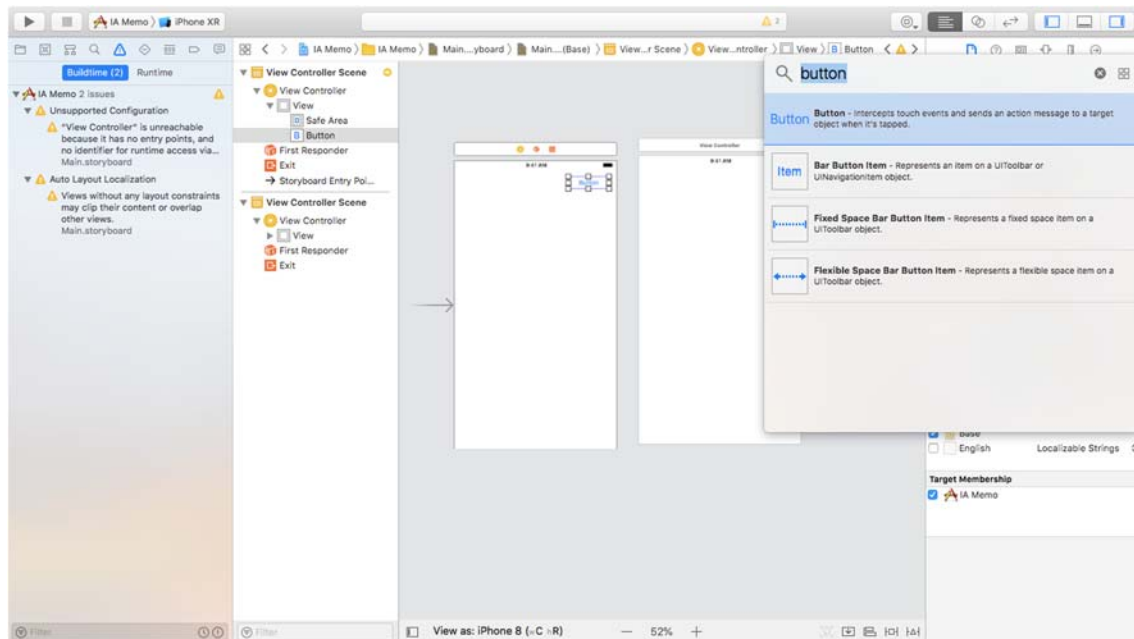
아이템을 선택할 수 있는 버튼을 눌러서 검색 창에 Controller 라고 쳐서 아무기능도 하지 않고 화면만 보여 주는 View Controller 를 선택하여 드래그드롭을 한다.



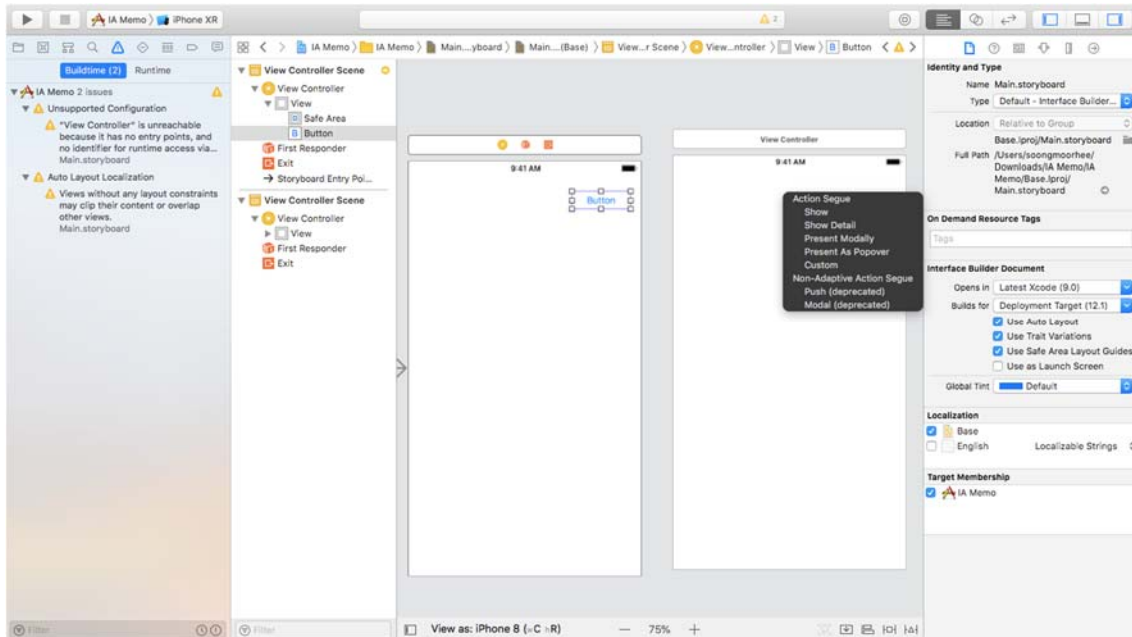
왼쪽에 있는 화살표가 제일 처음 보이는 화면을 나타내주고 있다.

두번째 화면은 첫번째 화면에서 어떠한 버튼을 누르면 보이도록 해야 할 것이다.

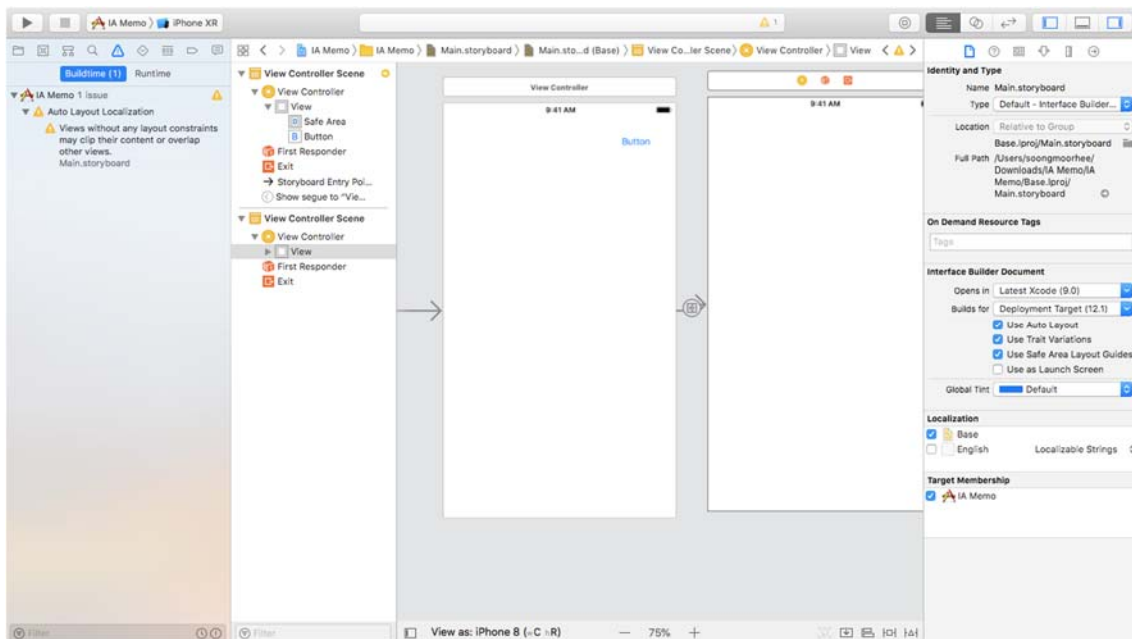
버튼을 눌러 검색창에서 Button 이라고 검색을 하여 드래그 드롭해서 적당한 위치에 놓는다..



추가한 버튼을 누르면 다음화면으로 넘어가도록 하기 위해서는 버튼을 클릭한 상태에서 control 키를 누르고 View Control 에 드래그를 해준다.



Action Segue 라는 창이 뜨는데 Show 를 누르면 화살표로 View Control 이 연결되는 것을 확인할 수 있다.

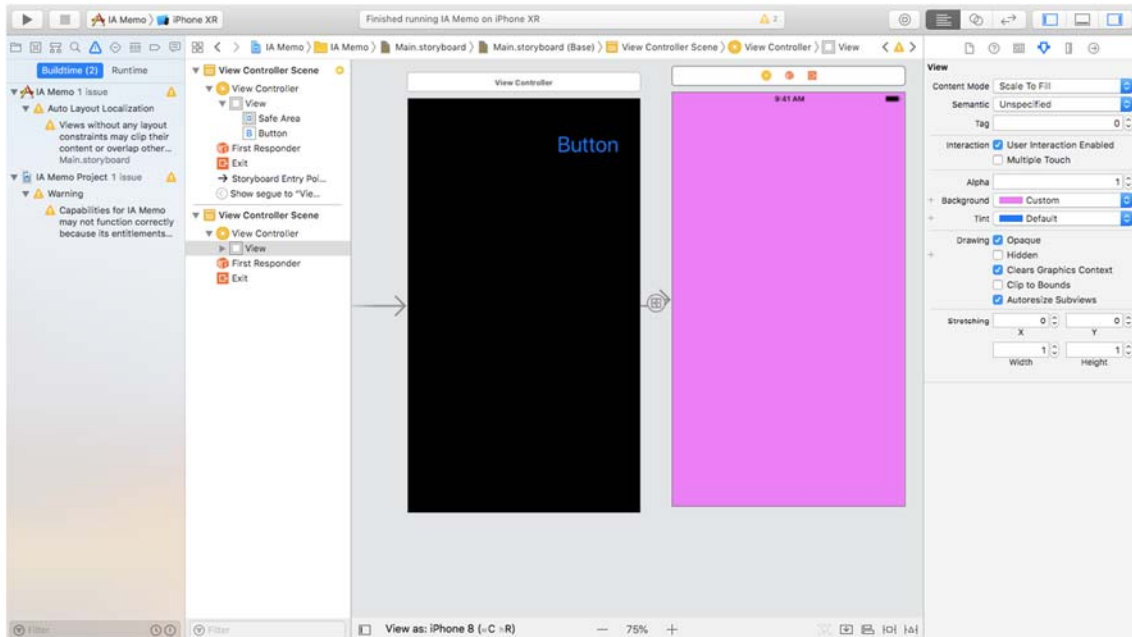


두 View Control 사이에 화살표로 연결되는 것을 확인 할 수 있다.

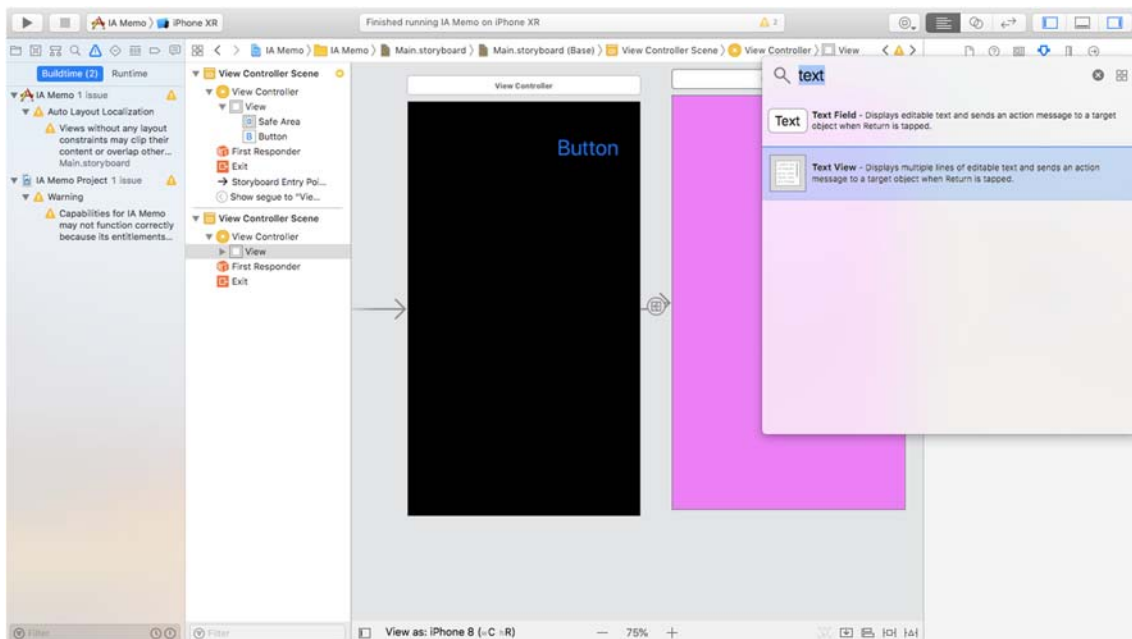
왼쪽에 화살표가 있는 View Control 이 열리고 이 화면에 있는 Button 을 클릭하면 연결된 두번째 View Control 페이지가 열린다.

두 화면의 색이 하얀색이라 구별이 안되므로 배경색을 변경하도록 해보자.

배경 색을 변경 할 View Control 을 선택을 하고 메뉴에서 4 번째 버튼을 클릭하면 Background 라는 항목이 있어 여기서 배경색을 변경해주면 된다.



아이템을 선택하는 버튼을 눌러서 검색에서 text 를 검색한다.

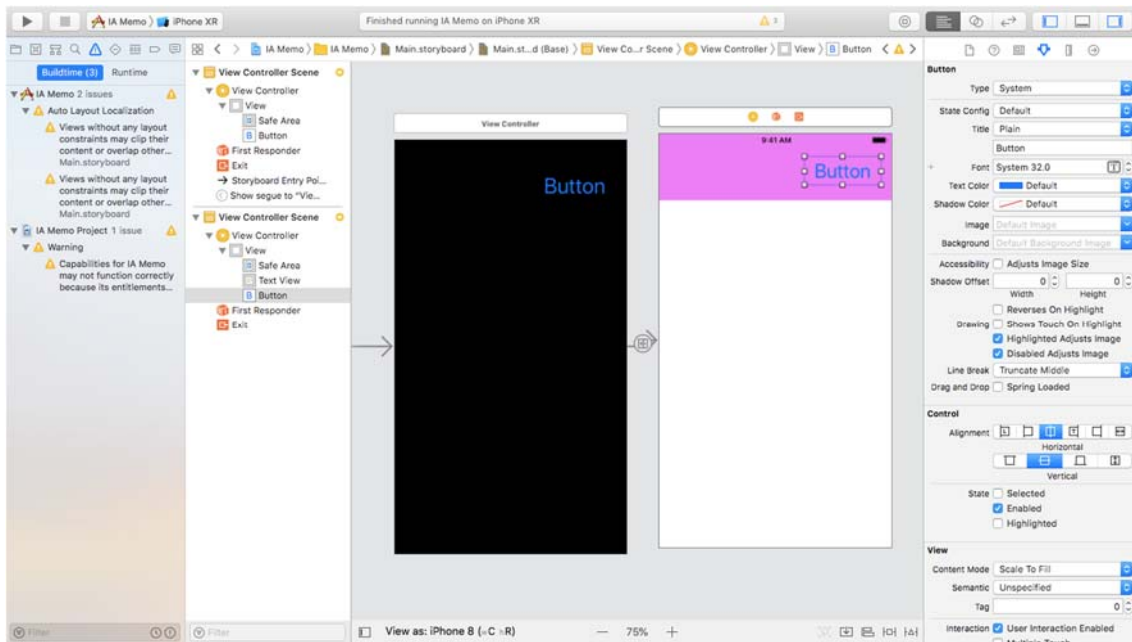


Text View 를 선택하여 두번째 View Control 에 드래그 드롭을 해준다.

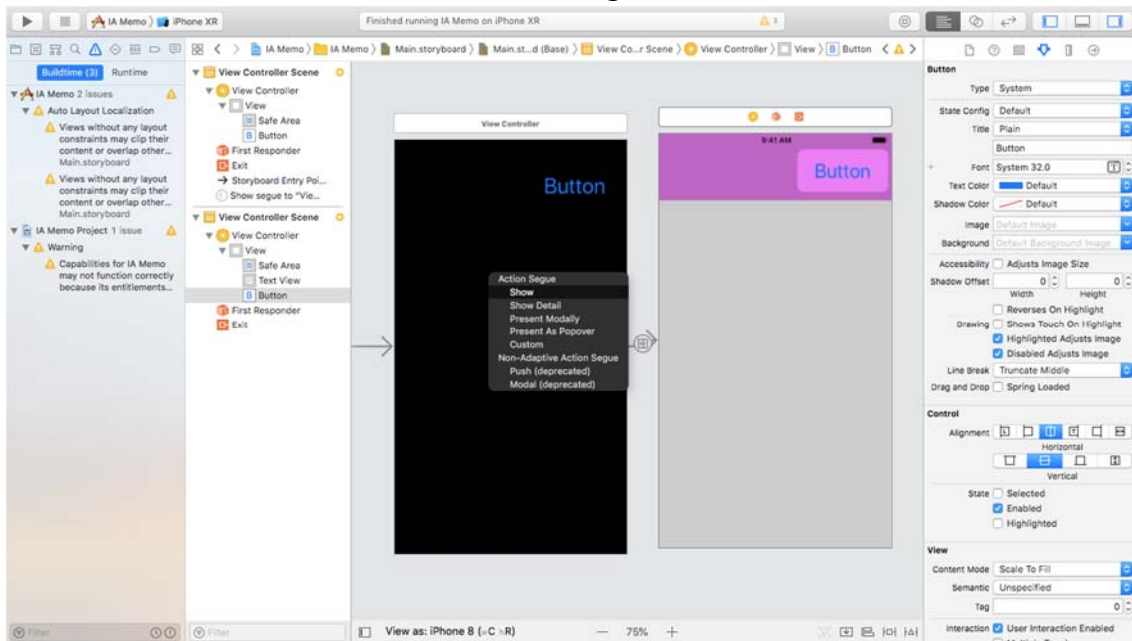
Text View 에 있는 내용은 왼쪽 창에 Text 항목에서 삭제하고 버튼을 추가한다.

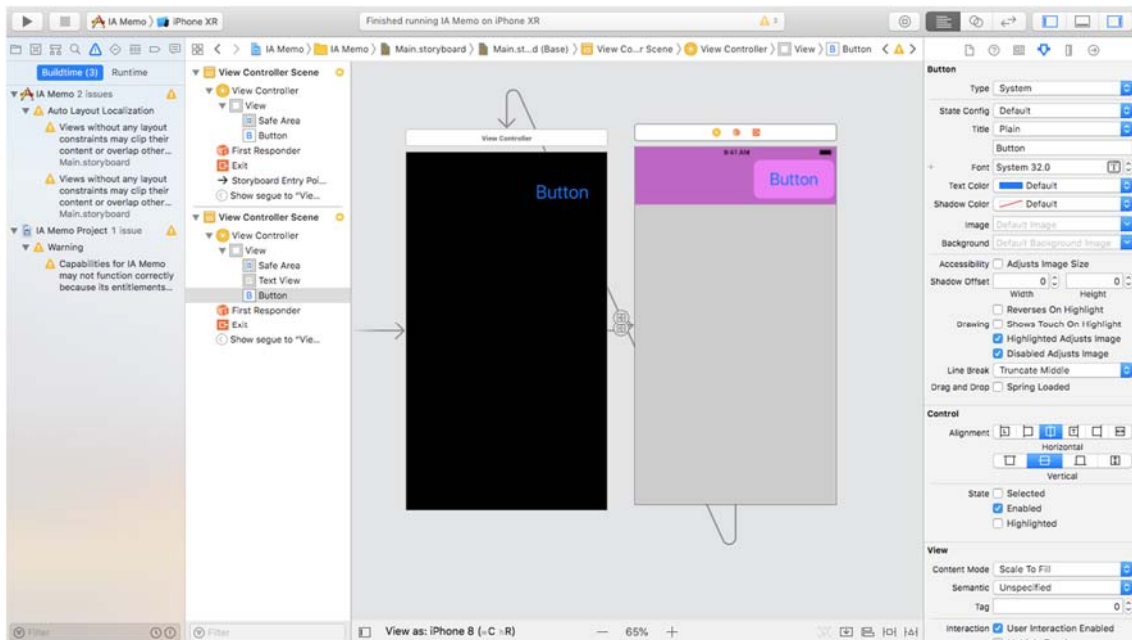
두번째 View Control 에서 Text 를 작성하고 Button 을 누르면 첫번째 View Control 로 이동하도록 한다.





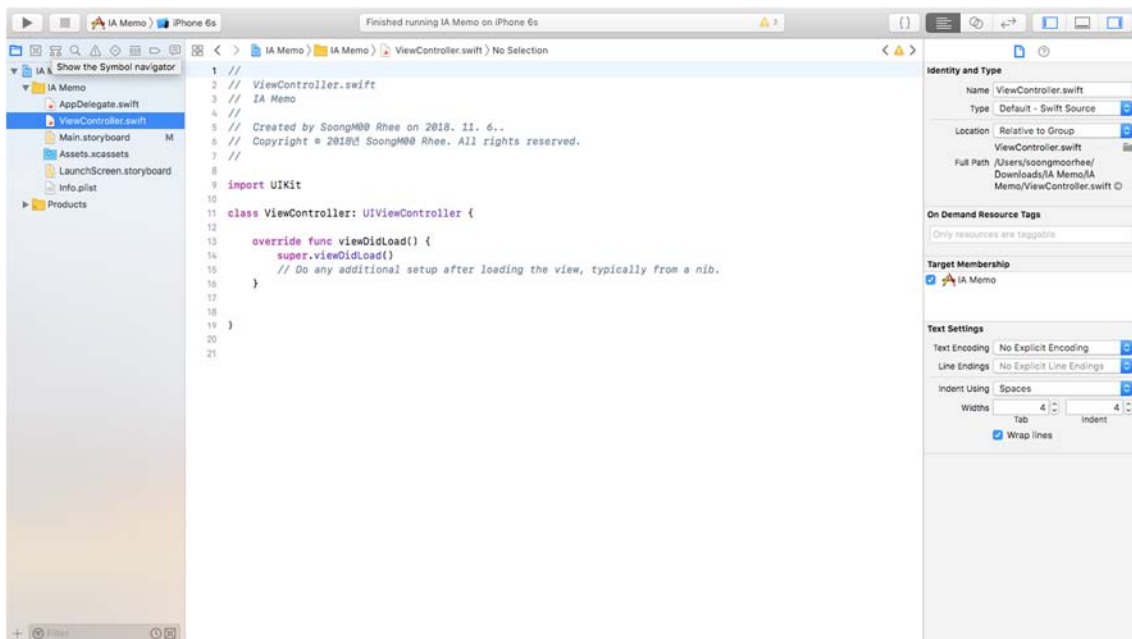
두번째 View Control 에 있는 버튼에서 Control 키를 누른 상태에서 마우스를 왼쪽 첫번째 View Control 로 드래그 드롭하고 Action Segue 에서 show 를 선택한다.





이제 두번째 View Control 에서 Button 을 누르면 Text View 에 있는 내용을 저장하게 하기 위해서는 ViewController.swift 파일이 있어야한다.

즉, View Control 하나당 ViewController.swift 가 하나씩 필요하므로 Class 파일을 만들어 줘야 한다.



44 행에서 `Y hz F r q w r o h u` 는 `#f0dvw` 의 이름이고 `X Y hz F r q w r o h u` 은 유형을 나타내고 있는 것이다

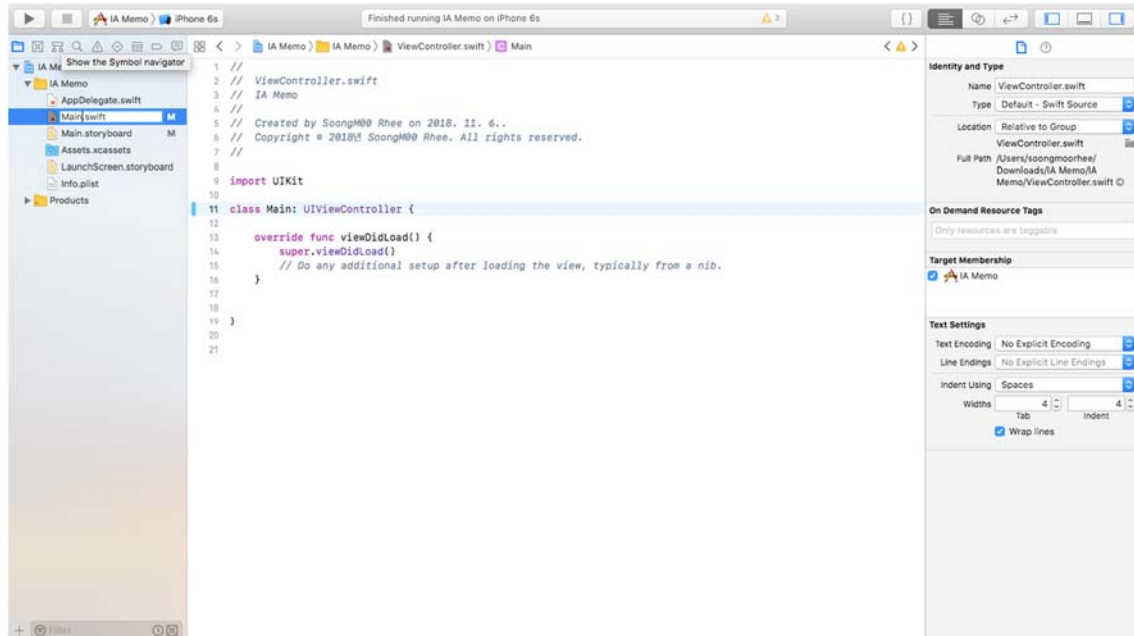
`Y hz F r q w r o h u` 는 이름이기 때문에 어떠한 이름이든 상관없이 사용이 가능하다

`Y hz F r q w r o h u` 이라는 `#f0dvw` 명을 `#p d h` 으로 변경해보자 유형은 그대로 둔다

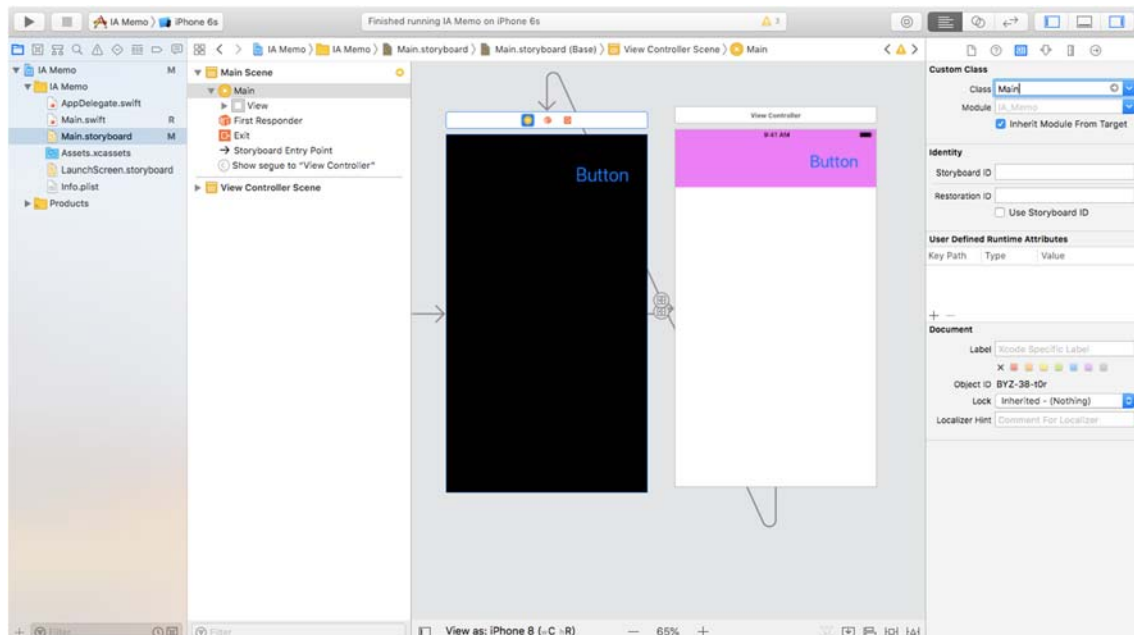


내비게이터에 있는 ViewController.swift는 #파일명이고 #변경한 #파일명은 #viewController명이다 #파일명과 #viewController명은 #꼭 #일치할 #필요성은 #없지만 #관리적인 #측면에서 #viewController명과 #파일명은 #같은 #것이 #좋다 #

viewController명을 #변경하였으므로 #파일명 #또한 #viewController명으로 #변경하는 #것이 #좋다 #

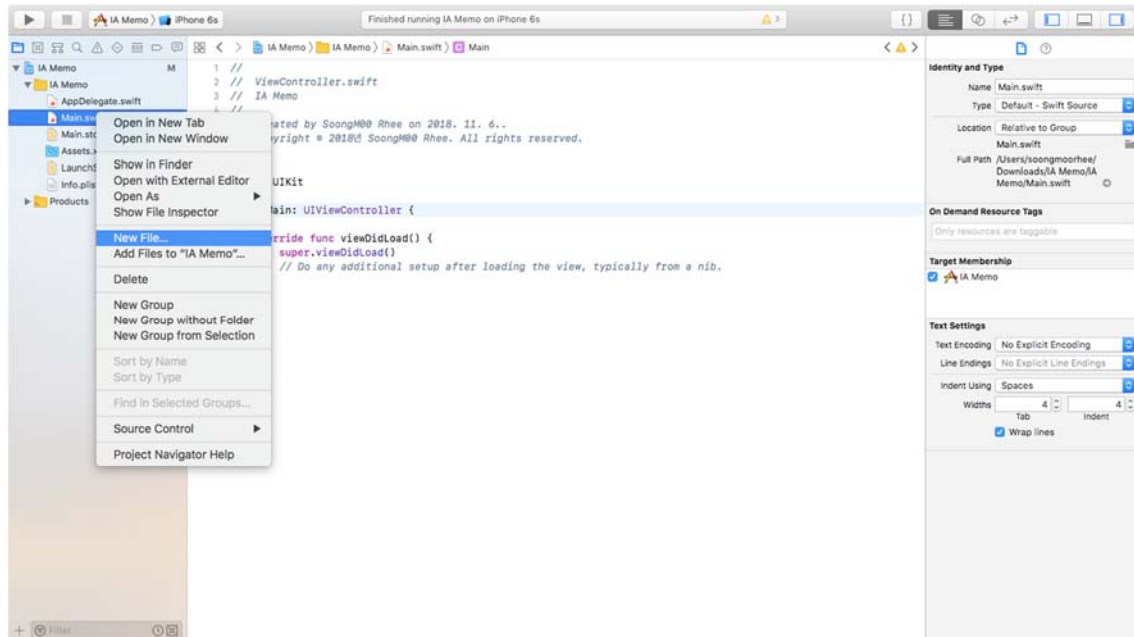


파일명을 변경을 했다면 Main.storyboard로 가서 View Control의 상단에 있는 왼쪽 노란색 버튼을 클릭을 하면 인스펙터 영역에 상단에 신문지 모양의 아이콘을 클릭하면 항목 중 class의 이름을 Main으로 변경한다.

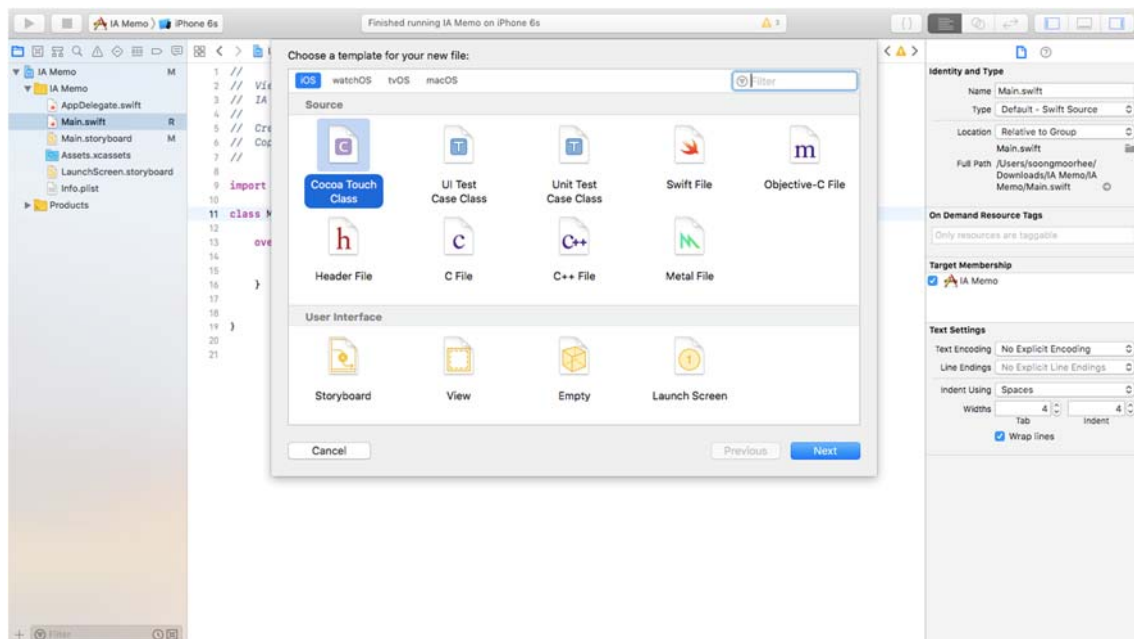


이제 두번째 View Control의 class 파일을 만들어 보자

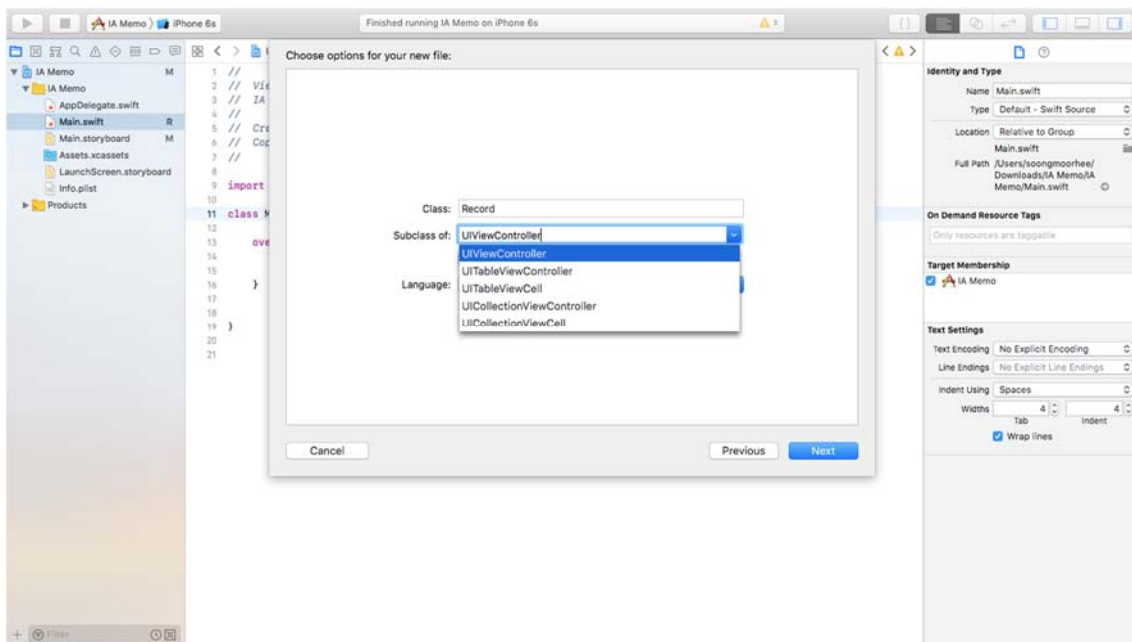
내비게이터에서 마우스 오른쪽 키를 누르고 팝업메뉴에서 New File 을 선택한다.



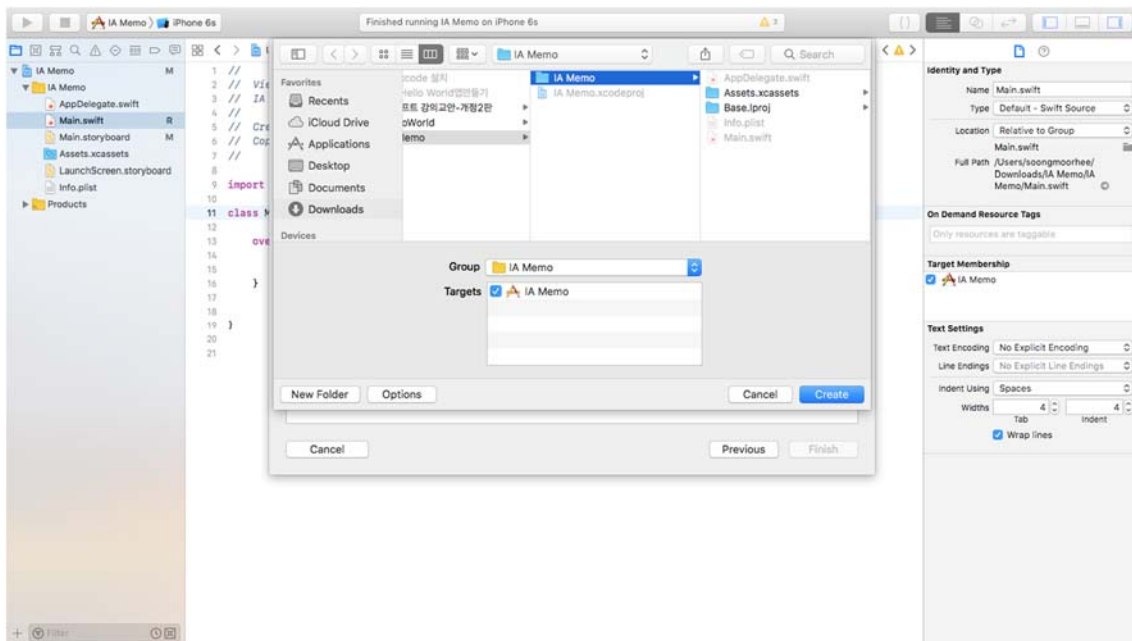
Cocoa Touch Class 를 선택하고 Next 를 한다.



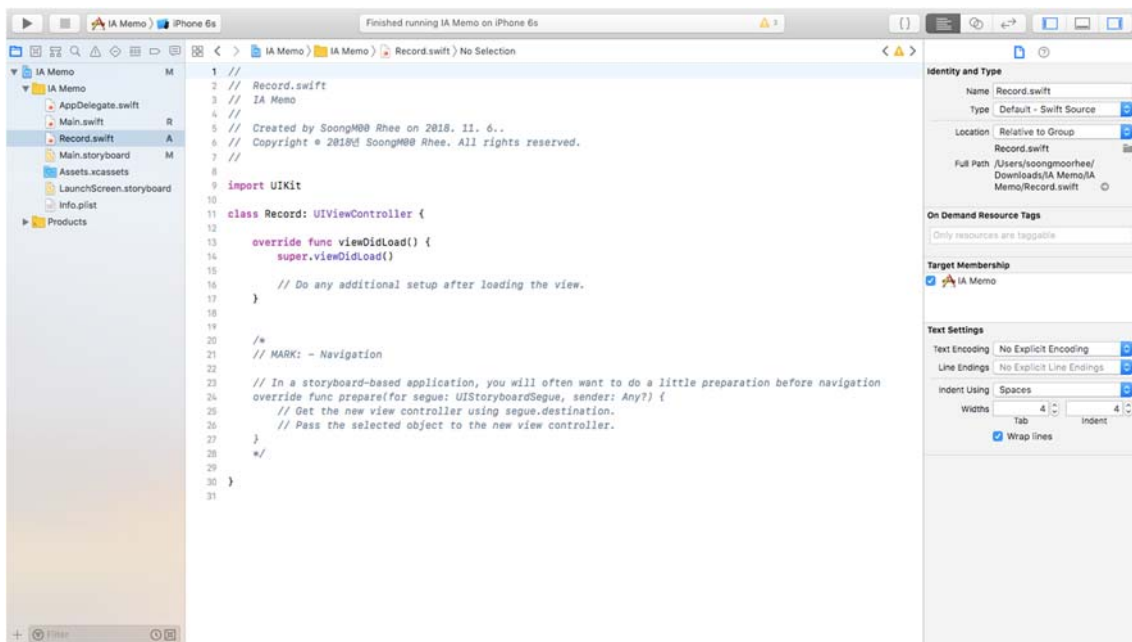
Class 명을 Record 로 주고 Subclass of 에서 UIViewController 로 지정해주고 Next 를 누른다.



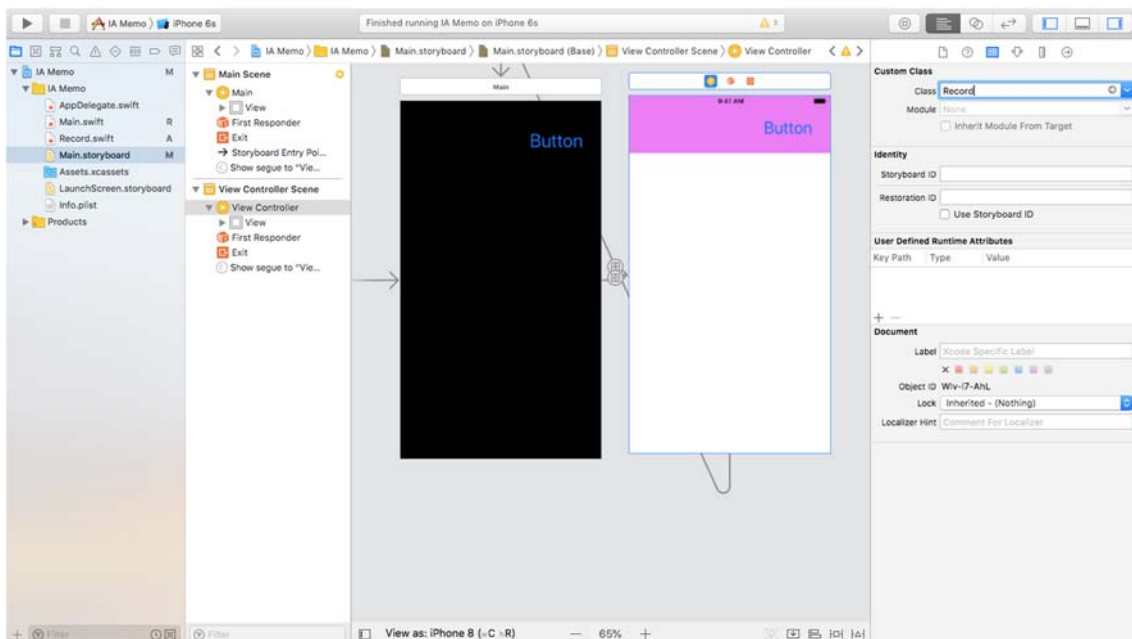
프로젝트 위치를 지정해주고 Create 를 누른다.



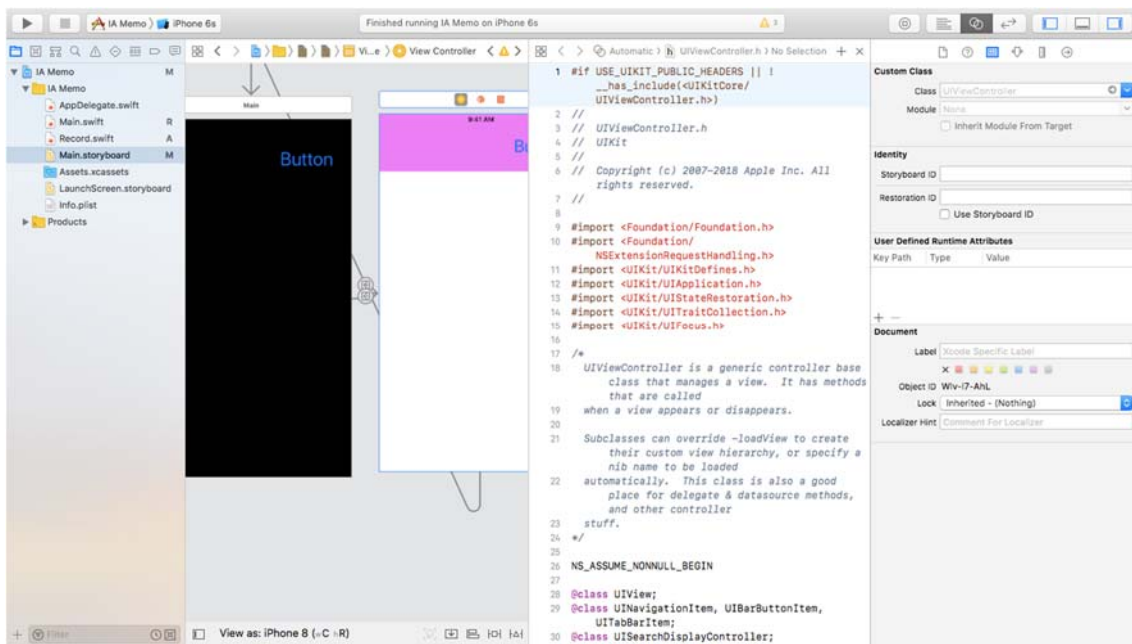
Recode 라는 swift 파일이 열린다.



storyboard 로 이동해서 두번째 View Control 의 상단에 있는 노란색 버튼을 누르고 인스펙트 메뉴에 신문지 모양을 눌러서 Class 명을 Record 로 지정을 해준다.



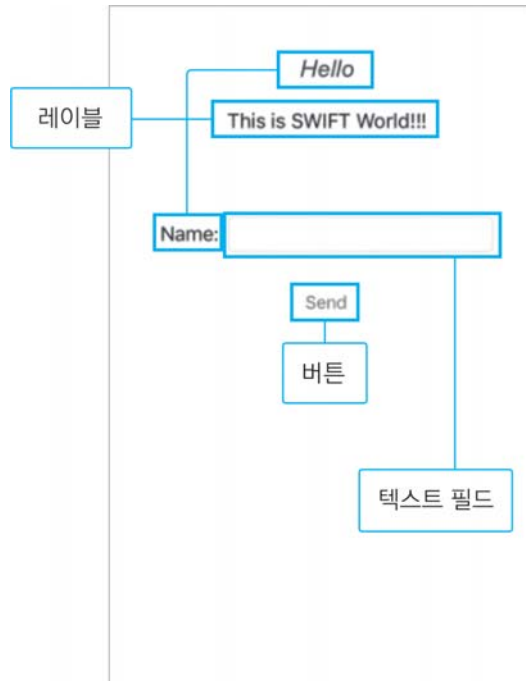
오른쪽 상단에 있는 메뉴에서 그림에 있는 가운데 버튼을 누르면 소스코드와 storyboard 가 분할되어 나온다.



## 02-4 스토리보드로 Hello World 앱 화면 꾸미기

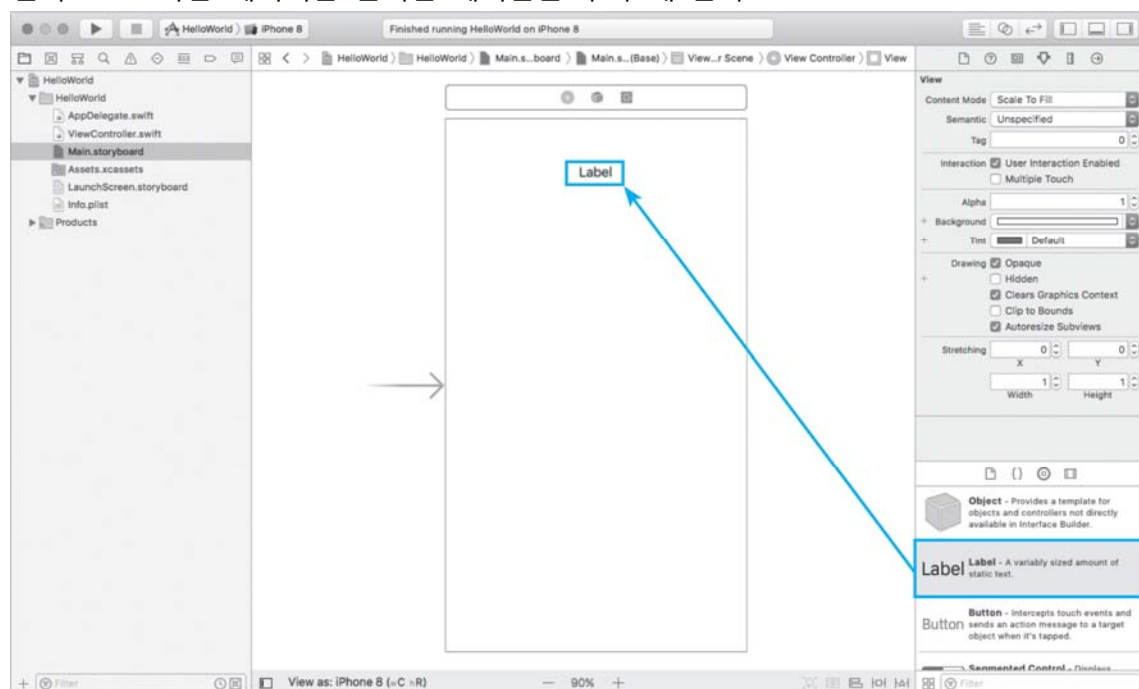
스토리보드를 사용하여 Hello World 앱의 화면을 꾸며 보겠습니다.

이 앱에서는 텍스트를 보여주는 레이블(객체)과 사용자가 직접 글을 입력할 수 있는 텍스트 필드(Text Field)객체, 이름을 전송하는 버튼(Button)객체를 사용한다.



### 1. 레이블 추가

먼저 'Hello'라는 메시지를 출력할 레이블을 추가 해 본다.

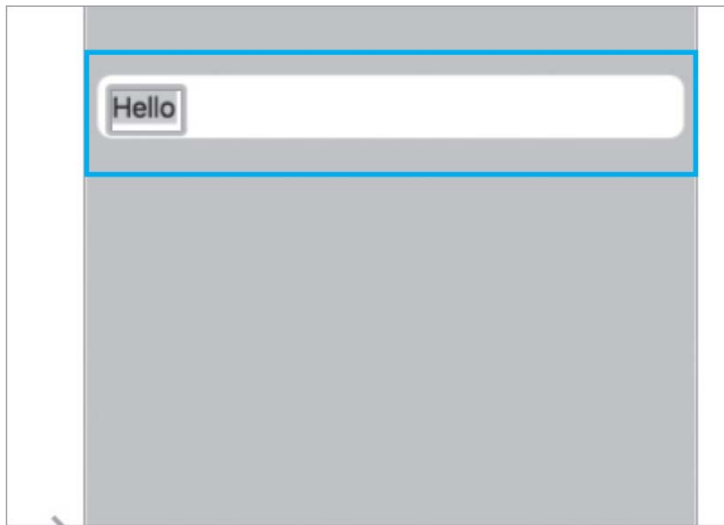


## 2. 레이블 크기 조절

레이블 길이 보다 출력하고자 하는 내용이 더 길 경우 내용이 잘려서 안 보인다.  
그러므로 출력하고자 하는 메시지가 충분히 출력될 수 있도록 레이블 크기를 조절해 보겠습니다.

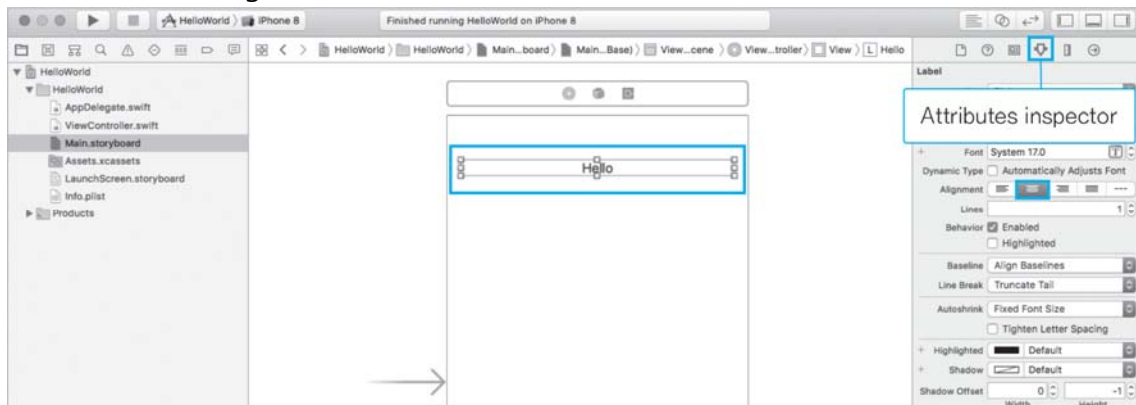


## 3. 레이블을 마우스로 더블 클릭한 후 내용을 "Hello"로 변경한다.



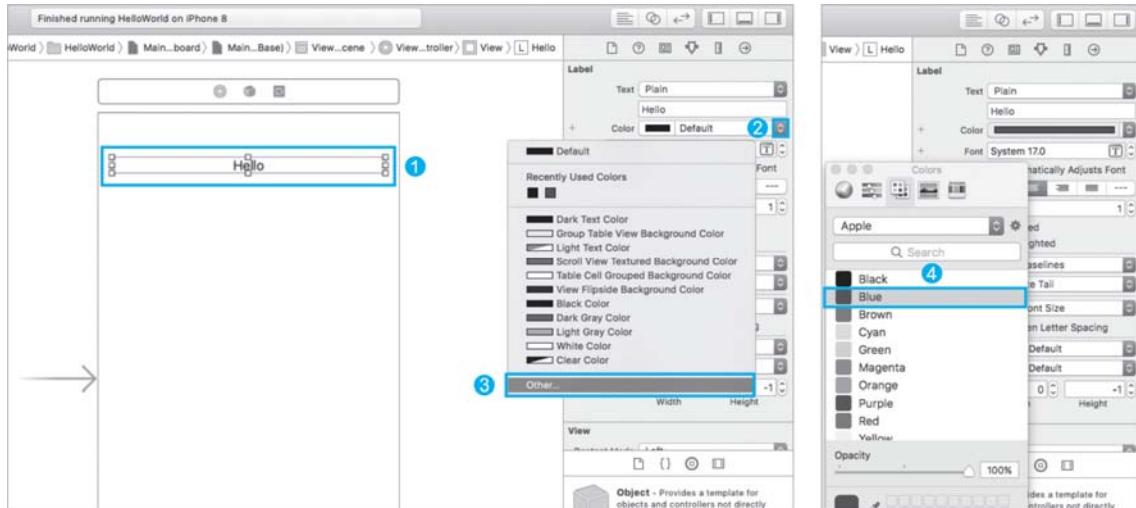
## 4. 레이블을 선택한 상태에서 화면 오른쪽 인스펙터 영역의 [Attributes inspector]버튼을 클릭한다.

그런 다음 정렬(Alignment)을 [가운데 맞춤 Alignment]으로 선택한다.



## 5. 레이블의 글자 색상과 서체 변경하기

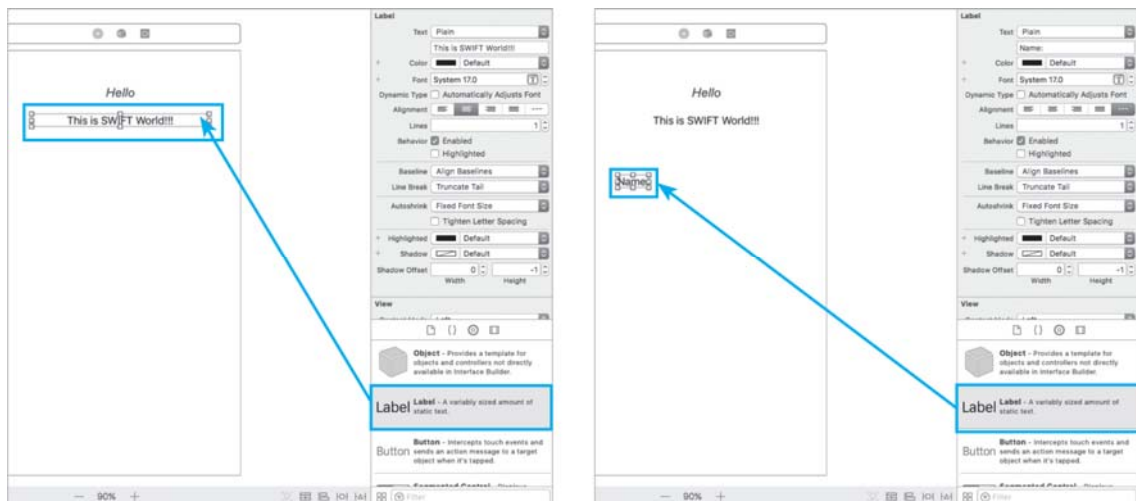
레이블의 텍스트 속성을 변경해 보자. 레이블을 선택한 후 오른쪽 인스펙터 영역의 색상(Color)을 [파란색]으로 선택해 변경한다.



같은 방법으로 서체는 [System Italic]으로, 글자 크기는 [20.0]으로 변경합니다.

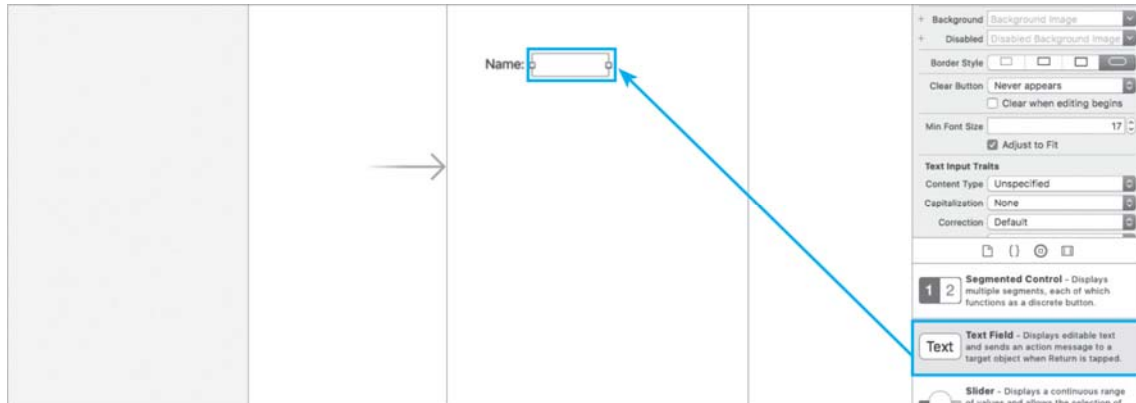
## 7. 레이블 두 개 더 추가하기

두번째 레이블의 내용에는 "This is SWIFT World"를, 세번째 레이블에는 "Name:"을 입력한다.





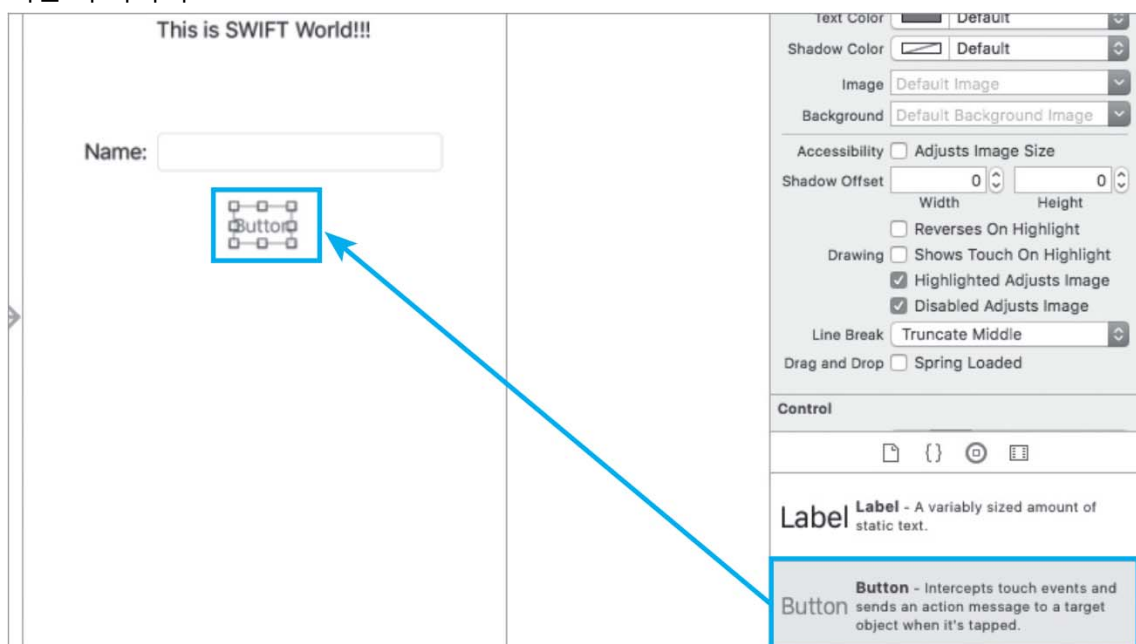
## 8. 텍스트 필드 추가하기




## 텍스트 필드 크기 조절



## 버튼 추가하기

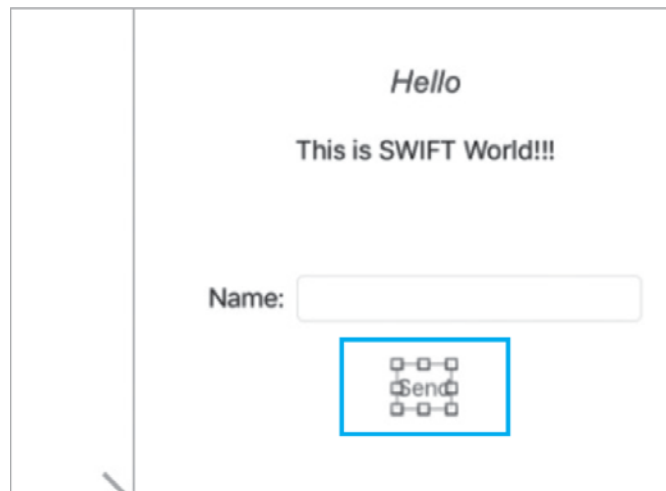


버튼의 글자를 'Send'로 변경



A UI mockup for a Swift World application. It features a light gray header bar on the left. The main content area has a white background with the text "Hello" in a large, bold, italicized font, followed by "This is SWIFT World!!!" in a smaller, bold font. Below this is a "Name:" label next to a text input field. At the bottom, there is a button with a blue border and a placeholder icon consisting of a grid of squares with the word "Send" in the center.

버튼의 글자를 'Send'로 변경

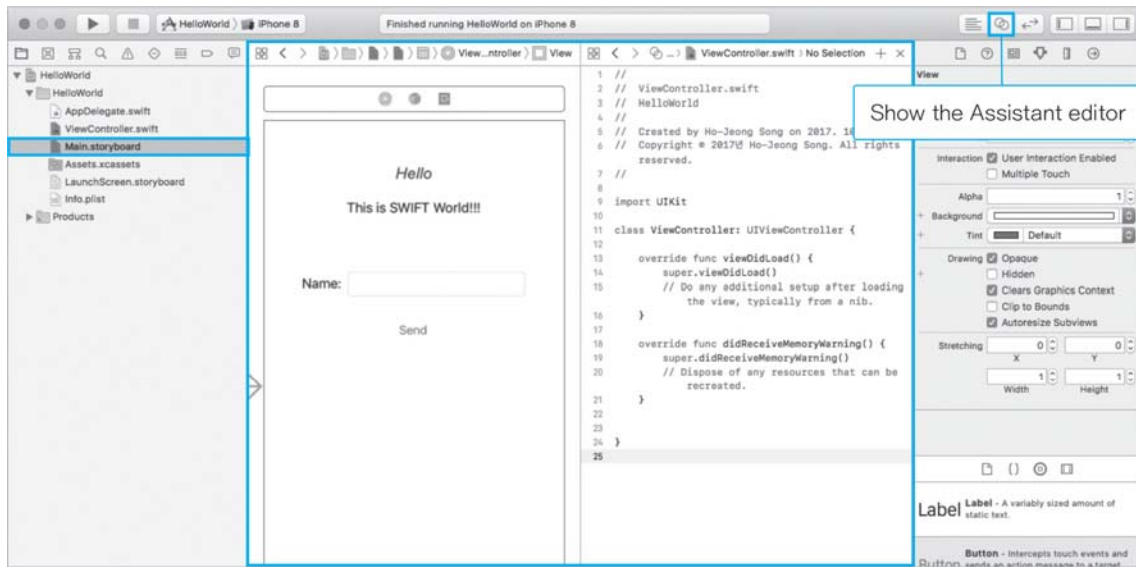


A UI mockup for a Swift World application, identical to the one above. It features a light gray header bar on the left. The main content area has a white background with the text "Hello" in a large, bold, italicized font, followed by "This is SWIFT World!!!" in a smaller, bold font. Below this is a "Name:" label next to a text input field. At the bottom, there is a button with a blue border and a placeholder icon consisting of a grid of squares with the word "Send" in the center.

## 02-4 아웃렛 변수와 액션 함수 추가하기

소스 작업을 위한 보조 편집기 영역 열기

화면 오른쪽 위의 [Show the Assistant editor] 버튼을 클릭하면 화면 가운데의 스토리보드 부분이 둘로 나누어지면서 왼쪽에는 스토리보드, 오른쪽에는 소스를 편집하는 영역이 나타납니다.

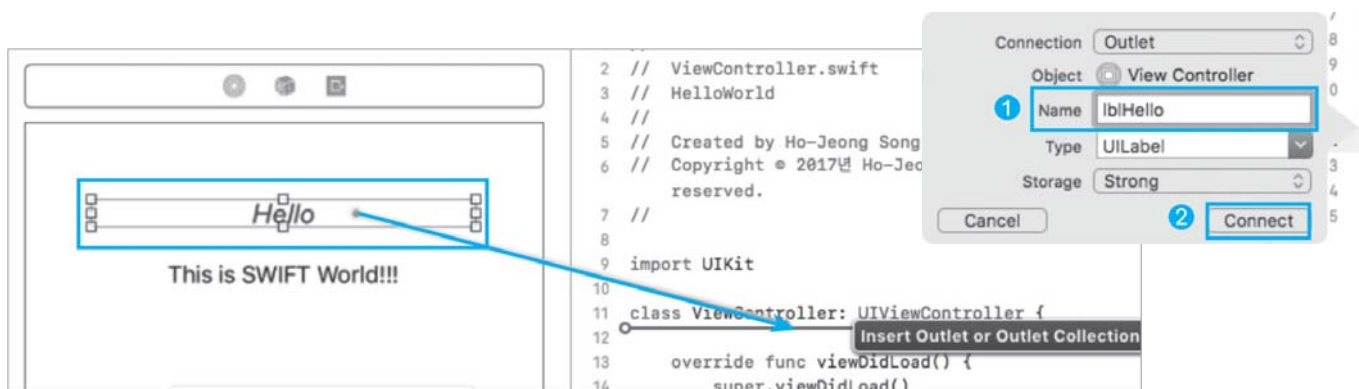


### 1. 레이블에 아웃렛 변수 추가하기

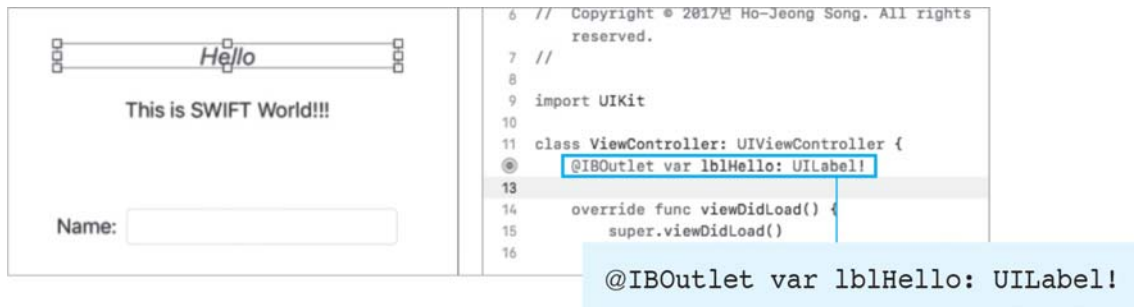
각각의 객체에 아웃렛 변수를 추가하는 방법은 매우 간단하다, 먼저 "Hello" 레이블을 마우스 오른쪽 버튼으로 선택한 후 오른쪽 보조 편집기 영역으로 드래그한다.

그림과 같이 연결선이 나타나면 이 연결선을 뷰 컨트롤러의 클래스 선언문 바로 아래에 끌어다 놓으세요.

아웃렛 변수는 일반적으로 클래스선언부 바로 아래에 추가한다.



레이블에 대한 아웃렛 변수가 추가되었다.



#### 4. 텍스트 필드에 아웃렛 변수 추가하기

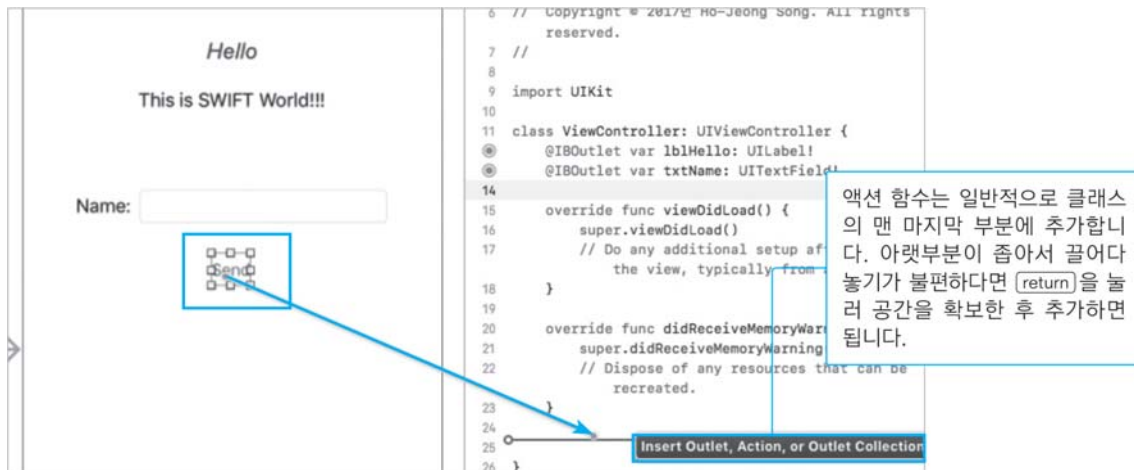


텍스트 필드에 대한 아웃렛 변수가 추가되었다.



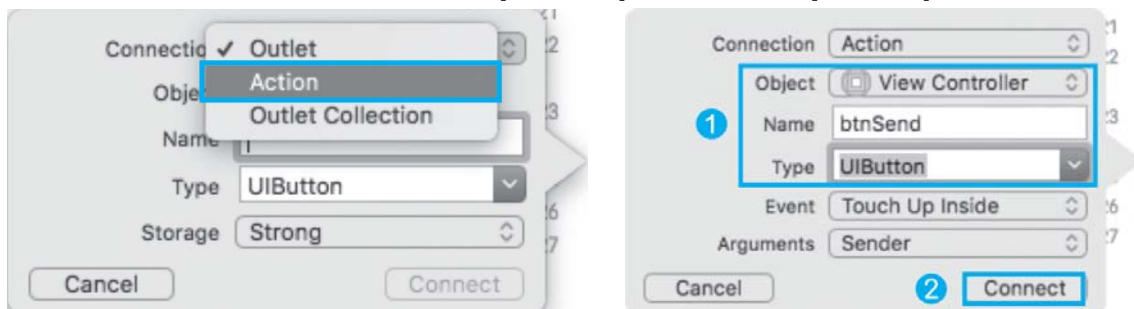
버튼에 대한 액션 함수 추가하기

1. 마우스 오른쪽 버튼으로 [Send] 버튼을 클릭한 후 보조 편집기 영역으로 드래그한다.

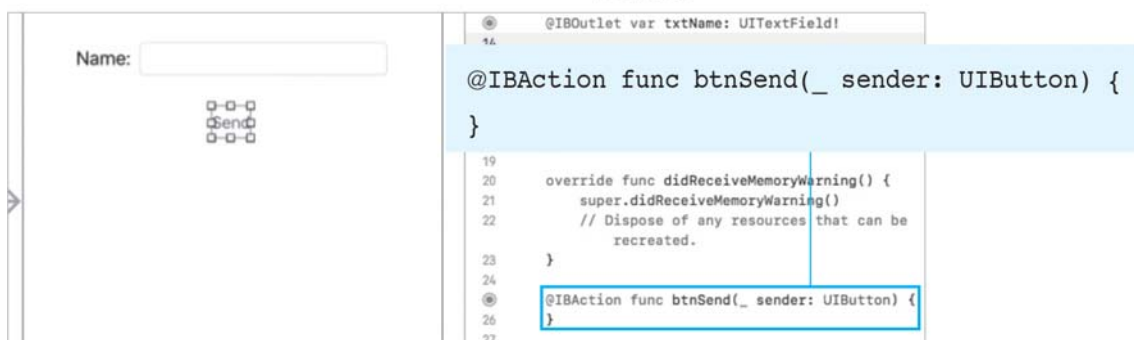


2. 다음과 같이 설정창이 나타난다. 그런데 액션 함수는 아웃렛 변수처럼 이름만 입력하고 넘어가면 안된다. 'Outlet'으로 선택되어 있는 연결(Connection)을 [Action]으로 변경해줘야 한다.

이름(Name)을 'btnSend'로 입력한 후 타입(Type)을 해당 객체에 맞게 변경한다. 여기서는 버튼에 액션을 추가하는 것이므로 [UIButton]을 선택한 후 [Connect]버튼을 클릭한다,



3. 버튼에 액션이 추가되었다.



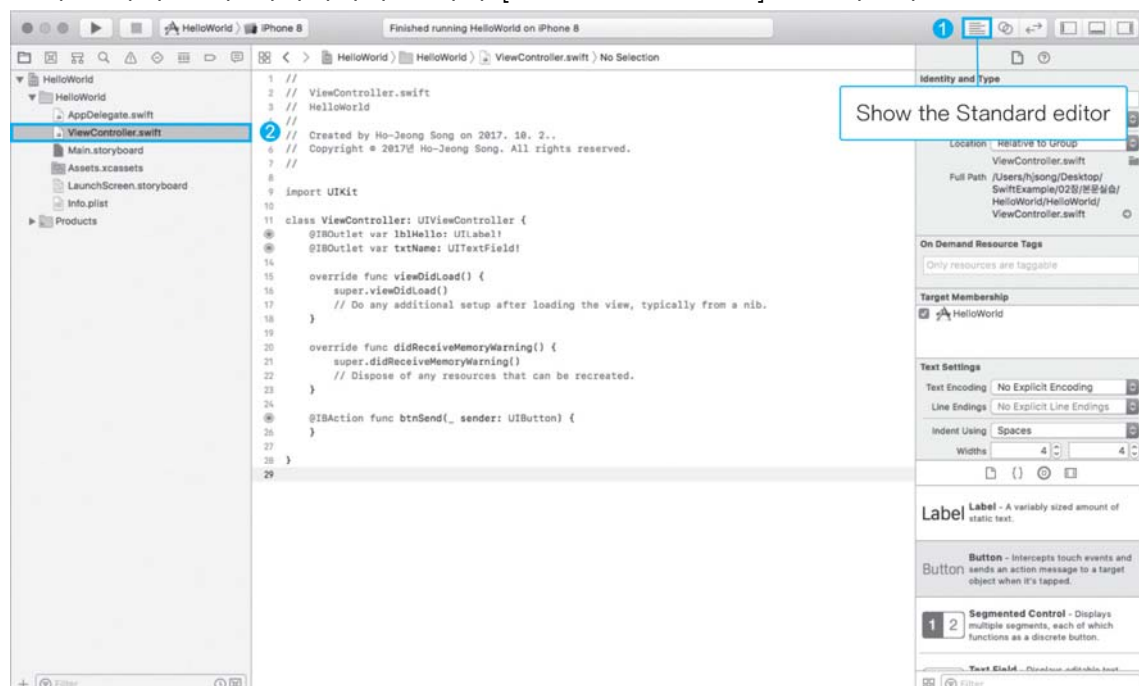
## 02-5 버튼클릭시동작구현하기

화면 모드를 '기본 편집기' 화면으로 수정한 후 코딩하기

1. 오른쪽 위의 [Show the Standard editor] 버튼을 클릭하여 화면 모드를 '스탠더드 에디터'로 수정한다.

그러므로 스토리보드와 보조 편집기 영역으로 나누어져 있던 화면이 다시 하나의 화면으로 바뀌게 된다.

그리고 나서 왼쪽 네비게이터 영역의 [ViewController.swift]를 선택한다.



2. 버튼 클릭 시 동작할 함수 코딩하기

[Send]버튼을 클릭했을 때 동작할 btnSend 액션 함수를 다음과 같이 코딩한다.



스위프트에서는 문자열을 합칠 때 '+' 연산자를 사용합니다.

```
lblHello.text = "Hello, " + "Ho-Jeong"
```

이렇게 하면 "Hello, "라는 문자열과 "Ho-Jeong"이라는 문자열을 합쳐서 lblHello.text 에 저장하게 된다.

**[스위프트 문법]** 스위프트에는 문장 맨 끝에 ';'이 없나요?

일반적인 프로그래밍 언어에는 문장 맨 끝에 ';'이 있어야 한다.

컴파일러에게 문장의 끝이라는 것을 알려 주는 일종의 약속이다.

하지만 스위프트에서는 특이하게도 문장의 끝에 ';'을 사용하지 않는다.

조금은 어색하겠지만 사용하다 보면 오히려 더 편하다고 생각할 수 있다.

하지만 다른 언어와 병행해서 사용해야 할 때는 주의해야 한다.

예를 들어 C 언어에서 ';'을 안 붙이거나 스위프트에서 ';'을 붙이는 경우가 종종 발생하니 조심해야 한다.