Wprowadzenie do programowania w środowisku Xcode.

Xcode to środowisko programistyczne używane do tworzenia aplikacji dla wielu platform Apple, takich jak iPhone, iPad lub Apple TV czy Apple Watch. Wykorzystywany jest język Swift. Xcode po raz pierwszy został wydany w 2003 roku. Jest to kompletny pakiet i za jego pomocą programiści mogą projektować interfejs użytkownika, pisać kod aplikacji, kompilować czy też testować kod oraz sprawdzać błędy w kodzie. Umożliwia także przesłanie aplikacji do sklepu Apple.

Xcode działa tylko w systemie macOS, co oznacza, że potrzebujemy komputera Mac do programowania, jeśli chcemy tworzyć aplikacje na iOS za pomocą Xcode. Wszyscy użytkownicy Mac mogą korzystać z Xcode za darmo, ale aby dystrybuować aplikacje na wielu platformach za pomocą sklepu App Store, należy być zarejestrowanym subskrybentem programu Apple Developer, która kosztuje 99 USD rocznie. Xcode jest aktualizowany raz w roku(około września-października), w tym samym czasie, gdy wydawana jest nowa główna wersja iOS. Każda aktualizacja Xcode przynosi ulepszenia, nowe funkcje, poprawki błędów i dostęp do najnowszych zestawów SDK. Przez cały rok wprowadzanych jest wiele mniejszych aktualizacji Xcode, w tym aktualizacje dla Swift. Jest bardzo prosty w użyciu i może być również używany przez początkujących programistów. Programiści mają również możliwość korzystania z konstruktora interfejsów oraz projektowania menu i okien.

Minimalne wymagania systemowe dla Xcode 12:

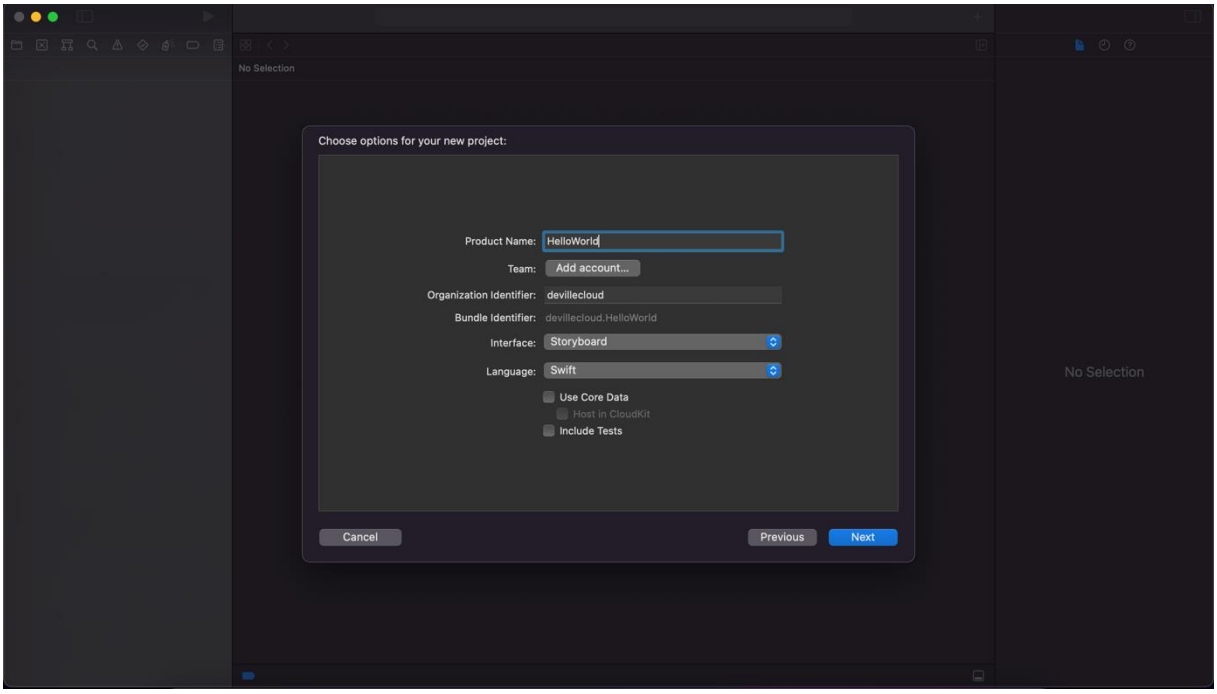
- Komputer Mac z systemem macOS 11 (Big Sur)
- 4 GB pamięci RAM, ale ponad 8 GB jest wygodniejsze.
- Co najmniej 8 GB wolnego miejsca
- Mac, MacBook, iMac lub Mac mini.

Ćwiczenie 1.

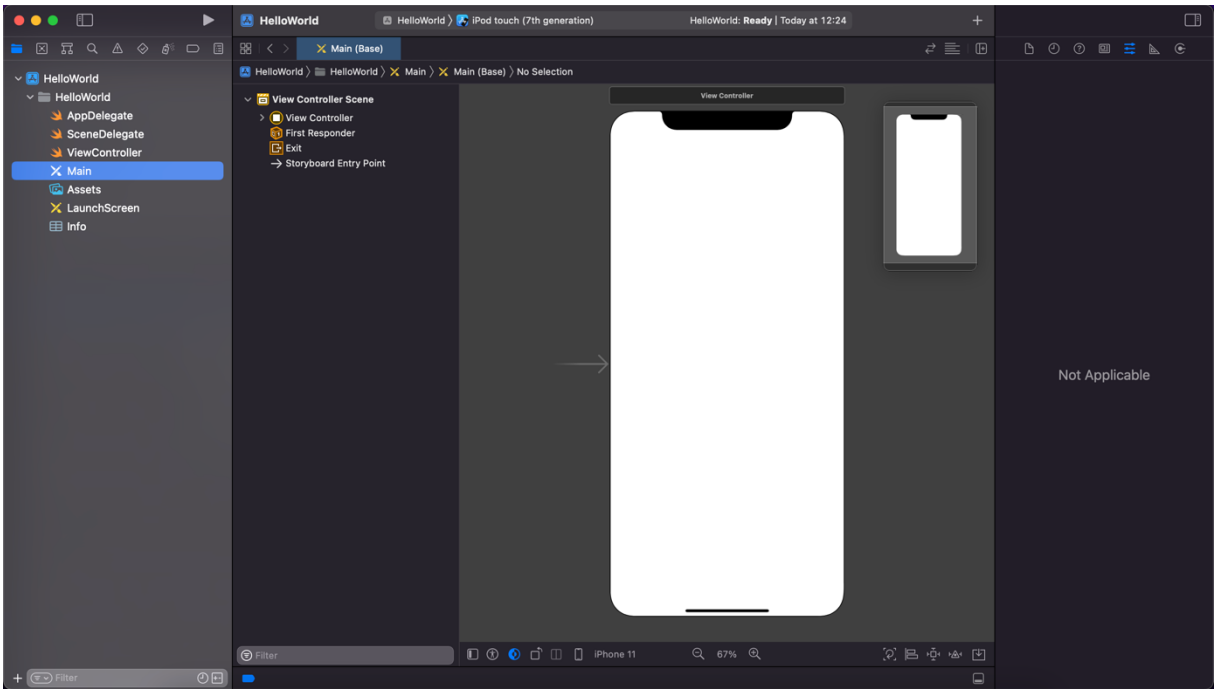
---

1. Uruchom aplikację Xcode na komputerze Mac. Zostaniesz powitany ekranem powitalnym Xcode. W oknie Choose a template for your new projekt wybierz iOS a następnie App.
2. Po wybraniu szablonu będziemy musieli wypełnić podstawowe dane projektu.
   Product Name - Nazwa Twojej aplikacji.
   Team - Jeżeli jesteś członkiem Apple Developer Portal możesz wybrać tutaj swój profil. Pozwoli Ci to na uruchomienie aplikacji na dowolnym urządzeniu z systemem iOS.
   Organization Identifier – nazwa organizacji
   Bundle Identifier - na podstawie Organization Identifier oraz Product Name jest określany unikalny identyfikator aplikacji.
   Interface – wybierz Storyboard.
   Language - język programowania aplikacji. Wybierz domyślny Swift
   Use Core Data - jeżeli potrzebujesz korzystać z biblioteki do baz danych Apple możesz z tej opcji skorzystać.
   Include Unit Tests - jeżeli chcesz pisać testy jednostkowe do aplikacji możesz to zaznaczyć.

Include Tests - Xcode daje możliwość pisania testów automatycznych Nie jest to potrzebne dla tego projektu.



3. Po wypełnieniu i naciśnięciu przycisku “Next” Xcode zapyta Cię gdzie utworzyć projekt. Możesz go zapisać na pulpicie albo dokumentach. W dalszej części można też zaznaczyć “Create Git repository on my Mac”, jeśli chcemy korzystać z systemu Git. Jest to rozproszony system kontroli wersji, którego jednym z zadań jest śledzenie zmian w kodzie.
4. Po kliknięciu przycisku Utwórz Xcode utworzy i otworzy nowy projekt. W naszym projekcie Main.storyboard i ViewController.swift są głównymi plikami, które użyjemy do zaprojektowania interfejsu użytkownika i napisania kodu aplikacji.
5. W Xcode logika biznesowa (plik kontrolera .swift) jest oddzielona od warstwy prezentacji danych (plik .storyboard). Interfejs Xcode umożliwia proste zaprojektowanie interfejsu graficznego. Kluczowe jest natomiast połączenie elementów graficznych z interfejsu z plikami języka Swift.



6. W początkowym projekcie oprócz plików Main oraz ViewController znajdziemy:
   AppDelegate.swift - Jest to klasa, w której będziemy mieli metody, które są wykonywane, gdy coś się wydarzy w naszej aplikacji (na przykład, gdy aplikacja przejdzie do pracy w tle). Możemy dodać niestandardowy kod do tych metod, aby dostosować zachowanie aplikacji.
   SceneDelegate.swift – jest podobny do AppDelegate
   Assets.xcassets - tam zapisujemy zasoby graficzne dla naszej aplikacji takie jak obrazy, kolory itp.
   LaunchScreen.storyboard - to tutaj definiujemy ekran "Splash" naszej aplikacji, ekran ładowania wyświetlany podczas uruchamiania aplikacji.

Sprawdź swoją wiedzę.

1. Scharakteryzuj krótko aplikację Xcode.
2. Jakie jest główne zadanie klasy AppDelegate.swift?

Praca ze Storyboard i Interface Builder

Jednym z głównych narzędzi w aplikacji Xcode jest Interface Builder. Używamy go do konstruowania interfejsów użytkownika dla swoich aplikacji łącząc je z kontrolerami widoków, co stanowi podstawę interfejsu użytkownika aplikacji. Interface Builder może początkowo wydawać się zniechęcający, ale jest to prosty program, biorąc pod uwagę, jak potężny jest.

Xcode posiada dołączone narzędzie o nazwie Interface Builder, które pozwoli nam tworzyć interfejsy użytkownika w sposób wizualny. Pliki .storyboard i .xib w naszym projekcie to opisy interfejsów użytkownika. Storyboard jest zbudowany za pomocą edytora wizualnego dostarczonego przez Xcode, w którym możemy układać i projektować interfejsy użytkownika aplikacji, dodając widżety z biblioteki multimediów, takie jak przyciski, widoki, widoki tabel, pola tekstowe itp. Jest to wizualna reprezentacja interfejsu użytkownika aplikacji na iOS.

Interfejs użytkownika Interface Builder składa się z 3 głównych obszarów:

- Po lewej stronie zobaczyć możemy konspekt dokumentu. Jest to hierarchiczna lista każdego elementu interfejsu użytkownika, Każdy ViewController utworzony w tym pliku jest dodawany do tej listy. Jest reprezentowany jako scena, do której można dodawać widoki. Lista ta znajduje się po lewej stronie, na prawo od listy plików. kontrolera widoku, sceny itp. w Storyboard.
- W środku znajduje się Edytor. Można go również nazwać kanwą, płótnem, ponieważ jest to miejsce, w którym tworzy się interfejs użytkownika.
- Po prawej stronie widać panel Inspector.

Panel Inspector posiada 7 głównych opcji zawartych w nim. Pomaga on nam sprawdzać i dostosowywać różne ustawienia.

- Inspektor plików podaje podstawowe informacje o aktualnie edytowanym pliku, takie jak plik Swift lub cykliczność. Najbardziej godne uwagi są ustawienia lokalizacji języka (na plik) i członkostwo docelowe, które określa, czy plik zostanie dodany do ostatecznego pakietu aplikacji.
- Obok znajduje się Inspektor historii, który dostarcza informacji o kontroli źródła (Git) dla tego pliku. Możemy zobaczyć, jak plik zmienił się w stosunku do poprzednich zatwierdzeń i jakie zmiany są w toku.
- Karta ze znakiem zapytania to Inspektor szybkiej pomocy. Zapewni dokumentację pomocy kontekstowej, na przykład w przypadku wybrania widoku etykiety lub obrazu. Szybka pomoc działa również w edytorze kodu, gdzie wyświetla informacje o klasach, typach, funkcjach itp.
- Inspektor tożsamości umożliwia zmianę tożsamości elementów interfejsu użytkownika, takich jak ich klasa niestandardowa, identyfikator przywracania, pary klucz-wartość. Za pomocą tego inspektora można również dostosować ustawienia ułatwień dostępu dla elementu interfejsu użytkownika.
- Inspektor atrybutów — ten, który wygląda jak strzałka suwaka — umożliwia zmianę różnych atrybutów elementów interfejsu użytkownika, w tym tekstu, czcionki, koloru, tła, trybów

widoku, tytułów przycisków, obrazów i wielu innych. Inspektor atrybutów jest jednym z najczęściej używanych paneli inspektorów.

- Inspektor rozmiarów (ikona linijki) może służyć do dostosowywania położenia i wymiarów elementów interfejsu użytkownika. Tutaj znajdziesz również ograniczenia dla elementu interfejsu użytkownika, a także ustawienia marginesów i układu.
- Inspektor połączeń służy do zarządzania gniazdami. Gniazdko to połączenie między elementem interfejsu użytkownika w Konstruktorze interfejsów a właściwością w pliku Swift.

Możemy podzielić tych 7 inspektorów na 2 grupy:

- Inspektorzy zasobów, tacy jak inspektorzy plików, historii, pomocy i tożsamości. Informują o zasobach, takich jak nazwa pliku lub nazwa klasy. Inspektorzy ci są najczęściej wykorzystywani do celów informacyjnych.
- Inspektorzy oparci na ustawieniach, tacy jak inspektorzy atrybutów, rozmiarów i gniazd. Pomagają one zmienić niektóre ustawienia lub atrybuty, takie jak czcionka lub szerokość. Inspektorzy ci są najczęściej wykorzystywani do budowy aplikacji.

Storyboard został po raz pierwszy wprowadzony w iOS 5, aby zaoszczędzić czas na tworzeniu interfejsów użytkownika dla aplikacji iOS. Można go zdefiniować jako sekwencję ekranów, z których każdy reprezentuje modele ViewController i Views. Storyboard, jest podejściem do łączenia różnych kontrolerów widoku w jeden. Storyboard składa się ze scen, z segues (przejściami) między nimi.

Przejścia między dwoma ekranami Storyboard wymagają obiektu segue, który reprezentuje przejście między dwoma kontrolkami widoku. Plik Storyboard pozwala na eleganckie definiowanie skomplikowanego interfejsu użytkownika bez użycia kodu. Ten plik można wykorzystać do wizualizacji połączeń między poszczególnymi ekranami aplikacji, które w Storyboard są nazywane scenam

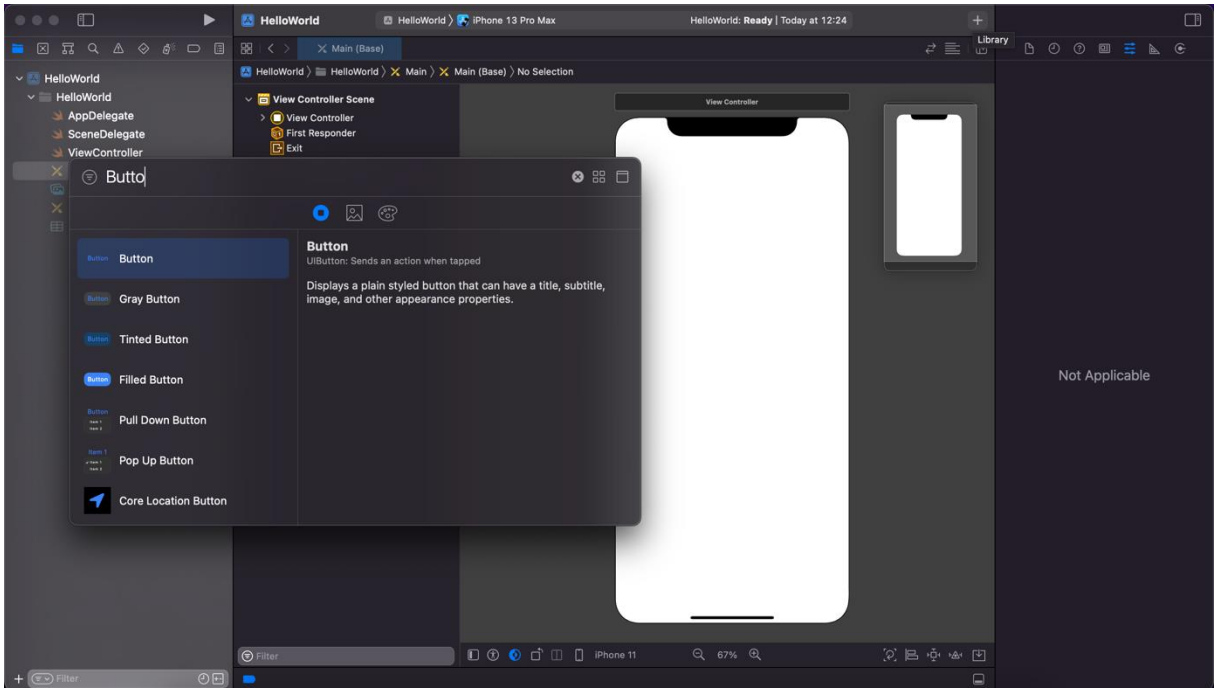
Jest to przede wszystkim użyteczne gdyż zamiast pisania kodu źródłowego możemy przeciągając i upuszczając modelować naszą aplikację. Aby przejść do Main.storyboard należy z listy plików, która znajduje się po lewej wybrać plik o tej nazwie.

Ćwiczenie 1.

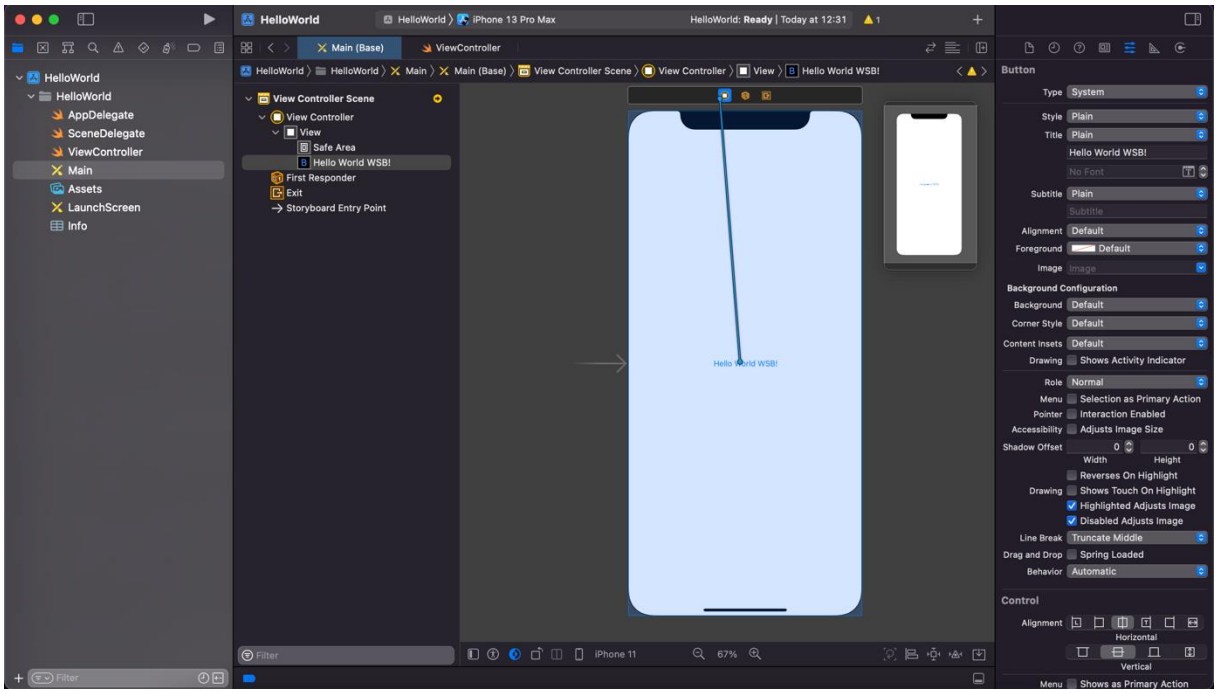
Stwórzmy naszą pierwszą aplikację na iOS, aby wyświetlić wiadomość użytkownikowi na ekranie. Zaczniemy od aplikacji Hello World na iOS.

1. Utwórz nowy projekt o nazwie HelloWorld oraz przejdź do pliku Main.storyboard

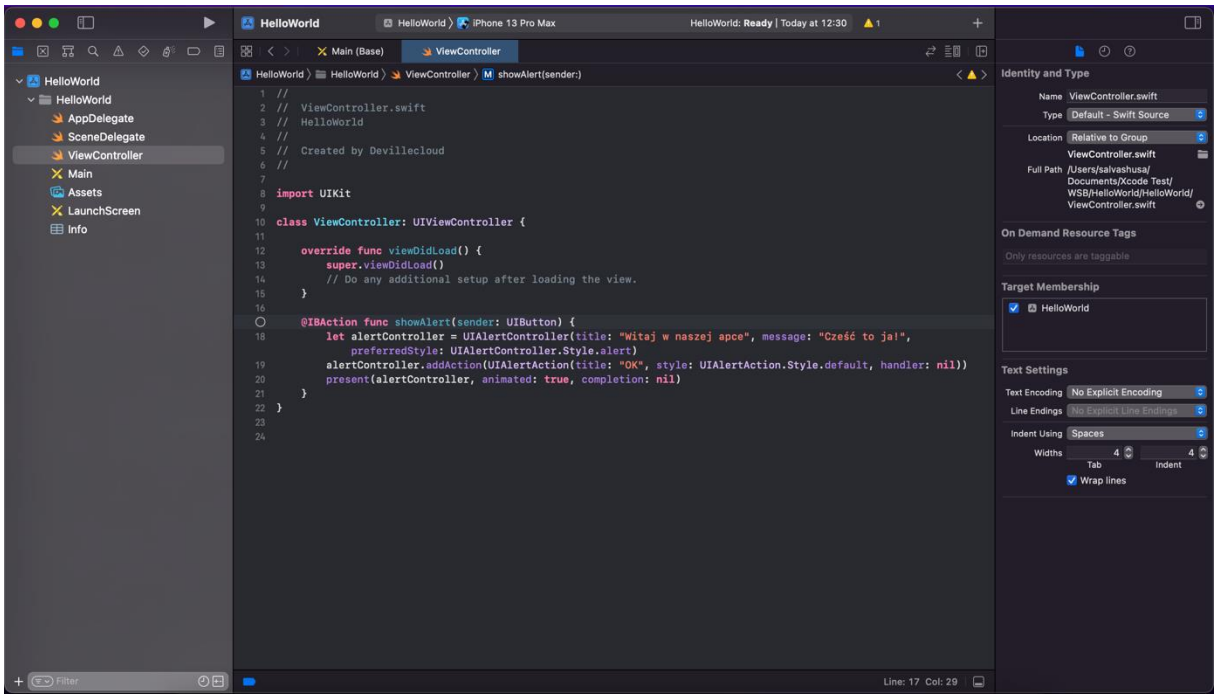
2. Najpierw dodamy przycisk Hello World do widoku. Kliknij przycisk biblioteki (+), aby wyświetlić bibliotekę obiektów. Następnie możesz wybrać dowolny z obiektów interfejsu użytkownika i przeciągnąć je i upuścić do widoku. Następnie możemy zmodyfikować style i tekst przycisku, aby dopasować je do swoich upodobań.



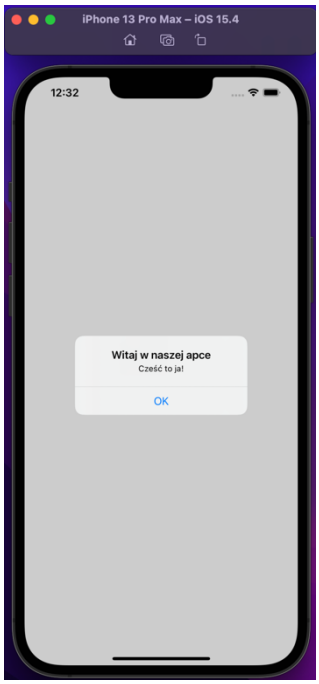
1. Podczas przeciągania obiektu Button do widoku zostanie wyświetlony zestaw poziomych i pionowych linii pomocniczych. Na środku ustaw obiekt Button.
2. Następnie zmień nazwę przycisku. Aby edytować etykietę przycisku, kliknij go dwukrotnie i nadaj mu nazwę clickButton.
3. Aby zmodyfikować styl przycisku w konstruktorze interfejsów, otwórz Inspektora atrybutów. Możesz kliknąć na niego w okienku po prawej stronie lub użyć skrótu klawiaturowego (CMD + Option + 4). Tutaj możemy modyfikować przycisk i różne inne właściwości, takie jak: Czcionka, Kolor tekstu, Tło i wiele innych. W tym przykładzie zmień tekst na "Hello World”.
4. Jeśli chcemy uzyskać dostęp do tego przycisku w naszym kodzie Swift musimy utworzyć IBOutlet. IBOutlets to skrót od interface builder outlets. Zawierają one odniesienie do obiektów w konstruktorze interfejsów w kodzie.Aby utworzyć własny przycisk IBOutlet można to zrobić, przytrzymując Option na klawiaturze i klikając plik ViewController.swift w Nawigatorze projektu. Teraz kliknij przycisk, przytrzymaj CTRL, a następnie przeciągnij go do pliku Swift. Spowoduje to otwarcie okna dialogowego IBOutlet. Kliknij przycisk Połącz, aby utworzyć IBOutlet.



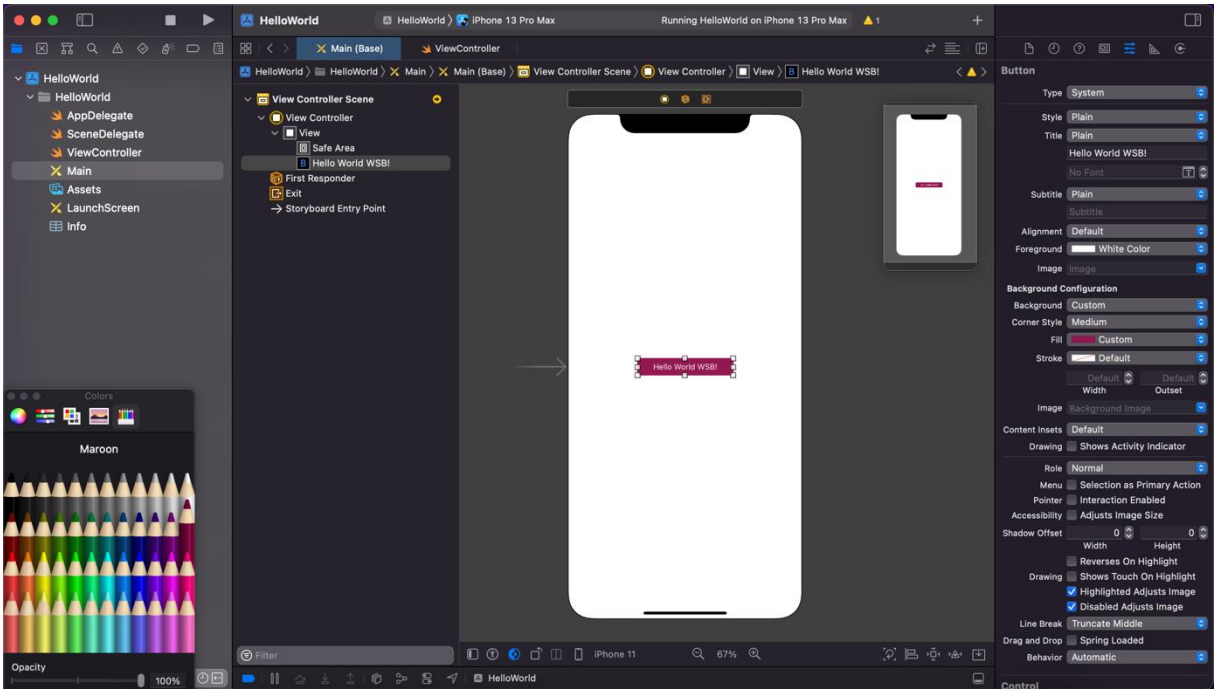
5. Następnie musimy nawiązać połączenie między przyciskiem "Hello World" a metodą showAlert, aby aplikacja reagowała, gdy ktoś dotknie przycisku Hello World.



6. Teraz wybierz, aby przełączyć się z powrotem do Konstruktora interfejsów. Naciśnij i przytrzymaj sterujący klawiatury, kliknij przycisk Hello World i przeciągnij go na ikonę Kontroler widoku. Zwolnij oba przyciski (mysz + klawiatura), a wyskakujące okienko pokazuje opcję w obszarze Wysłane wydarzenia. Wybierz go, aby nawiązać połączenie między przyciskiem a akcją.



7. W dolnej części interfejsu Interface Builder znajdziemy kilka ważnych elementów sterujących do tworzenia ograniczeń. W trzeciej opcji będziemy mieli narzędzia do generowania ograniczeń np. szerokość/wysokość czy też współczynnik proporcji (wysokość vs szerokość wewnątrz tego samego widoku).
8. Jak wspomniano wcześniej, nie trzeba pisać kodu, aby dostosować kontrolkę interfejsu użytkownika. Sprawdźmy, jak łatwo jest zmienić właściwości (np. kolor) przycisku. Wybierz przycisk "Hello World", a następnie kliknij inspektora Atrybuty w obszarze inspektora. Będziemy mieć dostęp do właściwości przycisku. Tutaj możesz zmienić styl przycisku, czcionkę, kolor tekstu, kolor tła itp. Spróbuj zmienić styl jak na poniższym obrazku.



Sprawdź swoją wiedzę.

1. Scharakteryzuj krótko Interface Builder.
2. Scharakteryzuj krótko Storyboard.
3. Scharakteryzuj krótko ViewController. Co można w nim robić? Umieścić?

Praca z symulatorem urządzeń Apple

Symulator to oddzielna aplikacja symulująca urządzenie iOS, takie jak iPhone i iPad. Mamy tutaj oddzielne menu pozwalające na kontrolowanie aplikacji i wykonywanie wielu czynności, tak jak w prawdziwym urządzeniu.

Oto lista wybranych czynności, które można wykonywać w symulatorze:

- wykonywanie zrzutów ekranu,
- obrót symulatora w lewą stronę (Cmd+strzałka w lewo),
- obrót symulatora w prawą stronę (Cmd+strzałka w prawo),
- gest wstrząśnięcia (Cmd+Ctrl+Z),
- przejście do ekranu głównego (Cmd+Shift+H),
- aktywowanie Siri (Cmd+Alt+Shift+H),
- ponowne uruchomienie urządzenia,
- symulowanie TouchID,
- symulowanie ostrzeżenia o niewielkiej ilości wolnej pamięci,
- symulowanie połączenia telefonicznego,
- włączenie klawiatury sprzętowej,
- wymuszenie użycia Force Touch,
- symulowanie monitora zewnętrznego,
- wysyłanie spreparowanych informacji o położeniu
- dostęp do systemowego dziennika zdarzeń,
- zainicjalizowanie synchronizacji z iCloud

Testowanie aplikacji

Xcode ma zintegrowane narzędzie do debugowania. To narzędzie uruchomi aplikację w czasie rzeczywistym, jednocześnie umożliwiając przeglądanie kodu źródłowego wiersz po wierszu, dzięki czemu można sprawdzić wszelkie błędy. Możemy także zobaczyć, ile aplikacja zużywa procesora i ile zasobów używa aplikacja na naszym fizycznym urządzeniu w porównaniu z innymi uruchomionymi aplikacjami. Nawigator testów uruchomia wszelkie dodatkowe testy, które chcemy wykonać.

Symulator iPhone’a

Jeśli nie mamy fizycznego iPhone'a lub iPada możemy wykorzystać iPhone Simulator bezpośrednio z poziomu Xcode. iPhone Simulator uruchamia się dość szybko i jest idealny do ciągłej pracy programistycznej.

Oto jak uruchomić aplikację w symulatorze iPhone'a:

1. Najpierw otwórz projekt iOS w Xcode
2. Następnie w lewym górnym rogu Xcode wybierz symulator, którego chcesz użyć.
3. Użyj krótkiego Command + R, otworzy to symulator, aby uruchomić projekt. Symulator symuluje iPhone'a w systemie macOS. Istnieje szeroka gama symulatorów dostępnych w XCode do uruchamiania aplikacji iOS w systemie macOS.
4. Xcode skompiluje i zbuduje aplikację, zainstaluje ją w symulatorze iPhone'a i uruchomi aplikację. Możesz teraz korzystać z aplikacji w symulatorze iPhone'a, tak jakby to był prawdziwy iPhone!

Uruchamiamy aplikację na iPhonie / iPadzie

1. Uruchamianie aplikacji na iPhonie odbywa się w taki sam sposób, jak uruchamianie jej na symulatorze iPhone'a. Musimy wykonać kilka kroków wcześniej, ale po tym, jak uruchomić aplikację.
2. Najpierw otwórz projekt iOS w Xcode.
3. Następnie podłącz iPhone'a do komputera Mac przez USB i wybierz Zaufaj na iPhonie, gdy pojawi się monit.
4. Następnie w lewym górnym rogu Xcode wybierz urządzenie iPhone
5. Na koniec kliknij przycisk Uruchom /Odtwórz lub naciśnij + R
6. Zanim będziesz mógł używać iPhone'a do debugowania aplikacji, musisz utworzyć bezpłatne konto Apple Developer ponieważ potrzebujemy Apple ID!
7. Za każdym razem, gdy aktualizujesz iPhone'a do nowej wersji iOS, Xcode będzie musiał pobrać tak zwane symbole debugowania z iPhone'a. Gdy tak się stanie, zobaczysz powiadomienie na pasku stanu Xcode. Symbole te są potrzebne do symbolizowania raportów o awariach. Symbole debugowania mogą gromadzić sporo gigabajtów pamięci, więc jeśli chcesz je od czasu do czasu usunąć, sprawdź folder ~ / Library / Developer / Xcode / DerivedData / na komputerze Mac.

Jeśli tworzymy aplikację do użytku produkcyjnego, nie testujemy tylko aplikacji za pomocą symulatora iPhone'a! Przetestujmy ją również na prawdziwym iPhonie.

Dodawanie aplikacji do sklepu App Store

Konto Apple Developer daje dostęp do dwóch rzeczy; narzędzia programistyczne Apple oraz możliwość przesyłania aplikacji do App Store lub Mac App Store. Powszechnie wiadomo, że aby założyć konto Apple Developer, należy uiścić roczną opłatę w wysokości 100 USD, ale jest to potrzebne tylko wtedy, gdy trzeba przesłać aplikację do jednego ze sklepów Apple. Jeśli chcemy tylko uzyskać dostęp do niezliczonych narzędzi, które Apple ma dla programistów wystarczy utworzone bezpłatne konto.

Aby utworzyć bezpłatne konto programisty Apple, wystarczy zalogować się do portalu Apple Developer za pomocą zwykłego identyfikatora Apple ID. Proces wprowadzania aplikacji do sklepu z aplikacjami Apple jest dość prosty ponieważ bezpośrednio Xcode bardzo ułatwia udostępnianie aplikacji z wykorzystaniem mechanizmu App Store Connect.

App Store Connect umożliwia monitorowanie sprzedaży, wyświetlanie raportów, odpowiadanie na recenzje i nie tylko. Każda aplikacja musi zostać zatwierdzona przez zespół Apple w zakresie wytycznych technicznych, projektowych i dotyczących treści, zanim zostanie umieszczona w sklepie z aplikacjami. Ten proces zatwierdzania może potrwać 2-3 tygodnie.

Sprawdź swoją wiedzę

1. Omów główne funkcje symulatora aplikacji Apple w tym iOS.
2. W jaki sposób dodać aplikację do sklepu App Store?