Programowanie mobilne

Wstęp do programowania w systemie iOS

dr inż. Andrzej Wilczyński - www.awilczynski.me

iOS (1)

- Platforma/System operacyjny dla urządzeń mobilnych (nie tylko telefonów/tabletów)
- System zamknięty – Apple uniemożliwia wprowadzanie własnych modyfikacji
- iOS SDK (iPhone SDK) - kompletny stos elementów i narzędzi służących do tworzenia aplikacji na system iOS

Dokumentacja dla deweloperów jest dostępna tutaj:

https://developer.apple.com/documentation/

iOS (2)

Platformę iOS można podzielić na cztery abstrakcyjne warstwy. Są to:

1. Core OS – najniższa warstwa, zapewniająca interakcję między sprzętem a oprogramowaniem w której skład wchodzi jądro systemu Darwin.
2. Core Services – zestaw podstawowych bibliotek do zarządzania pracą aplikacji i wątków, obsługą sieci i bazy danych.

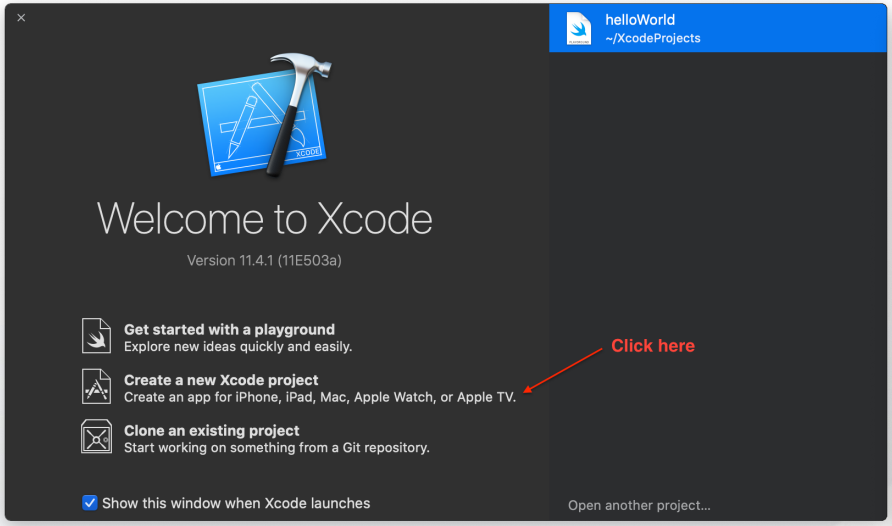
iOS (3)

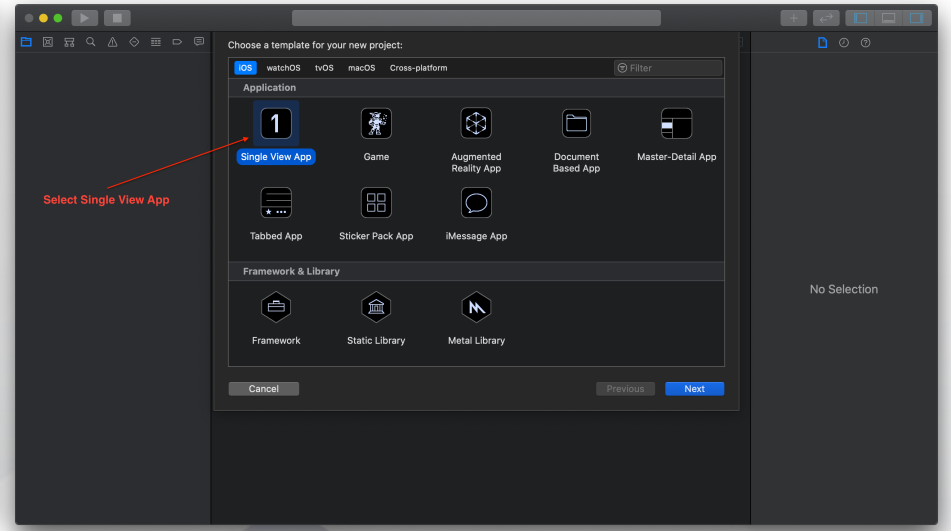
3. Media – warstwa w której skład wchodzi obsługa obrazu i dźwięku, odtwarzanie wideo itp.
4. Cocoa Touch – biblioteka interfejsu użytkownika z wykorzystaniem ekranu dotykowego.

Instalacja i konfiguracja środowiska

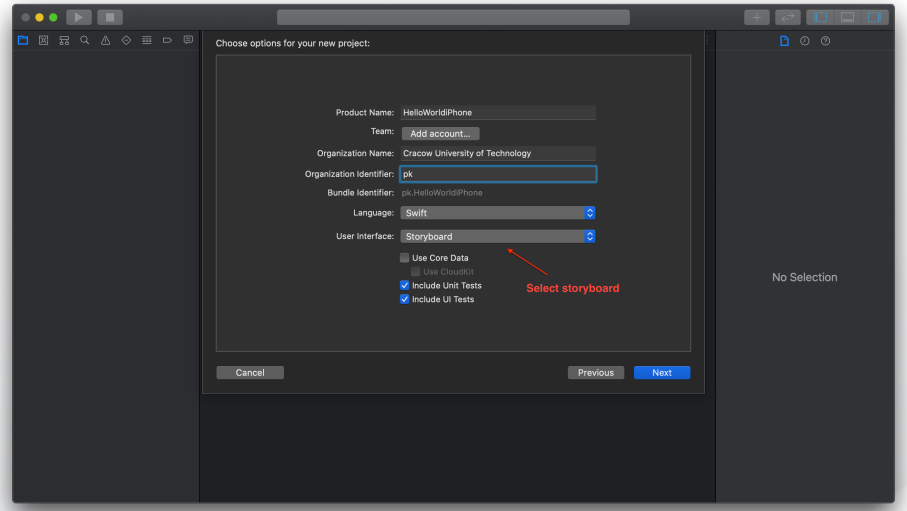
- System MaC OS X – (niekoniecznie potrzebne urządzenie, np. VirtualBox)
- XCode – wersja 10 lub 11

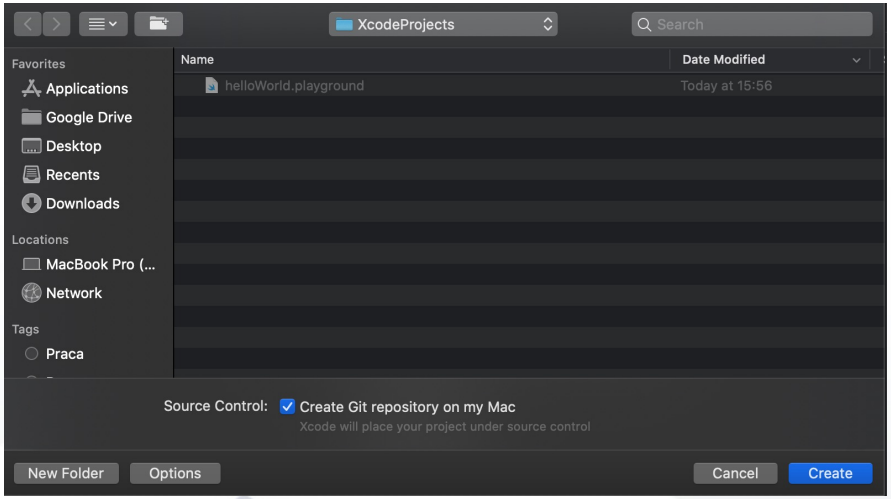
Tworzenie projektu w XCode 11 (1)



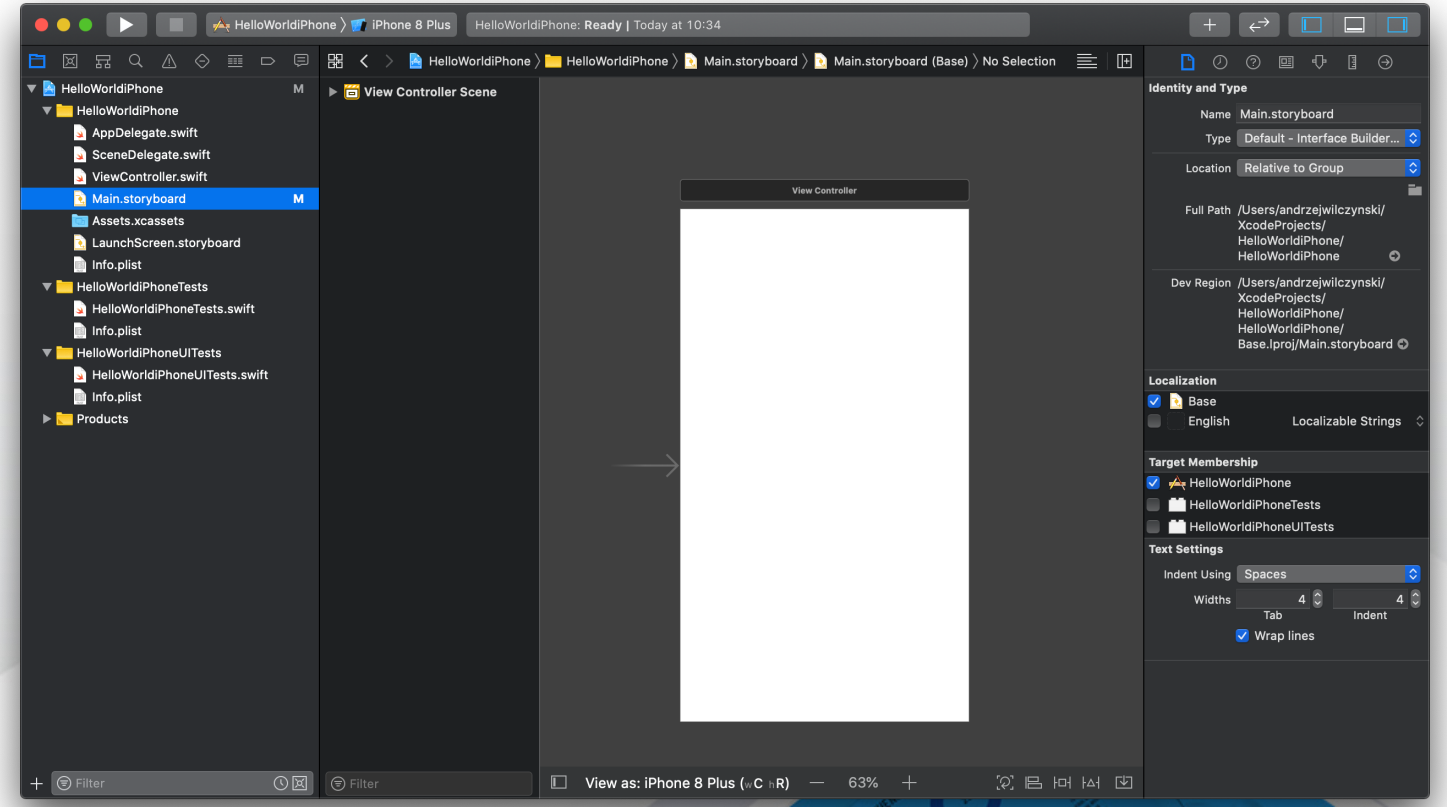


Tworzenie projektu w XCode 11 (2)

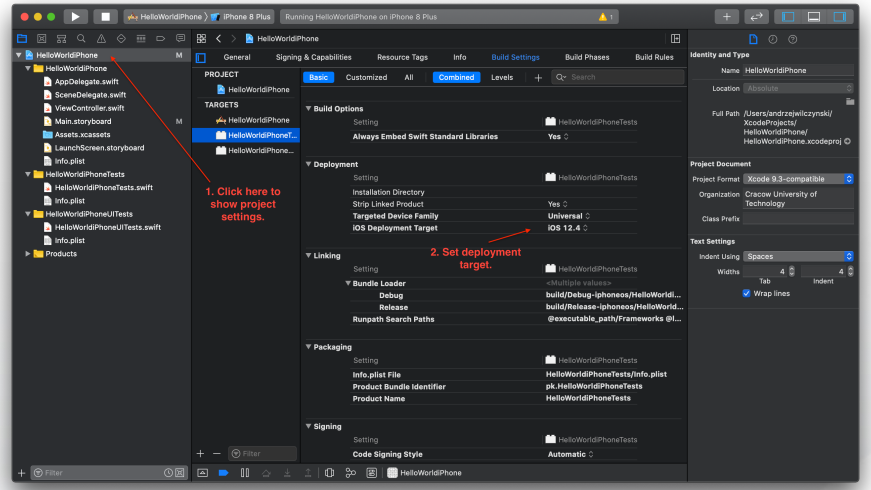


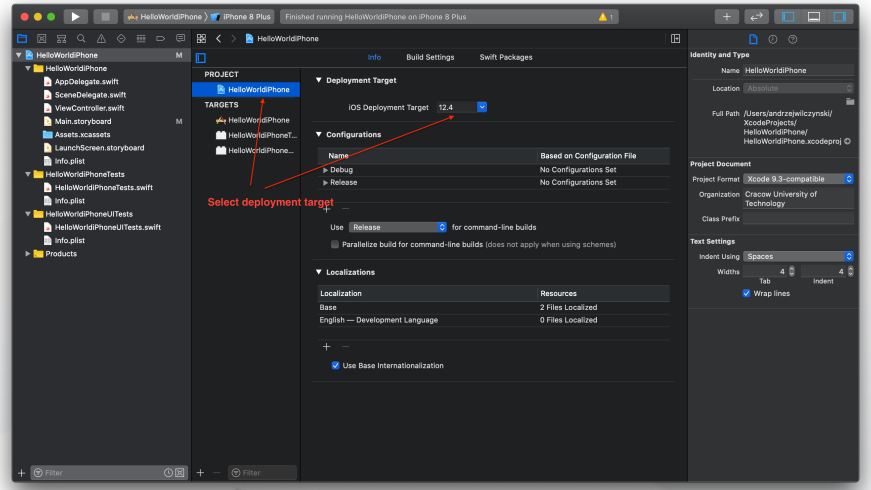


Tworzenie projektu w XCode 11 (3)



Właściwości projektu – deployment itp.

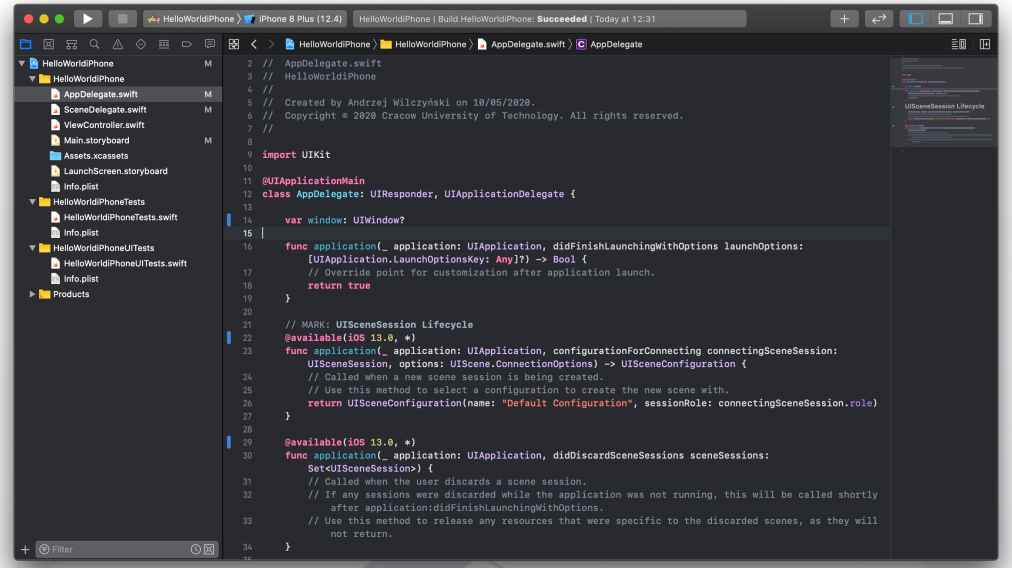




AppDelegate

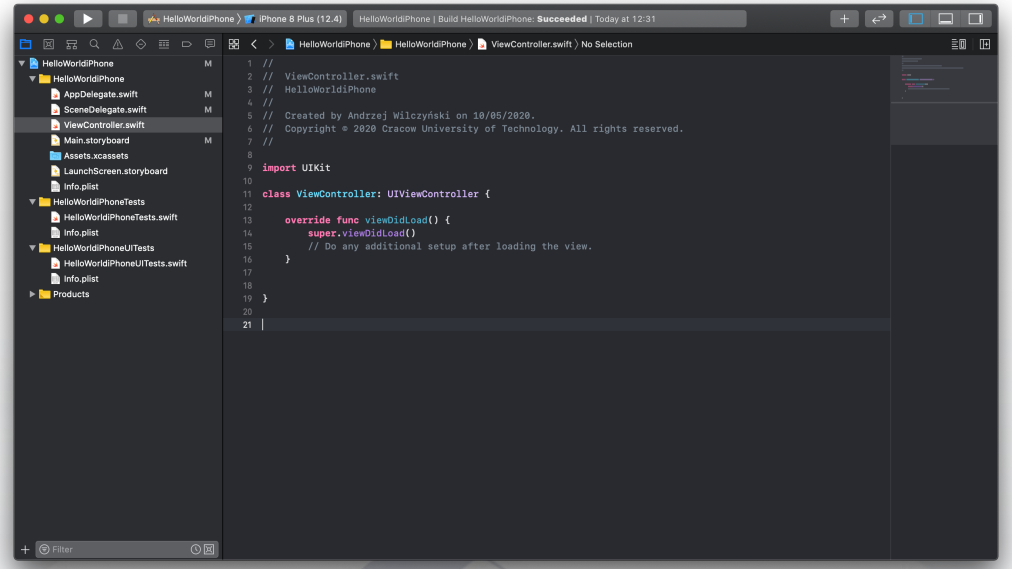
Delegat aplikacji, zawiera m.in. takie metody jak:

- `didFinishLaunchingWithOptions` - uruchamiana zaraz po starcie aplikacji
- `applicationWillResignActive` – uruchamiana przed przejściem aplikacji w stan nieaktywny
- `applicationDidEnterBackground` – uruchamiana po przejściu w tło
- `applicationWillTerminate` – wywoływana przed całkowitym zamknięciem aplikacji



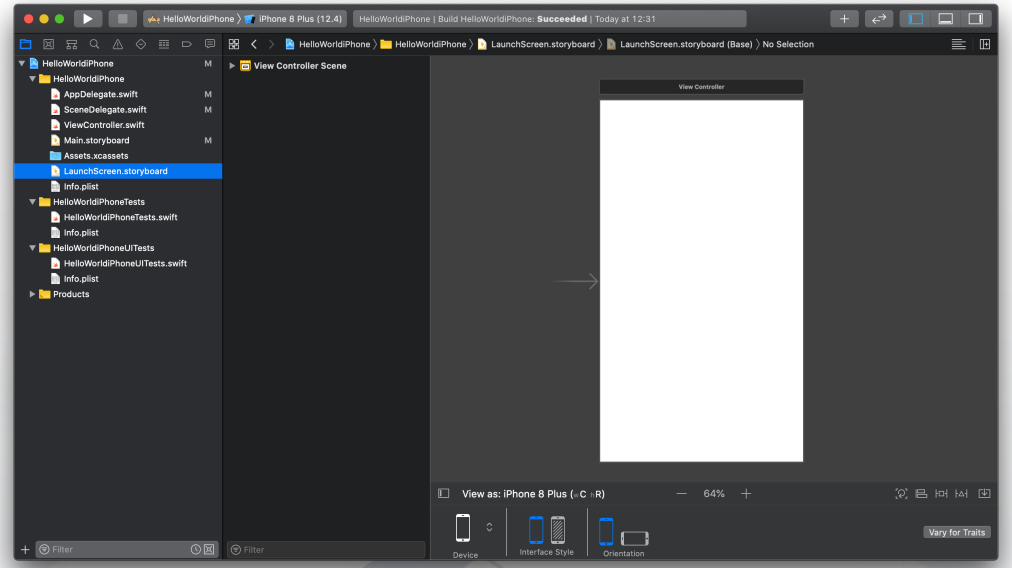
ViewController

Kontroler odpowiedzialny za zarządzanie widokiem aplikacji.



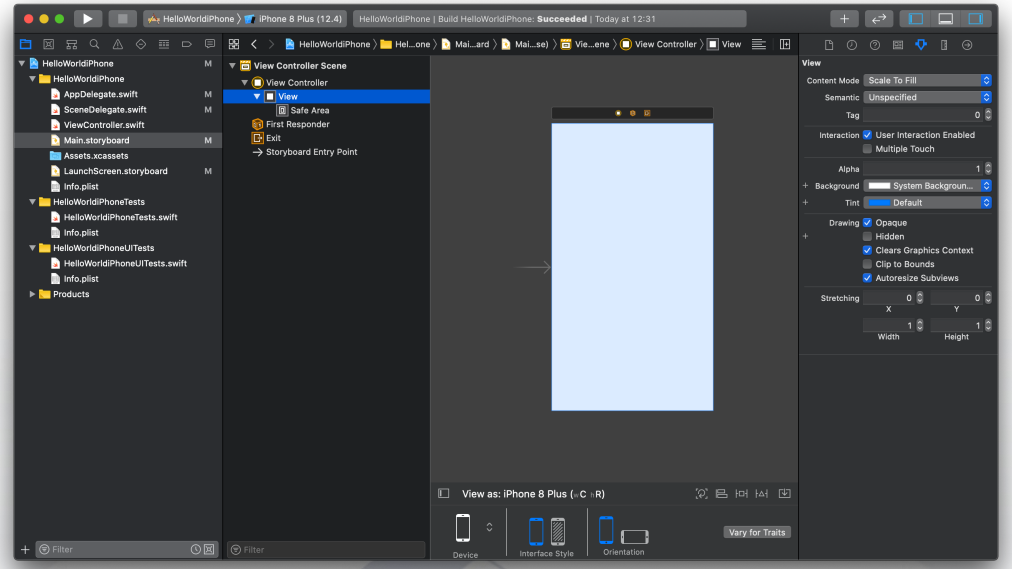
LaunchScreen.storyboard

Plik startowy w którym można zdefiniować rzeczy, które będą wyświetlane w momencie startu aplikacji.



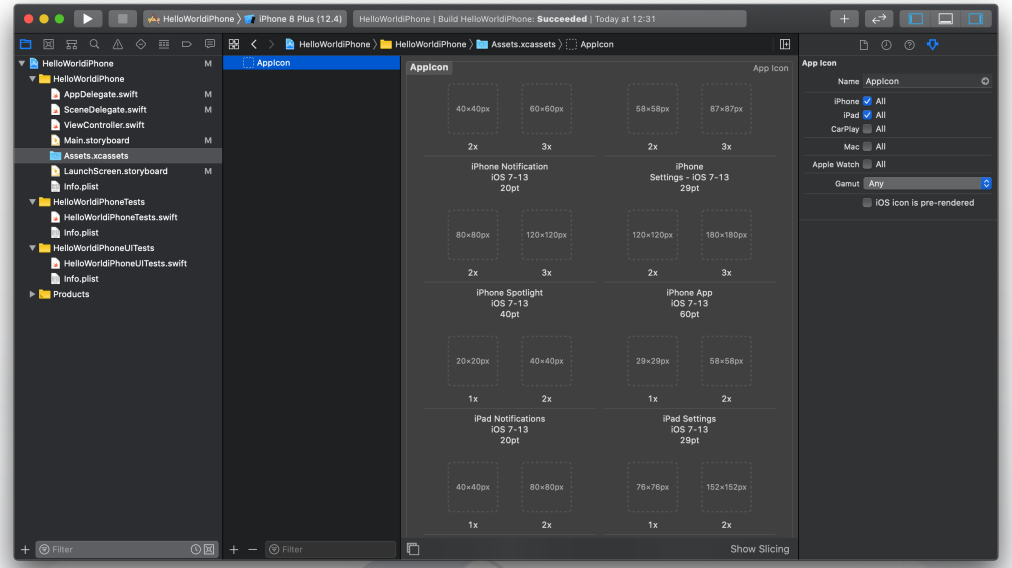
Main.storyboard

Punkt startowy dla interfejsu użytkownika, wizualna reprezentacja interfejsu, przedstawiająca ekrany i połączenia między nimi. Składa się z sekwencji scen, z których każda reprezentuje kontroler widoku i jego widoki.



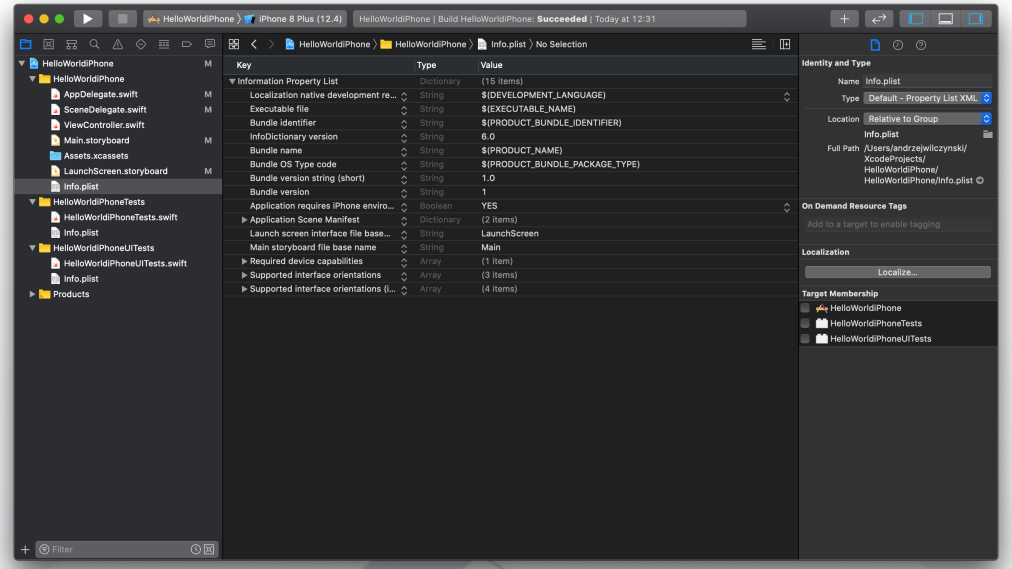
Katalog `Assets.xcassets`

Miejsce na media,
z których będzie
korzystać aplikacja.

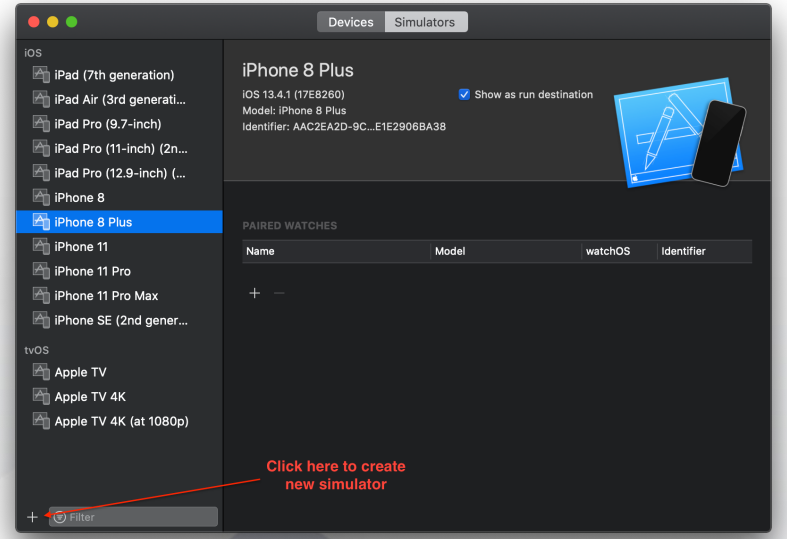


Plik `Info.plist`

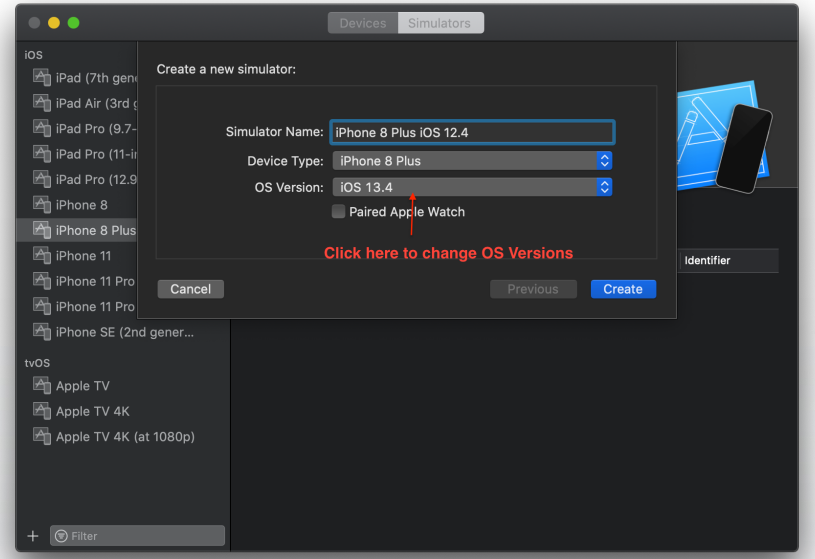
Plik w formacie xml zawierający wpisy konfiguracyjne aplikacji.

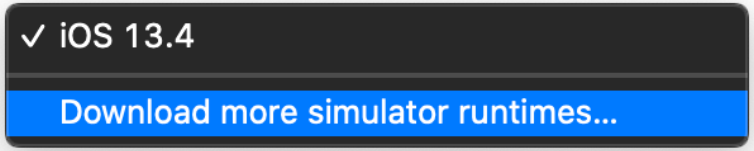


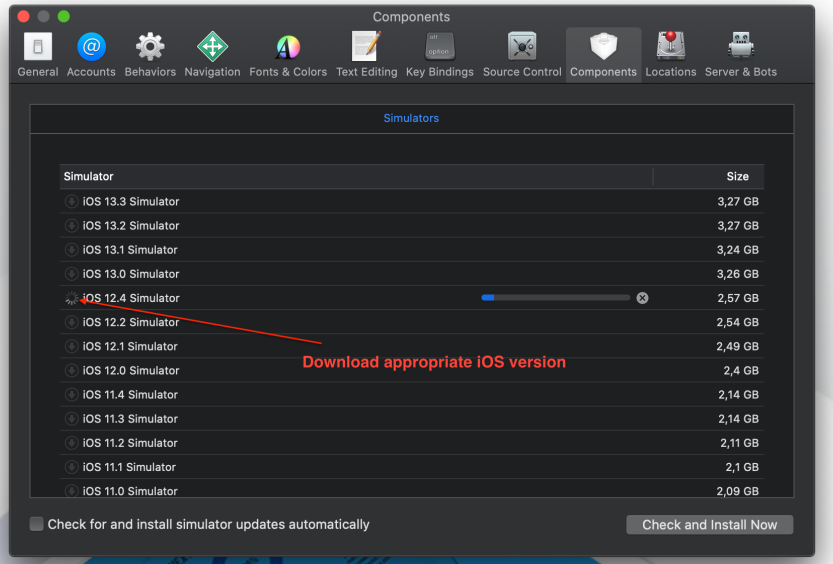
Konfiguracja symulatora dla iOS 12



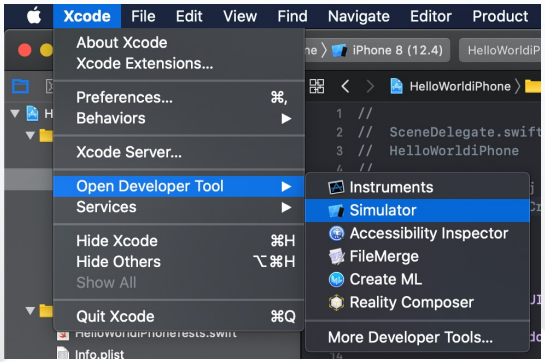
Konfiguracja symulatora dla iOS 12



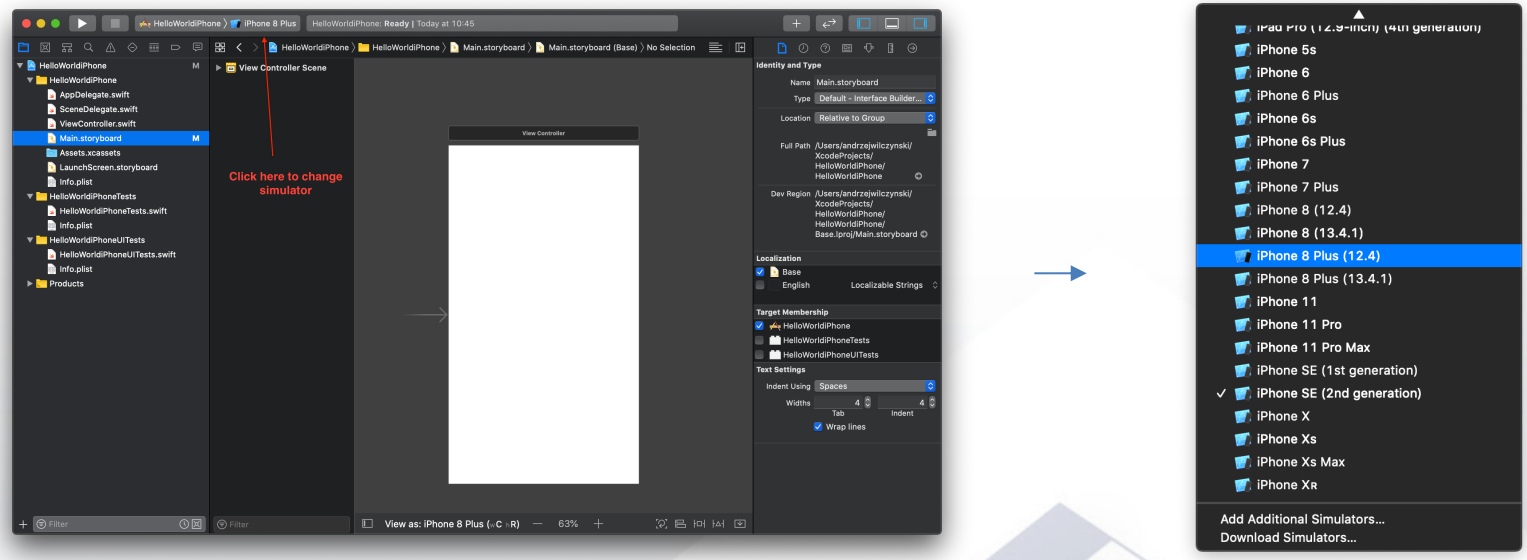




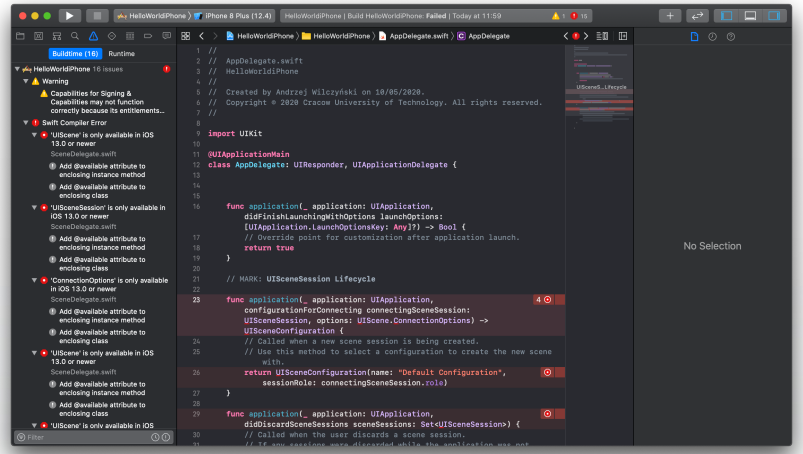
Uruchamianie symulatora

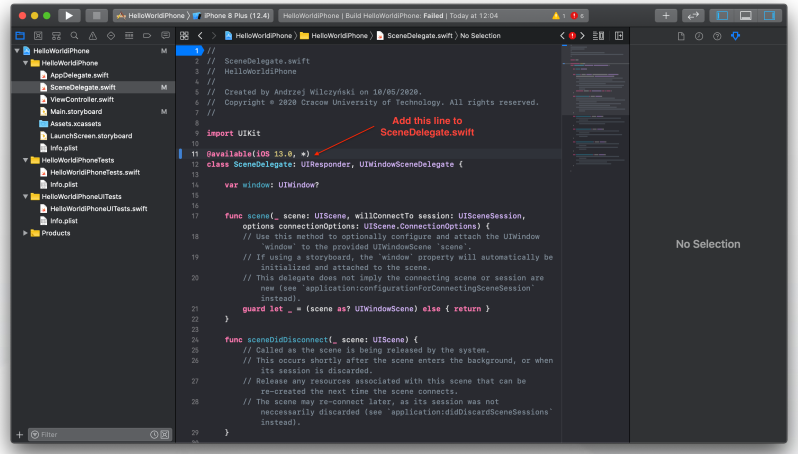


Uruchamianie aplikacji stworzonej w Xcode 11 na iOS 12 (1)

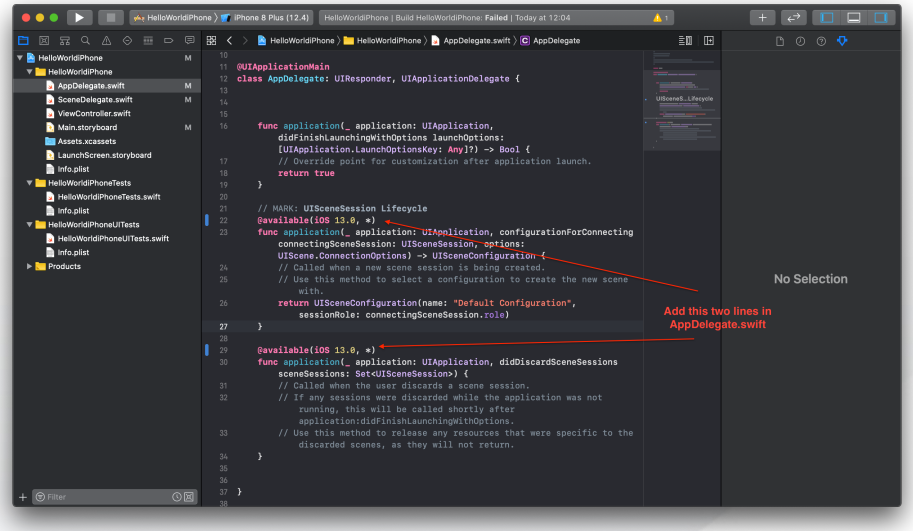


Uruchamianie aplikacji stworzonej w Xcode 11 na iOS 12 (2)

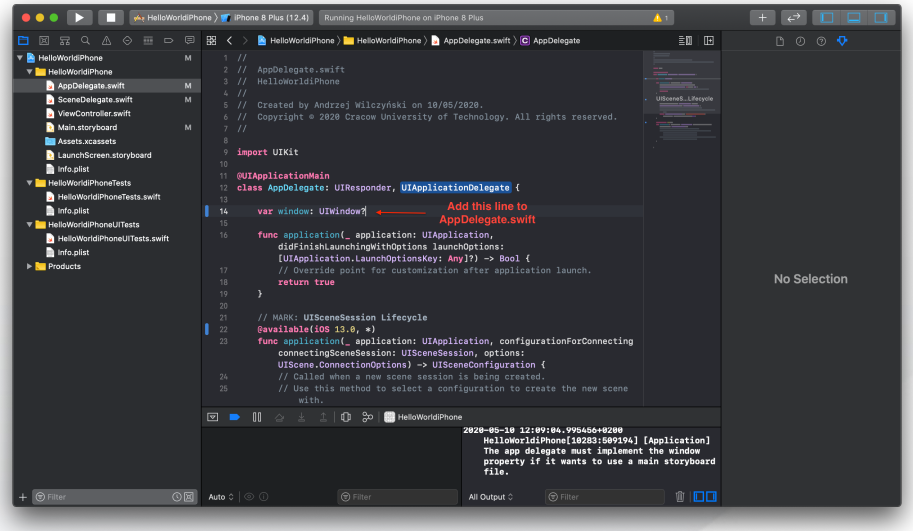


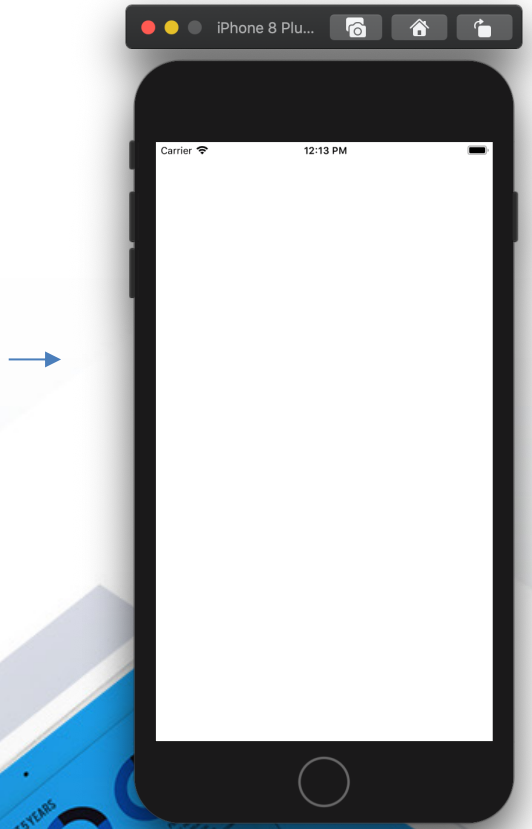


Uruchamianie aplikacji stworzonej w Xcode 11 na iOS 12 (3)

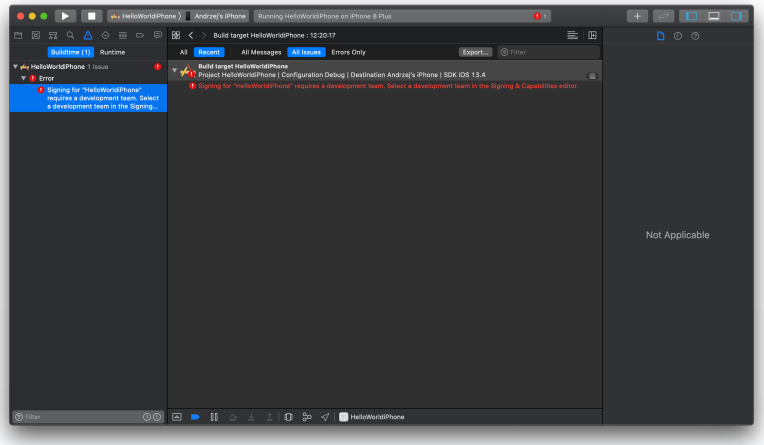


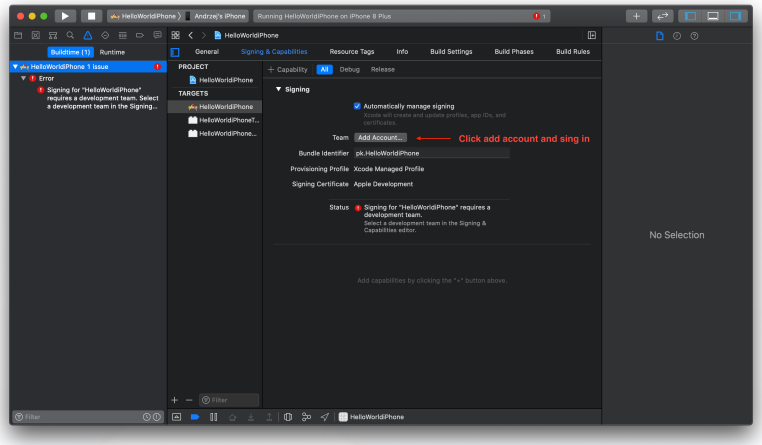
Uruchamianie aplikacji stworzonej w Xcode 11 na iOS 12 (4)

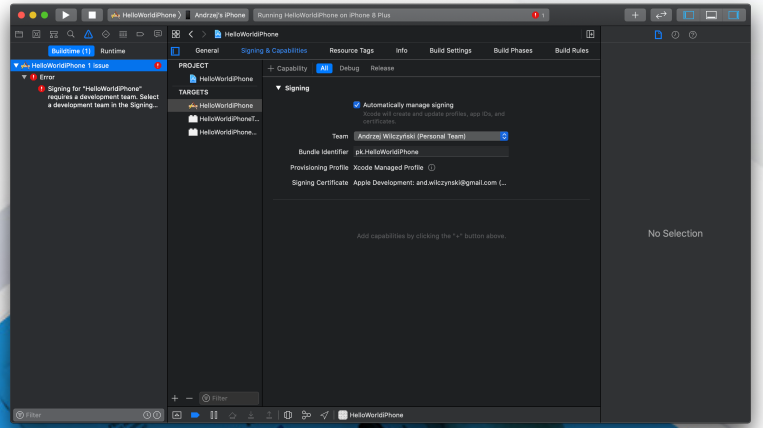




Uruchamianie aplikacji na urządzeniu fizycznym







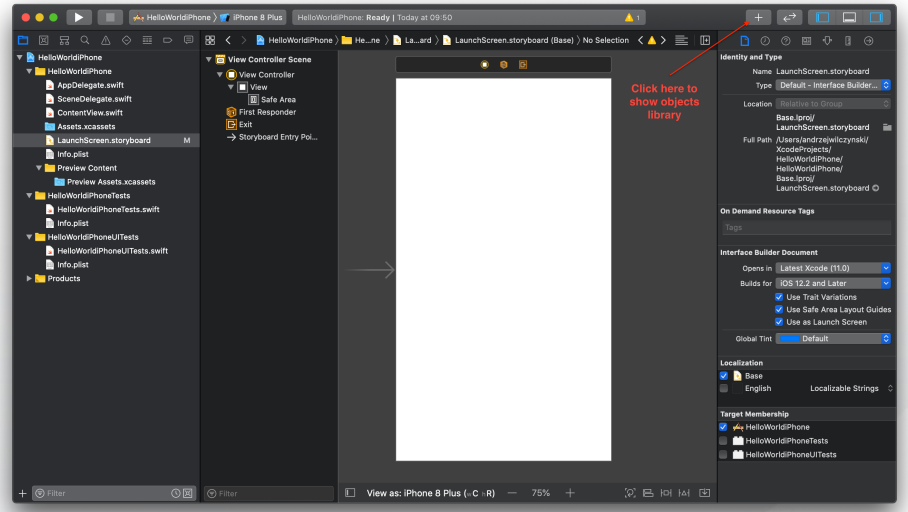
Storyboard

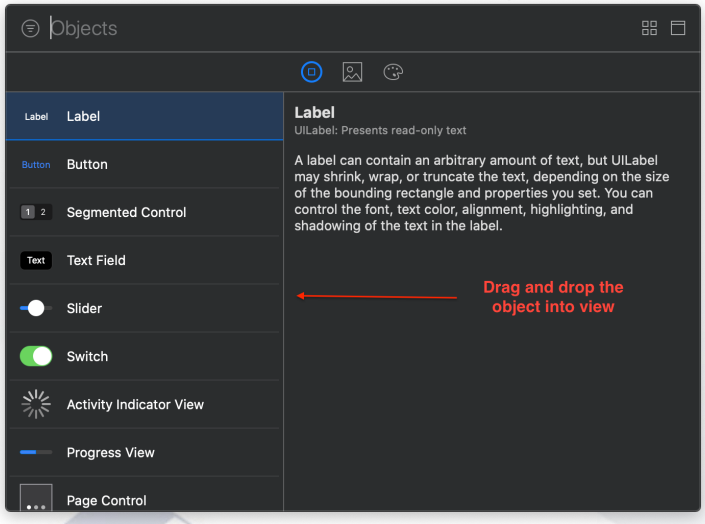
Storyboard składa się z wielu ekranów (widoków), każdy z tych widoków powinien być samowystarczalny.
W aplikacji Single View App domyślnie jest zdefiniowany jeden widok początkowy, który jest widokiem pustym. Możemy dodawać do niego elementy korzystając z *Interface Builder'a*.

Interface Builder (1)

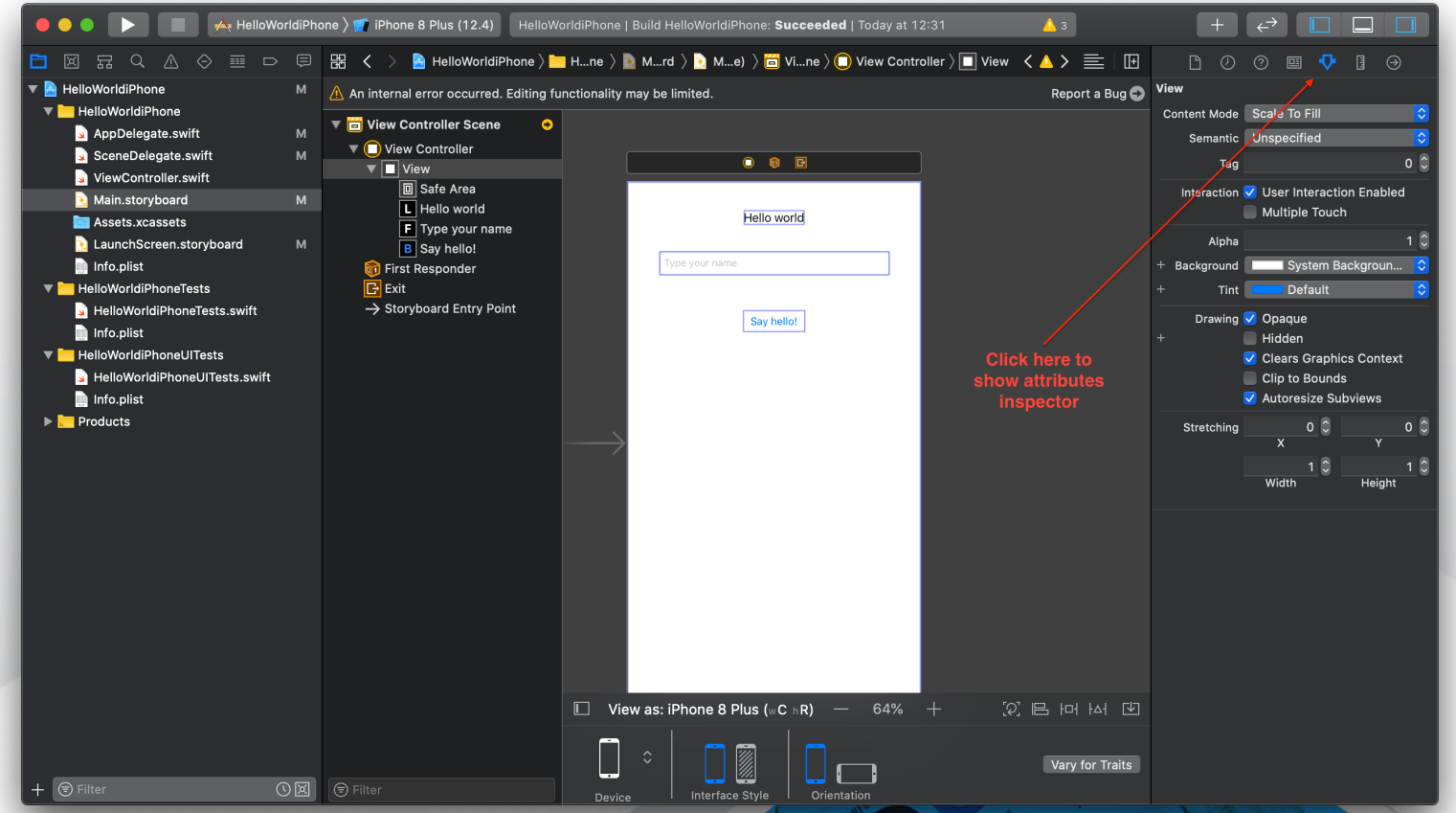
Moduł przeznaczony do umieszczania elementów interfejsu użytkownika w projekcie. W module *Interface Builder* możesz zobaczyć dużą strzałkę skierowaną do widoku. Ta strzałka wskazuje punkt początkowy sceny i prowadzi do widoku, który zostanie wczytany jako pierwszy po uruchomieniu aplikacji.

Interface Builder (2)



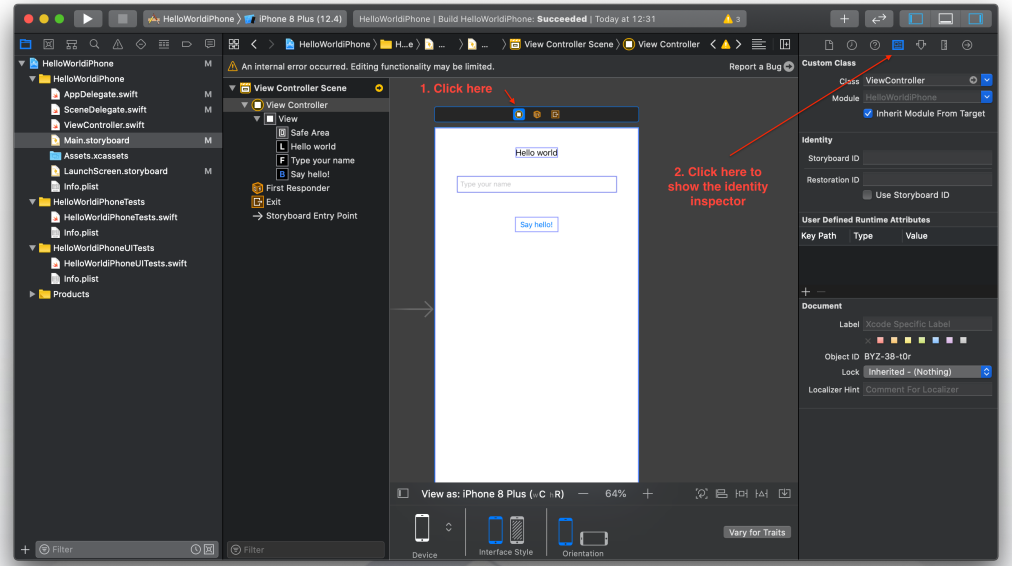


Interface Builder (3)



`ViewController`

Każdy widok musi mieć zdefiniowane narzędzie, którym można go kontrolować. W tym celu możemy stworzyć podklasę klasy `UIViewController`. Domyślnie jest już utworzona taka klasa i nazywa się ViewController.



Typy kontrolerów widoku

- System – elementy SDK
- Content – tworzone samodzielnie
- Containers – widoki i przepływ aplikacji

UIView

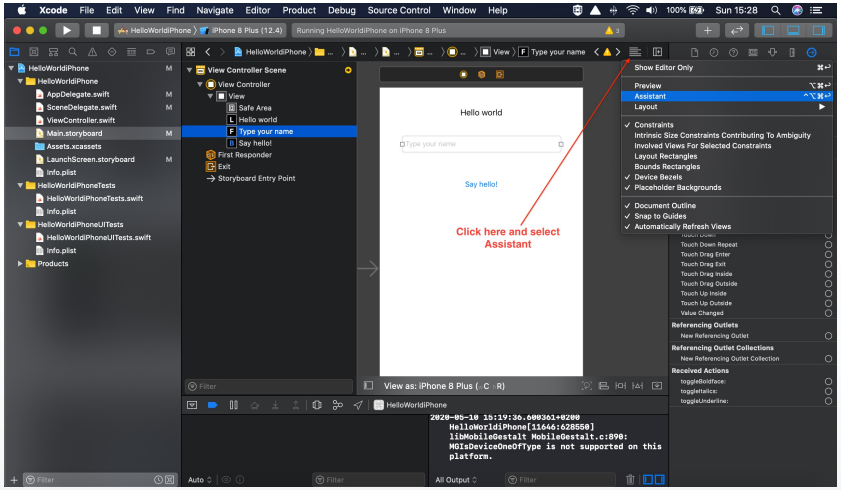
Jest to podstawa wszystkich kontrolek. Klasa `UIView` może być traktowana jako jedna z cegiełek, która jest budulcem konkretnej sceny. Kilka cegiełek połączonych ze sobą tworzą scenę przedstawiającą widok aplikacji. Każdy element umieszczony na scenie charakteryzuje się m.in. takimi atrybutami jak: `frame`, `bounds`, `center`, `transform` czy `alpha`.

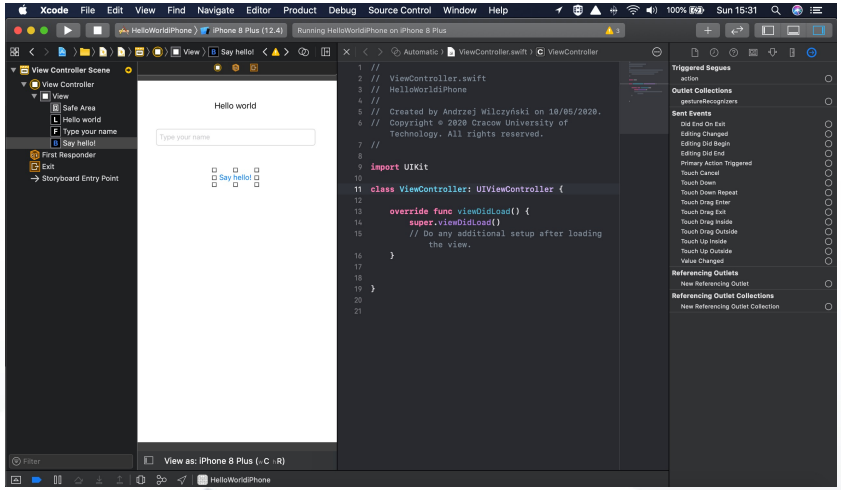
Outlety i akcje

Outlet pozwala obiektom w kodzie na odwoływanie się do elementów utworzonych w pliku .storyboard.

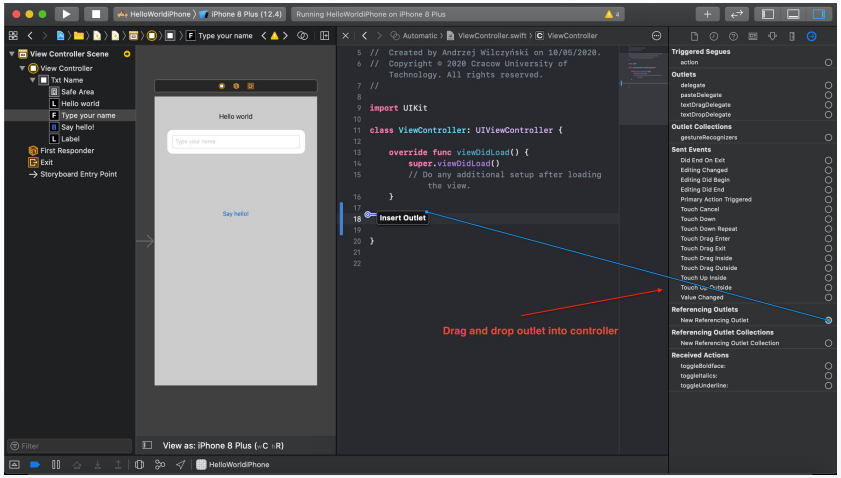
Akcje mapują metody w kodzie i elementy interfejsu użytkownika na zdarzenia.

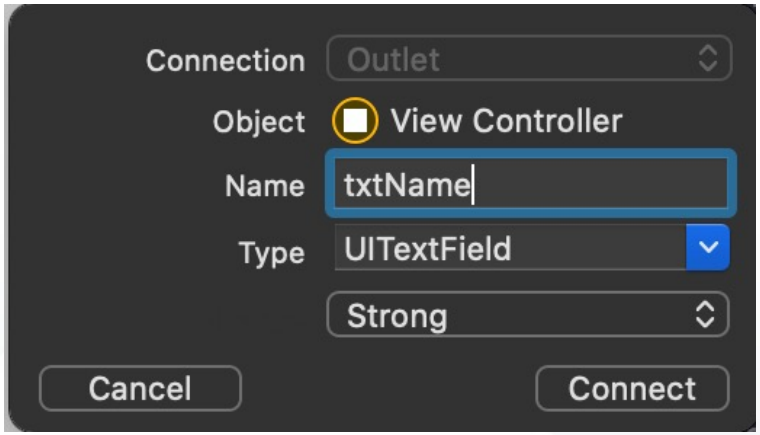
Assistant Editor



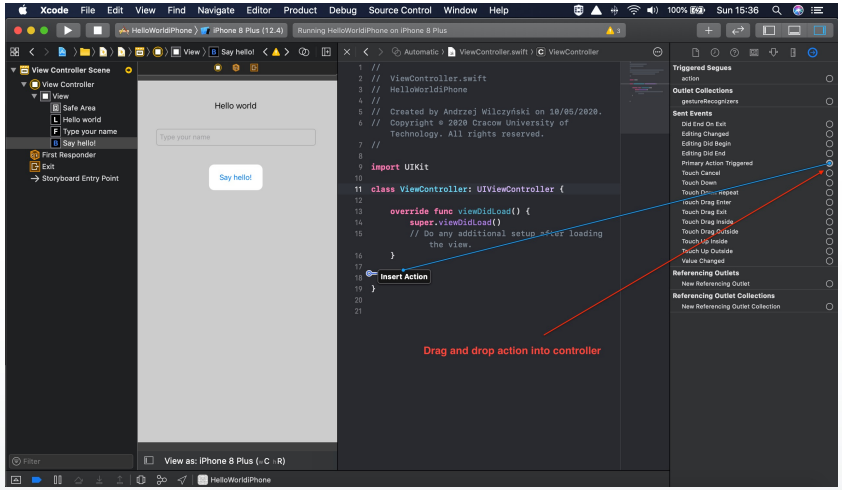


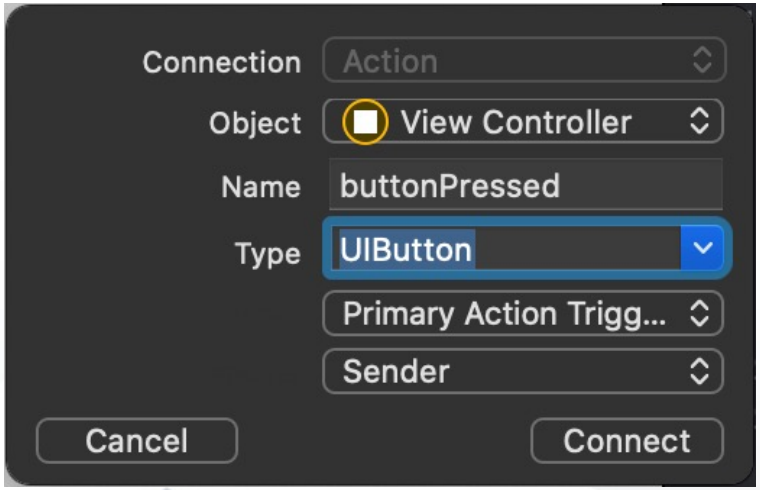
Dodawanie outletu



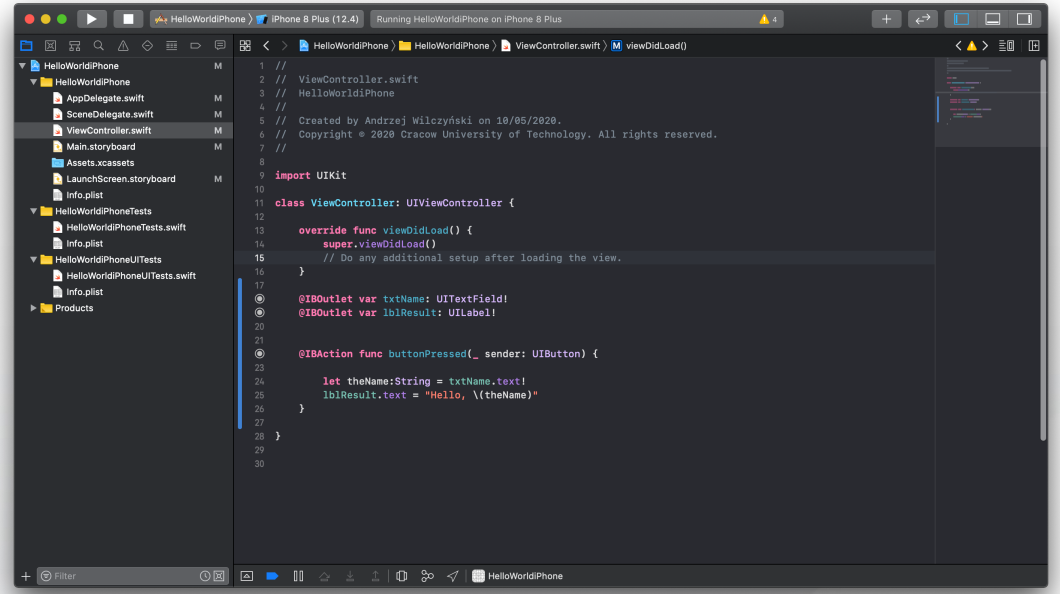


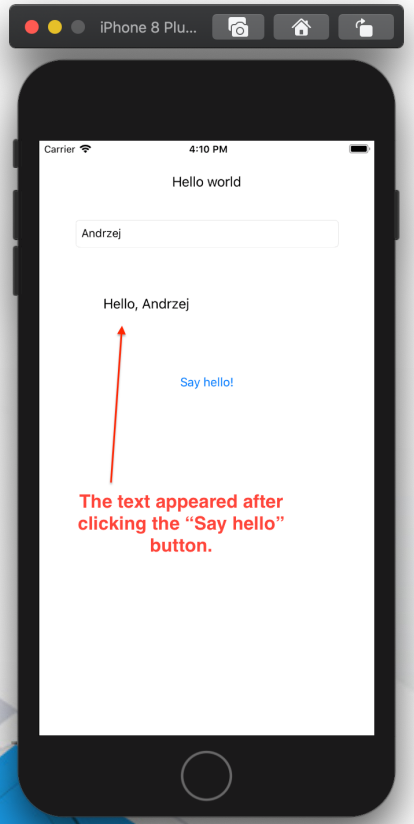
Dodawanie akcji





Przykład


HelloWorldiPhone
iPhone 8 Plus (12.4)
Running HelloWorldiPhone on iPhone 8 Plus
HelloWorldiPhone
HelloWorldiPhone
AppDelegate.swift
SceneDelegate.swift
ViewController.swift
Main.storyboard
Assets.xcassets
LaunchScreen.storyboard
Info.plist
HelloWorldiPhoneTests
HelloWorldiPhoneTests.swift
Info.plist
HelloWorldiPhoneUITests
HelloWorldiPhoneUITests.swift
Info.plist
Products
ViewController.swift
viewDidLoad()
// ViewController.swift
// HelloWorldiPhone
// Created by Andrzej Wilczyński on 10/05/2020.
// Copyright © 2020 Cracow University of Technology. All rights reserved.
import UIKit
class ViewController: UIViewController {
override func viewDidLoad() {
super.viewDidLoad()
// Do any additional setup after loading the view.
}
@IBOutlet var txtName: UITextField!
@IBOutlet var lblResult: UILabel!
@IBAction func buttonPressed(_ sender: UIButton) {
let theName:String = txtName.text!
lblResult.text = "Hello, \(theName)"
}
}


iPhone 8 Plu...
Carrier
4:10 PM
Hello world
Andrzej
Hello, Andrzej
Say hello!
The text appeared after clicking the “Say hello” button.

Dziękuję! ☺

Bibliografia

1. Mark A. Lassoff, Tom Stachowitz, *Podstawy jezyka Swift. Programowanie aplikacji dla platformy iOS*, Helion, 2016
2. Matt Neuburg, *OS 12. Wprowadzenie do programowania w Swifcie. Wydanie V*, Helion, 2019