



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.

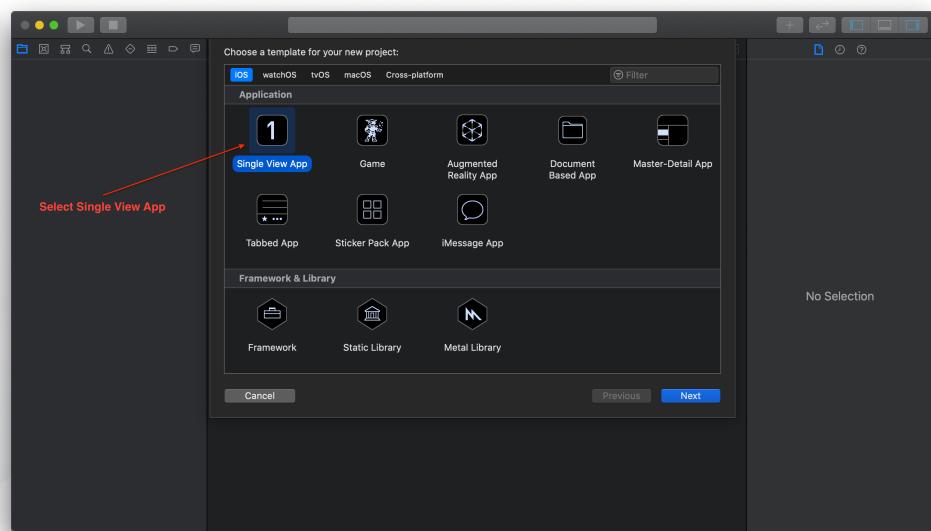


Instalacja i konfiguracja środowiska

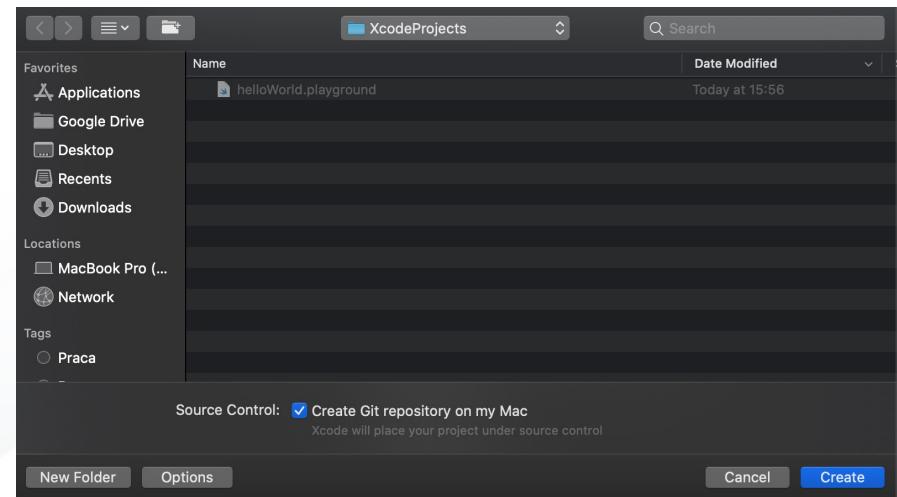
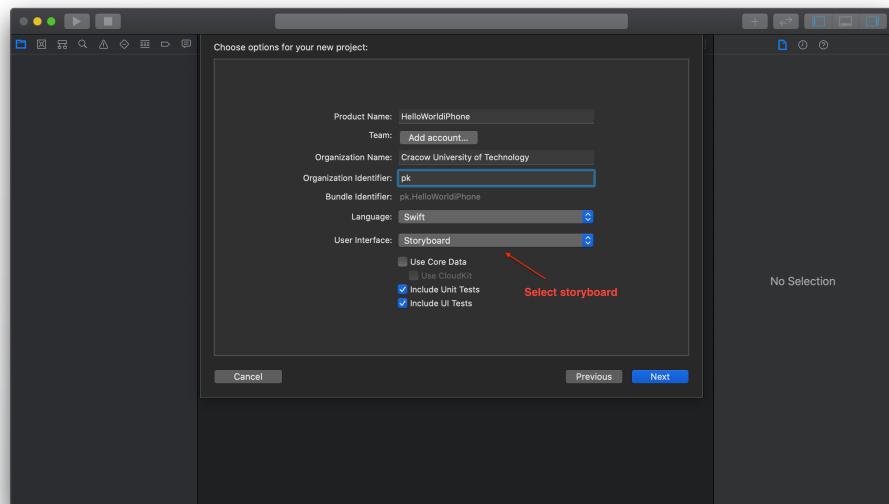
- System MaC OS X – (niekoniecznie potrzebne urządzenie, np. VirtualBox)
- XCode – wersja 10 lub 11



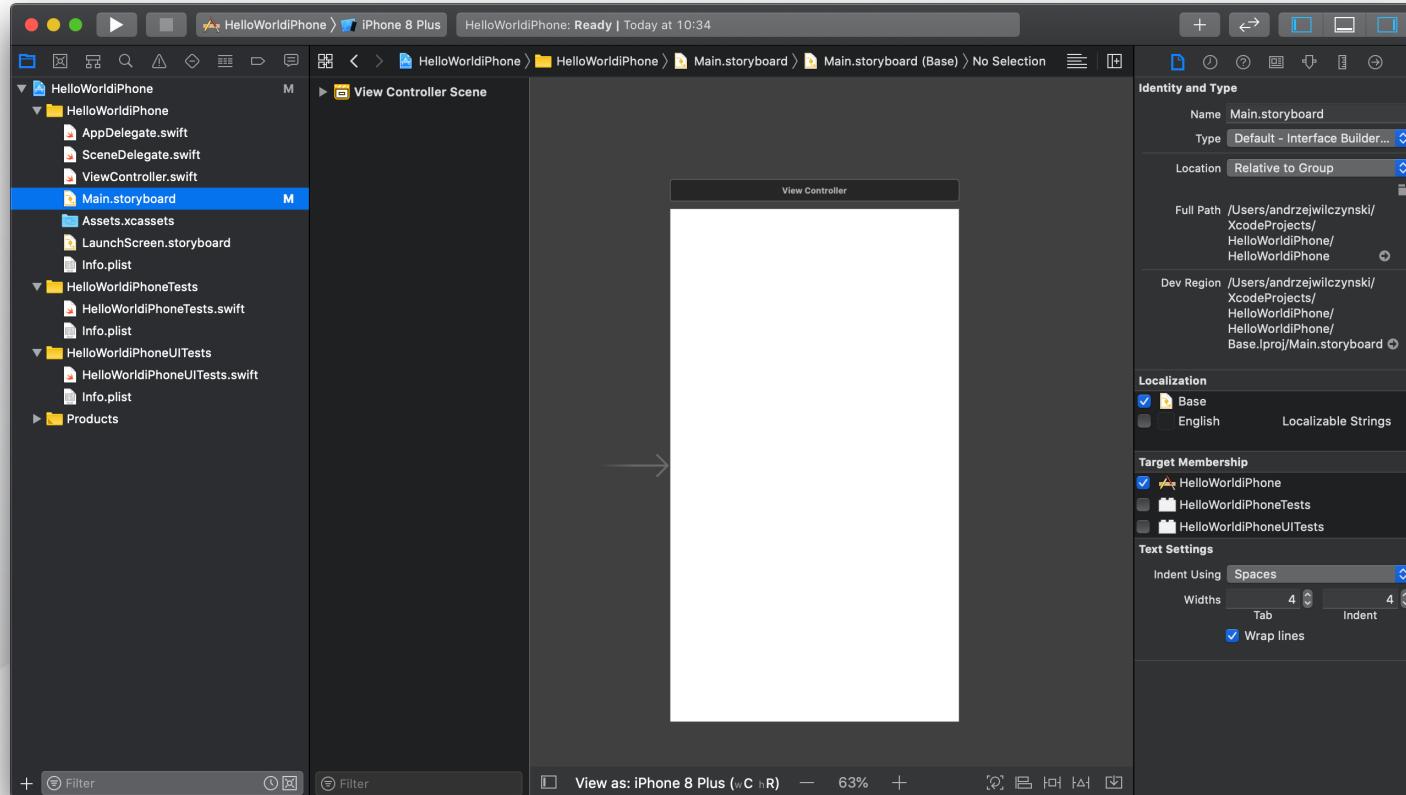
Tworzenie projektu w XCode 11 (1)



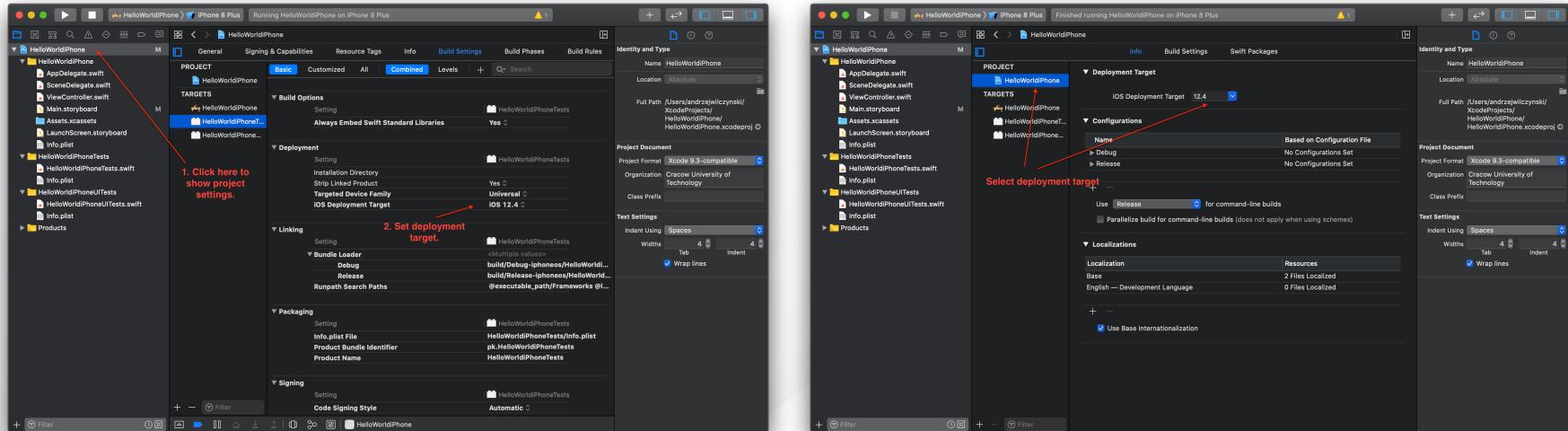
Tworzenie projektu w XCode 11 (2)



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

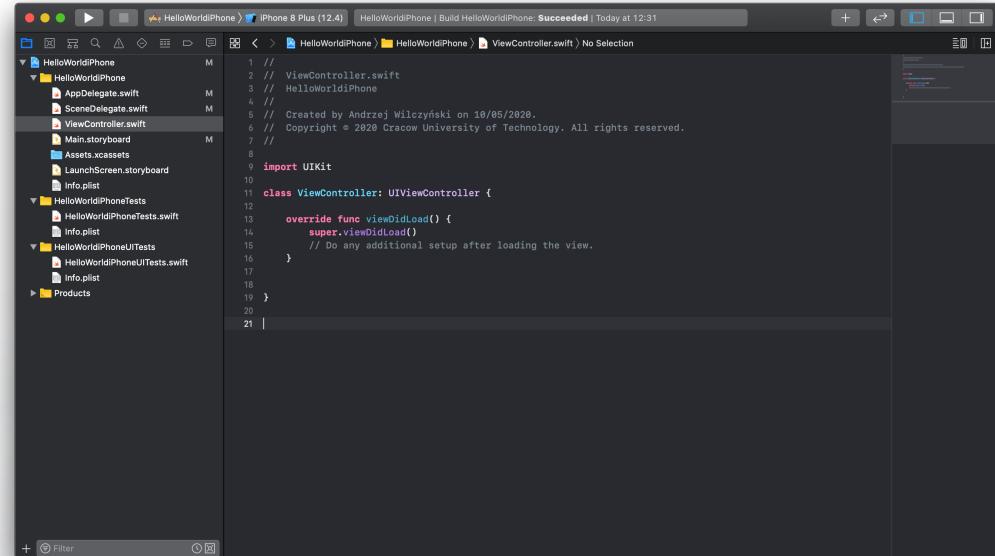


A screenshot of a Mac OS X desktop showing a Xcode project named "HelloWorldiPhone". The project structure is visible in the left sidebar, showing files like AppDelegate.swift, ViewController.swift, Main.storyboard, and Info.plist. The main editor window displays the contents of AppDelegate.swift. The code is written in Swift and defines the AppDelegate class, which implements the UIApplicationDelegate protocol. It includes methods for handling application launch options, scene sessions, and discarding scene sessions.

```
2 // AppDelegate.swift
3 // HelloWorldiPhone
4 //
5 // Created by Andrzej Wilczyński on 10/05/2020.
6 // Copyright © 2020 Cracow University of Technology. All rights reserved.
7 //
8 import UIKit
9
10 @UIApplicationMain
11 class AppDelegate: UIResponder, UIApplicationDelegate {
12
13     var window: UIWindow?
14
15     func application(_ application: UIApplication, didFinishLaunchingWithOptions launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
16         // Override point for customization after application launch.
17         return true
18     }
19
20     // MARK: UISceneSession Lifecycle
21     @available(iOS 13.0, *)
22     func application(_ application: UIApplication, configurationForConnecting connectingSceneSession: UISceneSession, options: UIScene.ConnectionOptions) -> UISceneConfiguration {
23         // Called when a new scene session is being created.
24         // Use this method to select a configuration to create the new scene with.
25         // Return a UISceneConfiguration(name: "Default Configuration", sessionRole: connectingSceneSession.role)
26     }
27
28     @available(iOS 13.0, *)
29     func application(_ application: UIApplication, didDiscardSceneSessions sceneSessions: Set) {
30         // Called when the user discards a scene session.
31         // If any sessions were discarded while the application was not running, this will be called shortly
32         // after application:didFinishLaunchingWithOptions.
33         // Use this method to release any resources that were specific to the discarded scenes, as they will
34         // not return.
35     }
36 }
```

ViewController

Kontroler
odpowiedzialny za
zarządzanie widokiem
aplikacji.



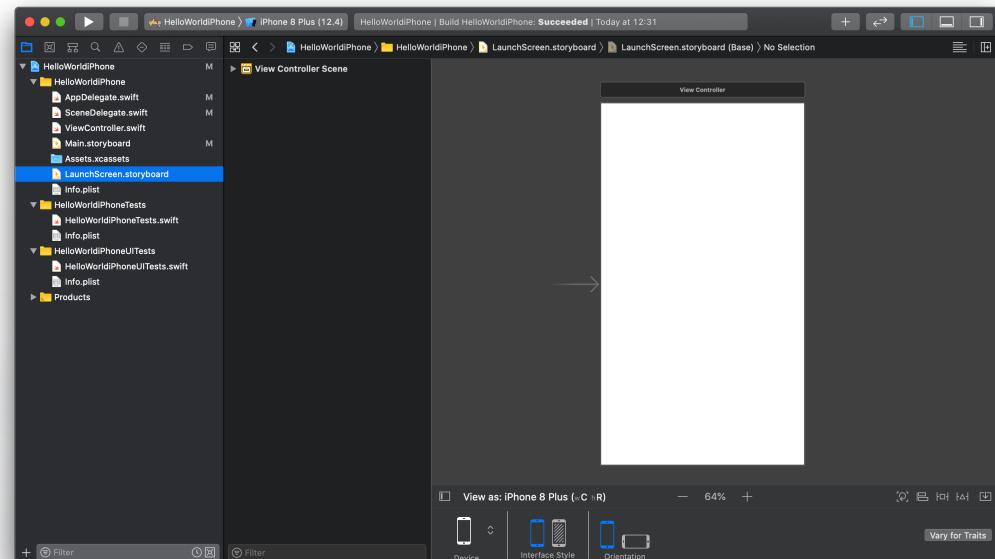
The screenshot shows the Xcode interface with the project 'HelloWordiPhone' open. The 'ViewController.swift' file is selected in the center editor window. The code in the file is:

```
1 // ViewController.swift
2 // HelloWorldiPhone
3 // Copyright © 2020 Cracow University of Technology. All rights reserved.
4 //
5 // Created by Andrzej Wilczyński on 10/05/2020.
6 // Import UIKit
7 import UIKit
8
9 class ViewController: UIViewController {
10
11     override func viewDidLoad() {
12         super.viewDidLoad()
13         // Do any additional setup after loading the view.
14     }
15
16 }
17
18
19
20
21 }
```



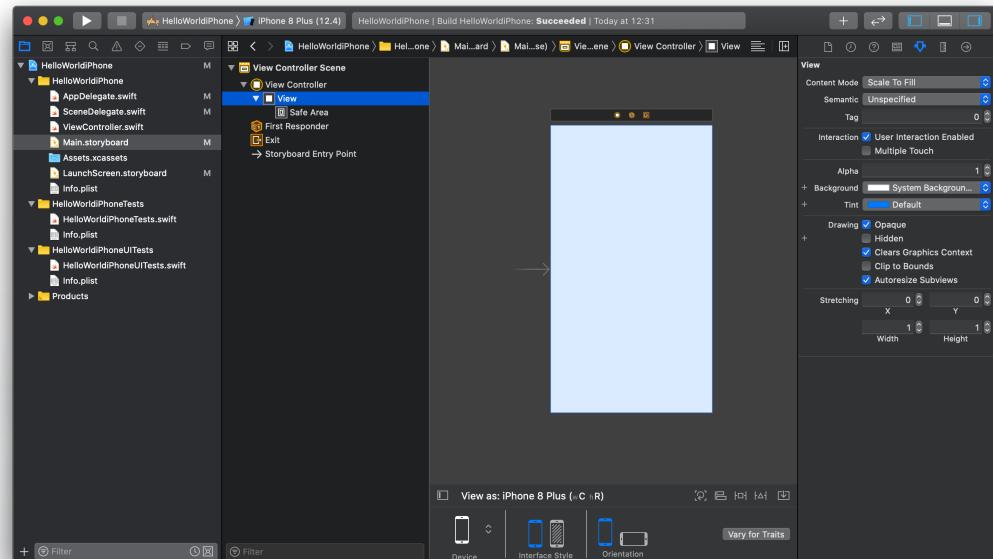
LaunchScreen.storyboard

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



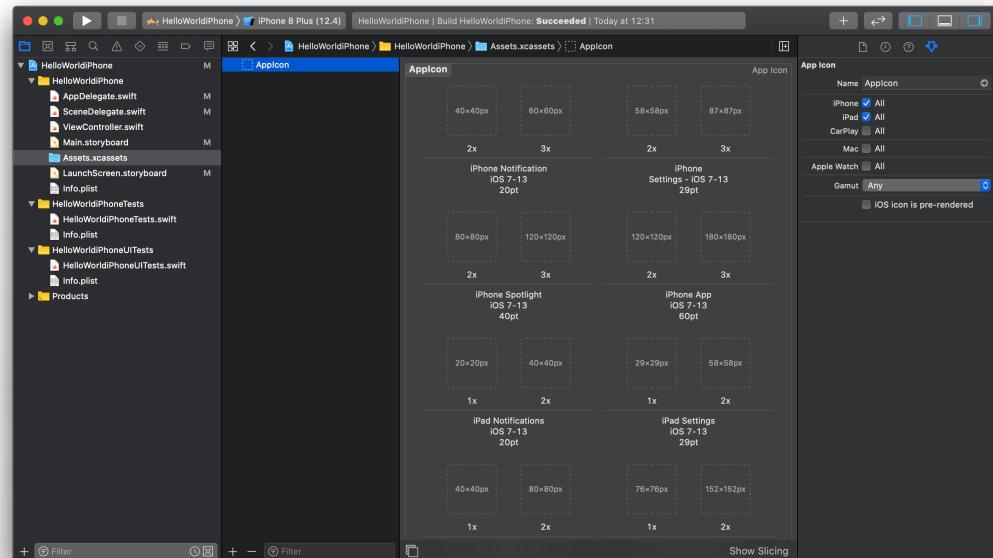
Main.storyboard

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



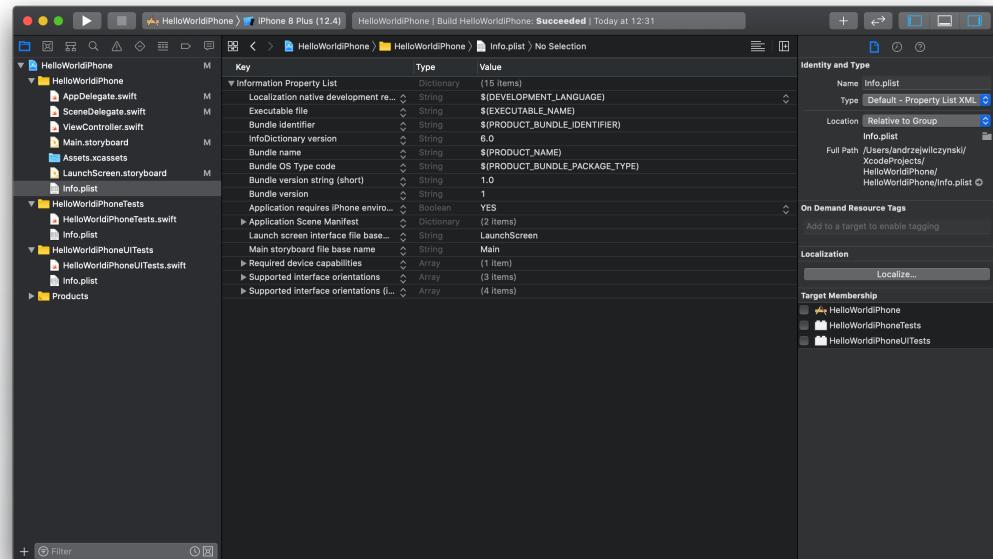
Katalog Assets.xcassets

Miejsce na media,
z których będzie
korzystać aplikacja.

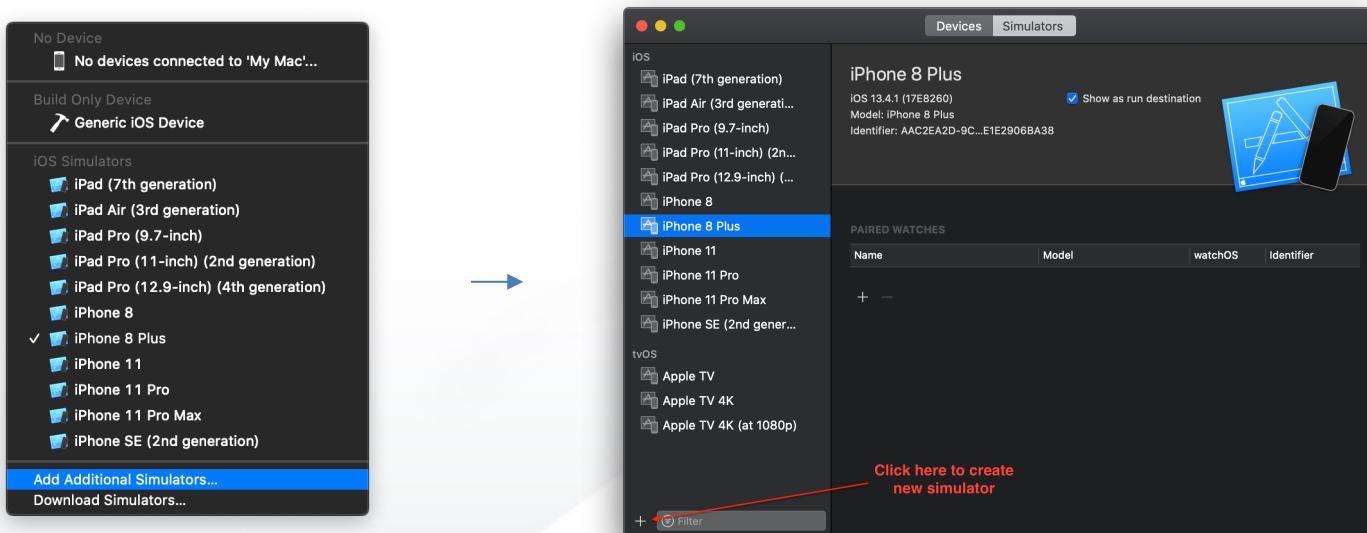


Plik Info.plist

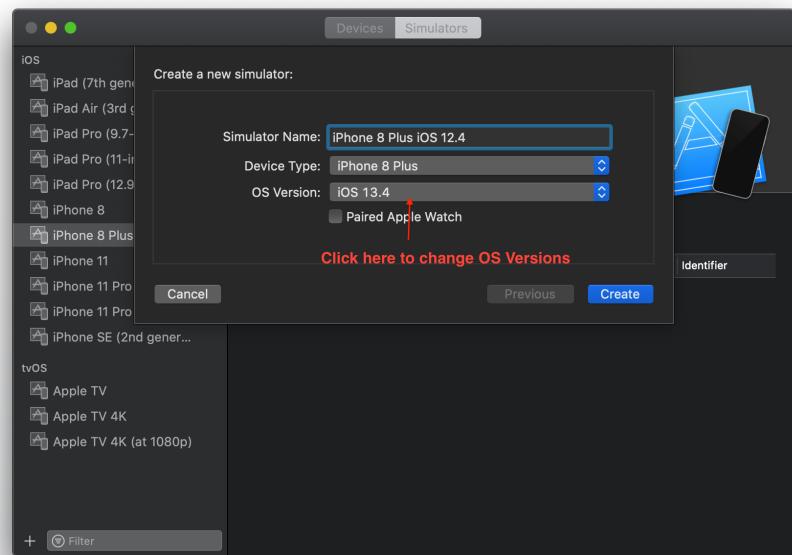
Plik w formacie xml
zawierający wpisy
konfiguracyjne aplikacji.



Konfiguracja symulatora dla iOS 12



Konfiguracja symulatora dla iOS 12

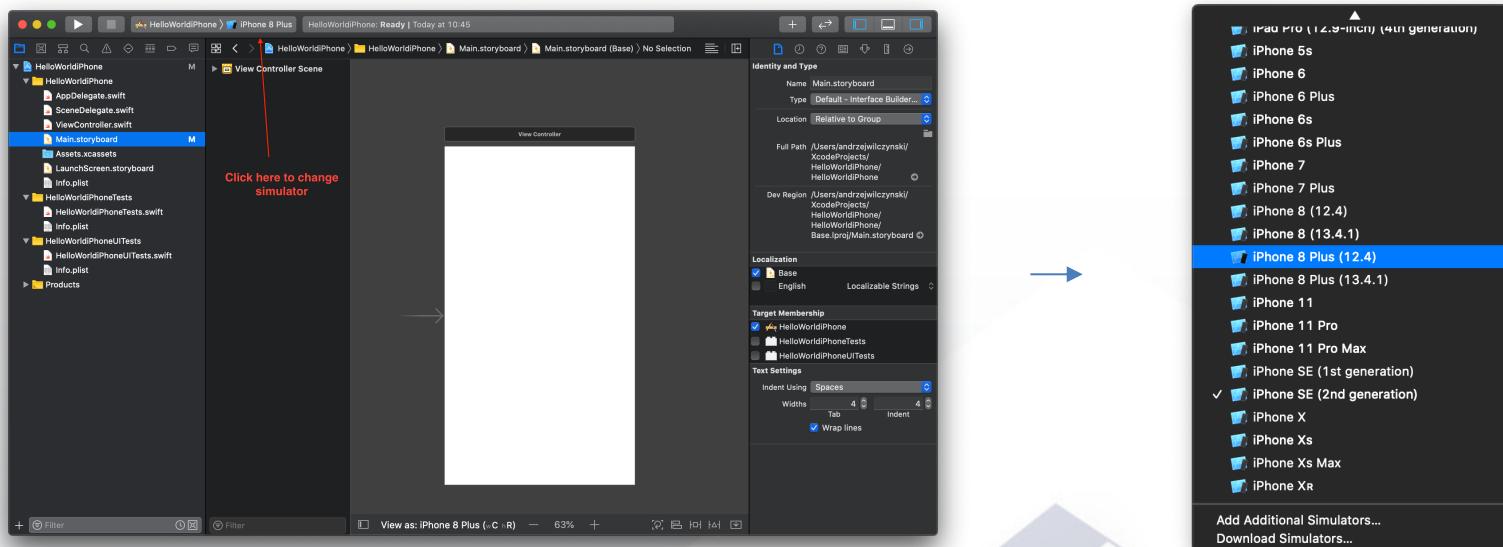


The screenshot shows the 'Components' tab with the 'Simulators' section selected. It lists several iOS simulators with their sizes: iOS 13.3 Simulator (3,27 GB), iOS 13.2 Simulator (3,27 GB), iOS 13.1 Simulator (3,24 GB), iOS 13.0 Simulator (3,26 GB), iOS 12.4 Simulator (2,57 GB), iOS 12.2 Simulator (2,54 GB), iOS 12.1 Simulator (2,49 GB), iOS 12.0 Simulator (2,4 GB), iOS 11.4 Simulator (2,14 GB), iOS 11.3 Simulator (2,14 GB), iOS 11.2 Simulator (2,11 GB), iOS 11.1 Simulator (2,1 GB), and iOS 11.0 Simulator (2,09 GB). A red arrow points from the text 'Download appropriate iOS version' to the 'iOS 12.4 Simulator' entry. Above the simulator list, a message says '✓ iOS 13.4' and 'Download more simulator runtimes...'.

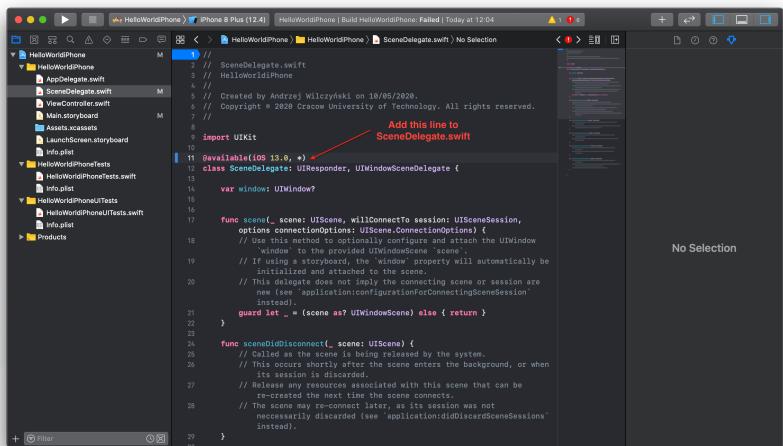
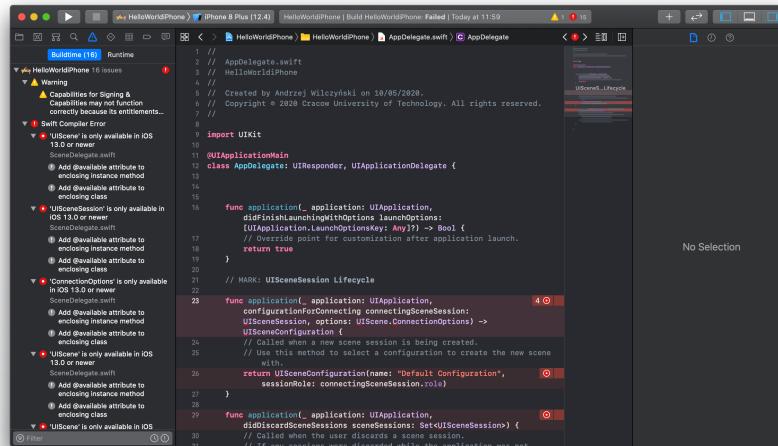
Uruchamianie symulatora



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



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



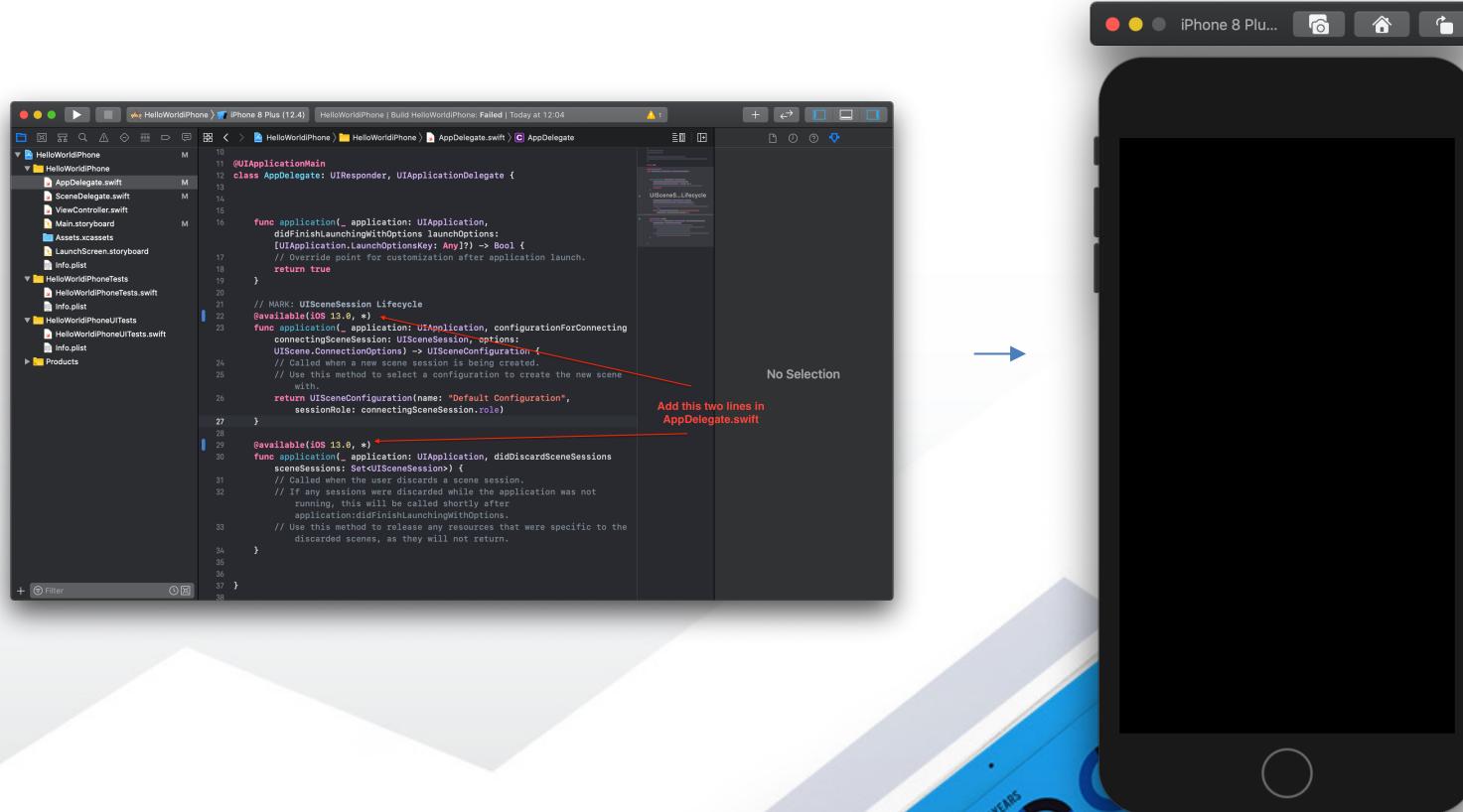
The image shows two screenshots of the Xcode IDE. The left screenshot displays a 'Build Issues' list for an 'iPhone 8 Plus (12.4)' target, showing numerous errors related to Swift Compiler Errors and 'UIScene' being only available in iOS 13.0 or newer. The right screenshot shows the same project structure, but the 'SceneDelegate.swift' file has been modified. A red arrow points from the error in the first screenshot to the code change in the second. The code change adds the line 'Add this line to SceneDelegate.swift' above the class definition:

```
@available(iOS 13.0, *)
```

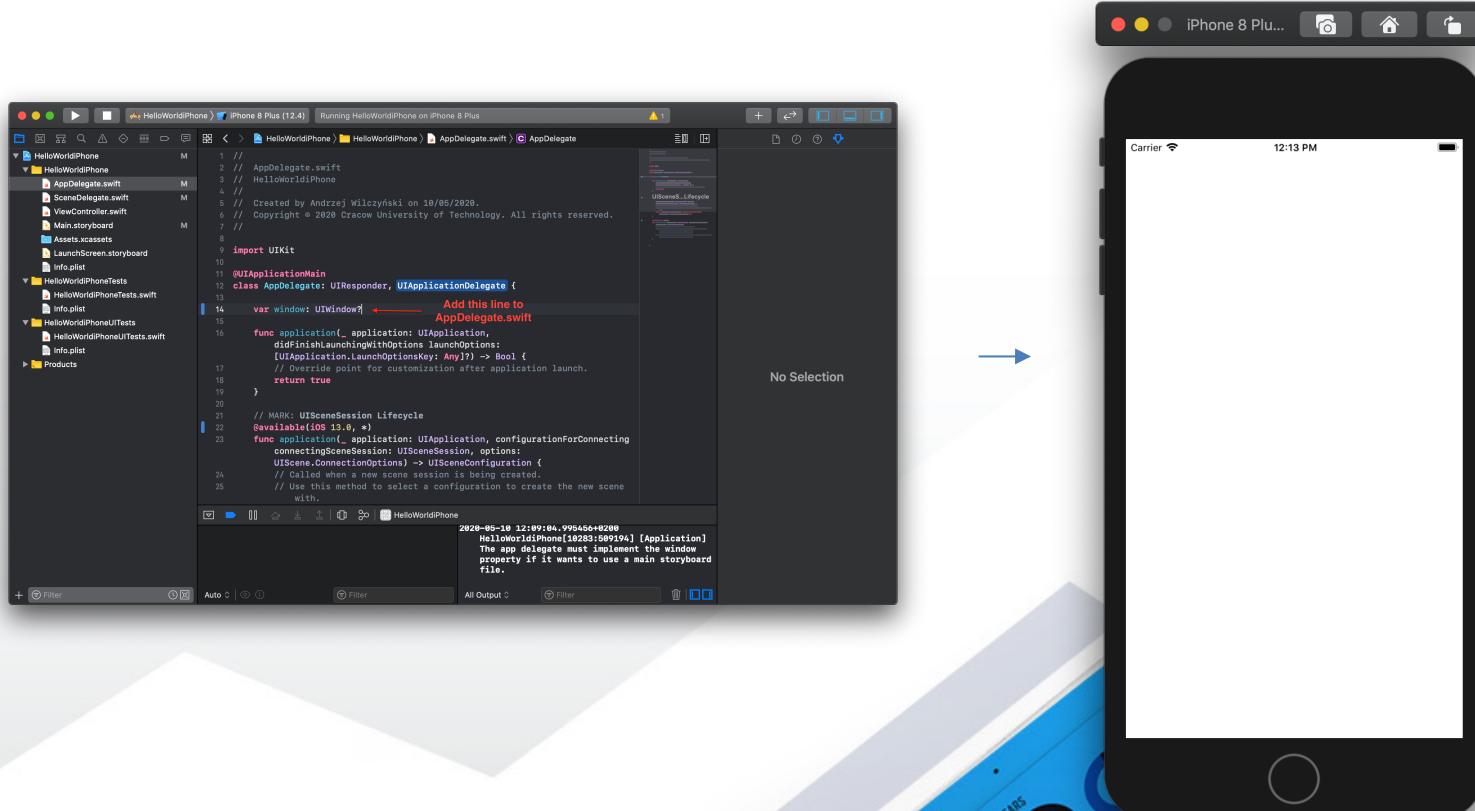
The rest of the code in the 'SceneDelegate.swift' file is as follows:

```
class SceneDelegate: UIResponder, UIWindowSceneDelegate {  
    var window: UIWindow?  
  
    func scene(_ scene: UIScene, willConnectTo session: UISceneSession, options: UIScene.ConnectionOptions) {  
        // Use this method to optionally configure and attach the UIWindow  
        // instance to the scene. If connecting to a storyboard,  
        // window will automatically be initialized and attached to the scene.  
        // This delegate does not imply the connecting scene or session are  
        // new (see 'application:configurationForConnectingSceneSession'  
        // instead).  
        guard let _ = (scene as? UIWindowScene) else { return }  
  
    func sceneDidDisconnect(_ scene: UIScene) {  
        // Called on the scene being released by the system.  
        // This may occur due to memory constraints or if the user quits the app.  
        // Release any resources associated with this scene that can be  
        // re-created the next time the scene connects.  
        // This delegate does not imply the connected scene or session  
        // will be re-created later, as its session was not  
        // necessarily discarded (see 'application:didDiscardSceneSessions'  
        // instead).  
    }  
}
```

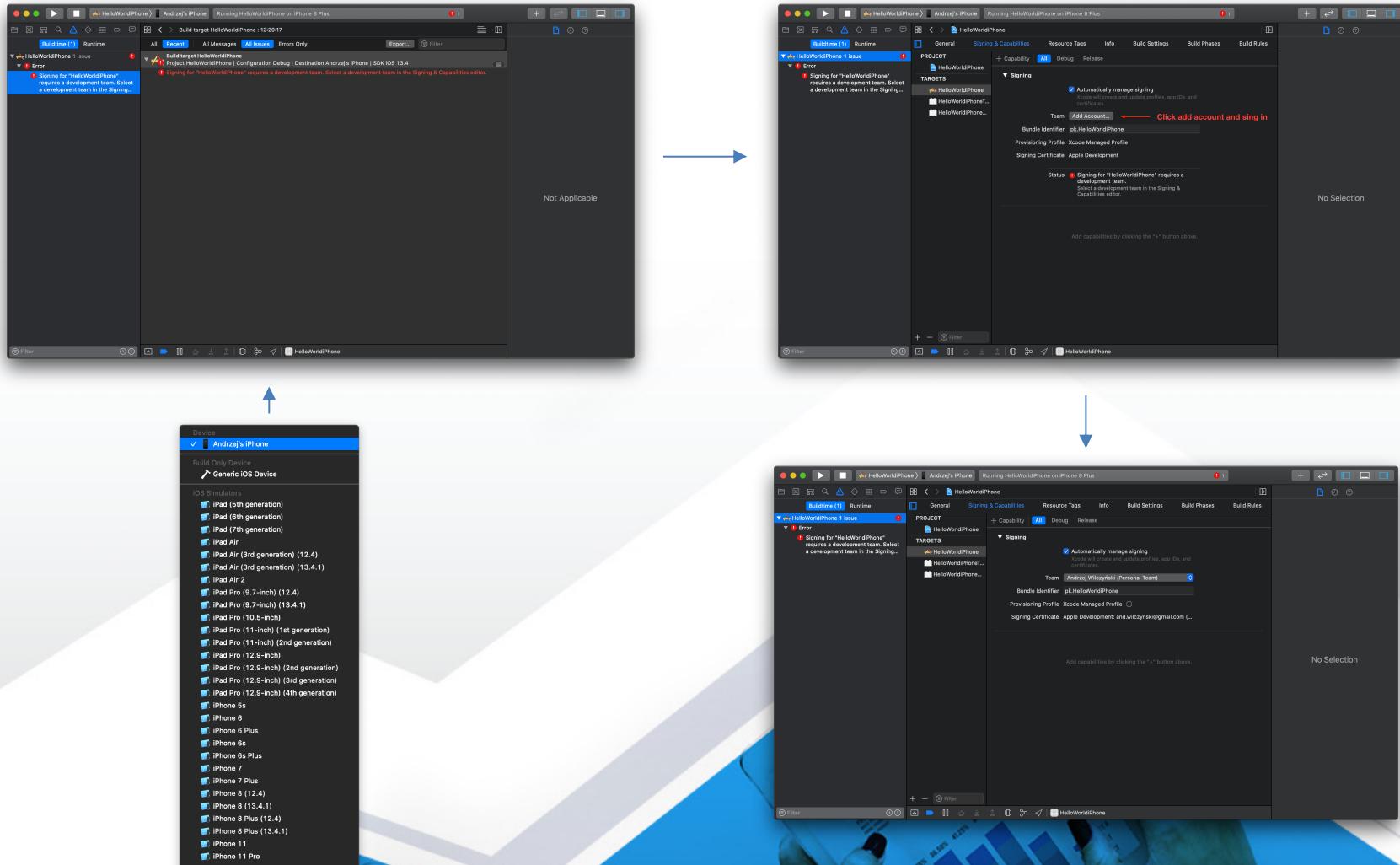
Uruchamianie aplikacji stworzonej w Xcode 11 na iOS 12 (3)



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*.

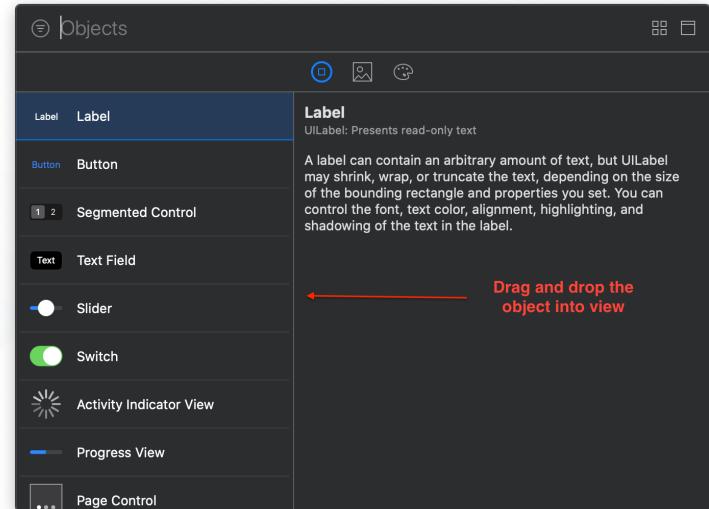
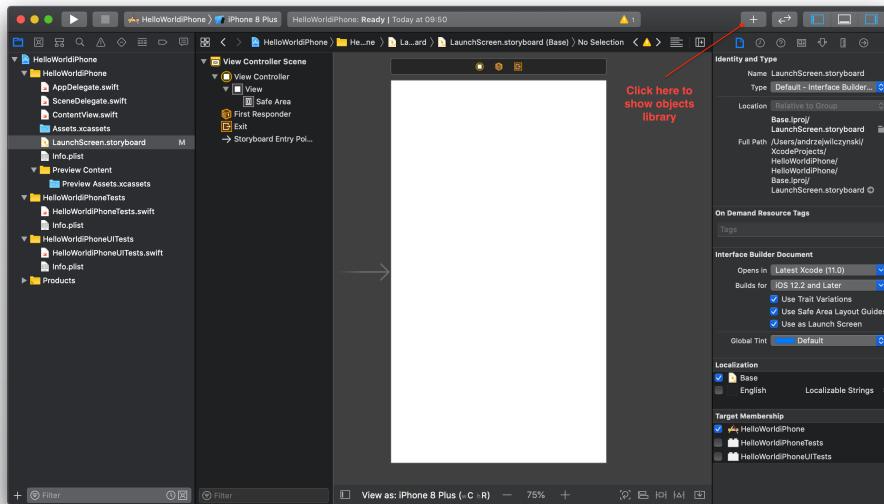


Interface Builder (1)

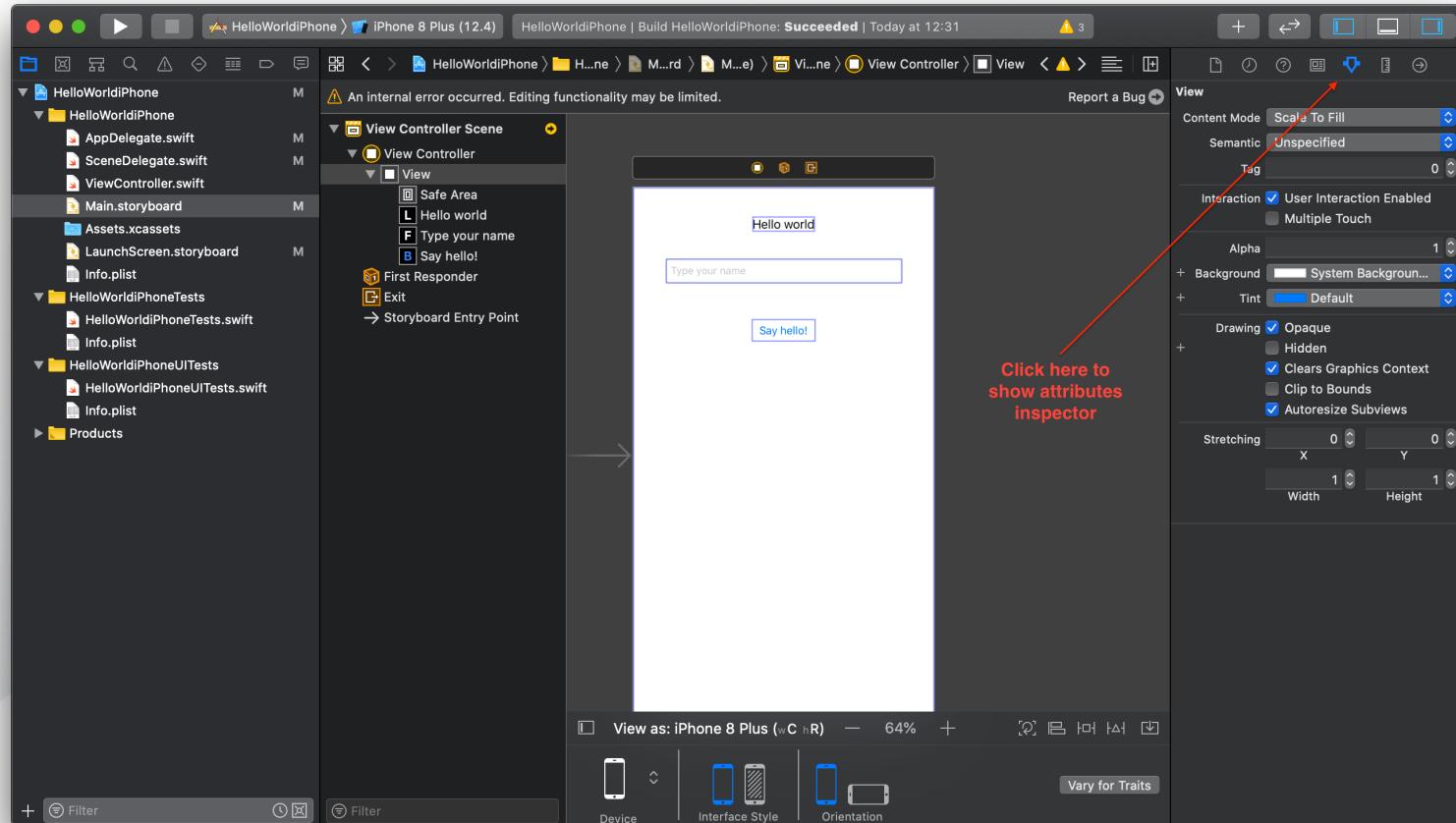
Moduł przeznaczony do umieszczania elementów interfejsu użytkownika w projekcie. W module *Interface Builder* możesz zobaczyć dużą strzałkę skierowaną do widoku. Ta strzałka wskazuje punkt początkowy sceny i prowadzi do widoku, który zostanie wczytany jako pierwszy po uruchomieniu aplikacji.



Interface Builder (2)

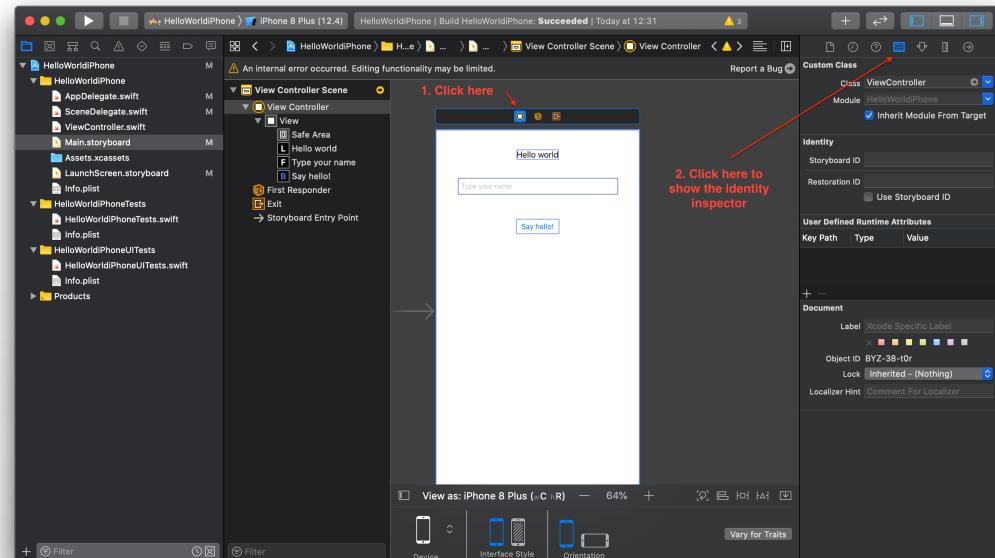


Interface Builder (3)



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



UI View

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`.



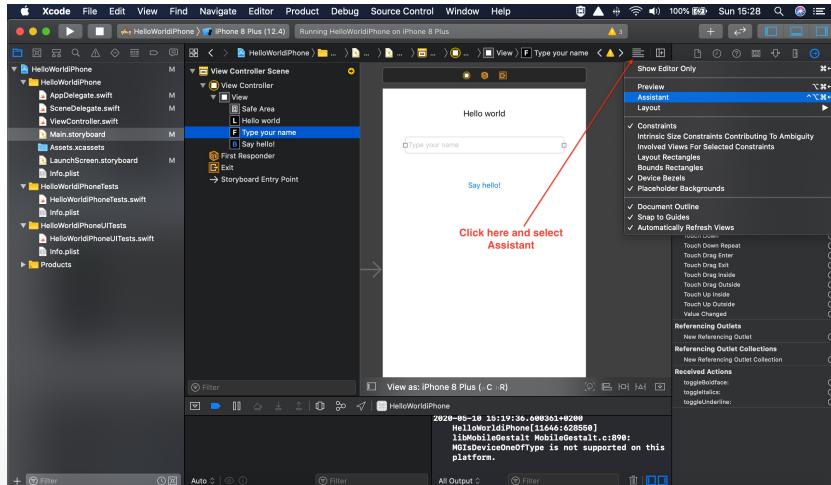
Outlety i akcje

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

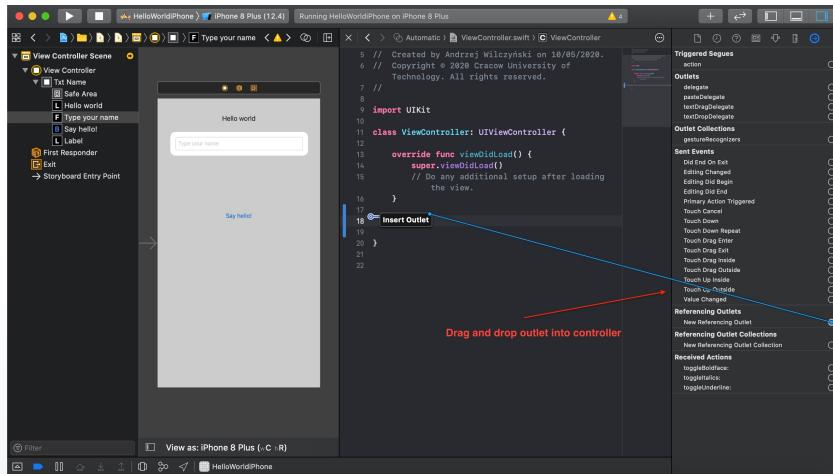


A screenshot of the Xcode interface showing the code editor for ViewController.swift. The code defines a UIViewController subclass named ViewController. It includes imports for UIKit and Foundation, and an override for viewDidLoad(). The right sidebar displays the Triggered Segues, Outlets Collections, and Sent Events for the storyboard elements. The bottom status bar shows "View as: iPhone 8 Plus (-C ||R)".

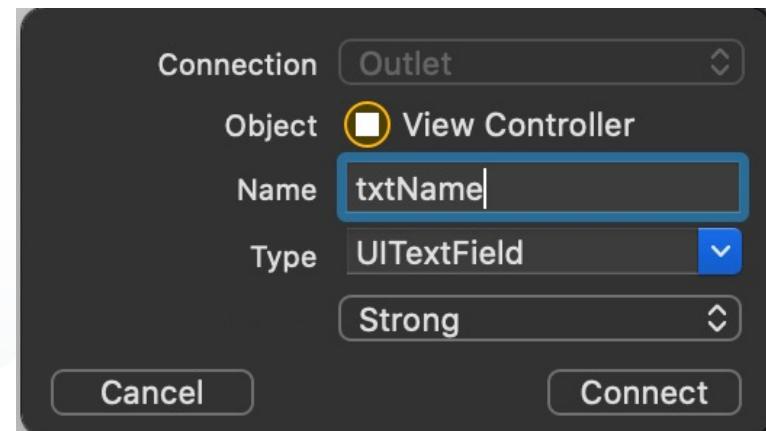
```
1 // ViewController.swift
2 // HelloWorldiPhone
3 //
4 //
5 // Created by Andrzej Wilczyński on 18/05/2020.
6 // Copyright © 2020 Krakow University of Technology. All rights reserved.
7 //
8 import UIKit
9
10 class ViewController: UIViewController {
11     override func viewDidLoad() {
12         super.viewDidLoad()
13         // Do any additional setup after loading the view.
14     }
15 }
16
17
18
19 }
20
21
```



Dodawanie outletu



```
5 // Created by Andrzej Wilczyński on 10/05/2020.  
6 // Copyright © 2020 Krakow University of  
Technology. All rights reserved.  
7 //  
8 import UIKit  
9  
10 class ViewController: UIViewController {  
11     override func viewDidLoad() {  
12         super.viewDidLoad()  
13         // Do any additional setup after loading  
the view.  
14     }  
15     @IBOutlet weak var txtName: UITextField!  
16     @IBOutlet weak var labelHello: UILabel!  
17     @IBAction func sayHello(_ sender: Any) {  
18         labelHello.text = "Hello " + txtName.text  
19     }  
20 }  
21  
22
```

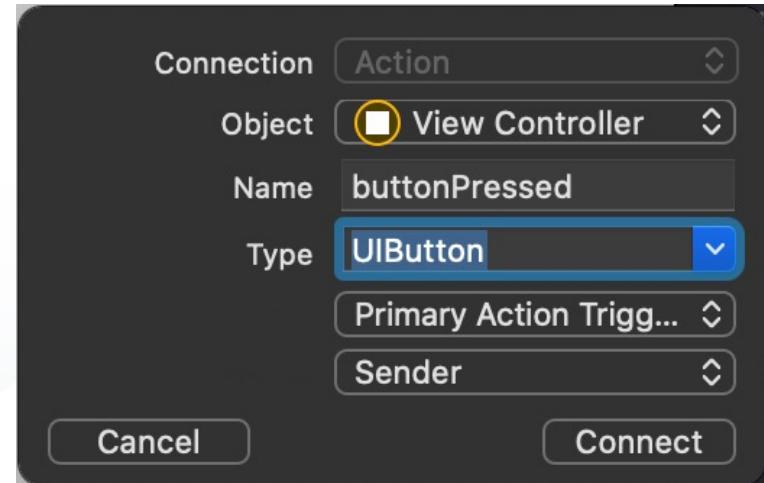


Dodawanie akcji

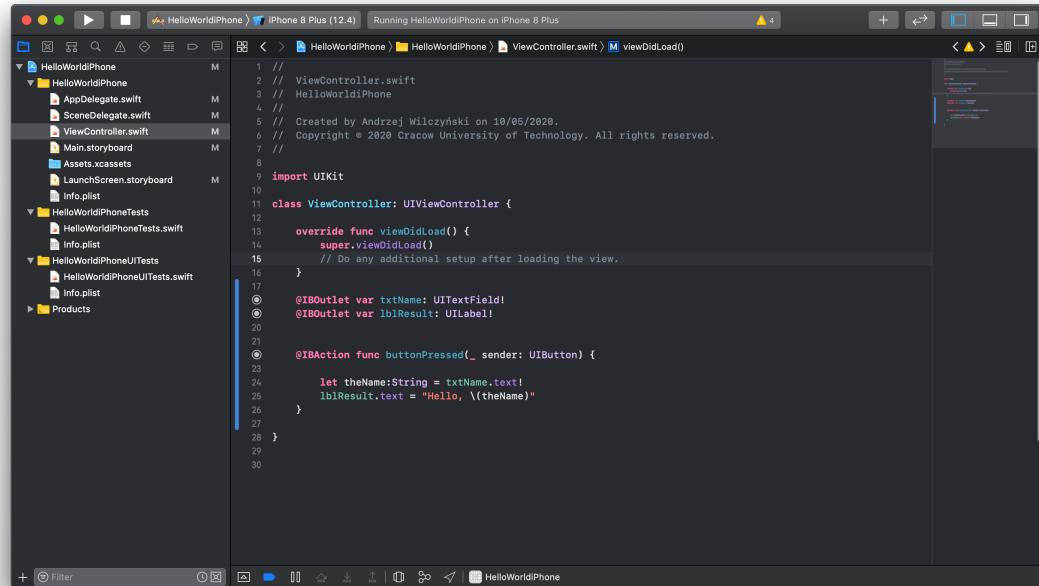
The screenshot shows the Xcode interface. On the left is the storyboard, featuring a single view controller with a text field labeled "Type your name" and a button labeled "Say hello!". In the center is the Swift code for the View Controller:

```
1 // ViewController.swift
2 // ViewController.swift - Created by Andrzej Wilczyński on 10/05/2020.
3 // Copyright © 2020 Cracow University of Technology. All rights reserved.
4
5 import UIKit
6
7 class ViewController: UIViewController {
8
9     override func viewDidLoad() {
10         super.viewDidLoad()
11         // Do any additional setup after loading the view.
12     }
13
14     @IBAction func sayHello(_ sender: UIButton) {
15         let alertController = UIAlertController(title: "Hello world", message: "Type your name", preferredStyle: .alert)
16         alertController.addTextField { (textField) in
17             textField.placeholder = "Type your name"
18         }
19         alertController.addAction(UIAlertAction(title: "Say hello!", style: .default, handler: { _ in
20             let name = alertController.textFields?.first?.text ?? "World"
21             self.showAlert(name: name)
22         }))
23     }
24
25     func showAlert(name: String) {
26         let alertController = UIAlertController(title: "Hello \(name)", message: "Welcome to Swift", preferredStyle: .alert)
27         alertController.addAction(UIAlertAction(title: "OK", style: .default, handler: nil))
28         self.present(alertController, animated: true, completion: nil)
29     }
30
31 }
```

A red arrow points from the storyboard's "Say hello!" button to the "Insert Action" button in the bottom right corner of the code editor.

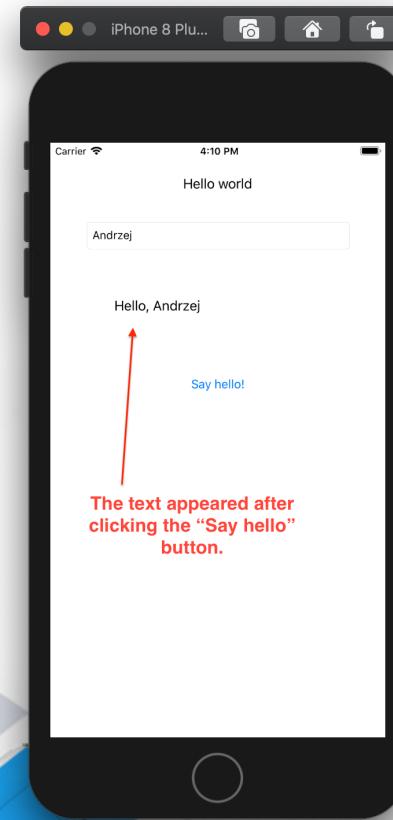


Przykład



A screenshot of the Xcode IDE showing the ViewController.swift file. The code implements a simple application that takes a name from a text field and displays a greeting message. The code includes imports for UIKit, defines a ViewController class, overrides viewDidLoad, and handles a button press to update a label with the user's name.

```
1 // Viewcontroller.swift
2 // HelloWorldiPhone
3 // Created by Andrzej Wilczyński on 10/05/2020.
4 // Copyright © 2020 Cracow University of Technology. All rights reserved.
5
6 import UIKit
7
8 class ViewController: UIViewController {
9
10    override func viewDidLoad() {
11        super.viewDidLoad()
12        // Do any additional setup after loading the view.
13    }
14
15    @IBOutlet var txtName: UITextField!
16    @IBOutlet var lblResult: UILabel!
17
18    @IBAction func buttonPressed(_ sender: UIButton) {
19
20        let theName:String = txtName.text!
21        lblResult.text = "Hello, \(theName)"
22    }
23
24}
25
26
27
28}
29
30}
```



Dziękuję! ☺



Bibliografia

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

