
View Controllers

MK Map View



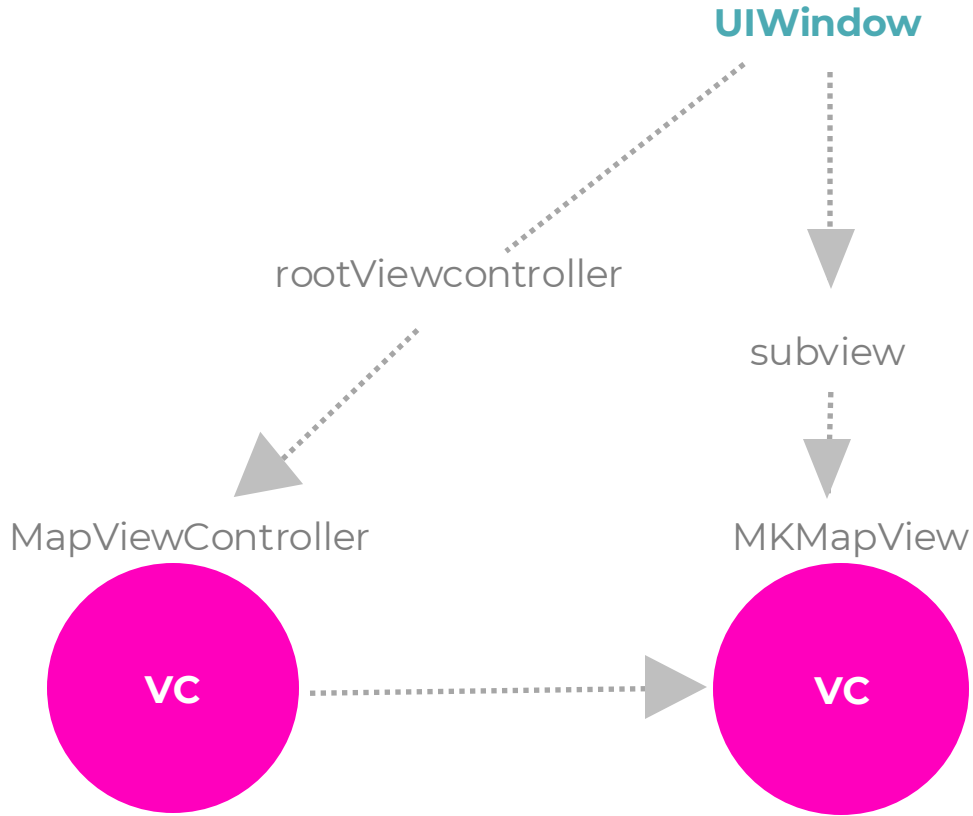
View Controller

Manages view hierarchy

Creates view objects

Handles events associated with view objects

View Controllers



- 2 ways to create ViewController:
1. Using storyboard in IB
 2. Programmatically overriding method `loadView()`

Each storyboard has only one Initial view controller. It is an entry point to the storyboard. This view controller shows first.

Add the MapKit framework

Each app has one main interface (main.storyboard)

When the app launches, the initial view controller gets set as the rootViewController of the window.

The main interface is set in the project settings >General > Deployment >Main Interface
Main corresponds to Main. Storyboard.

The screenshot shows the Xcode interface for a project named "WorldTrotter" on an iPhone 7. The top status bar indicates "Finished running WorldTrotter on iPhone 7" with 5 warnings. The left sidebar shows the "Issues" panel with 5 issues related to storyboard configuration. The right sidebar shows the "General" settings tab, with the "Deployment Info" section expanded, showing the following settings:

- Deployment Target: 11.2
- Devices: Universal
- Main Interface: Main
- Device Orientation: Portrait (checked), Landscape Left (checked), Upside Down (unchecked)

Setting Initial View Controller

Open Main.storyboard.

Drag View Controller onto canvas. Select the View of the ViewController – Delete.

Drag Map Kit View from object library (search) onto controller to set map view.

The screenshot displays the Xcode development environment. On the left, the 'View Controller Scene' is visible, containing a 'View Controller' object. The main canvas shows a storyboard with a 'Map Kit View' (MKMapView) placed on a 'View Controller' object. The right-hand panel shows the 'View Controller' properties, including 'Simulated Metrics' (Size, Top Bar, Bottom Bar) and 'View Controller' settings. The 'View Controller' section has 'Is Initial View Controller' checked, and 'Layout' options like 'Adjust Scroll View Insets' and 'Resize View From NIB' are also checked. The 'Extend Edges' section is checked for 'Under Top Bars' and 'Under Bottom Bars'. The 'Transition Style' is set to 'Cover Vertical' and 'Presentation' is 'Full Screen'. The 'Content Size' is set to 'Use Preferred Explicit Size' with a width of 375 and height of 667. The 'Key Commands' section is empty. At the bottom, the status bar shows 'View as: iPhone 8 (wC hR)' at 77% zoom. A 'Map Kit View' tooltip is visible at the bottom right, stating: 'Map Kit View - Displays maps and provides an embeddable interface to navigate map content.'

Setting Initial View Controller

Select View Controller and open Attributes inspector.

Check Is Initial View Controller

Notice the change of direction of the gray arrow

Now it points to View Controller now, not Conversion View Controller.

Test and run the app.

Simulated Metrics

- Size: Inferred
- Top Bar: Inferred
- Bottom Bar: Inferred

View Controller

Title:

Is Initial View Controller

Layout

- Adjust Scroll View Insets
- Hide Bottom Bar on Push
- Resize View From NIB
- Use Full Screen (Deprecated)

Extend Edges

- Under Top Bars
- Under Bottom Bars
- Under Opaque Bars

Transition Style: Cover Vertical

Presentation: Full Screen

- Defines Context
- Provides Context

Content Size: Use Preferred Explicit Size

Width: Height:

Key Commands

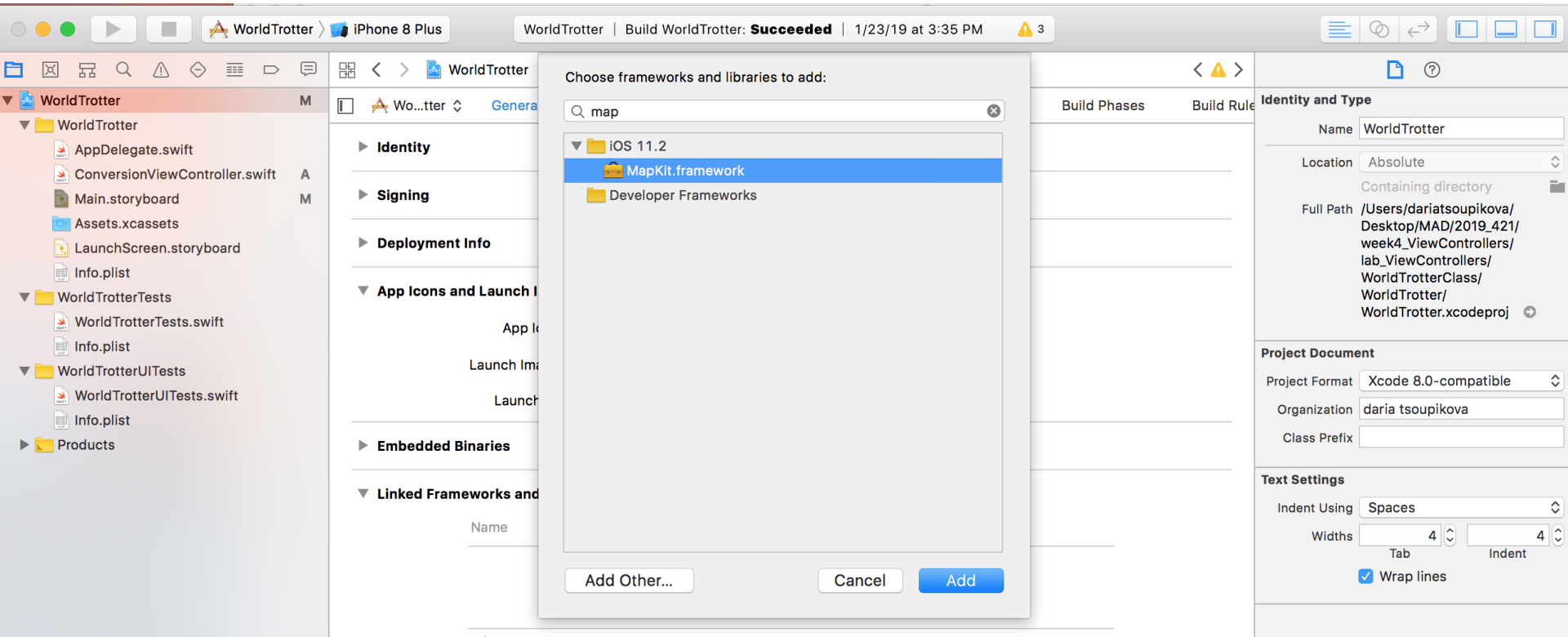


Add the MapKit framework

Import MapKit framework and link it to the view to load map view.

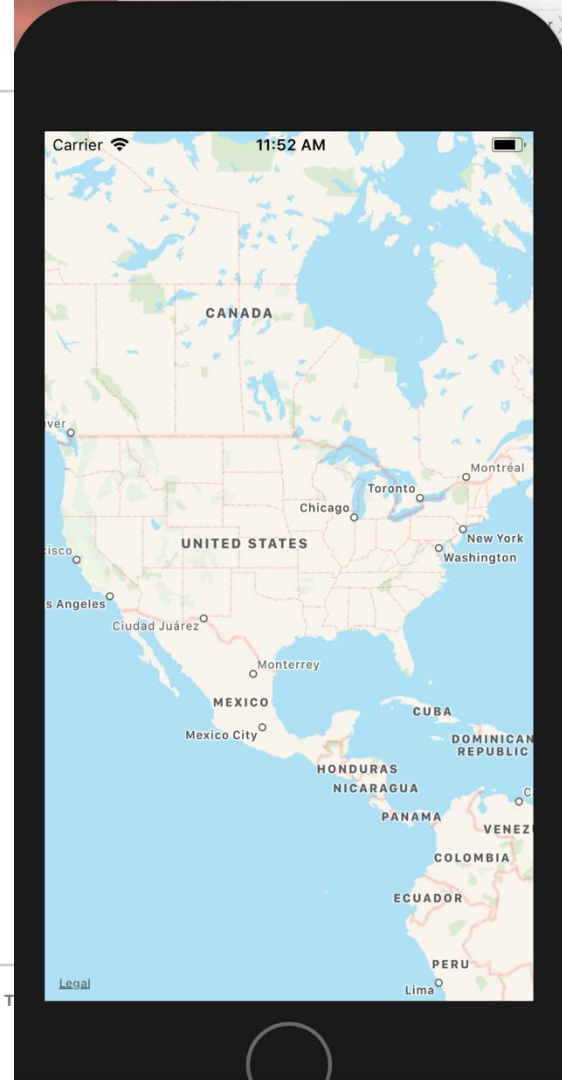
WorldTrotter project > Project Settings > General > Linked Frameworks and Libraries

Check + sign and search for MapKit framework. Click Add. Build and run to test map view.



Add the MapKit framework

Build and run to test map view.



Add the MapKit framework

Each app has one main interface (main.storyboard)

When the app launches, the initial view controller gets set as the rootViewController of the window.

The main interface is set in the project settings >General > Deployment >Main Interface
Main corresponds to Main. Storyboard.

The screenshot shows the Xcode interface with the project 'WorldTrotter' selected for the 'iPhone 7' target. The top status bar indicates 'Finished running WorldTrotter on iPhone 7' with 5 warnings. The left sidebar shows the 'Issues' pane with 5 issues: 'Unsupported Configuration' (2), 'Auto Layout Localization' (1), and 'Fixed width constraints may cause clipping' (1). The main area shows the 'General' settings tab for the target, with sections for Identity, Signing, and Deployment Info. The Deployment Info section shows: Deployment Target: 11.2, Devices: Universal, Main Interface: Main, and Device Orientation: Portrait and Landscape Left selected.

WorldTrotter > iPhone 7 Finished running WorldTrotter on iPhone 7 5

WorldTrotter

Buildtime (5) Runtime

WorldTrotter 5 issues

- Unsupported Configuration
 - Custom gesture recognizers should have a custom class set in the Identity inspector. Main.storyboard
 - "Conversion View Controller" is unreachable because it has no entry points, and no identifier for runtime... Main.storyboard
- Auto Layout Localization
 - Fixed width constraints may cause clipping. Main.storyboard
 - width = 250
- Unsupported Configuration

WorldTrotter

General Capabilities Resource Tags Info Build Settings Build Phases Build Rules

Identity

Signing

Deployment Info

Deployment Target 11.2

Devices Universal

Main Interface Main

Device Orientation Portrait Upside Down Landscape Left

Add Transition

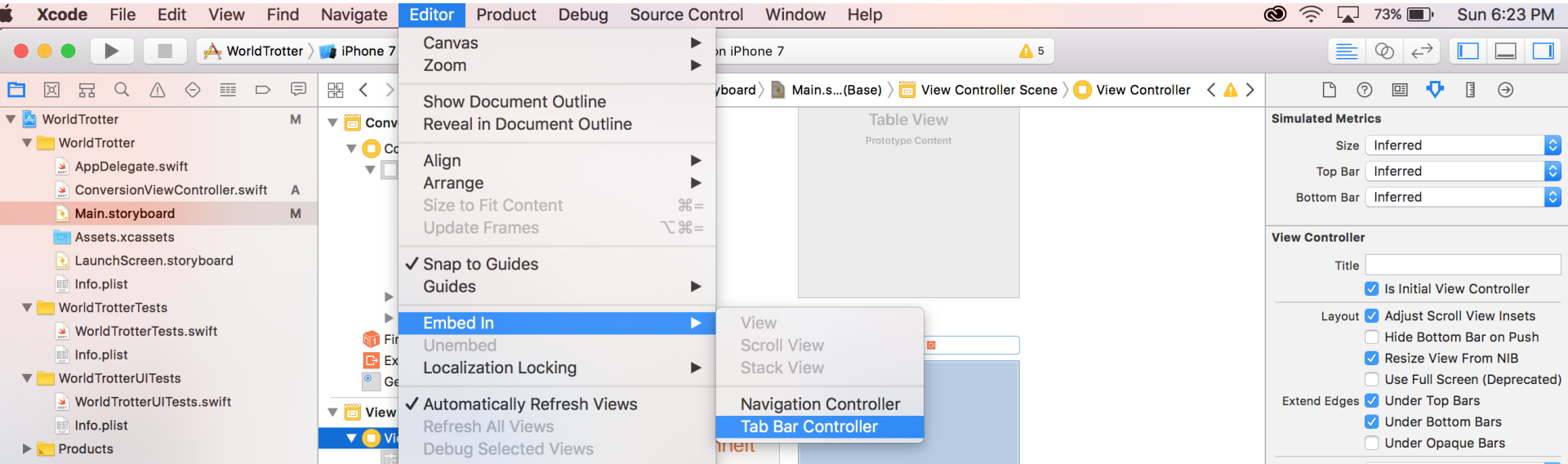
UITabBarController allows swap between the View Controllers.

Keeps array of view controllers

Maintains a menu to select view controllers

Open Main.storyboard> select View Controller

Editor > Embed In > Tab Bar Controller . Relationship arrow will be added to pointing from TabBar Controller to View Controller



Add Transition

Add Conversion ViewController to TabBarController to switch between the screens:

Control-drag from the TabBarController to the ConversionViewController
Choose: Relationship Segue > view controllers. Build, run and test switching views.

The screenshot shows the Xcode IDE with a storyboard for an iPhone 7. The storyboard contains a TabBarController with three tabs: 'Conversion View Controller', 'Table View', and 'Item'. The 'Conversion View Controller' tab is selected, and a segue is being added from the 'TabBarController' to the 'Conversion View Controller'. A context menu is open over the segue, showing the following options:

- Manual Segue
- Show
- Show Detail
- Present Modally
- Present As Popover
- Custom
- Relationship Segue
- view controllers
- Non-Adaptive Manual Segue
- Push (deprecated)
- Modal (deprecated)

The storyboard also shows a 'Table View' with 'Prototype Content' and an 'Item' with a 'MKMapView'. The 'Conversion View Controller' tab contains a 'View' with a 'Safe Area' and a 'Label' with the text 'degrees Fahrenheit is really 100 degrees Celsius'. The 'Table View' contains a 'Table View Controller' with a 'Table View' and a 'Table View Controller' with a 'Table View'.

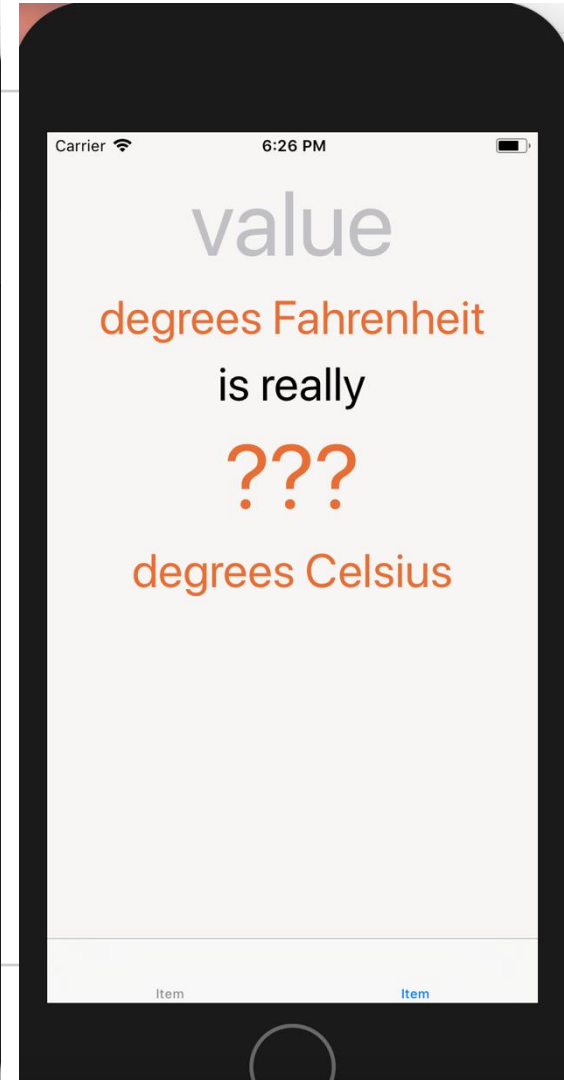
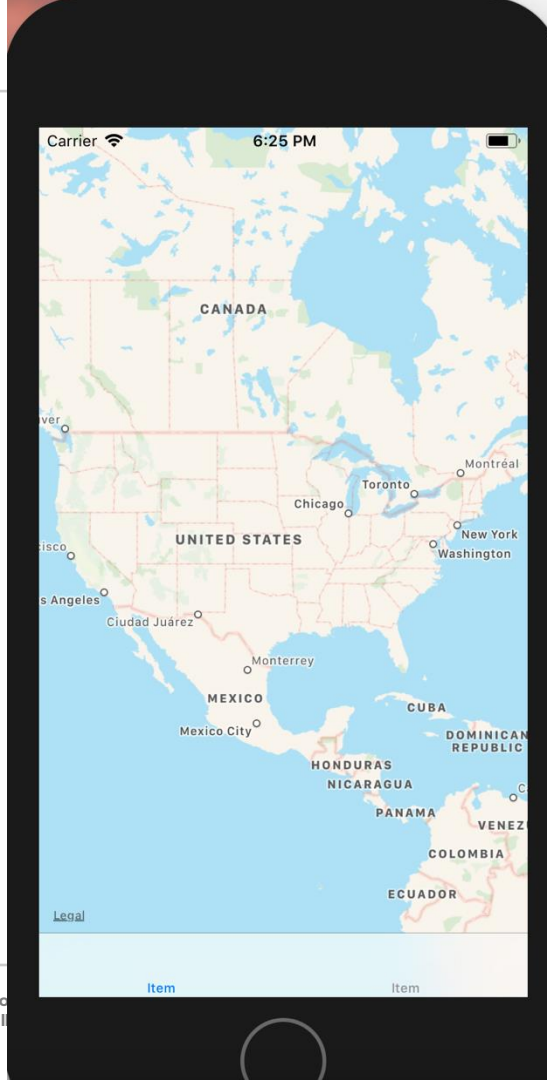
The right-hand pane shows the 'Simulated Metrics' and 'View Controller' settings. The 'View Controller' settings are:

- Title: [Empty]
- Is Initial View Controller
- Layout: Adjust Scroll View Insets, Hide Bottom Bar on Push, Resize View From NIB, Use Full Screen (Deprecated)
- Extend Edges: Under Top Bars, Under Bottom Bars, Under Opaque Bars
- Transition Style: Cover Vertical
- Presentation: Full Screen
- Content Size: Defines Context, Provides Context, Use Preferred Explicit Size
- Width: [0] Height: [0]

The 'Key Commands' section is empty.

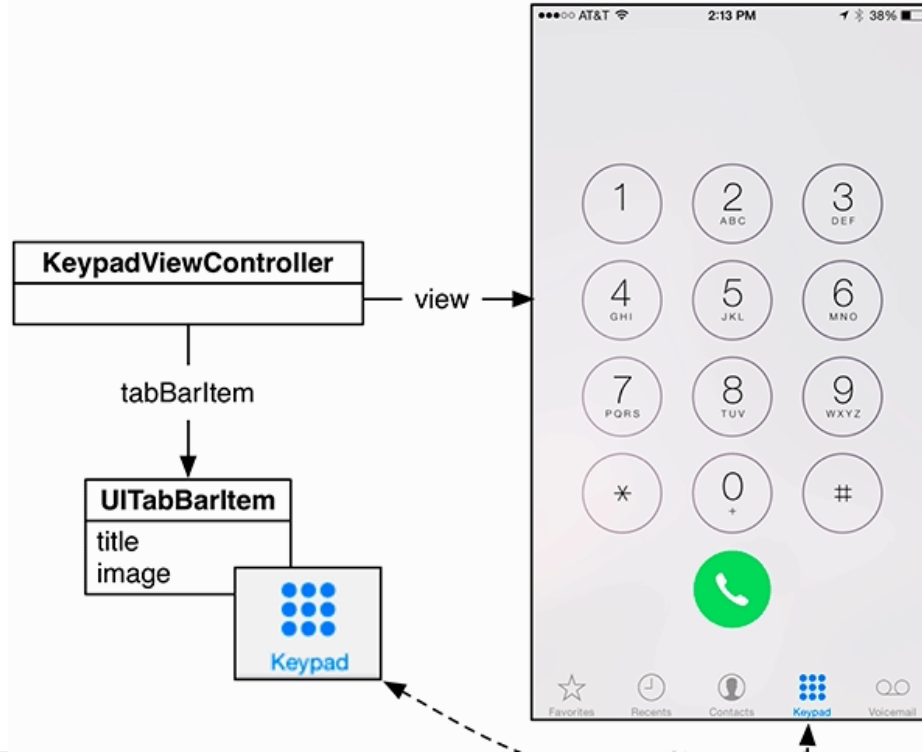
Add Transition

Tab menu bars to switch between the screens:



Modify the menu

Each tab bar in the menu can display a title and an image. (currently untitled “Item”)



Modify the menu

Add image by adding it to assets
Assets are files to be selected at runtime

Open Assets.xcassets

download from images folder:

Under Ch. 5 Xcode Assets

ConvertIcon.png

ConvertIcon@2x.png

ConvertIcon@3x.png

MapIcon.png

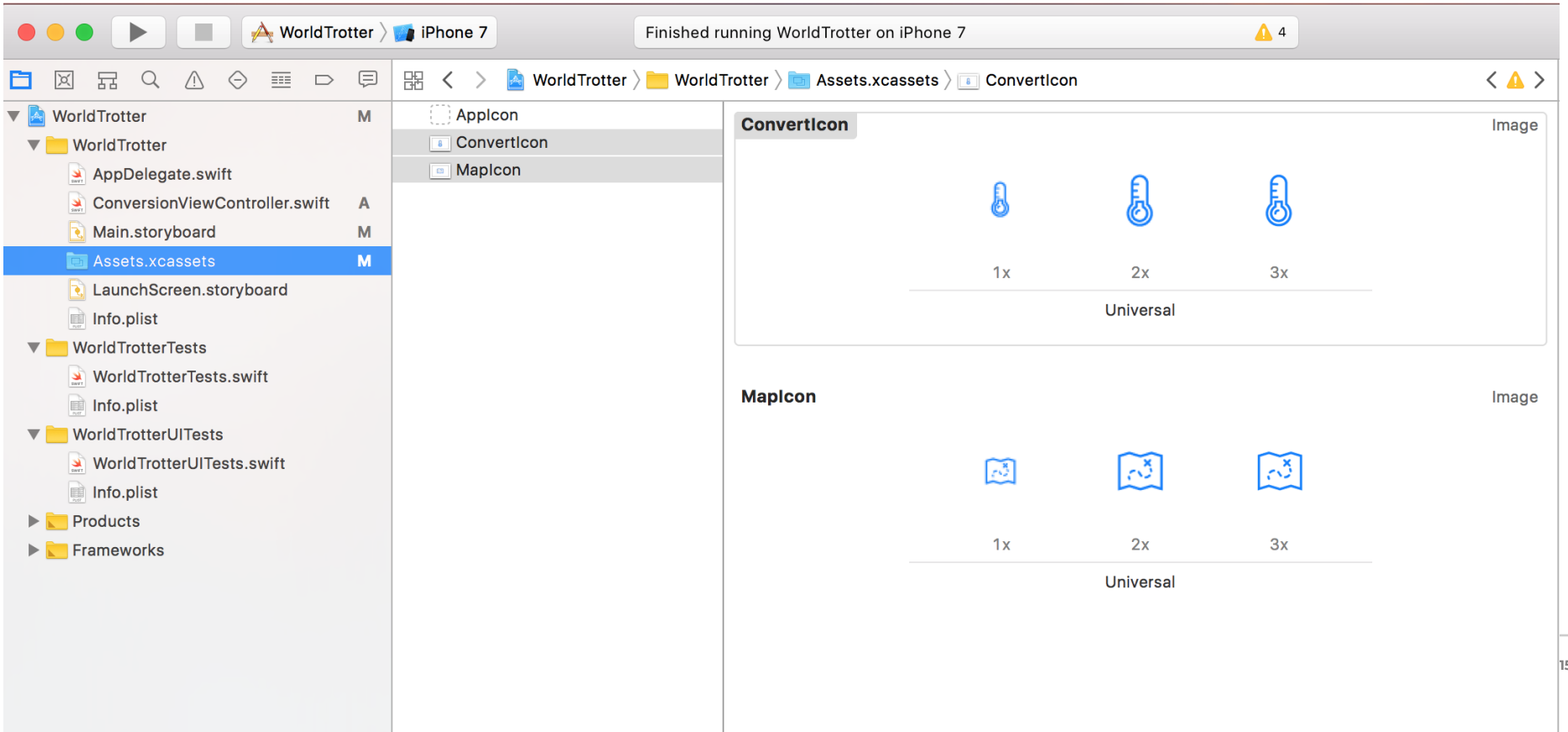
MapIcon@2x.png

MapIcon@3x.png



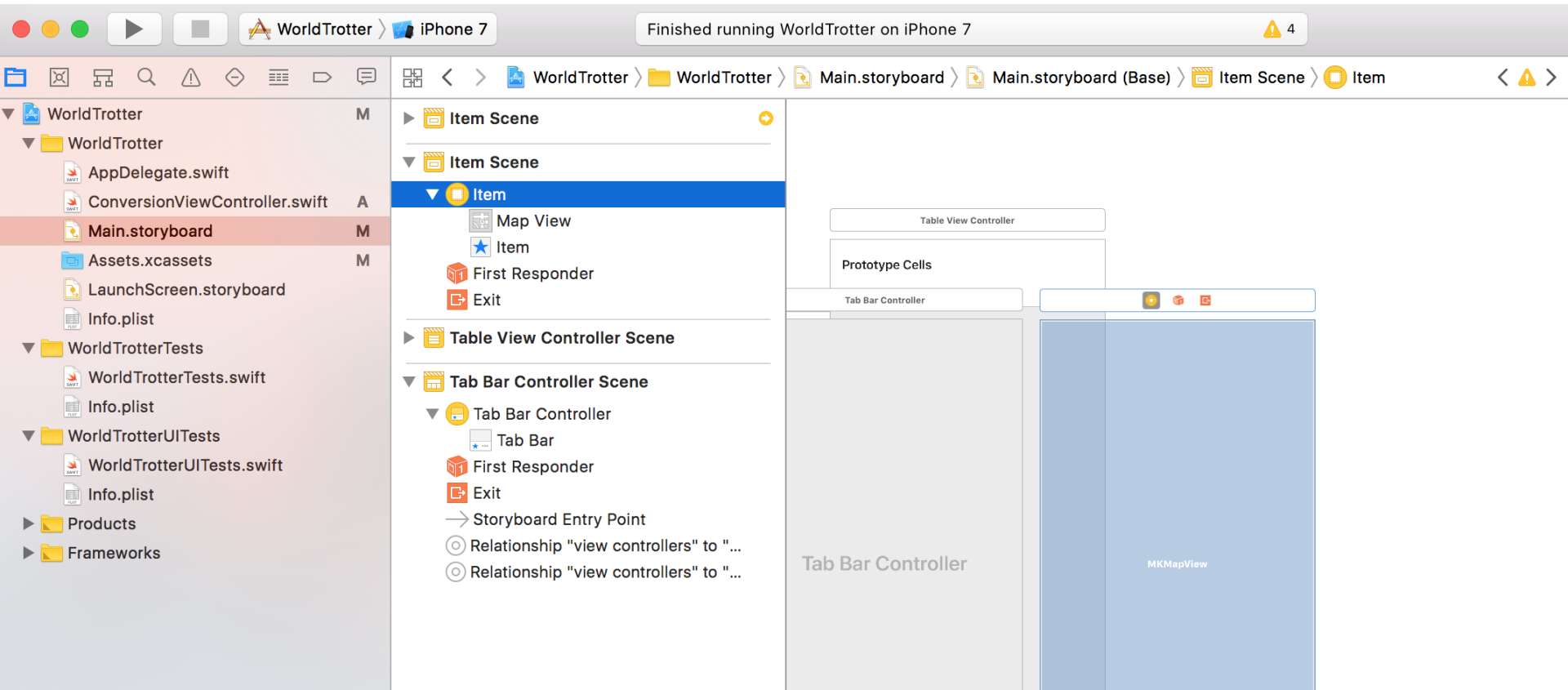
Modify the menu

Drag these images to images set list of the Assets Catalog.



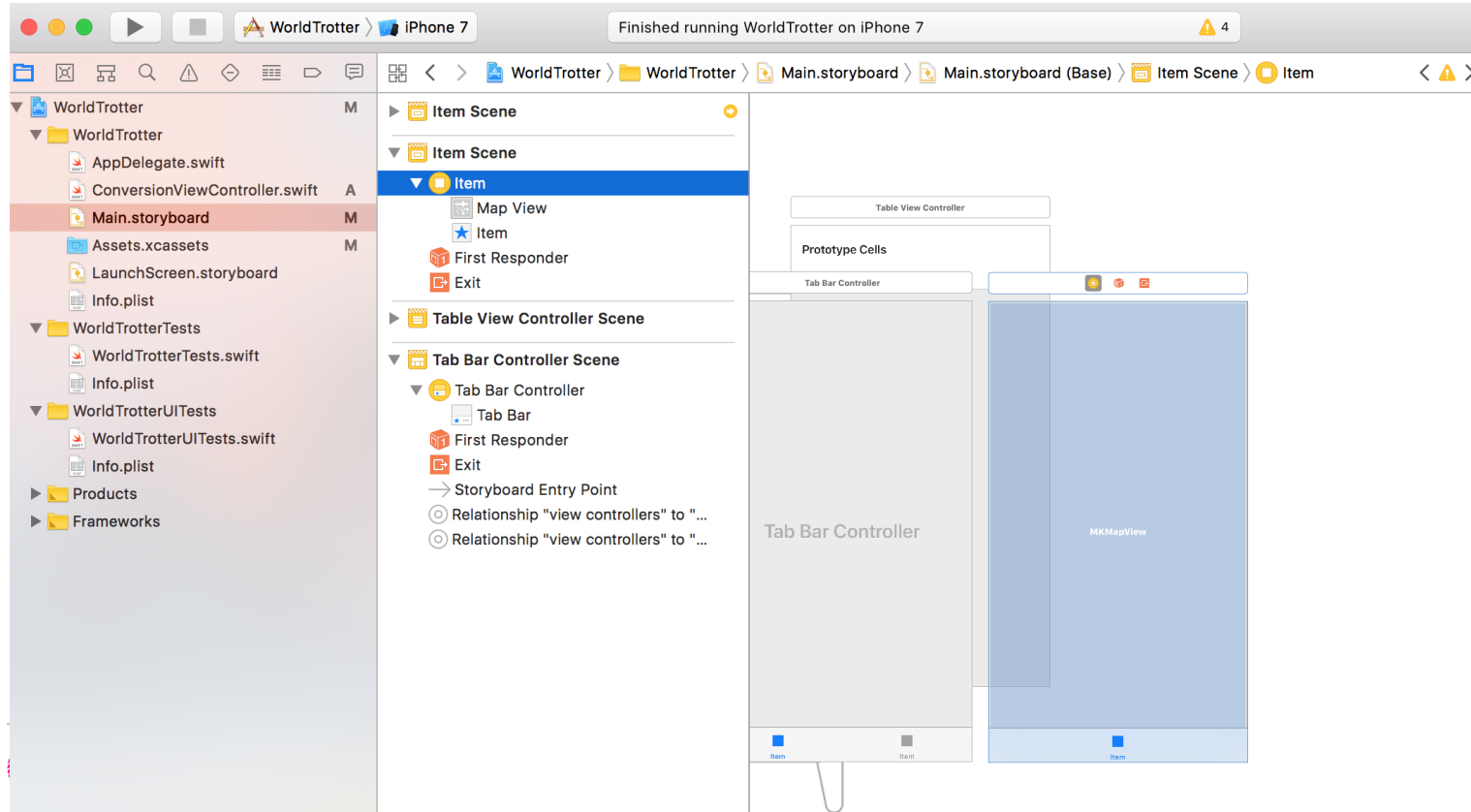
Modify the menu

In Main.storyboard select ViewController – it is named Item now



Modify the menu

Notice the tab bar is added to the bottom of the Interface



Modify the menu

Select this tab bar and open attributes
Change name to Map
Choose MapIcon as Image

Tab Bar Item

Badge Value

Default

System Item Custom

Selected Image Selected Image

Title Position Default Position

Drag and Drop Spring Loaded

Bar Item

Title Map

Image Image

Landscape ConvertIcon

Accessibility MapIcon

Tag 0

Enabled

Tab Bar Item

Badge Value

Default

System Item Custom

Selected Image Selected Image

Title Position Default Position

Drag and Drop Spring Loaded

Bar Item

Title Map

Image Image

Landscape Image (iPhone)

Accessibility Large Content Size Image

Tag 0

Enabled

Modify the menu

Select this tab bar and open attributes

Change name to Map

Choose MapIcon as Image

The screenshot displays the Xcode IDE interface for editing a storyboard. The top status bar shows the app is running on an iPhone 7. The left sidebar contains a project tree with 'WorldTrotter' selected. The middle-left pane shows a hierarchy of storyboard elements: Item Scene, Map Scene, Item, Map View, and Map. The main canvas shows a storyboard with a Tab Bar Controller and an MKMapView. The right-hand Attributes Inspector is open to the 'Tab Bar Item' section, where the 'Image' property is set to 'MapIcon' and the 'Title' is 'Map'. Other properties like 'Badge', 'System Item', and 'Selected Image' are also visible.

WorldTrotter > iPhone 7

Finished running WorldTrotter on iPhone 7

WorldTrotter > WorldTrotter > Main.storyboard > Main.storyboard (Base) > Item Scene > Item > Map

WorldTrotter

- WorldTrotter
 - AppDelegate.swift
 - ConversionViewController.swift
 - Main.storyboard
 - Assets.xcassets
 - LaunchScreen.storyboard
 - Info.plist
- WorldTrotterTests
 - WorldTrotterTests.swift
 - Info.plist
- WorldTrotterUITests
 - WorldTrotterUITests.swift
 - Info.plist
- Products
- Frameworks

Item Scene

- Map Scene
 - Item
 - Map View
 - Map
 - First Responder
 - Exit
- Table View Controller Scene
- Tab Bar Controller Scene
 - Tab Bar Controller
 - Tab Bar
 - First Responder
 - Exit
 - Storyboard Entry Point
 - Relationship "view controllers" to "...
 - Relationship "view controllers" to "...

Table View Controller

Prototype Cells

Tab Bar Controller

Tab Bar Controller

MKMapView

Tab Bar Item

Badge Value

System Item Custom

Selected Image Selected Image

Title Position Default Position

Drag and Drop Spring Loaded

Bar Item

Title Map

Image MapIcon

Landscape Image (iPhone)

Accessibility Large Content Size Image

Tag 0

Enabled

Modify the menu

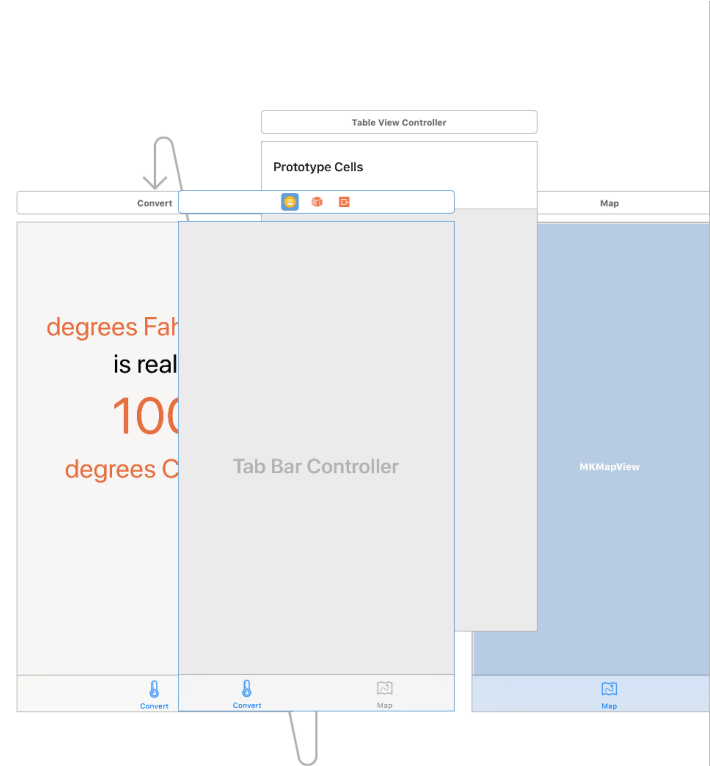
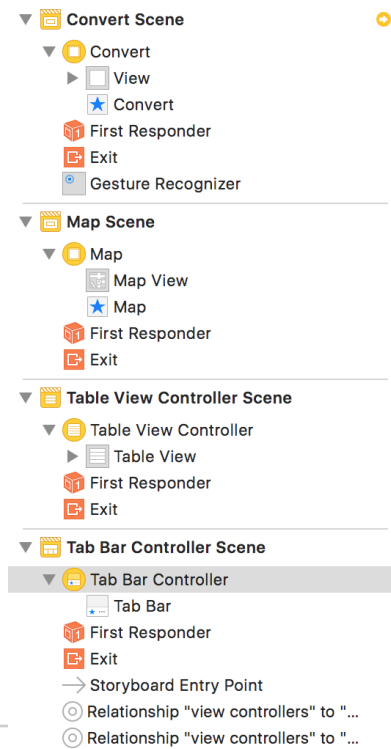
- Select tab bar in ConversionViewController
- Change name to “Convert”
- Choose ConvertIcon as Image

The screenshot displays the Xcode interface for editing a storyboard. On the left, the 'Convert Scene' is selected in the storyboard, showing a hierarchy: Convert Scene > Item > View > Convert. The main canvas shows a storyboard with a 'Table View Controller' and a 'Bar Controller'. The 'Bar Controller' contains a 'Tab Bar' with a 'Convert' tab. The text on the screen reads: 'degrees Fahrenheit is really 100 degrees Celsius'. The 'Attributes Inspector' on the right is open to the 'Tab Bar Item' section, showing the following settings:

- Badge: Value
- System Item: Default
- System Item: Custom
- Selected Image: Selected Image
- Title Position: Default Position
- Drag and Drop: Spring Loaded
- Bar Item:
 - Title: Convert
 - Image: ConvertIcon
 - Landscape: Image (iPhone)
 - Accessibility: Large Content Size Image
 - Tag: 0
 - Enabled

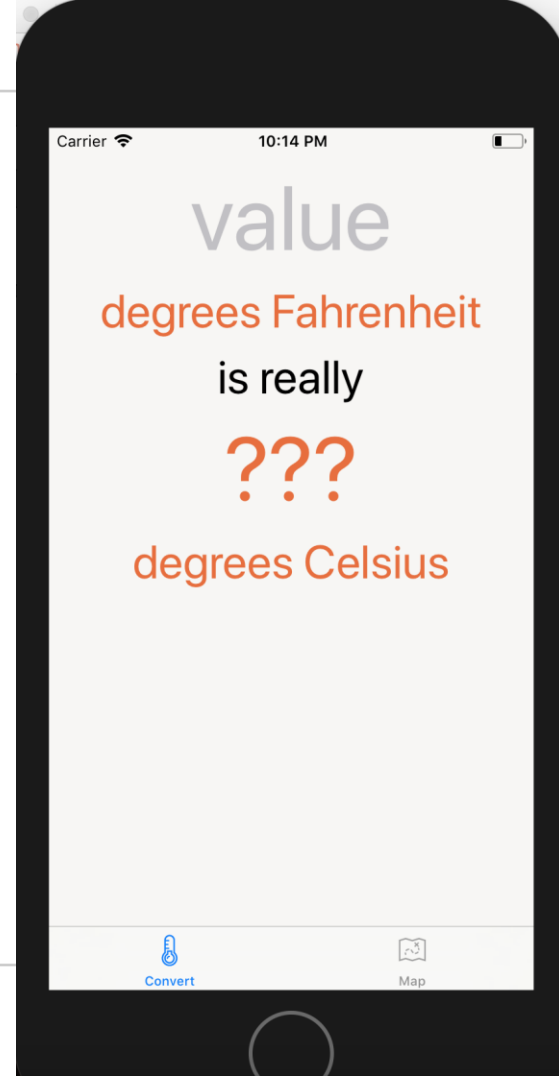
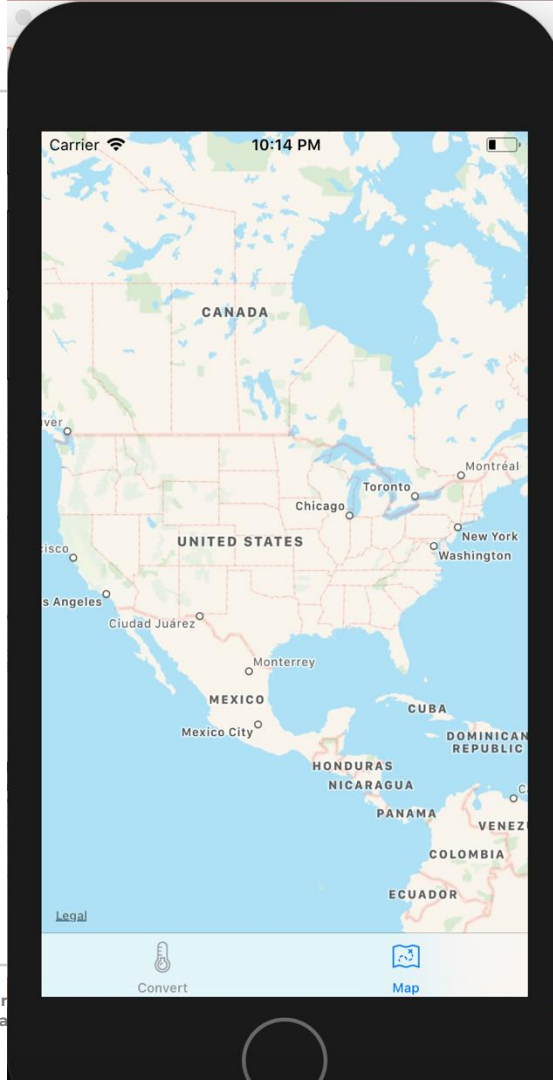
Modify the menu

Select Tab Bar Controller on canvas (or in storyboard)
Drag the Convert tab to be on the left



Modify the menu

Build, run and test

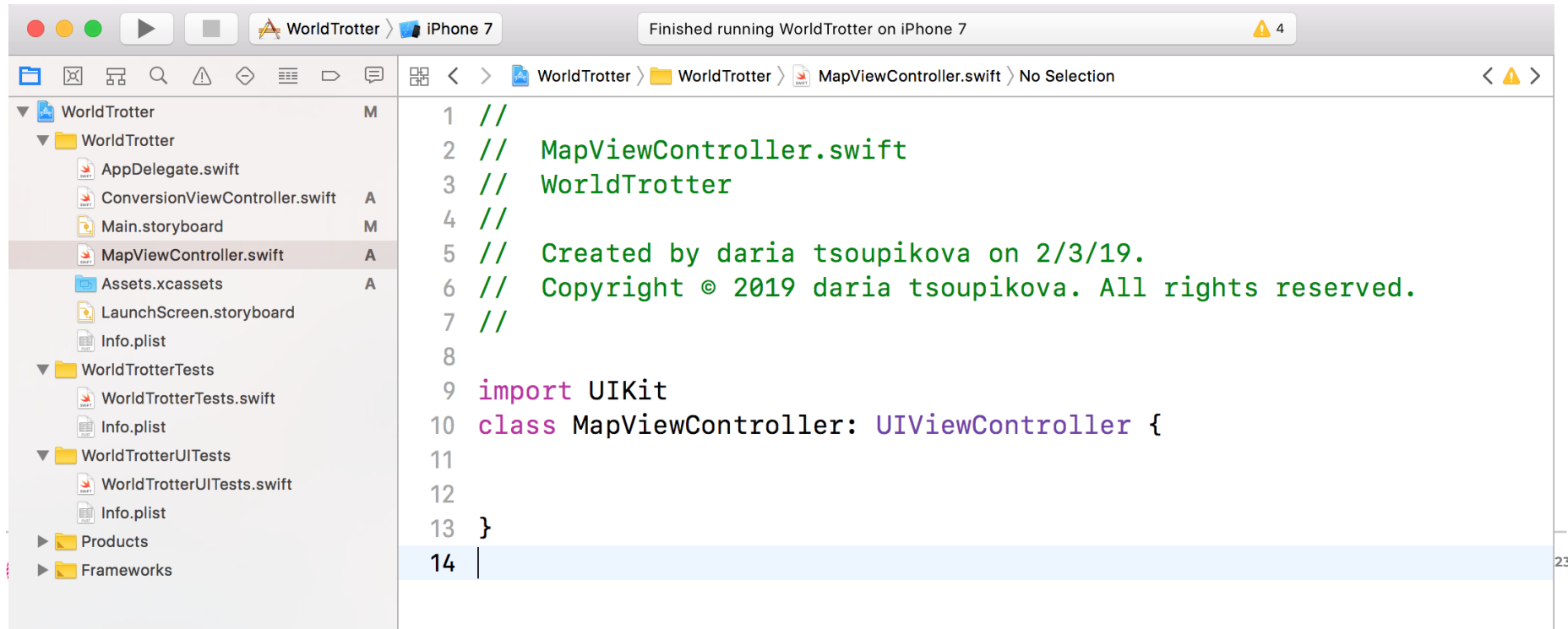


Optimize loading programmatically

MapViewController is not needed until user taps on the menu item

Create new Swift file "MapViewController"

Define a UIViewController subclass MapViewController

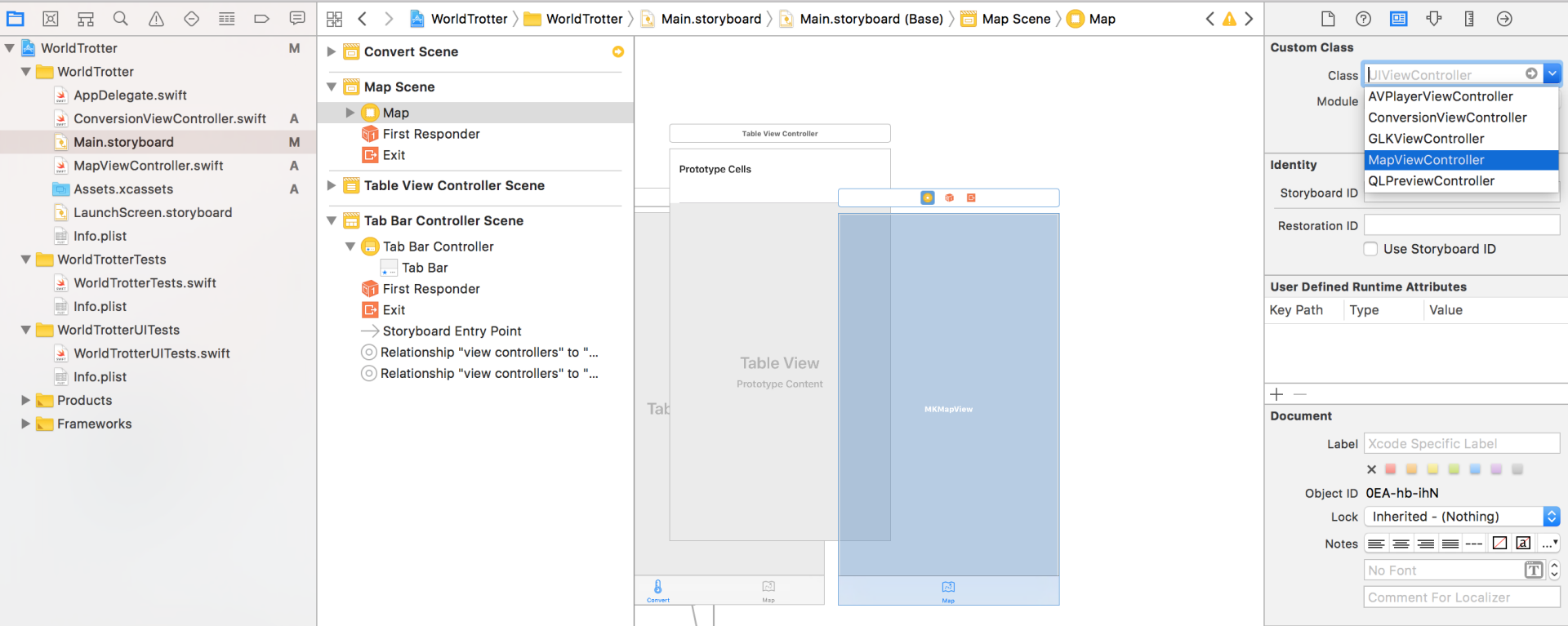


The screenshot shows the Xcode IDE interface. The top status bar indicates the app is finished running on an iPhone 7. The left sidebar shows the project structure for 'WorldTrotter', with 'MapViewController.swift' selected. The main editor area displays the Swift code for the new file, which includes a copyright notice and the beginning of a class definition for 'MapViewController'.

```
1 //
2 // MapViewController.swift
3 // WorldTrotter
4 //
5 // Created by daria tsoupikova on 2/3/19.
6 // Copyright © 2019 daria tsoupikova. All rights reserved.
7 //
8
9 import UIKit
10 class MapViewController: UIViewController {
11
12
13 }
14
```

Optimize loading programmatically

Open Main.Storyboard and select map's view controller
Open identity inspector and change the class to MapViewController class.



Lazy loading

Memory is limited. Test and see how long it takes to load the map...

Lazy loading make app run faster and makes memory more efficient.
Defers allocation until user needs it.

- Optimization
- The controller's view is not created until it needs to be presented on the screen
- Saves memory
- Improves performance

Optimize loading programmatically

In ConversionViewController.swift

Update viewDidLoad()

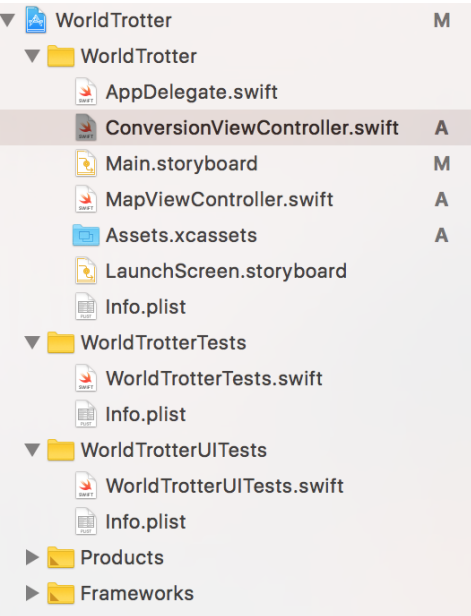
```
override func viewDidLoad() {  
super.viewDidLoad()  
  
print("ConversionViewController loaded its view")  
  
}
```



Optimize loading programmatically

In ConversionViewController.swift

Update viewDidLoad()



```
69         fahrenheitValue = nil
70     }
71 }
72
73 @IBAction func dismissKeyboard(_ sender:
74     UITapGestureRecognizer) {
75     textField.resignFirstResponder()
76 }
77
78 override func viewDidLoad() {
79     super.viewDidLoad()
80     print("ConversionViewController loaded its view")
81 }
```

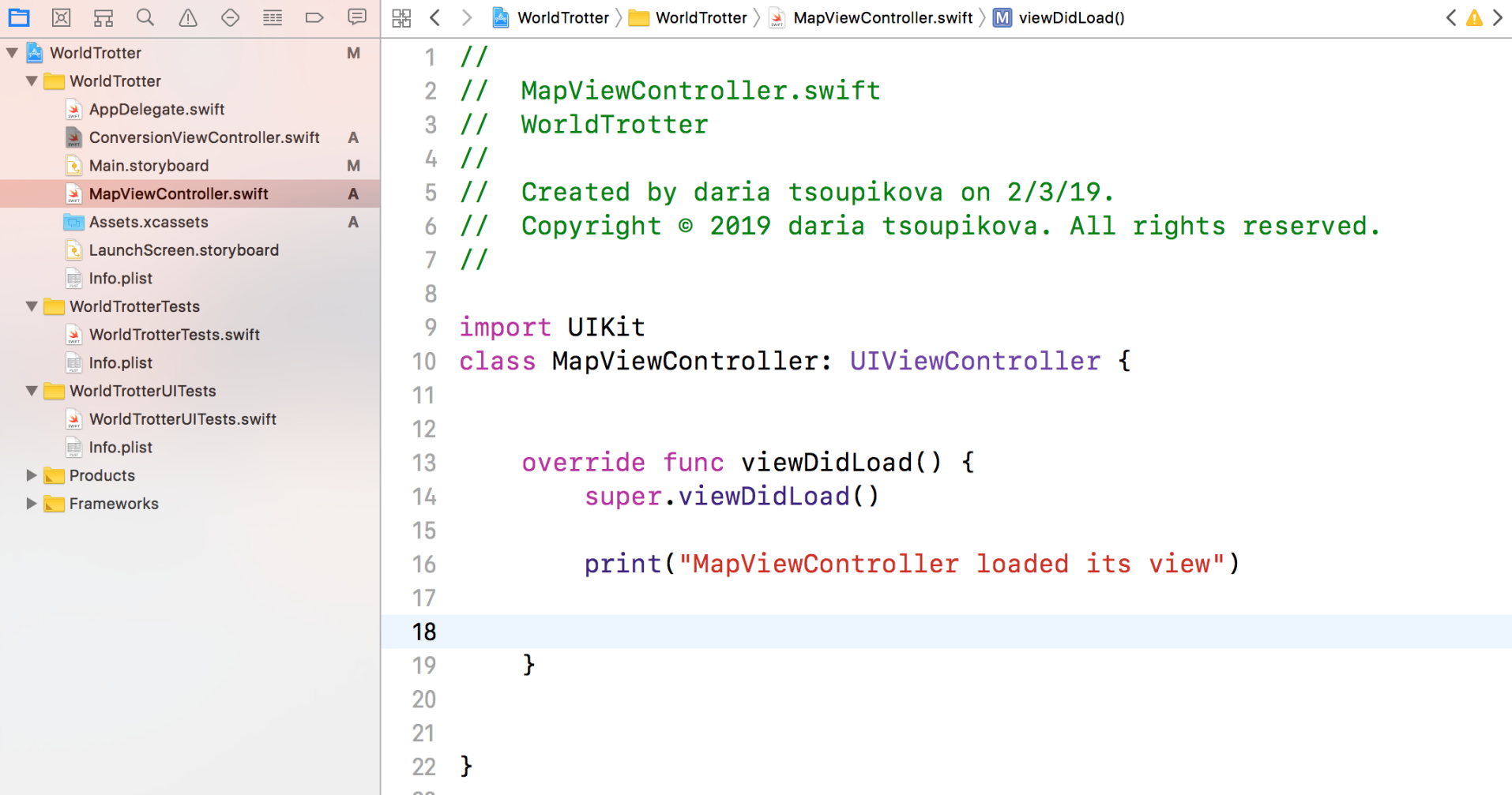
Optimize loading programmatically

In MapViewController.swift

Update viewDidLoad()

```
override func viewDidLoad() {  
    super.viewDidLoad()  
    // Do any additional setup after loading the view.  
  
    print("ConversionViewController loaded its view")  
}  
  
}
```

Optimize loading programmatically



The screenshot shows the Xcode interface with the project navigator on the left and the code editor on the right. The project navigator shows the project structure for 'WorldTrotter', with 'MapViewController.swift' selected. The code editor displays the following Swift code:

```
1 //
2 // MapViewController.swift
3 // WorldTrotter
4 //
5 // Created by daria tsoupikova on 2/3/19.
6 // Copyright © 2019 daria tsoupikova. All rights reserved.
7 //
8
9 import UIKit
10 class MapViewController: UIViewController {
11
12
13     override func viewDidLoad() {
14         super.viewDidLoad()
15
16         print("MapViewController loaded its view")
17
18
19     }
20
21
22 }
```

Optimize loading programmatically

Build, run and test.

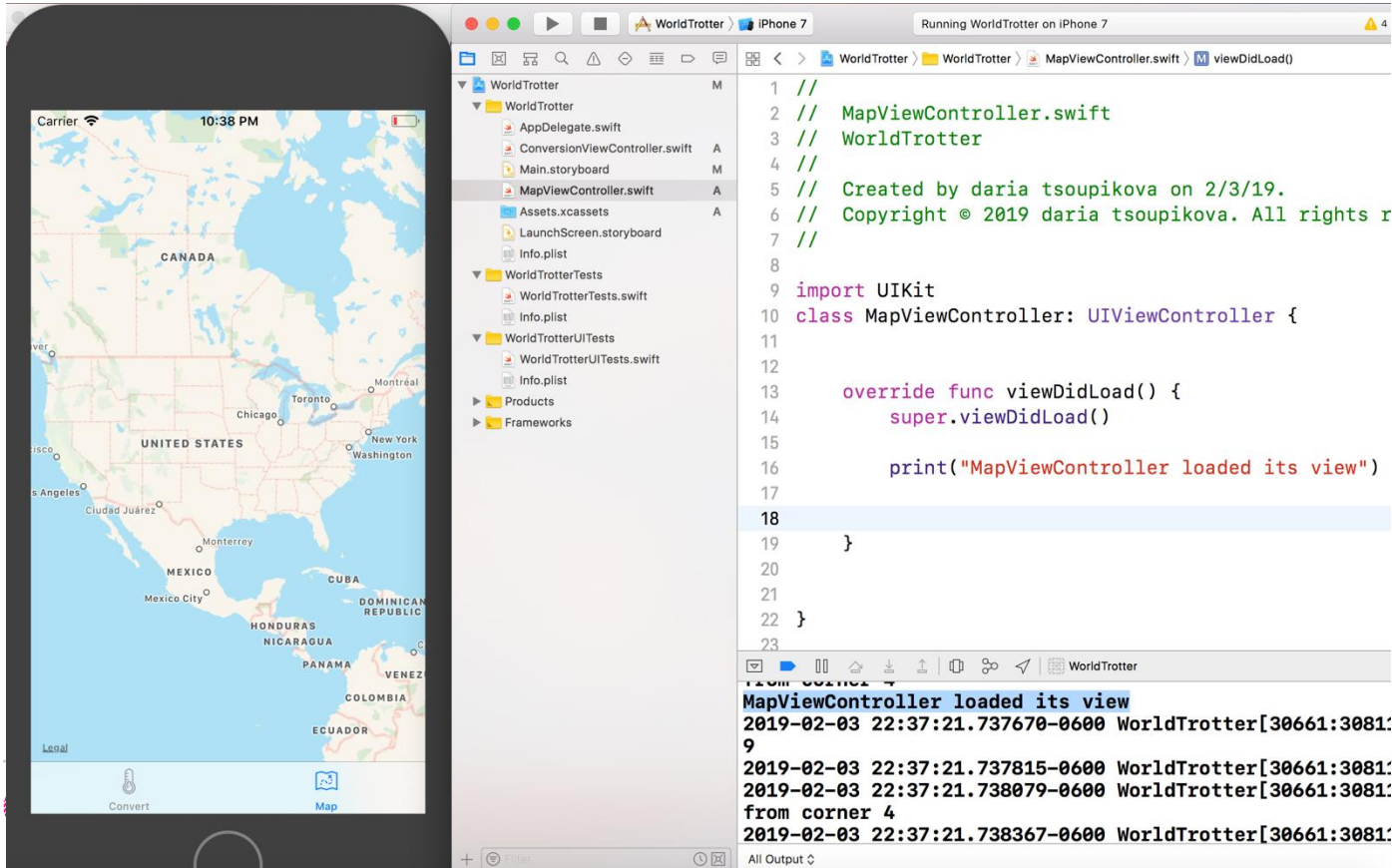
The image shows a screenshot of an iPhone simulator on the left and the Xcode editor on the right. The simulator displays a temperature conversion app with the text: "value degrees Fahrenheit is really ??? degrees Celsius". The Xcode editor shows the file structure of a project named "World Trotter" and the source code for "MapViewController.swift". The code defines a class "MapViewController" that inherits from "UIViewController" and overrides the "viewDidLoad()" method to print a log message. The "All Output" window at the bottom shows the execution output, including the log message and system information.

```
1 //
2 // MapViewController.swift
3 // WorldTrotter
4 //
5 // Created by daria tsoupikova on 2/3/19.
6 // Copyright © 2019 daria tsoupikova. All rights reserved.
7 //
8
9 import UIKit
10 class MapViewController: UIViewController {
11
12
13     override func viewDidLoad() {
14         super.viewDidLoad()
15
16         print("MapViewController loaded its view")
17
18     }
19 }
20
21
22 }
23
```

ConversionViewController loaded its view
2019-02-03 22:36:03.537198-0600 WorldTrotter[30661:3081:
MobileCoreServices.framework
2019-02-03 22:36:03.541215-0600 WorldTrotter[30661:3081:
MobileCoreServices.framework

Optimize loading programmatically

Build, run and test.



The image shows a screenshot of an iPhone simulator on the left and an Xcode editor on the right. The simulator displays a map of North America with labels for Canada, United States, Mexico, and other countries. The Xcode editor shows the source code for MapViewController.swift, which includes a viewDidLoad() method that prints a message to the console.

```
1 //  
2 // MapViewController.swift  
3 // WorldTrotter  
4 //  
5 // Created by daria tsoupikova on 2/3/19.  
6 // Copyright © 2019 daria tsoupikova. All rights reserved.  
7 //  
8  
9 import UIKit  
10 class MapViewController: UIViewController {  
11  
12     override func viewDidLoad() {  
13         super.viewDidLoad()  
14  
15         print("MapViewController loaded its view")  
16  
17     }  
18  
19 }  
20  
21  
22 }  
23
```

The console output shows the following log messages:

```
MapViewController loaded its view  
2019-02-03 22:37:21.737670-0600 WorldTrotter[30661:3081:9  
2019-02-03 22:37:21.737815-0600 WorldTrotter[30661:3081:  
2019-02-03 22:37:21.738079-0600 WorldTrotter[30661:3081:  
from corner 4  
2019-02-03 22:37:21.738367-0600 WorldTrotter[30661:3081:
```