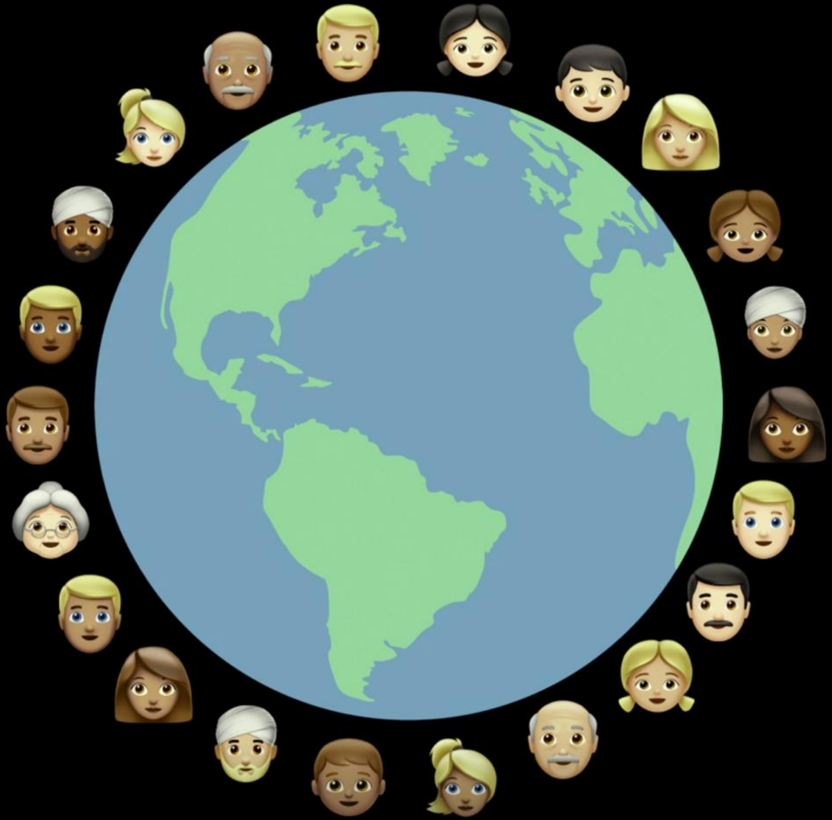
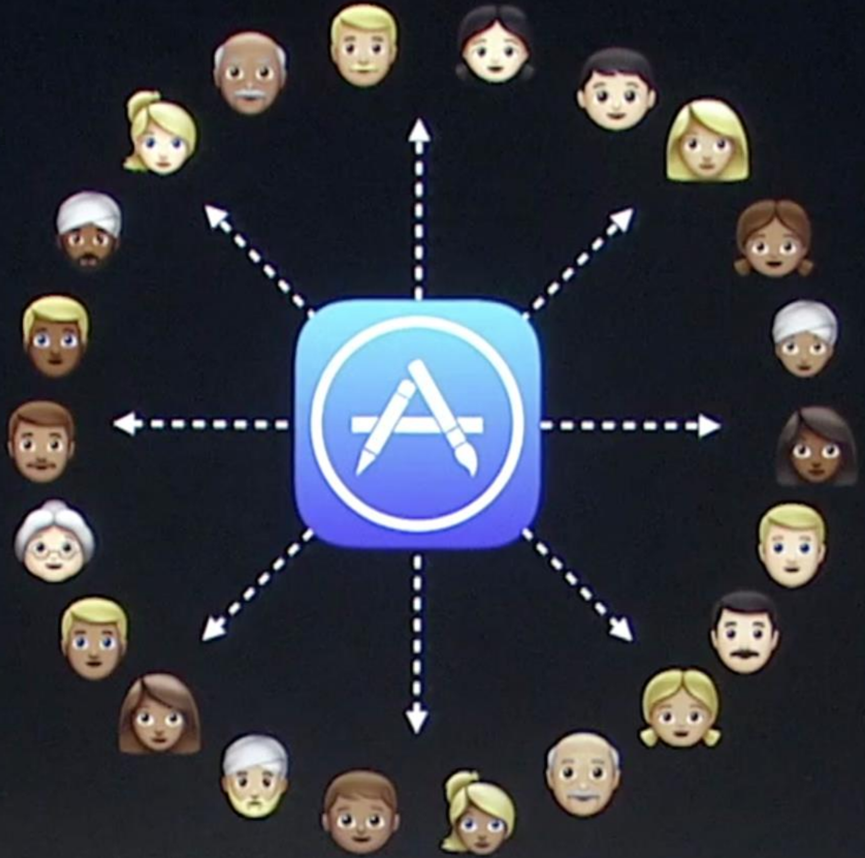
Internationalization
Localization

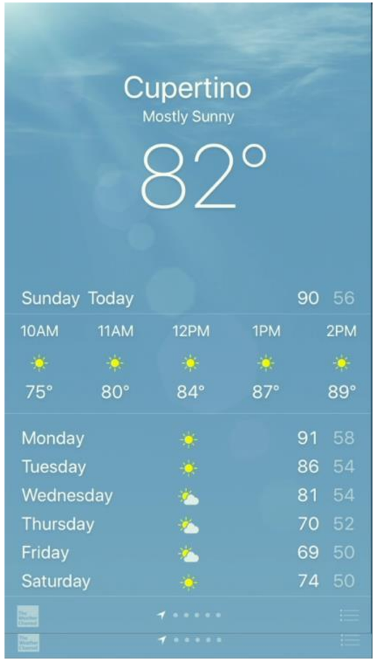




Internationalization

Providing local experiences for global users

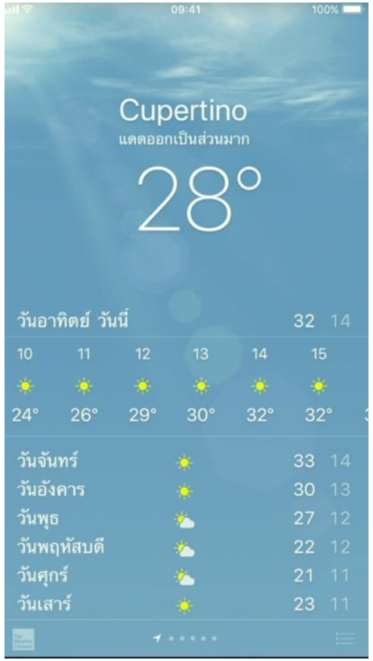
- **Strings management**
- **Formatting**
- **User Interface**


Cupertino
Mostly Sunny
82°

Sunday	Today	90	56

10AM	11AM	12PM	1PM	2PM
☀	☀	☀	☀	☀
75°	80°	84°	87°	89°

Monday	☀	91	58
Tuesday	☀	86	54
Wednesday	⛅	81	54
Thursday	⛅	70	52
Friday	⛅	69	50
Saturday	☀	74	50

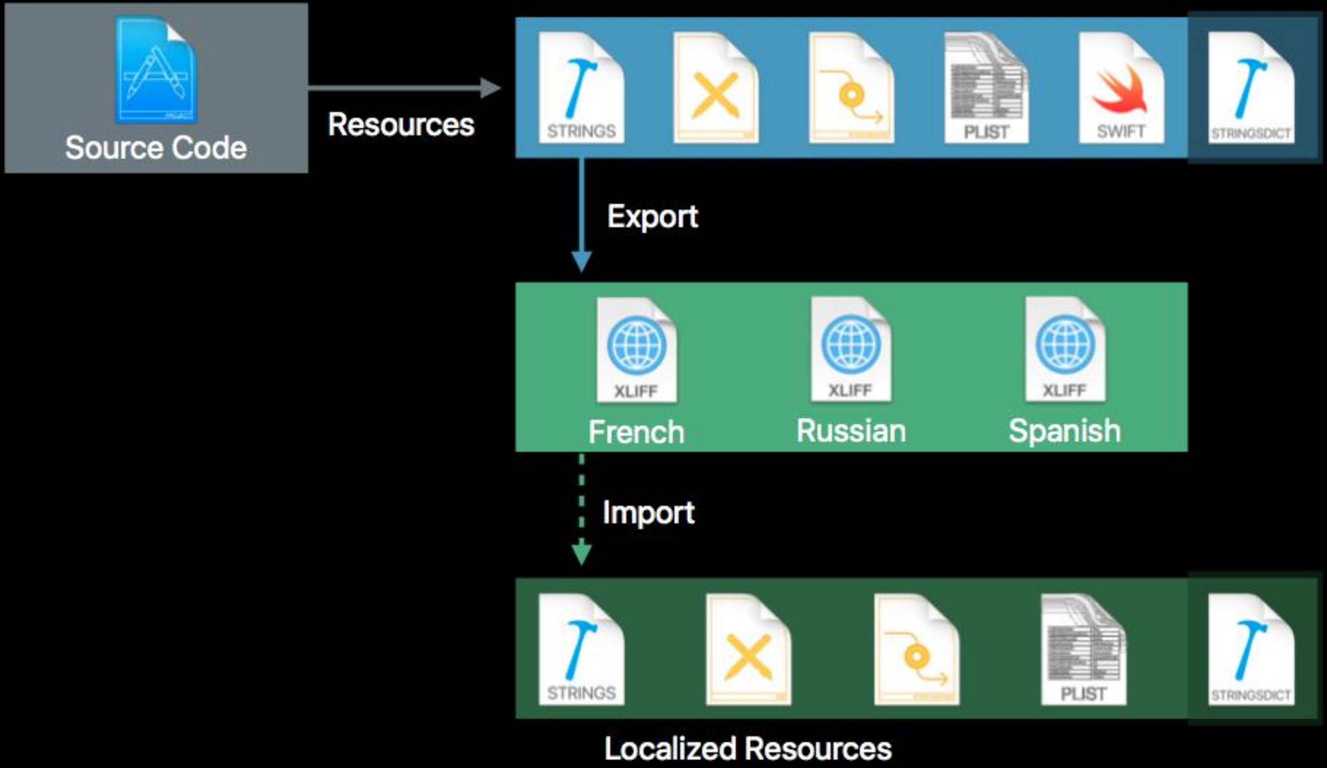

09:41 100%
Cupertino
แดดออกเป็นส่วนมาก
28°

วันอาทิตย์	วันนี้	32	14

10	11	12	13	14	15
☀	☀	☀	☀	☀	☀
24°	26°	29°	30°	32°	32°

วันจันทร์	☀	33	14
วันอังคาร	☀	30	13
วันพุธ	⛅	27	12
วันพฤหัสบดี	⛅	22	12
วันศุกร์	⛅	21	11
วันเสาร์	☀	23	11

Localization Process

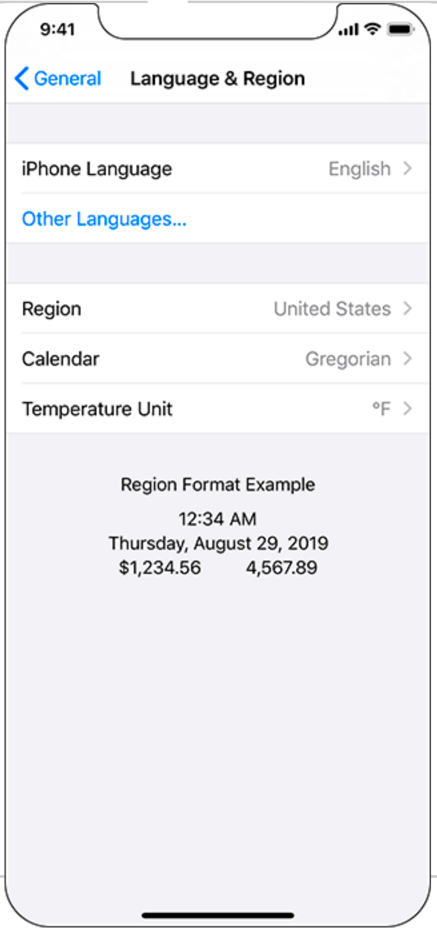


Internationalization

Internationalization (i18n) – the dynamic conversion of native cultural information (currency, number and date formats, language, etc.)

Localization (L10n) - is the process of providing appropriate data in the app based on the user's Language and Region Format setting. (Spanish)

Localization APIs
Easy to use in iOS
No precompillation is necessary


9:41

General | Language & Region

iPhone Language: English
Other Languages...

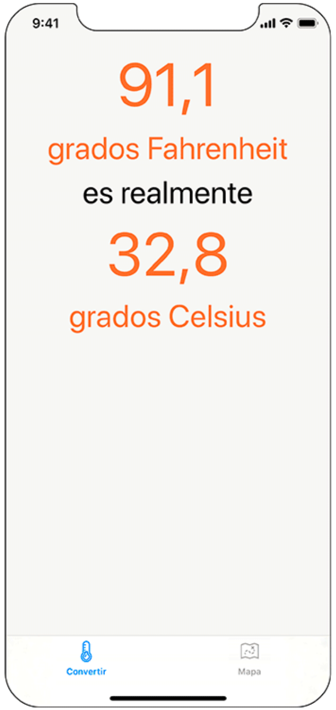
Region: United States
Calendar: Gregorian
Temperature Unit: °F

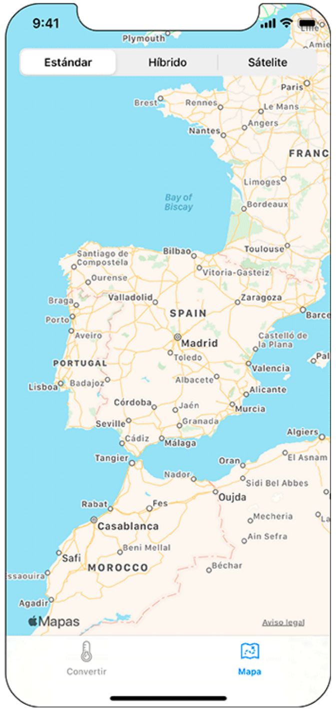
Region Format Example
12:34 AM
Thursday, August 29, 2019
$1,234.56 4,567.89

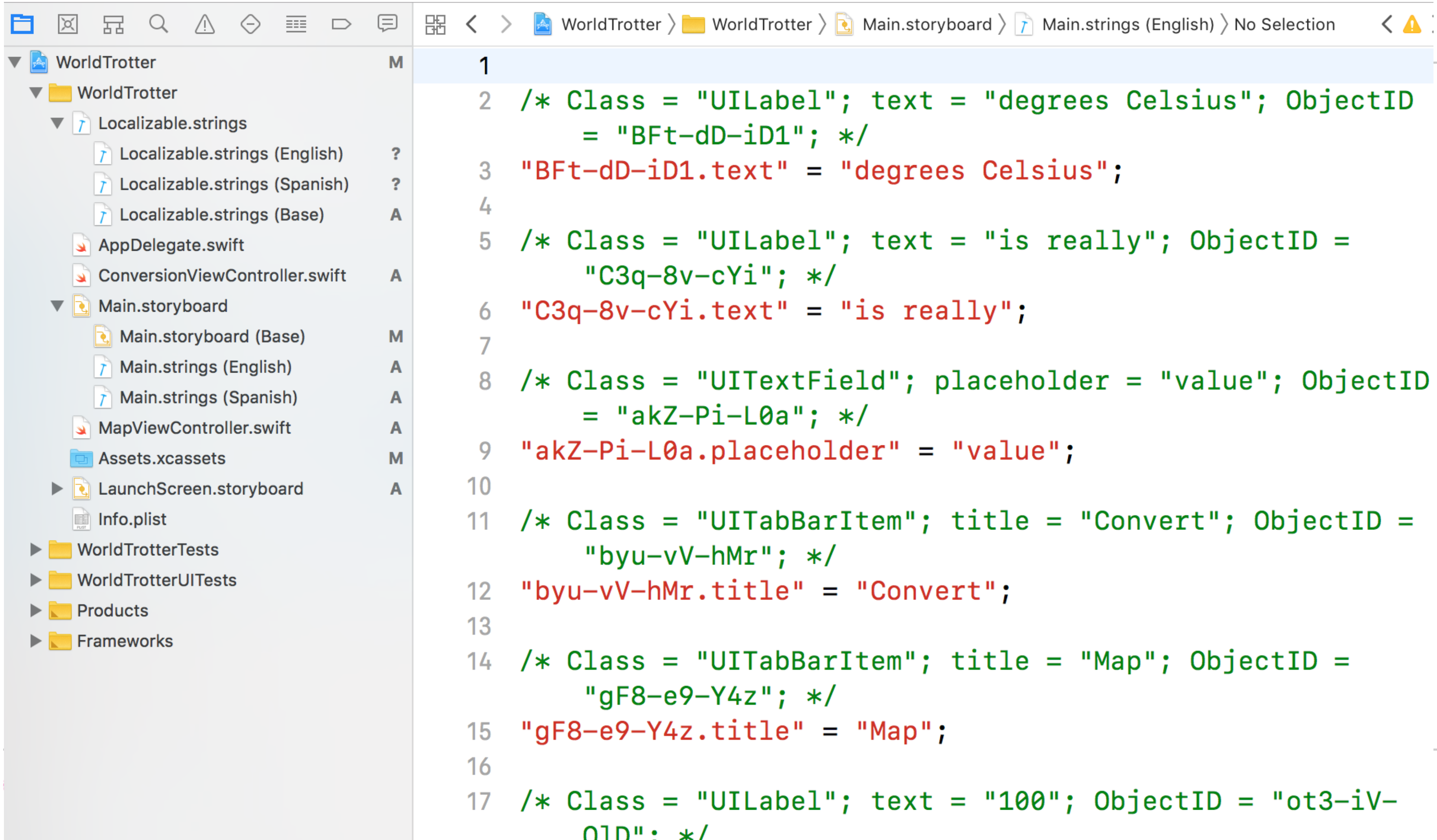
Internationalization

Internationalization

- Internationalizing the World Trotter App
- Localizing in Spanish






WorldTrotter 〉 WorldTrotter 〉 Main.storyboard 〉 Main.strings (English) 〉 No Selection
WorldTrotter M
WorldTrotter
Localizable.strings
Localizable.strings (English) ?
Localizable.strings (Spanish) ?
Localizable.strings (Base) A
AppDelegate.swift
ConversionViewController.swift A
Main.storyboard
Main.storyboard (Base) M
Main.strings (English) A
Main.strings (Spanish) A
MapViewController.swift A
Assets.xcassets M
LaunchScreen.storyboard A
Info.plist
WorldTrotterTests
WorldTrotterUITests
Products
Frameworks

```

/* Class = "UILabel"; text = "degrees Celsius"; ObjectID
    = "BFt-dD-iD1"; */
"BFt-dD-iD1.text" = "degrees Celsius";

/* Class = "UILabel"; text = "is really"; ObjectID =
    "C3q-8v-cYi"; */
"C3q-8v-cYi.text" = "is really";

/* Class = "UITextField"; placeholder = "value"; ObjectID
    = "akZ-Pi-L0a"; */
"akZ-Pi-L0a.placeholder" = "value";

/* Class = "UITabBarItem"; title = "Convert"; ObjectID =
    "byu-vV-hMr"; */
"byu-vV-hMr.title" = "Convert";

/* Class = "UITabBarItem"; title = "Map"; ObjectID =
    "gF8-e9-Y4z"; */
"gF8-e9-Y4z.title" = "Map";

/* Class = "UILabel"; text = "100"; ObjectID = "ot3-iV-
    O1D"; */
```

Internationalizing the app

NumberFormatter and NSNumber classes (Celsius label in ConversionViewController)

has

Locale property knows how to
display different regions symbols, dates, decimals, metric system, etc.

An instance of locale represents one region's settings for all the these variables.
Once you have that instance you can ask questions"
- Does this region use metric system?
- What currency symbol for this region?

let isMetric = curentLocle.usesMetricSystem
let currentSymbol = currentLoale.currencySymbol

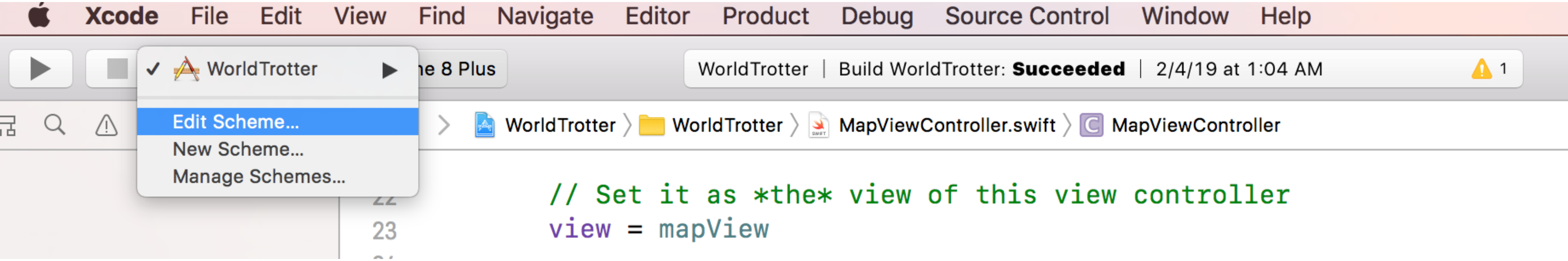
Internationalizing the app

Product> Scheme> Edit Scheme > options > App Regions > Europe > Spain

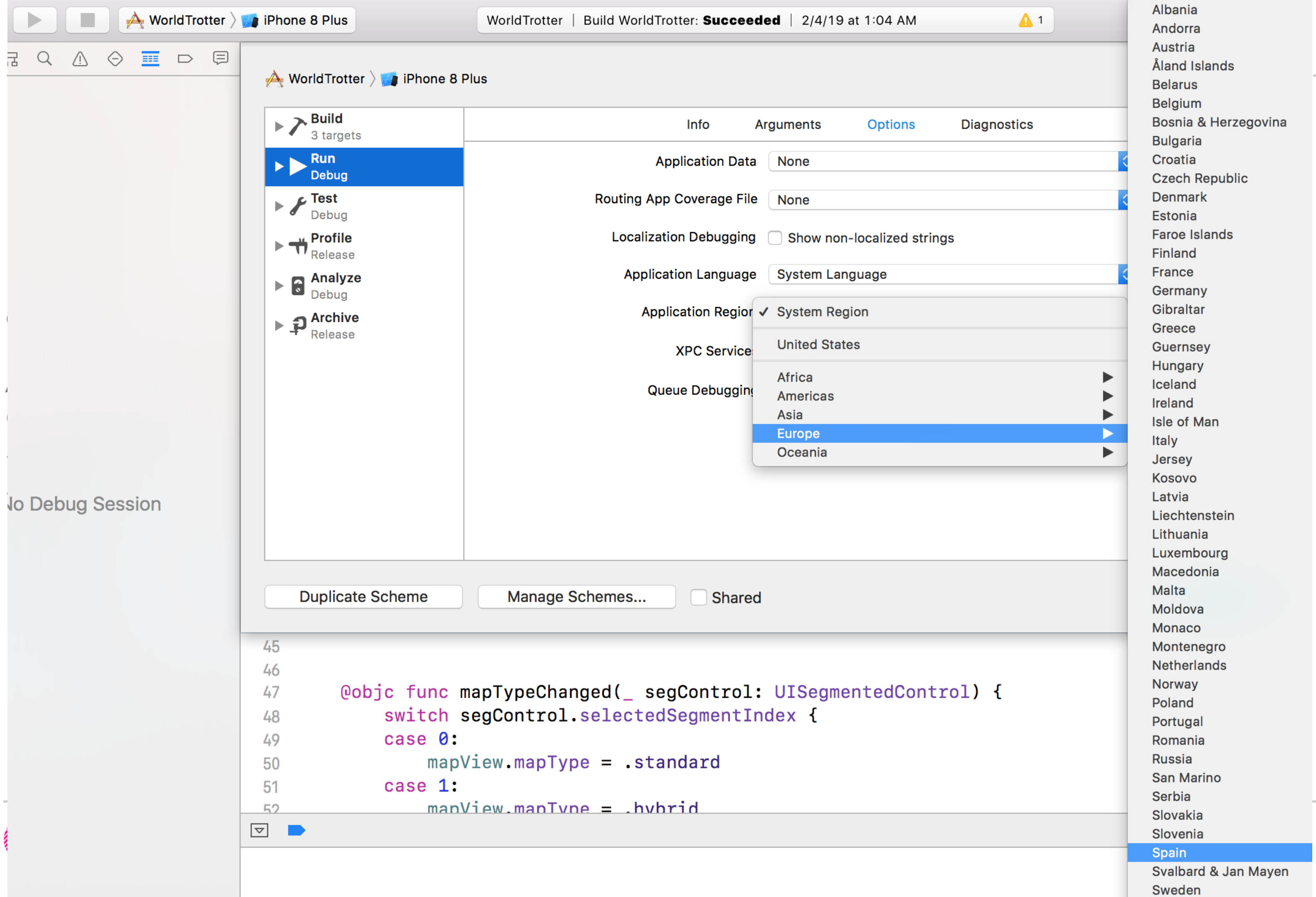
Build and run.
Notice the difference in Celsius / Fahrenheit decimals

In Spain decimal separator is comma instead of period
Type in decimal separators and the app will allow it, it only checks for a period instead of using locale-specific decimal separator.

“123,456.789 in the US - 123.456,789 in Spain



```
        // Set it as *the* view of this view controller
        view = mapView
```



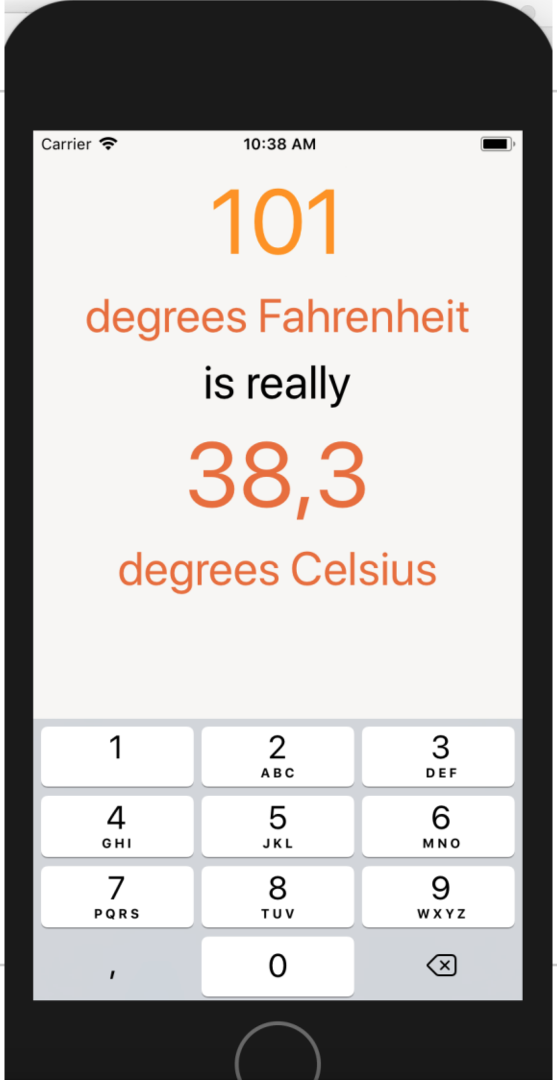
```

    @objc func mapTypeChanged(_ segControl: UISegmentedControl) {
        switch segControl.selectedSegmentIndex {
        case 0:
            mapView.mapType = .standard
        case 1:
            mapView.mapType = .hybrid
```

Internationalizing the app

Build and run.
Notice the difference in Celsius / Fahrenheit decimals

In Spain decimal separator is comma instead of period
Type in decimal separators and the app will allow it,
it only checks for a period instead of using
locale-specific decimal separator.

Type in several commas, the app allows it.
This needs to be fixed.

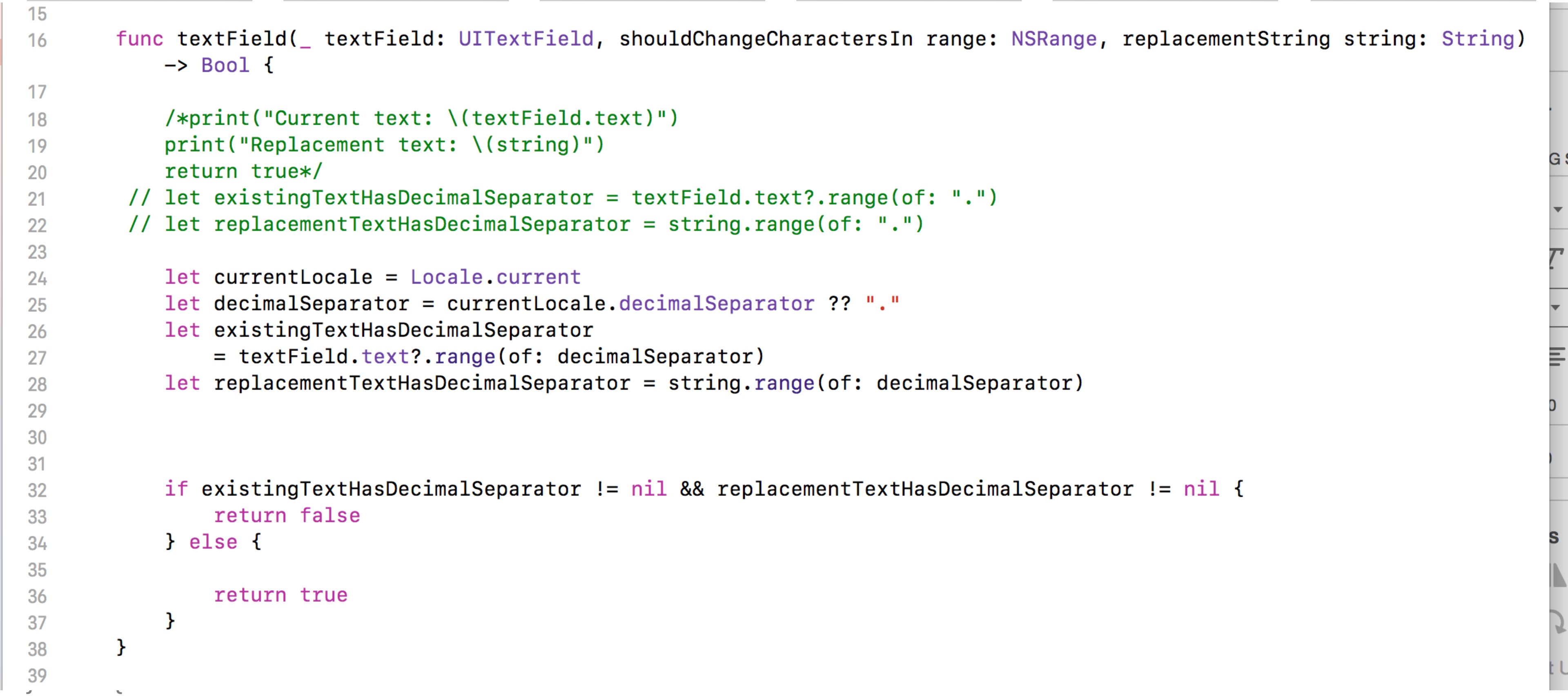


Internationalizing the app

Open ConversionController.swift update textfield function:

```
func textField(_ textField: UITextField,
             shouldChangeCharactersIn range: NSRange,
             replacementString string: String) -> Bool {

~~   let existingTextHasDecimalSeparator = textField.text?.range(of: ".")~~
~~   let replacementTextHasDecimalSeparator = string.range(of: ".")~~

   let currentLocale = Locale.current
   let decimalSeparator = currentLocale.decimalSeparator ?? "."

   let existingTextHasDecimalSeparator
         = textField.text?.range(of: decimalSeparator)
let replacementTextHasDecimalSeparator = string.range(of: decimalSeparator)

   if existingTextHasDecimalSeparator != nil,
      replacementTextHasDecimalSeparator != nil {
      return false
   } else {
      return true
```

Internationalizing the app



```swift

    func textField(_ textField: UITextField, shouldChangeCharactersIn range: NSRange, replacementString string: String)
        -> Bool {

        /*print("Current text: \(textField.text)")
        print("Replacement text: \(string)")
        return true*/
     // let existingTextHasDecimalSeparator = textField.text?.range(of: ".")
     // let replacementTextHasDecimalSeparator = string.range(of: ".")

        let currentLocale = Locale.current
        let decimalSeparator = currentLocale.decimalSeparator ?? "."
        let existingTextHasDecimalSeparator
            = textField.text?.range(of: decimalSeparator)
        let replacementTextHasDecimalSeparator = string.range(of: decimalSeparator)



        if existingTextHasDecimalSeparator != nil && replacementTextHasDecimalSeparator != nil {
            return false
        } else {

            return true
        }
    }

```

return true
}
}

Internationalizing the app

The app now should allow you to type in multiple decimal separators independent of user's region. But if you type in a number with comma, the conversion is not happening and ??? Is displayed.

The initializer does not know to to handle something other than a period as decimal separator.

You can fix it using **NumberFormatter** class- update fahrenheitFieldEditingChanged(_:) to convert the text field's string into a number in a locale-independent way.

Internationalizing the app

```
@IBAction func fahrenheitFieldEditingChanged(_ textField: UITextField) {

~~    if let text = textField.text, let value = Double(text) {~~
~~        fahrenheitValue = Measurement(value: value, unit: .fahrenheit)~~
    if let text = textField.text, let number = numberFormatter.number(from: text) {
        fahrenheitValue = Measurement(value: number.doubleValue, unit: .fahrenheit)

    } else {
        fahrenheitValue = nil
    }
}
```

Internationalizing the app

Uses **NumberFormatter** method **number(from:)**
To convert string into a number.

Because the number formatter is aware of the locale, it is able to convert the string into a number.

If the string contains a valid number, the method returns an instance of NSNumber.

NSNumber is a class that can represent a variety of number types, including Int, Float, Double, and more.

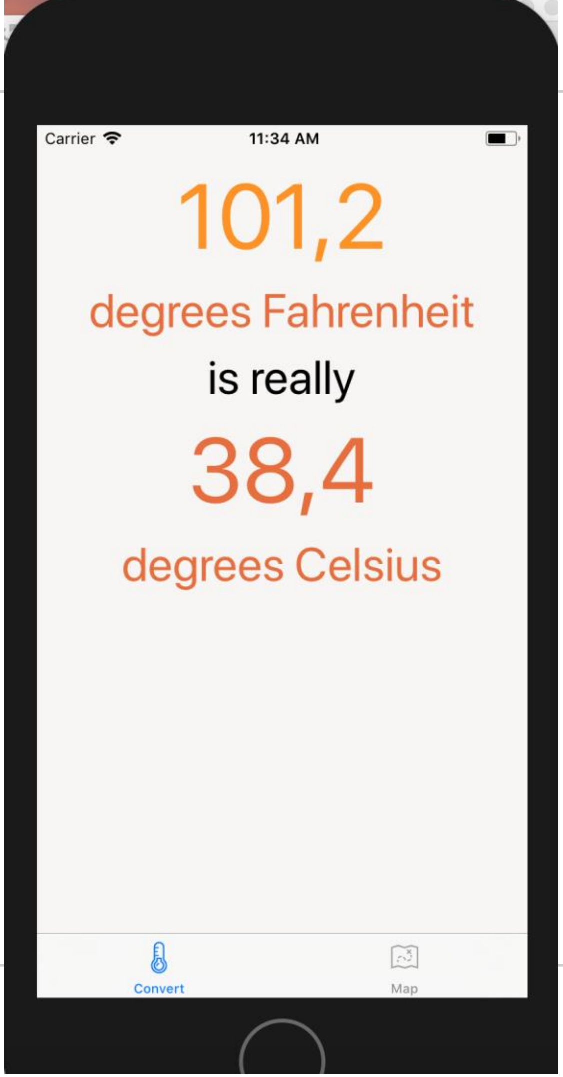
You can ask an instance of NSNumber for its value represented as one of those values.

You are doing that here to get the doubleValue of the number.

Internationalizing the app

Uses **NumberFormatter** method **number(from:)**
To convert string into a number.

Now that you are converting the string
in a locale-independent way, the text field's value
is properly converted to its Celsius value.



Base Internationalization

Localization usually involves either generating multiple copies of resources (like images, sounds, and interface files) for different regions and languages or creating and accessing strings tables (which you will see later in the chapter) to translate text into different languages.

When you build a target in Xcode, an application bundle is created. All of the resources that you added to the target in Xcode are copied into this bundle along with the executable itself. This bundle is represented at runtime by an instance of Bundle known as the main bundle. Many classes work with the Bundle to load resources.

Localizing a resource puts another copy of the resource in the application bundle. These resources are organized into language-specific directories, known as lproj directories. Each one of these directories is the name of the localization suffixed with lproj. For example, the American English localization is en_US, where en is the English language code and US is the United States of America region code, so the directory for American English resources is en_US.lproj. (The region can be omitted if you do not need to make regional distinctions in your resource files.) These language and region codes are standard on all platforms, not just iOS.

Base Internationalization

When a bundle is asked for the path of a resource file, it first looks at the root level of the bundle for a file of that name. If it does not find one, it looks at the locale and language settings of the device, finds the appropriate lproj directory, and looks for the file there.

Thus, just by localizing resource files, your application will automatically load the correct file.

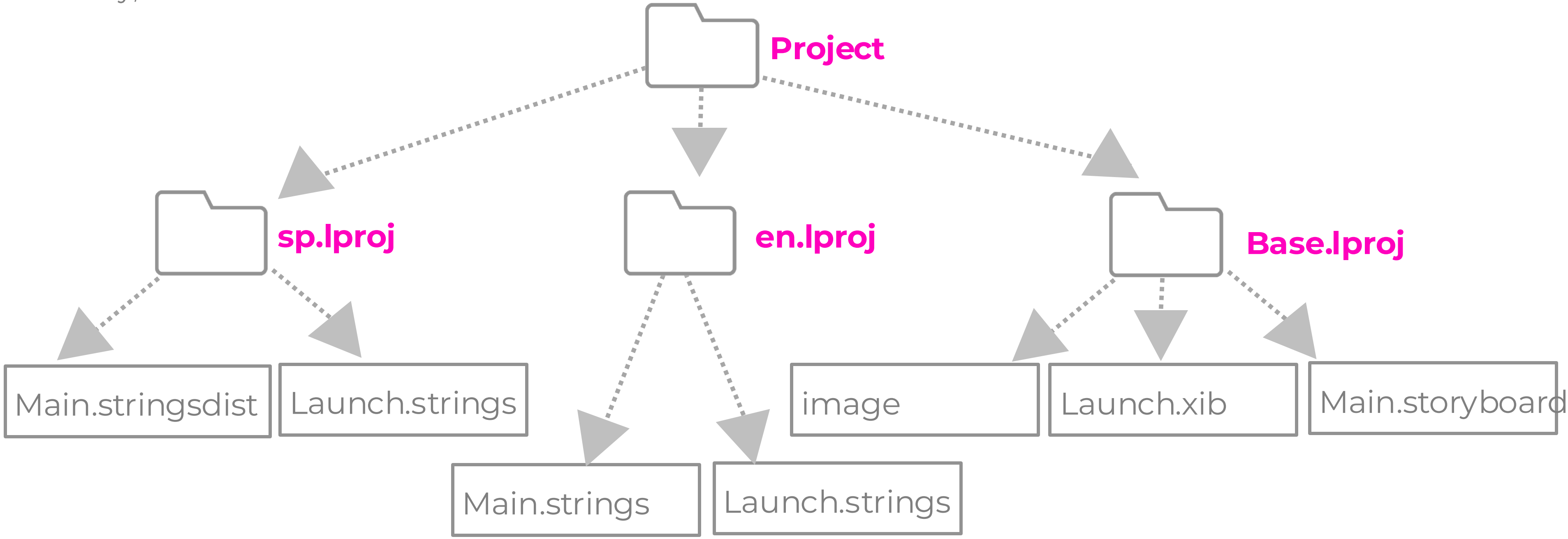
One option for localizing resource files is to create separate storyboard files and manually edit each string in each file. However, this approach does not scale well if you are planning multiple localizations. What happens when you add a new label or button to your localized storyboard? You have to add this view to the storyboard for every language. Not fun.

To simplify the process of localizing interface files, Xcode has **base internationalization.**

Base internationalization creates the Base.lproj directory, which contains the main interface files. Localizing individual interface files can then be done by creating just the Localizable.strings files. It is still possible to create the full interface files, in case localization cannot be done by changing strings alone.

Localized resources structure

Base internationalization creates the Base.lproj directory, which contains the main interface files. When a bundle is asked for the path of a resource file, it first looks at the root level of the bundle for a file of that name. If it does not find one, it looks at the locale and language settings of the device, finds the appropriate lproj directory, and looks for the file there.



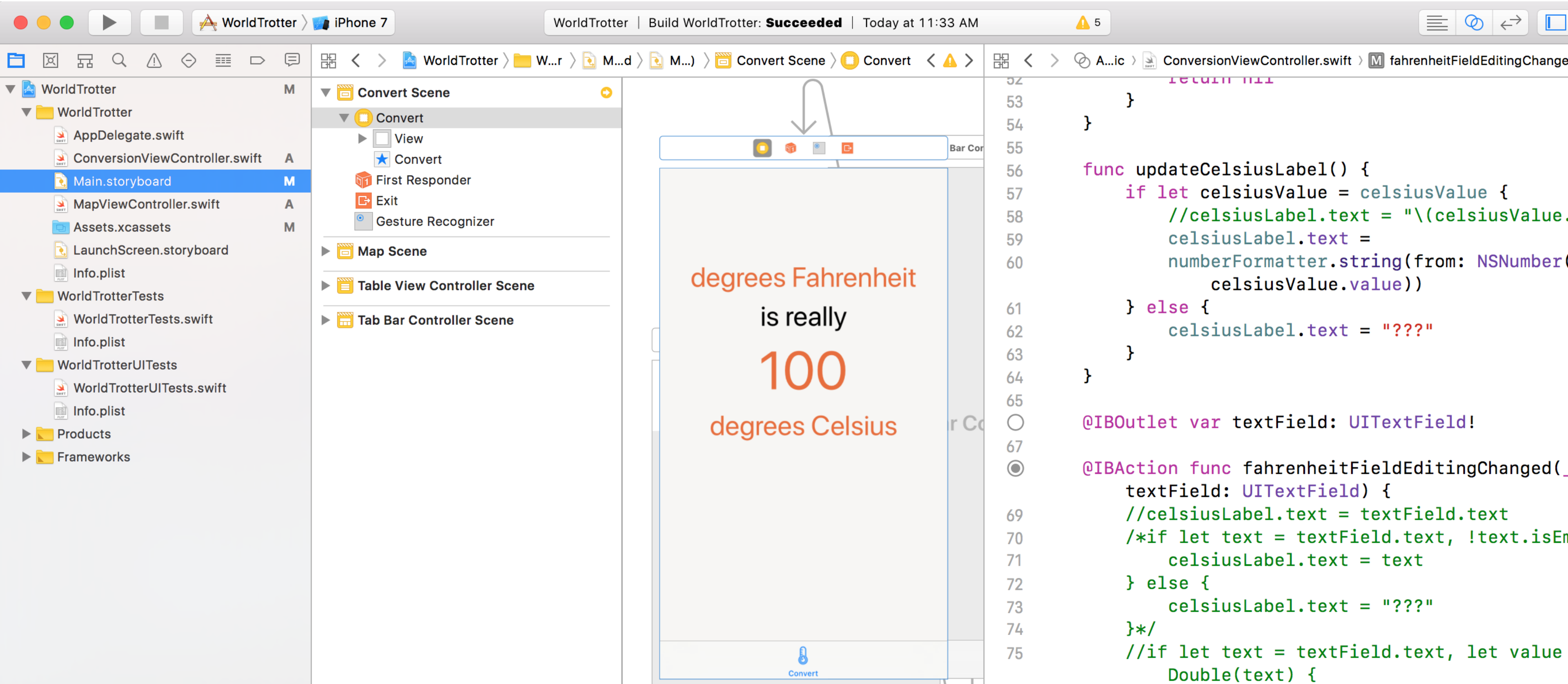
Base Internationalization

Apple’s Localization videos

https://developer.apple.com/videos/play/wwdc2017/401

Base Internationalization

Select Main.storyboard
Option+Command+return to open Preview



WorldTrotter | Build WorldTrotter: **Succeeded** | Today at 11:33 AM

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

Convert Scene
- Convert
  - View
  - Convert
- First Responder
- Exit
- Gesture Recognizer

Map Scene

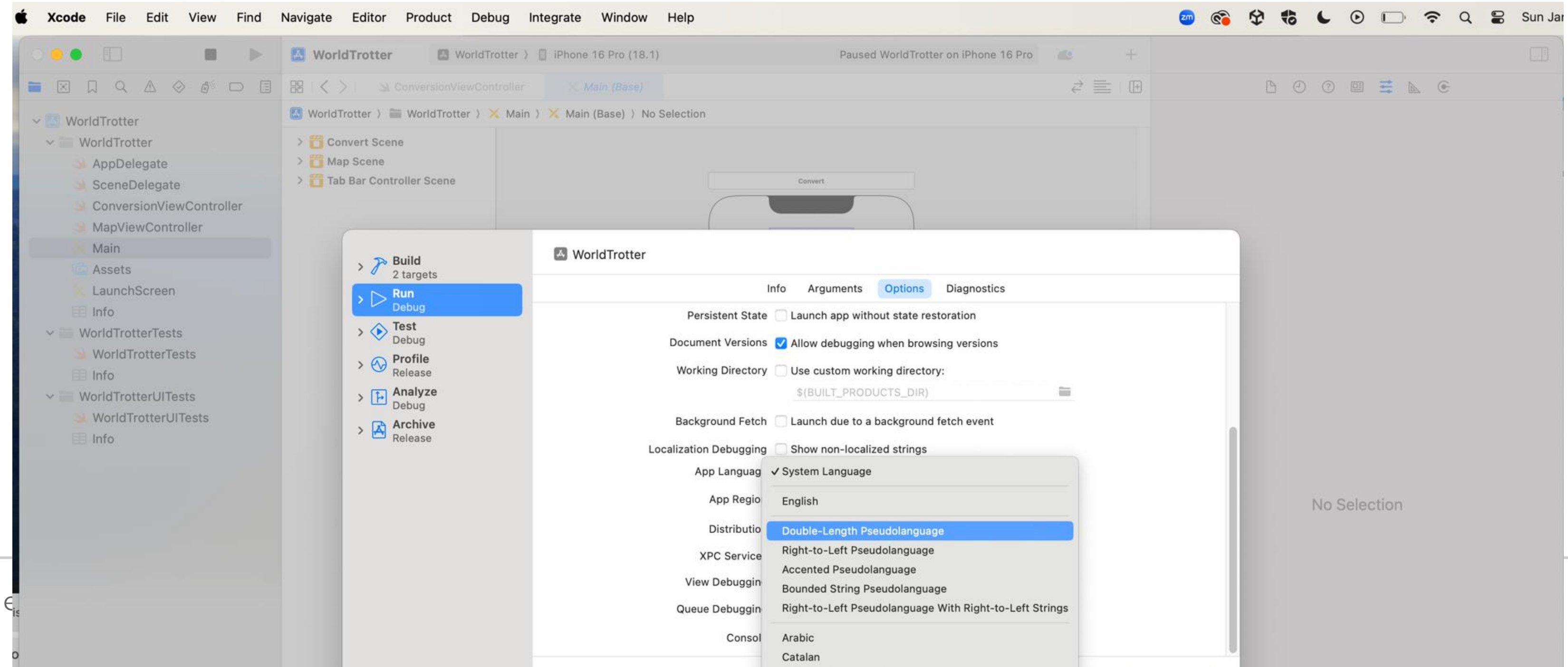
Table View Controller Scene

Tab Bar Controller Scene

```
        }
    }

    func updateCelsiusLabel() {
        if let celsiusValue = celsiusValue {
            //celsiusLabel.text = "\(celsiusValue.
            celsiusLabel.text =
            numberFormatter.string(from: NSNumber(
                celsiusValue.value))
        } else {
            celsiusLabel.text = "???"
        }
    }

    @IBOutlet var textField: UITextField!

    @IBAction func fahrenheitFieldEditingChanged(_
        textField: UITextField) {
        //celsiusLabel.text = textField.text
        /*if let text = textField.text, !text.isEm
            celsiusLabel.text = text
        } else {
            celsiusLabel.text = "???"
        }*/
        //if let text = textField.text, let value
            Double(text) {
```

Base Internationalization

Select Main.storyboard
Editor> Assistant
In Xcode, choose Product > Scheme > Edit Scheme. In the sheet that appears, select the Run scheme action in the left column, and click Options on the right. Choose one of the pseudolanguages at the bottom of the App Language pop-up menu and click Close in the sheet.



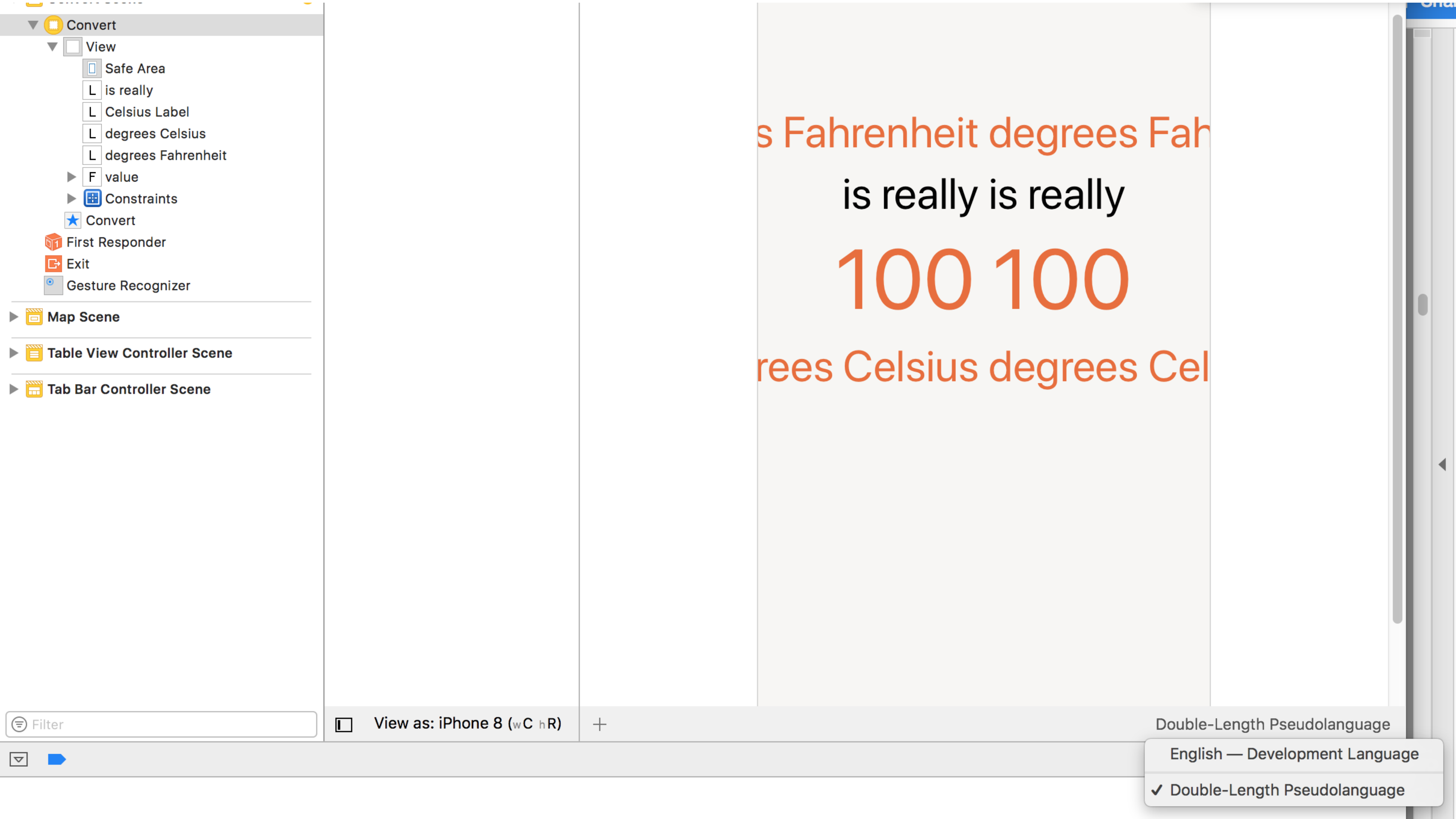
Persistent State ☐ Launch app without state restoration
Document Versions ☑ Allow debugging when browsing versions
Working Directory ☐ Use custom working directory:
$(BUILT_PRODUCTS_DIR)
Background Fetch ☐ Launch due to a background fetch event
Localization Debugging ☐ Show non-localized strings
App Languag ✓ System Language
App Regio
Distributio
XPC Service
View Debuggin
Queue Debuggin
Consol

Base Internationalization

Xcode supplies the built-in pseudolanguage to help you internationalize apps before receiving translations for all of strings and assets.

Pseudolanguage mimics languages that are more verbose by repeating whatever text string is in the text element. So, for example, “is nice” becomes “is nice is nice.”

Select double length Pseudolanguage

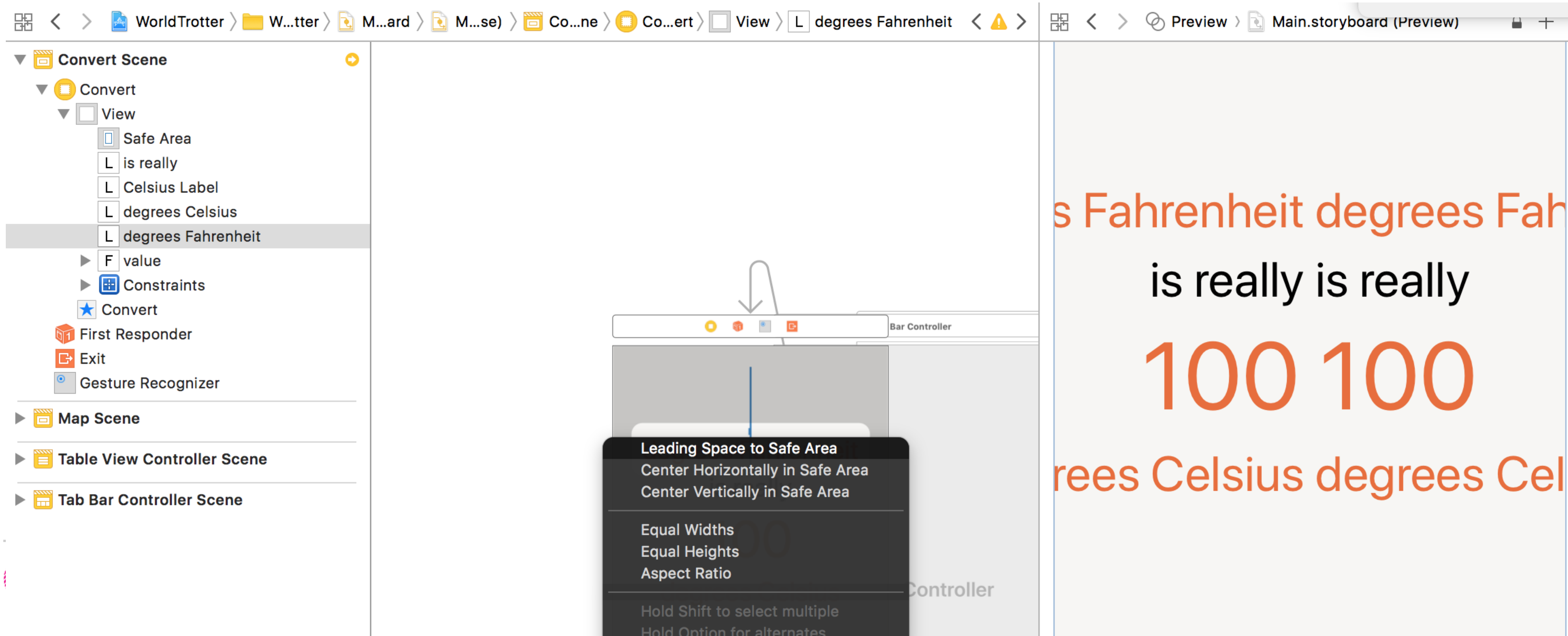

Convert
View
Safe Area
L is really
L Celsius Label
L degrees Celsius
L degrees Fahrenheit
F value
Constraints
Convert
First Responder
Exit
Gesture Recognizer
Map Scene
Table View Controller Scene
Tab Bar Controller Scene

s Fahrenheit degrees Fah
is really is really
100 100
rees Celsius degrees Cel

Filter
View as: iPhone 8 (wC hR)
Double-Length Pseudolanguage
English — Development Language
✓ Double-Length Pseudolanguage

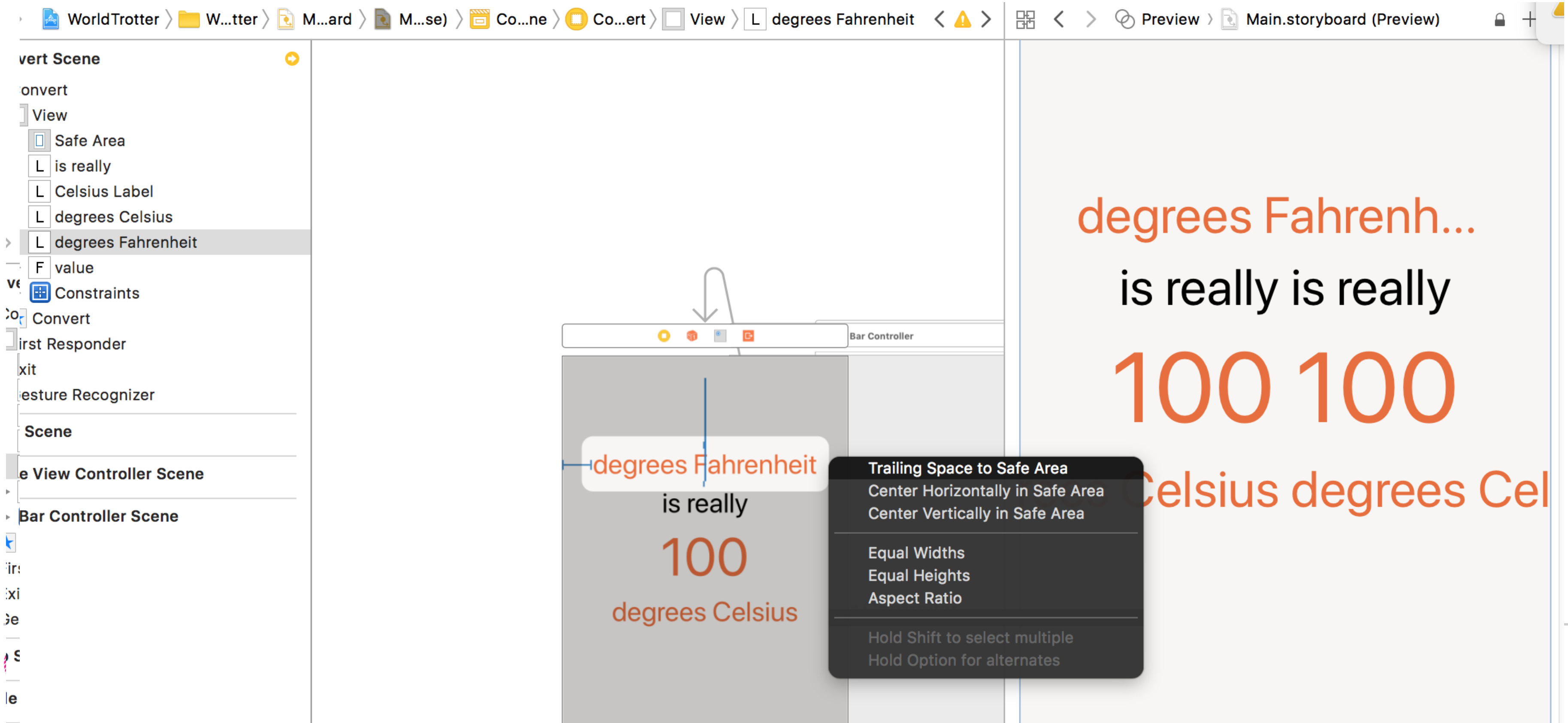
Base Internationalization

select the degrees Fahrenheit label. Control-drag from the label to the left side of the superview. When you do, a context-sensitive pop-up will appear giving you the constraints that make sense for this direction. Select Leading Space to Safe Area



Base Internationalization

Control-drag from the degrees Fahrenheit label to the right side of the superview and select Trailing Space to Safe Area (in the book “to Container Margin”.)



Base Internationalization

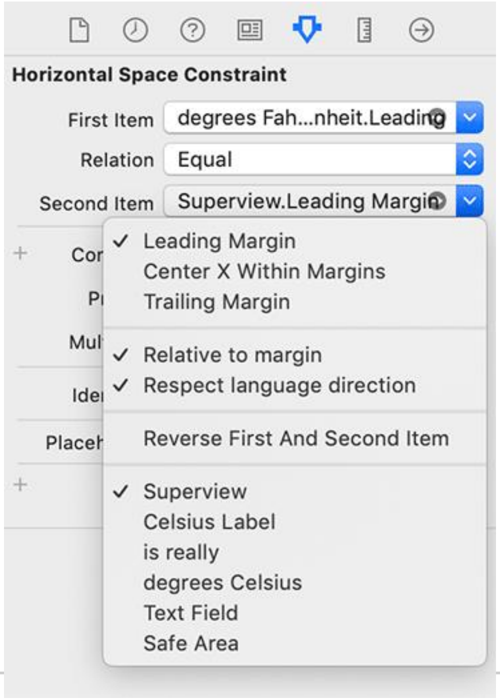
Select the leading constraint by clicking on the I-bar to the left of the label.
Open its attributes inspector and change the Relation to Greater Than or Equal and the Constant to 0.

Do the same for the trailing constraint.

Select the label and open its attributes inspector.
Change the Lines count to 0.

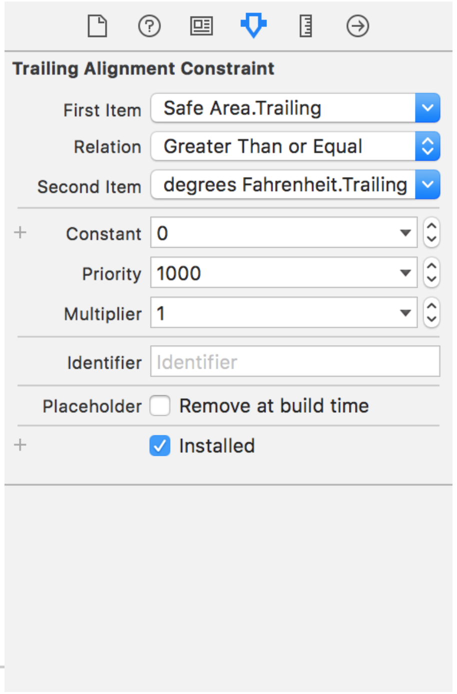
Now take a look at the preview assistant;
the label is no longer being truncated
and instead the text flows to a second line.

Change label alignment to center (in inspector)



Horizontal Space Constraint

First Item: degrees Fah...nheit.Leading
Relation: Equal
Second Item: Superview.Leading Margin



Trailing Alignment Constraint

First Item	Safe Area.Trailing
Relation	Greater Than or Equal
Second Item	degrees Fahrenheit.Trailing
Constant	0
Priority	1000
Multiplier	1
Identifier	Identifier
Placeholder	☐ Remove at build time
	✓ Installed

Base Internationalization

Repeat the steps above for the other labels. You will need to:

- Add a leading constraint to each label.
- Configure the constraint to be related to the superview’s leading margin.
- Set the constraints’ relation to Greater Than or Equal and the constant to 0. (A shortcut for editing a constraint is to double-click it.)
- Change the label’s line count to 0.
- Change the label’s alignment to Center.

Base Internationalization

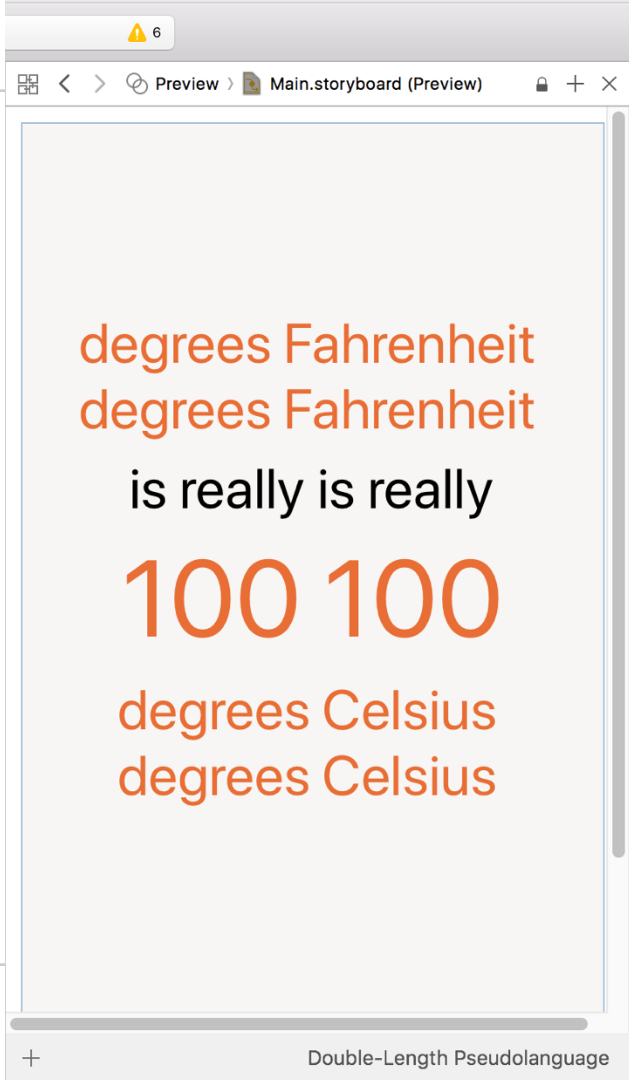
Repeat steps for each label:

- Add a leading and trailing constraint to each label.
- Set the constraints' relation to Greater Than or Equal and the constant to 0.
- Change the label's line count to 0.

Close the Preview window after done.

The app is Internationalized:

The app's its interface is now able to adapt to various languages and regions.



degrees Fahrenheit degrees Fahrenheit

is really is really

100 100

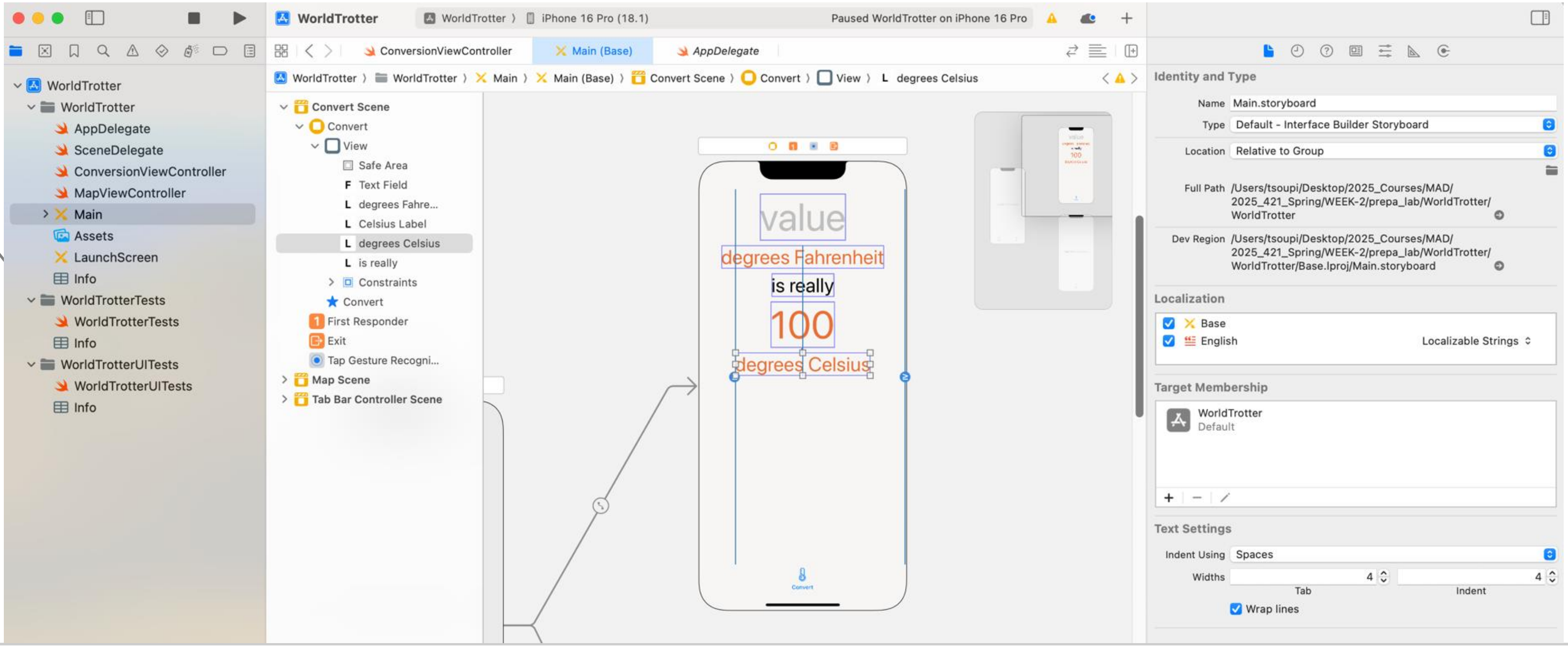
degrees Celsius degrees Celsius

Localization

Localizing the app

Click the the Main.storyboard , Open the File inspector

Make sure that the reference language is Base and the file type is Localizable Strings.



Identity and Type

Name	Main.storyboard
Type	Default - Interface Builder Storyboard
Location	Relative to Group
Full Path	/Users/tsoupi/Desktop/2025_Courses/MAD/2025_421_Spring/WEEK-2/prepa_lab/WorldTrotter/WorldTrotter
Dev Region	/Users/tsoupi/Desktop/2025_Courses/MAD/2025_421_Spring/WEEK-2/prepa_lab/WorldTrotter/WorldTrotter/Base.lproj/Main.storyboard

Localization

- ✓ Base
- ✓ English — Localizable Strings

Target Membership

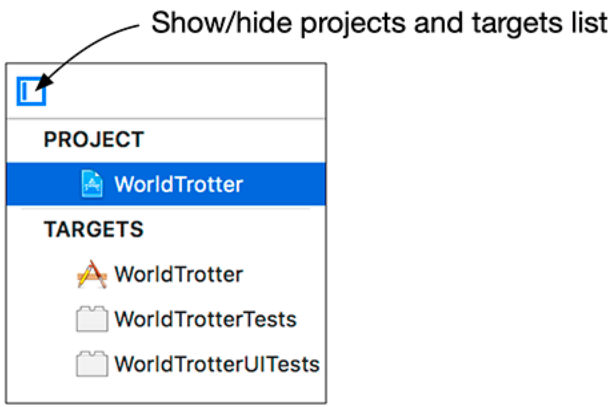
WorldTrotter
Default

Text Settings

Indent Using	Spaces
Widths	Tab 4, Indent 4
✓ Wrap lines	

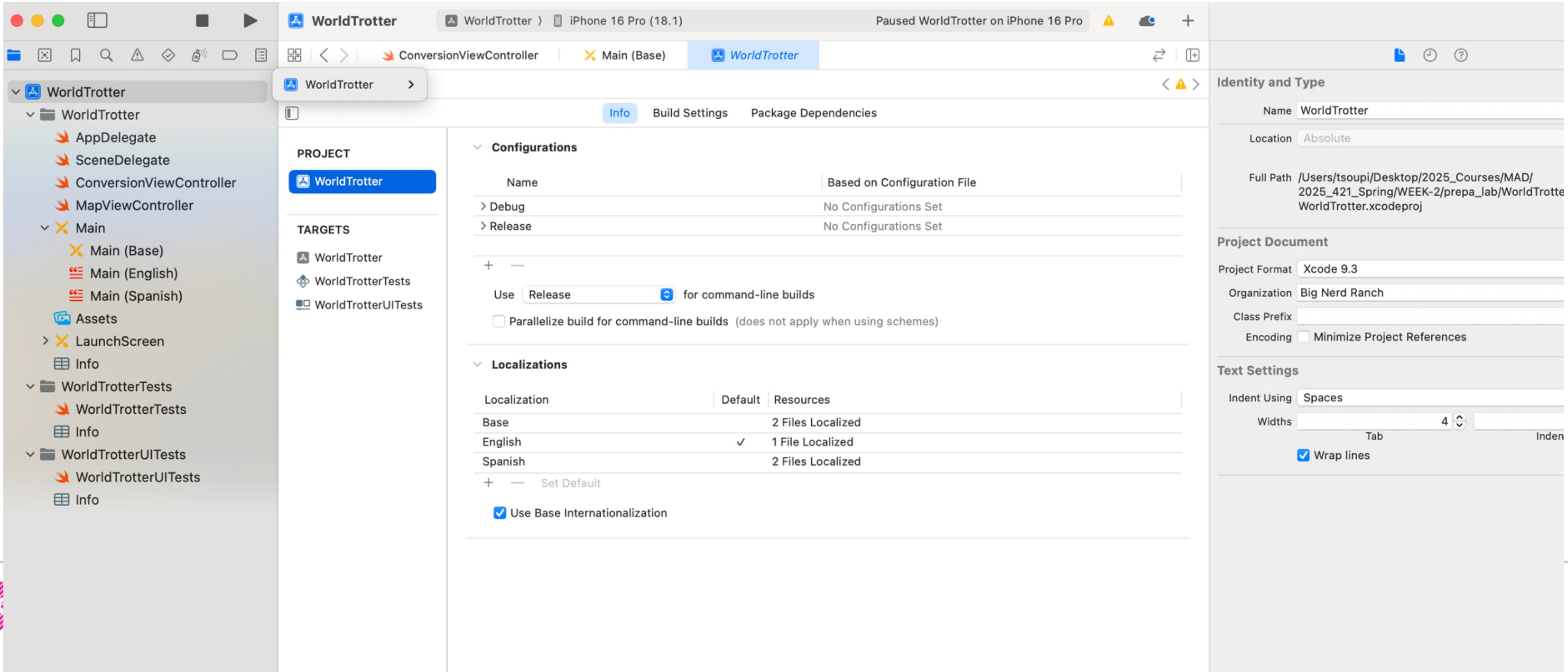
Localizing the app

Next, in the project navigator, select the WorldTrotter project at the top.



Localizing the app

Click the button under Localizations and select Spanish (es). In the dialog, uncheck the LaunchScreen.storyboard file; keep the Main.storyboard file checked. Make sure that the reference language is Base and the file type is Localizable Strings. Click Finish. This creates an es.lproj folder and generates the Main.strings file in it that contains all the strings from the base interface file.



Configurations

Name	Based on Configuration File
Debug	No Configurations Set
Release	No Configurations Set

Use Release for command-line builds

☐ Parallelize build for command-line builds (does not apply when using schemes)

Localizations

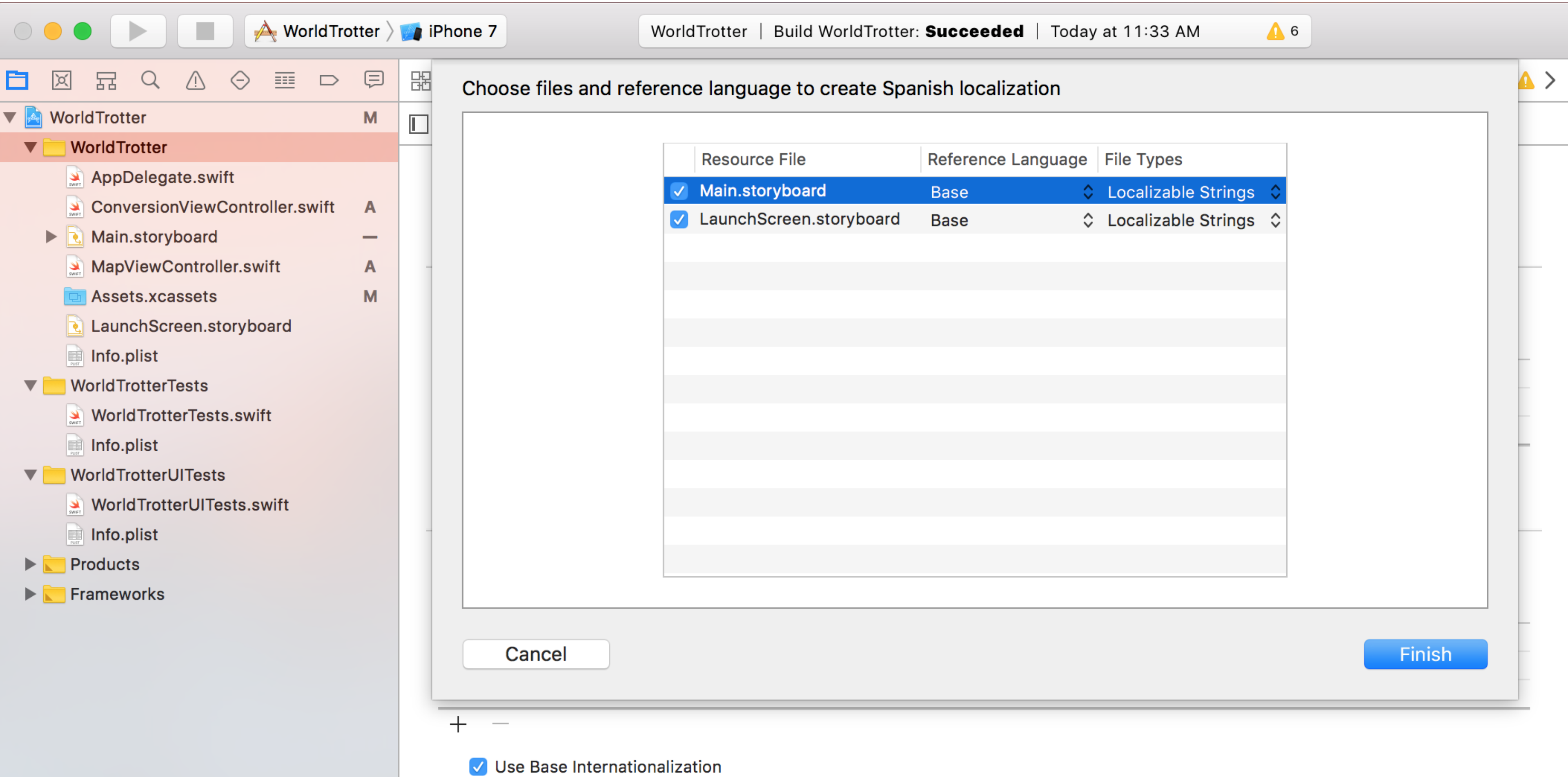
Localization	Default	Resources
Base		2 Files Localized
English	✓	1 File Localized
Spanish		2 Files Localized

Set Default

☑ Use Base Internationalization

Name	WorldTrotter
Location	Absolute
Full Path	/Users/tsoupi/Desktop/2025_Courses/MAD/2025_421_Spring/WEEK-2/prepa_lab/WorldTrotte WorldTrotter.xcodeproj
Project Format	Xcode 9.3
Organization	Big Nerd Ranch
Class Prefix	
Encoding	☐ Minimize Project References
Indent Using	Spaces
Widths	4 Tab / Inden
	☑ Wrap lines

Localizing the app



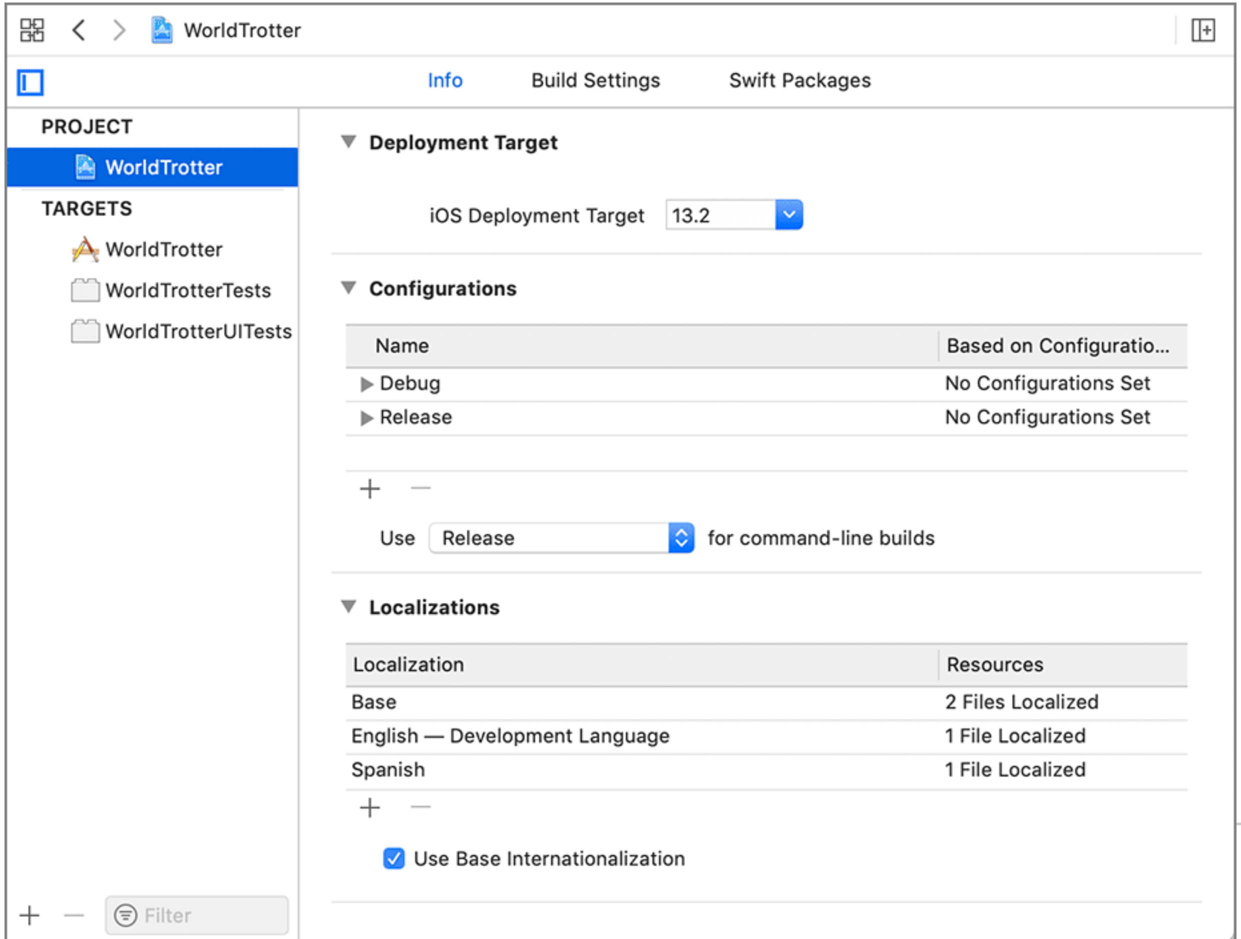
WorldTrotter | Build WorldTrotter: **Succeeded** | Today at 11:33 AM

Choose files and reference language to create Spanish localization

	Resource File	Reference Language	File Types
✓	Main.storyboard	Base	Localizable Strings
✓	LaunchScreen.storyboard	Base	Localizable Strings

✓ Use Base Internationalization

Localizing the app


WorldTrotter

Info | Build Settings | Swift Packages

PROJECT
- WorldTrotter

TARGETS
- WorldTrotter
- WorldTrotterTests
- WorldTrotterUITests

Deployment Target

iOS Deployment Target: 13.2

Configurations

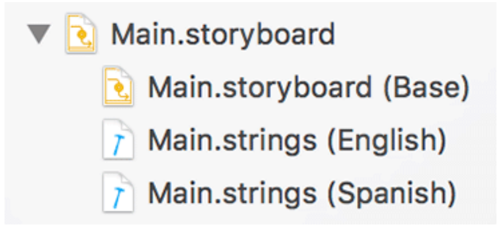
Name	Based on Configuratio...
▶ Debug	No Configurations Set
▶ Release	No Configurations Set

Use Release for command-line builds

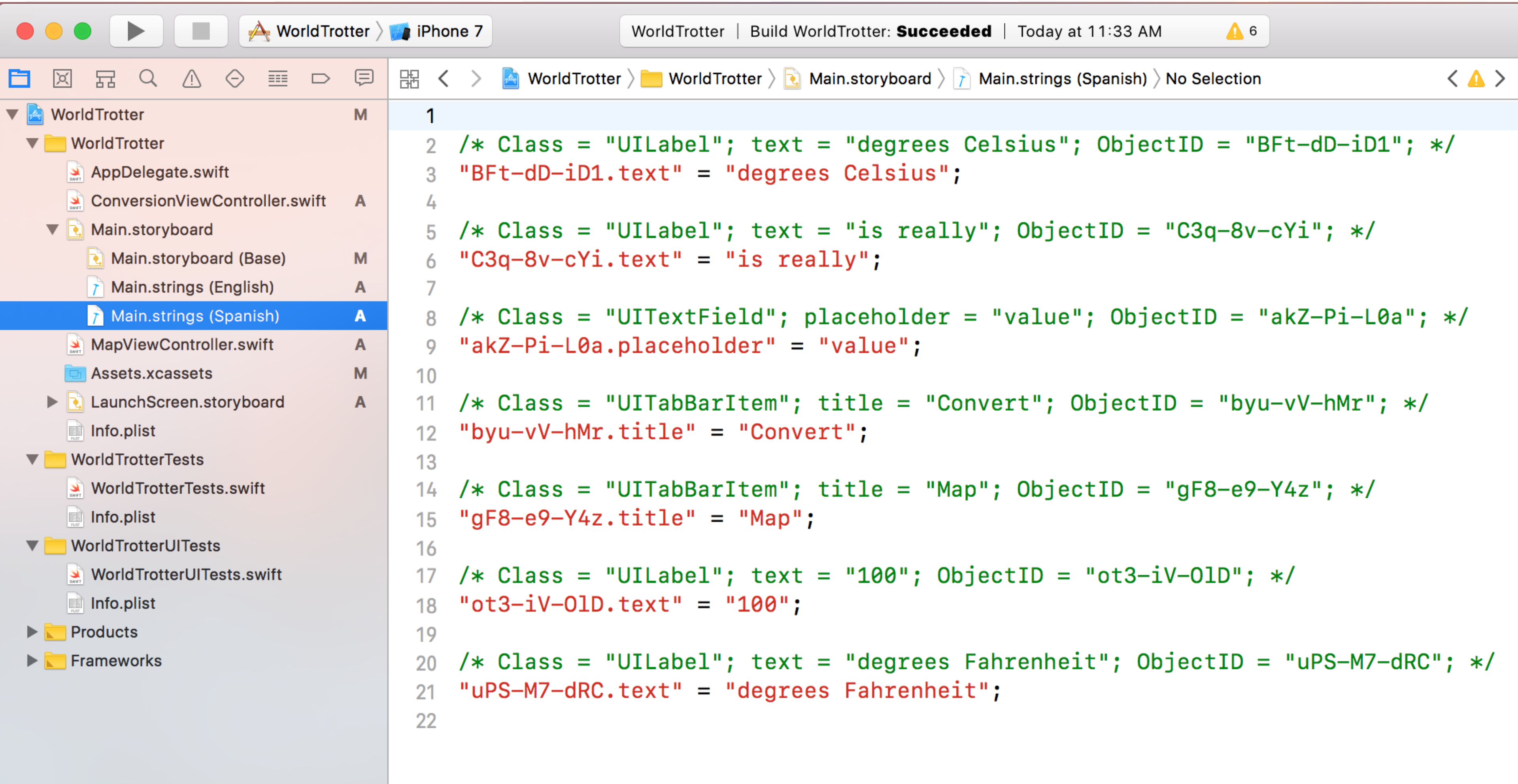
Localizations

Localization	Resources
Base	2 Files Localized
English — Development Language	1 File Localized
Spanish	1 File Localized

☑ Use Base Internationalization


- Main.storyboard
  - Main.storyboard (Base)
  - Main.strings (English)
  - Main.strings (Spanish)

Localizing the app



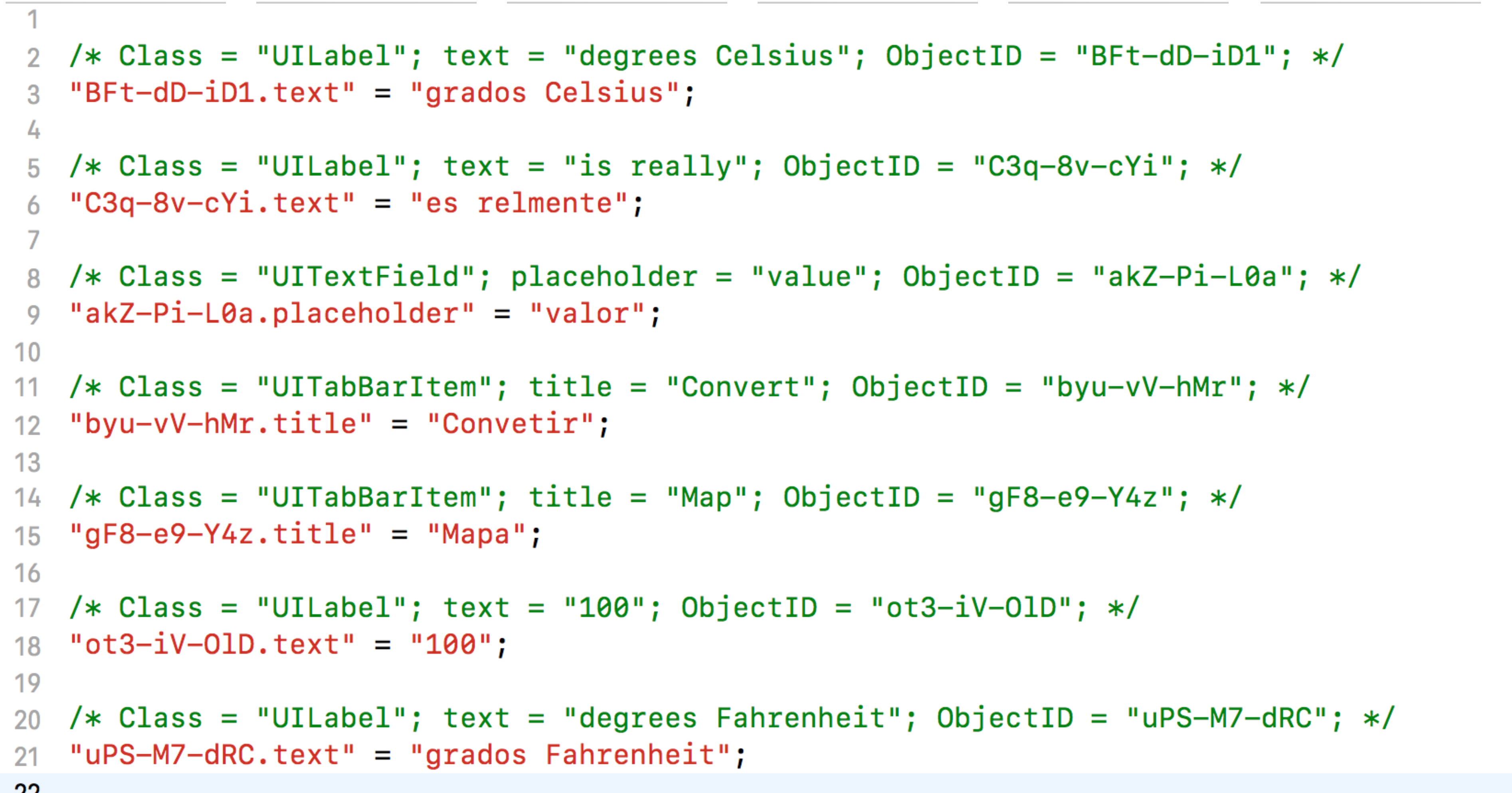
```

/* Class = "UILabel"; text = "degrees Celsius"; ObjectID = "BFt-dD-iD1"; */
"BFt-dD-iD1.text" = "degrees Celsius";

/* Class = "UILabel"; text = "is really"; ObjectID = "C3q-8v-cYi"; */
"C3q-8v-cYi.text" = "is really";

/* Class = "UITextField"; placeholder = "value"; ObjectID = "akZ-Pi-L0a"; */
"akZ-Pi-L0a.placeholder" = "value";

/* Class = "UITabBarItem"; title = "Convert"; ObjectID = "byu-vV-hMr"; */
"byu-vV-hMr.title" = "Convert";

/* Class = "UITabBarItem"; title = "Map"; ObjectID = "gF8-e9-Y4z"; */
"gF8-e9-Y4z.title" = "Map";

/* Class = "UILabel"; text = "100"; ObjectID = "ot3-iV-OlD"; */
"ot3-iV-OlD.text" = "100";

/* Class = "UILabel"; text = "degrees Fahrenheit"; ObjectID = "uPS-M7-dRC"; */
"uPS-M7-dRC.text" = "degrees Fahrenheit";

```

Localizing the app

You have to translate localized files yourself; Xcode is not that smart.

/* Class = "UITabBarItem"; title = "Map"; ObjectID = "6xh-o5-yRt"; */"6xh-o5-yRt.title" = "Map" **"Mapa";**
/* Class = "UILabel"; text = "degrees Celsius"; ObjectID = "7la-u7-mx6"; */"7la-u7-mx6.text" = "degrees Celsius" "**grados Celsius";**
/* Class = "UILabel"; text = "degrees Fahrenheit"; ObjectID = "Dic-rs-P0S"; */"Dic-rs-P0S.text" = "degrees Fahrenheit" "**grados Fahrenheit";**
/* Class = "UILabel"; text = "100"; ObjectID = "Eso-Wf-EyH"; */"Eso-Wf-EyH.text" = "100";
/* Class = "UITextField"; placeholder = "value"; ObjectID = "On4-jV-YlY"; */"On4-jV-YlY.placeholder" = "value" **"valor";**
/* Class = "UILabel"; text = "is really"; ObjectID = "wtF-xR-gbZ"; */"wtF-xR-gbZ.text" = "is really" **"es realmente";**
/* Class = "UITabBarItem"; title = "Convert"; ObjectID = "zLY-50-CeX"; */"zLY-50-CeX.title" = "Convert" **"Convertir";**

Localizing the app



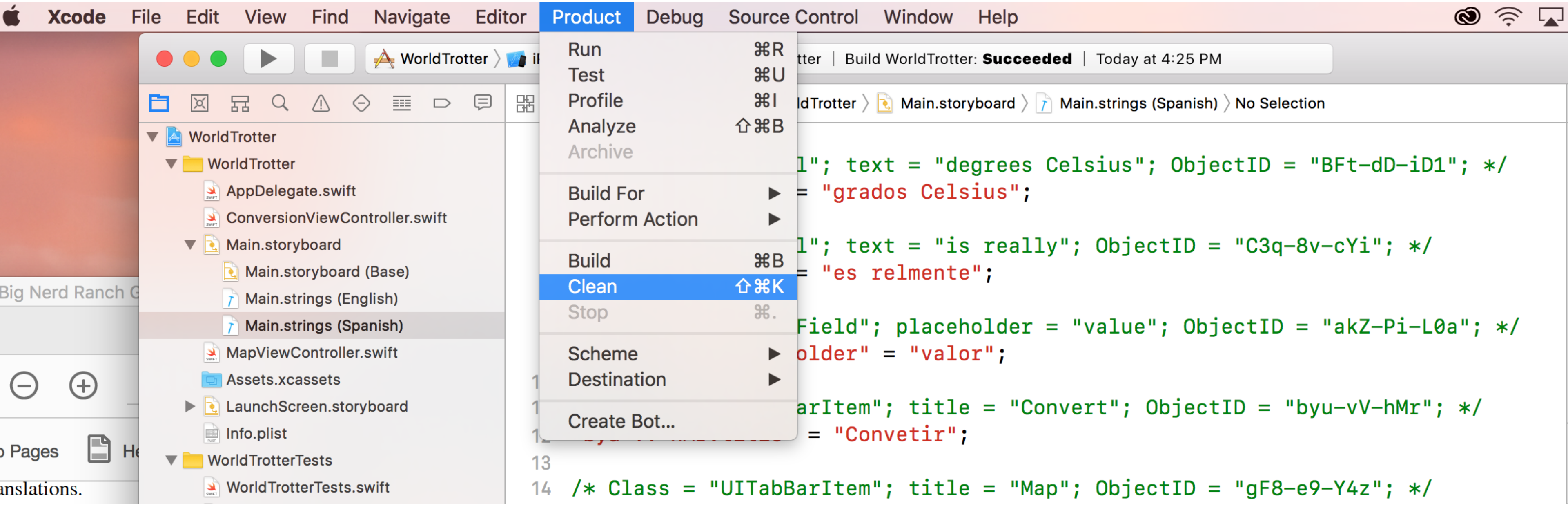
```

/* Class = "UILabel"; text = "degrees Celsius"; ObjectID = "BFt-dD-iD1"; */
"BFt-dD-iD1.text" = "grados Celsius";

/* Class = "UILabel"; text = "is really"; ObjectID = "C3q-8v-cYi"; */
"C3q-8v-cYi.text" = "es relmente";

/* Class = "UITextField"; placeholder = "value"; ObjectID = "akZ-Pi-L0a"; */
"akZ-Pi-L0a.placeholder" = "valor";

/* Class = "UITabBarItem"; title = "Convert"; ObjectID = "byu-vV-hMr"; */
"byu-vV-hMr.title" = "Convetir";

/* Class = "UITabBarItem"; title = "Map"; ObjectID = "gF8-e9-Y4z"; */
"gF8-e9-Y4z.title" = "Mapa";

/* Class = "UILabel"; text = "100"; ObjectID = "ot3-iV-OlD"; */
"ot3-iV-OlD.text" = "100";

/* Class = "UILabel"; text = "degrees Fahrenheit"; ObjectID = "uPS-M7-dRC"; */
"uPS-M7-dRC.text" = "grados Fahrenheit";
```

Localizing the app

1. Exit and restart Xcode to rebuild the app with new localization resources.

2. Product > Clean

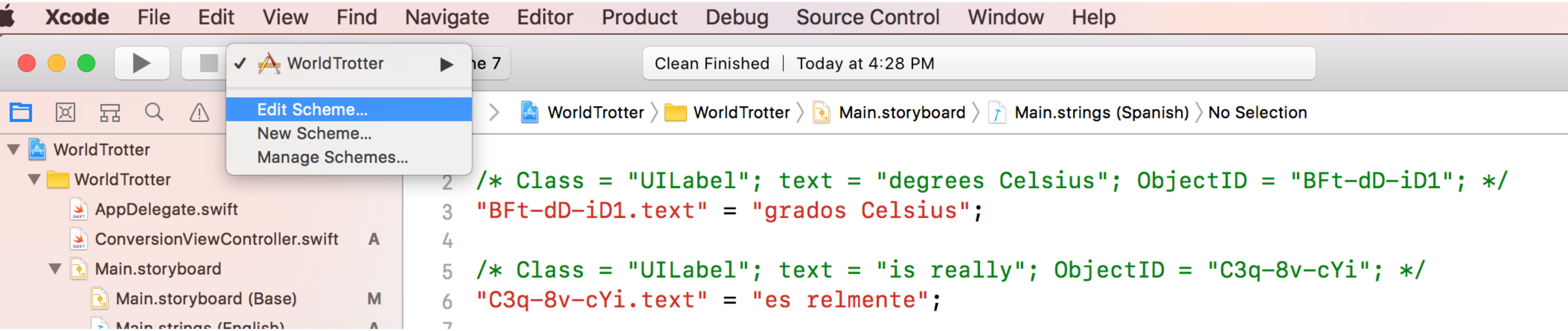
3. Press and hold the Option key while opening the Product menu and choose Clean Build Folder to entirely recompile, rebundle, and reinstall the app.



```
l"; text = "degrees Celsius"; ObjectID = "BFt-dD-iD1"; */
= "grados Celsius";

l"; text = "is really"; ObjectID = "C3q-8v-cYi"; */
= "es relmente";

Field"; placeholder = "value"; ObjectID = "akZ-Pi-L0a"; */
older" = "valor";

arItem"; title = "Convert"; ObjectID = "byu-vV-hMr"; */
= "Convetir";

/* Class = "UITabBarItem"; title = "Map"; ObjectID = "gF8-e9-Y4z"; */
```

Localizing the app

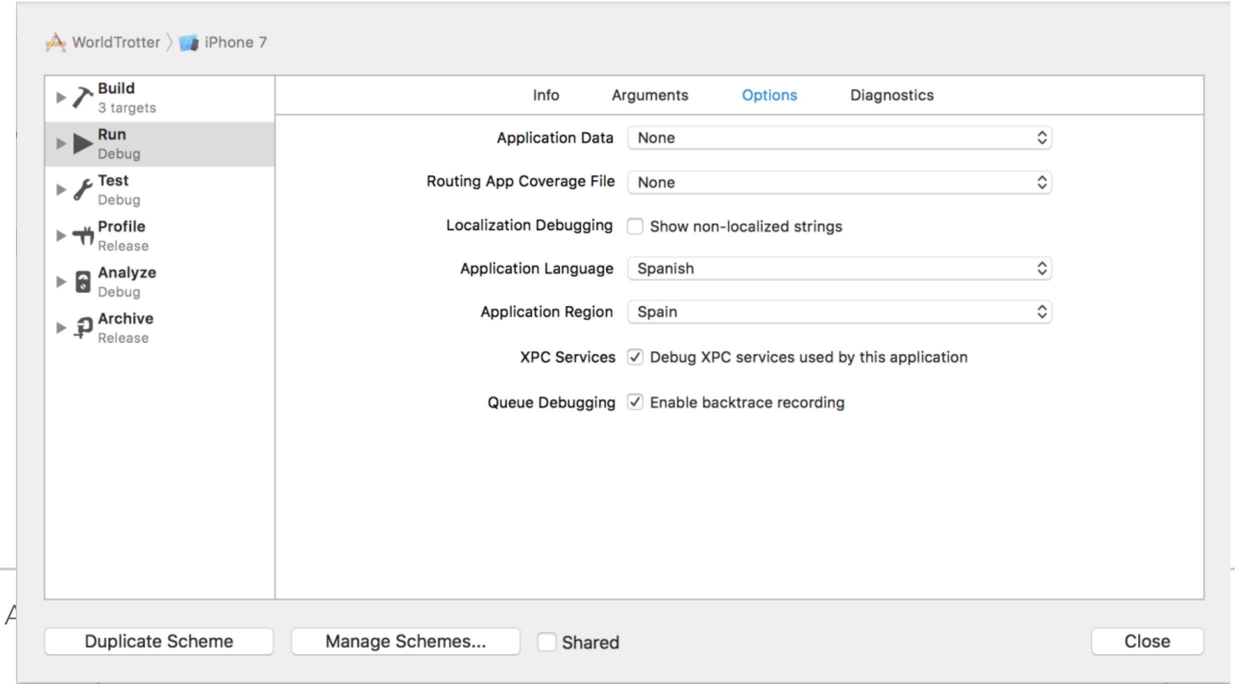
Open the active scheme pop-up and select Edit Scheme.
Make sure Run is selected on the lefthand side and open the Options tab.
Open the Application Language pop-up and select Spanish.
Spain is selected from the Application Region pop-up.


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
WorldTrotter
Edit Scheme...
New Scheme...
Manage Schemes...
Clean Finished | Today at 4:28 PM
WorldTrotter 〉 WorldTrotter 〉 Main.storyboard 〉 Main.strings (Spanish) 〉 No Selection
WorldTrotter
WorldTrotter
AppDelegate.swift
ConversionViewController.swift A
Main.storyboard
Main.storyboard (Base) M
Main.strings (English) A

```
/* Class = "UILabel"; text = "degrees Celsius"; ObjectID = "BFt-dD-iD1"; */
"BFt-dD-iD1.text" = "grados Celsius";

/* Class = "UILabel"; text = "is really"; ObjectID = "C3q-8v-cYi"; */
"C3q-8v-cYi.text" = "es relmente";
```

Localizing the app

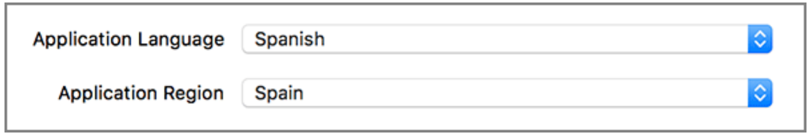
Open the active scheme pop-up and select Edit Scheme.
Make sure Run is selected on the lefthand side and open the Options tab.
Open the Application Language pop-up and select Spanish.
Spain is selected from the Application Region pop-up.



Application Data: None
Routing App Coverage File: None
Localization Debugging: ☐ Show non-localized strings
Application Language: Spanish
Application Region: Spain
XPC Services: ✓ Debug XPC services used by this application
Queue Debugging: ✓ Enable backtrace recording

Localizing the app

Open the active scheme pop-up and select Edit Scheme.
Make sure Run is selected on the lefthand side and open the Options tab.
Open the Application Language pop-up and select Spanish.
Spain is selected from the Application Region pop-up.



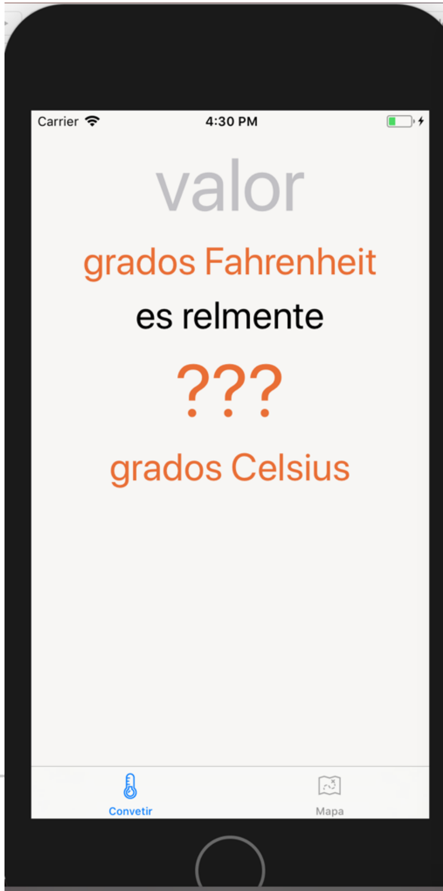
Application Language: Spanish
Application Region: Spain

Localizing the app

Open the active scheme pop-up and select Edit Scheme.
Make sure Run is selected on the lefthand side and open the Options tab.
Open the Application Language pop-up and select Spanish.
Spain is selected from the Application Region pop-up.

Build and run.

The constraints on the labels accommodate different lengths of text, and resize labels to fit.



Summary

Internationalization and localization are important for greatest public outreach.

Most of the apps are internationalized and localized for global market.

The app now
- converts between Celsius and Fahrenheit
- displays a map in a few different modules
- scales on all screen sizes
- is localized into another language

Assignment 2
- Internationalize and localize (In German) selected screen
- Include metric/ imperial conversion (TBD)