Localization

Expand the market for your app by supporting multiple languages and regions.

Overview

Localization is the process of translating and adapting your app into multiple languages and regions. Localize your app to provide access for users who speak a variety of languages, and who download from different App Store territories.

First, *internationalize* your code with APIs that automatically format and translate strings correctly for the language and region. Then add support for content that includes plural nouns and verbs by following language plural rules to increase the accuracy of your translations.

Translate and adapt your app

In Xcode, localization refers specifically to the set of resources for a specific language and region that you support.

You add a localization to your project and select the resources you want to include for that language and region. Export the localization and send the files to *localizers*, who translate the user-facing text and adapt resources for particular cultures and regions. Finally, you import the localized files and test the app in that language and region directly in Xcode.

When you release a localized version of your app, you can also localize the App Store information in App Store Connect for the specific territories where you offer your app.

For other localization tips, tools, and resources, see Expanding Your App to New Markets.

Important

In Xcode 15 and later, string catalogs are the recommended way to localize strings. In earlier versions of Xcode, use strings and `stringsdict` files. For more information about string catalogs, see Localizing and varying text with a string catalog.

Topics

Essentials

- Supporting multiple languages in your app

  Internationalize your app's strings, images, and other resource types to prepare for the translation process.

- Localizing and varying text with a string catalog

  Use a string catalog to translate text, handle plurals, and vary the text your app displays on specific devices.

- Using generated localizable symbols in your code

  Add keys directly to your string catalog that you can reference in your code using Xcode generated localizable symbols.

- {} Localizing Landmarks

  Add localizations to the Landmarks sample code project.

Strings and text

- Preparing your interface for localization

  Find text in your app that needs translation and verify that your interface adapts to translated text.

- Preparing your app's text for translation

  Make your app's text translatable by leveraging the localization APIs in the Foundation framework.

- Preparing dates, currencies, and numbers for translation

  Ensure that dates, currencies, and numbers display correctly across multiple languages and locales by using formatters.

Layouts and views

- Preparing views for localization

  Specify hints and add strings to localize your SwiftUI views.

- {} Autosizing views for localization in iOS

Add auto layout constraints to your app to achieve localizable views.

{} Localization-friendly layouts in macOS

This project demonstrates localization-friendly auto layout constraints.

Languages and regions

Adding support for languages and regions

Select the resources that you want to localize for each language and region you support.

Choosing localization regions and scripts

Add a language-only localization or localizations specific to regional variants and scripts.

Resources and assets

Adding resources to localizations

Include more resources in the localizations you add to your project.

Localizing assets in a catalog

Use asset catalogs to localize colors, images, symbols, watch complications, and more.

Translation and adaptation

Creating screenshots of your app for localizers

Share screenshots of your app with localizers to provide context for translation.

Exporting localizations

Provide the localizable files from your project to localizers.

Editing XLIFF and string catalog files

Translate or adapt the localizable files for a language and region that you export from your project.

Importing localizations

Import the files that you translate or adapt for a language and region into your project.

Locking views in storyboard and XIB files

Prevent changes to your Interface Builder files while localizing human-facing strings.

Testing

- Previewing localizations
  Test localizations in the SwiftUI preview or the Interface Builder preview.
- Testing localizations when running your app
  Run your app in each language and region you support to thoroughly test your app.

Legacy localization techniques

- Localizing strings that contain plurals
  Use a strings dictionary file to ensure correct localization of strings that contain language plurals.
- Creating width and device variants of strings
  Change a localized string for different interface widths and devices.

See Also

Interface

- Asset management
  Add app icons, images, strings, data files, machine learning models, and other resources to your projects, and manage how you load them at runtime.
- Accessibility Inspector
  Reveal how your app represents itself to people using accessibility features.