Article

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.

Overview

If you add keys to your string catalogs, Xcode can automatically generate symbols that you can use to reference the localizable strings in your code. For strings without format specifiers, Xcode creates static variables; for strings with format specifiers, Xcode creates functions with parameters.

First, you internationalize your app and use localizable strings in your code and interface files. Populate string catalogs when you build your targets and automatically generate comments to provide more context to translators. Then, optionally, generate localizable symbols to separate keys from their values so that you can iterate on your text without changing your code. You can use both of these approaches separately or in combination throughout your project.

For more information on creating string catalogs from localizable strings in your code, see Localizing and varying text with a string catalog.

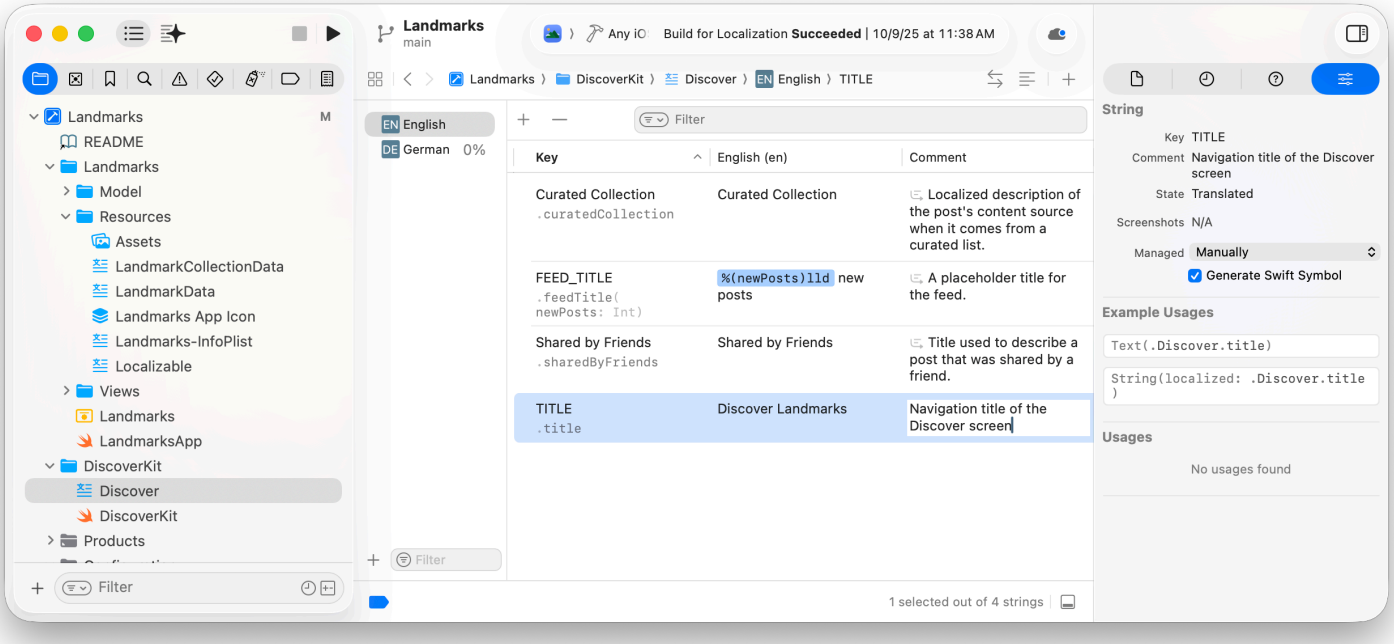
Enable automatic symbol generation

For older projects, you may need to enable Generate String Catalog Symbols in build settings. In the project editor, select the project in the sidebar and click Build Settings in the editor area. Enter “Generate String” in the filter field, and under Localization, set the Generate String Catalog Symbols setting to `Yes`.

Add keys to your string catalog

You can add localizable strings directly to your string catalog and then use symbols that Xcode generates for them in your code.

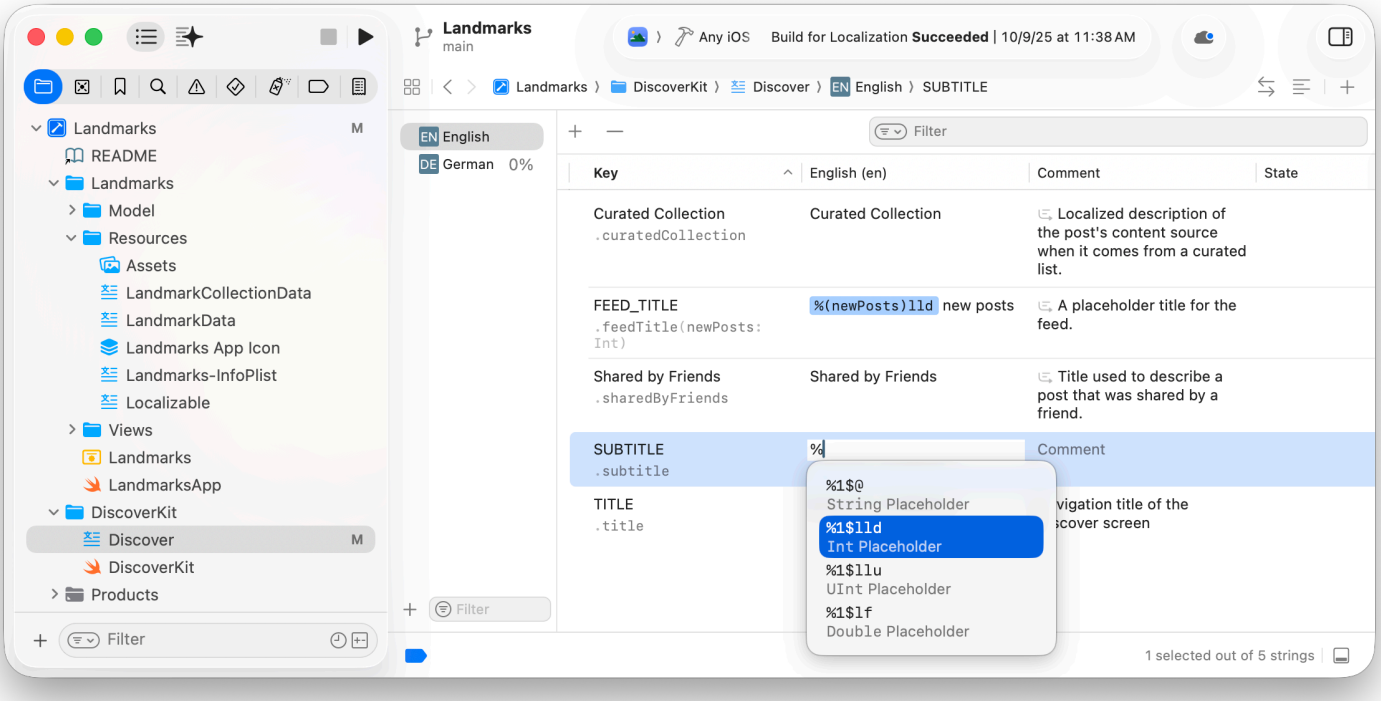
1. In the string catalog editor, select the source localization (English in this example) or another language in the sidebar.
2. Click the Add button (+) in the toolbar of the string catalog editor (on the far left of the filter field).
3. In the table, enter the key name, translation, and comment for translators.



For the other languages in the sidebar, Xcode shows a New icon in the State column.

Add localizable strings with variables

To add localizable strings with variables, enter the % character in the language column, and choose a string from the code completion menu that matches the type of the variable. For example, choose the format specifier for an integer or double placeholder from the menu. Then enter a name for the variable in the `variableName` placeholder text that Xcode highlights in the table. Press Return to finish typing and confirm the string.



You can add multiple variables of different types to the localizable string, but you can only use a variable name once.

Use generated localizable symbols in your code

Xcode generates symbols from the keys and translations that you enter in your string catalog so that you can access keys from your code. For localizable strings without variables, Xcode creates static properties, and for localizable strings containing format specifiers, Xcode creates functions with appropriate parameters.

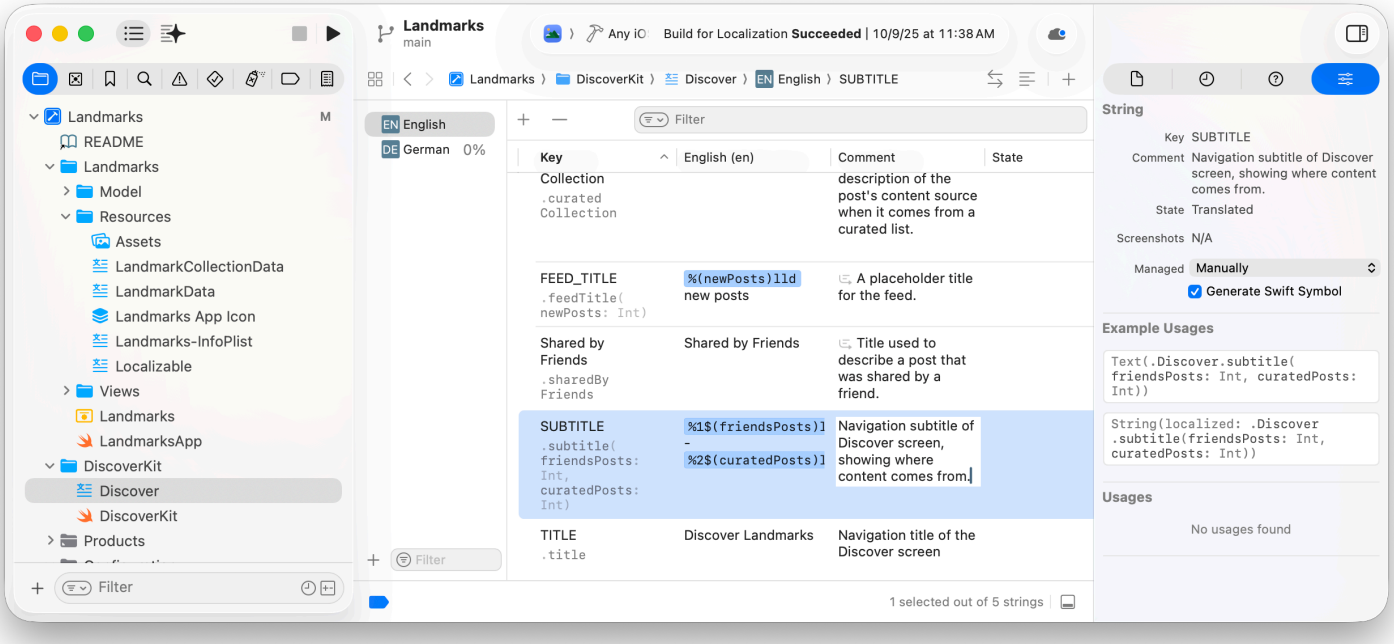
For keys in the default `Localizable` file, the symbol starts with a period (.) followed by the symbol name. Otherwise, it's followed by the table name, period, and then the symbol name, as in `.[table name].[symbol name]`.

For example, for a `TITLE` key without variables that you add to a `Discover` string catalog, Xcode creates a `.Discover.title` static property of type `LocalizedStringResource`. For a `SUBTITLE` key with `%1$(friendsPosts)lld – %2$(curatedPosts)lld` as the translation, Xcode creates a `.Discover.subtitle(friendsPosts: Int, curatedPosts: Int)` function that returns a `LocalizedStringResource`.

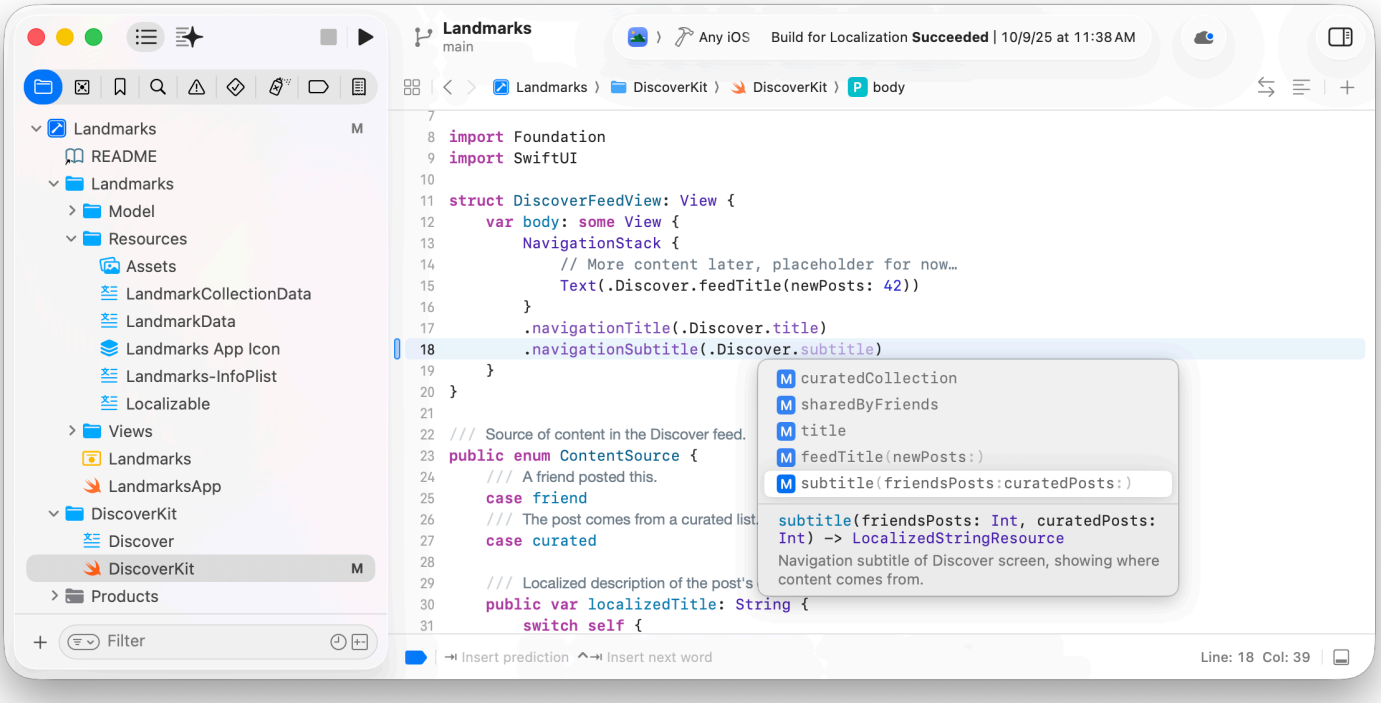
In SwiftUI, you can use the `LocalizedStringResource` type wherever you use localizable strings. Otherwise, you can use the generated static property or function in the `String` `init(localized:)` or `AttributedString` `init(localized:)` initializer.

First, become familiar with the style of generated localizable symbols and how to use them in your code. Select the key in the string catalog and open the Attributes inspector. If a Generate Swift

Symbol checkbox appears selected under String, then the string has a symbol. Code snippets for both SwiftUI and non-SwiftUI apps appear under Example Usages.



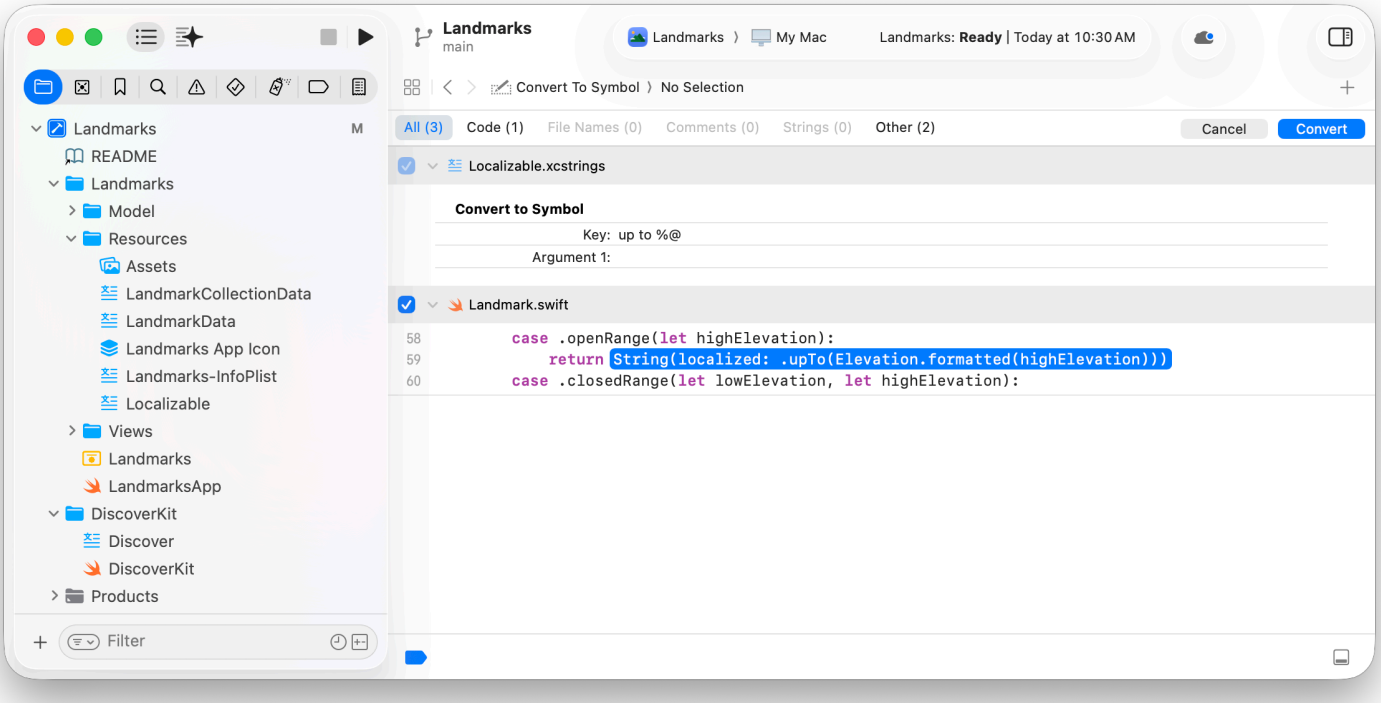
Then, use code completion to quickly enter these symbols while writing code in the source editor. Begin by entering a period followed by the key path. Then choose the symbol from the code completion menu that appears. For strings with format specifiers, replace the placeholder text with your variables.



Refactor code to use generated symbols

You can explicitly convert localizable strings in your code and interface files to generated symbols.

In the string catalog editor, select one or more keys in the table, Control-click a selected key, and choose Refactor > Convert Strings to Symbols. The generated symbol preview shows the changes to the key in the string catalog and the location where Xcode adds the symbol in the source code. To toggle between the current and modified code, click the highlighted code. To generate the symbol and apply the code changes, click Convert in the upper right corner.



Optionally, change the signature of the symbol by editing the localizable string that appears under Convert to Symbol. For example, change the signature to have a more semantic meaning or match your style if needed. For strings with format specifiers, you can also change the parameter names.

To undo your changes, select one or more keys and choose Convert Symbols to Strings from the pop-up menu. You can also switch existing generated symbols back to strings.

See Also

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.

Localizing Landmarks

Add localizations to the Landmarks sample code project.