Getting Started

Overview

Here's the condensed version of what the rest of this guide covers. Please see the online documentation for more tutorials and information:

1. Configure the Project
    1. Create a LocalizationManager in the scene.
    2. Create a display key for everything you wish to localize.
    3. Create a locale for every language you intend to support.
    4. (Optional) Setup the LocalizationManager to load additional translations from disk.
2. Localize the Project
    1. Provide localizations for display keys.
    2. Add locale-aware or language-sensitive MonoBehaviour scripts to GameObjects.
    3. Create a means for players to change the CurrentLocale.

Configure the Project

1) Create a new LocalizationManager and open TongueTwister

1. Right-click in the scene hierarchy.
2. Select “TongueTwister → Localizationmanager”
3. Select the new LocalizationManager game object.
4. In the inspector, click the button that says “Open in TongueTwister Window”

2) Create a new display key for every dialogue, label, menu item, etc.

1. Open the TongueTwister window if it isn’t open already.
2. Go to the “Editor” page.
3. Click the “Add Display Key” button.
4. Repeat this process for every label or UI element you’d like to eventually localize.

3) Create locales

1. Open the TongueTwister window if it isn’t open already.
2. Go to the “Locales” page.
3. Click the “Add New Locale” button.
4. Repeat this process for every locale (or language) you’d like to support in your project.

Localize the Project

1) Create a localization for every locale in every display key.

1. Open the TongueTwister window if it isn't open already.
2. Go to the "Editor" page
3. Select a display key in the tree view.
4. There are three ways to add a localization:
    a. Click the "Add Localization Button" from the tree view toolbar
    b. Right-click the display key and select "Add Localization"
    c. In the model view, scroll to the "Localizations" section and click "Add New Localization"
5. Create a localization for every configured locale.
    a. Set the localization's "locale" to one of the locales using the drop down in the model view.
    b. Set the localized text for the localization. This would be the translation for the given display key in respect to the selected locale.
6. Repeat this process until every display key has a localization for every configured locale.

*** Note**, *there is a tool on the Tools page of the TongueTwister window which will add an empty localization to every display key for each configured locale. This is very handy for quickly creating missing localizations.*

2) Make GameObjects "locale-aware"

When a GO is "locale-aware" or "language-sensitive", it changes some content to match the currently selected locale in the system. This is how you get real-time localization when users change the selected locale.

In this example, a game Object with a "UnityEngine.UI.Text" component would be given locale-aware behaviour by doing:

1. Select the GameObject with the Text component
2. In the inspector, add the "StandardLocalizableTextBehaviour" component
3. Ensure the TongueTwister window is open and a LocalizationManager has been selected
4. In the StandardLocalizationTextBehaviour component's section of the Inspector, choose a display key from the provided drop down.
5. (Optional) Select a Text component for the field. If one isn't provided, the system will attempt to find one with "GetComponent"

3) Allow users to change CurrentLocale

The easiest and quickest way to provide users with a means of changing locale is by using the Example Locale Dropdown prefab.

1. In the Project window of the Unity Editor, navigate to the directory: Assets/TongueTwister/Examples/Prefabs
2. Drag-and-drop the “ExampleLocaleDropdown” prefab into the scene.