Language Manager

Documentation

A language tag manager editor extension from TekkTech. Build by Tekk.

-Online How To-

Contents

Englisch

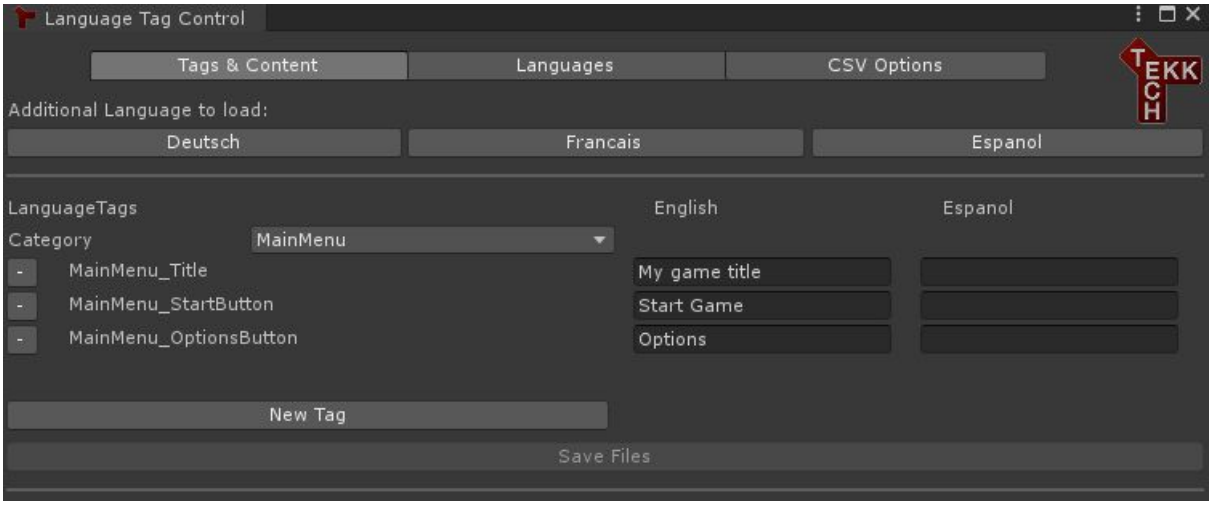
Introduction

The Language Manager allows you to create language tags for your project, so that you can access them at any time in the code and select them in text components, with the specially created MonoBehaviour, and you will always get the correct translation presented.

To open the editor window you have to click on **Tools/TekkTech/LanguageTagControl**.

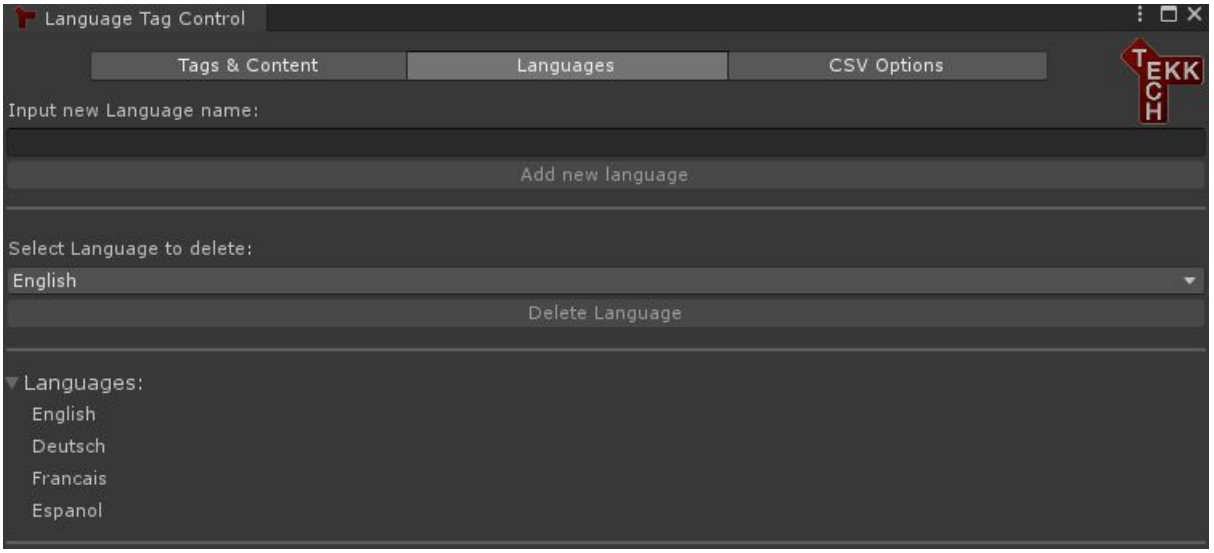
How To Use

Tags and Content



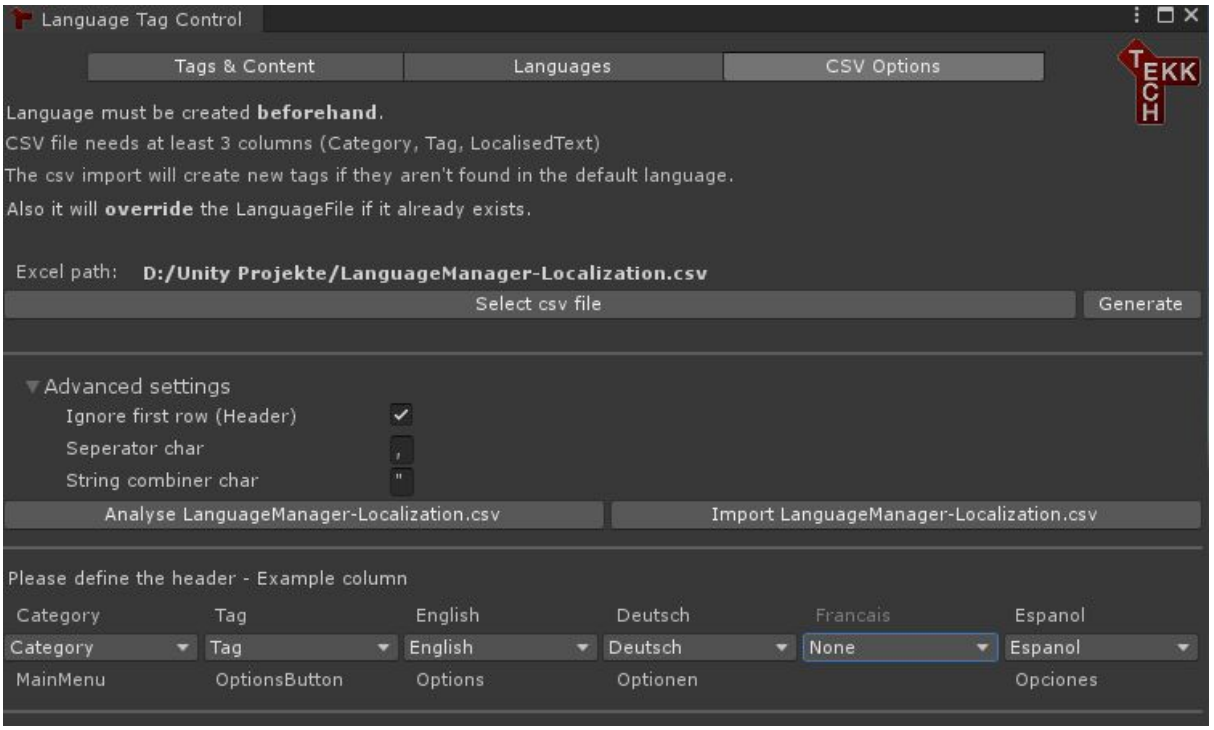
First, you can create new tags using the "New Tag" button. These are displayed in the list on the left. Using the drop-down menu, you can select the various categories that have been created. In the right area you can insert the localized text for each tag.
To save the tags and the text, press the "Save Files" button. This will create the tag inside the enum and refresh the assetdatabase. So you can use your tag inside the code and in LocalizedStrings in the inspector.

Languages



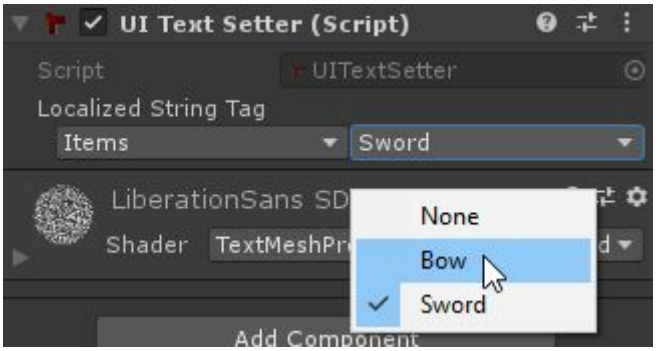
To support multiple languages, there is the possibility to create new ones via the text field in the "Language" tab. These will appear at the bottom as being added.
If an error has occurred, it is possible to select a language from the dropdown menu and then remove it.
Caution: This will also delete the internal localized file with all content in it.

CSV Options



In the third tab, "CSV Options", there is the option of importing a previously created CSV file (from Google Spreadsheets, MS Excel or Libre Office) or creating a fresh one from the existing tags and content entries.
For the generation, the separators stored in the "Advanced Settings" are used.
When analysing a CSV, the generated drop-down menus must be set to match the loaded CSV. At least the columns category, tags and a language must be assigned. When all required assignments are set, the "Import" button is selectable and the CSV is loaded and overwrites all tag contents if they already exist.

UITextSetter Utility

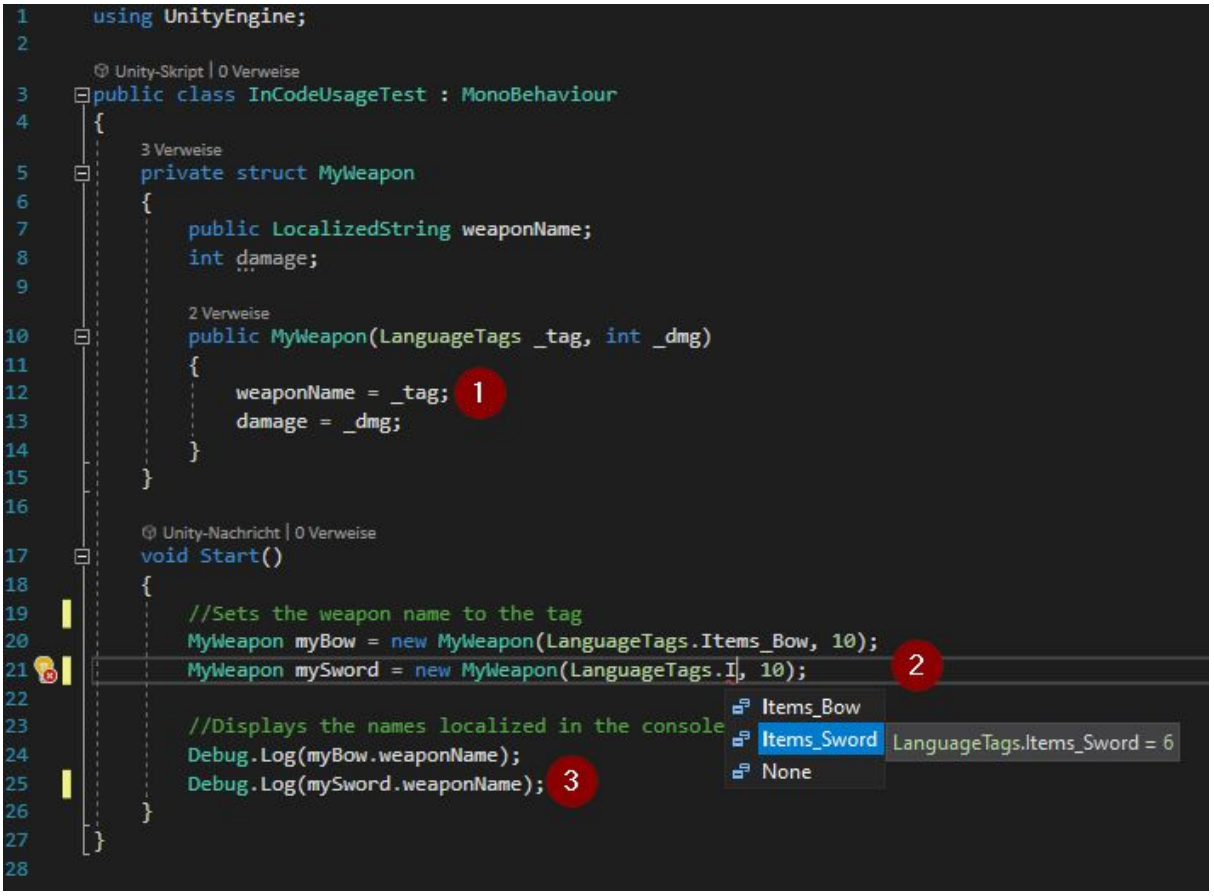


The UITextSetter script can be used for TMPro text fields that will be automatically localised. The corresponding tag can be set in the inspector.

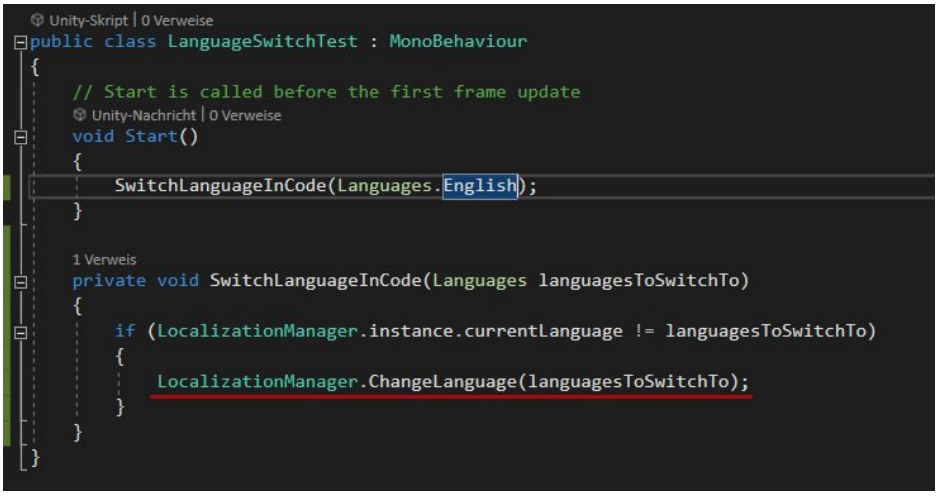
```
7   [RequireComponent(typeof(TextMeshProUGUI))]
    Unity-Skript | 0 Verweise
8   public class UITextSetter : MonoBehaviour
9   {
10      public LocalizedString LocalizedText;
11
12      private TextMeshProUGUI uiText;
13
        Unity-Nachricht | 0 Verweise
14      private void Start()
15      {
16          LocalizedText.SetAction(SetText);   (1)
17          uiText = this.GetComponent<TextMeshProUGUI>();
18          SetText();
19      }
20
        2 Verweise
21      public void SetText()
22      {
23          if(uiText is null)
24              uiText = this.GetComponent<TextMeshProUGUI>();
25
26          uiText.text = LocalizedText;   (2)
27      }
28  }
29
```

1. In the UISetter, the localised text is first given the UnityAction that is triggered by a LanguageSwitch. This is needed to be able to react in real time to a language change.
2. Here you can see how the TMP text field is set in the UnityAction method.

In Code Usage



1. A LanguageTag can be implicitly assigned to the localised string.
2. The tags can be selected from the generated enum "LanguageTags".
3. To access the localised content you can simply use it like a string. The currently set language is used automatically.



To switch the language in code, you only need to invoke the static method “ChangeLanguage” with the new language from the enum “Languages” on the LocalizationManager.

Deutsch

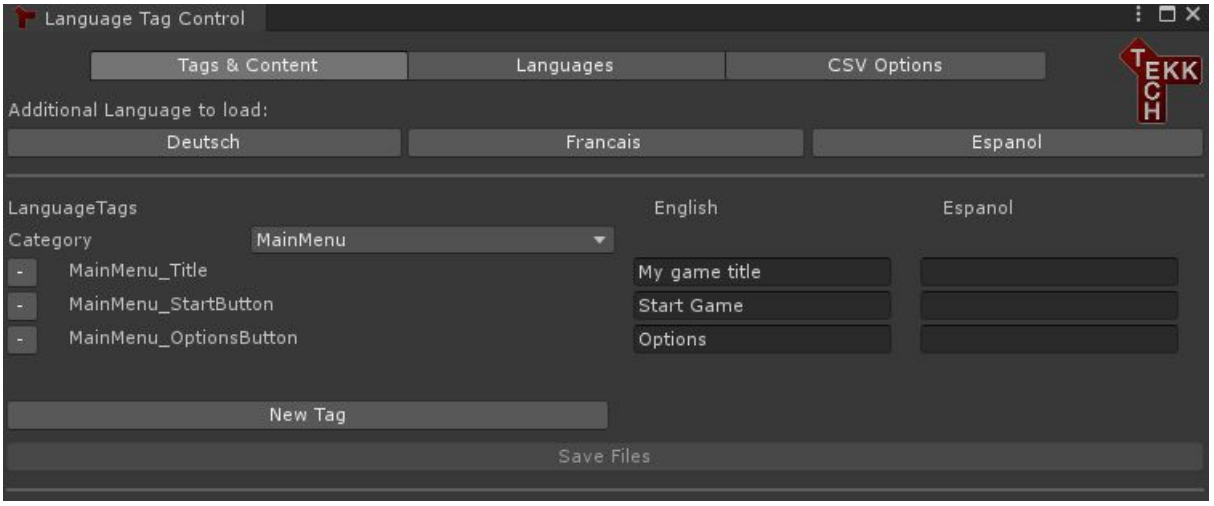
Einleitung

Der Language Manager erlaubt es Ihnen für ihr Projekt Sprach-Tags anzulegen, sodass Sie jederzeit im Code darauf zurückgreifen und in Text-Components, mit dem eigens dafür erstellten MonoBehaviour, diese auswählen können und sie dann immer die korrekte Übersetzung angezeigt bekommen.

Um das Editor Fenster zu öffnen, klickt man auf **Tools/TekkTech/LanguageTagControl**.

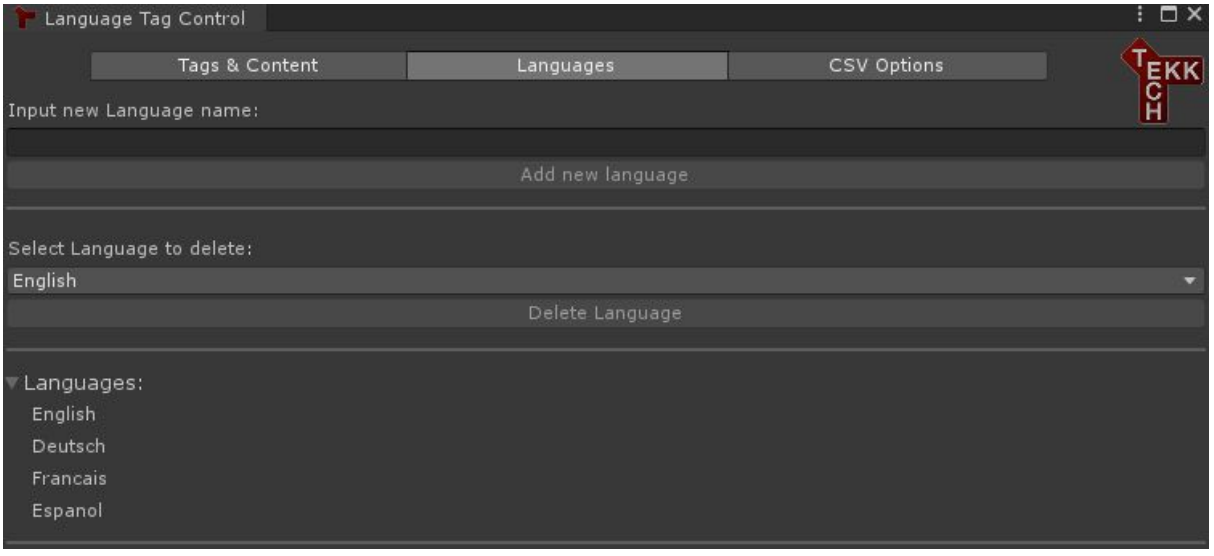
Verwendung

Tags und Inhalt



Als ersten kann man über den "New Tag"-Button neue tags anlegen. Diese werden in der Liste Links angezeigt. Über das Drop-Down Menü kann man die verschiedenen angelegten Kategorien auswählen.
Im rechten Bereich lässt sich zu jedem tag der lokalisierte Text einfügen. Damit die Tags und der Text gespeichert werden, muss auf den "Save Files"-Button gedrückt werden.

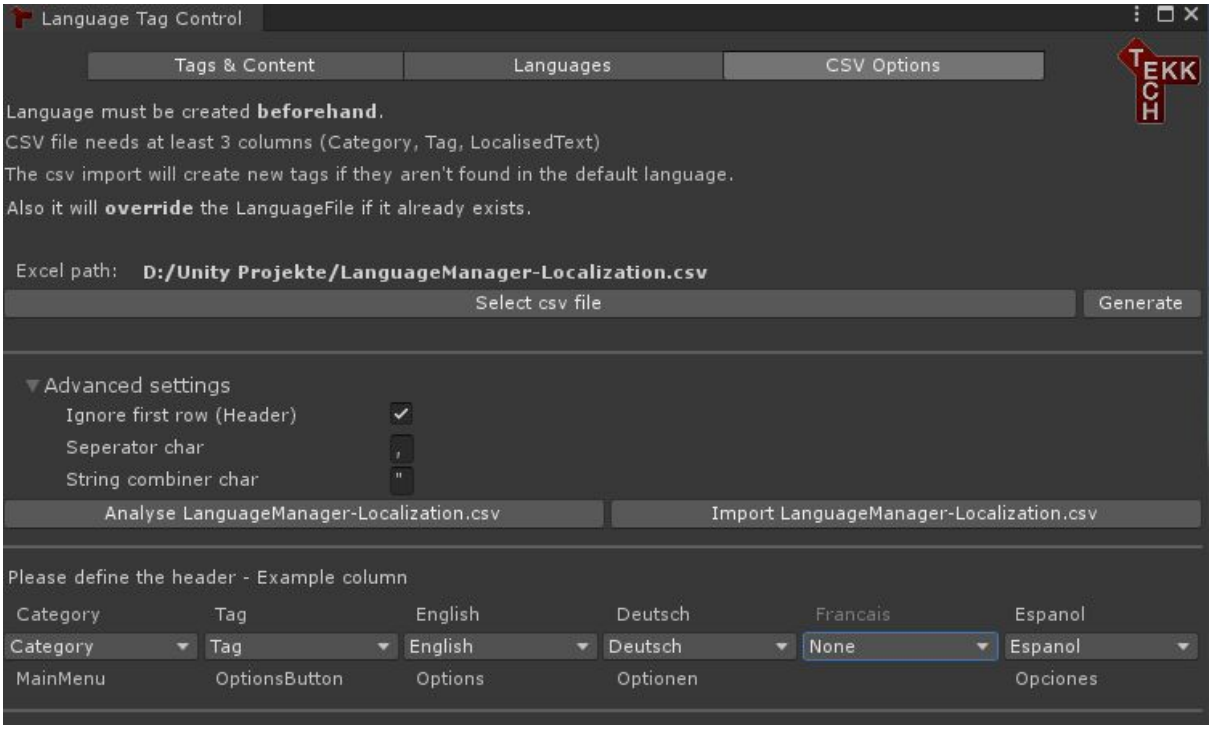
Sprachen



Um mehrere Sprachen zu unterstützen, gibt es im "Language" Reiter die Möglichkeit über das Textfeld welche anzulegen. Diese tauchen unten als hinzugefügt auf. Wenn ein Fehler passiert ist, ist es möglich eine Sprache aus dem Dropdown Menü auszuwählen und dann zu entfernen.

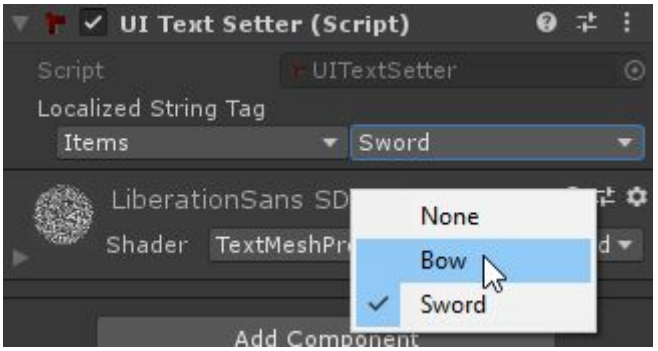
Achtung: Hierbei wird auch das interne lokalisierte File mit allen Inhalten dazu gelöscht.

CSV Optionen



Im dritten Reiter "CSV Options" gibt es die Möglichkeit eine vorher erstellte CSV Datei (aus Google Spreadsheets, MS Excel oder Libre Office etc) zu importieren oder eine frische aus den vorhandenen Tags und Content Einträgen zu erstellen. Für die Generierung werden die in den "Advanced Settings" hinterlegten Trennzeichen genutzt. Beim analysieren einer CSV muss man die generierten DropDown Menus so setzen wie es zu der geladenen CSV passt. Hierbei müssen mindestens die Kategorie, Tags und eine Sprache belegt werden.
Wenn alle erforderlichen Zuordnungen gesetzt sind, wird der "Import"-Button auswählbar und die CSV wird geladen und überschreibt alle Tag Inhalte, wenn diese bereits bestehen.

UITextSetter Utility

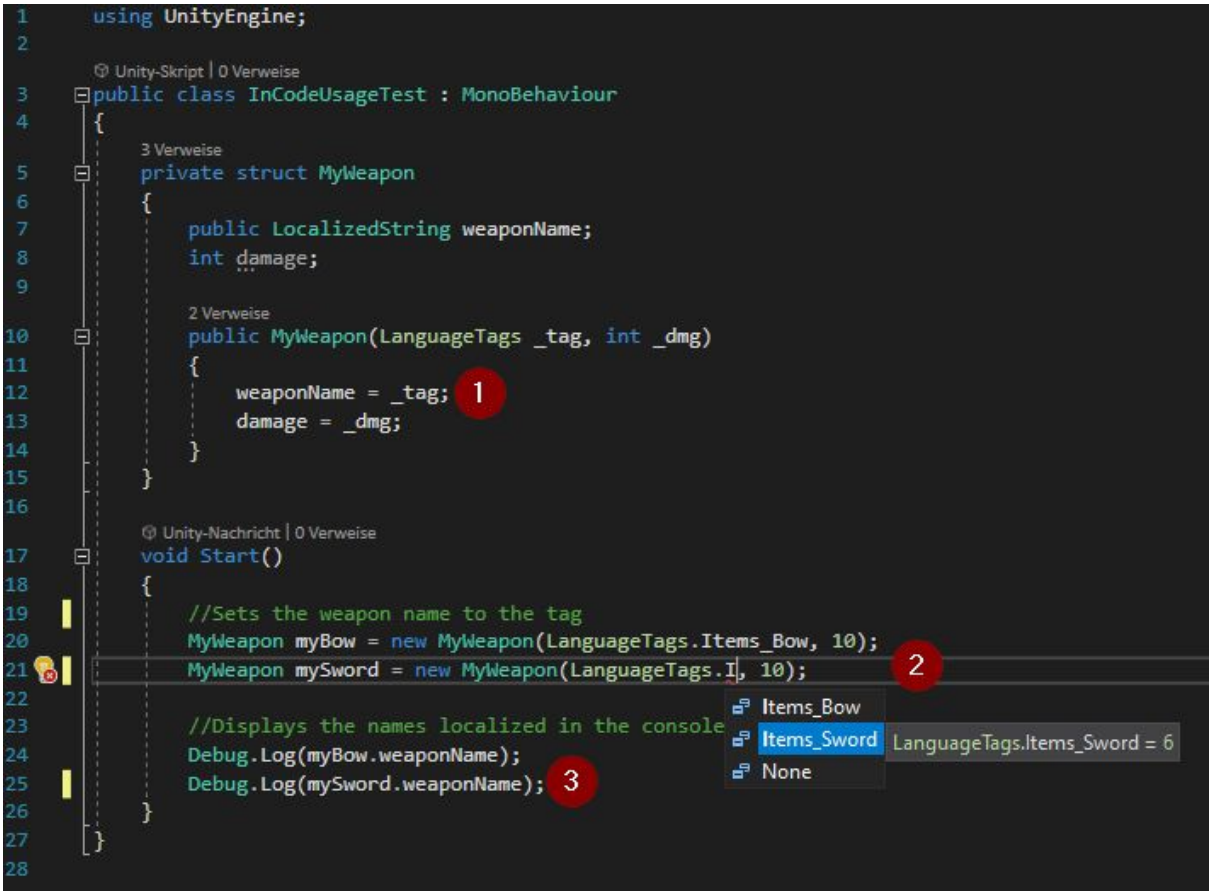


Der UITextSetter Skript kann für TMPro Text Felder genutzt werden, die automatisch lokalisiert werden sollen. Im Inspektor kann der zugehörige Tag gesetzt werden.

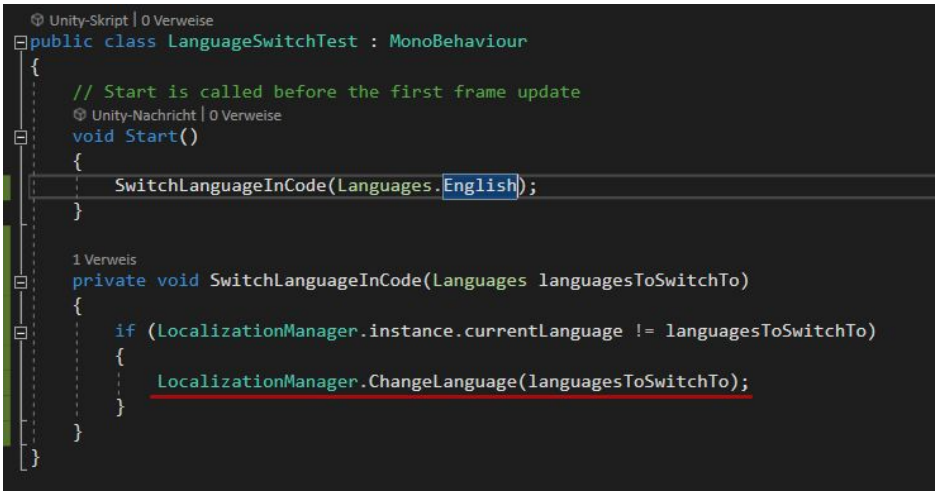
```
7     [RequireComponent(typeof(TextMeshProUGUI))]
      Unity-Skript | 0 Verweise
8     public class UITextSetter : MonoBehaviour
9     {
10        public LocalizedString LocalizedText;
11
12        private TextMeshProUGUI uiText;
13
          Unity-Nachricht | 0 Verweise
14        private void Start()
15        {
16            LocalizedText.SetAction(SetText);   1
17            uiText = this.GetComponent<TextMeshProUGUI>();
18            SetText();
19        }
20
          2 Verweise
21        public void SetText()
22        {
23            if(uiText is null)
24                uiText = this.GetComponent<TextMeshProUGUI>();
25
26            uiText.text = LocalizedText;   2
27        }
28    }
29
```

1. Im UISetter wird zunächst dem Lokalised Text die UnityAction mitgegeben, die bei einem LanguageSwitch ausgelöst wird. Das wird benötigt, um in realtime auf eine Sprachänderung reagieren zu können.
2. Hier kann man sehen, wie in der UnityAction Methode das TMP Textfeld gesetzt wird.

Nutzung im Code



1. Dem lokalisierte string kann implizit ein LanguageTag zugeordnet werden.
2. Die Tags können aus dem generierten Enum "LanguageTags" ausgewählt werden.
3. Um auf den lokalisierten Inhalt zuzugreifen, kann man ihn einfach wie einen String nutzen. Die aktuell eingestellte Sprache wird automatisch herangezogen.



Um die Sprache im Code umzuschalten, brauchen Sie nur die statische Methode "ChangeLanguage" mit der neuen Sprache aus dem Enum "Languages" auf dem LocalizationManager aufzurufen.