Localization

GameManager

The GameManager script is a crucial component that manages the game's state and the currently selected language. The GameManager ensures that there is only one instance of itself throughout the game, providing easy access to game-related functions and variables.

languageCode: This variable stores the currently selected language code (e.g., "en" for English). It is initialized with a default value of "en."

SetLanguageCode(string languageName): This method allows the game to switch between languages. It takes a languageName as input, converts it to a language code using the LanguageManager, and updates the languageCode variable. It also triggers a LanguageChanged event using the EventManager class to inform other components about the language change.

LanguageDropdownHandler

The LanguageDropdownHandler script is attached to a TextMesh Pro (TMP) dropdown, and is responsible for handling language selection by the player.

languageDropdown: This variable is a reference to the TMP dropdown component that displays a list of available languages.

Start(): This function ensures that the dropdown exists and is not null. If the dropdown is present, it clears any existing options and fetches the available language names from the LanguageManager. It then populates the dropdown options with these language names, providing players with an up-to-date list of language choices.

OnLanguageDropdownValueChanged(int index): This method is called when the player selects a different language from the dropdown. It takes the selected index as an argument and retrieves the selected language option from the dropdown. Then, it calls the SetLanguageCode method of the GameManager to change the game's language based on the player's selection.

LanguageManager

The LanguageManager script manages supported languages and their codes. It also provides methods to translate between language codes and language names.

languageDictionary: A dictionary that maps language codes (e.g., "en") to language names (e.g., "English").

Awake(): The script ensures that there is only one instance of LanguageManager, allowing it to persist between scenes. It also initializes the languageDictionary with language code and name pairs.

InitializeLanguageDictionary(): A method to populate the languageDictionary with language code and name pairs. You can add more languages as needed.

GetLanguageName(string languageCode): Retrieves the language name for a given language code.

GetLanguageCode(string languageName): Retrieves the language code for a given language name.

LocalizationManager

The LocalizationManager script manages the localization of the game. It loads translations from a CSV file and provides the ability to retrieve translated text based on the selected language.

translations: This dictionary stores translation data, where the keys are unique identifiers for text elements, and the values are dictionaries containing translations for different languages

csvFilePath: The path to the CSV file containing translation data.

Awake(): In the Awake method, the script ensures that there is only one instance of LocalizationManager, making it persist across scenes. It also loads translations from the CSV file using the LoadTranslations method.

LoadTranslations(): This method reads the translation data from the CSV file, parses it, and populates the translations dictionary with the appropriate translations for each language.

ParseCSVLine(string line): A private method for parsing a line of CSV data, handling quoted strings and commas within quotes.

GetTranslation(string key): Retrieves a translated text for a specific key based on the selected language code from the GameManager.

LocalizedText

The LocalizedText script is attached to TextMeshPro components and allows dynamic localization of text elements in the game.

translationKey: This variable specifies the key used to retrieve the localized text for the attached TextMeshPro component.

textMeshPro: A reference to the TextMeshPro component on the same GameObject.

Start(): In the Start method, the script retrieves the TextMeshPro component and calls the UpdateText method to set the initial text.

OnEnable() and OnDisable(): These methods subscribe and unsubscribe from the OnLanguageChanged event, allowing the text to be automatically updated when the selected language changes.

UpdateText(): This method updates the text content of the TextMeshPro component based on the selected language and translation key by calling LocalizationManager.Instance.GetTranslation(translationKey).

In summary, these scripts work together to enable language localization. The GameManager handles the game's state and language selection, while the LanguageDropdownHandler lets the player choose their preferred language. The LocalizationManager manages translation data and provides the translation for the specified language, and the LanguageManager is responsible for managing supported languages and their codes. Finally, the LocalizedText script allows dynamic localization of text elements.

How to add more languages:

If you want to add more languages, you should follow these two steps:

1. Update the csv file:
You should add a new column for each language you want to add. The column header should be in the format of “languageName(languageCode)”, without the quotes.
For example “French(fr)”.
Add translations for the newly added language(s).

2. Initialize Language Dictionary (LanguageManager)
Open the LanguageManager script and add new language code and name pairs to the languageDictionary. You can do this in the InitializeLanguageDictionary method.
For example:
languageDictionary["fr"] = "French";