Syncing String Table Collections

For each String Table Collection you want to connect to a Google Sheet, you need to add a Google Sheet extension to the String Table Collection's **Extensions list**. To do this, click the **Add(+)** button in the **Extensions** field. It is possible to add multiple extensions to a String Table Collection (for example, you might want to have a different sheet for each locale, and therefore a Google Sheet Extension for each sheet).

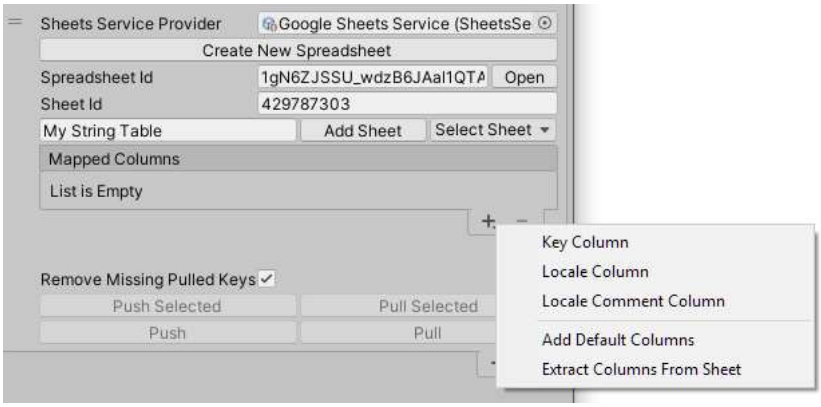
To sync a String Table to a Google Sheet, you need to connect it to a Sheets Service Provider asset. See Sheets Service Provider (Google-Sheets-Sheets-Service-Provider.html) for information on creating and configuring one.

Property	Description
Sheets Service Provider	Assign the Sheets Service Provider (Google-Sheets-Sheets-Service-Provider.html) Asset you want to use.
Create New Spreadsheet	Select this button to create a new spreadsheet for String Table data. Use this if a Google Sheet for this String Table does not already exist.
Spreadsheet Id	Enter the Spreadsheet ID from your Google Spreadsheet. In the Spreadsheet's Google URL, this is in the middle of the URL: https://docs.google.com/spreadsheets/d/**spreadsheetId**/edit#gid=sheetId
Sheet Id	Enter the Sheet ID from your Google Spreadsheet. In the Sheet's Google URL, this at the end of the URL: https://docs.google.com/spreadsheets/d/spreadsheetId/edit#gid=**sheetId**
Add Sheet	Create a new sheet based on the properties defined in the associated Sheets Service Provider's New Sheet Properties.
Select Sheet	Select a sheet from the Google Spreadsheet.
Mapped Columns	Assign specific types of data to specific columns. See **Mapped Columns**, below, for details.
Remove Missing Pulled Keys	Remove any keys that are not present in a sheet after a pull.
Push Selected	Push selected mapped columns to the Google sheet.
Pull Selected	Pull selected mapped columns from the Google sheet.
Push	Push all mapped columns to the Google sheet.
Pull	Pull all mapped columns from the Google sheet.

Mapped Columns

Use the **Mapped Columns** field to configure how your project pushes and pulls data from the spreadsheet. A mapped column can read and write to a single column's value and note property.

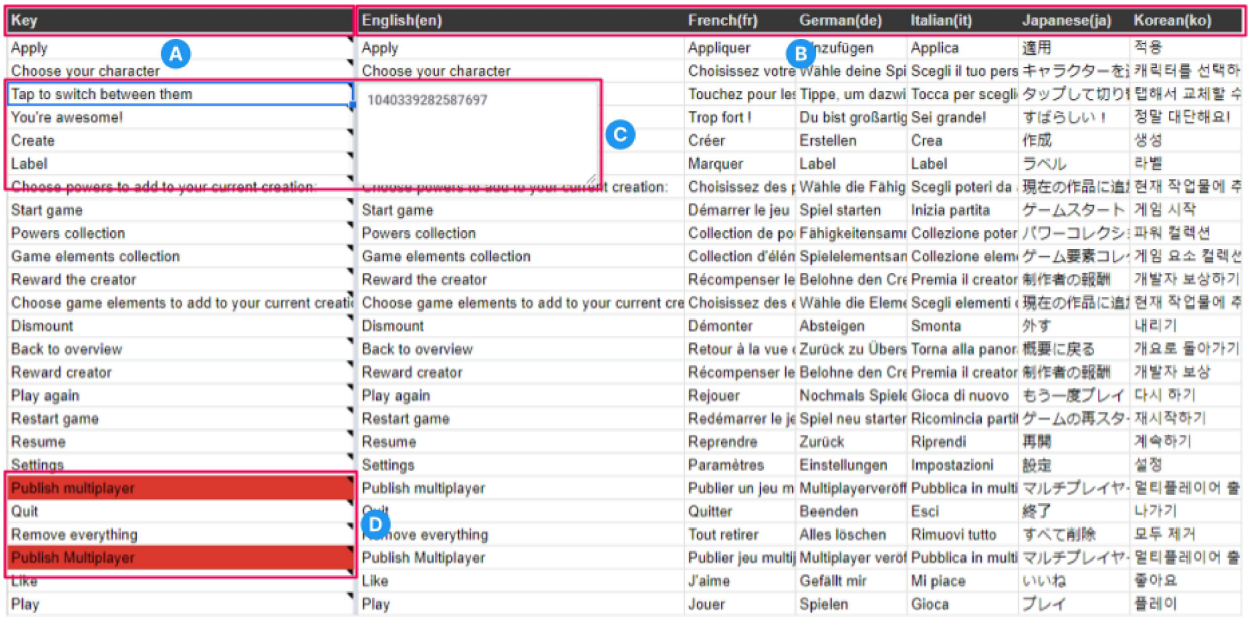
To add a new column mapping, select the **Add(+)** button.



Option	Description
Key Column	A **Key Column** includes the Key name in the value field and the Key Id in the note field. This means that it is possible to rename a key and even assign a custom Id value. 
Locale Column	The **Locale Column** includes the values from a single String Table for the selected Locale. If **Include Comments** is enabled, any comment metadata is included in the Sheet as notes. 
Locale Comment Column	The **Locale Comment Column** includes comment metadata as values. 
Add Default Columns	This adds a **Key Column** and a **Locale Column** for each Locale in the project.
Extract Columns From Sheet	If the Spreadsheet already contains locale data, this option reads the column titles and creates column mappings. This option requires OAuth authorization (Google-Sheets-Sheets-Service-Provider.html#oauth-authentication). It recognizes the following names: **Key Column** - “Key” - “Keys” **Locale Column** - Locale asset name - Locale identifier ToString - Locale identifier code **Locale Comment Column** - The same as the Locale Column appended by “Comments”.

NOTE
The Mapped Columns must include one Key Column or derived Key Column. Unity needs this key so that it can associate the sheet rows to the corresponding keys in the String Table Collection.

Example Google Sheet



Key	English(en)	French(fr)	German(de)	Italian(it)	Japanese(ja)	Korean(ko)
Apply	Apply	Appliquer	nzufügen	Applica	適用	적용
Choose your character	Choose your character	Choisissez votre	Wähle deine Spi	Scegli il tuo pers	キャラクターを	캐릭터를 선택하
Tap to switch between them	1040339282587697	Touchez pour les	Tippe, um dazwi	Tocca per scegli	タップして切り	탭해서 교체할 수
You're awesome!		Trop fort !	Du bist großartig	Sei grande!	すばらしい！	정말 대단해요!
Create		Créer	Erstellen	Crea	作成	생성
Label		Marquer	Label	Label	ラベル	라벨
Choose powers to add to your current creation:	t creation:	Choisissez des	Wähle die Fähig	Scegli poteri da	現在の作品に追	현재 작업물에 추
Start game	Start game	Démarrer le jeu	Spiel starten	Inizia partita	ゲームスタート	게임 시작
Powers collection	Powers collection	Collection de po	Fähigkeitensamı	Collezione poter	パワーコレクシ	파워 컬렉션
Game elements collection	Game elements collection	Collection d'élén	Spielelementsan	Collezione elem	ゲーム要素コレ	게임 요소 컬렉션
Reward the creator	Reward the creator	Récompenser le	Belohne den Cre	Premia il creator	制作者の報酬	개발자 보상하기
Choose game elements to add to your current creati	Choose game elements to add to your current cre	Choisissez des	Wähle die Eleme	Scegli elementi	現在の作品に追	현재 작업물에 추
Dismount	Dismount	Démonter	Absteigen	Smonta	外す	내리기
Back to overview	Back to overview	Retour à la vue	Zurück zu Übers	Torna alla panor	概要に戻る	개요로 돌아가기
Reward creator	Reward creator	Récompenser le	Belohne den Cre	Premia il creator	制作者の報酬	개발자 보상
Play again	Play again	Rejouer	Nochmals Spiele	Gioca di nuovo	もう一度プレイ	다시 하기
Restart game	Restart game	Redémarrer le je	Spiel neu starter	Ricomincia partit	ゲームの再スタ	재시작하기
Resume	Resume	Reprendre	Zurück	Riprendi	再開	계속하기
Settings	Settings	Paramètres	Einstellungen	Impostazioni	設定	설정
Publish multiplayer	Publish multiplayer	Publier un jeu m	Multiplayerveröff	Pubblica in multi	マルチプレイヤ	멀티플레이어 출
Quit	Quit	Quitter	Beenden	Esci	終了	나가기
Remove everything	Remove everything	Tout retirer	Alles löschen	Rimuovi tutto	すべて削除	모두 제거
Publish Multiplayer	Publish Multiplayer	Publier jeu multi	Multiplayer veröf	Pubblica in multi	マルチプレイヤ	멀티플레이어 출
Like	Like	J'aime	Gefällt mir	Mi piace	いいね	좋아요
Play	Play	Jouer	Spielen	Gioca	プレイ	플레이

A: The **Key column** includes the unique Key for each entry.
B: The **Locale columns**. Each Locale has a separate column for its localized values.
C: The **Key Id** is stored in the **note field** of each Key cell. When adding a new entry this field is empty and is populated when the next sync occurs.

> **TIP**
> You can assign an ID value to a new entry instead of using the auto-generated ID. To do this, right-click the field and select **insert note**. Custom IDs should use negative values to prevent future conflicts, see Table Keys (TableEntryKeys.html) for further information.

D: Duplicate keys are highlighted to indicate a possible issue. You should either combine or rename duplicates to prevent possible conflicts.

Custom columns

It is possible to add support for pushing and pulling custom data.

To create a custom column, you need to inherit from the abstract class SheetColumn (../api/UnityEditor.Localization.Plugins.Google.Columns.SheetColumn.html). A common use for a custom column is to support additional metadata. In this case, you can use the abstract classes KeyMetadataColumn (../api/UnityEditor.Localization.Plugins.Google.Columns.KeyMetadataColumn-1.html) and LocaleMetadataColumn (../api/UnityEditor.Localization.Plugins.Google.Columns.LocaleMetadataColumn-1.html) to simplify the process.

The following example demonstrates how to populate custom metadata into a column.

```
[Serializable]
[DisplayName("Custom Data")]
[Metadata(AllowedTypes = MetadataType.StringTableEntry)]
public class MyCustomDataMetadata : IMetadata
{
    public string someValue;
    public string someNoteValue;
}

/// <summary>
/// LocaleMetadataColumn is a version of SheetColumn only used for handling Metadata.
/// This can now be added to the Column Mappings for any Push or Pull request.
/// </summary>
public class MyCustomColumn : LocaleMetadataColumn<MyCustomDataMetadata>
{
    public override PushFields PushFields => PushFields.ValueAndNote; // For our example we use both value and note.

    public override void PullMetadata(StringTableEntry entry, MyCustomDataMetadata metadata, string cellValue, string cellNote)
    {
        // Metadata will be null if the entry does not already contain any.
        if (metadata == null)
        {
            metadata = new MyCustomDataMetadata();
            entry.AddMetadata(metadata);
        }

        metadata.someValue = cellValue;
        metadata.someNoteValue = cellNote;
    }

    public override void PushHeader(StringTableCollection collection, out string header, out string headerNote)
    {
        // The title of the Google Sheet column
        header = "My Custom Data";
        headerNote = null;
    }

    public override void PushMetadata(MyCustomDataMetadata metadata, out string value, out string note)
    {
        // Metadata will never be null as this is only called if the entry contains a metadata entry.
        value = metadata.someValue;
        note = metadata.someNoteValue;
    }
}
```

Synchronizing

The **Push** option sends all data from the String Table Collection to the chosen sheet in the Google Sheet. **Push Selected** does the same, but it only sends the selected column and the **Key** column.

The **Pull** option downloads the mapped columns and updates the String Table Collection. Pull Selected does the same as **Pull**, but only for the selected mapped column and the **Key** column.

If **Remove Missing Pulled Keys** is enabled, any keys that are not in the Spreadsheet during a Pull are removed from the String Table Collection.

The script API GoogleSheets (../api/UnityEditor.Localization.Plugins.Google.GoogleSheets.html) allows you to **Push** and **Pull** from custom code.
