String Tables

Overview

String Tables provide a way to centralize your localized text into one (or several) known locations, and then reference the entries within a string table from other assets or code in a robust way that allows for easy re-use of localized text.

String Tables can be defined in C++, loaded via CSV file, or created as an asset.

Creating a String Table

C++

String Tables can be created in C++ by using the `LOCTABLE` family of macros:

- `LOCTABLE_NEW` - Creates an empty string table instance.
- `LOCTABLE_FROMFILE_X` - Creates a string table instance and populates it from the given CSV file. When using the `_ENGINE` variant, the file is relative to the Engine content directory, and when using the `_GAME` variant, the file is relative to the Game content directory.
- `LOCTABLE_SETSTRING` - Sets an entry in a string table.
- `LOCTABLE_SETMETA` - Sets optional meta-data for an entry in a string table.

```
// Create and populate a string table using only C++
LOCTABLE_NEW("CodeStringTable", "CodeStringTable");

LOCTABLE_SETSTRING("CodeStringTable", "HelloWorld", "Hello
World!");
LOCTABLE_SETMETA("CodeStringTable", "HelloWorld", "Comment", "This
is a comment about hello world");

LOCTABLE_SETSTRING("CodeStringTable", "GoodbyeWorld", "Goodbye
World!");
LOCTABLE_SETMETA("CodeStringTable", "GoodbyeWorld", "Comment",
"This is a comment about goodbye world");

// Create and populate a string table from a CSV file
LOCTABLE_FROMFILE_GAME("CSVStringTable", "CSVStringTable",
"StringTableCSV/TestStrings.csv");
```

CSV

String Tables defined in CSV can be loaded using the `LOCTABLE_FROMFILE_X` macros, or can also be imported to/exported from a String Table asset.

The CSV format used for string tables has two mandatory columns, “Key” and “SourceString”, with any extra columns being parsed as meta-data for your string table entries.

```
Key,SourceString,Comment
"HelloWorld","Hello World!","This is a comment about hello world"
"GoodbyeWorld","Goodbye World!","This is a comment about goodbye
world"
```

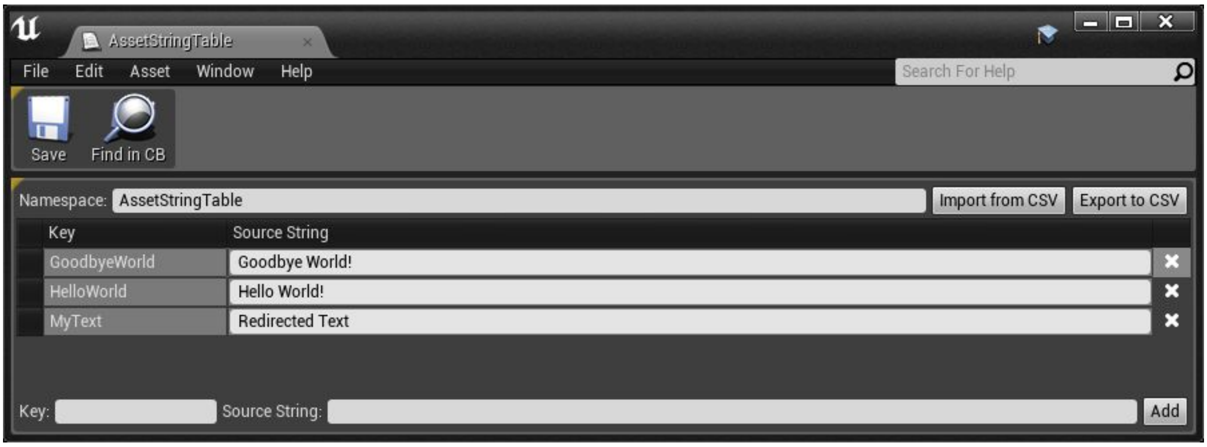
Note: Quotes in CSV files are escaped using two sets of quotes (`""`) rather than C-style escaping (`\"`), however other control characters in your string (including tabs and new-lines) should be escaped using C-style escaping.

Note: CSV string tables aren’t staged automatically. It is recommended that you place your CSV string tables in a known folder and then add them to your “Additional Non-Asset Directories to Package” in your project “Packaging” settings.

Note: CSV string tables loaded using the `LOCTABLE_FROMFILE_X` macros will be watched for changes and automatically re-imported if they change on disk (editor builds only).

Assets

String Table assets can be created via the Content Browser (“Miscellaneous” -> “String Table”), and can be managed through the String Table editor.



Namespace: AssetStringTable

Key	Source String
GoodbyeWorld	Goodbye World!
HelloWorld	Hello World!
MyText	Redirected Text

Key:
Source String:

Note: String Table assets are binary, so take care to avoid causing merge conflicts with them. It is recommended that you keep them more granular than when using C++ or CSV to minimize your conflict domain.

Note: The String Table editor cannot currently edit meta-data, however any meta-data imported from a CSV file is preserved.

Referencing a String Table

C++

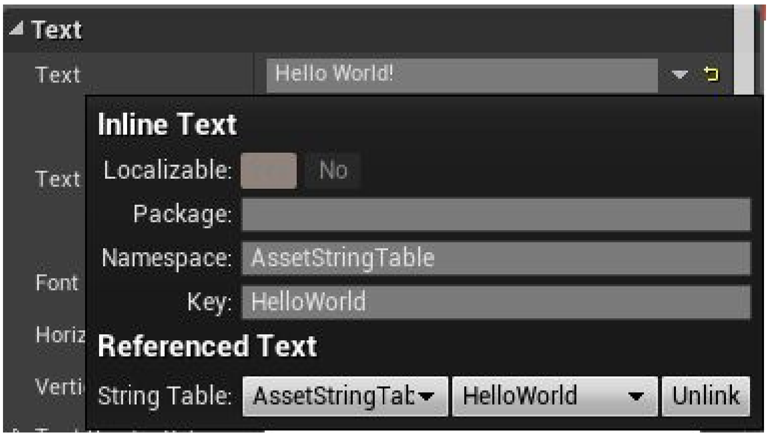
String Tables can be referenced in C++ using either the `LOCTABLE` macro, or the static `FText::FromStringTable` function. The underlying logic is identical, although the macro is easier to type but will only work with literal values, whereas the function will work with both literal and variable arguments.

INI

String Tables can be referenced in INI files using the `LOCTABLE` macro syntax.

Assets

String Tables can be referenced via `FText` properties. Opening the advanced combo for a text property allows you to link that property instance to a string table entry, rather than define your localized text inline.



Text: Hello World!

Inline Text

Localizable: No

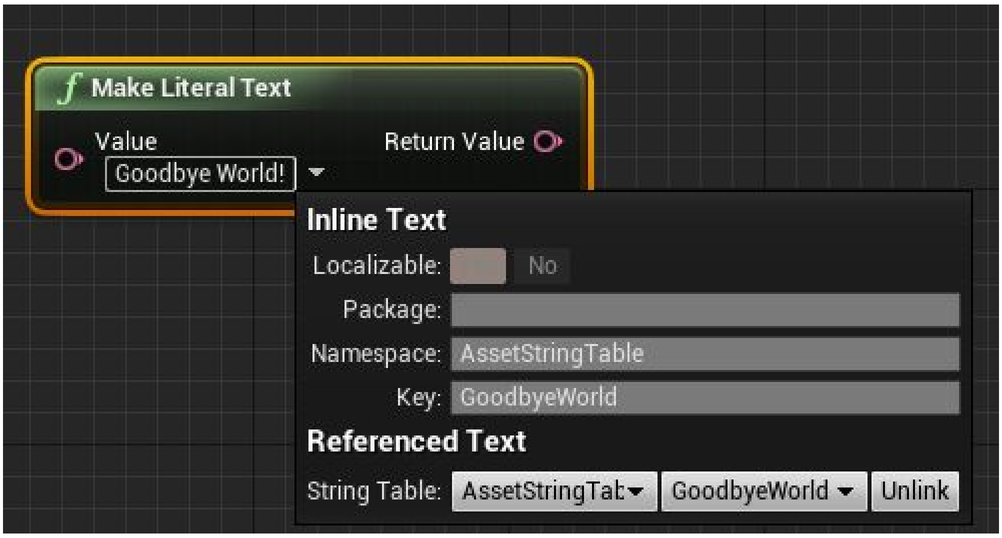
Package:

Namespace: AssetStringTable

Key: HelloWorld

Referenced Text

String Table: AssetStringTab HelloWorld Unlink



Make Literal Text

Value: Goodbye World!

Return Value

Inline Text

Localizable: No

Package:

Namespace: AssetStringTable

Key: GoodbyeWorld

Referenced Text

String Table: AssetStringTab GoodbyeWorld Unlink

Redirecting a String Table

We generally recommend that you avoid renaming or moving string tables where possible, however should you need to, there is a way to redirect both whole tables, and individual keys.

INI

String Tables can be redirected via an INI file.

The following will redirect an entire string table to point to another string table:

```
[Core.StringTable]
+StringTableRedirects=(OldStringTable="Foo", NewStringTable="Bar")
```

The following will redirect a string string table entry to another string table entry:

```
[Core.StringTable]
+StringTableRedirects=(StringTable="Foo", OldKey="Bar",
NewKey="Baz")
```

Assets

Moving or renaming a String Table asset will leave an asset redirector behind. This will be processed and followed as any other asset redirector would.