INTERNATIONALIZATION SCRIPT INTERFACE

View online: https://www.construct.net/en/make-games/manuals/construct-3/scripting/scripting-reference/plugin-interfaces/internationalization

The `IInternationalizationObjectType` interface derives from IObjectType to add APIs specific to the Internationalization plugin.

Internationalization is sometimes written as the shorthand i18n, referring to the fact the word starts with an I, ends with an N, and has 18 other letters in between.

Supported APIs

Many features of the Internationalization plugin merely access the browser-provided Intl APIs, such as for formatting dates and times, identifying plural types, and so on. When writing code, you may as well access these APIs directly rather than through the script interface, so those APIs are not duplicated here. Instead the script interface only provides access to string lookups for translations.

Internationalization APIs

locale

Set or get a string representing the currently set locale as a BCP 47 language tag, e.g. `"en-US"`.

addString(context, str)

Add a localized string to the provided context in the localization data for the current locale.

saveToJSONString()

Returns a string of JSON representing the state of the Internationalization object.

loadFromJSONString(str)

Load the state of the Internationalization object from a string of JSON data.

setContext(context)

getContext()

Set and get a string representing the current context for string lookups.

---

pushContext(context)

popContext()

Push or pop to the context stack. When a string is pushed, it is used as the base for all relative context lookups.

> *Consider using `createContext()` as a more convenient alternative API, as it saves you from having to remember to call `popContext()`.*

---

createContext(context)

Return an `I18NLookupContext` representing the given context. This provides a convenient way to look up several strings relative to that context. See the section below for more details.

---

lookup(context, ...args)

lookupPlural(context, count, ...args)

Look up a string at the given context, and substitute placeholders with the following arguments. The context may be relative if a context has been pushed. The plural variant also makes use of the `count` argument to select the appropriate plural variant for the current locale. See the Internationalization manual entry for more details on placeholders, relative contexts and pluralization.

I18NLookupContext

The `createContext()` method returns an `I18NLookupContext` which can be used for more conveniently looking up lots of strings from the same context. This works similarly to using `pushContext()`, but takes advantage of JavaScript's garbage collector to avoid having to need a following call to `popContext()`. This makes it more convenient and less error-prone to use than pushing and popping contexts. It's also uniquely available to the script interface, as Construct's event system does not support allow a comparable API.

The code sample below demonstrates how it can be used.

```
const intl = runtime.objects.Internationalization;

// Normal lookup having to repeat the context
let string1 = intl.lookup("forest-world.chapter-1.intro-text.title");
let string2 = intl.lookup("forest-world.chapter-1.intro-text.message");
let string3 = intl.lookup("forest-world.chapter-1.intro-text.start");

// Shorter lookups using a context
```

```
const ctx = intl.createContext("forest-world.chapter-1.intro-text");

string1 = ctx.lookup(".title");
string2 = ctx.lookup(".message");
string3 = ctx.lookup(".start");
```

I18NLookupContext APIs

---

lookup(context, ...args)

lookupPlural(context, count, ...args)

These methods are equivalent to those on the Internationalization script interface, but may use contexts relative to the context this `I18NLookupContext` was created with.

---

createContext(context)

Create another `I18NLookupContext` for the given context, which must be relative. This allows creating further `I18NLookupContext` objects for more deeply nested contexts.