SceneGraph Localization

Table of Contents

You can supply localized versions of the strings and package graphical images used in a SceneGraph application package. The SceneGraph application can then automatically insert the localized versions of the strings and graphical images that you supply, according to the current locale as set by the user.

You can also localize your application from your server using URL directives based on the current locale as set by the user.

> The application package localization features rely on the locales and locale IDs currently supported by Roku. The list of these supported locales can be found in the description of the **ifDeviceInfo** GetCurrentLocale() method.

Localizing Strings in the Application Package

For each string to be localized, provide localized translations of the string, in as many languages as you want that are currently supported by Roku. You can provide localized strings in either the TS or XLIFF XML localization file formats as supported by Roku. The string localization files should be placed in the `pkg:/locale/` subdirectory for the supported locale. For example, if you want to include a `translations.ts` localization file in the TS format for Canadian French, place the file in the `pkg:/locale/` subdirectory named for the locale ID `fr_CA`:

```
pkg:/locale/fr_CA/translations.ts
```

For strings defined in XML markup in a **<children>** element, or **<interface>** element field strings, the strings will be automatically translated by the SceneGraph application, if translations of the string exist in the localization files. If no translation file exists for the current locale, or no translation of the string exists in the translation file, the original string will be used.

For strings assigned using BrightScript in a **<script>** element, you can use the `tr()` function to localize a string, if you have provided a translation of the string in a `translations.xml` file in the XLIFF XML format in the package directory `pkg:/locale/`*`locale_ID/`* directory . The `tr()` function is a global BrightScript utility function that looks for the string in the `translations.xml` file, then finds the corresponding string translation in the file for the current locale. To use the `tr()` function to localize a specific *source* string:

```
tr(String source) as String
```

For example, to translate the string "hello world" in BrightScript:

```
m.greetinglabel.text = tr("hello world")
```

The `tr()` function will look for the `pkg:/locale/`*`locale_ID`*`/translations.xml` file, and look in the file for the "hello world" source string. If the source string exists in the file, with a translated string, the translated string will be returned, and assigned to the `text` field of the **Label** node object `m.greetinglabel`. If no translation file exists, or the source string and translated string does not exist in the file, the original source string will be returned, and assigned to the `text` field.

The `tr()` function also supports string substitutions. For example:

```
text = tr("Video will start in %1 seconds").Replace("%1", numSeconds.ToStr())
```

Localizing Graphical Images in the Application Package

For each graphical image to be localized, provide each localized image in the `pkg:/locale/` directory, in as many languages as you want that are currently supported by Roku. Then use the following format for the value of the `uri` field of the **Poster** node:

```
pkg:/locale/images/localized_image
```

For example, to provide a localized version of `myPoster.jpg`, set the **Poster** node `uri` field as follows:

```
<Poster uri = pkg:/locale/images/myPoster.jpg />
```

This causes the application to search through the `pkg:/locale` directory in the following order to find the localized graphical image. The first graphical image of the specified name found in this search sequence is returned to be rendered by the SceneGraph application.

1. not localized
   `pkg:/locale/images/image_name`
2. current locale
   `pkg:/locale/locale_id/images/image_name`
3. default
   `pkg:/locale/default/images/image_name`
4. US English
   `pkg:/locale/en_US/images/image_name`

If none of those files exist, the URL access will fail to return a file.

For example, if you want to have a localized version of a graphical image for Canadian French, and a default image for all other languages, place two versions of the image in the `pkg:/locale` directory as follows:

```
pkg:/locale/fr_CA/images/image_name_canadian_french
pkg:/locale/default/images/image_name_default
```

In Brightscript in a **<script>** element, you can also use the getLocalizedAsset() method in **ifLocalization** to get a localized graphical image. For example:

```
localize = createObject("RoLocalization")
bannerposter = m.top.findNode("bannerPoster")
bannerposter.uri = localize.GetLocalizedAsset("images", "banner.png")
```

Is equivalent to:

```
bannerposter = m.top.findNode("bannerPoster")
bannerposter.uri = "pkg:/locale/images/banner.png"
```

TS File Example

The following is an example of a `translations.ts` file in the TS XML format placed in the `pkg:/locale/fr_CA` directory to translate English to Canadian French:

Example of a translations.ts file in the TS XML format

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE TS>
<TS version="2.0" language="fr_CA" sourcelanguage="en_US">
<defaultcodec>UTF-8</defaultcodec>
<context>
  <name>default</name>
  <message>
    <source>Hello</source>
    <translation>Bonjour</translation>
  </message>
  <message>
    <source>Goodbye</source>
    <translation>Au revoir</translation>
  </message>
  <message>
    <source>Christmas</source>
    <translation>Noel</translation>
  </message>
</context>
</TS>
```

XLIFF File Example

The following is an example of a `translations.xml` file in the XLIFF XML format placed in the `pkg:/locale/fr_CA` directory to translate English to Canadian French:

Example of a translations.xml file in the XLIFF XML format

```
<?xml version="1.0" encoding="UTF-8"?>
<xliff version="1.2" xmlns="urn:oasis:names:tc:xliff:document:1.2">
<file source-language="en-US" target-language="fr-CA" >
<body>
  <trans-unit id="0">
    <source>Hello</source>
    <target>Bonjour</target>
  </trans-unit>
  <trans-unit id="1">
    <source>Goodbye</source>
    <target>Au revoir</target>
  </trans-unit>
</body>
</file>
</xliff>
```