Localizing Menu Commands

原文<https://docs.microsoft.com/zh-cn/visualstudio/extensibility/localizing-menu-commands>

本文内容

Localizing Command Names

Localizing Other Text Resources

Incorporating Localized Resources into the Project

See Also

You can provide localized text for menu and toolbar commands by creating localized .vsct files and localized .resx files for your VSPackage, and then updating the project files to incorporate the changes.

For information about how to localize the installation experience, see [Localizing VSIX Packages](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/localizing-vsix-packages).

Localizing Command Names

**1.** In VSPackages, menu commands and toolbar buttons are defined in the .vsct file.

In **Solution Explorer**, change the name of the .vsct file from *filename*.vsct to *filename*.en-US.vsct.

**2.** Make a copy of *filename*.en-US.vsct for each localized language.

Name each copy *filename.Locale*.vsct, where *Locale* is a particular culture name. For a list of culture name values, see [Locale IDs Assigned by Microsoft](https://msdn.microsoft.com/en-us/library/windows/apps/jj657969.aspx).

These *filename.Locale*.vsct files will contain the localized menu text for your package.

**3.** Open each *filename.Locale*.vsct file to localize the text.

**a.** Modify the [ButtonText](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/buttontext-element) element values as appropriate for the particular language.

**b.** If you will provide localized icons, modify the [Bitmap](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/bitmap-element) values to point to the target files.

The following example shows English and Spanish button text for a command to open a Family Tree Explorer tool window.

[FamilyTree.en-US.vsct]

<Button guid="guidLocalizedPackageCmdSet" id="cmdidFamilyTree" priority="0x0100" type="Button">

<Parent guid="guidSHLMainMenu" id="IDG\_VS\_WNDO\_OTRWNDWS1"/>

<Icon guid="guidImages" id="bmpPic2" />

<Strings>

<CommandName>cmdidFamilyTree</CommandName>

<ButtonText>Family Tree Explorer</ButtonText>

</Strings>

</Button>

[FamilyTree.es-ES.vsct]

<Button guid="guidLocalizedPackageCmdSet" id="cmdidFamilyTree" priority="0x0100" type="Button">

<Parent guid="guidSHLMainMenu" id="IDG\_VS\_WNDO\_OTRWNDWS1"/>

<Icon guid="guidImages" id="bmpPic2" />

<Strings>

<CommandName>cmdidFamilyTree</CommandName>

<ButtonText>Explorar el arbol genealogico</ButtonText>

</Strings>

</Button>

Localizing Other Text Resources

Text resources other than command names are defined in resource (.resx) files.

**1.** Rename VSPackage.resx to VSPackage.en-US.resx.

**2.** Make a copy of the VSPackage.en-US.resx file for each localized language.

Name each copy VSPackage.*Locale*.resx, where *Locale* is a particular culture name.

**3.** Rename Resources.resx to Resources.en-US.resx.

**4.** Make a copy of the Resources.en-US.resx file for each localized language.

Name each copy Resources.*Locale*.resx, where *Locale* is a particular culture name.

**5.** Open each .resx file to modify the string values as appropriate for the particular language and culture. The following example shows the localized resource definition for the title bar of a tool window.

[Resources.en-US.resx]

<data name="ToolWindowTitle" xml:space="preserve">

<value>Family Tree Explorer</value>

</data>

[Resources.es-ES.resx]

<data name="ToolWindowTitle" xml:space="preserve">

<value>Explorador del arbol genealogico</value>

</data>

Incorporating Localized Resources into the Project

You must modify the assemblyinfo.cs file and the project file to incorporate the localized resources.

**1.** From the **Properties** node in **Solution Explorer**, open assemblyinfo.cs or assemblyinfo.vb in the editor.

**2.** Add the following entry.

[assembly: NeutralResourcesLanguage("en-US", UltimateResourceFallbackLocation.Satellite)]

This sets US English as the default language.

**3.** Unload the project.

**4.** Open the project file in the editor.

**5.** Locate the ItemGroup element that contains EmbeddedResource elements.

**6.** In the EmbeddedResource element that calls VSPackage.en-US.resx, replace the ManifestResourceName element with a LogicalName element, set to VSPackage.en-US.Resources, as follows.

<EmbeddedResource Include="VSPackage.en-US.resx">

<MergeWithCTO>true</MergeWithCTO>

<LogicalName>VSPackage.en-US.Resources</LogicalName>

</EmbeddedResource>

**7.** For each localized language, copy the EmbeddedResource element for VsPackage.en-US, and set the **Include** attribute and **LogicalName** element of the copy to the target locale, as shown in the following example.

**8.** To each localized VSCTCompile element, add a ResourceName element that points to Menus.ctmenu, as shown in the following example.

<ItemGroup>

<VSCTCompile Include="LocalizedPackage.es-ES.vsct">

<ResourceName>Menus.ctmenu</ResourceName>

</VSCTCompile>

</ItemGroup>

**9.** Save the project file and reload the project.

**10.** Build the project.

This creates a main assembly, and resource assemblies for each language. For information on localizing the deployment process, see [Localizing VSIX Packages](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/localizing-vsix-packages)

See Also

[Extending Menus and Commands](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/extending-menus-and-commands)

[MenuCommands Vs. OleMenuCommands](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/menucommands-vs-olemenucommands)

[Globalizing and Localizing Applications](https://docs.microsoft.com/zh-cn/visualstudio/ide/globalizing-and-localizing-applications)