Adding a Menu Controller to a Toolbar

本文内容

Prerequisites

Creating a Menu Controller

Implementing the Menu Controller Commands

Testing the Menu Controller

See Also

This walkthrough builds on the [Adding a Toolbar to a Tool Window](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/adding-a-toolbar-to-a-tool-window) walkthrough and shows how to add a menu controller to the tool window toolbar. The steps shown here also can be applied to the toolbar that is created in the [Adding a Toolbar](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/adding-a-toolbar) walkthrough.

A menu controller is a split control. The left side of the menu controller shows the last-used command, and it can be run by clicking it. The right side of the menu controller is an arrow that, when clicked, opens a list of additional commands. When you click a command on the list, the command runs, and it replaces the command on the left side of the menu controller. In this way, the menu controller operates like a command button that always shows the last-used command from a list.

Menu controllers can appear on menus but they are most often used on toolbars.

Prerequisites

Starting in Visual Studio 2015, you do not install the Visual Studio SDK from the download center. It is included as an optional feature in Visual Studio setup. You can also install the VS SDK later on. For more information, see [Installing the Visual Studio SDK](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/installing-the-visual-studio-sdk).

Creating a Menu Controller

To create a menu controller

**1.** Follow the procedures described in [Adding a Toolbar to a Tool Window](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/adding-a-toolbar-to-a-tool-window) to create a tool window that has a toolbar.

**2.** In TWTestCommandPackage.vsct, go to the Symbols section. In the GuidSymbol element named **guidTWTestCommandPackageCmdSet**, declare your menu controller, menu controller group, and three menu items.

<IDSymbol name="TestMenuController" value="0x1300" />

<IDSymbol name="TestMenuControllerGroup" value="0x1060" />

<IDSymbol name="cmdidMCItem1" value="0x0130" />

<IDSymbol name="cmdidMCItem2" value="0x0131" />

<IDSymbol name="cmdidMCItem3" value="0x0132" />

**3.** In the Menus section, after the last menu entry, define the menu controller as a menu.

<Menu guid="guidTWTestCommandPackageCmdSet" id="TestMenuController" priority="0x0100" type="MenuController">

<Parent guid="guidTWTestCommandPackageCmdSet" id="TWToolbarGroup" />

<CommandFlag>IconAndText</CommandFlag>

<CommandFlag>TextChanges</CommandFlag>

<CommandFlag>TextIsAnchorCommand</CommandFlag>

<Strings>

<ButtonText>Test Menu Controller</ButtonText>

<CommandName>Test Menu Controller</CommandName>

</Strings>

</Menu>

The TextChanges and TextIsAnchorCommand flags must be included to enable the menu controller to reflect the last selected command.

**4.** In the Groups section, after the last group entry, add the menu controller group.

<Group guid="guidTWTestCommandPackageCmdSet" id="TestMenuControllerGroup"

priority="0x000">

<Parent guid="guidTWTestCommandPackageCmdSet" id="TestMenuController" />

</Group>

By setting the menu controller as the parent, any commands placed in this group will appear in the menu controller. The priority attribute is omitted, which sets it to the default value of 0, because it will be the only group on the menu controller.

**5.** In the Buttons section, after the last button entry, add a Button element for each of your menu items.

<Button guid="guidTWTestCommandPackageCmdSet" id="cmdidMCItem1" priority="0x0000"

type="Button">

<Parent guid="guidTWTestCommandPackageCmdSet" id="TestMenuControllerGroup" />

<Icon guid="guidImages" id="bmpPic1" />

<CommandFlag>IconAndText</CommandFlag>

<Strings>

<ButtonText>MC Item 1</ButtonText>

<CommandName>MC Item 1</CommandName>

</Strings>

</Button>

<Button guid="guidTWTestCommandPackageCmdSet" id="cmdidMCItem2" priority="0x0100"

type="Button">

<Parent guid="guidTWTestCommandPackageCmdSet" id="TestMenuControllerGroup" />

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

<CommandFlag>IconAndText</CommandFlag>

<Strings>

<ButtonText>MC Item 2</ButtonText>

<CommandName>MC Item 2</CommandName>

</Strings>

</Button>

<Button guid="guidTWTestCommandPackageCmdSet" id="cmdidMCItem3" priority="0x0200"

type="Button">

<Parent guid="guidTWTestCommandPackageCmdSet" id="TestMenuControllerGroup" />

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

<CommandFlag>IconAndText</CommandFlag>

<Strings>

<ButtonText>MC Item 3</ButtonText>

<CommandName>MC Item 3</CommandName>

</Strings>

</Button>

**6.** At this point, you can look at the menu controller. Build the project and start debugging. You should see the experimental instance.

a. On the **View / Other Windows** menu, open **Test ToolWindow**.

b. The menu controller appears on the toolbar in the tool window.

c. Click the arrow on the right-hand side of the menu controller to see the three possible commands.

Notice that when you click a command, the title of the menu controller changes to display that command. In the next section, we will add the code to activate these commands.

Implementing the Menu Controller Commands

**1.** In TWTestCommandPackageGuids.cs, add command IDs for your three menu items after the existing command IDs.

public const int cmdidMCItem1 = 0x130;

public const int cmdidMCItem2 = 0x131;

public const int cmdidMCItem3 = 0x132;

**2.** In TWTestCommand.cs, add the following code at the top of the TWTestCommand class.

private int currentMCCommand; // The currently selected menu controller command

**3.** In the TWTestCommand constructor, after the last call to the AddCommand method, add code to route the events for each command through the same handlers.

for (int i = TWTestCommandPackageGuids.cmdidMCItem1; i <=

TWTestCommandPackageGuids.cmdidMCItem3; i++)

{

CommandID cmdID = new

CommandID(new Guid(TWTestCommandPackageGuids.guidTWTestCommandPackageCmdSet), i);

OleMenuCommand mc = new OleMenuCommand(new

EventHandler(OnMCItemClicked), cmdID);

mc.BeforeQueryStatus += new EventHandler(OnMCItemQueryStatus);

commandService.AddCommand(mc);

// The first item is, by default, checked.

if (TWTestCommandPackageGuids.cmdidMCItem1 == i)

{

mc.Checked = true;

this.currentMCCommand = i;

}

}

**4.** Add an event handler to the TWTestCommand class to mark the selected command as checked.

private void OnMCItemQueryStatus(object sender, EventArgs e)

{

OleMenuCommand mc = sender as OleMenuCommand;

if (null != mc)

{

mc.Checked = (mc.CommandID.ID == this.currentMCCommand);

}

}

**5.** Add an event handler that displays a MessageBox when the user selects a command on the menu controller:

private void OnMCItemClicked(object sender, EventArgs e)

{

OleMenuCommand mc = sender as OleMenuCommand;

if (null != mc)

{

string selection;

switch (mc.CommandID.ID)

{

case c.cmdidMCItem1:

selection = "Menu controller Item 1";

break;

case TWTestCommandPackageGuids.cmdidMCItem2:

selection = "Menu controller Item 2";

break;

case TWTestCommandPackageGuids.cmdidMCItem3:

selection = "Menu controller Item 3";

break;

default:

selection = "Unknown command";

break;

}

this.currentMCCommand = mc.CommandID.ID;

IVsUIShell uiShell =

(IVsUIShell) ServiceProvider.GetService(typeof(SVsUIShell));

Guid clsid = Guid.Empty;

int result;

uiShell.ShowMessageBox(

0,

ref clsid,

"Test Tool Window Toolbar Package",

string.Format(CultureInfo.CurrentCulture,

"You selected {0}", selection),

string.Empty,

0,

OLEMSGBUTTON.OLEMSGBUTTON\_OK,

OLEMSGDEFBUTTON.OLEMSGDEFBUTTON\_FIRST,

OLEMSGICON.OLEMSGICON\_INFO,

0,

out result);

}

}

Testing the Menu Controller

**1.** Build the project and start debugging. You should see the experimental instance.

**2.** Open the **Test ToolWindow** on the **View / Other Windows** menu.

The menu controller appears in the toolbar in the tool window and displays **MC Item 1**.

**3.** Click the menu controller button to the left of the arrow.

You should see three items, the first of which is selected and has a highlight box around its icon. Click **MC Item 3**.

A dialog box appears with the message **You selected Menu controller Item 3**. Notice that the message corresponds to the text on the menu controller button. The menu controller button now displays **MC Item 3**.

See Also

[Adding a Toolbar to a Tool Window](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/adding-a-toolbar-to-a-tool-window)

[Adding a Toolbar](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/adding-a-toolbar)