Adding a Toolbar to a Tool Window

原文

<https://docs.microsoft.com/zh-cn/visualstudio/extensibility/adding-a-toolbar-to-a-tool-window>

本文内容

Prerequisites

Creating a Toolbar for a Tool Window

Adding the Toolbar to the Tool Window

Testing the Toolbar in the Tool Window

See Also

This walkthrough shows how to add a toolbar to a tool window.

A toolbar is a horizontal or vertical strip that contains buttons bound to commands. The length of a toolbar in a tool window is always the same as the width or height of the tool window, depending on where the toolbar is docked.

Unlike toolbars in the IDE, a toolbar in a tool window must be docked and cannot be moved or customized. If the VSPackage is written in umanaged code, the toolbar can be docked on any edge.

For more information about how to add a toolbar, see [Adding a Toolbar](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/adding-a-toolbar).

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 Toolbar for a Tool Window

**1.** Create a VSIX project named TWToolbar that has both a menu command named **TWTestCommand** and a tool window named **TestToolWindow**. For more information, see [Creating an Extension with a Menu Command](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/creating-an-extension-with-a-menu-command) and [Creating an Extension with a Tool Window](https://docs.microsoft.com/zh-cn/visualstudio/extensibility/creating-an-extension-with-a-tool-window). You need to add the command item template before adding the tool window template.

**2.** In TWTestCommandPackage.vsct, look for the Symbols section. In the GuidSymbol node named guidTWTestCommandPackageCmdSet declare a toolbar and a toolbar group, as follows.

<IDSymbol name="TWToolbar" value="0x1000" />

<IDSymbol name="TWToolbarGroup" value="0x1050" />

**3.** At the top of the Commands section, create a Menus section. Add a Menu element to define the toolbar.

<Menus>

<Menu guid="guidTWTestCommandPackageCmdSet" id="TWToolbar" type="ToolWindowToolbar">

<CommandFlag>DefaultDocked</CommandFlag>

<Strings>

<ButtonText>Test Toolbar</ButtonText>

<CommandName>Test Toolbar</CommandName>

</Strings>

</Menu>

</Menus>

Toolbars cannot be nested like submenus. Therefore, you do not have to assign a parent. Also, you do not have to set a priority, because the user can move toolbars. Typically, initial placement of a toolbar is defined programmatically, but subsequent changes by the user are persisted.

**4.** In the Groups section, define a group to contain the commands for the toolbar.

<Group guid="guidTWTestCommandPackageCmdSet" id="TWToolbarGroup" priority="0x0000">

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

</Group>

**5.** In the Buttons section, change the parent of the existing Button element to the toolbar group so that the toolbar will be displayed.

<Button guid="guidTWTestCommandPackageCmdSet" id="TWTestCommandId" priority="0x0100" type="Button">

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

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

<Strings>

<ButtonText>Invoke TWTestCommand</ButtonText>

</Strings>

</Button>

By default, if a toolbar has no commands, it does not appear.

Because the new toolbar is not automatically added to the tool window, the toolbar must be added explicitly. This is discussed in the next section.

Adding the Toolbar to the Tool Window

**1.** In TWTestCommandPackageGuids.cs add the following lines.

public const string guidTWTestCommandPackageCmdSet = "00000000-0000-0000-0000-0000"; // get the GUID from the .vsct file

public const int TWToolbar = 0x1000;

**2.** In TestToolWindow.cs add the following using statement.

using System.ComponentModel.Design;

**3.** In the TestToolWindow constructor add the following line.

this.ToolBar = new CommandID(new Guid(TWTestCommandPackageGuids.guidTWTestCommandPackageCmdSet), TWTestCommandPackageGuids.TWToolbar);

Testing the Toolbar in the Tool Window

**1.** Build the project and start debugging. The Visual Studio experimental instance should appear.

**2.** On the **View / Other Windows** menu, click **Test ToolWindow** to display the tool window.

You should see a toolbar (it looks like the default icon) at the top left of the tool window, just below the title.

**3.** On the toolbar, click the icon to display the message **TWTestCommandPackage Inside TWToolbar.TWTestCommand.MenuItemCallback()**.

See Also

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