为子菜单添加最近使用列表

原文

https://docs.microsoft.com/zh-cn/visualstudio/extensibility/adding-a-most-recently-used-list-to-a-submenu

本指导以Adding a Submenu to a Menu里的指导为基础，说明如何为子菜单添加动态列表，这是创建最近使用列表菜单（MRU）的基础。

创建动态菜单需要在菜单里使用占位符。当菜单显示时，Visual Studio IDE查询VSPackage获得占位符里应该显示的命令项。动态列表可以出现在菜单的任何地方，但是动态列表通常作为一个单独的子菜单或者显示在菜单的底部。这样就可以在动态列表变化时不影响菜单里其他项的位置。本指导中的动态列表将显示在现有子菜单的底部，用一条分隔线来与菜单里其它项分开。

从技术上讲，动态列表也可以应用到工具栏上，但是我们不建议这样做，因为工具栏应该保持不变除非用户手动改动它。我们将创建一个包含四个项的MRU菜单，每次选中一项后进行自动排列（选中项移到菜单顶部）。

更多关于菜单和.vsct文件的信息，参见Commands, Menus, and Toolbars。

准备

请安装好Visual Studio SDK。详见Visual Studio SDK。

创建扩展

按照Adding a Submenu to a Menu的步骤创建子菜单，接下来我们来修改它。这里VSPackage的名字是Adding a Menu to the Visual Studio Menu Bar里使用的TopLevelMenu。

创建动态列表

1. 打开TestCommandPackage.vsct文件。

2. 在Symbols里名为guidTestCommandPackageCmdSet的GuidSymbol节点中添加MRUListGroup组和cmdidMRUList命令项的声明，如下所示。

```
<IDSymbol name="MRUListGroup" value="0x1200"/>
<IDSymbol name="cmdidMRUList" value="0x0200"/>
```

3. 在Group里添加上一步声明过的菜单组。

```
<Group guid="guidTestCommandPackageCmdSet" id="MRUListGroup"
       priority="0x0100">
    <Parent guid="guidTestCommandPackageCmdSet" id="SubMenu"/>
```

```
</Group>
```

4. 在Buttons里现有按钮节点后面添加新节点用于表示第二步声明过的按钮。

```
<Button guid="guidTestCommandPackageCmdSet" id="cmdidMRUList"
    type="Button" priority="0x0100">
    <Parent guid="guidTestCommandPackageCmdSet" id="MRUListGroup" />
    <CommandFlag>DynamicItemStart</CommandFlag>
    <Strings>
        <CommandName>cmdidMRUList</CommandName>
        <ButtonText>MRU Placeholder</ButtonText>[1]
    </Strings>
</Button>
```

DynamicItemStart标志说明命令项可以被动态生成。

5. 生成解决方案并调试，测试刚才新添加的命令项的显示。

在TestMenu菜单里点击这个新的Sub Menu子菜单以显示其中的新项MRU Placeholder。在执行下个步骤的动态MRU菜单后，子菜单打开时这个项每次都会被替换为那个动态列表。

填充 MRU 菜单列表

1. 在TestCommandPackageGuids.cs[2]文件里TestCommandPackageGuids类中的现有命令项ID声明后添加以下代码。

```
public const string guidTestCommandPackageCmdSet = "00000000-0000-0000-0000-00000000"; //[3]
public const uint cmdidMRUList = 0x200;
```

2. 在TestCommand.cs文件里添加如下using语句。

```
using System.Collections;
```

3. 在调用最后一个AddCommand方法之后添加如下代码，下面的步骤会定义InitMRUMenu方法。

```
this.InitMRUMenu(commandService);
```

4. 把以下代码添加到TestCommand类，这些代码初始化MRU菜单里要显示的项。

```
private int numMRUItems = 4;
private int baseMRUID = (int)TestCommandPackageGuids.cmdidMRUList;
private ArrayList mruList;

private void InitializeMRUList()
{
    if (null == this.mruList)
    {
        this.mruList = new ArrayList();
        if (null != this.mruList)
        {
            for (int i = 0; i < this.numMRUItems; i++)
            {
                this.mruList.Add(string.Format(CultureInfo.CurrentCulture,
                    "Item {0}", i + 1));
```

[1] MIAOW 注：对于 Visual Studio 2017（其它版本没试过，估计也一样），ButtonText 应该设置为空，你可以去试试不为空会发生什么。
[2] MIAOW 注：我的 VS 自动生成的叫 TestCommandPackage.cs
[3] MIAOW 注：这里的 GUID 要去.vsct 文件里找。

```
            }
        }
    }
}
```

5. 把InitMRUMenu方法添加到InitializeMRUList方法之后，初始化MRU菜单里的命令项列表。

```
private void InitMRUMenu(OleMenuCommandService mcs)
{
    InitializeMRUList();
    for (int i = 0; i < this.numMRUItems; i++)
    {
        var cmdID = new CommandID(
            new Guid(TestCommandPackageGuids.guidTestCommandPackageCmdSet), this.baseMRUID
 + i);
        var mc = new OleMenuCommand(
            new EventHandler(OnMRUExec), cmdID);
        mc.BeforeQueryStatus += new EventHandler(OnMRUQueryStatus);
        mcs.AddCommand(mc);
    }
}
```

你必须为每个可能在MRU菜单里出现的项创建一个菜单命令对象。IDE会为每一个MRU菜单项调用OnMRUQueryStatus方法。在托管代码里，IDE知道不再有更多菜单项的唯一办法是先创建所有可能的项。[4]如果你愿意的话，可以在OnMRUQueryStatus方法里使用mc.Visible = false;来使项不可见，然后在菜单被创建后使用mc.Visible = true;使项可见。

6. 把如下的OnMRUQueryStatus方法添加到InitMRUMenu方法后面，为每个MRU菜单项设置显示文本。

```
private void OnMRUQueryStatus(object sender, EventArgs e)
{
    OleMenuCommand menuCommand = sender as OleMenuCommand;
    if (null != menuCommand)
    {
        int MRUItemIndex = menuCommand.CommandID.ID - this.baseMRUID;
        if (MRUItemIndex >= 0 && MRUItemIndex < this.mruList.Count)
        {
            menuCommand.Text = this.mruList[MRUItemIndex] as string;
        }
    }
}
```

7. 把如下OnMRUExec方法添加到OnMRUQueryStatus方法后面，用于处理MRU菜单项的选中事件。这个方法把选中项移到列表顶端并且用一个消息框（message box）显示选中项。

```
private void OnMRUExec(object sender, EventArgs e)
{
    var menuCommand = sender as OleMenuCommand;
    if (null != menuCommand)
    {
        int MRUItemIndex = menuCommand.CommandID.ID - this.baseMRUID;
        if (MRUItemIndex >= 0 && MRUItemIndex < this.mruList.Count)
        {
            string selection = this.mruList[MRUItemIndex] as string;
            for (int i = MRUItemIndex; i > 0; i--)
```

[4] MIAOW 注：反正我是没看懂这句话，不过因 OnMRUQueryStatus 方法注册在 BeforeQueryStatus 事件中，所以每次需要显示这些菜单项前 IDE 都会自动调用 OnMRUQueryStatus。如此就只需更改 mruList 中顺序也能使菜单命令项本身调换位置了。

```
            {
                this.mruList[i] = this.mruList[i - 1];
            }
            this.mruList[0] = selection;
            System.Windows.Forms.MessageBox.Show(
                string.Format(CultureInfo.CurrentCulture,
                            "Selected {0}", selection));
        }
    }
}
```

测试 MRU 菜单列表

按照以下步骤测试MRU菜单

1. 生成解决方案并调试
2. 点击TestMenu菜单里的Invoke TestCommand，就会出现一个对话框说明你选择了这个命令项。

> 注意
>
> 这一步强制加载了 VSPackage 使得 MRU 菜单显示正确。如果跳过这一步，MRU 菜单将不会显示。

3. 点击Test Menu菜单里的Sub Menu会在其底部一条分割线下显示四个项。如果你点击了Item 3，会弹出一个消息框显示"Selected Item 3"。（如果没有这四个菜单项，请确保你遵照了之前步骤。）
4. 再次打开子菜单，注意此时Item 3已经在列表顶部了，而其他项被往下挤了一格。点击Item 3会发现消息框仍旧显示"Selected Item 3"，这说明上一步中是命令项本身而不光是显示的文本移动到了新的位置。[5]

另见

Dynamically Adding Menu Items。

[5] MIAOW 注：这只是个假象啦，并不是命令本身变化了位置。最上面的那个的 id 一直是 0x200,向下依次加 1。正好使对话框显示的 mruList[menuCommand.CommandID.ID - this.baseMRUID]与 Text 属性一致（Text 属性不也是从更新后的 mruList 里来的吗）。不信？在 OnMRUQueryStatus 里最后加上一句，比如让 id 为 0x201 的 Checked 属性为 true。发现那个 Checked 的勾总是在从上往下第二的位置。不就说明命令本身没有变过位置过吗？