Windows 8 开发 31 日

第 16 日

上下文菜单

译者：BeyondVincent(破船)

时间：2013.4.25

版本： 2.0

关于破船

程序猿砌墙于云南昆明!

长期扎根移动软件开发!

爱跑步爱打篮球爱运动!

命中无大富大贵之面相!

愿健康与平淡相随一生!

你可以发邮件与破船取得联系: BeyondVincent@gmail.com

还可以关注破船的微博: 腾讯微博和新浪微博。

这里是破船的个人博客，欢迎光临：破船之家

关于 Windows 8 开发 31 日翻译

Windows 8 开发 31 日是由 Jeff Blankenburg 和 Clark Sell 原创的。

官方站点：http://31daysofwindows8.com/

涉及到两个版本：

XAML/C#（由 Jeff Blankenburg 撰写）

HTML5/JS （由 Clark Sell 撰写）

其中涉及到的资源和相关代码请到这里下载：

https://github.com/csell5/31DaysOfWindows8

在这里，由于破船对 HTML5/JS 不熟悉，所以只翻译 XAML/C#相关主题。

建议大家前往看原创内容，如果看不明白，再来这里看我翻译的相关内容。

如果翻译不正确的地方，可以通过上面的联系方式告诉破船。

破船祝你阅读愉快！

目录

关于破船 2

关于 Windows 8 开发 31 日翻译 3

目录 4

第 16 日上下文菜单 5

1.0. 介绍 5
1.1. 什么是上下文菜单 5
1.2. 确定 Element 的位置 6
1.3. 创建上下文菜单 7
1.4. 在 TextBox 中启动上下文菜单 10
1.5. 总结 13

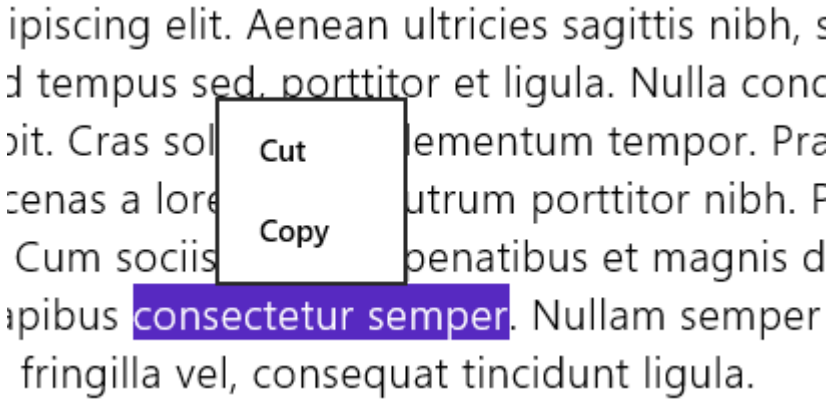
第 16 日上下文菜单

1.0. 介绍

今天，我们来学习上下文菜单。程序中，当右键单击某个对象时，时不时的会有小小的弹出菜单命令。[什么时候使用上下文菜单微软提供了非常详细的指导](#) VS.什么时候使用 AppBar 控件来替代，在本文中，都将按照这些规则来处理。

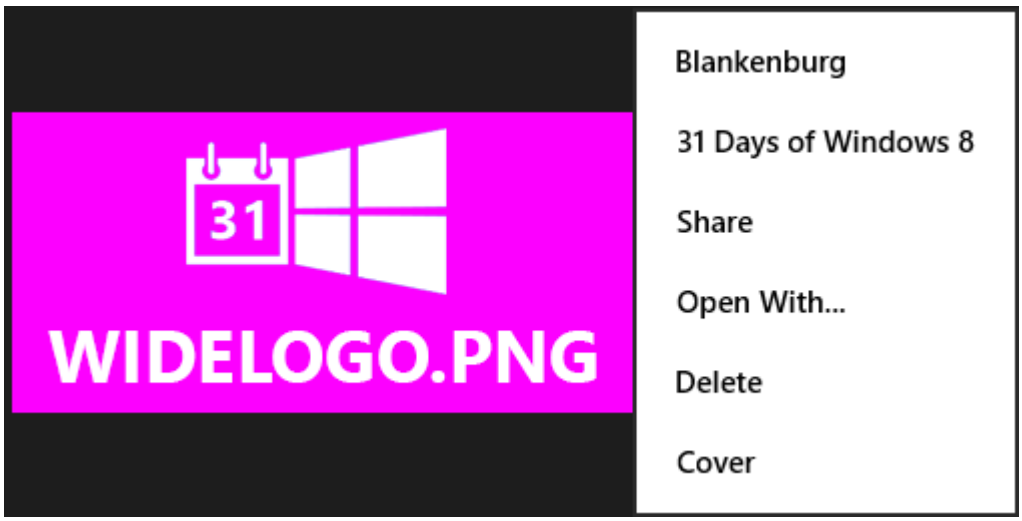
1.1. 什么是上下文菜单

如果你使用过 Windows 8，你可能已经遇见到过上下文菜单了。经常在一些不可以选择的对象上右键单击，或者在 text 文本上进行操作时，会出现上下文菜单。下面是一个上下文菜单截图：



(图片来自 http://msdn.microsoft.com/library/windows/apps/Hh465308)

也可以在一个不可选的 element 上显示一个上下文菜单，如下面截自示例程序中的一个图：



在图片上右键单击，就可以在显示出上下文菜单（下面我将介绍如何显示）。菜单中的每一个 command item 都有对应的 action，但点击 item 时，相应的 action 会被执行。

1.2. 确定 Element 的位置

你可能已经注意到了上下文菜单出现在被选中 element 的附近。这其实上并不复杂。实际上，当我们创建弹出菜单时，首先需要确定出被点击 element 所在的位置，然后将位置传递给弹出菜单控件。下面的方法是确定 element 的位置：

```
privateRectGetRect(object sender)
```

```
{
FrameworkElement element = sender asFrameworkElement;
GeneralTransformelementTransform = element.TransformToVisual(null);
Pointpoint = elementTransform.TransformPoint(newPoint());
returnnewRect(point, newSize(element.ActualWidth, element.ActualHeight));
}
```

如上所示，我可以传递“sender”对象到这个方法中，然后返回一个 Rect 对象，该 rect 中有一个 piont（element 对象的左上角位置），以及 size(element 的尺寸)。

1.3. 创建上下文菜单

有了 GetRect()这个方法，那么在控件附近创建上下文菜单就很简单了。在示例中，我给图片添加了一个 RightTapped 事件，如下：

```
privateasyncvoidElement_RightTapped(object sender, RightTappedRoutedEventArgs e)
{
PopupMenu p = newPopupMenu();
p.Commands.Add(newUICommand("31 Days of Windows 8", (command) => { ((Grid)Logo.Parent).Background =
newSolidColorBrush(Colors.Orange); }));
awaitp.ShowForSelectionAsync(GetRect(sender), Placement.Right);
}
```

如上所示，创建了一个简单的 PopupMenu 菜单，并添加了一个 command（执行代码再 lambda 表达式中）然后调用 ShowForSelectionAsync()方法将其显示出来。ShowForSelectionAsync()方法有两个参数：

第一个是从 GetRect()方法获取到的 Rect 值。传递 sender 进去，就可以返回相应的 Rect。

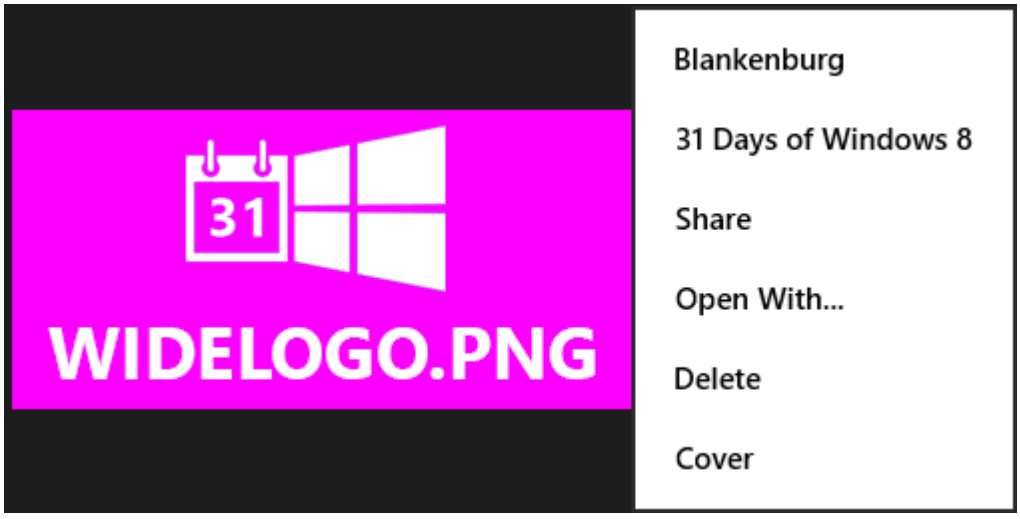
第二个是上下文菜单的位置。在我的示例中，我将其设置为 Placement.Right。

这里有 4 个可选项，不过别随便使用这些。微软关于上下文菜单的指导是这样的：上下文菜单应该出现在被操作 element 的上方，除非上下文菜单掩盖了别的 element，则将上下文菜单放在被操作 element 对象的旁边或者下边是可以接受的。请保持 Windows 8 对用户的统一体验，除非你有其它更好的理由。

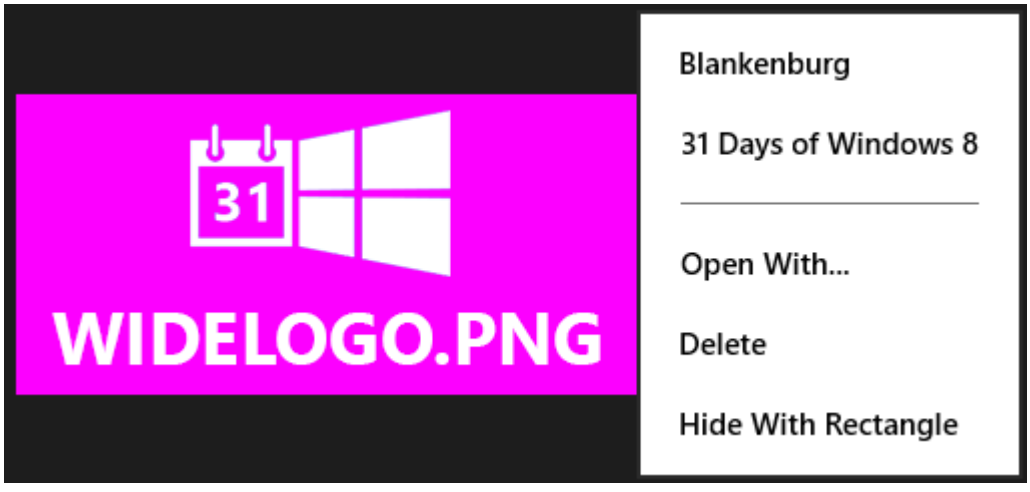
运行上面的代码，可以看到如下上下文菜单的效果：

当我选择选项时，页面的背景色变为 orange 色。如下：

这就是如何显示一个上下文菜单。那么我们可以在菜单中添加多少个 command 呢？答案是 6 个。当添加多余 6 个 command 时，会出现错误。如下图，我之前显示过，是有 6 个 command 的菜单：



最后，你也可以创建一个可选项：UICommandSeparator()：一个简单的水平线代替一个 command。它同样会占据 6 个位置中的一个。所以，当你用它的时候，先考虑好了。下面是相关截图：



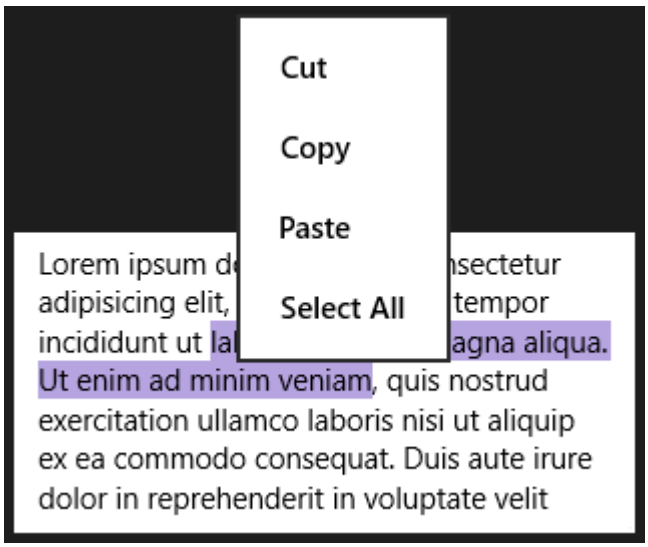
下面是创建水平线的代码：

```
p.Commands.Add(newUICommand("31 Days of Windows 8", (command) => { ((Grid)Logo.Parent).Background = newSolidColorBrush(Colors.Orange); }));
p.Commands.Add(newUICommandSeparator());
```

以上就是所有关于上下文菜单的使用！先现在应该可以在程序中添加上下文菜单了。那么如何与 TextBox 中的一些文字进行交互呢？

1.4. 在 TextBox 中启动上下文菜单

实际上，处理过程与上面的类似，只不过确定上下文菜单出现的位置有点不同。刚开始，当我尝试解决这个问题时，我希望给已经存在的上下文菜单添加 command(默认情况下，在 TextBox 上右键单击，会有系统提供的上下文菜单)：



然而事实证明，这是做不到的。在我的程序中，我来谈谈如何保留原来系统提供的选项，并且在上下文菜单的底部添加一个“Delete”。为了实现这样的效果，首先需要通过创建一个 event handler，在这个 event handler 中，创建一个新的上下文菜单，以取消默认的菜单，然后重新创建所有 item 对应的功能。如下，是我创

建的 event handler：

, and then recreate all of the functionality of the old one.

To do this, we create a new event handler like this:

```
protectedoverridevoidOnNavigatedTo(NavigationEventArgs e)
{
Lipsum.ContextMenuOpening += Lipsum_ContextMenuOpening;
}
```

```
protectedoverridevoidOnNavigatingFrom(NavigatingCancelEventArgs e)
{
Lipsum.ContextMenuOpening -= Lipsum_ContextMenuOpening;
}
```

同时，当离开页面时，我也写了一行代码来移除 event handler。

在这个 event handler 中，我我通过 Handled=true 来取消原来的调用，然后创建自己的上下文菜单，并在适当的地方调用剪贴板(clipboard)。本文只是简单的使用剪贴板，明天，我将用一篇文章专门介绍剪贴板的不同用法。

另外，确定上下文菜单在 TextBox 内部正确的位置，不同于只是确定页面中控件的位置。下面是相关方法 GetTextBoxRect()：

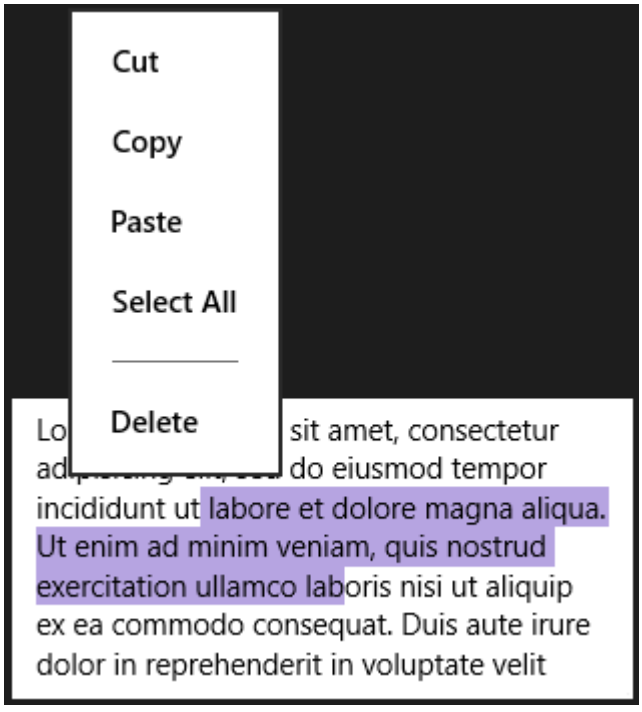
```
privateRectGetTextBoxRect(TextBox t)
{
Rect temp = t.GetRectFromCharacterIndex(t.SelectionStart, false);

GeneralTransform transform = t.TransformToVisual(null);
Pointpoint = transform.TransformPoint(newPoint());
point.X = point.X + temp.X;
point.Y = point.Y + temp.Y;

returnnewRect(point, newSize(temp.Width, temp.Height));
}
```

下面是的方法中，创建了我们自己的 PopupMenu 控件（注意，在这里我使用了一个二级方法来判断是那个 command 被 tap 了 ,而不是使用之前用的的 lambda，这样可读性要更好一点，但是代码会更多更长。）

```
asyncvoidLipsum_ContextMenuOpening(object sender, ContextMenuEventArgs e)
{
e.Handled = true;
TextBox t = (TextBox)sender;

PopupMenu p = newPopupMenu();
p.Commands.Add(newUICommand("Cut", null, 0));
p.Commands.Add(newUICommand("Copy", null, 1));
p.Commands.Add(newUICommand("Paste", null, 2));
p.Commands.Add(newUICommand("Select All", null, 3));
p.Commands.Add(newUICommandSeparator());
p.Commands.Add(newUICommand("Delete", null, 4));

varselectedCommand = awaitp.ShowForSelectionAsync(GetTextBoxRect(t));

if (selectedCommand != null)
      {
String text;
DataPackage d;

switch ((int)selectedCommand.Id)
            {
case 0: //CUT
text = t.SelectedText;
t.SelectedText = "";
                        d = newDataPackage();
d.SetText(text);
Clipboard.SetContent(d);
break;
case 1: //COPY
text = t.SelectedText;
                        d = newDataPackage();
d.SetText(text);
Clipboard.SetContent(d);
break;
case 2: //PASTE
text = awaitClipboard.GetContent().GetTextAsync();
t.SelectedText = text;
break;
case 3: //SELECT ALL
t.SelectAll();
break;
```

```
case 4: //DELETE
t.SelectedText = "";
break;
            }
        }
}
```

下面是运行效果：



1.5. 总结

今天我们学习了如何创建上下文菜单。上下文菜单是非常好的方法：特别是为不可选的 element 提供交互，或者与邻近的 element 进行交互。

点击下面的图片，可以下载本文相关的代码：

明天，我将介绍，在 Windows 8 开发中，可以使用剪贴板的哪些功能。到时候见！

感谢你的阅读！

如果对这篇文章有什么想法想法，可以与破船联系，破船的联系方式在文章开头。

破船