#### 工具条任务

[在工具条中添加任意控件](#_bookmark137)

[在工具条按钮中加入图片](#_bookmark138)

[在按钮中插入分隔符](#_bookmark139)

[将水平工具条的位置调整为垂直](#_bookmark140)

[创建工具条](#_bookmark141)

[以编程方式停靠一个工具条](#_bookmark142)

[增加工具条中图片的尺寸](#_bookmark143)

[调整工具条的外观](#_bookmark144)

[让C1ToolBar显示为Internet Explorer（IE）默认工具条的样式](#_bookmark145)

[让工具条按钮中的图片更明亮](#_bookmark146)

[指定一个停靠/浮动位置](#_bookmark147)

[打开自定义功能](#_bookmark148)

[截断C1ToolBar中的文本](#_bookmark149)

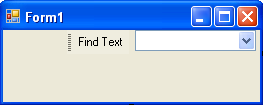
###### 在工具栏中添加任意控件

按以下步骤在工具条中添加任意控件：

1. 在窗体上添加C1CommandDock控件。
2. 拖动一个C1ToolBar控件到C1CommandDock控件内。
3. 右键点击窗体上的C1ToolBar控件。在上下文菜单中选择**Edit**。弹出**Link to Command**编辑器。在**Link to Command**编辑器中按如下设置：
   * **Text**文本框设置为**Find Text:**
   * **Name**设置为**c1CommandControl1**
   * **Command Type**设置为**C1CommandControl**

点击编辑器对话框中的**确认**按钮。

1. 在创建新命令列表框中选择**C1CommandControl。**
2. 在**Properties**下拉列表中选择**c1CommandLink1**，然后设置**ButtonLook**属性值为**Text**。这个工具条按钮显示为文本样式。
3. 在工具箱中选择Windows Form标签，拖动一个**ComboBox**控件到**C1ToolBar**控件的右侧。组合框显示如下图所示：



1. 在**Properties**下拉列表中选择**c1CommandControl1**，然后选中**Control**属性，在属性值的下拉列表中选择**comboBox1**。
2. 编译运行程序。运行时工具条显示如下图所示：



###### 在工具条中加入图片

此主题假定你已经创建了一个工具条按钮。下列步骤演示了如何添加一个图片到工具条按钮中：

1. 在源文件中定位到需要加入图片的位置。
2. 键入下列代码来添加一个图片到你的工具条按钮中。

Visual Basic代码

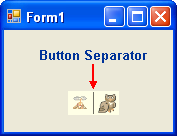
|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| cNew.Image = System.Drawing.Image.FromFile("C:\bitmap\New.bmp") | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| cNew.Image = System.Drawing.Image.FromFile("C:\\bitmap\\New.bmp"); | |

###### 在按钮中插入分隔符

1. 右键点击第二个工具条按钮，然后在上下文菜单中选择**Properties**。
2. 在外观属性类别中设置**Delimiter**属性值为**True**。
3. 保存运行程序。按钮分隔符显示如下图所示：



###### 将水平工具条的位置调整为垂直

按以下步骤使用**Horizontal**属性将工具条位置从水平改变为垂直。

1. 创建一个工具条。
2. 使用如下代码使工具条显示为垂直：

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| toolbar.Horizontal = False | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| toolbar.Horizontal = False; | |

###### 创建工具条

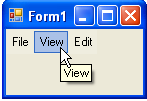
你可以在设计时或者通过代码来创建一个工具条。点击下列任何一个链接以展开通过设计器或者代码方式的步骤。

在设计时创建一个工具条

按如下步骤使用**Link to Command**编辑器创建一个C1ToolBar。

1. 在窗体中加入一个**C1ToolBar**控件，然后右键点击**C1ToolBar**控件并且从上下文菜单中选择**Edit**。弹出被选择工具条的**Link to Command**编辑器。
2. 在**Text**文本框中输入**File**，然后点击**OK**按钮。
3. 在**Properties**下拉框中选择**c1CommandLink1**。设置这个File工具条按钮对应的CommandLink的**ButtonLook**属性值为**Text**。这个File工具条按钮以文本方式显示。
4. 右键点击C1ToolBar控件并且在上下文菜单中选择**Append Item**。将在当前的CommandLink后添加一个新的命令连接。弹出**Link to Command**编辑器。
5. 在**Text**文本框中输入**Edit**，然后点击**OK**按钮。
6. 在**Properties**下拉列表中选择**c1CommandLink2**。设置这个**Edit**工具条按钮对应的CommandLink的**ButtonLook**属性值为**Text**。
7. 右键点击Edit工具条按钮然后在上下文菜单中选择**Insert Item**。将在其后添加一个新的命令连接。弹出**Link to Command**编辑器。
8. 在**Properties**下拉列表中选择**c1CommandLink3**。设置这个View工具条按钮对应的CommandLink的**ButtonLook**属性值为**Text**。
9. 编译运行程序。

在运行时，工具条显示如下图所示：



以编程方式的创建工具条

按以下步骤以编程方式创建一个包含文本按钮的C1ToolBar：

1. 添加C1.Win.C1Command引用到项目中。
2. 在代码中申明命名空间，然后添加一个用来容纳工具条的**C1CommandHolder**。

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| Imports C1.Win.C1Command  Dim ch As C1CommandHolder = C1CommandHolder.CreateCommandHolder(Me) | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| using C1.Win.C1Command;  C1CommandHolder ch = C1CommandHolder.CreateCommandHolder(this); | |

1. 创建一个新的C1ToolBar，然后将它添加到窗体中。

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| Dim tb As New C1ToolBar() Me.Controls.Add(tb) | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| C1ToolBar tb = new C1ToolBar(); this.Controls.Add(tb); | |

1. 将C1CommandHolder分派给C1ToolBar，然后创建一个新的工具条命令。新添加的命令的名字设置为**File**。

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| tb.CommandHolder = ch  Dim cFile As New C1Command() cFile.Text = "File" | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| tb.CommandHolder = ch;  C1Command cFile = new C1Command(); cFile.Text = "File"; | |

1. 为新命令创建一个新的CommandLink，然后将新CommandLink加入到工具条中。

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| Dim cl As C1CommandLink  cl = New C1CommandLink(cFile) tb.CommandLinks.Add(cl) | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| C1CommandLink cl = new C1CommandLink(cFile); tb.CommandLinks.Add(cl); | |

1. 让这个CommandLink显示为文本样式，然后创建另外一个名字为**View**的命令。

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| cl.ButtonLook = ButtonLookFlags.Text Dim cView As New C1Command() cView.Text = "View" | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| cl.ButtonLook = ButtonLookFlags.Text; C1Command cView = new C1Command(); cView.Text = "View"; | |

1. 为新的View命令创建一个新的CommandLink，然后将它加入到工具条中。

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| cl = New C1CommandLink(cView) tb.CommandLinks.Add(cl) | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| cl = new C1CommandLink(cView); tb.CommandLinks.Add(cl); | |

1. 让这个**View** CommandLink显示为文本样式，然后创建另外一个名字为**Edit**的命令。

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| cl.ButtonLook = ButtonLookFlags.Text Dim mEdit As New C1Command() mEdit.Text = "Edit" | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| cl.ButtonLook = ButtonLookFlags.Text; C1Command mEdit = new C1Command(); mEdit.Text = "Edit"; | |

1. 为新的Edit命令创建一个新的CommandLink，然后将它添加到工具条中。

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| cl = New C1CommandLink(mEdit) tb.CommandLinks.Add(cl) | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| cl = new C1CommandLink(mEdit); tb.CommandLinks.Add(cl); | |

1. 让Edit CommandLink显示为文本样式。

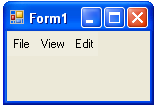
Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | Copy Code复制代码 |
| cl.ButtonLook = ButtonLookFlags.Text | |

C#代码

|  |  |
| --- | --- |
| C# | Copy Code复制代码 |
| cl.ButtonLook = ButtonLookFlags.Text; | |

1. 保存后运行程序。工具条显示如下图所示：



###### 以编程的方式停靠一个工具条

按以下步骤以使用C1CommandDock让C1ToolBar可停靠：

**注**：此主题假定你已使用代码创建了一个C1ToolBar。

1. 创建一个新的C1CommandDock并且赋值给dock变量。

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| Dim dock As New C1CommandDock() | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| C1CommandDock dock = new C1ComandDock(); | |

1. 将C1CommandDock控件添加到窗体中。

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| Me.Controls.Add(dock) | |

C# C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| this.Controls.Add(dock); | |

###### 增加工具条中图片的尺寸

为了显示更大的图标或位图，需要增加工具条按钮的尺寸。这将相应的增加图片的尺寸。可以使用工具条的**MinButtonSize**属性来实现。**MinButtonSize**属性的默认尺寸是24 pixels。

按如下步骤在设计时增加工具条按钮的尺寸：

* 打开**C1Toolbar**任务菜单，在C1ToolBar的MinButtonSize文本框中输入**50 pixels**。

或者

* 在属性窗口中设置C1ToolBar的**MinButtonSize**属性值为**40 pixels**。

下面图片中工具条的最小按钮尺寸被增加到40像素。



###### 调整工具条的外观

1. 在**Properties**下拉列表中选择C1ToolBar。
2. 展开C1ToolBar对象的**Font**属性。设置下列字体属性：

Font.Name = **Arial**

Font.Size = **9**

1. 设置下列C1ToolBar属性：

[BackColor](#_bookmark1182) = **#0066CC** [BackHiColor](#_bookmark1183) = **Pale Turquoise** ForeColor = **White** [ForeHicolor](#_bookmark1203) = **#003399**

1. 展开C1ToolBar的**Border**属性。设置下列边框属性：

[Border](#_bookmark1185).Light Color = **Pale Turquoise**

[Border](#_bookmark1185).Style = **Outset**

[Border](#_bookmark1185).Width = **3**

在运行时工具条显示如下图所示：



**让C1ToolBar显示为Internet Explorer（IE）默认工具条的样式**

此主题演示了如何使用**ButtonLookFlags**枚举使C1ToolBar显示为IE的默认工具条的样式。**ButtonLookFlags**枚举包含的值有：*Text*，*TextAndImage*，*Image*和*Default*。这些值让你有能力改变工具条按钮的样式，显示为文本，文本和图片，或者只是图片。

下列代码演示了如何使C1ToolBar显示为IE的默认工具条的样式。下列使用**ButtonLookFlags**枚举中所有的值（*TextAndImage*, *Image*, *Default*和*Text*）来让C1ToolBar显示为类似IE中工具条的样式：

1. 创建第一个工具条按钮并且显示为图片样式。下列代码演示了如何用**ButtonLookFlags**枚举让工具条按钮显示为图片：

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| Dim ch As C1CommandHolder = C1CommandHolder.CreateCommandHolder(Me) Dim tb As New C1ToolBar()  Me.Controls.Add(tb) tb.CommandHolder = ch  Dim cNew As New C1Command() Dim cl As C1CommandLink cNew.Text = "New"  cNew.Image = System.Drawing.Image.FromFile("C:\Images\New.bmp") cl = New C1CommandLink(cNew)  tb.CommandLinks.Add(cl)  'Use the ButtonLookFlags enumeration to make the toolbar button appear as an image  cl.ButtonLook = ButtonLookFlags.Image | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| C1CommandHolder ch = C1CommandHolder.CreateCommandHolder(this); C1ToolBar tb = new C1ToolBar(); | |

this.Controls.Add(tb); tb.CommandHolder = ch;

C1Command cNew = new C1Command();

cNew.Text = " New ";

cl = new C1CommandLink(cNew); tb.Commandlinks.Add(cl);

//Use the ButtonLookFlags enumeration to make the toolbar button appear as an image

cl.ButtonLook = ButtonLookFlags.Image;

1. 创建第二个工具条按钮并且使其显示为文本和图片样式。下列代码演示了如何使用**ButtonLookFlags**枚举使工具条按钮显示为文本和图片样式。

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| Dim cOpen As New C1Command() cOpen.Text = "Open"  cl = New C1CommandLink(cOpen)  cOpen.Image = System.Drawing.Image.FromFile("C:\Images\FileOpen.bmp") tb.CommandLinks.Add(cl)  'Use the ButtonLookFlags enumeration to make the toolbar button appear as text and an image  cl.ButtonLook = ButtonLookFlags.TextAndImage | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| C1Command cOpen = new C1Command(); cOpen.Text = "Open";  cl = new C1CommandLink(cOpen);  cOpen.Image = System.Drawing.Image.FromFile("C:\Images\FileOpen.bmp"); tb.CommandLinks.Add(cl);  //Use the ButtonLookFlags enumeration to make the toolbar button appear as text and an image | |

cl.ButtonLook = ButtonLookFlags.TextAndImage;

1. 创建第三个工具条按钮并且使其显示为默认样式。下列代码演示了如何使用**ButtonLookFlags**枚举使工具条安娜显示为默认样式。

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| Dim cSave As New C1Command() cSave.Text = "Save"  cl = New C1CommandLink(cSave)  tb.CommandLinks.Add(cl)  cSave.Image = System.Drawing.Image.FromFile("C:\Images\FileSave.bmp") 'Use the ButtonLookFlags enumeration to make the toolbar button appear as default  cl.ButtonLook = ButtonLookFlags.Default | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| C1Command cSave = new C1Command(); cSave.Text = "Save";  cl = new C1CommandLink(cSave);  tb.CommandLinks.Add(cl);  cSave.Image = System.Drawing.Image.FromFile("C:\Images\FileSave.bmp")  //Use the ButtonLookFlags enumeration to make the toolbar button appear as default  cl.ButtonLook = ButtonLookFlags.Default; | |

1. 创建第四个工具条按钮并且使其显示为文本样式。下列代码演示了如何使用**ButtonLookFlags**枚举使工具条按钮显示为文本样式：

Visual Basic代码

Visual Basic 复制代码

Dim cFavorites As New C1Command() cFavorites.Text = "Favorites"

cl = New C1CommandLink(cFavorites) tb.CommandLinks.Add(cl)

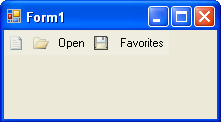
'Use the ButtonLookFlags enumeration to make the toolbar button appear as text

cl.ButtonLook = ButtonLookFlags.Text

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| C1Command cFavorites = new C1Command(); cFavorites.Text = "Save";  cl = new C1CommandLink(cFavorites); tb.CommandLinks.Add(cl);  //Use the ButtonLookFlags enumeration to make the toolbar button appear as text  cl.ButtonLook = ButtonLookFlags.Text; | |

1. 保存后运行程序。C1ToolBar显示为类似下图中的工具条。



**注**：你的工具条中的图片和上图中工具条中的图片会有不同。四个工具条按钮显示的样式分别为图片，文本和图片，默认，图片。

###### 让工具条按钮中的图片更明亮

**SmoothImages**属性值默认为**True**，非选择状态的图片以平滑的方式显示。如果设置这个属性值为**False**，非选择状态的图片以更明亮的方式显示。

如下表格说明了**SmoothImages**属性设置为**True**和**False**时不同的显示效果：

|  |  |
| --- | --- |
| 属性 | 工具条图片 |
| **False** |  |
| **True** |  |

###### 指定一个停靠/浮动位置

此主题演示了如何使用代码设置C1CommandDock的位置。使用如下代码将工具条停靠在上方。

Visual Basic代码

|  |  |
| --- | --- |
| Visual Basic | 复制代码 |
| dock.Dock = DockStyle.Top | |

C#代码

|  |  |
| --- | --- |
| C# | 复制代码 |
| dock.Dock = DockStyle.Top; | |

**注**：如果想让工具条停靠在下方，左方或者右方，改变枚举值*Top*为相应的位置：*Bottom*，*Left*或者*Right*。

###### 打开自定义功能

设置**CustomizeButton**属性值为**True**可以打开自定义功能，允许用户在运行时自定义工具条。参照[动态自定义工具条](#_bookmark74)获取这个功能的更多信息。

###### 截断C1ToolBar中的文本

按以下步骤使用**WrapText**属性来截断C1ToolBar中的文本。

* 1. 设置**ButtonWidth**属性为一个大于0的值。
  2. 设置**WrapText**属性值为**True**。