**WPF扩展库文档**

Ping9719.WpfEx

V 0.0.6

目录

[一、 开始使用 3](#_Toc116719176)

[二、 转换器 4](#_Toc116719177)

[1. Enum2BoolConverter（枚举转布尔） 4](#_Toc116719178)

[2. Enum2IntConverter（枚举转数字） 4](#_Toc116719179)

[3. Enum2StrConverter（枚举转字符串） 4](#_Toc116719180)

[4. List2StrConverter（集合转字符串） 4](#_Toc116719181)

[三、 控件 5](#_Toc116719182)

[5. UserControlBase（用户控件扩展） 5](#_Toc116719183)

[四、 窗体 6](#_Toc116719184)

[五、 绑定与命令 7](#_Toc116719185)

[6. 绑定 7](#_Toc116719186)

[7. 绑定代码片段 7](#_Toc116719187)

[8. 命令 8](#_Toc116719188)

[(1) 不带参数命令 8](#_Toc116719189)

[(2) 带参数命令 9](#_Toc116719190)

[六、 其他 10](#_Toc116719191)

[9. 将WPF控件保存为图片 10](#_Toc116719192)

[七、 升级记录 11](#_Toc116719193)

# 开始使用

引用资源，在文件（App.xaml）中

|  |
| --- |
| <Application.Resources>  <ResourceDictionary>  <ResourceDictionary.MergedDictionaries>  <ResourceDictionary Source="pack://application:,,,/Ping9719.WpfEx;component/Theme.xaml"/>  </ResourceDictionary.MergedDictionaries>  </ResourceDictionary>  </Application.Resources> |

名称空间

|  |
| --- |
| xmlns:pi="https://github.com/ping9719/wpfex" |

# 转换器

## Enum2BoolConverter（枚举转布尔）

将枚举转为布尔，一般用于单选框。

|  |
| --- |
| <RadioButton Content="男" IsChecked="{Binding Sex, Converter={StaticResource Enum2BoolConverter}, ConverterParameter='man'}"/> |

## Enum2IntConverter（枚举转数字）

将枚举转为数字

## Enum2StrConverter（枚举转字符串）

将枚举转为字符串（支持Description特性）

## List2StrConverter（集合转字符串）

将数组、集合按照默认逗号分隔转为字符串

# 控件

## UserControlBase（用户控件扩展）

属性

|  |  |
| --- | --- |
| IsLoadedVisible | 是否已加载并显示界面 |
| IsInDesignMode | 是否处于设计模式 |

方法

|  |  |
| --- | --- |
| \ |  |

事件

|  |  |
| --- | --- |
| LoadedVisibleFirst | 首次加载并显示控件时发生 |
| LoadedVisible | 加载并显示控件时发生 |

# 窗体

无

# 绑定与命令

## 绑定

在XMAL中

|  |
| --- |
| <TextBlock Text="{Binding Var}"></TextBlock> |

在cs文件中

|  |
| --- |
| UcHomeVModel ucHomeVModel = new UcHomeVModel();  public UcHome()  {  InitializeComponent();  this.DataContext = ucHomeVModel;  } |

在ViewModel 文件夹中

|  |
| --- |
| using Ping9719.WpfEx.Mvvm;  public class UcHomeVModel : BindableBase  {  private string var;  public string Var  {  get { return var; }  set { SetProperty(ref var, value); }  }  } |

## 绑定代码片段

工具--代码片段管理器--Visual C#--添加

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <CodeSnippets xmlns="http://schemas.microsoft.com/VisualStudio/2005/CodeSnippet">  <CodeSnippet Format="1.0.0">  <Header>  <Title>propmvvm</Title>  <Shortcut>propmvvm</Shortcut>  <Description>属性和支持MVVM模式字段的代码片段</Description>  <Author>夏诗评</Author>  <SnippetTypes>  <SnippetType>Expansion</SnippetType>  </SnippetTypes>  </Header>  <Snippet>  <Declarations>  <Literal>  <ID>type</ID>  <ToolTip>属性类型</ToolTip>  <Default>int</Default>  </Literal>  <Literal>  <ID>property</ID>  <ToolTip>属性名</ToolTip>  <Default>MyProperty</Default>  </Literal>  <Literal>  <ID>field</ID>  <ToolTip>支持此属性的变量</ToolTip>  <Default>myVar</Default>  </Literal>  </Declarations>  <Code Language="csharp"><![CDATA[private $type$ $field$;  public $type$ $property$  {  get { return $field$; }  set { SetProperty(ref $field$, value); }  }  $end$]]>  </Code>  </Snippet>  </CodeSnippet>  </CodeSnippets> |

## 命令

### 不带参数命令

在XMAL中

|  |
| --- |
| <Button Command="{Binding MyCommand}" CommandParameter="abc" Content="按钮"/> |

在ViewModel 文件夹中

|  |
| --- |
| public class MainWindowViewModel : BindableBase  {  public ICommand MyCommand { get => new DelegateCommand<string>(My); }  //执行方法  public void My(string obj)  {  //code  }  } |

### 带参数命令

在XMAL中

|  |
| --- |
| <Button Command="{Binding MyCommand}" Content="按钮"/> |

在ViewModel 文件夹中

|  |
| --- |
| public class MainWindowViewModel : BindableBase  {  public ICommand MyCommand { get => new DelegateCommand(My); }  //执行方法  public void My()  {  //code  }  } |

# 其他

## 将WPF控件保存为图片

|  |  |
| --- | --- |
| void WpfHelp.SaveToImg(this FrameworkElement visual, string fileName) | 保存为本地文件 |
| MemoryStream SaveToImg(this FrameworkElement visual) | 保存到内存流 |

# 升级记录