WPF

项目结构

- XAML文档实际上也是类的一部分，最后也会被编译为C#代码
- XAML下的C#代码构成类似于前后端的关系
- ```
  App实际上是程序的入口
  StartupUri="MainWindow.xaml"
  ```

布局

- StackPanel

  > StackPanel主要用于垂直或水平排列元素、在容器的可用尺寸内放置有限个元素，元素的尺寸总和(长/高)不允许超过StackPanel的尺寸, 否则超出的部分不可见。

- WrapPanel

  > WrapPanel默认排列方向与StackPanel相反、WrapPanel的Orientation默认为Horizontal。
  > WrapPanel具备StackPanel的功能基础上具备在尺寸变更后自动适应容器的宽高进行换行换列处理。

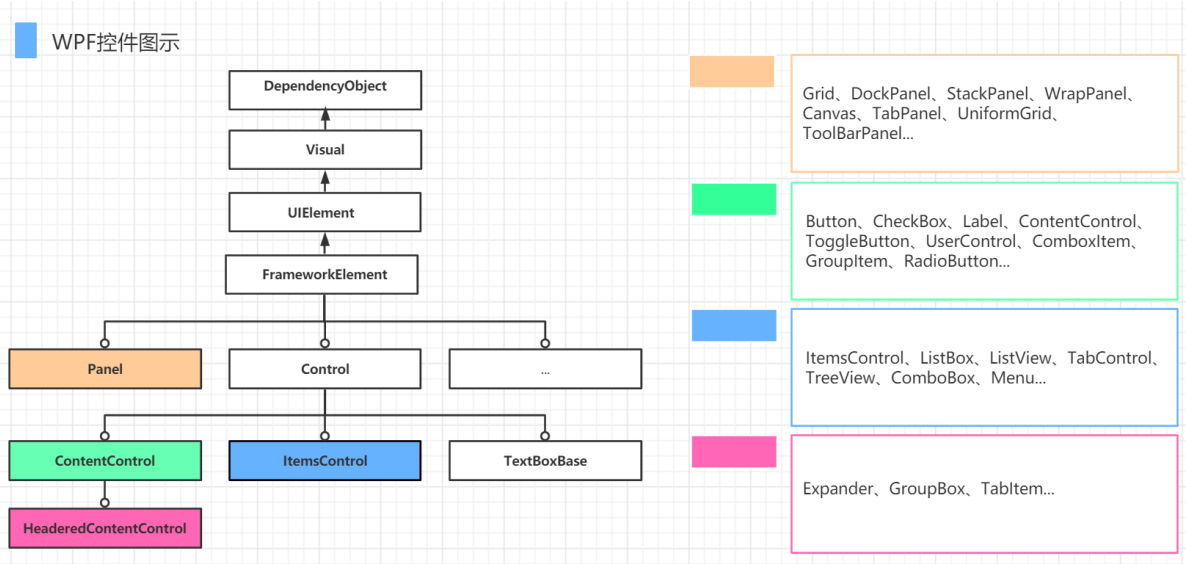
- DockPanel

  > 默认DockPanel中的元素具备DockPanel.Dock属性, 该属性为枚举具备: Top、 Left、 Right、 Bottom。
  > 默认情况下, DockPanel中的元素不添加DockPanel.Dock属性, 则系统则会默认添加 Left。
  > DockPanel有一个LastChildFill属性, 该属性默认为true, 该属性作用为, 当容器中的最后一个元素时, 默认该元素填充DockPanel所有空间。

- Grid

  > Grid具备分割空间的能力。RowDefinitions / ColumnDefinitions 用于给Grid分配行与列。
  > ColumnSpan / RowSpan 则用于设置空间元素的 跨列与阔行。

控件


  ```

- ContentControl 类

  > 内容属性为Content，只能设置一次，但可以嵌套

- ItemsControl 类

  > 此类控件大多数属于显示列表类的数据、设置数据源的方式一般通过 ItemSource 设置。

- HeaderedContentControl 类

  > 相对于ContentControl来说、这类控件即可设置Content, 还有带标题的Header。
  > 像比较常见的分组控件GroupBox、TabControl子元素TabItem、它们都是具备标题和内容的控件。

- 常用控件
  - TextBlock: 用于显示文本, 不允许编辑的静态文本。
  - TextBox: 用于输入/编辑内容的控件、作用与winform中TextBox类似, Text设置输入显示的内容。
  - Button: 简单按钮、Content显示文本、Click可设置点击事件、Command可设置后台的绑定命令。
  - ComboBox: 下拉框控件, ItemSource设置下拉列表的数据源, 也可以显示设置。

样式（重点：触发器）

- 可设置样式
  - 字体(FontFamily)
  - 字体大小(FontSize)
  - 背景颜色(Background)
  - 字体颜色(Foreground)
  - 边距(Margin)
  - 水平位置(HorizontalAlignment)
  - 垂直位置(VerticalAlignment)
- 写入位置
  - 写入Windows.Resources
  - 写入Application.Resources（推荐）
  - 样式中指定key，和TargetType，后文在相应控件中用代码指定：

    ```
    <Style = "{StaticResources key}"/>
    ```

  - **实例优先级高于样式**
- 触发器
  - Triggers指定方法

    ```
    <style.triggers>
        <trigger property = "" Value = "">
            <setter property = "" value = "" target = ""/>
        </trigger>
    </style.triggers>
    ```

  - MultiTrigger：指定Conditions，在都满足的情况下，触发指定的Setters
  - EventTrigger：指定RoutedEvent和Actions
  - DataTrigger：用法类似

模板

- 控件模板：指定控件的样式

```
<Setter Property="Template">
    <Setter.Value>
        <ControlTemplate TargetType="{x:Type ListBoxItem}">
            <Grid>
                <Border x:Name="back"/>
                <Border x:Name="line" Margin="5 10 5 10 "/>
                <ContentPresenter/>
            </Grid>
            <ControlTemplate.Triggers>
                <Trigger Property="IsMouseOver" Value="True">
                    <Setter Property="Background" Value="#F0F6F6"
TargetName="back"/>
                </Trigger>
                <Trigger Property="IsSelected" Value="True">
                    <Setter Property="Foreground" Value="{Binding Color}"/>
                    <Setter Property="FontWeight" Value="Bold"/>
                    <Setter Property="Background" Value="{Binding Color}"
TargetName="back"/>
                    <Setter Property="Opacity" Value="0.05"
TargetName="back"/>
                    <Setter Property="BorderThickness" Value="2 0 0 0"
TargetName="line"/>
                    <Setter Property="BorderBrush" Value="{Binding Color}"
TargetName="line"/>
                </Trigger>
            </ControlTemplate.Triggers>
        </ControlTemplate>
    </Setter.Value>
</Setter>
```

  - TemplateBinding：与外部属性联系
  - ControlTemplate.Triggers：触发器设置
  - 可独立于样式声明模板
- 数据模板（DataTemplate）
  - CellTemplate：适用于DataGrid

```
<DataGridTemplateColumn Header="操作" Width="100" >
    <DataGridTemplateColumn.CellTemplate>
        <DataTemplate>
            <StackPanel Orientation="Horizontal"
VerticalAlignment="Center" HorizontalAlignment="Left">
                <Button Content="编辑"/>
                <Button Margin="8 0 0 0" Content="删除" />
            </StackPanel>
        </DataTemplate>
    </DataGridTemplateColumn.CellTemplate>
</DataGridTemplateColumn>
```

  - DataTemplate

```
 <Window.Resources>
```

```
        <DataTemplate x:Key="comTemplate">
            <StackPanel Orientation="Horizontal" Margin="5,0">
                <Border Width="10" Height="10" Background="{Binding
Code}"/>
                <TextBlock Text="{Binding Code}" Margin="5,0"/>
            </StackPanel>
        </DataTemplate>
    </Window.Resources>
    <Grid>
        <StackPanel Orientation="Horizontal"
HorizontalAlignment="Center">
            <ComboBox Name="cob" Width="120" Height="30" ItemTemplate="
{StaticResource comTemplate}"/>
            <ListBox Name="lib" Width="120" Height="100" Margin="5,0"
 ItemTemplate="{StaticResource comTemplate}"/>
        </StackPanel>
    </Grid>
```

- 关于模板的用法课上讲的不好，具体请参考：

  https://www.cnblogs.com/zh7791/p/11449492.html

  https://www.cnblogs.com/zh7791/p/11421386.html

Binding（绑定）

- 绑定源
  1. 绑定至元素：指定ElementName与Path
  2. 绑定至Source/RelativeSource，暂不介绍
  3. 绑定至DataContext： 从当前的元素树向上查找到第一个非空的DataContext属性为源对象。

     方法：

     - 前端：

       ```
       <Window.DataContext>
           <local:Mainviewmodel/>
       </Window.DataContext>
       ```

     - 后台

       ```
       this.DataContext = new Mainviewmodel()
       ```

- 绑定方法
  - OneWay：由数据源到显示
  - OneWayToSource ： 由显示到数据源
  - TwoWay：双向变化
  - OneTime（根据第一次源的变化来绑定）
  - default：默认

MVVM（视图通过绑定与UI分离）

- Model-View-ViewModel，目的是分离视图（view）与模型（model），降低代码的耦合程度

> MVVM旨在利用WPF中的数据绑定函数，通过从视图层中几乎删除所有GUI代码（代码隐藏），更好地促进视图层开发与模式其余部分的分离。不需要用户体验（UX）开发人员编写GUI代码，他们可以使用框架标记语言（如XAML），并创建到应用程序开发人员编写和维护的视图模型的数据绑定。角色的分离使得交互设计师可以专注于用户体验需求，而不是对业务逻辑进行编程。

- 常用框架：Prism（重点介绍），MVVMlight，微软框架
- 绑定事件
- 实现属性的动态绑定

案例分析：下载器

> 界面设计；下载API直接给出，略作讲解；MVVM模式实现

- 文件下载器
  1. 将下载函数放在后台，直接用 Click="..." 绑定
  2. 让Viewmodel与显示分离，用MVVM模式绑定响应函数

  > 使用现有框架：添加Prism.Core

  3. 实现检测文件名，生成下载路径
  4. 用文件管理器窗口选择下载地址
  5. 使用多线程进行下载，不影响窗口正常操作
  6. 实现进度条管控，利用绑定改进（绑定的实时刷新需要用接口通知前端数据发生变化）
  7. 实现取消下载的操作，并同时删除目录下的文件
  8. 实现显示下载速率的操作（留作作业）
     p.s. 需要解决的问题还有很多，线程可能存在的并发死锁，下载期间对下载器的操作，暂停，压缩文件检测不出来等

作业

- 小试身手（4分）
  - 创建一个TextBlock，实现下载状态的显示，要求使用绑定实现。
  - 没有下载任务或下载被取消时显示"Free"；下载中显示"Downloading..."；下载完成后显示"Complete"。
  - 允许对设计要求做出合理的调整或改进，但需要阐述理由
- 略有难度（4+1分）
  1. 使用TextBlock，固定传输速率单位，实现下载速度的显示，要求使用绑定实现（4分）

     > 一种可行的思路：实例化一个Timer定时器，每过一段时间扫描一次下载进度（对应已下载Byte数），统计本次扫描到上次扫描的下载字节数，除以间隔时间即得下载速率

  2. 实现下载速度单位的自适应（1分）

     自动选择适合的下载速度计量单位，要求取值合理（即显示的数值避免过小或过大，例如一定得小于1024）
- 您是大佬（1分）
  1. 利用课上所学知识，对界面设计向符合自身审美的方向进行改进，颜值至上（0.5分）
  2. 服务器实战：给定地址，自动下载目录下所有文件（0.5分）
- 受我一拜！（0分）

- 创建一个Button，并使其具有暂停下载的功能（实际上是断点续传）

References

https://www.cnblogs.com/zh7791/category/1528742.html （来自博客园大佬的教程，建议深度学习orz）