　在WPF中有两种树:逻辑树（Logical Tree）和可视树（Visual Tree），XAML是表达WPF的一棵树。逻辑树完全是由布局组件和控件构成。

一个简单的程序来观察下逻辑树与视觉树

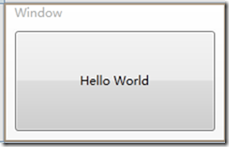
<Window x:Class="WpfApplication1.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

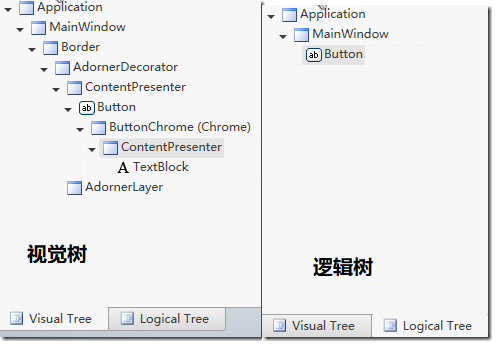
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">

<Button Content="Hello World" Width="200" Height="100"/>

</Window>



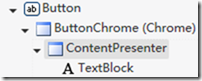
**图2-1**上代码呈现的元素树结构如下



**图2-2 左侧为视觉树,右侧为逻辑树**

**我们看出以下特性**

* WPF启动程序的根元素均为Application
* 逻辑树与XAML的布局结构是相同的
* 视觉树是根据控件的模板来呈现的,我们很难猜测视觉树的结构,因为控件还可以自定义模板

我们可以看到逻辑树上的元素一个视觉树局部元素的一个组合,比如Button,由三个元素组成,ButtonChrome,ContentPresenter,TextBlock  


**3.与ASP.NET服务器控件比较(控件为逻辑树,HTML为视觉树)**

ASP.NET的服务器控件是一个逻辑树,一个服务器控件由若干的HTML组成,但我们知道我们是无法在服务器端知道HTML代码的,即ASP.NET没有视觉树的概念.

**4.与JavaScript客户端控件比较(一个根逻辑树,HTML为视觉树)**

如果你用过jQuery等前端js库的话,比如Tab的运用,一句简单的话就可以创建一堆HTML出来,Tab本身则是一个控件,但其本身没有逻辑树的概念  
以上两者是web服务器端与客户端的比较,由于两者状态无法共享,在交互方面没有客户端来的方便.所以WPF的逻辑树与视觉树概念并不是新的概念.

**5.组装控件**

为WPF创建一个新控件是非常简单的,一般有以下两种方式  
1.采用用户控件

<UserControl x:Class="WpfApplication1.LoginView"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="300" d:DesignWidth="300">

<Grid>

<Label Content="用户名:" Height="28" HorizontalAlignment="Left" Margin="56,89,0,0" Name="label1" VerticalAlignment="Top" />

<Label Content="密 码:" Height="28" HorizontalAlignment="Left" Margin="56,134,0,0" Name="label2" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="111,94,0,0" Name="textBox1" VerticalAlignment="Top" Width="120" />

<PasswordBox Height="23" HorizontalAlignment="Left" VerticalAlignment="Top" Width="120" Margin="111,134,69,146"></PasswordBox>

<Button Content="确定" Height="23" HorizontalAlignment="Left" Margin="156,179,0,0" Name="button1" VerticalAlignment="Top" Width="75" />

</Grid>

</UserControl>



2.采用模板

<ContentControl>

<ContentControl.ContentTemplate>

<DataTemplate>

<Grid>

<Label Content="UserName:" Height="28" HorizontalAlignment="Left" Margin="56,89,0,0" Name="label1" VerticalAlignment="Top" />

<Label Content="Password:" Height="28" HorizontalAlignment="Left" Margin="56,134,0,0" Name="label2" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="111,94,0,0" Name="textBox1" VerticalAlignment="Top" Width="120" />

<PasswordBox Height="23" HorizontalAlignment="Left" VerticalAlignment="Top" Width="120" Margin="111,134,69,146"></PasswordBox>

<Button Content="Summit" Height="23" HorizontalAlignment="Left" Margin="156,179,0,0" Name="button1" VerticalAlignment="Top" Width="75" />

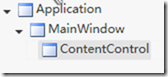
</Grid>

</DataTemplate>

</ContentControl.ContentTemplate>

</ContentControl>

采用模板的时候,逻辑树将变得更少,视觉树将保持不变



**注意:不要将模板内的控件纳入逻辑树范围内,否则你会很失望地无法找到模板内部的元素.**