第 10 章

布局——药师的桃花岛

> 郭靖见她（黄蓉）在花丛中东一转西一晃，霎时不见了影踪。急忙追去，只奔出十余丈远，立时就迷失了方向。只见东南西北都有小径，却不知走向哪一处好。他走了一阵，似觉又回到了原地。想起在归云庄之时，黄蓉曾说那庄子布置虽奇，却哪及桃花岛阴阳开阖、乾坤倒置之妙。这一迷路，若是乱闯，定然只有越走越糟。于是坐在一株桃树之下，只待黄蓉来接。
>
> ——《射雕英雄传》：“第十六回　九阴真经”[1]

这是郭靖初上桃花岛的一段，黄岛主按照五行生克和阴阳八卦的变化来布置桃花岛，因此常人无法近得桃花岛半步。

WPF 的布局也有这样的类似巧妙之处，不懂得其中奥秘是寸步难行。一旦悟透其间相生相克的道理，则如履平地。为了对布局有一个感性认识，我们为木木同学量身定做了一个现代版黄药师出题招婿故事，故事的名字叫做“布局巧设桃花岛，木木憨娶俏黄蓉”。

（1）憨木木误闯桃花宝岛。
（2）老顽童试解桃花玄机。
（3）黄岛主演绎布局精妙。
（4）接下来做什么。

10.1　憨木木误闯桃花宝岛

话说桃花岛软件公司药师有两件烦心事：一是公司技术集体转型，从 WinForm 集体转型到 WPF 急需 WPF 专业型人才；二是药师丧妻之后与女儿蓉儿相依为命，对她宠爱无比。眼看已经出落成十八九的大姑娘，但甚是娇纵，毫无规矩，希望找个青年才俊将其许配。

药师是个聪明之人，一摸胡须想何不将招聘技术人员和招婿合为一起，这样将来也好继承我桃花岛软件公司的大好基业。于是药师大笔一挥，写下了如此这般的招良才贤婿的广告：

小女蒲柳弱质，性又顽劣，原难侍奉君子。有道男大当婚，女大当嫁。老夫愿得一良才贤婿，继承家业，共享天伦之乐。

药师的招婿广告贴出之后，出现了一个奇特的景象。一时间桃花岛外熙熙攘攘，恍若闹市；桃花岛上寂然无声，门可罗雀。还是因为岛上奇门八卦之阵，青年才俊们都近不得岛半步。他们知道桃花

岛凶险，都不愿意打头阵，害怕错失良机。于是他们的目光集中在倚在墙角的那个人身上——木木。所有人都认为论长相和才智，再没有一个人比木木差了。因此不妨让木木一试，既不用担心木木会娶到黄蓉，也可以多了解桃花岛的情况。于是欧阳克公子不由木木分说，抓住他的领脖，就将他甩向了桃花岛。

木木醒来已是夜深，忽听到一阵箫声。似浅笑，似低诉，柔靡万端。木木不由痴了，打从娘胎出来，第一次听到如此这般的天籁之音。正自沉吟，忽听得前面发出一阵急促喘气之声，正是一人盘膝而坐。这时那洞箫声情致飘忽，缠绵宛转。便似一个女子一会儿叹息，一会儿呻吟，一会儿又软语温存，柔声叫唤。木木年纪尚小，对男女之事不甚了了。听到箫声时感应甚淡，听了也不以为意。但对面那人却是气喘愈急，听他呼吸声真是痛苦难当，正拼了全力来抵御箫声的诱惑。

这时木木有些害怕，为了给自己壮胆，他不由跟着箫声哼了一首周杰伦的“七里香”。木木那五音不全，还走调的歌曲似乎是箫声的劲敌，立刻打破了箫声的神秘气氛。眼看那人作势便待跃起，听到木木的《七里香》，心中一静，重新盘膝而坐，闭目运功。过了良久，月光从花树中照射下来。木木才看清那人面容，须发苍然，并未全白，原来此人正是传说中的老顽童——周伯通。

10.2 老顽童试解桃花玄机

周伯通微微笑了笑，说道：“你上岛是为招婿而来？”。木木有些不好意思了，说到：“前辈，我是过来看看热闹的，但是没想到被人给丢了上来。”老顽童又仔细打量了木木一下，掩口而笑：“黄老邪怎么可能看上你，阔鼻大耳，身高不足 1.7 米。走吧，我带你去看看一些好玩意。”不由分说，又抓住木木的领脖，往桃花岛的最高峰奔去。

约摸一柱香的工夫，周伯通和木木登上了桃花岛的最高峰——首阳。周伯通在桃花岛独居已久，无聊之极。忽有一个人与他说话解闷，大感愉悦。他拍着木木的肩说：“你往下看，桃花岛的所有奥秘都在此？”木木往下看，只见桃花开得正艳，东一片，西一片。一阵晕眩，也看不出所以然。老顽童呵呵一笑：“在桃花岛上，实在是无聊，于是我天天坐在这儿看着一片一片桃花发呆。结果有天老天爷发脾气了，下了好大的暴雨。把我的桃花打得七零八落，我着实有些惆怅。这个时候有道彩虹徐徐而起，将那桃花林连成一片，突然间我终于明白了……”老顽童故作停顿，眼中发光：“一切皆因布局！”

老顽童从上衣口袋，拿出一支笔，又从裤袋里拿出了一张皱皱巴巴的纸铺开说：“你看那些一片一片桃花杂乱无章，而我看它们则非常规律。”木木大感兴趣，于是一老一小就蹲在那儿开始研究起来。

大多数 GUI 程序都有许多控件，这些控件如何放置，放置控件的容器大小改变时又要如何调整控件，这样的主题称为“布局”（layout）。黄老邪是做软件的，因此他的桃花岛布置完全合乎布局的这种思路。他将桃花岛分为 6 个区域，每个区域是一种布局，而每个区域的桃树林则可以看成是一个个控件。

木木听着老顽童的话，向第 1 个区域看去，只见几片红、浅红和白等不同颜色的桃树林东一片西一

片，没有丝毫异常。接着他又观察第 2 个区域，这里的桃树林是同样宽度，整整齐齐地沿纵向摆放。他的视线又转到了第 3 个区域，只见这个区域又划分成了若干个小区域。每个小区域还不一样，有的平行摆放了若干桃林，有的一行摆放不下，又换做第 2 行。木木头已经大了，不过还是接着看第 4 个区域。这个区域更是奇特，所有的桃林一层层由外到内不停嵌套。木木一阵晕眩，有点站立不稳，幸好老顽童站在旁边搀扶住了他。木木稍加休息，接着看第 5 个区域。这个区域倒并不是特别奇特，不同颜色的桃树林排成规则的格网。接着第 6 个区域，这个区域的桃树排成一个圈状，如图 10-1 所示。

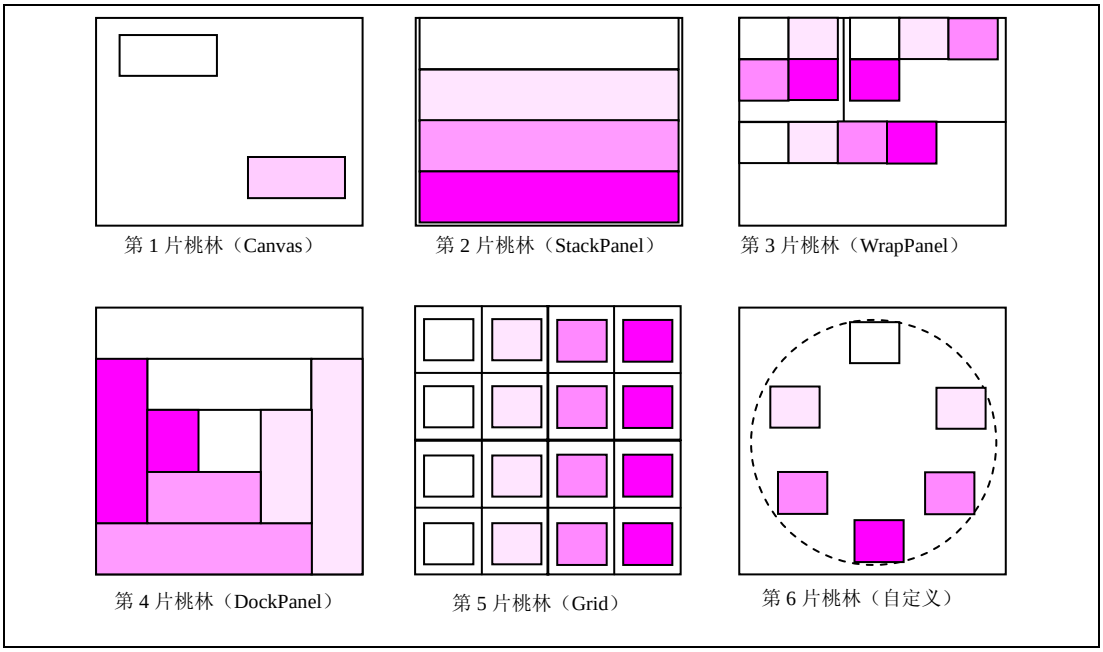

图 10-1　桃花岛 6 个区域的布局

老顽童嘻嘻一笑说到：“我初看这些桃林，也和你一般，看得时间久了也是非常晕眩。但是后来我看穿了黄老邪的秘密，再看这些桃林，发现特别规整。第 1 个是 Canvas 布局；第 2 个是 StackPanel 布局；第 3 个是 WrapPanel 布局；第 4 个你看起来比较晕眩的是 DockPanel 布局，实际上这种布局，一旦明白就会觉得非常简单；第 5 个和第 6 个布局看似平常，实际上未必简单。第 5 个是 Grid 布局，而第 6 个则是黄老邪自定义的一种布局。”

木木听老顽童这么一说，学习布局的兴趣骤然大增，于是打开笔记本开始逐个学习布局。

10.2.1　Canvas

Canvas 是基本面板，仅仅支持用与设备无关的坐标来定位元素。这是一种传统的布置用户界面的方式，在 Win32、MFC，甚至 WinForm 时期都是这样做的。

Canvas 用 4 个附加属性 Left、Top、Right 和 Bottom 来定位子元素，用代码 10-1 所示的代码可以模仿桃花岛上的第 1 个桃林区域（完整示例详见 mumu_layout 工程）。

```
<Page x:Class="mumu_layout.Page1"
      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="308" d:DesignWidth="294"
     Title="第一片桃林仿真 by 木木">
    <Border BorderThickness = "2" BorderBrush = "Black" Margin = "5">
        <Canvas>
            <Button Canvas.Left = "24" Canvas.Top = "50" Background="#00000000"
              Content = "Left=24,Top=50"/>
            <Button  Canvas.Right="24" Canvas.Bottom="50" Background ="#FFFFCCFF"
              Content = "Right=24,Bottom=50"/>
        </Canvas>
    </Border>
</Page>
```

代码 10-1　Page1.xaml 文件

程序运行结果如图 10-2 所示。

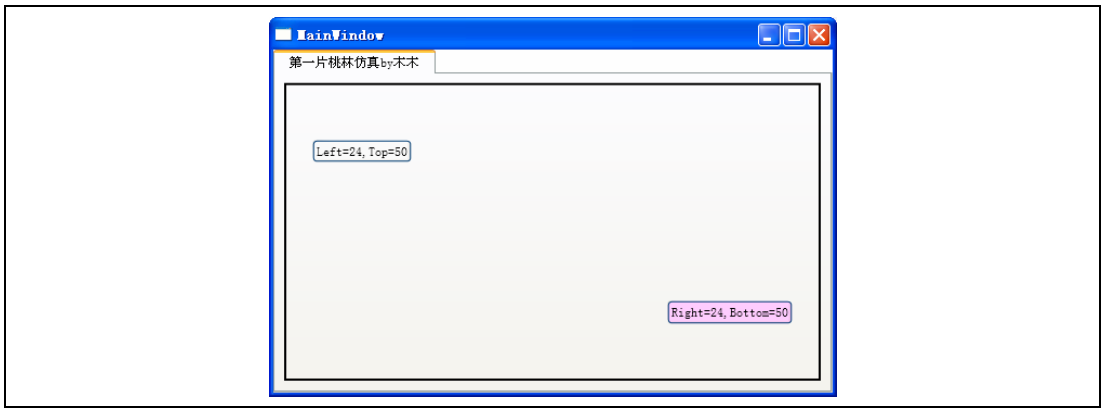

图 10-2　第一片桃林仿真

Canvas 只允许设置一个横向和纵向成对的坐标，如 Left 和 Top，以及 Right 和 Bottom 等。如果设置了 Left 和 Right 或者 Top 和 Bottom，则忽略一个值。

木木做完了这个仿真还是有些疑问，问到：“周大哥，这么原始的面板会有什么用呢？”老顽童看木木这么快做完一个例子，倒还真有些惊喜，说到：“你比我以前的一个结拜师弟要聪明得多。这种面板由于简单，自然效率就高，用在矢量绘图上是再合适不过了。”木木听了若有所思，紧接着开始学习第二种面板……

10.2.2　StackPanel

StackPanel 是一种非常受欢迎的面板，用于顺序垂直或者水平的排列子元素。它通过 Orientation 属性来控制水平（Horizontal）和垂直（Vertical）排列，默认值是纵向。模仿桃花岛的第 2 个区域的代码如代码 10-2 所示。

```
    <Page x:Class="mumu_layout.Page2"
          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="308" d:DesignWidth="251"
         Title="第二片桃林仿真 by 木木">
        <Border BorderThickness = "2" BorderBrush = "Black" Margin = "5">
           <StackPanel>
               <Button Background="#00000000"  MinHeight="50" MinWidth="50"
Content = "1"/>
               <Button Background ="#FFFFCCFF" MinHeight="50" MinWidth="50"
Content = "2"/>
               <Button Background ="#FFFF9BFF" MinHeight="50" MinWidth="50"
Content = "3"/>
               <Button Background ="#FFFF00FF" MinHeight="50" MinWidth="50"
Content = "4"/>
           </StackPanel>
        </Border>
    </Page>
```

代码 10-2　Page2.xaml 文件

程序运行结果如图 10-3 所示。

修改 Orientation 属性，桃林就可以由垂直排列变为水平排列，如代码 10-3 所示。

```
    <StackPanel Orientation="Horizontal">
```

代码 10-3　修改 Oritentation 属性为 Horizontal

程序运行结果如图 10-4 所示。

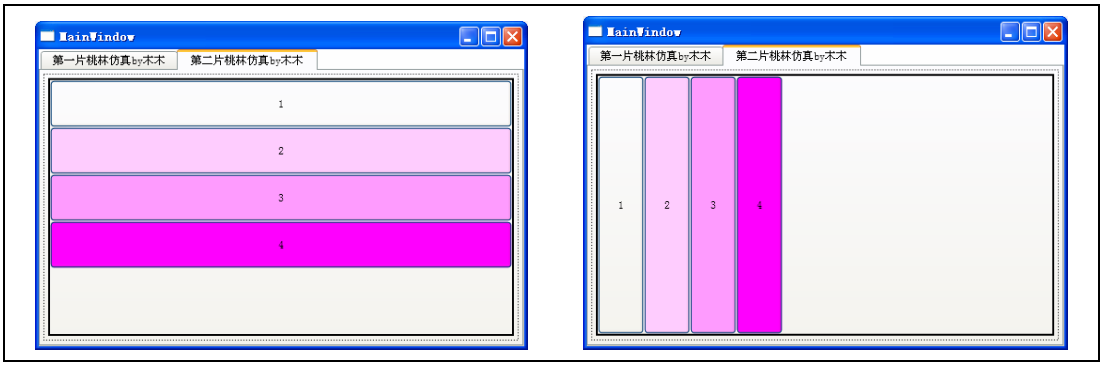

图 10-3　第 2 片桃林仿真　　图 10-4　第 2 片桃林变换之一：Orientation = "Horizontal"

修改 StackPanel 的一处属性（FlowDirection）如下：

```
    <StackPanel Orientation="Horizontal" FlowDirection = "RightToLeft">
```

程序运行结果如图 10-5 所示。

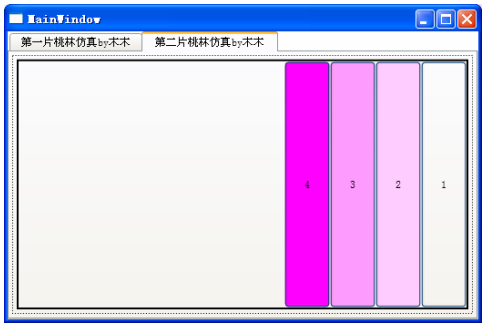

图 10-5　第 2 片桃林变换之二：Orientation = "Horizontal″FlowDirection = "RightToLeft"

10.2.3　WrapPanel

WrapPanel 与 StackPanel 类似，与 StackPanel 不同的是除了会自动垂直和水平排列子元素以外，当没有空间放置子元素时会自动将其放置在下一行或者下一列中，它特别适用子元素个数不确定的情况。

WrapPanel 有 3 个控制其行为的属性，如表 10-1 所示。

表 10-1　WrapPanel 的 3 个属性

属性	描述
Orientation	类似 StackPanel 的 Orientation，默认为水平排列
ItemHeight	允许子元素的最大高度，任何比 ItemHeight 高的子元素都将被截断
ItemWidth	允许子元素的最大宽度，任何比 ItemWidth 宽的子元素都将被截断

使用 WrapPanel 模拟第 3 个桃林的代码如代码 10-4 所示。

```
    <Page x:Class="mumu_layout.Page3"
          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="233" d:DesignWidth="251"
         Title="第三片桃林仿真 by 木木">
        <Border BorderThickness = "2" BorderBrush = "Black" Margin = "5">
            <WrapPanel ItemWidth = "120">
                <Button Background="#00000000"  MinWidth = "50"  Content = "1"/>
                <Button Background ="#FFFFCCFF" MinWidth = "50"  Content = "2"/>
                <Button Background ="#FFFF9BFF" MinWidth = "50"  Content = "3"/>
                <Button Background ="#FFFF00FF" MinWidth = "150"  Content = "4 MinWidth
= 150"/>
            </WrapPanel>
        </Border>
    </Page>
```

代码 10-4　Page3.xaml 文件

程序运行结果如图 10-6 所示，第 4 个按钮由于长度超出 WrapPanel 指定的范围，因此被截断。而且随着窗口的大小改变，子元素会自动地变换其位置。

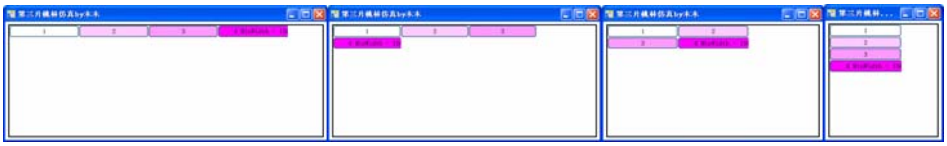

图 10-6　水平排列的按钮随着窗口的宽度变小重新排列

将其改为垂直排列，运行结果如图 10-7 所示。

图 10-7　垂直排列的按钮随着窗口的高度变小重新排列

10.2.4　DockPanel

DockPanel 可以使子元素停靠在面板的某一条边上，然后拉伸元素以填满全部宽度或高度。它有一个 Dock 附加属性，子元素用 4 个值来控制其停靠，即 Left（默认）、Top、Right 和 Bottom。注意 Dock 并没有 Fill 值，默认情况下最后一个添加到 DockPanel 的子元素将填满所有剩余的空间；除非 DockPanel 的 LastChildFill 属性设置为 false。

模拟第 4 片桃林的代码如代码 10-5 所示。

```
<Page x:Class="mumu_layout.Page4"
     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"
    Title="第四片桃林仿真 by 木木">
   <Border BorderThickness = "2" BorderBrush = "Black" Margin = "5">
       <!—两种不同属性值的设置，LastChildFill 默认为 true-->
        <!--<DockPanelLastChildFill = "false">-->
      <DockPanel>
         <Button Background="#00000000"  DockPanel.Dock = "Left"  Content = "1"/>
         <Button Background ="#FFFFCCFF" DockPanel.Dock = "Top"  Content = "2"/>
         <Button Background ="#FFFF9BFF" DockPanel.Dock = "Right"  Content = "3"/>
         <Button Background ="#FFFF00FF" DockPanel.Dock = "Bottom"  Content = "4 "/>
         <Button Background="#00000000"  DockPanel.Dock = "Left"  Content = "5"/>
         <Button Background ="#FFFFCCFF" DockPanel.Dock = "Top"  Content = "6"/>
         <Button Background ="#FFFF9BFF" DockPanel.Dock = "Right"  Content = "7"/>
         <Button Background ="#FFFF00FF" DockPanel.Dock = "Bottom"  Content = "8 "/>
      </DockPanel>
   </Border>
</Page>
```

代码 10-5　Page4.xaml 文件

程序运行结果如图 10-8 所示。

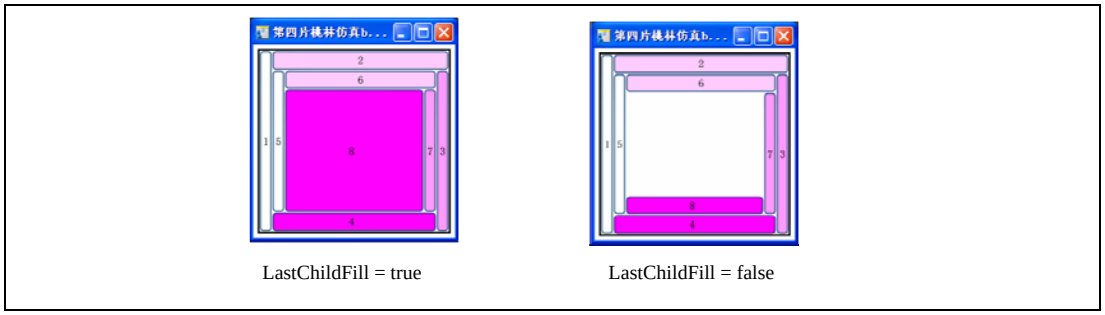

图 10-8　第 4 片桃林 DockPanel

写完这些代码后，突然老顽童说到：“且慢，木木老弟，你可以用 DockPanel 模拟出 StackPanel 的效果么？”

木木想了想说：“这有何难！”于是写下了如代码 10-6 所示。

```
<Page x:Class="mumu_layout.Page5"
      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="281" d:DesignWidth="267"
     Title="第四片桃树林 by 木木 DockPanel 模拟 StackPanel">
    <Border BorderThickness = "2" BorderBrush = "Black" Margin = "5">
        <DockPanel>
            <Button Background="#00000000"  DockPanel.Dock = "Top"  Content = "1"/>
            <Button Background ="#FFFFCCFF" DockPanel.Dock = "Top"  Content = "2"/>
            <Button Background ="#FFFF9BFF" DockPanel.Dock = "Top"  Content = "3"/>
            <Button Background ="#FFFF00FF" DockPanel.Dock = "Top"  Content = "4 "/>
        </DockPanel>
    </Border>
</Page>
```

代码 10-6　Page5.xaml 文件

木木运行程序看了之后，虽然大致和自己想的差不多，但是最后一个按钮的面积过大。正在木木思考如何修改时，老顽童抢过笔记本，在 DockPanel 标签中加上了“LastChildFill = false”。一切变得完美了，如图 10-8 所示。两人相视一笑。

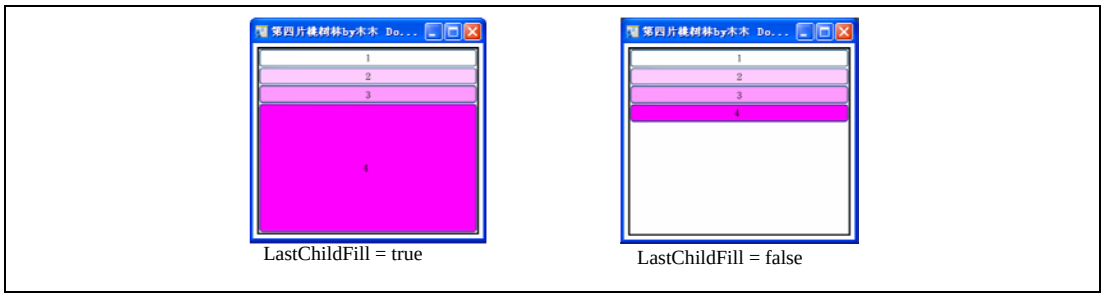

图 10-9　LastChildFill 的不同设置的效果

10.2.5 Grid

Grid 是 WPF 中常用且功能最为强大的布局，允许在一个多行多列的表中排列子元素。它有 4 个常用附加属性，如表 10-2 所示。

表 10-2 Grid 的 4 个常用附加属性

属性	说明
Column	标识子元素所在的列（以 0 为第 1 列）
Row	标识子元素所在的行（以 0 为第 1 行）
ColumnSpan	标识子元素占据的列数
RowSpan	标识子元素占据的行数

Grid 的行和列尺寸有如下 3 种单位。

（1）绝对尺寸（Absolute Sizing）：设置 Height 或 Width 为一个设备无关的值，当 Grid 的尺寸改变时绝对尺寸的行和列不会改变。

（2）自动尺寸（Autosizing）：设置 Height 或 Width 为 Auto，对于一行，这是最高元素的高度；对于一列，这是最宽元素的宽度。

（3）比例尺寸（Proportional sizing）或者称为“星号尺寸”（Star sizing）：设置 Height 或 Width 为一种*号的特殊语法，可以使行和列按比例来分配可用的区域，一个采用比例尺寸的行和列会随着 Grid 的尺寸的改变而改变。

木木不太理解这种星号语法（Star Syntax），老顽童顺手拿过笔记本，写下了如代码 10-7 所示的例子：

```
<Page x:Class="mumu_layout.Page6"
      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="564" d:DesignWidth="406"
     Title="星号尺寸 Demoby 老顽童">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition/>
            <RowDefinition/>
            <RowDefinition/>
            <RowDefinition/>
        </Grid.RowDefinitions>
        <Grid Grid.Row = "0">
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width = "100"/>
                <ColumnDefinition Width = "*"/>
            </Grid.ColumnDefinitions>
            <Button Background="#00000000" Grid.Column = "0"  Content = "Width = 100"/>
            <Button Background ="#FFFFCCFF" Grid.Column = "1"  Content = "Width = *"/>
        </Grid>
```

```
        <Grid Grid.Row = "1">
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width = "100"/>
                <ColumnDefinition Width = "*"/>
                <ColumnDefinition Width = "*"/>
                <ColumnDefinition Width = "*"/>
            </Grid.ColumnDefinitions>
            <Button Background="#00000000" Grid.Column = "0"  Content = "Width = 100"/>
            <Button Background ="#FFFFCCFF" Grid.Column = "1"  Content = "Width = *"/>
            <Button Background ="#FFFF9BFF" Grid.Column = "2"  Content = "Width = *"/>
            <Button Background ="#FFFF00FF" Grid.Column = "3"  Content = "Width = *"/>
        </Grid>
        <Grid Grid.Row = "2">
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width = "100"/>
                <ColumnDefinition Width = "*"/>
                <ColumnDefinition Width = "2*"/>
                <ColumnDefinition Width = "*"/>
            </Grid.ColumnDefinitions>
            <Button Background="#00000000" Grid.Column = "0"  Content = "Width = 100"/>
            <Button Background ="#FFFFCCFF" Grid.Column = "1"  Content = "Width = *"/>
            <Button Background ="#FFFF9BFF" Grid.Column = "2"  Content = "Width = 2*"/>
            <Button Background ="#FFFF00FF" Grid.Column = "3"  Content = "Width = *"/>
        </Grid>
        <Grid Grid.Row = "3">
            <Grid.ColumnDefinitions>
                <ColumnDefinition Width = "100"/>
                <ColumnDefinition Width = "*"/>
                <ColumnDefinition Width = "2*"/>
                <ColumnDefinition Width = "*"/>
            </Grid.ColumnDefinitions>
            <Button Background="#00000000" Grid.Column = "0"  Content = "Width = 100"/>
            <Button Background ="#FFFFCCFF" Grid.Column = "1"  Content = "Width = 3*"/>
            <Button Background ="#FFFF9BFF" Grid.Column = "2"  Content = "Width = 6*"/>
            <Button Background ="#FFFF00FF" Grid.Column = "3"  Content = "Width = 3*"/>
        </Grid>
    </Grid>
</Page>
```

代码 10-7　Page6.xaml 文件

这是一个嵌套 panel 的示例，一个 4 行的 Grid 中每一行嵌套一个 Grid，如图 10-10 所示。

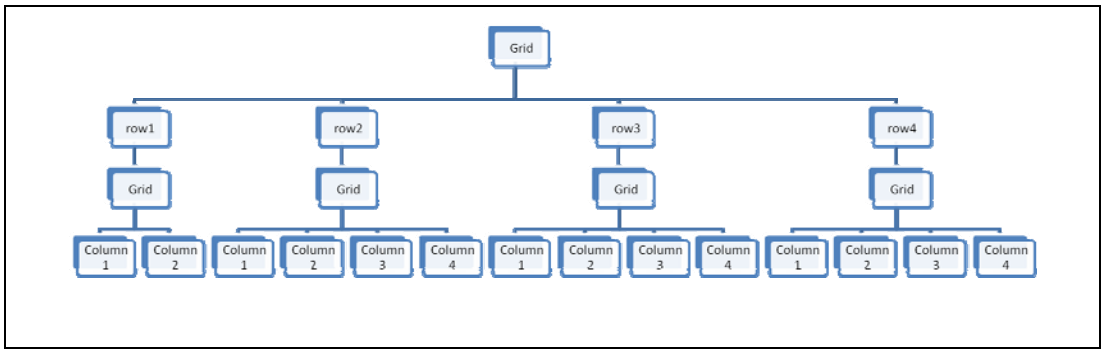

图 10-10　一个嵌套 panel 的例子

程序运行结果如图 10-11 所示。

图 10-11　程序运行结果

这种语法基本上有这样的特点，老顽童在纸上写到：

（1）当一个行的高度或列的宽度设置为*时会占据所有的剩余空间。

（2）当多行或者多列使用*时剩余的空间会被等量地分配给这些行或列。

（3）行或列可以在*之前放一个系数，如 2*和 6*，表示按比例比其他行或列占据更多的空间。在同一个 Grid 中一个宽度为 2*的列的宽度是一个宽度为*（1*的缩写）列的两倍。

“而且还有一个好玩的东西”，老顽童说着，又十指如飞，把程序代码做了如下修改“通过将 Grid 的属性 IsSharedSizeScope 设置成 true（代码①），然后将行和列的属性 SharedSizeGroup 设置为一个大小写敏感的字符串，表示这个组的名称（代码②和③）。不仅可以将一个 Grid 里面的行和列设置成同样的高度或宽度，甚至是不同 Grid 的行和列也可以设置成同样的高度或者宽度。”老顽童说着说着，都手舞足蹈起来，不过木木确实没觉得这个属性有什么好玩的，……如下为代码 10-8。

```
<Page
    ……
    Title="共享尺寸 Demoby 老顽童">
①   <Grid Grid.IsSharedSizeScope = "true">

    ……
      <Grid Grid.Row = "0">
        <Grid.ColumnDefinitions>
②           <ColumnDefinition Width = "Auto" SharedSizeGroup = "a"/>
            <ColumnDefinition Width = "Auto" SharedSizeGroup = "a"/>
        </Grid.ColumnDefinitions>
        ……

      </Grid>

          ……

      <Grid Grid.Row = "2">
        <Grid.ColumnDefinitions>
            <ColumnDefinition Width = "100"/>
            <ColumnDefinition Width = "*"/>
③           <ColumnDefinition Width = "2*" SharedSizeGroup = "a"/>
```

```
                <ColumnDefinition Width = "*"/>
            </Grid.ColumnDefinitions>
……

        </Grid>

……
    </Grid>
</Page>
```

代码 10-8　Page7.xaml 文件

程序运行结果如图 10-12 所示。

图 10-12　共享尺寸属性使 SharedSizeGroup 名称一样的行和列保持同一高度或者宽度

“还有一个特别好玩的称之为 GridSplitter，你有了它就可以用鼠标和键盘来改变尺寸。比如……”老顽童说着，又噼哩叭啦敲了一通，“成了，就是这样……”如代码 10-9 所示。

```
<Page x:Class="mumu_layout.Page8"
    ……
    Title="GridSplitterDemoby 老顽童">
   <Grid Grid.IsSharedSizeScope = "true">
      <Grid.RowDefinitions>
         <RowDefinition/>
         <RowDefinition/>
         <RowDefinition/>
         <RowDefinition/>
      </Grid.RowDefinitions>
①     <GridSplitter Grid.Row="2" HorizontalAlignment="Stretch"
                    VerticalAlignment="Top" Background="Black"
                    ShowsPreview="true" ResizeDirection="Rows"
                    Height="5"/>
      <Grid Grid.Row = "0">
        ……
      </Grid>
      <Grid Grid.Row = "1">
        ……
      </Grid>
      <Grid Grid.Row = "2">
```

```
                <Grid.ColumnDefinitions>
                    <ColumnDefinition Width = "100"/>
                    <ColumnDefinition Width = "*"/>
                    <ColumnDefinition Width = "2*" SharedSizeGroup = "a"/>
                    <ColumnDefinition Width = "*"/>
                </Grid.ColumnDefinitions>
②               <GridSplitter Grid.Column="2" HorizontalAlignment="Left"
                              VerticalAlignment="Stretch" Background="Black"
                              ShowsPreview="true" Width="10"/>
③               <Button Background="#00000000" Grid.Column = "0"  Content
= "Width = 100" Margin="5"/>
                <Button Background ="#FFFFCCFF" Grid.Column = "1"  Content
= "Width = *" Margin="5"/>
                <Button Background ="#FFFF9BFF" Grid.Column = "2" Content
= "SharedSizeGroup = a" Margin="5"/>
                <Button Background ="#FFFF00FF" Grid.Column = "3"  Content
= "Width = *" Margin="5"/>
            </Grid>
            <Grid Grid.Row = "3">
                ……
            </Grid>
        </Grid>
    </Page>
```

代码 10-9　GridSplitter 的实现

老顽童添加了两个 GridSplitter，其中一个是在第 3 行（代码①处）。它位于第 3 行的顶部，这是由 VerticalAlignment 属性所决定的；另外该 GridSplitter 的 HorizontalAlignment 属性为 Stretch，因此该 GridSplitter 会横跨所有的列。ResizeDirection 用来设置该 GridSplitter 用来调整行的还是列的一个属性，在这里用来调整行的。另外一个 GridSplliter 是在第 2 行的 Grid 中添加的（代码②）。老顽童为了能够将两个 GridSplitter 都显现出来，有意将几个按钮的 Margin 属性设置为 5，以便为 GridSplitter 留出显示的空间（代码③）。

老顽童写完之后，很快运行程序，然后用鼠标不断地拖拉着两个 GridSplitter，表情很沉醉。“真不知道怎么能乐成这样”木木小声嘀咕着。

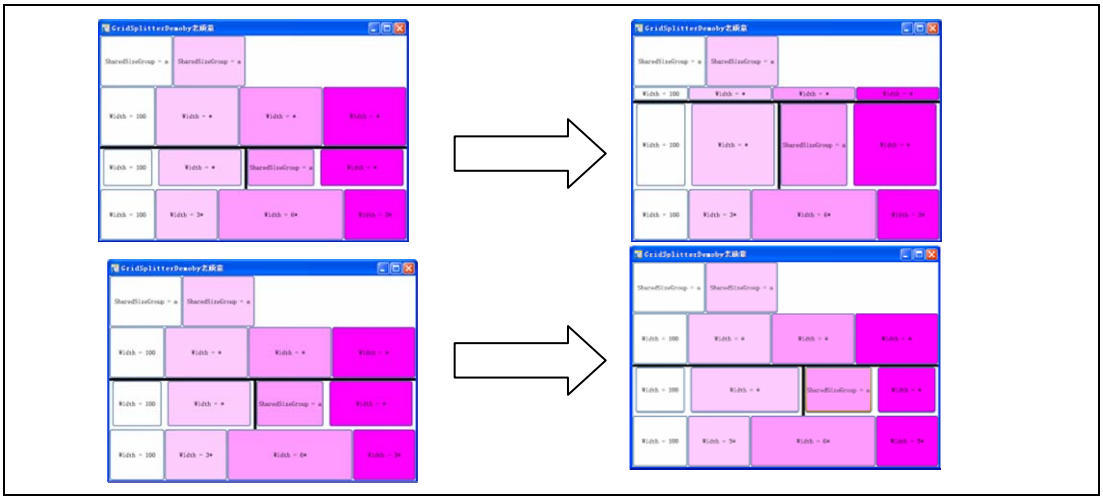

图 10-13　利用 GridSplitter 调整列宽和行高

“其实改变 GridSplitter 的属性 ResizedBehavior 之后,还可以这样玩”老顽童正欲往下说，突然一阵箫声又响起。箫声由远及近，不到半柱香的功夫，一个青衣怪客就近得前来……

10.3 黄岛主演绎布局精妙

那青衣怪客左手拿箫，右手慢慢从脸上揭下一层人皮面具。但见他形相清癯，丰姿隽爽，萧疏轩举，湛然若神，正是桃花岛主黄药师。老顽童嘻嘻一笑，说道：“黄老邪，你用碧海潮生曲降我不住，倒被这傻小子轻松破解。”黄药师不由把木木仔细打量了一番，只见此人阔鼻大耳。还冒着三分傻气，实在不像身怀绝技之人。

黄药师袍袖一翻，木木的笔记本顷刻间就到他手上。药师一瞥，不由大惊，内心暗想：“好小子，原来你们正在研究我桃花岛布局的秘密。”心中不禁又有了几分欢喜之意，但是仍不动声色，问到：“傻小子，你上我桃花岛来干什么？”木木顿时满脸通红，老顽童说：“黄老邪，你是真傻还是假傻，当然是来上门娶你的宝贝丫头啊！”

说话间，黄药师已经看过老顽童和木木这几天游戏玩乐的代码。黄药师叹了一口气说：“伯通，你果然是个编程奇才，居然悟到了我桃花岛布置最精妙之处。”老顽童乐得手舞足蹈起来：“哈哈，老邪服输了吧。”药师眉头微微一皱，淡淡笑道：“不过你忽视了研究桃林本身。”老顽童一惊，随即又喜笑颜开，说：“老邪，赶紧讲讲，要不我拜你为师了。”话没说完咚咚 3 个响头……

10.3.1 桃树林的属性

药师其实也是喜好编程之人，看周伯通发自真心地佩服，说：“伯通不敢当，我愿意共同讨论讨论。说的不对，请你多指教。”3 人走到首阳峰一亭上，岛上下起小雨，飘飘洒洒，甚是舒服，悉听药师演绎布局。

桃花岛上的布局实际上是一个槽（slot）模型，其中每个区域（父对象）分配给桃林（子对象）一个槽。桃林能自由占用这个槽中空间的任何部分，该功能通过桃树林的 3 个属性，即 Margin、HorizontalAligment 和 VerticalAlignment 来实现。它们都是 FrameworkElement 的属性，而大多数控件都要继承 FrameworkElement。Margin 允许子控件在槽内部获得一个围绕自身的缓冲空间，HorizontalAligment 和 VerticalAlignment 决定子控件如何占用槽中的保留空间。

说着，黄岛主写下了代码 10-10（详见工程 mumu_layout2）。

```
<Page x:Class="mumu_layout2.Page1"
      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"
     Title="桃林的属性 by 黄药师" >
    <Border Background="LightBlue"
         BorderBrush="Black"
         BorderThickness="2"
         CornerRadius="45"
         Padding="25">
```

```
            <StackPanel Name="StackPanel1" Background = "White">
                <TextBlock FontSize="18" HorizontalAlignment="Center"
Margin="0,0,0,15" Text = "StackPanel1">
                </TextBlock>
                <Border BorderThickness  = "1" BorderBrush = "black" >
                    <Button Margin="5,10,15,20" >Normal</Button>
                </Border>
                <Border BorderThickness  = "1" BorderBrush = "black" >
                    <Button Margin="5,10,15,20" HorizontalAlignment = "Left">Left</Button>
                </Border>
                <Border BorderThickness  = "1" BorderBrush = "black" >
                    <Button HorizontalAlignment = "Right"
Margin="5,10,15,20">Right</Button>
                </Border>
                <Border BorderThickness  = "1" BorderBrush = "black" >
                    <Button Margin="5,10,15,20" HorizontalAlignment =
"Center">Center</Button>
                </Border>
                <TextBlock>Button.Margin="5,10,15,20"</TextBlock>
            </StackPanel>
        </Border>
    </Page>
```

代码 10-10　Page1.xaml 文件

运行该程序，结果如图 10-14 所示。

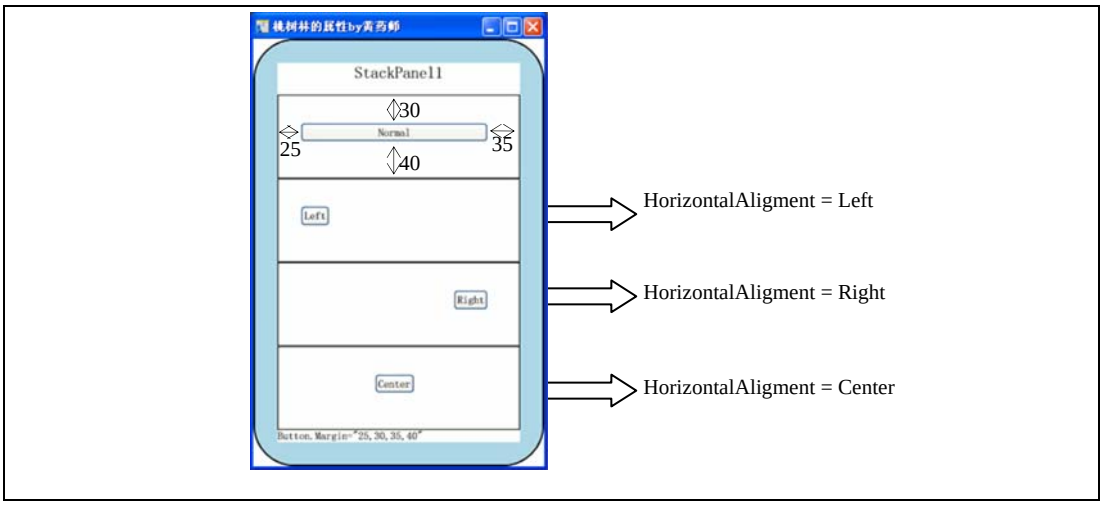

图 10-14　运行结果

老顽童笑了笑说：“老邪啊，只是我的注意力都集中在各种布局上了，忽视了控件本身的属性，这个算不上精妙啊。”黄药师没有理会，又添加了几行代码。如代码 10-11 所示（代码①、②和③处）。

```
                    <Border BorderThickness  = "1" BorderBrush = "black" >
                        <Button   Margin="5,10,15,20"   HorizontalAlignment   =
        "Left" Content="Left">
    ①                       <Button.LayoutTransform>
                                <RotateTransform Angle = "15" />
                            </Button.LayoutTransform>
                        </Button>
                    </Border>
```

```
                    <Border BorderThickness = "1" BorderBrush = "black" >
                        <Button      HorizontalAlignment      =      "Right"
    Margin="5,10,15,20" Content="Right">
②                           <Button.LayoutTransform>
                                <RotateTransform Angle = "45" />
                            </Button.LayoutTransform>
                        </Button>
                    </Border>
                    <Border BorderThickness = "1" BorderBrush = "black" >
                        <Button   Margin="5,10,15,20"   HorizontalAlignment   =
    "Center" Content="Center">
③                           <Button.LayoutTransform>
                                <RotateTransform Angle = "75" />
                            </Button.LayoutTransform>
                        </Button>
```

代码 10-11　给按钮添加上 LayoutTransform 属性（Page2.xaml 文件）

“伯通，要不你来运行这个程序。”药师微微笑道。周伯通按了一下“F5”键，只见后面 3 个按钮都按照不同角度进行旋转，周伯通又惊又喜。如图 10-15 所示。药师说道：“这是桃树林的另一种变换。”

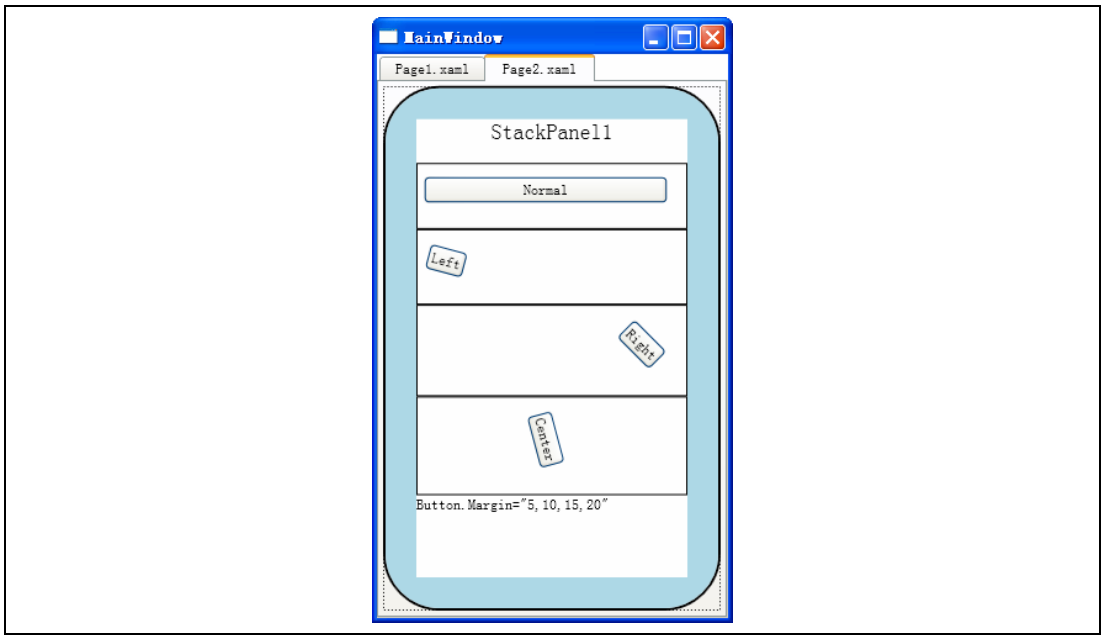

图 10-15　3 个按钮按照不同角度旋转变换

木木倒是陷入了沉思良久，拱手问到：“黄前辈……”药师眉头微皱喝道：“什么前辈不前辈，我黄老邪最烦人喊我前辈前辈个不停。”木木没有在乎这几句话，继续问到：“我刚才观察了一下 Button 除了有一个 LayoutTransform 属性，还有一个 RenderTransform 属性，为什么你不用后者呢？”黄老邪转怒为喜：“好小子，这其间的区别在于此。”说着又写下代码 10-12。

```
<Button   Content="LayoutTransform">
    <Button.LayoutTransform>
        <RotateTransform Angle = "15" />
```

```
    </Button.LayoutTransform>
</Button>
<Button  Content="RenderTransform">
     <Button.RenderTransform>
         <RotateTransform Angle = "15" />
      </Button.RenderTransform>
</Button>
```

代码 10-12　LayoutTransform 和 RenderTransform（Page3.xaml 文件）

运行结果如图 10-16 所示。

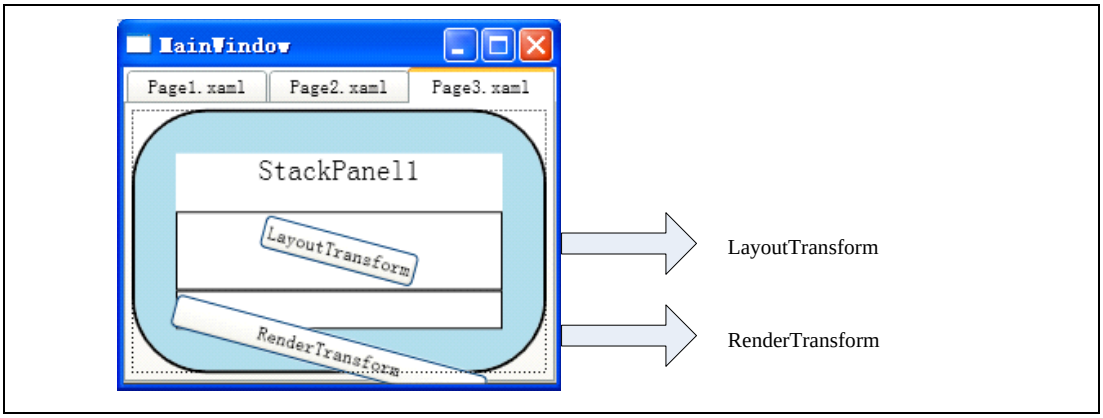

图 10-16　运行结果

“看出区别了没有，LayoutTransform 参与布局，而 RenderTransform 并不参与，因此应用 RenderTransform 的按钮会扩展到面板边界的外面。”药师说到。

“黄岛主，”木木已经不敢喊前辈二字了：“您前面的五片桃林，我大致能看个明白，唯独这第六片桃林……”殊不知这第六片桃林是黄药师集大成之作，黄药师虽然城府极深，但木木这一问也是挠到他的痒处，不禁问到：“怎样？”木木略一沉吟，说到：“我初看第六片桃树林，没有觉得什么稀奇的。相反现在越学越觉得第六片桃树林不简单，以前听周大哥说过，这个是岛主自定义的一个布局。现在想想的确任何一个布局都无法做到，因此觉得十分困惑，不明白黄岛主您是如何做到的？”

黄药师久居岛上，自己经常沉浸于这其间的精彩之处，但谁有人知，谁有人晓，听到木木这一说，虽没有任何赞誉之词，但却有遇到知音的感觉。黄药师脸上不动声色，但是心里大感快慰。“黄老邪，快讲讲你的自定义布局，我也好奇得紧。”老顽童嚷到，“不过老邪，这回老顽童是真佩服你了。”

药师见两人真诚，于是也不再有所保留，开始讲起了他的自定义布局……

10.3.2　自定义布局

要掌握自定义布局，必须了解布局的两个阶段。实际上确定控件最佳尺寸的经历了两个阶段，第 1 个阶段为测量（Measure）阶段，即父元素询问子元素所期望的尺寸，从而确定自身的尺寸；第 2 阶

段为布置（Arrange）阶段，在这个期间每个父元素会告知子元素的尺寸和位置。

从编程模型来看，具体到两个重载函数，即 MeasureOverride 和 ArrangeOverride，如代码 10-13 所示。

```
protectedoverride Size MeasureOverride(Size constraint)
```

代码 10-13　MeasureOverride 函数声明

MeasureOverride 传递的参数为 Size 类型，实际是上一级父元素告知当前元素可分配的空间（availableSize）；返回的参数 Size 类型，是该元素所期望的空间（desiredSize），理想的情况是如代码 10-14 所示。

```
protected override Size ArrangeOverride(Size finalSize)
```

代码 10-14　ArrangeOverride 函数声明

ArrangeOverride 传递和返回的参数同样是 Size 类型，传递的参数指定是该元素摆放所用的尺寸（finalSize）；返回参数同为该元素及其子元素所占用的尺寸。

黄药师见他们还是有很多疑惑，于是拿过木木的笔记本，把第六片桃林实现的原理用代码写了下来。首先黄药师自定义了一个 CustomPanel。该类从 Panel 派生而来，主要用于重载药师刚刚说的两个函数。如代码 10-15 所示。

```
CustomPanel.cs
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Media;

namespace mumu_layout2
{
    public class CustomPanel : Panel
    {
        public CustomPanel()
            : base()
        {
        }

        protected override Size MeasureOverride(Size availableSize)
        {
            double maxChildWidth = 0.0;
            double maxChildHeight = 0.0;
            foreach (UIElement child in InternalChildren)
            {
                child.Measure(availableSize);
                maxChildWidth = Math.Max(child.DesiredSize.Width, maxChildWidth);
                maxChildHeight = Math.Max(child.DesiredSize.Height, maxChildHeight);
            }
            double idealCircumference = maxChildWidth * InternalChildren.Count;
            double idealRadius = (idealCircumference / (Math.PI * 2) + maxChildHeight);

            Size ideal = new Size(idealRadius * 2, idealRadius * 2);
            Size desired = ideal;
            if (!double.IsInfinity(availableSize.Width))
            {
                if (availableSize.Width < desired.Width)
```

```
                desired.Width = availableSize.Width;
            }
            if (!double.IsInfinity(availableSize.Height))
            {
                if (availableSize.Height < desired.Height)
                    desired.Height = availableSize.Height;
            }
            return desired;
        }

        protected override Size ArrangeOverride(Size finalSize)
        {
            Rect layoutRect;
            if (finalSize.Width > finalSize.Height)
            {
                layoutRect = new Rect((finalSize.Width - finalSize.Height) / 2,
0, finalSize.Height, finalSize.Height);
            }
            else
            {
                layoutRect = new Rect(0, (finalSize.Height - finalSize.Width) /
2, finalSize.Width, finalSize.Width);
            }
            double angleInc = 360 / InternalChildren.Count;
            double angle = 0;
            foreach (UIElement child in InternalChildren)
            {
                Point childLocation = new Point(layoutRect.Left +
(layoutRect.Width - child.DesiredSize.Width) / 2, layoutRect.Top);

                child.RenderTransform = new RotateTransform(angle,
child.DesiredSize.Width / 2, finalSize.Height / 2 - layoutRect.Top);
                angle += angleInc;
                child.Arrange(new Rect(childLocation, child.DesiredSize));
            }
            return finalSize;
        }
    }
}
```

代码 10-15　自定义一个面板

黄药师随后新建一个页面，在其中调用 CutstomPanel，如代码 10-16 所示。

```
<Page ……
     xmlns:local="clr-namespace:mumu_layout2">
    <local:CustomPanel>
        <Button Background="#00000000" MinWidth="100">1</Button>
        <Button Background ="#FFFFCCFF" MinWidth="100">2</Button>
        <Button Background ="#FFFF9BFF" MinWidth="100">3</Button>
        <Button Background ="#FFFF00FF" MinWidth="100">4</Button>
        <Button Background="#FFFFCCFF" MinWidth="100">5</Button>
    </local:CustomPanel>
</Page>
```

代码 10-16　在页面中放置 CustomPanel

药师轻点运行，果然桃花岛的第六片桃林样子展现出来。这时老顽童是喜得手舞足蹈，木木却一反常态，细心体会黄岛主的代码。药师暗暗赞许，觉得此人确是可造之材。

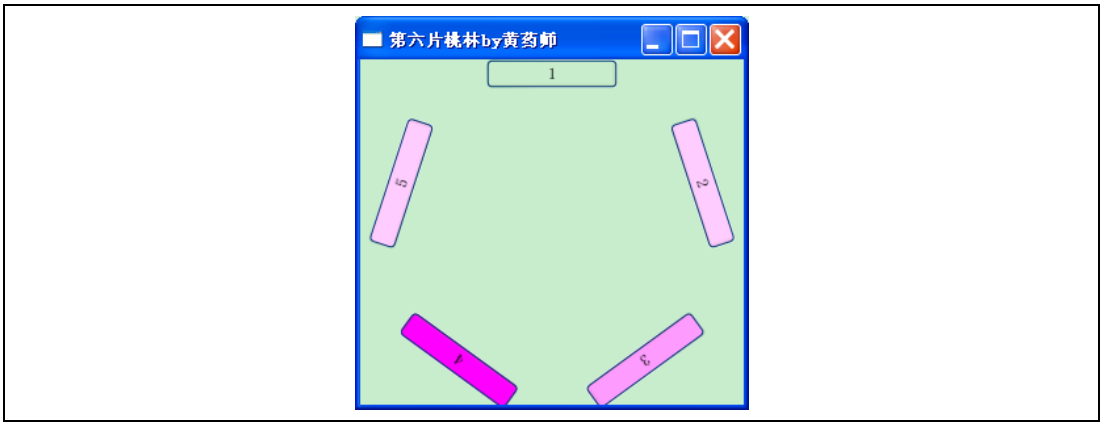

图 10-17　第 6 片桃林的布局

突然身后有人轻轻一笑，竹林里已闪身出一轻盈女子，长发披肩，全身白衣，头发上束了条金带，更是灿然生光。木木不由抬头，见这少女一身装束犹如仙女一般，不禁看得呆了。这少女正是黄蓉，“来，蓉儿”，不等药师招手，黄蓉早已飞一般地扑到爹爹怀里。药师笑到：“木木，你已通过我的测验，可以成为我的女婿了。”

木木还在发愣，被老顽童推了一个踉跄，“还不拜见岳父大人。”有道是布局巧设桃花岛，木木憨娶俏黄蓉。

10.4　接下来做什么

这是全书当中出现的第一个故事，后面还会出现类似这样的故事。布局解决的是窗口如何摆放控件的问题。接下来，我们需要知道的就是窗口摆放的内容——控件。

参考文献

[1] “北风之神”风清远整理，“云中孤雁”制作《金庸全集典藏版 射雕英雄传》，“第十六回　九阴真经”。