Windows Phone Mango编程实践

***Windows Phone Mango Programming Practice***

第三篇 XNA游戏篇

# 游戏开发新功能

曾经看到一部茶道的宣传短片印象深刻，其中将泡茶的整个过程做成了3D模拟的动画讲述茶道。如果我们在Windows Phone 智能手机中制作一款茶道游戏，“以茶行道，以茶雅志”，那将是多美好的一件事。中国茶道追求“真”，通过直觉体悟达到对人生、对功利的精神上的超越，自省其身。茶道本身犹如程序逻辑，程序人生犹如饮茶品茗，勿忘子曰：“君子喻于义，小人喻于利。”本例中我们组合Silverlight或XNA的框架开发一款3D茶具，作为Windows Phone茶道游戏的引子，为读者实现茶道游戏感悟程序人生起到抛砖引玉的作用。

## Mango新功能概述

本章参考和引用Windows Phone官方教程和开发培训包，以及MSDN Windows Phone开发文档。

XNA Game Studio 4.0 是一款游戏开发产品，它基于Windows Phone的Microsoft Visual Studio 2010 Express,，为游戏开发者提供强大而简单的C#编程语言。 XNA Game Studio 4.0 包括XNA 框架和框架内容管道, 从而提供了一种简单而灵活的方式把3D模型，材质，声音和其他资源导入到你的游戏中，并且一个专注于游戏的应用编程接口简化了Xbox 360®, Windows®和现在的 Windows Phone ®上的游戏开发。

### Silverlight和XNA的集成

最大的新功能就是Silverlight和XNA框架在应用程序中的整合。This enables a wide range of new scenarios, such as games that use the WebBrowser control to perform Open Authentication for social network integration and applications that display GPU-accelerated 2D and 3D content.这使Windows Phone应用程序的能力更强，比如在游戏中调用WebBrowser控件执行社会网络的身份认证和集成，应用程序使用GPU加速显示2D和3D画面。

### 执行模型和应用程序快速切换

Windows Phone管理应用程序生命周期的变化的结果，就是实现Fast application switching is a general change in the way application lifetime is managed on Windows Phone.快速的应用切换。 With the new changes to the execution model, titles are no longer implicitly terminated by another application or an interruption (such as a phone call).由于对执行模型的新变化，用户离开游戏去浏览网站，或者接听电话都能很快恢复游戏，而无需重新加载游戏内容或游戏状态，只要有足够的设备资源。

### Windows Phone事件探查器

Mango包括在Windows Phone应用程序代码和内存的探查器，使开发人员能够分析他们的游戏和应用程序的性能。这包括支持CPU和内存的可视化分析系统。

### Combined Motion API

Mango支持指南针和陀螺仪的API，并有新的API来访问原有的传感器。 Mango将加速度传感器、指南针传感器以及陀螺仪的原始数据进行高层次的封装，更适合游戏应用程序使用传感器的数据。这组高度封装的API称之为Motion API。

其次，在Mango中终于允许你直接来访问摄像头的原始的帧数据，除此之外还包括闪光灯、自动对焦、快门按钮等。

## 跨平台编译

为了达到XNA Game Studio 4.0开发不同平台的游戏程序的目标，程序设计师可以先使用 Visual Studio 2010 或 Visual Studio 2010 Express for Windows Phone 建立游戏项目，并设计游戏的功能，然后使用鼠标的右键点选 [Solution Explorer] 窗口中的项目名称，再从出现的菜单选择 [Create Copy of Project for **XXX**] 功能 (其中的 XXX 代表：Windows、Xbox 360、或 Windows Phone)，就可以依据目前项目的内容建立可以在指定平台上执行的游戏程序项目，Visual Studio 2010 或 Visual Studio 2010 Express for Windows Phone 就会为我们建立新的游戏项目。

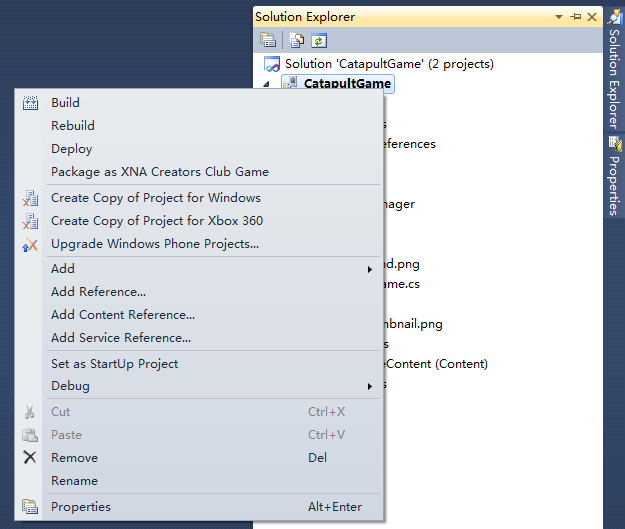


图15-1 Create Copy of Project

执行 [Create Copy of Project for **XXX**] 功能 (其中的 XXX 代表：Windows、Xbox 360、或Windows Phone)虽然可以依据现有的项目内容建立新的项目，但是Visual Studio 2010或Visual Studio 2010 Express并未将现有的项目内容复制一份到新的项目中，而是让新的专案和现有的项目共享同一份源代码以及Content Pipeline项目，换句话说，当您修改现有项目的原始码时，相当于修改新建立的项目的原始码，免去维护多份源代码造成的负担。

XNA Game Studio项目定义符号如**# if**条件编译指令指定游戏所兼容的平台。您可以使用相同的源代码编译生成运行于 Windows、 Xbox 360 或者 Windows Phone的应用程序。

源代码在不同的平台上执行时需要支持不同的硬件操作，因此应使用XNA Game Studio条件编译符号**# if**、**#elif**、**#else**、**#endif**指定。

例如，游戏，需要执行Windows、 Xbox 360和Windows Phone，但针对每个平台调用不同的服务（如输入设备），可以使用指令类似于下面的样式。

#if WINDOWS

// Execute code that is specific to Windows

#elif XBOX

// Execute code that is specific to Xbox 360

#elif WINDOWS\_PHONE

// Execute code that is specific to Windows Phone

#else

// Print a compile-time error message

#error The platform is not specified or is unsupported by this game.

#endif

## 性能优化

不同的硬件平台提供不同级别的性能。即使在同一平台上，例如PC上的处理能力、 图形功能和硬盘性能的不同也会造成应用程序的性能的差异。

通过代码对游戏的调整，包括放慢运行速度和减少使用的硬件等方式，可以为所有平台上的游戏玩家提供更完美的游戏体验。

### 硬件性能

下面的代码示例演示如何确定的计算机中可用的CPU数量，以及单处理器和多处理器计算机的执行各自特定的任务。

// Scale appropriately for the number of processors on this computer

if (Environment.ProcessorCount == 1)

{

// Perform tasks specific to single processor systems.

}

else

{

// Perform tasks specific to multi-processor systems.

}

### 运行效率

通过每秒调用 Update 方法的次数，可以降低游戏的性能级别。当低于默认的刷新频率（默认值为 60帧/秒 [fps]），需要将GameTime.IsRunningSlowly 设置为 **true**。

如果游戏的Update不适合60 帧/秒，可以通过修改TargetElapsedTime的值来更改刷新频率默认值。如果游戏刷新频率降低至不足30帧/秒时，需要在游戏的初始化代码将 IsRunningSlowly设置为Ture。

// Alert the program when the frame rate falls below 30 frames per second.

TargetElapsedTime = new TimeSpan(10000000L / 30L);

## 读写数据

XNA Game Studio类库不提供访问Windows Phone的数据存储，如果要实现数据存储需要使用**System.IO.IsolatedStorage**命名空间。

1. 添加引用的声明；

using System.IO.IsolatedStorage;

1. 使用IsolatedStorageFile.GetUserStoreForApplication获得IsolatedStorageFile对象创建文件夹和文件夹，以及读写文件；

在Windows中调用IsolatedStorageFile.GetUserStoreForApplication 会产生InvalidOperationException结果，因此需要GetUserStoreForDomain方法代替。

在跨平台编译时需要做如下的处理。

#if WINDOWS

IsolatedStorageFile savegameStorage = IsolatedStorageFile.GetUserStoreForDomain();

#else

IsolatedStorageFile savegameStorage = IsolatedStorageFile.GetUserStoreForApplication();

#endif

1. 使用IsolatedStorageFile.OpenFile打开文件，使用IsolatedStorageFileStream将数据写入文件。

protected override void OnExiting(object sender, System.EventArgs args)

{

// Save the game state (in this case, the typed text).

IsolatedStorageFile savegameStorage = IsolatedStorageFile.GetUserStoreForApplication();

// open isolated storage, and write the savefile.

IsolatedStorageFileStream fs = null;

fs = savegameStorage.OpenFile(SAVEFILENAME, System.IO.FileMode.Create);

if (fs != null)

{

// just overwrite the existing info for this example.

fs.WriteByte(gameState); // a single byte

if ((typedText != null) && (typedText.Length > 0))

{

fs.Write(System.Text.Encoding.UTF8.GetBytes(typedText), 0, typedText.Length);

}

fs.Close();

}

base.OnExiting(sender, args);

}

读取数据的方法类似。

protected override void Initialize()

{

gameState = 0;

IsolatedStorageFile savegameStorage = IsolatedStorageFile.GetUserStoreForApplication();

// open isolated storage, and write the savefile.

if(savegameStorage.FileExists(SAVEFILENAME))

{

IsolatedStorageFileStream fs = null;

try

{

fs = savegameStorage.OpenFile(SAVEFILENAME, System.IO.FileMode.Open);

}

catch (IsolatedStorageException e)

{

// The file couldn't be opened, even though it's there.

// You can use this knowledge to display an error message

// for the user (beyond the scope of this example).

}

if (fs != null)

{

// Reload the last state of the game. This consists of the UI mode

// and any text that was typed on the keyboard input screen.

byte[] saveBytes = new byte[256];

int count = fs.Read(saveBytes, 0, 256);

if (count > 0)

{

// the first byte is the mode, the rest is the string.

gameState = saveBytes[0];

if (count > 1)

{

typedText = System.Text.Encoding.UTF8.GetString(saveBytes, 1, count - 1);

}

fs.Close();

}

}

}

base.Initialize();

}

## 动手实践——组合Silverlight和XNA框架的3D应用

在Windows Phone Mango之前，不得不选择Silverlight或XNA的框架来构建Windows Phone应用程序，二者只能选其一。While some classes could be shared across frameworks, only one framework could be used for visuals.Starting with Windows Phone OS 7.1, you can combine Silverlight and the XNA Framework into a single application by using the new [SharedGraphicsDeviceManager](http://translate.googleusercontent.com/translate_c?hl=zh-CN&rurl=translate.google.com.hk&sl=auto&tl=zh-CN&u=http://go.microsoft.com/fwlink/%3FLinkId%3D219277&usg=ALkJrhgFhfGDZ9O0tnQgIoo1yXmADq9iMg) and the [UIElementRenderer](http://translate.googleusercontent.com/translate_c?hl=zh-CN&rurl=translate.google.com.hk&sl=auto&tl=zh-CN&u=http://go.microsoft.com/fwlink/%3FLinkId%3D217352&usg=ALkJrhguaXcd6IG1Olix5GXo806SH_iIDw) class.从Windows Phone Mango开始，游戏应用程序可以将Silverlight和XNA框架组合到SharedGraphicsDeviceManager和UIElementRenderer的新类中使用。

曾经看到一部茶道的宣传短片印象深刻，其中将泡茶的整个过程做成了3D模拟的动画讲述茶道。如果我们在Windows Phone 智能手机中制作一款茶道游戏，“以茶行道，以茶雅志”，那将是多美好的一件事。中国茶道追求“真”，通过直觉体悟达到对人生、对功利的精神上的超越，自省其身。茶道本身犹如程序逻辑，程序人生犹如饮茶品茗，勿忘子曰：“君子喻于义，小人喻于利。”

本例中我们组合Silverlight或XNA的框架开发一款3D茶具，作为Windows Phone茶道游戏的引子，为读者实现茶道游戏感悟程序人生起到抛砖引玉的作用。

### 新建游戏应用程序

1. 确认已经安装Windows Phone 操作系统开发平台，详细信息请见[Installing Windows Phone Developer Tools](http://msdn.microsoft.com/en-us/library/ff402530(v=VS.92).aspx)。
2. 启动Visual Studio 2010 Express或者Visual Studio 2010。如果出现注册窗体，您可以注册或暂时关闭该窗体。
3. 通过菜单创建新应用程序，选择**File | New Project**。

在**New Project**的窗口展开**Visual C＃**模板，然后选择**Silverlight for Windows Phone**模板。

1. 选择**Windows Phone 3D Graphics Application**模板，填写项目名称为MyLittleTeapot。

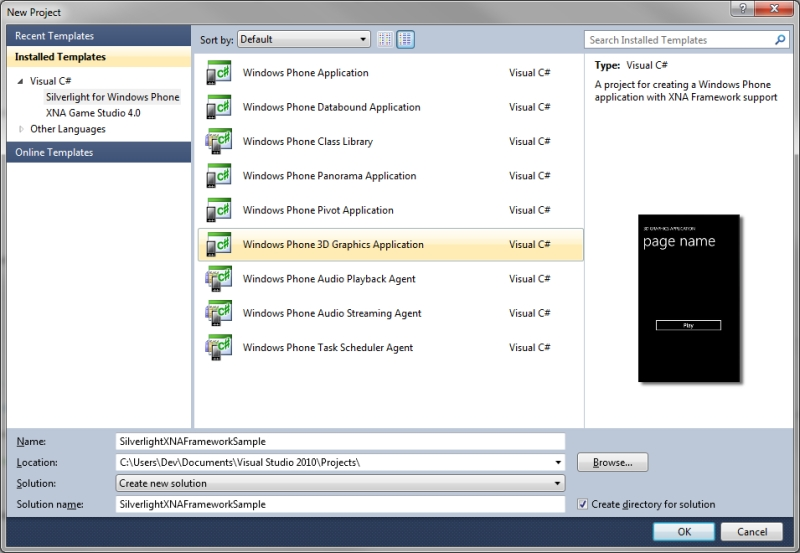


图15-2 新建应用程序

点击**OK**，Visual Studio创建一个新的项目并打开**MainPage.xaml**。

1. 选择Windows Phone运行目标环境为**Windows Phone Emulator**。

### 加载3D类

增加名称为3D Primitives的文件夹在，为此[Solution Explorer]中右键点击项目名称MyLittleTeapot，选择 [Add | New Folder]，将新建的文件夹命名为3D Primitives。

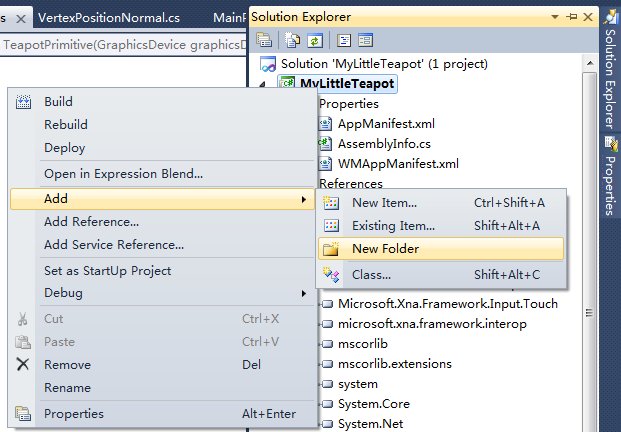


图15-3 新建文件夹

将MyLittleTeapot\Assets\3D Primitives文件夹中的类BezierPrimitive.cs、GeometricPrimitive.cs、TeapotPrimitive.cs和VertexPositionNormal.cs，添加到工程中的3D Primitives文件夹下。添加的结果如图 3D类。

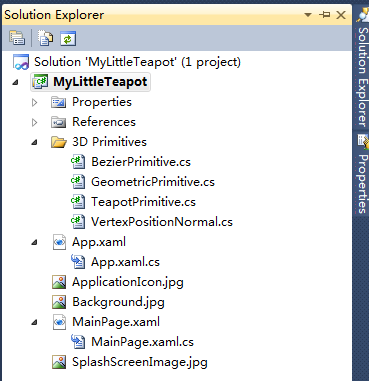


图15-4 3D类

### 加载3D图形

在**[Solution Expplorer]**中打开MainPage.xaml.cs，在MainPage类的构造函数之前加入如下代码，声明3D模型、ContentManager、GameTimer、SpriteBatch和UI元素渲染类。

在MainPage类的构造函数中，使用ContentManager实现了原先XNA程序的Content，使用GameTimer控制游戏的Update，并添加UIElementRenderer的Update事件。

Silverlight/XNA Project: SudukoArticle File: MainPage.xaml.cs

GameTimer timer;

ContentManager content;

SpriteBatch spriteBatch;

// 3D teapot we display behind the Silverlight page

TeapotPrimitive teapot;

Color teapotColor = Color.Black;

float teapotYaw, teapotPitch;

// Indicates if the controls are visible

bool panelVisible = true;

// For rendering the XAML

UIElementRenderer elementRenderer;

// Constructor

public MainPage()

{

InitializeComponent();

// Get the application's ContentManager

content = (Application.Current as App).Content;

// Create a timer for this page

timer = new GameTimer();

timer.UpdateInterval = TimeSpan.Zero;

timer.Update += OnUpdate;

timer.Draw += OnDraw;

// Use the LayoutUpdate event to know when the page layout has completed so we can

// create the UIElementRenderer

LayoutUpdated += new EventHandler(MainPage\_LayoutUpdated);

}

重载OnNavigatedTo方法，调用SharedGraphicsDeviceManager的SetSharingMode方法传递参数true，将Silverlight的图形驱动使用XNA类库渲染。在当前图形驱动中创建3D模型，启动游戏时间控制。

Silverlight/XNA Project: SudukoArticle File: MainPage.xaml.cs

protected override void OnNavigatedTo(NavigationEventArgs e)

{

// Set the sharing mode of the graphics device to turn on XNA rendering

SharedGraphicsDeviceManager.Current.GraphicsDevice.SetSharingMode(true);

// Create a SpriteBatch for rendering content

spriteBatch = new SpriteBatch(SharedGraphicsDeviceManager.Current.GraphicsDevice);

// Create the teapot

teapot = new TeapotPrimitive(SharedGraphicsDeviceManager.Current.GraphicsDevice);

// Start the GameTimer

timer.Start();

base.OnNavigatedTo(e);

}

重载OnNavigatedFrom方法，停止游戏时间控制，调用SharedGraphicsDeviceManager的SetSharingMode方法传递参数false。

Silverlight/XNA Project: SudukoArticle File: MainPage.xaml.cs

protected override void OnNavigatedFrom(NavigationEventArgs e)

{

// Stop the GameTimer

timer.Stop();

// Set the sharing mode of the graphics device to turn off XNA rendering

SharedGraphicsDeviceManager.Current.GraphicsDevice.SetSharingMode(false);

base.OnNavigatedFrom(e);

}

### 加载Sivlerlight控件

在**[Solution Expplorer]**中打开MainPage.xaml，将XAML的代码替换为如下的代码。添加三个颜色的进度条、四个控制颜色按钮和一个控制颜色面板的按钮。



图15-6 MainPage

Silverlight/XNA Project: SudukoArticle File: MainPage.xaml

<Grid x:Name="LayoutRoot" Background="#000084FF">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="Auto"/>

</Grid.RowDefinitions>

<Button Click="TogglePanelVisibility" Margin="1,0,-1,0">Toggle Color Panel</Button>

<!-- Element to allow us to rotate the teapot with touch input -->

<Canvas Grid.Row="1" ManipulationDelta="TeapotManipulationDelta" Background="Transparent" />

<!-- The slider minimum is set to -20 and maximum to 275 because it is

difficult to grab the slider control when at it's maximum or minimum

values. So a buffer has been added to the desired range of 0-255 -->

<StackPanel x:Name="stackPanel" Grid.Row="2" Margin="0" RenderTransformOrigin="0,0">

<StackPanel.RenderTransform>

<CompositeTransform/>

</StackPanel.RenderTransform>

<Slider Name="sliderRed"

Minimum="-20" Maximum="275" Value="50"

SmallChange="5" LargeChange="20"

Background="Red" Foreground="Red"

ValueChanged="slider\_ValueChanged" />

<Slider Name="sliderGreen"

Minimum="-20" Maximum="275" Value="50"

SmallChange="5" LargeChange="20"

Background="Green" Foreground="Lime"

ValueChanged="slider\_ValueChanged" />

<Slider Name="sliderBlue"

Minimum="-20" Maximum="275" Value="50"

SmallChange="5" LargeChange="20"

Background="Blue" Foreground="Blue"

ValueChanged="slider\_ValueChanged" />

<StackPanel Height="100" Orientation="Horizontal" HorizontalAlignment="Center">

<!-- Buttons to set the teapot to specific colors -->

<Button Content="" Click="redButton\_Click" HorizontalAlignment="Center" Height="75" VerticalAlignment="Center" BorderThickness="3" Background="Red" Width="75" />

<Button Content="" Click="greenButton\_Click" HorizontalAlignment="Center" Height="75" VerticalAlignment="Center" BorderThickness="3" Background="Lime" Width="75" />

<Button Content="" Click="blueButton\_Click" HorizontalAlignment="Center" Height="75" VerticalAlignment="Center" BorderThickness="3" Background="Blue" Width="75" />

<Button Content="" Click="blackButton\_Click" HorizontalAlignment="Center" Height="75" VerticalAlignment="Center" BorderThickness="3" Background="Black" Width="75" />

</StackPanel>

</StackPanel>

</Grid>

在MainPage.xaml中添加动画效果。

Silverlight/XNA Project: SudukoArticle File: MainPage.xaml

<phone:PhoneApplicationPage.Resources>

<Storyboard x:Name="AnimatePanelOut">

<DoubleAnimation Duration="0:0:0.7" To="356" Storyboard.TargetProperty="(UIElement.RenderTransform).(CompositeTransform.TranslateY)" Storyboard.TargetName="stackPanel" d:IsOptimized="True">

<DoubleAnimation.EasingFunction>

<BackEase EasingMode="EaseInOut"/>

</DoubleAnimation.EasingFunction>

</DoubleAnimation>

</Storyboard>

<Storyboard x:Name="AnimatePanelIn">

<DoubleAnimationUsingKeyFrames Storyboard.TargetProperty="(UIElement.RenderTransform).(CompositeTransform.TranslateY)" Storyboard.TargetName="stackPanel">

<EasingDoubleKeyFrame KeyTime="0" Value="356"/>

<EasingDoubleKeyFrame KeyTime="0:0:.5" Value="0">

<EasingDoubleKeyFrame.EasingFunction>

<BackEase EasingMode="EaseInOut"/>

</EasingDoubleKeyFrame.EasingFunction>

</EasingDoubleKeyFrame>

</DoubleAnimationUsingKeyFrames>

</Storyboard>

</phone:PhoneApplicationPage.Resources>

### 事件处理

在MainPage.xaml.cs中重载OnDraw方法。

Silverlight/XNA Project: SudukoArticle File: MainPage.xaml.cs

/// <summary>

/// Allows the page to draw itself.

/// </summary>

private void OnDraw(object sender, GameTimerEventArgs e)

{

// Render the Silverlight controls using the UIElementRenderer

elementRenderer.Render();

// Clear the screen to a solid color

SharedGraphicsDeviceManager.Current.GraphicsDevice.Clear(Color.CornflowerBlue);

// Draw the teapot

DrawTeapot(e);

spriteBatch.Begin();

// Using the texture from the UIElementRenderer,

// draw the Silverlight controls to the screen

spriteBatch.Draw(elementRenderer.Texture, Vector2.Zero, Color.White);

spriteBatch.End();

}

实现OnDraw 方法中调用的3D茶壶的方法DrawTeapot。

Silverlight/XNA Project: SudukoArticle File: MainPage.xaml.cs

private void DrawTeapot(GameTimerEventArgs e)

{

float aspectRatio = SharedGraphicsDeviceManager.Current.GraphicsDevice.Viewport.AspectRatio;

// Construct the world, view, and projection matrices

Matrix world = Matrix.CreateFromYawPitchRoll(teapotYaw, teapotPitch, 0f);

Matrix view = Matrix.CreateLookAt(new Vector3(0, 0, 2.5f), Vector3.Zero, Vector3.Up);

Matrix projection = Matrix.CreatePerspectiveFieldOfView(1, aspectRatio, 1, 10);

// Draw the teapot

teapot.Draw(world, view, projection, teapotColor);

}

添加Silverlight的UI控件事件响应函数。

Silverlight/XNA Project: SudukoArticle File: MainPage.xaml.cs

private void SetTeapotColor(Color c)

{

teapotColor = c;

sliderBlue.Value = c.B;

sliderGreen.Value = c.G;

sliderRed.Value = c.R;

}

private void redButton\_Click(object sender, RoutedEventArgs e)

{

SetTeapotColor(Color.Red);

}

private void greenButton\_Click(object sender, RoutedEventArgs e)

{

SetTeapotColor(Color.Lime);

}

private void blueButton\_Click(object sender, RoutedEventArgs e)

{

SetTeapotColor(Color.Blue);

}

private void blackButton\_Click(object sender, RoutedEventArgs e)

{

SetTeapotColor(Color.Black);

}

private void slider\_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)

{

if (sliderRed == null || sliderGreen == null || sliderBlue == null)

return;

SetTeapotColor(new Color((int)sliderRed.Value, (int)sliderGreen.Value, (int)sliderBlue.Value));

}

private void TogglePanelVisibility(object sender, RoutedEventArgs e)

{

if (panelVisible)

{

AnimatePanelOut.Completed += new EventHandler(AnimatePanelOut\_Completed);

AnimatePanelOut.Begin();

panelVisible = false;

}

else

{

AnimatePanelIn.Begin();

stackPanel.Visibility = System.Windows.Visibility.Visible;

panelVisible = true;

}

}

void AnimatePanelOut\_Completed(object sender, EventArgs e)

{

stackPanel.Visibility = System.Windows.Visibility.Collapsed;

}

使用Silverlight的手势输入响应用户的手指触控。

Silverlight/XNA Project: SudukoArticle File: MainPage.xaml

<!-- Element to allow us to rotate the teapot with touch input -->

<Canvas Grid.Row="1" ManipulationDelta="TeapotManipulationDelta" Background="Transparent" />

Silverlight/XNA Project: SudukoArticle File: MainPage.xaml.cs

// This event handler is hooked up in the XAML

private void TeapotManipulationDelta(object sender, ManipulationDeltaEventArgs e)

{

teapotYaw = MathHelper.WrapAngle(teapotYaw + (float)e.DeltaManipulation.Translation.X \* .01f);

teapotPitch = MathHelper.WrapAngle(teapotPitch + (float)e.DeltaManipulation.Translation.Y \* .01f);

}

### 渲染Silverlight控件

前来讲到我们在MainPage.xaml.cs中声明UIElementRenderer类，并在MainPage的构造函数中添加了LayoutUpdated事件处理关联函数MainPage\_LayoutUpdated。下面实现MainPage\_LayoutUpdated方法。

UIElementRenderer类实现渲染Silverlight UI元素为Texture2D。

Silverlight/XNA Project: SudukoArticle File: MainPage.xaml.cs

void MainPage\_LayoutUpdated(object sender, EventArgs e)

{

// Create the UIElementRenderer to draw the Silverlight page to a texture.

// Verify the page has a valid size

if (ActualWidth <= 0 && ActualHeight <= 0)

{

return;

}

int width = (int)ActualWidth;

int height = (int)ActualHeight;

// See if the UIElementRenderer is already the page's size

if ((elementRenderer != null) &&

(elementRenderer.Texture != null) &&

(elementRenderer.Texture.Width == width) &&

(elementRenderer.Texture.Height == height))

{

return;

}

// Dispose the UIElementRenderer before creating a new one

if (elementRenderer != null)

{

elementRenderer.Dispose();

}

elementRenderer = new UIElementRenderer(this, width, height);

}

### 在模拟器中运行

按F5运行应用程序，或者点击Start Debugging按钮运行，如图15-7 Start Debugging。



图15-7 Start Debugging

模拟器中显示的3D茶壶。

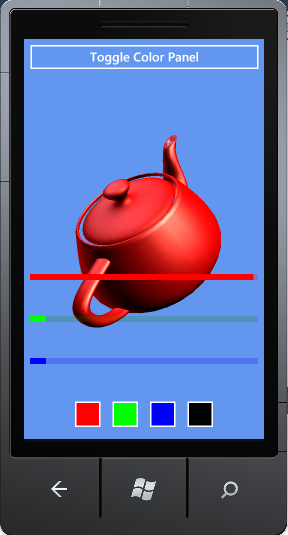
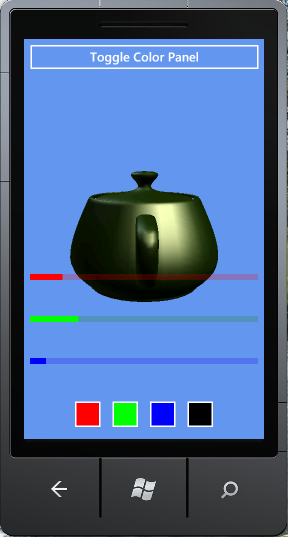
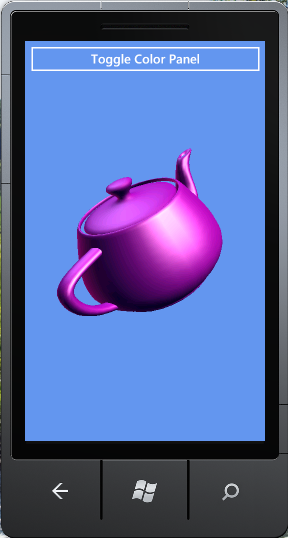
  

图15-8 3D茶具旋转