Windows Phone Mango编程实践

***Windows Phone Mango Programming Practice***

第一篇 基础篇

# Windows Phone入门——探索火星

《周易》曰：“云雷，屯。君子以经轮。”意思是说乌云骤起，雷声交动，象征着事业初创的艰苦时期。君子应铁肩担道义，负起经略天下的大义。移动互联时代的竞争是生态系统的角斗和资源整合能力的考验，作为移动互联网竞争的制高点——智能终端操作系统三足鼎立的局面已然形成，此时正是山雨欲来风起云涌之时，正是英雄开创事业之初。

## 概述——开创新领域

《周易》曰：“云雷，屯。君子以经轮。”意思是说乌云骤起，雷声交动，象征着事业初创的艰苦时期。君子应铁肩担道义，负起经略天下的大义。移动互联时代的竞争是生态系统的角斗和资源整合能力的考验，作为移动互联网竞争的制高点——智能终端操作系统三足鼎立的局面已然形成，此时正是山雨欲来风起云涌之时，正是英雄开创事业之初。

正如我之前在Windows Phone OS 7.0的开发中使用Silverlight和XNA应用程序时所体验到的那样，我很快就认定这是一个很好的发展方向。虽然我接触这两个平台以及托管代码的时间并不长，但如此快速的开发出视觉效果如此出色的应用程序还是让我印象深刻。这一全新应用程序平台的另一个值得关注的特性是，硬件标准化以及以编程方式对其进行访问的标准化。具体而言，Windows Phone将支持开发人员能够以统一和可靠的方式访问的一组核心硬件。

Widows Phone的到来乃顺势而为，作为以软件创新改变世界的程序员而言，岂能只作壁上观。智者顺势而谋，抓紧天下光明的时机，明断是非，赶紧做事。

本书开篇Windows Phone的入门的应用程序——探索火星，是非常有趣的应用程序。Windows Phone是基于云计算的智能终端操作系统，探索火星应用程序正是与Windows Azure平台的Dallas提供的数据服务通讯，将美国国家航空航天局（National Aeronautics and Space Administration，NASA）提供的火星探测行动拍摄到的图像呈现给大家。NASA 提供的有关火星任务的图像数据非常有趣，开发浏览火星漫游图片的Windows Phone应用程序，感觉很棒。

## 什么是微软的“Dallas”？

微软的“Dallas”的社区技术预览（CTP3），是由Windows Azure 和SQL Azure构建的信息服务，能够让开发者与信息工作者在任何平台上使用优质的第三方数据集和内容。

在Windows Azure 平台 (Windows Azure, SQL Azure Database)强大功能和规模下, 通过结合非云端与云端的无关的私有或共有的数据，“Dallas”赋予开发者通过桌面或移动设备，来构建和管理创新的应用程序的能力。通过单一市场，“Dallas” 使得开发者通过访问复杂数据集来构建全新的分析与报表方案。内容提供商也可以在全球水平上，将他们的数据提供给数百万开发者，这将带来新的增长与获利机会。

“Dallas”项目（现为Windows Azure Marketplace）是微软提供的数据交易和分享平台,即能够让数据提供商通过“Dallas”平台公布API使数据消费者以OData协议共享数据。Dallas CTP 3的发布，带来大量令人兴奋的改进，开发使用从“Dallas””订阅数据的应用程序变得更加容易，并增加大量的新功能。

“Dallas”是如何运作？

“Dallas”在一个统一标准的供给与收费框架下,将来自于领先的商业数据提供商与权威的公共数据源的数据与影像引入至单个位置。另外，“Dallas”API允许开发者和信息工作者在任何平台、应用程序、业务工作流上使用这些优质信息 。此外，“Dallas”允许Office Excel 和SQL Server 用户立刻可以将私有数据与“Dallas”数据混合，来创建新的围绕分析与报表的方案。

## 动手实践初探Dallas——探索火星

### 先决条件

编程开始之前我们需要注册Dallas账户，订阅Dallas数据服务。

Dallas的注册网址：<https://www.sqlazureservices.com/>

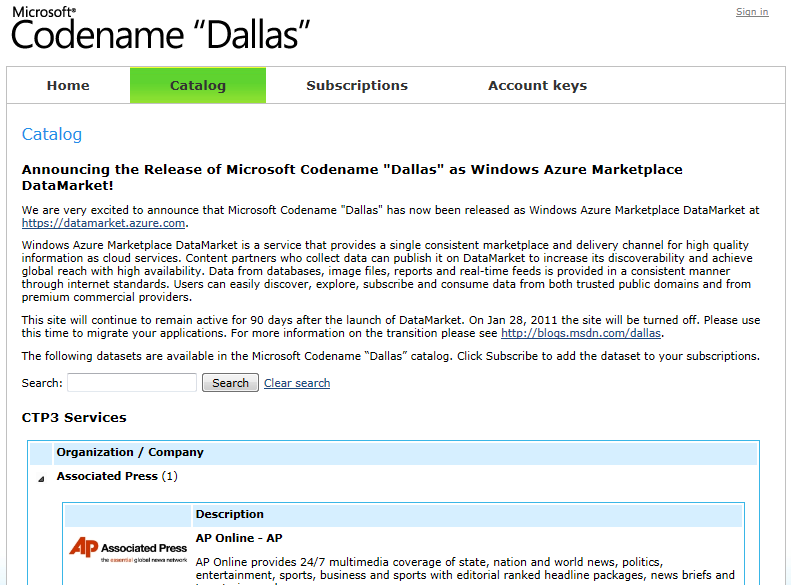


图2-1 Dallas网页

使用Windows Live登陆后进入注册页面，填入用户信息。

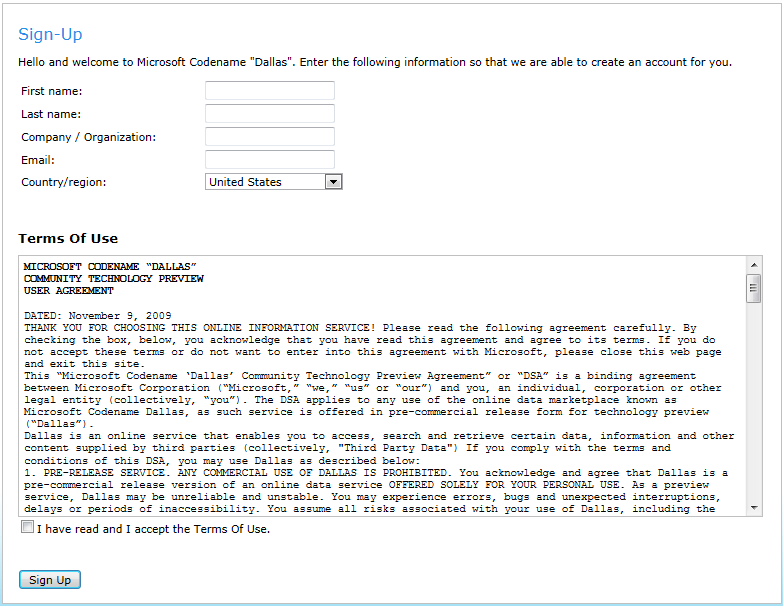


图2-2 注册页面

注册成功后会自动生成默认的Account Keys，对于开发者而言，就此拥有了通行另一个奇妙时空的钥匙。在Account Keys的页面，有一篇重要的文章指导我们整合应用程序由CTP2向CTP3升级*Migrating your application from CTP2 to CTP3*。如果之前您做过Dallas CTP2的应用程序，那么花时间了解Dallas的变化是很有必要的。

点击“Edit”编译账户。

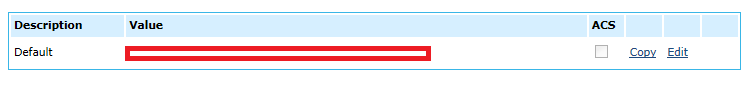


图2-3 编译账户

在账户编辑修改账户的信息，包括描述和选择“Enable this key for use with the Access Control Service”属性，点击**[Save changes]**。

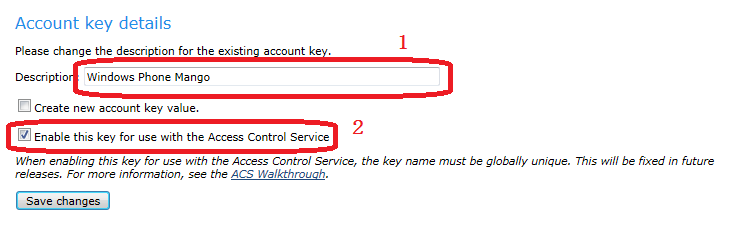


图2-4 Account Key

在Catalog页面，搜索美国宇航局NASA提供的数据服务，并订阅该数据服务，点击Subscribe订阅。

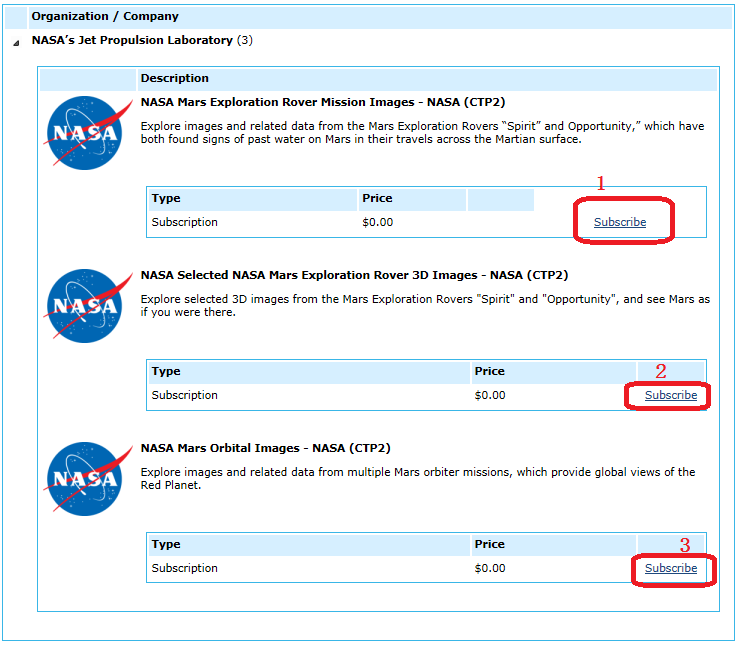


图2-5 订阅NASA服务

### 创建程序

以管理员身份打开**Visual Studio 2010**或者**Visual Studio 2010 Express**，新建**Silverlight for Windows Phone**工程，类型为**Windows Phone Application**，工程名称为[**MarsImageViewer**]，如图2-6 新建Windows Phone应用程序。

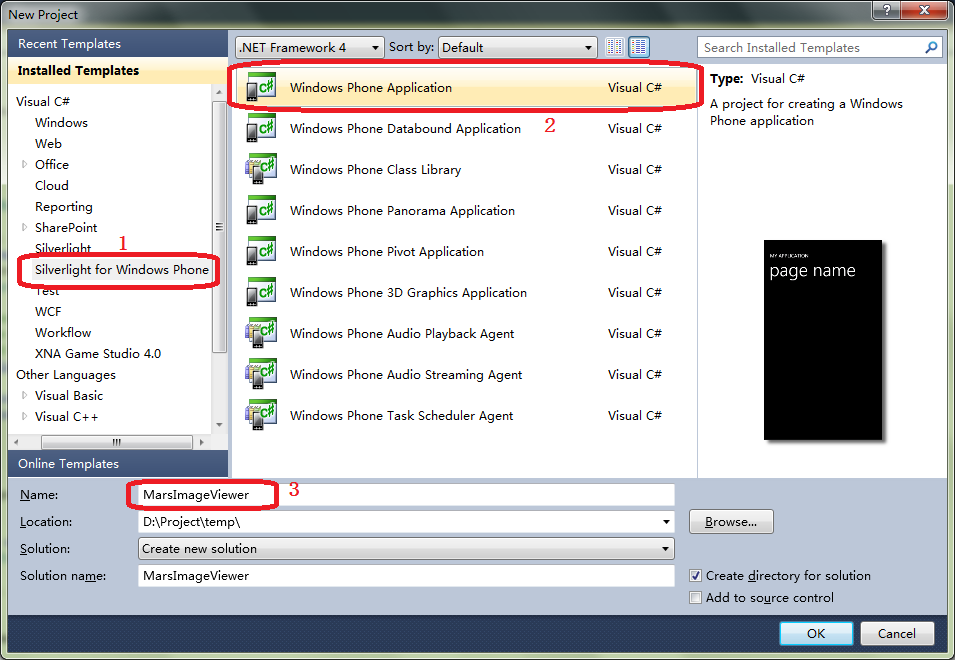


图2-6 新建Windows Phone应用程序

如果之前使用过 Windows Presentation Foundation (WPF) 或 Silverlight，则您应该不会对看到的内容感到陌生。您会看到一个具有流行手机外观的设计图面、一个具有一些基本控件的工具箱以及多个具有其相关C# 代码分离文件的 XAML 文件。

### 添加资源

在工程中添加Images文件夹，存放图片资源。在**[MarsImageViewer]**点击右键，选择**[Add]**→**[New Folder]**，设定新文件夹的名称为Images。

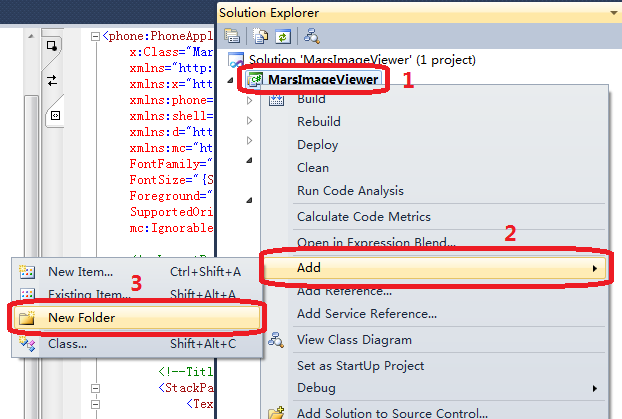


图2-7 添加Images文件夹

在Images文件夹中添加Assert\Images文件夹下的所有图片资源，并修改图片的属性[Build Action]为“Content”，[Copy to Output Directory]为“Copy if newer”，如图2-8 修改图片资源属性。

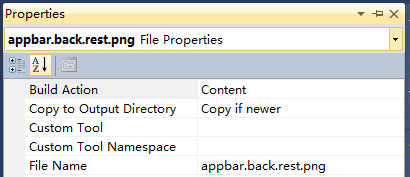


图2-8 修改图片资源属性

### 页面布局

与使用 Silverlight 相同，Windows Phone 应用程序模板提供了一个 App.xaml 文件和一个 MainPage.xaml 文件。打开MainPage.xaml文件，将首页的布局按照如下的代码修改。

Silverlight Project: MarsImageViewer File: MainPage.xaml

<Grid x:Name="LayoutRoot" Background="Transparent">

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<!—TitlePanel包含应用程序的名称和页面的标题-->

<StackPanel x:Name="TitlePanel" Grid.Row="0" Margin="24,24,0,12">

<TextBlock x:Name="PageTitle"

Text="探索火星"

Margin="-3,-8,0,0"

Style="{StaticResource PhoneTextTitle2Style}"/>

</StackPanel>

<!--ContentPanel -->

<Grid x:Name="ContentGrid" Grid.Row="1">

<Image Height="619"

HorizontalAlignment="Left" Name="MarsImage"

Stretch="Fill"

VerticalAlignment="Top"

Width="480" />

</Grid>

</Grid>

此布局中有一个标题“探索火星”和显示图片的Image控件。下面我们添加应用程序栏，关于应用程序栏的更多内容在后续的章节介绍。

Silverlight Project: MarsImageViewer File: MainPage.xaml

<phone:PhoneApplicationPage.ApplicationBar>

<shell:ApplicationBar IsVisible="True" IsMenuEnabled="False" Opacity="0">

<shell:ApplicationBarIconButton

x:Name="appbar\_BackButton"

IconUri="/Images/appbar.back.rest.png"

Text="Back"

Click="appbar\_BackButton\_Click">

</shell:ApplicationBarIconButton>

<shell:ApplicationBarIconButton

x:Name="appbar\_ForwardButton"

IconUri="/Images/appbar.next.rest.png"

Text="Next"

Click="appbar\_ForwardButton\_Click">

</shell:ApplicationBarIconButton>

<shell:ApplicationBarIconButton

x:Name="appbar\_AboutButton"

IconUri="/Images/appbar.feature.settings.rest.png"

Text="Help"

Click="appbar\_AboutButton\_Click">

</shell:ApplicationBarIconButton>

</shell:ApplicationBar>

</phone:PhoneApplicationPage.ApplicationBar>

在应用程序栏中，设定了三个“Back”、“Next”和“Help”按钮，其中Help按钮的触发事件中实现页面跳转。

设定页面支持横向视图和纵向视图。在XAML中设定SupportedOrientations属性为PortraitOrLandscape，那么应用程序会根据Windows Phone手机的重力感应，自动选择页面显示的方式为横行视图还是纵向视图。本例中设定默认的视图方式为纵向视图，同时也支持横向视图。

Silverlight Project: MarsImageViewer File: MainPage.xaml

SupportedOrientations="PortraitOrLandscape" Orientation="Portrait"

### 与Dallas通信

在MainPage.xaml.cs文件中添加引用。

Silverlight Project: MarsImageViewer File: MainPage.xaml.cs

using System.Xml.Linq;

using System.IO;

using System.Windows.Media.Imaging;

using System.Net.Browser;

using System.Xml;

私有变量定义，包括显示图片的列表和索引，以及与Dallas通讯所用的Windows Live ID和Account Key。

Silverlight Project: MarsImageViewer File: MainPage.xaml.cs

private int index;

private List<string> ImageIdList;

private string MarsCurrentImageId;

private BitmapImage imgsrc;

/\* Microsoft Project Codename “Dallas” changes the way information is exchanged by offering a wide range of content

from authoritative commercial & public sources in a single marketplace, making it easier to find and purchase the

data that you need to power your applications and analytics. \*/

private const string WindowsLiveId = "<Your Windows Live ID>";

private const string AccountKey = "<Your account key>";

getImageIDs 函数旨在启动Web服务检索图像索引信息。在MainPage的构造函数中，调用getImageIDs方法与Dallas通讯获取图片ID列表，在HttpWebRequest.BeginGetResponse 方法中异步处理获取图片的方法getImage。

Silverlight Project: MarsImageViewer File: MainPage.xaml.cs

// Constructor

public MainPage()

{

InitializeComponent();

index = 0;

ImageIdList = new List<string>();

imgsrc = new BitmapImage();

//获取图像ID

getImageIDs();

}

private void getImageIDs()

{

Uri serviceUri = new Uri("https://api.sqlazureservices.com/NasaService.svc/MER/Images?missionId=1&$format=atom10");

// 创建向dallas发送的请求

WebRequest recDownloader = (HttpWebRequest)WebRequestCreator.ClientHttp.Create(serviceUri);

// 身份验证

recDownloader.Credentials = new NetworkCredential(WindowsLiveId, AccountKey);

// 获取服务器返回的数据

recDownloader.BeginGetResponse(ar =>

{

// 获取httpwebrequest

HttpWebRequest r = (HttpWebRequest)ar.AsyncState;

// 服务器的应答

HttpWebResponse response = null;

try

{

response = (HttpWebResponse)r.EndGetResponse(ar);

}

catch (Exception ex)

{

// returned an error.

this.Dispatcher.BeginInvoke(delegate() { MessageBox.Show(ex.Message); });

return;

}

// 读取服务器应答数据

using (StreamReader reader = new StreamReader(response.GetResponseStream(), true))

{

// 图像ID列表

XmlReader xmlReader = XmlReader.Create(reader);

xmlReader.Read();

while (xmlReader.ReadState != ReadState.EndOfFile)

{

xmlReader.ReadToFollowing("entry");

if (xmlReader.ReadState != ReadState.EndOfFile)

{

xmlReader.ReadToFollowing("m:properties");

xmlReader.ReadToFollowing("d:ImageId");

ImageIdList.Add(xmlReader.ReadElementContentAsString());

}

}

xmlReader.Close();

reader.Close();

foreach (string ImageId in ImageIdList)

{

MarsCurrentImageId = ImageId;

break;

}

Deployment.Current.Dispatcher.BeginInvoke(() =>

{

// get image by ID

if (MarsCurrentImageId != null)

{

getImage(MarsCurrentImageId);

}

});

}

}, recDownloader);

}

getImage函数类似于getImageIDs函数，其区别在于请求Web服务的URL。getImage函数调用Web资源的异步请求，将检索到的ID号作为查询条件获取图片资源，并调用ShowImage函数在MainPage.xaml的Image控件中显示图片。

Silverlight Project: MarsImageViewer File: MainPage.xaml.cs

private void getImage(string ID)

{

if (ID == null)

{

return;

}

Uri serviceUri = new Uri("https://api.sqlazureservices.com/NasaService.svc/MER/Images/" + ID + "?$format=raw");

WebRequest imgDownloader = (HttpWebRequest)WebRequestCreator.ClientHttp.Create(serviceUri);

// 身份验证

imgDownloader.Credentials = new NetworkCredential(WindowsLiveId, AccountKey);

// 获取服务器返回的数据

imgDownloader.BeginGetResponse(br =>

{

// 获取请求

HttpWebRequest r = (HttpWebRequest)br.AsyncState;

// 服务器应答

HttpWebResponse response = null;

try

{

response = (HttpWebResponse)r.EndGetResponse(br);

}

catch (Exception ex)

{

// 异常捕获

this.Dispatcher.BeginInvoke(delegate() { MessageBox.Show(ex.Message); });

return;

}

Stream ImageStream = response.GetResponseStream();

ShowImage(ImageStream, response.ContentLength);

}, imgDownloader);

}

Silverlight只能在UI Thread中更新显示控件属性。多线程编码时，需要借助于Dispatcher实现跨线程访问。

Silverlight Project: MarsImageViewer File: MainPage.xaml.cs

private void ShowImage(Stream imageStream, long imageSize)

{

BinaryReader br = new BinaryReader(imageStream);

byte[] ImageBytes = new byte[imageSize];

br.Read(ImageBytes, 0, ImageBytes.Length);

MemoryStream msa = new MemoryStream(ImageBytes);

Deployment.Current.Dispatcher.BeginInvoke(delegate

{

imgsrc.SetSource(msa);

MarsImage.Source = imgsrc;

});

}

### 事件处理

这一小节中我们增加应用程序栏的触控事件处理。应用程序栏的“Back”和“Next”按钮点击时，显示图像列表中对应上一张和下一张的探索火星的照片。

Silverlight Project: MarsImageViewer File: MainPage.xaml.cs

private void appbar\_BackButton\_Click(object sender, EventArgs e)

{

if (index > 0)

{

index--;

MarsCurrentImageId = ImageIdList[index];

getImage(MarsCurrentImageId);

}

}

private void appbar\_ForwardButton\_Click(object sender, EventArgs e)

{

if ((index + 1) < ImageIdList.Count)

{

index++;

MarsCurrentImageId = ImageIdList[index];

getImage(MarsCurrentImageId);

}

}

在MainPage.xaml中关联事件处理函数的代码如下。

Silverlight Project: MarsImageViewer File: MainPage.xaml

<phone:PhoneApplicationPage.ApplicationBar>

<shell:ApplicationBar IsVisible="True" IsMenuEnabled="False" Opacity="0">

<shell:ApplicationBarIconButton

x:Name="appbar\_BackButton"

IconUri="/Images/appbar.back.rest.png"

Text="Back"

Click="appbar\_BackButton\_Click">

</shell:ApplicationBarIconButton>

<shell:ApplicationBarIconButton

x:Name="appbar\_ForwardButton"

IconUri="/Images/appbar.next.rest.png"

Text="Next"

Click="appbar\_ForwardButton\_Click">

</shell:ApplicationBarIconButton>

<shell:ApplicationBarIconButton

x:Name="appbar\_AboutButton"

IconUri="/Images/appbar.feature.settings.rest.png"

Text="Help"

Click="appbar\_AboutButton\_Click">

</shell:ApplicationBarIconButton>

</shell:ApplicationBar>

</phone:PhoneApplicationPage.ApplicationBar>

### 页面跳转

应用程序栏的“Help”按钮我们实现页面跳转，由MainPage.xaml跳转至About.xaml。About.xaml显示软件的相关制作信息，同时在About.xaml中的应用程序栏添加返回MainPage.xaml的事件处理。跳转的方式为使用导航服务实现页面跳转。

Silverlight Project: MarsImageViewer File: MainPage.xaml.cs

private void appbar\_AboutButton\_Click(object sender, EventArgs e)

{

NavigationService.Navigate(new Uri("/About.xaml ", UriKind.Relative));

}

在About.xaml中的跳转中，由导航服务判断页面是否可以后退，如果可以的话执行后退操作。在本例中我们设置了两个页面，进入About.xaml的先前页面肯定是MainPage.xaml。但是有可能程序发生由休眠状态被重新激活，为避免导航失败，如果应用程序导航无法回退时，指定导航的URI。

Silverlight Project: MarsImageViewer File: About.xaml.cs

private void appbar\_BackButton\_Click(object sender, EventArgs e)

{

if (NavigationService.CanGoBack)

{

NavigationService.GoBack();

}

else

{

NavigationService.Navigate(new Uri("/MainPage.xaml ", UriKind.Relative));

}

}

### 调试应用程序

按F5运行应用程序，或者点击Start Debugging按钮运行，如图2-9 Start Debugging。



图2-9 Start Debugging

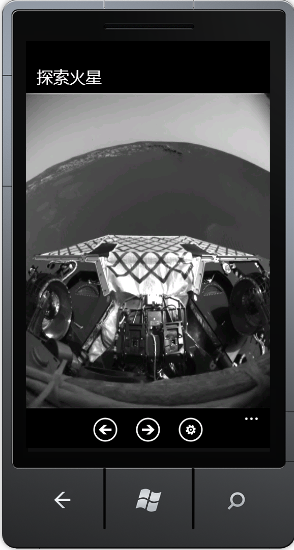
 

图2-10 应用程序运行结果

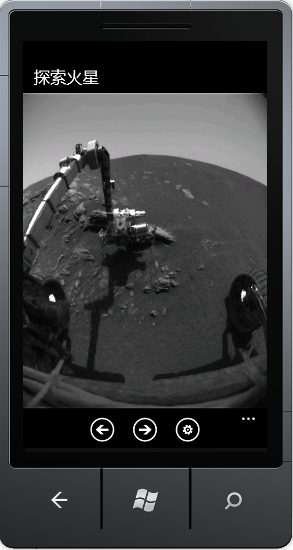
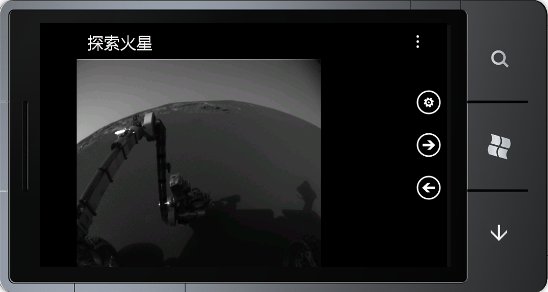
 

图2-11 纵向视图和横向视图