**Silverlight 客戶端桌面模式**

(OOB 模式)

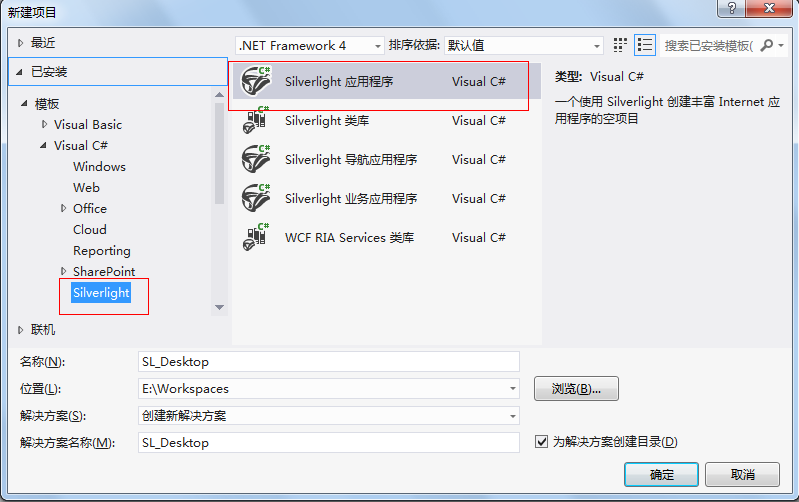
Silverlight 很酷的名字，在以前版本 Silverlight 只能承載在Web 上面，由於科技不斷的更新現在Silverlight也可以做成桌面版模式去運行，是外觀可以更好的給用戶體現，還可以調用用戶一些公用的軟件如 Office 等功能。

本篇學習文章只介紹如何創建一個Silverlight 桌面程序 (OOB模式)。

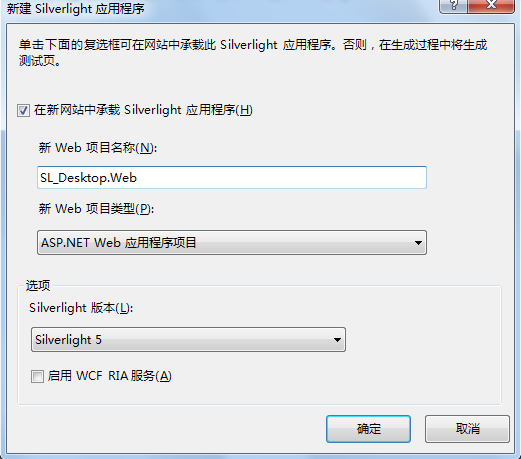
開發條件：必須是Silverlight 4.0 或以上版本，使用VS 2010 SP1或以上版本。

本篇文章使用的是 VS 2012 和 Silverlight 5.0因爲使用VS 2010 SP1 安裝上比較麻煩。

第一步:創建項目方案 如圖：



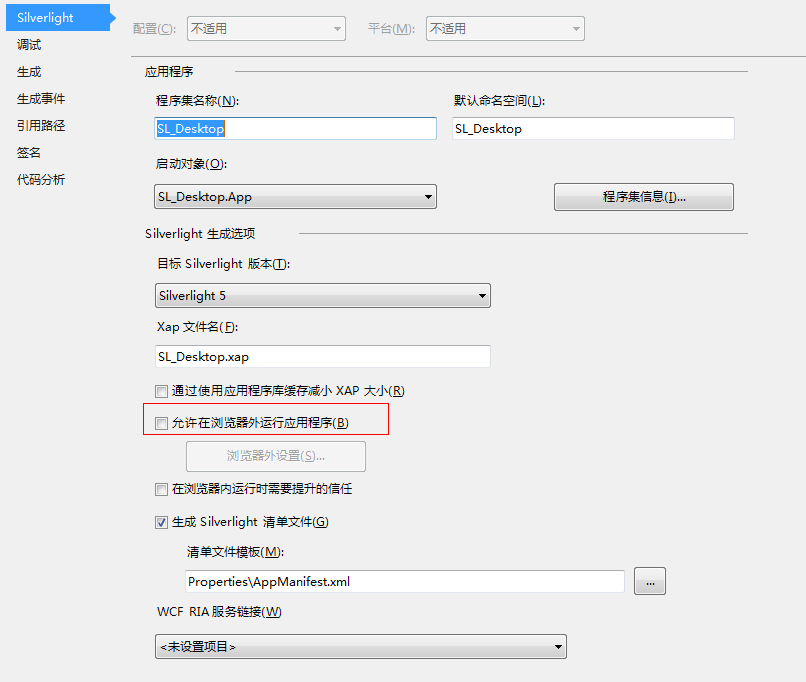
這裡我把方案和項目名稱起名為 SL\_Desktop 然後按確定出現下圖：



繼續確定

這裡我們看到了兩個項目SL\_Desktop 和 SL\_Desktop.Web 一個是 Silverlight的界面開發，另外一個承載的Silverlight顯示的程序。

第二步:我們就需要設置一下Silverlight 的程序，在SL\_Desktop 項目右鍵屬性在Silverlight 選項中看到”允許在游覽器外運行應用程序” 上打勾，如圖:



然後點擊 ”游覽器外設置” 按鈕，首先設置窗口你想顯示的標題名稱，設置默認啓動后寬度和高度你也可以設置啓動后的窗體位置默認居中

設置快捷鍵名稱還可以寫上應用程序的説明内容，設置快捷鍵的圖標，圖標一共有4种格式

重要的一部，UI的美觀度還有需要啓動 3D 效果就需要把使用GOD加速勾上。

“顯示安裝菜單”默認是勾上，這個是在程序的右鍵可以安裝程序到用戶機子上，可以按開發者需要來設置

爲了運行時的讓用戶的操作系統對程序的信任度提高我們把“在游覽器外運行時需要提升的信任”勾上

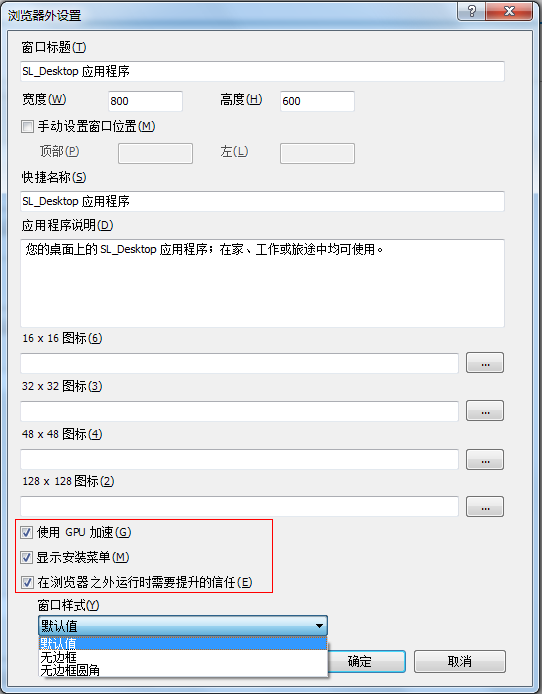
最後出現”窗口樣式” 選項

-默認值就像: Windows 的文件游覽器一樣有最小按鈕 最大化按鈕和關閉按鈕；

無邊框: 也就沒有最小按鈕 最大化按鈕和關閉按鈕和標題欄；

無邊圓角框: 在無邊框設置使用圓角外觀顯示，這個選項個人比較喜歡。

設置完成保存，設置界面如下圖：



爲了顯示Silverlight 程序的簡單在 MainPage.xaml 添加了一些 文字和按鈕如圖：

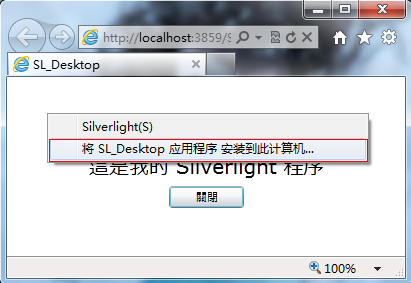


在按鈕添加點擊事件，用來關閉程序。只要調用 Application.Current.MainWindow.Close(); 就可以了

第三步：運行看到結果如圖：



奇怪怎麽還是在Web上面顯示呢？ 別急還有其他步驟的在 Silverlight 右鍵看到菜單中”將 SL\_Desktop 应用程序 安裝到此計算機” 點擊后，在Windows 的桌面和開始菜單的所有程序都會出現一個快捷方式”SL\_Desktop 应用程序”

如果已經安裝后怎麽卸載呢可以在程序，可以在運行該程序后右鍵點擊程序然後右鍵會出現”刪除該程序應用”就會自動卸載完。

就這麽簡單程序就可以在 Web 模式 或者 桌面模式下運行

**Silverlight 程序中判斷用戶是使用Web游覽器調用還是使用桌面模式調用**

第一步：在MainPage.xaml 界面中增加一個TextBlock 控件 代碼如下：

<TextBlock x:Name="Txt\_Msg" HorizontalAlignment="Center" Margin="0,60,0,0" TextWrapping="Wrap" VerticalAlignment="Center" FontSize="20" FontWeight="Bold"/>

第二部：在MainPage.xaml.cs 文件的默認構造方法中增加判斷后在上面的TextBlock顯示結果，判斷方法只要使用Application.Current.IsRunningOutOfBrowser 屬性就可以知道當前程序使用什麽模式顯示的。

代碼如下:

public MainPage()

{

InitializeComponent();

if (Application.Current.IsRunningOutOfBrowser == true)

{

//使用桌面模式

this.Txt\_Msg.Text = "当前调用的是使用桌面模式";

}

else

{

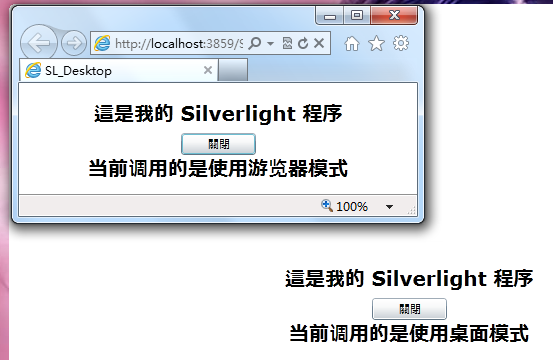
//使用的游览器

this.Txt\_Msg.Text = "当前调用的是使用游览器模式";

}

}

第三部:運行結果如下：



**Silverlight創建安裝桌面程序菜單**

在用戶的體現中鼠標右鍵在Web程序裏面是一種不友好的方式，所以最好還是在界面中顯示安裝按鈕，之前已將實現了判斷在游覽器中使用還是在游覽器中使用。這樣我們就可以顯示和隱藏該安裝按鈕。

第一步：檢查用戶是否有安裝該程序到桌面使用Application.Current.InstallState屬性就可以了；

第二步：安裝方法只要調用 Application.Current.Install() 就可以安裝到用的計算機上；

第三部：用戶需要知道安裝情況，可以註冊 Application.Current.InstallStateChanged 事件來監控安裝情況；

完整修改代碼如下

/// <summary>

/// Silverlight Demo

/// </summary>

public partial class MainPage : UserControl

{

public MainPage()

{

InitializeComponent();

this.Btn\_Install.Visibility = System.Windows.Visibility.Collapsed;

if (Application.Current.IsRunningOutOfBrowser == true)

{

//使用桌面模式

this.Txt\_Msg.Text = "当前调用的是使用桌面模式";

}

else

{

//使用的游览器

this.Txt\_Msg.Text = "当前调用的是使用游览器模式";

}

this.displayInstallState();

Application.Current.InstallStateChanged += Current\_InstallStateChanged;

}

/// <summary>

/// 检查安装状态

/// </summary>

private void displayInstallState()

{

switch (Application.Current.InstallState)

{

case InstallState.InstallFailed:

//不能将该应用程序安装为在浏览器外部运行

this.Txt\_Status.Text = "不能将该应用程序安装为在浏览器外部运行";

break;

case InstallState.Installed:

//已经将该应用程序安装为在浏览器外部运行

this.Txt\_Status.Text = "已经将该应用程序安装为在浏览器外部运行";

break;

case InstallState.Installing:

//正在将此应用程序安装为在浏览器外部运行

this.Txt\_Status.Text = "正在将此应用程序安装为在浏览器外部运行";

break;

case InstallState.NotInstalled:

//尚未将该应用程序安装为在浏览器外部运行

this.Txt\_Status.Text = "尚未将该应用程序安装为在浏览器外部运行";

this.Btn\_Install.Visibility = System.Windows.Visibility.Visible;

break;

default:

break;

}

}

//安装状态

void Current\_InstallStateChanged(object sender, EventArgs e)

{

this.displayInstallState();

}

//关闭按钮

private void Button\_Click\_1(object sender, RoutedEventArgs e)

{

Application.Current.MainWindow.Close();

}

//安装按钮

private void Btn\_Install\_Click(object sender, RoutedEventArgs e)

{

try

{

Application.Current.Install();

}

catch (InvalidOperationException)

{

MessageBox.Show("应用已经安装.");

}

catch (Exception)

{

this.Txt\_Status.Text = "安装失败";

}

}

}

**Silverlight桌面程序自動更新功能**

現在程序的安裝都完成了還差什麽呢？大家一定猜了就是更新問題，因爲這個也算半個桌面程序那麽如果原程序更改了，客戶端又不知道這樣導致程序用戶的不能正常使用程序了。總不能要用戶每次打開游覽器去做刪除安裝這個程序吧！現在我們就來解決這個問題。

所需要的方法 Application.Current.CheckAndDownloadUpdateAsync() 當然也有事件可以知道更新的狀態只要註冊Application.Current.CheckAndDownloadUpdateCompleted 事件就可以清楚知道程序更新情況了

在界面上添加一個更新按鈕註冊點擊事件内容事件Application.Current.CheckAndDownloadUpdateAsync()；

在程序的構造方法中註冊事件 Application.Current.CheckAndDownloadUpdateCompleted 事件中 e.UpdateAvailable 可以知道 更新是否成功；

不過這裡有很糾結的地方就是更新當然需要重啓程序才能算真正的完成可惜在Silverlight程序卻沒有沒有重啓程序的方法不過我想以後微軟會加入這些方法。

代碼如下

/// <summary>

/// Silverlight Demo

/// </summary>

public partial class MainPage : UserControl

{

public MainPage()

{

InitializeComponent();

this.Btn\_Install.Visibility = System.Windows.Visibility.Collapsed;

if (Application.Current.IsRunningOutOfBrowser == true)

{

//使用桌面模式

this.Txt\_Msg.Text = "当前调用的是使用桌面模式";

}

else

{

//使用的游览器

this.Txt\_Msg.Text = "当前调用的是使用游览器模式";

}

this.displayInstallState();

Application.Current.CheckAndDownloadUpdateCompleted += Current\_CheckAndDownloadUpdateCompleted;

Application.Current.InstallStateChanged += Current\_InstallStateChanged;

}

//更新事件

void Current\_CheckAndDownloadUpdateCompleted(object sender, CheckAndDownloadUpdateCompletedEventArgs e)

{

if (e.UpdateAvailable == true && e.Error == null)

{

MessageBox.Show("应用新版本已经下载成功，请重新启动程序。");

Application.Current.MainWindow.Close();

}

else if (e.UpdateAvailable == false && e.Error == null)

{

MessageBox.Show("已经是最新版本了。");

}

else if (e.Error != null)

{

MessageBox.Show("在检测应用更新时, 出现以下错误信息:" + Environment.NewLine + Environment.NewLine + e.Error.Message);

}

}

/// <summary>

/// 检查安装状态

/// </summary>

private void displayInstallState()

{

switch (Application.Current.InstallState)

{

case InstallState.InstallFailed:

//不能将该应用程序安装为在浏览器外部运行

this.Txt\_Status.Text = "不能将该应用程序安装为在浏览器外部运行";

break;

case InstallState.Installed:

//已经将该应用程序安装为在浏览器外部运行

this.Txt\_Status.Text = "已经将该应用程序安装为在浏览器外部运行";

break;

case InstallState.Installing:

//正在将此应用程序安装为在浏览器外部运行

this.Txt\_Status.Text = "正在将此应用程序安装为在浏览器外部运行";

break;

case InstallState.NotInstalled:

//尚未将该应用程序安装为在浏览器外部运行

this.Txt\_Status.Text = "尚未将该应用程序安装为在浏览器外部运行";

this.Btn\_Install.Visibility = System.Windows.Visibility.Visible;

break;

default:

break;

}

}

//更新

private void Button\_Click\_2(object sender, RoutedEventArgs e)

{

Application.Current.CheckAndDownloadUpdateAsync();

}

//安装状态

void Current\_InstallStateChanged(object sender, EventArgs e)

{

this.displayInstallState();

}

//关闭按钮

private void Button\_Click\_1(object sender, RoutedEventArgs e)

{

Application.Current.MainWindow.Close();

}

//安装按钮

private void Btn\_Install\_Click(object sender, RoutedEventArgs e)

{

try

{

Application.Current.Install();

}

catch (InvalidOperationException)

{

MessageBox.Show("应用已经安装.");

}

catch (Exception)

{

this.Txt\_Status.Text = "安装失败";

}

}

}

運行結果

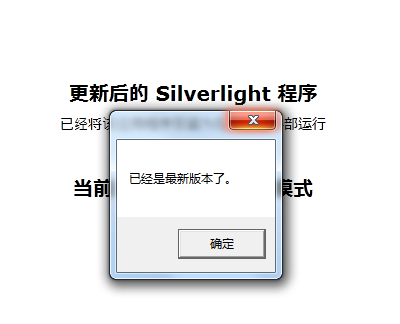




重新啓動程序



結果版本就一樣了，如果這是后再點擊更新就會出現下圖：



好了現在安裝到更新都有了我們還要為用戶想一下，如果用戶的電腦沒有連接網絡的情況或者使用中斷綫桌面模式獲取服務信息時候就會出現錯誤這個用戶就會覺得你的程序也太差了吧！

解決方法：就是讓程序去檢查是否有網絡存在；

我們可以使用 NetworkInterface.GetIsNetworkAvailable() 方法獲取是否在網絡連綫狀態；

當然我們也要註冊監控網絡的事件 NetworkChange.NetworkAddressChanged 來為程序作出更好的用戶體現。

實現代碼：

首先添加檢查網絡的方法

//檢查網絡狀態

private void checkNetworkStatus()

{

if (NetworkInterface.GetIsNetworkAvailable() == true)

{

this.Txt\_NetState.Text = "當前網絡處於連接狀態";

}

else

{

this.Txt\_NetState.Text = "當前網絡處於離綫狀態";

}

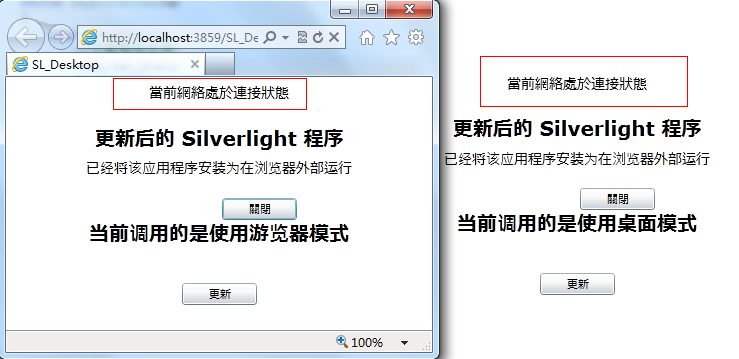
}

在構造方法中增加

this.checkNetworkStatus();

NetworkChange.NetworkAddressChanged += NetworkChange\_NetworkAddressChanged;

運行查看結果



本篇教程完畢！ 謝謝讀者！

大家可以去下載示例代碼。