Windows 8 开发 31 日

第 18 日

文件关联和程序合约

译者：BeyondVincent(破船)

时间：2013.4.25

版本： 2.0

关于破船

程序猿砌墙于云南昆明!

长期扎根移动软件开发!

爱跑步爱打篮球爱运动!

命中无大富大贵之面相!

愿健康与平淡相随一生!

你可以发邮件与破船取得联系: BeyondVincent@gmail.com

还可以关注破船的微博: 腾讯微博和新浪微博。

这里是破船的个人博客，欢迎光临：破船之家

关于 Windows 8 开发 31 日翻译

Windows 8 开发 31 日是由 Jeff Blankenburg 和 Clark Sell 原创的。

官方站点：http://31daysofwindows8.com/

涉及到两个版本：

XAML/C#（由 Jeff Blankenburg 撰写）

HTML5/JS （由 Clark Sell 撰写）

其中涉及到的资源和相关代码请到这里下载：

https://github.com/csell5/31DaysOfWindows8

在这里，由于破船对 HTML5/JS 不熟悉，所以只翻译 XAML/C#相关主题。

建议大家前往看原创内容，如果看不明白，再来这里看我翻译的相关内容。

如果翻译不正确的地方，可以通过上面的联系方式告诉破船。

破船祝你阅读愉快！

目录

第 18 日文件关联和程序合约

1.0. 介绍

今天，我将介绍 Windows 8 中非常酷的一对特征：文件关联和程序合约。我们来看看在 Windows 8 中与文件关联有关的两个特征：

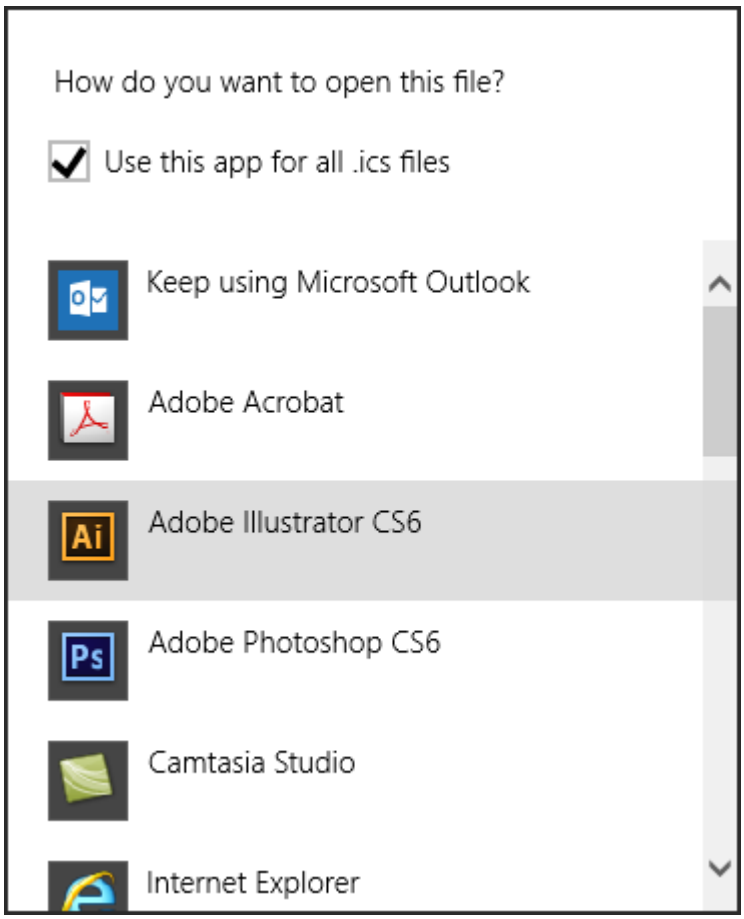
- 将我们的程序注册为这样一个程序：可以打开确定类型的文件，比如.png，或者自定义的扩展名，.31days。
- 当用户尝试打开一个我们的程序不支持的文件【比如.xls 文件】，那么可以让程序显示一个兼容的程序列表，供用户选择以打开。

最后，我也会介绍一些通过我们的程序与用户系统注册的一些方法，包括 AutoPlay。

1.1. 注册打开确定的文件类型

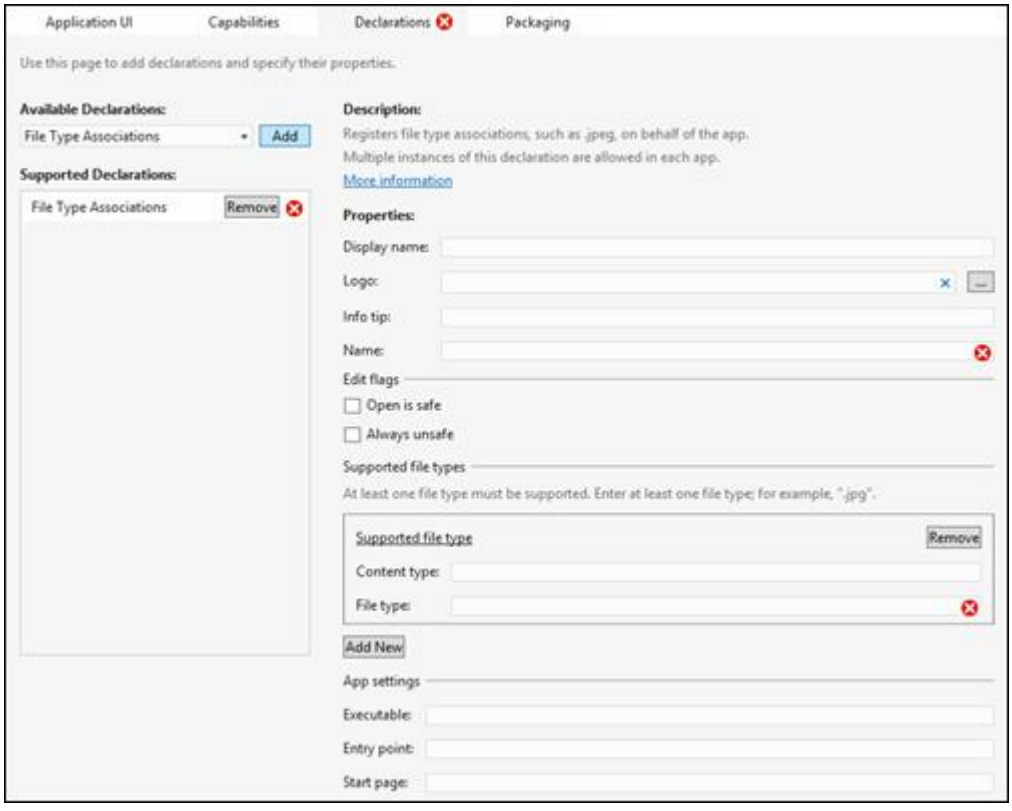
在这里，如何使用户在我们的程序中打开一个确定类型的文件，你可能想，我们可以创建一个图片编辑器，这样的话，用户所涉及到的各种图片格式，我们都应该能打开它。另外，我们也可以创建一个 XML 编辑器来处理 cosumes.xml 文

件。在这两种情况下，我们可能想要把程序与用户系统注册，那样，当用户想要打开某个文件时，会出现类似下面的提示：

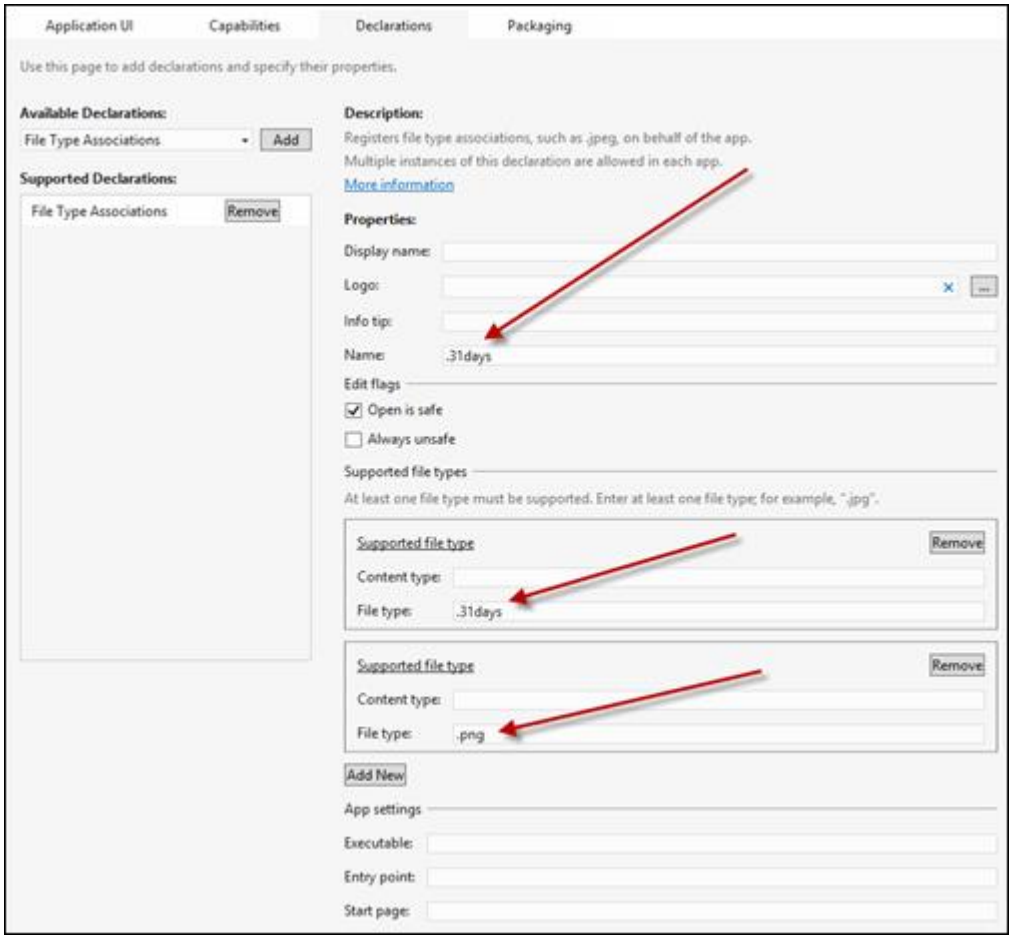


当程序打开的文件类型适合于我们的程序时，要想在上面的列表中看到我们的程序，操作起来很简单，只需要修改一下 package.appxmanifest 文件即可。打开 manifest 文件，选择 declarations 选项卡。在“Available Declarations”选择列表中，选择“File Type Associations”，然后点击 Add。

这是，你会看到一个新的表单显示在屏幕上，这个表单是与文件关联相关的。默认情况下，这里会有几个红色的 X，需要你来填写相关内容。

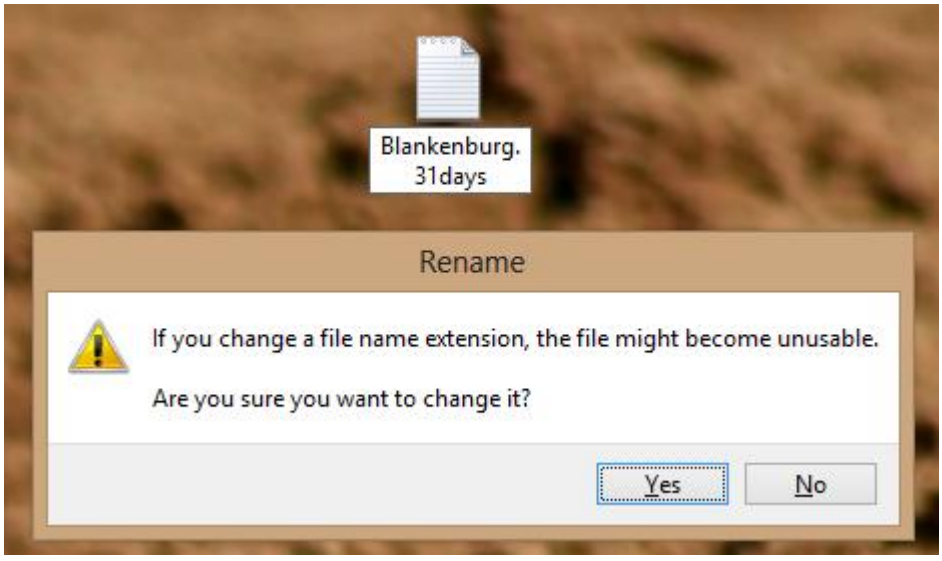


在这里，我选择两种不同的文件类型。第一种是我自定义的扩展名：“31days”，这种类型是由我创建的。另外注册的一种类型是“png”，这样，就可以用这个程序打开图片。

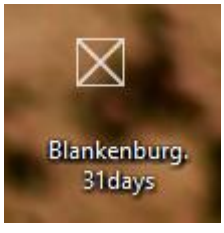


第一个值 Name，在这里只是一个名称。由于我们希望程序能够处理不同的文件类型，所以这里设置了 2 个文件类型。在表单的底部，可以指定打开这些文件的入口点。比如，如果是一个图片，你可能希望打开的是图片编辑页面，或者只是将其作为邮件的附件，这取决于你。

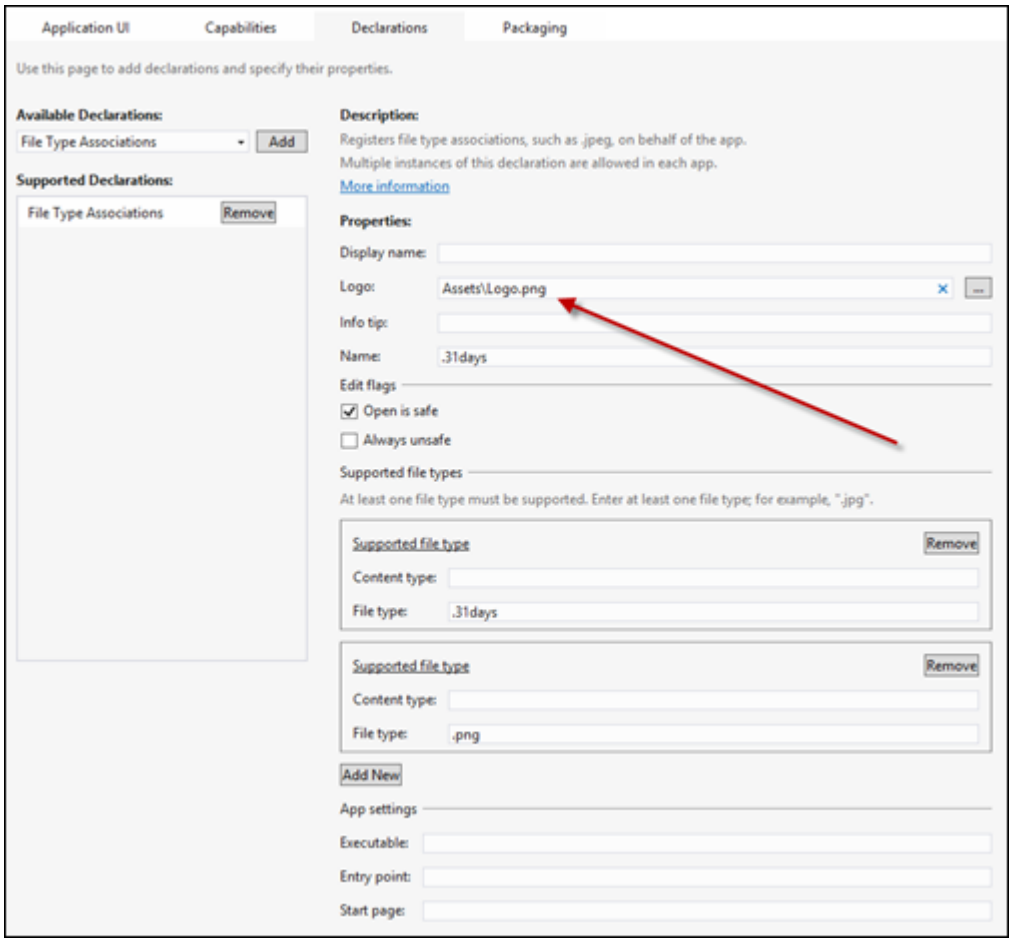
当你做完了上面的步骤，那么基本的工作就完成了。现在运行一次程序，将其安装到你的系统中，然后尝试使用“Open With”打开一个指定的类型。最简单的方法就是转到桌面，创建一个新的 Text 文件，然后将其名称修改为 Blankenburg.31days。会得到如下提示，选择 yes：



然后你会看到刚刚创建的文件的图片发生了变化，图片看起来是来自工程中的一个默认图片。现在唯一的问题就是我没有设置这样的一个图片，所以不确定这个图片来自什么地方。下面是桌面上显示的文件：

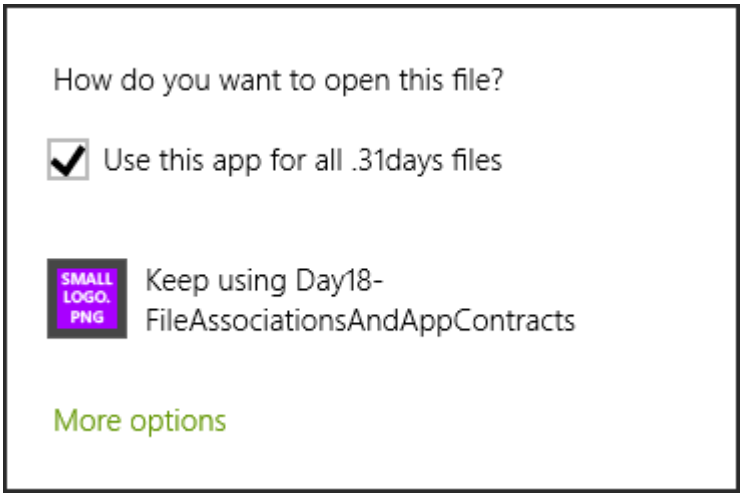


这看起来有点不好，当一个程序是某个文件类型的默认打开程序时，我想要显示自己的 logo 或者 icon。就像 Excel 有一个 Excel 图标，.31days 文件应该有一个 31 Days 图标。为了达到这个效果，打开 package.appxmanifest 文件，在指定 name 的地方，指定一个 logo，即可，如下：

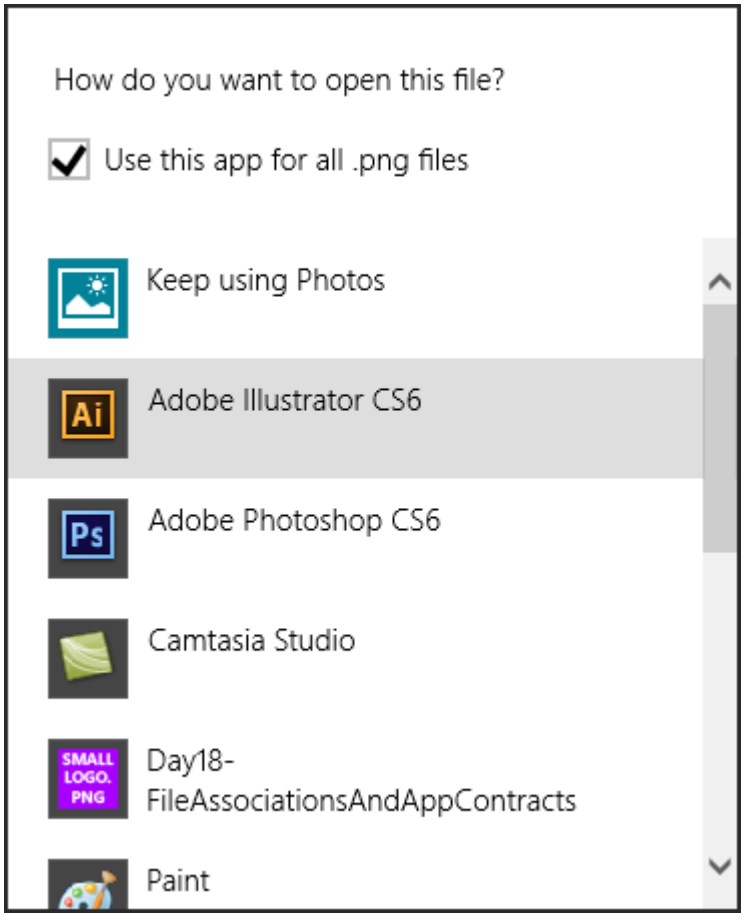


你可以使用你喜欢的任意图标，不过图标应该是个正方形。在这里，我选择 Logo.png。修改之后，文件看起来如下（记得运行一下程序，更新在系统中的安装内容）：

另外，当在该文件上单击右键，选择“Open With…”，会看到如下菜单：



你会发现不用任何修改，这里就已经使用了 SmallLogo.png 图片。最后，我们还为我们的程序注册了“.png”文件，下面是打开 png 时的截图：



Ok，现在我已经演示了如何在 Windows 8 中设置程序的文件关联。下一节中，我将介绍当从我们的程序打开一个文件到另外一个程序时，如何提示用户。

译者注：在程序的 App.xaml.cs 文件中，可以通过重写 Application 中的 OnFileActivated 方法，来捕获打开文件的事件，并做相应的处理，如下代码：

```
protectedoverridevoid OnFileActivated(FileActivatedEventArgs args)
{
// 在这里写与打开文件相关的代码
}
```

1.2. 在别的程序中打开一个文件

你的程序不可能打开每一种文件类型。例如，Microsoft Outlook 确实擅长于接收图片、文本和文件，但是如果你尝试打开邮件中的一个 Excel 文件时，文件将在 Excel 中打开，而不是 Outlook。在这里，我们也想在自己的程序中提供类似的功能给用户。为了达到这个目的，我们需要研究一下 Windows.System.Launcher 类。

在第一个示例中，我将在没有任何选项或设置的情况下打开我们的文件(Excel 文件来自第 17 日)。系统会使用默认的程序来打开我们的文件，而不会有任何的选择或警告。

```
privateasyncvoid OpenExcelButton_Click(object sender, RoutedEventArgs e)
{
Uri uri = newUri("ms-appx:///Assets/PeriodicTable.xls");
StorageFile sf = awaitStorageFile.GetFileFromApplicationUriAsync(uri);
awaitLauncher.LaunchFileAsync(sf);
}
```

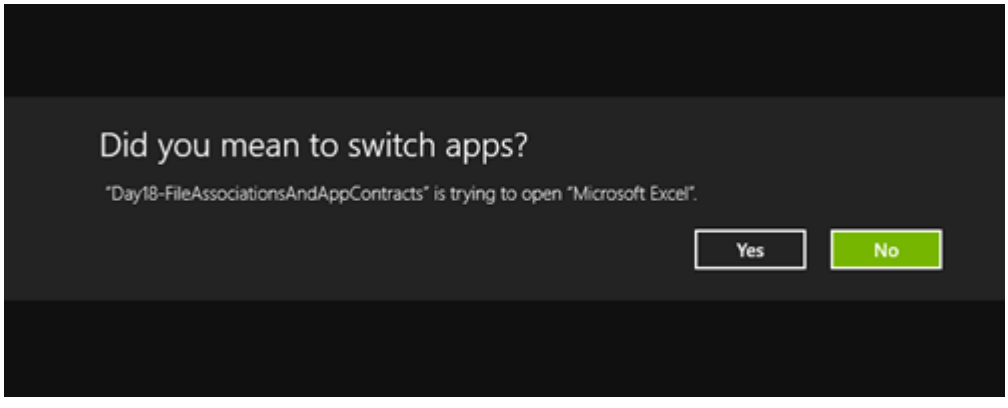
如上代码所示，非常简单。我调用了 Launcher.LaunchFileAsync()方法，并传递给它一个 StorageFile 对象。然而，有时候，当用户将要启动别的一个程序时，你想给用户一个提示：你将要启动另外一个程序，以让用户决定他是不是真的想要启动。这种情况的话，我们可以创建一个 LauncherOptions 对象，然后将其当做参数传递给 LaunchFileAsync()。如下：

```
privateasyncvoid OpenExcelButtonWithWarning_Click(object sender, RoutedEventArgs e)
{
Uri uri = newUri("ms-appx:///Assets/PeriodicTable.xls");
StorageFile sf = awaitStorageFile.GetFileFromApplicationUriAsync(uri);

LauncherOptions options = newLauncherOptions();
      options.TreatAsUntrusted = true;

bool action = awaitLauncher.LaunchFileAsync(sf, options);
}
```

如上代码运行时，在屏幕上会弹出一个警告，让用户知道另外一个程序将被启动。

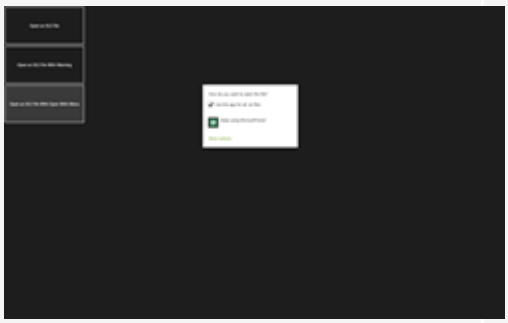


最后，有时候你希望用户可以自己选择程序来打开相关的文件。在这里，我想要弹出的窗口与上一节中用户选择“Open With…”弹出的相同。要实现这样的效果，我们只需要设置另外一个选项：DisplayApplicationPicker。如果你还想修改这

个弹出框显示的位置，那么还有一点工作要处理，不过做法与第 16 日的上下文菜单非常类似：需要确定弹出框显示在屏幕的什么位置。

如果你不指定一个具体位置的话，弹出对话框将会显示在屏幕中间。就我自己而言，我跟喜欢将这个弹出框显示在用户所选择的控件附近。下面是不同的截图（点击看大图）：

在屏幕中间显示对话框



在指定位置显示对话框

下面是相关代码：（如果你不想定位弹出框，）

```
privateasyncvoid OpenExcelButtonWithOpenWithMenu_Click(object sender, RoutedEventArgs e)
{
Uri uri = newUri("ms-appx:///Assets/PeriodicTable.xls");
StorageFile sf = awaitStorageFile.GetFileFromApplicationUriAsync(uri);

LauncherOptions options = newLauncherOptions();
        options.DisplayApplicationPicker = true;
        options.UI.InvocationPoint = GetPosition(sender asFrameworkElement);
        options.UI.PreferredPlacement = Placement.Below;

bool action = awaitLauncher.LaunchFileAsync(sf, options);
}
```

```
privatePoint GetPosition(FrameworkElement sender)
{
```

```
GeneralTransform transform = sender.TransformToVisual(null);
Point location = transform.TransformPoint(newPoint());
      location.Y = location.Y + sender.ActualHeight;

return location;
}
```

Ok，上面介绍了 3 中不同的方法来启动一个文件。在本文的最后一节，我将介绍程序的合约，以及介绍如何使我们的程序在用户的其它一些列表中可用。

1.3. 合理使用程序合约

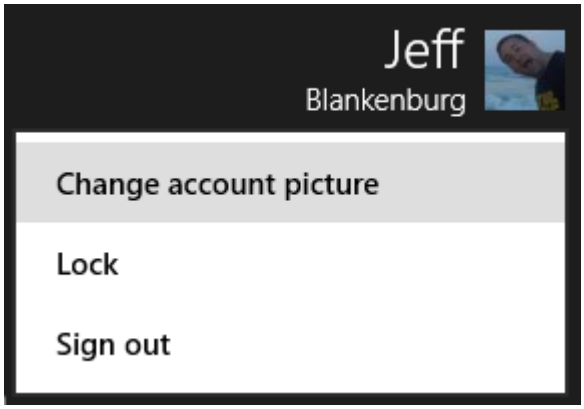
当在阅读本文时，有些开发者可能会认为将程序注册为可以打开每一种文件类型是一个好主意，因为这样可以经常保持程序和 logo 在用户面前。

请不要这样做。

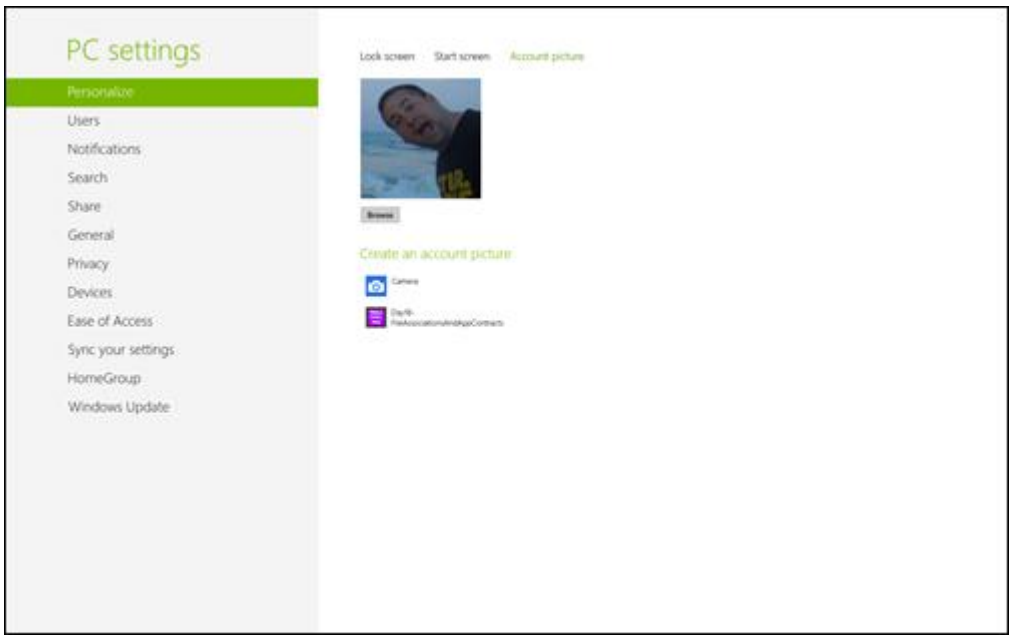
这个规则适用于程序合约。请不要滥用文件关联，特别是当你的程序实际上根本不能提供相关的功能时。当你提交程序到商店时，会专门检查这项。

1.3.1. Account Picture Provider

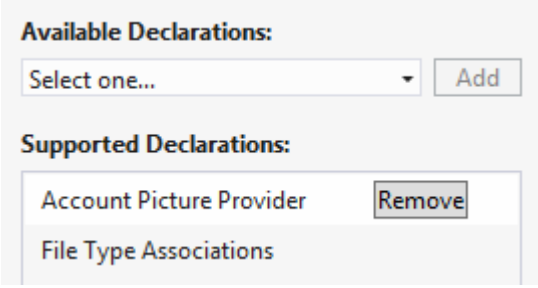
在 Windows 8 中，如果你的程序可以提供给用户创建一个 Account Picture，那么你可以声明这个功能。在开始屏幕，你可以看到这样的一个菜单：



当你注册了程序之后，可以在下图中，看到包含你的程序的一个列表：



要实现这样的一个效果，只需要将 Account Picture Provider 添加到 package.appxmanifest 文件的 declarations 选项卡中即可。如下：



默认情况下，这将启动你的程序，当然，你也可以指定自己的入口点。

1.3.2. AutoPlay 和 Protocols

这里有其它三种程序合约可以供我们订阅，并且每个合约都需要另外一个入口的声明。

“AutoPlay Content”：当有新的媒体插入到设备中，如 USB 或 DVD，“AutoPlay Content”可以识别出来。如果你的程序注册了该 content 类型，那么你的程序将出现在显示出来的 AutoPlay 列表中。关于 AutoPlay 在 MSDN 上有一篇非常好的文章，你可以使用不同的值。

“AutoPlay Device”基本上也是相同的概念，只不过是注册指定的设备类型，比如照相机、打印机或 USB 设备。

“Protocol”有点不同。它用户我们注册一个 URI，比如“mailto:”，甚至可以使用自定义的 URI，比如“31days:”。Protocol 的注册是另外一种方法，让我们的程序在适当的时候被使用。

1.4. 总结

今天，我介绍了一些扩展的点，可以使我们的程序更加的优秀。如果用户希望我们的程序能打开图片，那么，当用户尝试打开一个图片时，我们应该将我们的程序显示在程序列表中。

点击下图，可以下载本文的相关代码：

明天我将介绍 Windows 8 中另外一个非常强大的机制：文件选择器。这可以让用户选择他们设备上的文件，来提供给我们的程序使用。到时候见！

感谢你的阅读！

如果对这篇文章有什么想法想法，可以与破船联系，破船的联系方式在文章开头。

破船