

创建Block

千一网络 译

本文翻译自 <http://www.popfly.ms/>

因译者水平有限，加之时间仓促，书中难免有误，敬请批评指正。



2007-12-3

创建Block

概览

在本指南中你可以学到如何在Popfly中创建一个Block。推荐你使用不同于Popfly的开发环境创建你的Block，因为Popfly的JavaScript编辑器目前还缺少某些功能，比如语法高亮。（运行于 Popfly）。我们推荐 Microsoft Visual Web Developer 2005 Express，可以从<http://msdn.microsoft.com/vstudio/express/vwd/>免费下载。

一个Block是一个包含于单一JavaScript文件（.js）的中间件，它驱动供用户代码调用的方法，一个Block也可以用于资源存储，诸如XAML文件，图像等等。

一个Block可以作为外源服务，比如Web服务，或者它可以是一个简单的一个有用的程序库，比如：一个给定了半径计算圆面积的程序。一个Block也可以作为显示层：从其它的Block读取数据并以有意义的形式显示，用户可以与之交互。在本指南中我们将使用Soapbox (<http://soapbox.msn.com/>) 作为我们创建一个Block的Web服务示例。

JavaScript 类布局

一个用户可能用决定将许多Block用于一个网页中，因此将你的 JavaScript 封装成一个对象并且不要使用全局函数是很重要的，使用 Silverlight 的事件钩子抛出的异常。你可以按你的需要定义任意多个类，但是有一点你的对象需要的函数

规则 1: 创建类，而不是脚本

每一个Block都需要有一个类定义，成员函数添加在这个类原型上。比如：

```
function SoapBoxClass(){
    //一些代码
}
```

上面的代码声明了一个SoapBoxClass函数，该函数可以像下面一样通过修改原型来进行扩展。

函数

你想暴露给用户的函数应该在类原型中定义；通过在prototype中增加一个合适的匿名函数来创建一个新入口来实现这一点。

比如：

```
SoapBoxClass.prototype.search = function (<parameter>) {
    //一些代码
};
```

以下是一个在全局名称空间中声明函数的示例，在 Popfly 里这样做是一件糟糕的事情。这是因为你无法控制哪些函数已经存在，同样地你的Block的行为也是不可预知的。

```
function degtorad(deg){
    return deg * (2 * Math.PI) / 360;
}
```

和Popfly交互

因为Block是一个中间件，你应该在Popfly里面调用，并和用户交互。

从Web Server获取数据

许多控件都会从第三方获取数据，比如从一个网站。如果你从一个非Popfly域名的站点获取数据，浏览器安全沙盒将会提示用户权限或毫无提示地抛出一个异常。为了正常读取，Popfly提供两个方法来让控件从任何服务器上获取数据：`getXML`和`getText`。

getXML – XML数据

```
<xmlObject> = environment.GetXml(<url>)
```

<xmlObject>是HTTP响应读取和解析后的XML文档。

<url>是HTTP请求的URL，格式如：`http://blogs.msdn.com/coding4fun/rss.aspx?Tags=c4fnews`。

比如：

```
var url = "http://soapbox.msn.com/rss.aspx?searchTerm="+searchTerm;
var resultXML = environment.getXml(url);
```

getText – 文本数据

```
<text> = environment.GetText(<url>)
```

<text>是http响应的body。

<url>是HTTP请求的URL，格式如：`http://blogs.msdn.com/coding4fun/`。

输出结果

结果可以以两种方式输出：返回一个对象，或以HTML的形式添加到输出页面。因为Popfly就是从多个数据源获取数据并结合成“mashup”，所以我们更希望你写的所有函数都返回一个对象。显示信息的Block不适用本建议要求。

返回对象

如果你想返回一个对象，就需要类定义，除非返回的是JavaScript内置对象，比如一个数组。要为每一个Block声明公共属性，你可以使用`this`关键词动态地创建，就像下面说明的：

```
function SoapBoxVideo(title, link, pubDate, description, category, playURL,
streamingURL, copyright, thumbnailURL, credits, toStringHTML)
{
  this.Title = title;
  this.Link = link;
  this.DatePublished = pubDate;
  this.Description = description;
  this.Category = category;
  this.ThumbnailURL = thumbnailURL;
  this.PlayURL = playURL;
```

```
this.Copyright = copyright;
this.ThumbnailURL = thumbnailURL;
this.Credits = credits;
this.Copyright = copyright;
this.StreamingURL = streamingURL;
this.DescriptionHTML = toStringHTML;
}
```

然后，getFeaturedVideos方法返回一个SoapBoxVideo对象数组，如下：

```
SoapBoxVideo.prototype.getFeaturedVideos = function() {
    var soapBoxVideoArray = new Array();
    soapBoxVideoArray[0] = new SoapBoxVideo( "Foo Bar" );
    //一些代码
    return soapBoxVideoArray;
};
```

toString()

尽管许多对象不应该直接输出HTML，Popfly也希望把结果显示在浏览器上。你返回的所有对象应该实现一个toString函数，它将返回一个HTML字符串来展现对象。

比如：

```
SoapBoxVideo.prototype.toString = function(){
    var html = ""; // if undefined the concated string will be "undefined" +
    if(this.Description)
    {
        html += "<p>" + this.Description + "</p>";
    }

    return html;
};
```

异常&错误

一个通常的规则是：Block不能包含任何异常处理。较好的做法是告诉用户失败的信息并退出，而不是让Block继续运行。只有在极少数的情况下需要捕获异常。

如果你希望给出一条错误信息给用户，比如如果输入值无效，抛出一个字符串类型的异常，该字符串为你想要显示的信息。在所有可能的情况下避免使用alert()这样可恶的信息。

MetaData

除了JavaScript，每一个Block还需要一个XML文档来描述。这个XML文档用以描述其它Block如何与这个Block进行交互，哪些方法可以调用，这些方法有哪些参数。文档用以描述Block的相关内容：关键词，描述，一些Logo。在这个开发工具（本文档随一个开发工具同步发行。译者注）中包含了一个XML schema文件（BlockSchema.xsd）。它包含了合法的XML结构定义，用以验证你写的XML。

这个XSD读起来有一点痛苦，As the XSD can be a bit of a pain to read there is a simple sample below that covers the basics.需要注意的是所有的值和XML节点都是

大小写敏感的，因为确保写的时候要确保和你JavaScript中的大小写相同。

第一行是XML的头部，像下面一样进行标识：

```
<?xml version="1.0" encoding="utf-8"?>
```

Block节点

XML文档的最高级（根）元素是block节点，它拥有一个class属性。

```
<block class="SoapBoxClass">
```

class—你的Block的类的JavaScript函数名称。

block节点下面有四个包含文字内容的子节点，它四个子节点之下不再包含节点。

providerName节点

```
<providerName>SoapBox</providerName>
```

providerName应该是为本Block提供服务或公司的名称：它不应该是Block的作者。如果Block没有提供者，此时名称可以任意写（比如Block的作者）。

providerUrl节点

```
<providerUrl>http://soapbox.msn.com/</providerUrl>
```

providerUrl是提供者主页的绝对URL上面遵从相同的方针，就是说：如果你没有使用任何第三方数据，它应该链接到你的主页，但是如果你使用了第三方数据你就应该链接到他们的主页。

providerLogoUrl节点

```
<providerLogoUrl>/content/components/icons/soapBoxLogo.png</providerLogoUrl>
```

providerLogoUrl应该是提供者Logo的绝对URL。这将显示在我们需要给予对数据提供进行感谢的任何地方。如果你没有使用第三方数据，使用你的Logo链接。

blockIconUrl

```
<blockIconUrl>/content/components/icons/soapBox.png</blockIconUrl>
```

blockIconUrl应该是一个16x16大小的Logo的绝对URL。这些图标将显示在mashup编辑器的Block搜索菜单中。

block节点还有另外两个子节点：operations和objects。

operations节点

operations节点包含operation节点，该节点包含每一个你希望被用户或其它Block调用的函数；如果一个函数没有在此被包含它将不会显示在mashup编辑器中。你可以在你的Block逻辑中使用任意多的函数，但是要控件哪一个函数用以暴露给用户。警告：（在设计器中。译者注）用户不应该调用所有函数，除非是在编辑器中。

```
<operations>
```

该节点没有属性，并且唯一的应用是作为operation节点的父节点。

operation节点

你想通过设计器暴露给用户的每一个操作，都应该具有operation节点，该节点只有一个属性：name。

```
<operation name="search">
```

name—该节点描述的JavaScript函数的名称。

operation节点有三个子节点：description、inputs和outputs。

description节点

该节点值用在编辑器中以显示人类可读的关于该操作或函数的描述，以便用户在调用时知道什么是需要的。它应该以人类友好的方式来书写。

```
<description>
```

```
Search SoapBox to find videos that you want to watch.
```

```
</description>
```

inputs节点

inputs节点是每个input节点的父节点。每一个input应该直接映射到operation节点中描述的函数的参数上。input节点出现的顺序必须和JavaScript函数中参数的顺序相一致。

```
<inputs>
```

input节点

input节点有三个属性（name、required和type）和三个子节点：description、defaultValue和constraints。所有的属性和子节点都是必须的，但可以是空值。

```
<input name="searchTerm" required="true" type="string">
```

name—JavaScript参数名称。

required—参数是否是必须的，允许值为true和false。

type—期望值的类型（参见MetaData类型系统）。

description节点

description节点使用人类的方式来描述输入值。

```
<description>The term to search on.</description>
```

defaultValue节点

defaultValue节点应该包含一个可以让你的Block工作的值。当用户拖拽一个Block到设计界面时，编辑器将会把该默认值放在输入框中。这样用户可以预览你的Block并得到结果—用户将推测这个Block如何工作。

```
<defaultValue>comedy</defaultValue>
```

constraints节点

constraints节点的使用情况是：当你希望限制输入值时。比如，你可能希望限制用户只能输入州的名称。

outputs节点

outputs节点是output节点的父节点，子节点只能是一个output。该节点呈现operation节点描述的函数的返回值。

```
<outputs>
```

output节点

output节点描述函数的返回值类型。output节点具有三个属性，没有子节点。

```
<output isArray="true" type="custom" object="SoapBoxVideo" />
```

isArray—函数是否返回一个数组，如果是，则值为true，否则为false。

type—期望的值类型（参见MetaData类型系统），如果你使用的是custom类型，还应该具有相应的object节点，object节点的名称应该和这里的object属性值匹配。

object—如果你希望返回一个自定义对象，该值为自定义对象的名称。同时需要在object节点中描述自定义对象。

objects节点

block节点的最后一个子节点是objects，该节点是所有object节点的父节点。每一个object节点描述了一个你在JavaScript中使用的自定义类。这允许编辑器指出如何将block整合在一起，并提供给用户关于你使用的JavaScript类的更多信息。objects节点没有属性，并且只有名称为object的子节点。

object节点

object节点只有一个属性：name，具有名称为field的子节点用以表示JavaScript类的每一个属性。

```
<object name="SoapBoxVideo">
```

name—该object节点要呈现的JavaScript类的名称。

field节点

field节点呈现JavaScript类的属性。理论上，每一个属性都应该在此呈现，除非你想要隐藏起来不让用户看到。该节点具有三个属性没有子节点。

```
<field name="Title" type="title" isArray="false" />
```

name—该field节点要呈现的JavaScript属性的名称。

type—期望的值类型（参见MetaData类型系统），如果你使用类型的话。

isArray—该属性是否是数组，如果是，该属性值为true，否则为false。

MetaData类型系统

JavaScript是一个弱类型的语言，很少像你见过的C#或者Java语言一样需要类型。这给程序员巨大的处理弹性，但是这使得推断出给出的变量的潜在结构成了一件困难的事。Popfly想要帮助用户更容易地创建出mashup，这就需要给出的变量的结构以及使用的意图。比如，一个Block可能返回一个链接到图像的URL；在JavaScript中，这些URL应该是字符串。Popfly并不需要知道这些是URL，而需要知道这是指向图像的URL。这种情况下Popfly可以推荐用户使用一个图像显示Block来整合Block。这个工具中附加了一个类型系统的XML Schema定义（XSD），它包含了所有的有效类型。以下是其中的一部分，包含了描述。

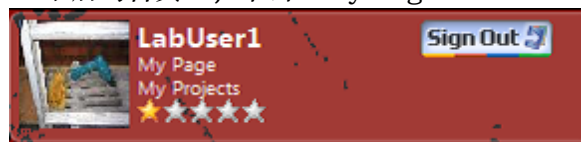
Name	Description
title	A short string that describes the object - should not contain

	HTML
url	Any URL that doesn' t fit one of the other URL types
color	A hex value color such as #FF0000
imageUrl	A URL pointing to a full sized image
feedUrl	A URL pointing to an RSS/ATOM feed
description	A description of the object - can contain HTML of any length
latitude	A latitude value in decimal format
longitude	A longitude value in decimal format
thumbnailUrl	A URL pointing to a thumbnail image
location	A string that represents an address
custom	A custom object that needs to be declared in the manifest file of that block
videoUrl	A URL pointing to a video file - should not be a flash video, but rather the actual video that could be played in any viewer
ipAddress	An IP address
name	The name of something - should be short and not contain HTML
firstName	The first (given) name of a person
lastName	The last (family) name of a person
emailAddress	An email address
phoneNumber	A phone number - should include country code
city	The name of a city
state	The name of a state, US only
countryOrRegion	The name of a country or region
zipCode	The ZIP code of a location, US only
ISBN	An International Standard Book Number (ISBN)
UPC	An Universal Product Code (barcode) number
stockSymbol	A stock ticker symbol

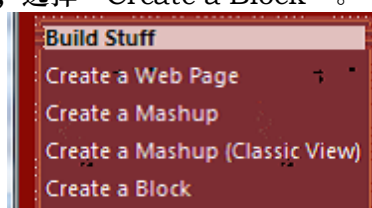
测试Block

要测试Block，你需要将其放在Popfly上，要这样做请参照下面的步骤。

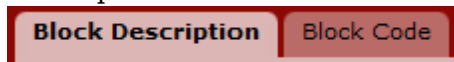
- 1) 在Popfly网站上，登录后的首页上，单击“My Page”。



- 2) 在左边的手动导航菜单上，选择“Create a Block”。

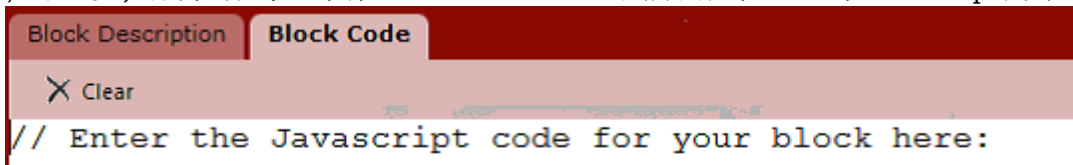


- 3) 在Block编辑器上, 你需要粘贴你的JavaScript代码和你的XML meta数据。第一个选中的标签是“Block Description”标签: 这里放是XML主体数据的地方。



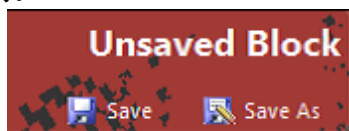
在文本框空白区域粘贴你为Block写的XML, 该XML应该和上面的MetaData一节的定义匹配。

- 4) 下一步, 你需要在第二个标签“Block Code”中粘贴你的Block的JavaScript代码。

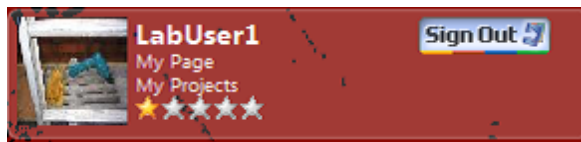


在文本框空白区域粘贴你的Block的JavaScript: 去除所有的调试代码, 一旦这个Block被共享这些代码便可能会被其它用户所执行, 此时出现这些调试代码是不合适的。

- 5) 现在, 你要做的是保存你的工程。不要忘了使用一个有意义的名字, 该名字在Block搜索菜单中会被所有用户看到。



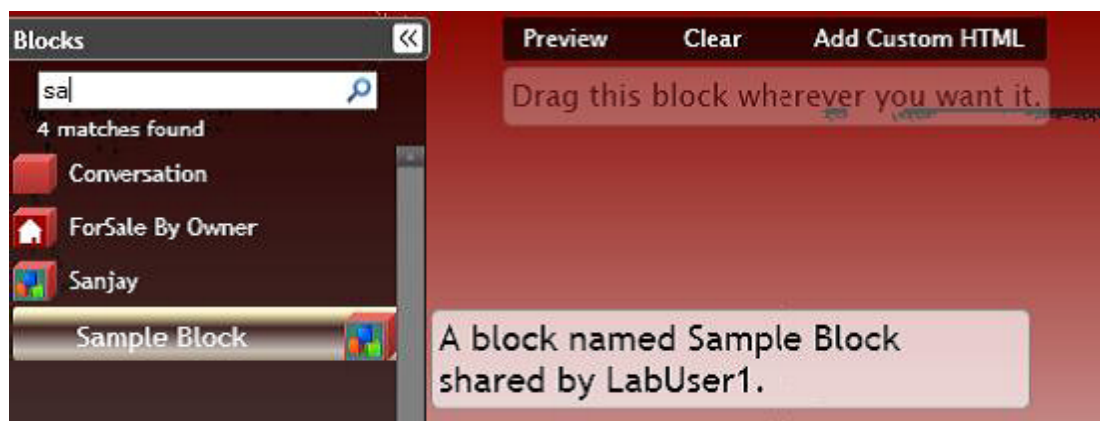
- 6) 为了让其它用户使用你的Block, 你需要共享它, 要这样做请跳转到你的“My Projects”页面。



- 7) 在你的“My Projects”页面上, 定位到你保存的Block, 单击“Share”链接。



- 8) 一旦共享, 你就可以在mashup编辑器的“Blocks”区域看到你共享的Block。



9) 就像运行其它Block一样，运行你的Block。

辅助方法

为了让开发生活更轻松一点，为了让跨浏览器支持更简洁一点，每一个Block将通过Microsoft AJAX Library来访问暴露的方法。关于此库的更多信息请参见：<http://ajax.asp.net/docs/ClientReference/default.aspx>，它包含JavaScript数组扩展和新添加的对象如string builder。

Popfly 扩展

除了Microsoft AJAX Library之外，Popfly为IE和FireFox实现了一系列方法。

SelectSingleNode(<element>)

选择在XML节点对象和文档中第一个匹配<element>的元素。

心得和提示

- 保持较少的函数。如果你的控件只做一件事倒好，如果它做十件不同的事，所有的都暴露给用户，那么我们需要谈谈。
- 你的代码将会被以明文的方式下载，所有的注释和源代码都会轻易被访问，如果你有任何不希望被拷贝的那么不要将其包含在里面。请遵守：保持清洁。
- JavaScript不支持纯种。耗时的操作将会让浏览器在一定时间内无法响应，如果程序不终止将会让浏览器“死掉”。
- 公用方法应该简单易懂，并且对垃圾输入有容错的功能。我们的目标用户可能不是专业的开发者也可能是专业的开发者，越简单越好，在这里对复杂的控件代码可是没有奖励的。
- 不要试图创建你自己的HTTP对象来连接到第三方Web服务，那是非常不明智的。
- 你看到的那些cookie不是你自己的，不要使用它们。你的Block将在一个沙盒环境中执行，最初对你的代码来说将会有cookie，但那里你不会访问到任何重要的cookie。即使你可以偷取它或者使它无效，也不要这么做，这不会威胁服务和用户，这只是错误的不好友的做法。
- 跨浏览器支持是你的朋友。我们喜欢Firefox (2.0+)，也希望我们的产品在上面运行得很好。不要使用IE独有的方法。话又说回来，一些方法太难以让人抵抗，比如selectSingleNode，因此我们在Firefox里也实现该方法。如果你使用getElementsByTagName也很好。相反也是同样道理，请避免使用IE 6+不支持的方法和对象。
- 你的Block代码必须是整个地没有压缩地存储在单一的.js文件中。资源可以被链接，你可以使用任意多的类，只要你想，但是只能有一个文件。
- 警报不是最好的调试器。如果你使用alert()来调试，你应该考虑使用Firebug，使用Venkman或Visual Studio来代替。关于这些工具的更多信息可以在“你可以随手找到的一些工具”一节找到。
- 在Firefox的XML节点元素中没有.text，使用.firstChild.nodeValue代替。
- 不要使用()来索引一个数组，使用[]代替，比如：

```
var someNumbers = new Array();
someNumbers(0) = 1; //不好, IE 中可以正常工作, 其它浏览器则不。
someNumbers[0] = 1; //很好
```

关于Firefox的说明

当我们说我们喜欢Firefox时，我们是真心的。我们为摆脱坏习惯付出了许多努力，但我们仍没有摆脱坏习惯，我们不想重蹈这些坏习惯，你应该真心希望让你的Block在Firefox 1.5+和IE 6+中都能正常工作，至少应该在FF 2.0和IE 7.0中能够正常工作。要实现这点，记住下面的……

- 我们已经在辅助方法中概述一些在两个浏览器中均实现了的方法。

你可以随手找到的一些工具

针对 Firefox

Firebug—随时带上它，不要想着不使用它而写一行JavaScript。

<http://www.getfirebug.com/>

<http://www.getfirebug.com/docs.html>

Firebug是网站开发者工具中的瑞士军刀。它可以用来对任何DOM元素进行观察和报警，你可以在不工作时发现变化。它也可以轻易截取所有从你浏览器接收或发出的网络流量，你可以看到来自第三方Web调用的数据是否是你希望的。它也拥有轻便的JavaScript错误控制台调试器^{††}，如果你希望中断JavaScript运行添加关键词debugger，Firebug将会在那一行中断。

网站 (<http://www.getfirebug.com/docs.html>) 上有一些关于如何最好地使用Firebug的指南和视频。只有一个警告：在Firebug中使用特定的关键词如debugger和console将会导致在其它浏览器中运行错误，因此在你完成之后请确认已经把这些关键词删除。

针对Internet Explorer

DevToolbar

<http://www.microsoft.com/downloads/details.aspx?FamilyID=e59c3964-672d-4511-bb3e-2d5e1db91038&displaylang=en>

在设计阶段针对DOM元素观察和报警比较有用，该工具也用在JavaScript Block创建时，尤其是具有可视界面的。

HTTPWatch

<http://www.httpwatch.com/>

HTTPWatch让你监视所有从你浏览器接收或发出的网络流量，你可以看到来自第三方Web调用的数据是否是你希望的。试用版的不幸是其内容预览器只能查看部分站点而不是所有站点。

Fiddler (请参见<http://www.cftea.com/c/2007/08/Y4QW6ZTWLPZPO5RE.asp>。译者注)

<http://www.fiddlertool.com/fiddler/>

如果你没有HTTPWatch的license，Fiddler是你截取和观察所有HTTP流量的杰出工具。你也可以使用NetMon，这是一款针对普通网络流量的调试工具。

对所有人

理论上，为了正常地测试你的Block，你需要一个网站，本地或远程的。当然，为每一个人设置一个站点是不现实的，因此，作为替代我们提供了一个运行于Visual Studio的测试工具，用它可以创建一个Web服务。如果你不愿意使用Visual Studio，希望使用你自己的编辑器，你仍然可以通过打开工程使用它来建立一个站点，然后运行它，然后利用它给出的本地地址浏览。你对文件所做的任何直接改动，都会直接反应出来，因为你不需要返回到工程中。

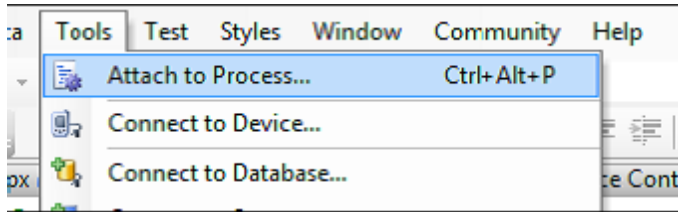
Visual Studio

对web开发者来说，这是强有力的工具，它在JavaScript方面有一些轻便的特性：语法高亮和调试。尽管如此，调试工作只在IE中进行，你还需要添加正确的IE进程以实现断点操作，如下：

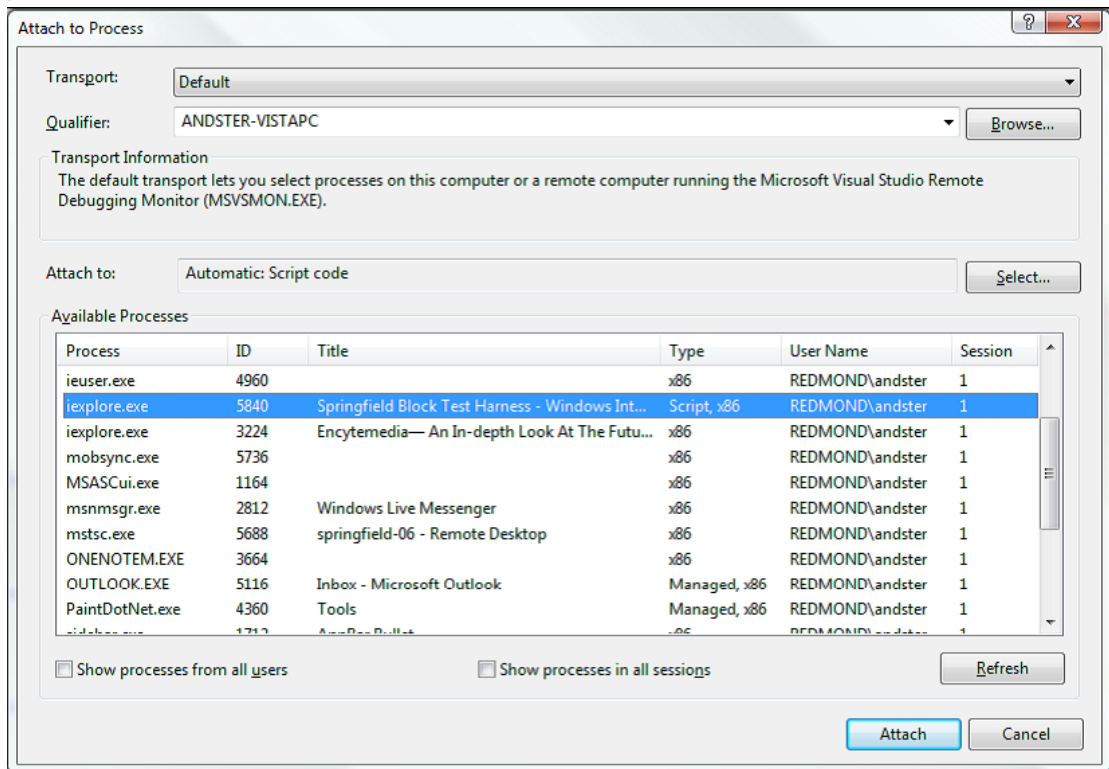
- 1) 在Visual Studio中给你的JavaScript插入断点是很普遍的，在希望设置断点的行

按F9或在左边距栏的地方单击左键。

- 2) 运行 (F5) 工程或从浏览器浏览网页，不过请注意，浏览一个本地文件暗示着受安全权限限制，至于限制哪些能做取决于你的浏览器。
- 3) 然后到Tools->Attach to Process



- 4) 标明IE进程正在处理你的JavaScript，就是指当前正在展示你的测试网页的进程，然后Attach



- 5) 现在，你可以运行你的脚本并处理断点，IE将会停止，你可以切回到Visual Studio中使用调试窗口，你也可以这样处理其它语言。