A  使用alert() 和document.write() 方法监视变量值

如果要中断代码的运行，监视变量的值，则使用alert() 方法；

如果需要查看的值很多，则使用document.write() 方法，避免反复单击“确定”按钮；

B  使用window.onerror 事件

当页面出现异常时，onerror 事件会在window 对象上触发。它能在一定程度上告诉开发者相关的错误信息。

示例：

|  |
| --- |
| <script type="text/javascript"> |
| **function** myerror(\_message,\_url,\_line) |
| { |
| alert("错误信息：" + \_message |
| +"\n错误的URI：" + \_url |
| +"\n错误的行数：" + \_line |
| ); |
|  |
| **return** **true**; //屏蔽系统的事件 |
| } |
| //绑定错误事件 |
| window.onerror = myerror; |
|  |
| //触发错误示例： |
| window,onload = test; |
| </script> |

**注意：**在IE 中，触发error 事件后，正常的代码会继续运行，所有的变量和数据都会保存下来，在其onerror 事件处理方法中可以正常访问到；而在Firefox 中，触发error 事件后，一切都结束，所有的变量和数据都将被销毁。

C  使用 try...catch 语句找错误

示例：

|  |
| --- |
| <script type="text/javascript"> |
| **try** |
| { |
| alert(触发异常); |
| } |
| **catch** (\_ex) //可以省略“\_ex”参数 |
| { |
| **var** err = "错误信息"; |
| **for** (**var** i **in** \_ex) |
| { |
| err += "\n参数名：" + i |
| + "\t参数值：" + \_ex[i]; |
| } |
| alert(err); //打印错误 |
| } |
| **finally**  //finally 可以被省略... |
| { |
| alert("finally 总是会运行"); |
| } |
| </script> |

**注意：**try...catch 并不能很好的处理JavaScript 的语法错误。

示例：

|  |
| --- |
| <script type="text/javascript"> |
| **try** |
| { |
| alert("触发语法错误")); //多了半边“)” |
| } |
| **catch** (\_ex) //可以省略\_ex参数 |
| { |
| **var** err = "错误信息"; |
| **for** (**var** i **in** \_ex) |
| { |
| err += "\n参数名：" + i |
| + "\t参数值：" + \_ex[i]; |
| } |
| alert(err); //打印错误 |
| } |
| </script> |

该示例并没有进入catch 块中。

D  使用相关调试器

**在IE 和Firefox 浏览器中，可以使用相关的调试器或插件对JavaScript 进行调试。**

●  **在Firefox 浏览器中，可以使用其自带的“错误控制台”。**操作步骤如下：

      打开Firefox 浏览器 → 在菜单条“工具”中 → 选择“错误控制台”即可。

     在没有其他插件的情况下，其自带的“错误控制台”是一个非常不错的选择。

另外，在Firefox 浏览器中，还有一些很不错的调试器，如：Venkman、Firebug 等。

         Venkman 调试器安装后，可以在Firefox 浏览器 → 在菜单条“工具”中 → 选择“JavaScript  Debugger ”命令启用；

         Firebug 调试器安装后，可以在Firefox 浏览器 → 在菜单条“工具”中 → 选择“Firebug”→ 选择“打开 Firebug”即可；

●  **在IE 浏览器中，可以使用 Microsoft  Script  Debugger 调试器**

       Microsoft  Script  Debugger 是微软随IE 4 一同发布的一个IE插件，可以从微软的官方网站上免费下载。

       下载安装以后，必须将IE 浏览器的调试选项打开才能使用。操作步骤如下：

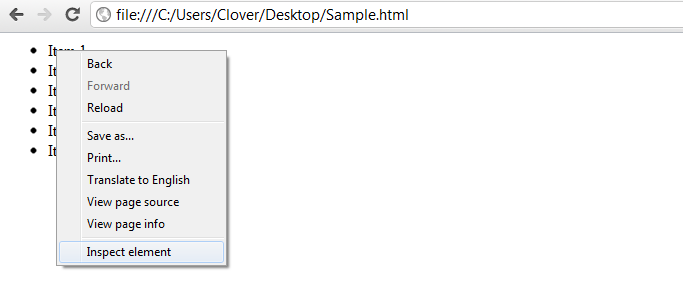
       1>   打开IE 浏览器 → 选择菜单栏的“工具”→ “Internet 选项”命令 → “高级”选项卡 → 将“禁用脚本调试（Internet Explorer ）”复选框中的勾去掉即可。

       2>   当IE 浏览器正在浏览页面时，运行Microsoft  Script  Debugger 调试器工具即可进行调试。

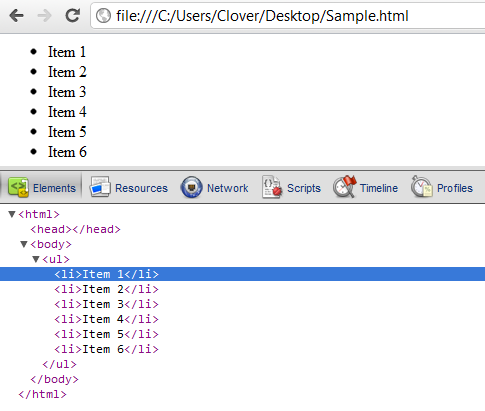
       在Microsoft  Script  Debugger 调试器的 Running  Document 面板中选择开启的页面文件（只读），然后按F9 可以设置断点调试。另外，其Command  Window 面板也是一个很有用的功能，它能在代码断点停止时，在其中输入变量名并回车，便可看到此时变量的值；Command  Window 面板甚至可以接受简单的JavaScript 命令。但Microsoft  Script  Debugger 调试器自身还存在一个bug 问题。

1.HTML的调试

* 将鼠标放在任意元素上，右键Inspect Element，即可查看该元素的HTML内容。（也可以通过F12打开开发者工具）



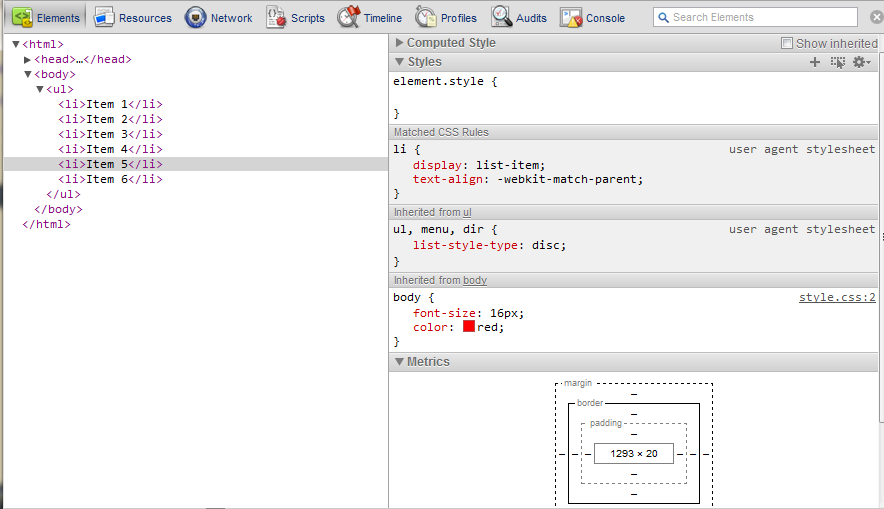
　　效果如下



* 在Elements面板中 将鼠标放在任意元素上，HTML界面上对应的元素会被选中
* 右击元素，可以直接修改或者是移除元素，修改后的元素效果会马上反应在页面上

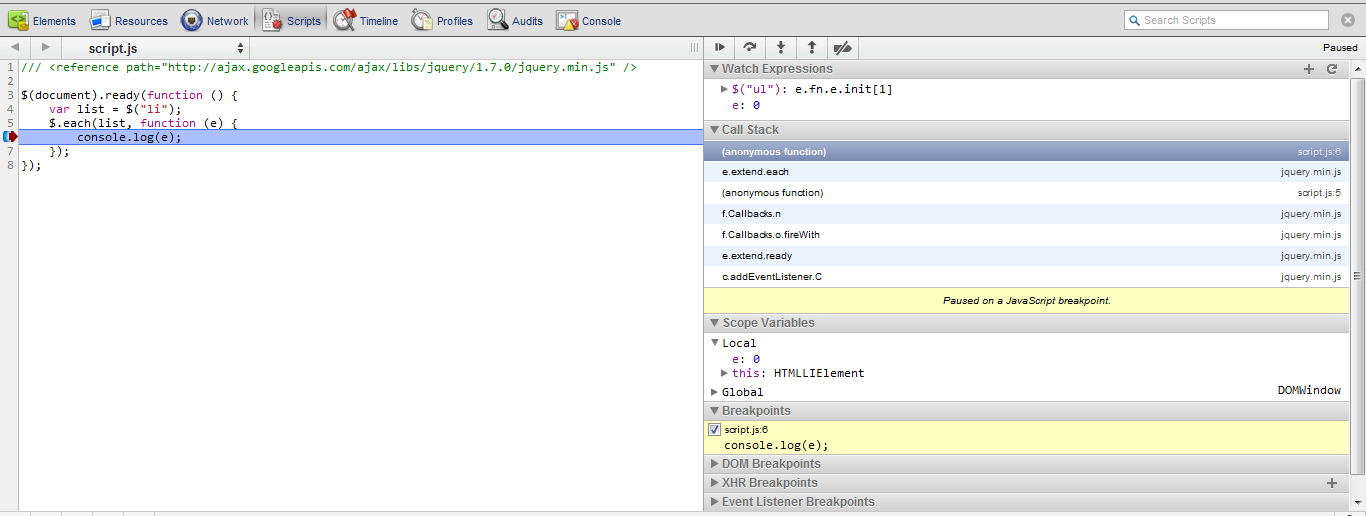
2.CSS的调试

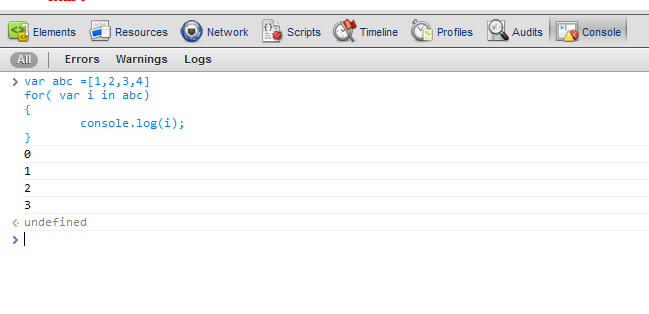
* 在开发者工具的右侧面板可以找到对应的CSS调试框



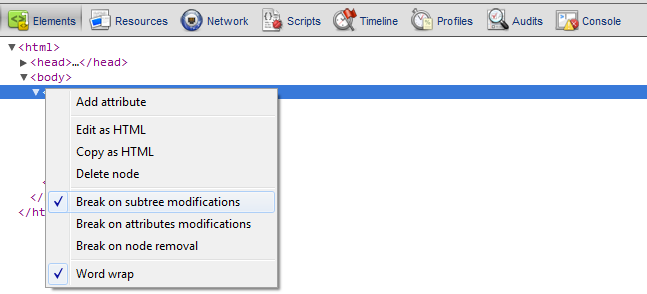
* Styles分类下可以看到该元素的详细样式，而且你还可以直接修改它 （点击一个样式，或者双击空白处）
* 样式的右侧会提示该样式来自哪个文件的哪一行
* 被加了删除线的样式是无效的，或者被覆盖了
* 将鼠标放在样式上的时候，右侧会出现一个CheckBox，用于启用和禁用当前样式

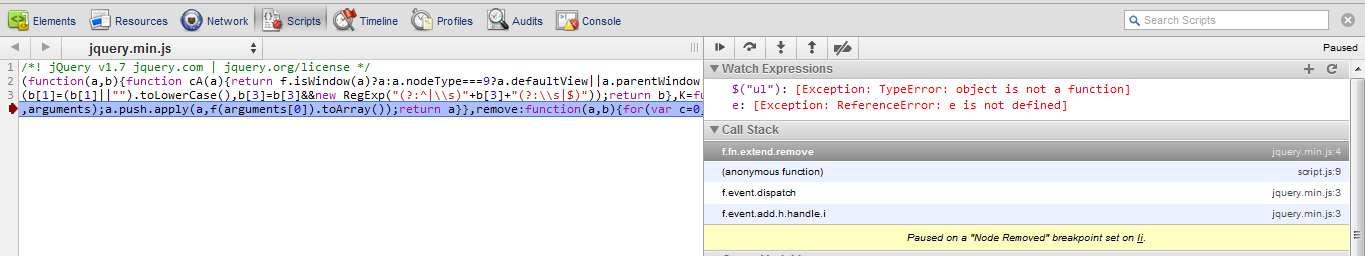
3.Javascript调试

* 首先是Javascript调试框　　
* 左侧是javascript面板，可以加断点（绿色），当前正运行到的行（红色）
* 位于script.js的下拉框可以选择不同的js文件调试，（如果你的js文件有结构错误，可能没法被加载到列表中）
* 右上方是javascript内容搜索
* Watch Expressions可以运行和查看表达式和变量
* Call Stack显示当前堆栈，单击可以切换到堆栈中的不同位置
* Scope variables，显示当前上下文变量
* Breakpoints， 断点
* console.log该方法会输出到Console中，（放弃alert吧，用调试和console.log）
* 在Console可以直接运行javascript，很方便 (如果你要验证js，不需要写个新页面，直接在这里运行就是了)



* 现在可以很方便的调试Javascript了，不过有的时候，你发现页面上元素被修改了，但是你不知道被什么代码修改了，那可以给DOM增加断点



* 当ul中的元素被改变的时候触发断点，如下：（注意右边的堆栈，那可以帮你找到答案：谁动了你的奶酪）　　

还有很多高级功能，例如资源和网络分析，javascript性能分析（内存泄漏）等等，有兴趣研究下