# [Zen Coding: 一种快速编写HTML/CSS代码的方法](http://www.qianduan.net/zen-coding-a-new-way-to-write-html-code.html)

在本文中我们将展示一种新的使用仿CSS选择器的语法来快速开发HTML和CSS的方法。它由[Sergey Chikuyonok](http://www.smashingmagazine.com/author/sergey-chikuyonok/)开发。

你在写HTML代码(包括所有标签、属性、引用、大括号等)上花费多少时间？如果你的编辑器有代码提示功能，你编写的时候就会容易些，但即便如此你还是要手动敲入很多代码。

在JavaScript方面，当我们想要在一个页面上获取某个特定的元素时，我们就会遇到同样的问题，我们必须写很多代码，这就变得难于维护和重用。JavaScript框架应运而生，它们同时引入了CSS选择器引擎。现在，你可以使用简单的CSS表达式来获取DOM元素，这相当酷。

但是，如果你不仅仅可以用CSS的选择器布局和定位元素，还能**生成代码**会怎么样？比如，如果你这样写：

|  |
| --- |
| div#content>h1+p |

…然后就可以看到这样的输出：

|  |
| --- |
| <**div** id="content">  <**h1**></**h1**>  <**p**></**p**>  </**div**> |

有些迷惑吧？今天，我将向你介绍[Zen Coding](http://code.google.com/p/zen-coding/)，一组用于快速HTML和CSS编码的工具。最初由[Vadim Makeev在2009年4月提出](http://pepelsbey.net/2009/04/zen-coding-concept/)(文章为俄语)，由鄙人(也就是我)开发了数月并最终达到比较成熟的状态。Zen Coding由两个核心组件组成：一个缩写扩展器(缩写为像CSS一样的选择器)和上下文无关的HTML标签对匹配器。看一下这个演示视频来看一下它们能为你做些什么。



**注意：**该视频原版位于Vimeo，但是要看的话需要翻[和谐]墙先，地址在这里：<http://vimeo.com/7405114>，上面的视频是我费尽周折从Vimeo上下载下来上传到优酷的，上传后质量竟被大打折扣了，囧。youtube上也有一份视频，是基于Aptana的演示，一样很精彩：<http://www.youtube.com/watch?v=ug84Ypwqfzk>。PS:貌似youtube要比Vimeo翻[和谐]墙容易些，不过如何翻[和谐]墙不在本站讨论范围。

如果你想跳转到详细介绍和使用指南，请看一下演示页面并立刻下载你适用的插件：

**Demo**

* [Demo](http://zen-coding.ru/demo/) (使用 Ctrl + , 展开缩写，需要JavaScript支持)
* [中文版演示](http://labs.qianduan.net/zencoding/)

**下载(完全支持)**

* [Aptana](http://code.google.com/p/zen-coding/downloads/detail?name=Zen.Coding-Aptana.v0.5.zip) (跨平台);
* Coda, via [TEA for Coda](http://github.com/sergeche/tea-for-coda/downloads) (Mac);
* Espresso, via [TEA for Espresso](http://github.com/sergeche/tea-for-espresso/downloads) (Mac);

**下载(部分支持，只支持“展开缩写”)**

* [TextMate](http://code.google.com/p/zen-coding/downloads/detail?name=Zen%20Coding%20for%20TextMate%20v0.3.1.zip) (只能用于Mac机,Windows可以使用E-text编辑器替代);
* [TopStyle](http://code.google.com/p/zen-coding/downloads/detail?name=TopStyle.Zen.Coding.1.0.zip);
* [Sublime Text](http://code.google.com/p/zen-coding/downloads/detail?name=Sublime.Zen.Coding.1.1.3.zip);
* [GEdit](http://www.kryogenix.org/days/2009/09/21/zen-coding-for-gedit);
* [Dreamweaver CS4](http://zen-coding.googlecode.com/files/Zen.Coding-Dreamweaver.v0.4.zip)
* [editArea在线编辑器](http://zen-coding.ru/demo/);
* [Zen Coding在线编辑器中文版](http://labs.qianduan.net/zencoding/)

现在让我们看一下这些工具是如何工作的吧。

**展开缩写**

展开缩写功能将类似CSS的选择器转换为XHTML代码。术语“缩写”可能会有点儿难以理解。为什么不直接称之为“CSS选择器”呢？嗯，首要原因是语义化：“选择器”意为选择一些东西，但是在这里我们事实上是生成 一些东西，**是写一个长代码的较短的替代**。其次，它只是使用真实的CSS选择器语法的一个小的子集，并添加了一些新的操作符。

这里是一个支持的属性和操作符的列表：

* E  
  元素名称(div, p);
* E#id  
  使用id的元素(div#content, p#intro, span#error);
* E.class  
  使用类的元素(div.header, p.error.critial). 你也可以联合使用class和idID: div#content.column.width;
* E>N  
  子代元素(div>p, div#footer>p>span);
* E+N  
  兄弟元素(h1+p, div#header+div#content+div#footer);
* E\*N  
  元素倍增(ul#nav>li\*5>a);
* E$\*N  
  条目编号 (ul#nav>li.item-$\*5);

正如你能看到的，你已经知道如何使用Zen Coding了：只是些一个简单的仿CSS选择器(呃，“缩写”抱歉)，就像这样…

|  |
| --- |
| div#header>img.logo+ul#nav>li\*4>a |

…然后调用”展开缩写”行为。

这里有两个新增的操作符：元素倍增和条目编号。比如，如果你想生成5个<li>元素，你可以简单的写位li\*5。它也将同样重写全部子代元素。如果你想写4个<li>元素，每个里面都有一个<a>标签，你就可以简单的写为li\*4>a，这样会生成以下HTML代码：

|  |  |
| --- | --- |
| 1  2  3  4 | <**li**><**a** href=""></**a**></**li**>  <**li**><**a** href=""></**a**></**li**>  <**li**><**a** href=""></**a**></**li**>  <**li**><**a** href=""></**a**></**li**> |

最后一个——条目编号用于当你想用索引标记重复的元素的情况。假设你想生成class为item1、item2和item3的3个<div>元素。你可以写成这样的缩写，div.item$\*3:

|  |  |
| --- | --- |
| 1  2  3 | <**div** class="item1"></**div**>  <**div** class="item2"></**div**>  <**div** class="item3"></**div**> |

只需在你想要索引出现的任何class或id属性上添加一个美元符号即可，而且想要多少都可以。那么，这样…

|  |
| --- |
| div#i$-test.class$$$\*5 |

会被转换成为:

|  |  |
| --- | --- |
| 1  2  3  4  5 | <**div** id="i1-test" class="class111"></**div**>  <**div** id="i2-test" class="class222"></**div**>  <**div** id="i3-test" class="class333"></**div**>  <**div** id="i4-test" class="class444"></**div**>  <**div** id="i5-test" class="class555"></**div**> |

你会看到，当你写a的缩写的时候，输出是<a href=”"></a>。或者，如果你写img，输出就是<img src=”" alt=”" />。

Zen Coding是如何知道什么时候应该为生成的标签添加默认的属性或者跳过关闭标签的？有一个专门的文件，名为[zen\_settings.js](http://code.google.com/p/zen-coding/source/browse/branches/serge.che/aptana/zen_settings.js)描述了输出元素。这是一个简单的JSON文件，描述每种语言的缩写(是的，你可以为不同的句法定义缩写，比如HTML、XSL、CSS等)。通用的语言缩写定义看起来就像这样：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | 'html': { 'snippets': { 'cc:ie6': '<!--[if lte IE 6]>**\n\t**${child}|**\n**<![endif]-->',  ...  },    'abbreviations': { 'a': '<a href=""></a>', 'img': '<img src="" alt="" />',  ...  }  } |

**元素类型**

Zen Coding有两个主要的元素类型：**“片段(snippets)” 和 “缩写(abbreviations)”。**片段就是随意的代码碎片，而缩写是标签定义。通过片段，你可以写出你想要的任何代码，它也会照你写的格式输出；但是你必须手动的格式化它(使用\n 和\t实现换行和缩进) 并将${child}变量放到你想要输出子元素的地方，就像这样：cc:ie6>style。如果你不使用${child}变量，子元素将会输出于代码片段的**后面**。

有了缩写，您必须编写标记定义，而且语法是非常重要的。通常，你必须写一个简单的带有所有默认的属性的标签，比如: <a href=”"></a>。当Zen Coding被加载后，它会解析一个标签定义到一个描述该标签的名字、属性(包括它们的顺序)以及该标签是否为空的特定的对象中。所以，如果你写<img src=”" alt=”" />，你会告诉Zen Coding这个标签必须是空的，然后“扩展缩写”行为就会在输出之前为它使用特定的规则。

对于片段和缩写，你可以添加一个管道符号，它告诉Zen Coding当缩写被展开的时候光标会被定位到哪里。默认的，Zen Coding 将光标放在空属性的引号中间以及开始和关闭标签的中间。

**例子**

那么，这里解释一下当你写了一个缩写并召唤“展开缩写”行动时发生的事情。首先，它将一个完整的缩写分开为独立的元素：这样div>a 会被分成div 和a 元素，当然也会维持他们的关系。然后，每个元素，解析器先在代码片段内而后在缩写中寻找定义。如果它找不到，将会使用元素的名字作为新的标签，并为其添加缩写中定义的id和class。比如，如果你写mytag#example，解析器在片段或缩写中找不到mytag定义，它就会输出<mytag id=”example”><mytag>。

我们制作了很多[默认的CSS](http://code.google.com/p/zen-coding/wiki/ZenCSSPropertiesEn)和[HTML缩写和片段](http://code.google.com/p/zen-coding/wiki/ZenHTMLElementsEn)。你会发现学习使用Zen Coding可以增加你的生产力。

**HTML 标签对匹配器**

对于HTML编码者的另一个非常常见的任务是寻找一个元素的标签对。例如你想选择整个<div id=”content”>标签并将其移动到其它地方或者删除它。或者有可能你在寻找一个关闭标签并想知道它属于那个开始标签。

不幸的是，很多现代开发工具在该功能方面有所欠缺。那么我就决定写一个我自己的标签对匹配器作为Zen Coding的一部分。不过它依然在beta阶段并尚存一些问题，但它可以工作的很不错并很快。不是浏览整个文档(像通常的那种HTML标签对匹配器的做法)，它从光标的当前位置开始寻找相关的标签。这使得它非常快并且上下文无关：它甚至可以用于这段**JavaScript代码片段**：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | **var** table = '<table>'; **for** (**var** i = 0; i < 3; i++) {  table += '<tr>'; **for** (**var** j = 0; j < 5; j++) {  table += '<td>' + j + '</td>';  }  table += '</tr>';  }  table += '</table>'; |

**使用缩写包裹**

这真的是一个很酷的特性，它将缩写和标签对匹配器的功能合并到一起了。你有多少才发现你需要**添加一个包裹元素以修正一个浏览器bug**？或者你需要添加一个装饰，比如一个背景图片或者边框到一个块级内容？你必须写开始标签，临时打断你的代码，找到相关的点然后关闭标签。这就是“使用缩写包裹”能帮助你的地方。

该功能相当简单：它要求你输入缩写，然后执行适当的“展开缩写”行动并将你期望的文本放到你缩写的最后一个元素里面。如果你没有选择任何文本，它就会启动标签对匹配器并使用结果。它同样能搞清楚你的光标的位置：标签的内容里面或者是开始和关闭标签中间。依赖于它的位置，它会包裹标签的内容或标签本身。

缩写包裹为包裹个别行引入了一个特定的缩写句法。简单跳转到倍增操作符后面的数字，比如：ul#nav>li\*>a。当Zen Coding 发现一个使用未定义的倍增数的时候，它会将它作为一个重复元素：你的章节中有多少行，它就会输出多少次，并将每行的内容放到重复元素的最后一个子元素里面。

如果你在这段文本外面包裹这段缩写div#header>ul#navigation>li.item$\*>a>span：

|  |  |
| --- | --- |
| 1  2  3  4  5 | About Us  Products  News  Blog  Contact Up |

你将会得到以下结果：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | <**div** id="header">  <**ul** id="navigation">  <**li** class="item1"><**a** href=""><**span**>About Us</**span**></**a**></**li**>  <**li** class="item2"><**a** href=""><**span**>Products</**span**></**a**></**li**>  <**li** class="item3"><**a** href=""><**span**>News</**span**></**a**></**li**>  <**li** class="item4"><**a** href=""><**span**>Blog</**span**></**a**></**li**>  <**li** class="item5"><**a** href=""><**span**>Contact Up</**span**></**a**></**li**>  </**ul**>  </**div**> |

你可以看到，Zen Coding是一个强大的文本处理工具。

**快捷键**

* Ctrl+,  
  展开缩写
* Ctrl+M  
  匹配对
* Ctrl+H  
  使用缩写包括
* Shift+Ctrl+M  
  合并行
* Ctrl+Shift+?  
  上一个编辑点
* Ctrl+Shift+?  
  下一个编辑点
* Ctrl+Shift+?  
  定位匹配对

这些快捷键是可以自定义的。

**在线演示**

你已经学到很多关于Zen Coding如何工作以及它是如何使你的编码更容易了。现在为什么不自己尝试一下呢？因为Zen Coding是用纯JavaScript开发并迁移到Python，它甚至可以用于浏览器内部，这令它成为引入到CMS的首选。

* [Demo](http://zen-coding.ru/demo/) (使用 Ctrl + , 展开缩写，需要JavaScript支持)
* [中文版演示](http://labs.qianduan.net/zencoding/)

**支持的编辑器**

Zen Coding并不依赖某个特定的编辑器。它是一个只处理文本的出色的组件：它获取文本、做一些处理并放回新的文本(或索引，用于标签匹配)。Zen Coding由JavaScript和Python编写，所以它实际上可以运行于任何平台。在Windows，你可以运行JavaScript版本，而Mac和Linux 分支可以使用Python版。

如果让你的编辑器支持Zen Coding，你需要写一个特定的可以在你的编辑器和Zen Coding之间转换数据的插件。问题是一个编辑器可能不会完整的支持Zen Coding因为它本身的插件系统。比如，[TextMate](http://macromates.com/)通过使用脚本输出替换当前行很容的就支持了“展开缩写”功能，但是它不能处理标签对匹配因为没有标准的方法请求TextMate来选择内容。

**完全支持**

* [Aptana](http://code.google.com/p/zen-coding/downloads/detail?name=Zen.Coding-Aptana.v0.5.zip) (跨平台);
* Coda, via [TEA for Coda](http://github.com/sergeche/tea-for-coda/downloads) (Mac);
* Espresso, via [TEA for Espresso](http://github.com/sergeche/tea-for-espresso/downloads) (Mac);

**部分支持(只支持“展开缩写”)**

* [TextMate](http://code.google.com/p/zen-coding/downloads/detail?name=Zen%20Coding%20for%20TextMate%20v0.3.1.zip) (只能用于Mac机,Windows可以使用E-text编辑器替代);
* [TopStyle](http://code.google.com/p/zen-coding/downloads/detail?name=TopStyle.Zen.Coding.1.0.zip);
* [Sublime Text](http://code.google.com/p/zen-coding/downloads/detail?name=Sublime.Zen.Coding.1.1.3.zip);
* [GEdit](http://www.kryogenix.org/days/2009/09/21/zen-coding-for-gedit);
* [Dreamweaver CS4](http://zen-coding.googlecode.com/files/Zen.Coding-Dreamweaver.v0.4.zip)
* [editArea在线编辑器](http://zen-coding.ru/demo/);
* [Zen Coding在线编辑器中文版](http://labs.qianduan.net/zencoding/)

Aptana是我主要的开发环境，它使用一个JavaScript版本的Zen Coding。它也包含很多其它的我用于日常工作的工具，所以任何一个新的Zen Coding版本都将会首先对Aptana可用，然后部署到Python并兼容其它的编辑器。

Coda和Espresso 插件被杰出的[Text Editor Actions](http://onecrayon.com/tea/) (TEA) 平台支持，由[Ian Beck](http://beckism.com/)开发。原始的源代码[在GitHub上](http://github.com/onecrayon)，但我还是制作了[我自己的分支](http://github.com/sergeche/)以整合Zen Coding的特性。

**总结**

很多尝试过Zen Coding的人都说它改变了他们写页面的方式。当然还有很多事情要做，还有很多的编辑器需要被支持以及一些文档要写。请浏览[现在的文档](http://code.google.com/p/zen-coding/w/list) 以及[源代码](http://code.google.com/p/zen-coding/source/browse/#svn/branches/serge.che)以寻找你的问题的答案。希望你喜欢Zen Coding!