**[inline-block 前世今生](http://ued.taobao.com/blog/2012/08/15/inline-block/" \o "inline-block 前世今生)**

2012年08月15日 | [前端技术](http://ued.taobao.com/blog/category/%e5%89%8d%e7%ab%af%e6%8a%80%e6%9c%af/)

[](http://ued.taobao.com/blog/wp-content/uploads/2012/08/inline-block-banner.png)

曾几何时，display:inline-block 已经深入「大街小巷」，随处可见 「display:inline-block; \*display:inline; \*zoom:1; 」这样的代码。如今现代浏览器已经全面支持这个属性值了，上面的代码只是为了兼容 IE6、7 而已。那么你真的了解 inline-block 了吗？本文将带你深入剖析该属性值的前世今生，让你更好的理解和运用 inline-block。（本文约定 display:inline-block 简写为 inline-block）

开篇我们来看几个问题：

* IE6、7 真的不支持 display:inline-block 吗？
* display:inline-block 后的元素为什么会产生水平空隙，这真的是 bug 吗？
* 如何更好的解决 display:inline-block 元素间产生的水平空隙？

**一、inline-block 前世**

**1.认知**

也许有人问你为何要写「 display:inline-block; \*display:inline; \*zoom:1; 」 来兼容 IE6、7 时，你会立马答道：因为 IE6、7 不支持 display:inline-block 呗!不知道何时起，惯性思维给开发者带来了这样一个可怕的概念。万物都是辩证的，当你写下这些的时候，可曾怀疑过大众观点真的可靠吗？也许你认为这些无关 紧要，实现效果就好。但是如果不能理解每个属性或属性值的根本，你将永远无法全面的了解它，人云亦云只会让你浅尝辄止，止步不前。那么这里就涉及到所谓的 「CSS 学习瓶颈」的问题了，这个问题张鑫旭[《说说CSS 学习中的瓶颈》](http://www.zhangxinxu.com/wordpress/?p=2523" \t "_blank)一文有详细阐述，虽然部分观点我不是很赞同，但是中心思想还是很值得思考的。文中有几个不错的问题这里也列举出来供大家观摩：

1. line-height:150% 和 line-height:1.5 的区别是？
2. float 为何会让外部容器高度塌陷？这是 bug？（我的答案在[《那些年我们一起清除过的浮动》](http://www.iyunlu.com/view/css-xhtml/55.html" \o "浮动前世今生，清除浮动综合解决方案" \t "_blank)）
3. vertical-align 的表现为何在IE7, IE8, IE9 下表现不尽相同？其中的渲染机制是？

好了，回到 inline-block 的认知的问题，我的观点是：

**IE 从 5.5 开始就已经支持 display:inline-block 了，只是支持的并不是那么完善**。

在 msdn 微软开发者社区，找到了 IE 从5.5 开始支持 inline-block 的证据：

The **inline-block** value is supported starting with Internet Explorer 5.5. You can use this value to give an object a layout without specifying the object’s height or width.

这里明确指出：从 IE5.5 开始支持 inline-block。

链接：<http://msdn.microsoft.com/zh-cn/library/ie/ms530751(v=vs.85).aspx>

那么既然 IE5.5 开始就已经支持了 inline-block，为何我们还要写那么一坨 CSS 呢？同时我们知道 IE6、7 中 display:inline-block 是可以触发[hasLayout](http://msdn.microsoft.com/en-us/library/bb250481(VS.85).aspx" \t "_blank)的，触发了 [hasLayout](http://msdn.microsoft.com/en-us/library/bb250481(VS.85).aspx" \t "_blank)的元素表现出来的特征就是一个独立的矩形容器，可以设置宽高而且不受外部元素的影响，类似于现代浏览器中的 Block formatting contexts （块级格式化上下文）的概念。

下面来做一个详细的测试，分别看看 IE6 中 inline 元素和 block 元素的表现：

**1）inline 元素 display:inline-block**

IE6 中截图如下：



.dib-inline, .dib-block {  
width:100px;  
height:30px;  
line-height:30px;  
text-align:center;  
}  
.dib-inline {  
display:inline-block;  
}

测试表明：IE6 中 inline 元素只要触发了[hasLayout](http://msdn.microsoft.com/en-us/library/bb250481(VS.85).aspx) 其表现就类似于 inline-block，这里设置 display:inline-block; 或者 zoom:1; 等其他属性值可以触发 [hasLayout](http://msdn.microsoft.com/en-us/library/bb250481(VS.85).aspx" \t "_blank)，表现出来是一样的。

[**查看 DEMO**](http://www.iyunlu.com/demo/inline-block-ie/)

**2）block 元素 display:inline-block**

IE6 中截图如下：

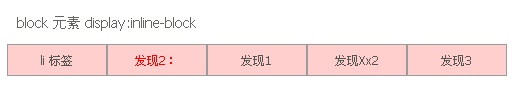


.dib-inline, .dib-block {  
width:100px;  
height:30px;  
line-height:30px;  
text-align:center;  
}  
.dib-block{  
display:inline-block;  
}

测试表明：IE6 中 block 元素即使触发了 hasLayout 也不能具有 inline-block 元素不换行的特性。想要 block 元素支持 inline-block 元素的特性，我们可以这样做：

.dib-block {  
display:inline;  
zoom:1;  
}

首先让 block 元素转化为 inline 元素，强制其不换行；然后通过 zoom:1 触发[hasLayout](http://msdn.microsoft.com/en-us/library/bb250481(VS.85).aspx" \t "_blank)，使其可以设置宽高。修复后的 截图如下：



**3）结合现代浏览器**

综上，现代浏览器都支持 display:inline-block ，IE6、7 inline 元素也可以达到同样的效果，IE6、7 block 元素需要设置 display:inline; zoom:1; 它们结合在一起便是：

display:inline-block; /\* 现代浏览器 +IE6、7 inline 元素 \*/  
\*display:inline; /\* IE6、7 block 元素 \*/  
\*zoom:1;

为了不让支持 CSS2.1 inline-block 的浏览器 重置为 inline，我们针对 IE6、7 做一个 hack。由于现代浏览器也开始支持 zoom 属性，这里只是希望 IE6、7 中生效，所以还是 hack 一下比较合适。至此产生了我们熟悉的兼容各个浏览器的 inline-block 写法。

**小结：IE6、7 并不是不支持 inline-block，只是 block 元素需要做一些处理来达到 inline-block 的效果。**

**2. 到底什么是 inline-block**

说了很多，或许很多朋友还不是太明白到底什么是 inline-block？W3C 在 CSS2.1 [The ‘display’ property](http://www.w3.org/TR/CSS2/visuren.html#display-prop)中描述如下：

This value causes an element to generate an inline-level block container. The inside of an inline-block is formatted as a block box, and the element itself is formatted as an atomic inline-level box.

大致意思就是：inline-block 后的元素创建了一个行级的块容器，该元素内部（内容）被格式化成一个块元素，同时元素本身则被格式化成一个行内元素。

直白一点的意思就是：inline-block 的元素既具有 block 元素可以设置宽高的特性，同时又具有 inline 元素默认不换行的特性。当然不仅仅是这些特性，比如 inline-block 元素也可以设置 vertical-align 属性。简而言之：

**inline-block 后的元素就是一个格式化为行内元素的块容器( Block container )**

怎么样？听起来还不错吧！

**3. inline-block 缘从何起？**

前面已经证明了 IE 5.5 开始就支持了 inline-block，那么 IE5.5 是什么时候发布的呢？话说当年网景与 IE 大战，IE5.5 那是何等的风骚……（好吧，此处略去十页）。从[维基百科的资料来](http://zh.wikipedia.org/zh/Internet_Explorer_5" \t "_blank)看，IE5.5 beta1 的发布时间是：**1999年12月**，最终版本是**2000年7月**。那么 W3C 标准中是何时才出现 inline-block 这个值的呢？

在 [CSS1](http://www.w3.org/TR/CSS1/#display) 规范中，「display」的值仅包括： block | inline | list-item | none 。CSS2.1中才添加了 inline-block 属性值。一丝继续舔着手指，用那苦逼的英语水平终于翻到了这份草案：[http://www.w3.org/TR/2002/WD-CSS21-20020802/visuren.html#display-prop](http://www.w3.org/TR/2002/WD-CSS21-20020802/visuren.html" \l "display-prop" \t "_blank)， 这份草案的日期是 2002年8月2日，纳尼！！！原来我们纠结了半天的 inline-block ， IE5.5 至少提前两年就提出来了啊！难道是微软给 W3C 提议后，CSS 2.1才加入的？（不过我看到 W3C 官网有一个关于是否增加 inline-block 的投票）好吧这个问题也许有一天 IE 某个开发者写《 IE回忆录》的时候我们才能了解到其中的内幕。如果找到更早关于 inline-block 的 CSS草案，也麻烦告知一丝一声。好吧，如果你还不相信，打微软官方电话问问吧**800-820-3800**（不是 DHC 哦！）。

原来我们一直讨论的 inline-block 在 IE 6、7中和 CSS2.1 中的（现代浏览器所支持的） inline-block 上压根不是一个东东嘛，IE6、7 中的 inline-block 更像是 IE 的**私有属性值**，他们本身就不具有可比性。简单、绝对的认为 IE6、7 不支持 inline-block 好比一叶障目，看到前面，却看不到后面，太过于片面。诚然，IE6、7 的 [hasLayout](http://msdn.microsoft.com/en-us/library/bb250481(VS.85).aspx" \t "_blank) 给我们带来了很多麻烦，但是不得不承认微软的 IE 在网页多语言文本混排上的先进性，尤其是 CJK 文字和西文的混排，超越其他浏览器至少5年。

总结：

* **IE5.5 后开始支持 inline-block， 但是它所支持的 inline-block 不能等同于 CSS2.1 中的 inline-block，因为 IE5.5 比 CSS2.1 更早提出 inline-block 的概念并作为所谓的**「**私有属性值**」**使用，所以二者表现出来的效果是不完全一致。**
* **IE 5.5、6、7 、8（Q）中 block 元素对 inline-block 支持不完整，如果要达到类似的效果，需要先设置为 display:inline，然后使用 zoom:1 等**触发[hasLayout](http://msdn.microsoft.com/en-us/library/bb250481(VS.85).aspx)。
* **IE 5.5、6、7 、8（Q）中 inline 元素欲达到 inline-block 的效果只需直接设置此属性值或使用 zoom:1 等均可。**

各浏览器对 display 属性的支持情况请参阅：[《各浏览器对 ‘display’ 特性值的支持程度不同》](http://www.w3help.org/zh-cn/causes/RM8001" \t "_blank)

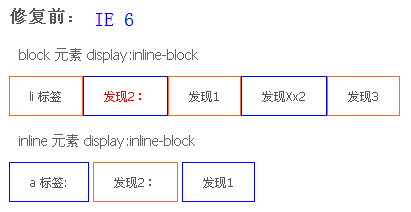


**二、inline-block 今生**

**1. display:inline-block 后的元素为什么会产生水平空隙，这真的是 bug 吗？**

这么一个神奇的属性，为何大家一直避而远之呢？这恐怕还得从 inline-block 元素之间产生的水平空隙（间隙）说起吧。

[**参照 DEMO**](http://www.iyunlu.com/demo/inline-block/)

* 现代浏览器中 inline 和 block 元素 display:inline-block 后均会产生水平空隙；  
  
* IE6、7，IE8（**Q**）模拟 display:inline-block 后分两种情况：  
    
  IE6、7，IE8（**Q**）中：inline 元素会产生空隙，block 元素不会产生空隙。

看看 inline 元素默认的表现情况如何？原来默认就有空隙存在！它们是谁？是空白符（white space）！



W3C[9.1 White space](http://www.w3.org/TR/html401/struct/text.html#h-9.1)中规定以下元素属于空白符（white space）：

* ASCII 空格 (&#x0020;)
* ASCII 制表符 (&#x0009;)
* ASCII 换页符 (&#x000C;)
* 零宽度空格 (&#x200B;)「这个在闭合浮动中也有运用到」

[9.3.2 Controlling line breaks](http://www.w3.org/TR/html401/struct/text.html#h-9.3.2) 中进一步阐述：

A line break is defined to be a carriage return (&#x000D;), a line feed (&#x000A;), or a carriage return/line feed pair. All line breaks constitute[white space.](http://www.w3.org/TR/html401/struct/text.html#whitespace)

For more information about SGML’s specification of line breaks, please consult the [notes on line breaks](http://www.w3.org/TR/html401/appendix/notes.html#notes-line-breaks) in the appendix.

折行被定义为一个回车符（&#x000D;），一个换行符 line feed (&#x000A;)，或者一个回车、换行的组合。所有的折行构成了空白符。

有关 SGML 规范中折行的更多信息，请参阅附录中关于折行的注释。

通常情况下，对于多个连续的空白符（空格，换行符，回车符等），浏览器会将他们合并为一个空白符。CSS 中由 white-space 这个属性来控制：

**white-space**：normal | pre | nowrap | pre-wrap | pre-line

**默认值**：normal

* normal：默认处理方式。
* pre：用等宽字体显示预先格式化的文本，不合并文字间的空白距离，当文字超出边界时不换行。可查阅 pre 对象
* nowrap：强制在同一行内显示所有文本，直到文本结束或者遭遇 br 对象。
* pre-wrap：用等宽字体显示预先格式化的文本，不合并文字间的空白距离，当文字碰到边界时发生换行。
* pre-line：保持文本的换行，不保留文字间的空白距离，当文字碰到边界时发生换行。



注：IE7及更早浏览器不支持 CSS2.1 新增的 pre-wrap | pre-line。

所以这并不是 inline-block 后产生的 bug，而是因为 inline-block 具有 inline 元素固有的特性。那么为何 IE6、7 block 元素没有产生空隙呢？其实前面也提到了 IE 的[hasLayout](http://msdn.microsoft.com/en-us/library/bb250481(VS.85).aspx" \t "_blank)，具有独立性，所以产生 [hasLayout](http://msdn.microsoft.com/en-us/library/bb250481(VS.85).aspx" \t "_blank) 的元素之间表现出来互不影响，这也再次表明 IE6、7 中的 inline-block 不能等同于 CSS2.1 中的 inline-block。如果非要说是有 bug， IE6、7 block 元素 inline-block 后不产生空隙才是 bug。

测试表明删除换行符后，inline 元素间的空隙就「消失」了：



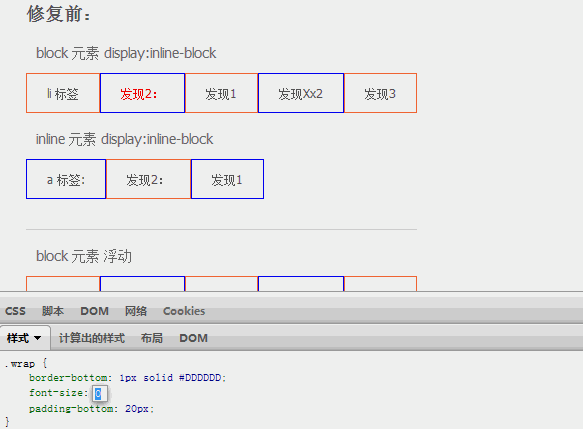
**2.去掉 inline-block 产生的空隙**

为了让各个浏览器表现一致，更好的还原视觉设计搞，很多时候我们需要去掉 inline-block 产生的空隙。

上一节中我们已经知道产生空隙的根本性原因是：

**HTML 中的**换行符、空格符、制表符等产生了空白符，而这些归根结底都是字符，那么它们的大小都是 受 font-size 来控制的，字体大小直接导致 inline 或者 inline-block 后元素之间空隙的大小，把 inline-block 元素间的空隙认为总是某个固定大小是错误的。

用 GIF 动画的形式来表明对应关系：



很清楚的看到，当 font-size:0 的时候元素间的空隙都为0了，或许到这里你会感到很欣喜了，原来掌握的根本性原因这么简单就搞定了啊！

然，**理想是丰满的，现实是骨感的。**

大部分浏览器是支持 font-size:0 的。很明显，我们要和 IE 6、7 这两个妖孽进行一番战斗。

**font-size:0 的支持情况**

1）Chrome

低版本的 chrome 浏览器为了不让文字过小不利于阅读，默认是不支持 font-size:0 的，还好我们有 -webkit-text-size-adjust 这个私有属性来控制，当设为 none 时就支持字体大小为 0 了。我已经记不清楚 chrome 从哪个版本开始支持 font-size:0 了，反正我用 chrome 19 是支持了（有知道的朋友，烦请告诉一丝一声，最好有官方更新说明）。但是，**-webkit-text-size-adjust:none; 会直接导致页面文字无法缩放，这对于用户来说显然是不友好的**。所以-webkit-text-size-adjust:none; 一定要慎用，确保使用的地方没有大面积的文字。

-webkit-text-size-adjust:none 的使用场景实例参阅：[http://vip.etao.com/](http://vip.etao.com/?spm=1002.1.0.20)



2）Safari

Safari 5 依旧不支持 font-size:0 ，不过相信这些浏览器厂商都意识到了这个问题，在 Mac 平台最新的 Safari 6 已经很好的支持 font-size:0 了。

3）Firefox，Opera

经测试，Firefox12，Opera 10 ，这次表现不错，支持 font-size:0 。

4）IE

* IE8 以上支持 font-size:0;
* IE6、7 inline 元素 inline-block 后设置 font-size:0 始终有 1px 的空隙。

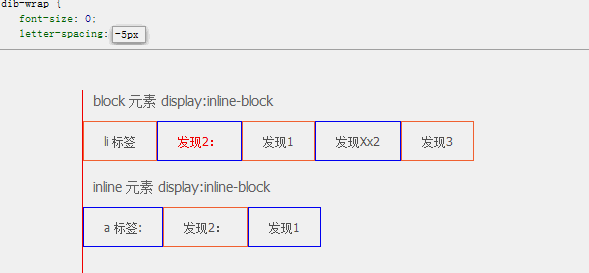
是不是一下子又开始头疼了？没关系，让我们请出**letter-spacing** 和**word-spacing** 二位大神。既然空白符也是字符，那么二位大神肯定是可以搞定它们的。

* **letter-spacing : normal** | *length* （检索或设置对象中的文字之间的间隔）
* **word-spacing : normal** | *length*（检索或设置对象中的单词之间插入的空隔）

normal： 默认间隔  
length： 用长度值指定间隔，允许为负值。

还等什么，我们赶紧试试吧：

[**参照 DEMO**](http://www.iyunlu.com/demo/inline-block/)

* 第一步：使用 font-size:0经测试发现，chrome、firefox、IE8+、opera，inline 或 block 元素都没有空隙了；  
  **Safari 5.1.7 由于不支持 font-size:0 ，仍然存在空隙；**  
    
  IE6、7、8（Q），inline 元素 inline-block 后始终存在 1px 左右的空隙。
* 第二步：处理 Safari 不支持 font-size:0 的问题上面已经指出 letter-spacing 是支持负值的，那么这个负值到底取多少合适呢？经过测试得出的结论是：**inline-block 产生的空隙与父级元素继承或者设定的 font-family、font-size 有关，通常情况下，12px 大小的 tahoma 字体，inline-block 后元素间产生的空隙（间隙）大约是 5px；**各个字体详细情况请参阅[《inline-block空隙–letter-spacing与字体大小/字体关系数据表》](http://www.iyunlu.com/view/css-xhtml/58.html)**。**Firefox 中 letter-spacing 负值的绝对值大于空隙大小后，会导致元素整体位置向右偏移；**Safari 中 letter-spacing 负值的绝对值大于空隙大小后，内部会发生重叠。**
* 第三步：修复 IE6、7 中始终存在的 1px 空隙  
  既然 letter-spacing 已经无能为力了，那就试试 word-spacing 吧，直接设置 word-spacing:-1px。这里需要注意的是，letter-spacing 和 word-spacing 同时使用可能导致冲突，所以我们需要在 IE6、7 中 hack 掉 letter-spacing。最终代码如下：

font-size:0;/\* 所有浏览器 \*/  
letter-spacing:-5px;/\* Safari 等不支持字体大小为 0 的浏览器 \*/  
\*letter-spacing:normal;  
word-spacing:-1px;/\* IE6、7 \*/

* 第四步：子元素重置回正常值  
  上述所有操作都是在父元素设置的，那么子元素都会继承这些属性，字体大小为0了，子元素就什么都看不到了，这并不是我们想要的。 同时字符和单词间距我们也要把它重置为默认值。「font-size: 12px; letter-spacing: normal; word-spacing: normal;」
* 最后：inline-block 更好的复用  
  或许你会担心每次我都要去看字体和空隙之间大小的关系吗？其实不然，通常情况下，全局字体都已经在 body 中指定了，根据全局字体设置合适的 letter-spacing 负值即可。如此一来，我们便可以放心大胆的使用 inline-block 了，结合 OOCSS 的思想，可以抽离出两个复用的类，在需要设置 inline-block 元素的父级元素上定义一个**「.dib-wrap」**，该元素自身定义为**「.dib」**。这里还有一个问题需要注意的是：由于 inline-block 具有 inline 元素的特性，在垂直方向上很多时候我们并不希望元素以**「vertical-align:baseline」**方式来呈现，所以在**「.dib-wrap」**中统一重置为**「vertical-align:top」**即可。

**3. 去除 inline-block 空隙终极解决方案（2012年8月17日更新）**

.dib-wrap {  
font-size:0;/\* 所有浏览器 \*/  
\*word-spacing:-1px;/\* IE6、7 \*/  
}  
.dib-wrap .dib{  
font-size: 12px;  
letter-spacing: normal;  
word-spacing: normal;  
vertical-align:top;  
}  
@media screen and (-webkit-min-device-pixel-ratio:0){  
/\* firefox 中 letter-spacing 会导致脱离普通流的元素水平位移 \*/  
.dib-wrap{  
letter-spacing:-5px;/\* Safari 等不支持字体大小为 0 的浏览器, N 根据父级字体调节\*/  
}  
}  
.dib {  
display: inline-block;  
\*display:inline;  
\*zoom:1;  
}

其实在[YUI 3](http://yui.yahooapis.com/3.5.1/build/cssgrids/grids.css" \t "_blank)中也全面运用了 inline-block 作为基础布局，[YUI 3](http://yui.yahooapis.com/3.5.1/build/cssgrids/grids.css" \t "_blank) 是这样解决的：

.yui3-g {  
letter-spacing: -0.31em; /\* webkit: collapse white-space between units \*/  
\*letter-spacing: normal; /\* reset IE < 8 \*/  
word-spacing: -0.43em; /\* IE < 8 && gecko: collapse white-space between units \*/  
}

.yui3-u {  
display: inline-block;  
zoom: 1; \*display: inline; /\* IE < 8: fake inline-block \*/  
letter-spacing: normal;  
word-spacing: normal;  
vertical-align: top;  
}

显然，这里纯粹使用了 letter-spacing 和 word-spacing 来控制元素间的空隙，局限性极大，-0.31em 和 -0.43em 只是因为 YUI 3 全局[cssfonts.css](http://yui.yahooapis.com/3.6.0/build/cssfonts/cssfonts.css" \t "_blank)里设置是：「body { font:13px/1.231 arial,helvetica,clean,sans-serif; }」。

当然，如果你坚持使用把 html 写在一行的方式来达到去除 inline-block 空隙的目的，我只能说：**一切以牺牲结构来兼容表现的行为都是耍流氓**！所以探讨此种方式去除空隙也将是无意义的，不在本文和作者考虑范围之内。

**4. 结局——本文产生的一些观点如下：**

* **IE5.5 后开始支持 inline-block， 比 CSS2.1 更早提出 inline-block 的概念并作为所谓的**「**私有属性值**」**使用。但是它所支持的 inline-block 不能等同于 CSS2.1 中的 inline-block，IE 5.5、6、7 、8（Q）中 block 元素对 inline-block 支持不完整，因此二者表现出来的效果是不完全一致。**
* **产生 inline-block 空隙的根本性原因是：HTML 中的换行符、空格符、制表符等合并为空白符，字体大小不为 0 的情况下，空白符自然占据一定的宽度，因此产生了元素间的空隙。**
* **慎用 -webkit-text-size-adjust:none，它将会导致页面无法通过缩放来改变字体大小。**

**三、inline-block 未来**

如今，Mac 平台下的 Safari 6 已经支持 font-size:0 了，相信很快 Windows 平台的 Safari 如果发布 5.X 的更新，也会支持字体为 font-size:0 了。等到 IE6、7 灭亡之后，世界就真真儿的美妙了！最后说一点：**inline-block 与 float 也是无法直接比较的，请不要再讨论 inline-block 和 float 哪个更好的话题了。**inline-block 从 IE5.5 一路走来，存在即是合理，以后有时间在总结一下 inline-block 与 float 的使用场景的区别。

在不改变 CSS 定位机制的前提下，inline-block 应该是首选，而不是以「奇淫技巧」存在的。有感打油诗一首：

网事如烟  
CSS 红尘里  
inline-block 知多少  
你在这头  
inline-block 在那头  
用与不用  
它就在那里  
不悲不喜

PS：

1. 为了更好的排版，本文使用繁体中文引号 「」代替简体中文小蝌蚪引号；
2. 中英文混排的时候英文首尾各加一个空格；
3. 以后文章有需要的时候也都将使用 gif 动画配合说明。

**测试环境**

|  |  |
| --- | --- |
| **操作系统版本:** | Windows 7 企业版 6.1（内部版本 7600） |
| **浏览器版本:** | IE6 IE9 Firefox 14.0.1 Chrome 19.0.1084.46 Safari 5.1.7(7534.57.2) Opera 12.50 |
| **最后更新时间:** | 2012-8-22 |

一丝  
2012-8-10 于杭州