haslayout 是Windows Internet Explorer渲染引擎的一个内部组成部分。在InternetExplorer中，一个元素要么自己对自身的内容进行计算大小和组织，要么依赖于父元素来计算尺寸和组织内容。为了调节这两个不同的概念，渲染引擎采用了 hasLayout 的属性，[属性值](https://baike.baidu.com/item/%E5%B1%9E%E6%80%A7%E5%80%BC" \t "https://baike.baidu.com/item/hasLayout/_blank)可以为true或false。当一个元素的 hasLayout属性值为true时，我们说这个元素有一个布局（layout）

要想更好的理解 css， 尤其是 IE 下对 css 的渲染，haslayout 是一个非常有必要彻底弄清楚的概念。大多IE下的显示错误，就是源于 haslayout。如果它设置成了true，它就不得不去渲染它自己，因此元素不得不扩展去包含它的流出的内容。例如浮动或者很长很长的没有截断的单词，如果haslayout没有被设置成true，那么元素得依靠某个祖先元素来渲染它。这就是很多的ie bugs诞生的地方。

当一个元素有一个布局时，它负责对自己和可能的子孙元素进行尺寸计算和定位。简单来说，这意味着这个元素需要花更多的代价来维护自身和里面的内容，而不是依赖于祖先元素来完成这些工作。因此，一些元素默认会有一个布局。当我们说一个元素“拥有layout”或“得到layout”，或者说一个元素“has layout” 的时候，我们的意思是指它的微软专有属性 hasLayout 被设为了 true。一个“layout元素”可以是一个默认就拥有 layout 的元素或者是一个通过设置某些 CSS 属性得到 layout的元素。通过 IE Developer Toolbar 可以查看 IE 下 HTML元素是否拥有haslayout，在 IE Developer Toolbar 下，拥有 haslayout的元素，通常显示为“haslayout = -1”。

负责组织自身内容的元素将默认有一个布局，主要包括以下元素（不完全列表）：

\* body and html

\* table, tr, th, td

\* img

\* hr

\* input, button, file, select, textarea, fieldset

\* marquee

\* frameset, frame, iframe

\* objects, applets, embed

对于并非所有的元素都默认有布局，微软给出的主要原因是“性能和简洁”。如果所有的元素都默认有布局，会对性能和内存使用上产生有害的影响。

如何激发 haslayout？

大部分的 IE 显示错误，都可以通过激发元素的 haslayout 属性来修正。可以通过设置 css 尺寸属性(width/height)等来激发元素的 haslayout，使其“拥有布局”。如下所示，通过设置以下 css 属性即可。

\* display: inline-block

\* height: (任何值除了auto)

\* float: (left 或 right)

\* position: absolute

\* width: (任何值除了auto)

\* writing-mode: tb-rl

\* zoom: (除 normal 外任意值)

Internet Explorer 7 还有一些额外的属性(不完全列表):

\* min-height: (任意值)

\* max-height: (除 none 外任意值)

\* min-width: (任意值)

\* max-width: (除 none 外任意值)

\* overflow: (除 visible 外任意值)

\* overflow-x: (除 visible 外任意值)

\* overflow-y: (除 visible 外任意值)

\* position: fixed

其中 overflow-x 和 overflow-y 是 css3 盒模型中的属性，目前还未被浏览器广泛支持。

对于内联元素(默认即为内联的元素，如 span，或 display:inline; 的元素)，

width 和 height 只在 IE5.x 下和 IE6 或更新版本的 quirks 模式下触发 hasLayout 。而对于IE6，如果浏览器运行于标准兼容模式下，内联元素会忽略 width 或 height 属性，所以设置 width 或 height不能在此种情况下令该元素具有 layout。

zoom 总是可以触发 hasLayout，但是在 IE5.0 中不支持。

具有“layout” 的元素如果同时 display: inline ，那么它的行为就和标准中所说的 inline-block很类似了：在段落中和普通文字一样在水平方向和连续排列，受 vertical-align影响，并且大小可以根据内容自适应调整。这也可以解释为什么单单在 IE/Win 中内联元素可以包含块级元素而少出问题，因为在别的浏览器中display: inline 就是内联，不像 IE/Win 一旦内联元素拥有 layout 还会变成 inline-block。

haslayout 问题的调试与解决

当网页在 IE 中有异常表现时，可以尝试激发 haslayout 来看看是不是问题所在。常用的方法是给某元素 css 设定 zoom:1。使用 zoom:1 是因为大多数情况下，它能在不影响现有环境的条件下激发元素的 haslayout。而一旦问题消失，那基本上就可以判断是haslayout 的原因。然后就可以通过设定相应的 css 属性来对这个问题进行修正了。建议首先要考虑的是设定元素的width/height 属性，其次再考虑其他属性。

对 IE6 及更早版本来说，常用的方法被称为霍莉破解(Holly hack)，即设定这个元素的高度为 1%(height:1%;)。需要注意的是，当这个元素的 overflow 属性被设置为 visible 时，这个方法就失效了。或者使用 IE的条件注释。

对 IE7 来说，最好的方法是设置元素的最小高度为 0 (min-height:0;)。

haslayout 问题引起的常见 bug

IE6 及更低版本的双空白边浮动 bug

bug 修复: display:inline;

IE5-6/win 的 3 像素偏移 bug

bug 修复: \_height:1%;

IE6 的躲躲猫(peek-a-boo) bug

bug 修复: \_height:1%;

IE6/7负margin隐藏Bug：

bug 修复:去掉父元素的hasLayout；或者赋hasLayout给子元素,并添加position:relative;

需要注意的是，hasLayout属性是微软特有的过时属性，在IE8、IE9中，hasLayout属性已经被废弃。上文中的InternetExplorer都是指IE7、IE6及以下版本。