--

[position > absolute relative 1](#_Toc18159)

[position:absolute 基础特性 1](#_Toc20290)

[包裹性 1](#_Toc2503)

[abs的包裹性 1](#_Toc30506)

[fl的包裹性 1](#_Toc20954)

[破坏性 2](#_Toc11313)

[abs滥用 3](#_Toc9509)

[absolute与负margin布局对比 3](#_Toc3281)

[说明文字在大背景中的定位 3](#_Toc24293)

[将absolute元素置于body 3](#_Toc2900)

[absolute与元素隐藏 3](#_Toc11391)

[说明 3](#_Toc21133)

[1、可用性隐藏 3](#_Toc15486)

[2、回流与渲染 3](#_Toc24458)

[3、配合JavaScript的控制 3](#_Toc13374)

[absolute与左右内容不同高，背景相同布局 4](#_Toc6985)

[absolute实现等高布局 4](#_Toc6815)

[absolute对前后元素的影响，以及在ie6/7中的不同表现 4](#_Toc4968)

[前元素abs化，遮蔽原图层 4](#_Toc18462)

[+IE6/7下margin与absolute元素重叠 4](#_Toc632)

[float元素与absolute元素（line/block） 4](#_Toc26856)

[+ie6 7异常 5](#_Toc22603)

[absolute绝对定位的非绝对定位用法 5](#_Toc14458)

[abs特性 5](#_Toc2456)

[文字的阴影效果 5](#_Toc2864)

[自适应 6](#_Toc2495)

[relative 6](#_Toc7548)

[relative针对absolute的常用方式 6](#_Toc10168)

[通过外部元素relative限制内部absolute 6](#_Toc7575)

[relative其他特性 7](#_Toc3956)

[与left 7](#_Toc19427)

[于纵深度 7](#_Toc31536)

## position > absolute relative

|  |  |
| --- | --- |
|  | CSS 相对|绝对(relative/absolute)定位系列（一）  <http://www.zhangxinxu.com/wordpress/2010/12/css-%E7%9B%B8%E5%AF%B9%E7%BB%9D%E5%AF%B9%E5%AE%9A%E4%BD%8D%E7%B3%BB%E5%88%97%EF%BC%88%E4%B8%80%EF%BC%89/>  CSS 相对/绝对(relative/absolute)定位系列（二）  <http://www.zhangxinxu.com/wordpress/2010/12/css-%E7%9B%B8%E5%AF%B9%E7%BB%9D%E5%AF%B9relativeabsolute%E5%AE%9A%E4%BD%8D%E7%B3%BB%E5%88%97%EF%BC%88%E4%BA%8C%EF%BC%89/> |
|  | jQuery powerFloat万能浮动层下拉层插件  http://www.zhangxinxu.com/wordpress/2010/12/jquery-powerfloat%E4%B8%87%E8%83%BD%E6%B5%AE%E5%8A%A8%E5%B1%82%E4%B8%8B%E6%8B%89%E5%B1%82%E6%8F%92%E4%BB%B6/ |

position:absolute 基础特性

|  |  |  |  |
| --- | --- | --- | --- |
|  | position:absolute与float:left两者有两大共性：包裹性，破坏性。  ===  包裹性：换种说法就是让元素inline-block化：  div标签默认宽度是100%显示的，但是一旦被absolute属性缠上，则100%默认宽度就会变成自适应内部元素的宽度；  span标签默认是不响应height width的，但是一旦被abs后，则会响应用户定义的高宽；  ===  float也是典型的inline-block化元素，这种元素的inline-block化适用于任何水平的标签。例如平时我们要让span标签支持width属性，可能要设置：  span { display:block; width:100px; }  有float:left/position:absolute撑腰的情况下，display属性就是多余的：  span { float:left; width:100px; }  span { position:absolute; width:100px; }  ===  破坏性：元素脱离文档流  浮动的破坏性在与于脱离line box链进行前移，但是还是会在页面line box流中占据位置（宽度存在）；  而绝对定位在页面中不占据任何空间，在高度与宽度上完全塌陷；（看起来垂直于页面上升了，在底层页面感受不到存在）（ps的图层）  浮动：高度虚化  abs：高度、宽度虚化，内部元素abs后，不会去撑开父元素的高宽，会导致高宽塌陷，于是，需要设定一个高度值（或是足以撑开高度的值）  当元素同时存在abs定位，左浮动，具体展示依靠abs定位  ===  [添加relative div与span还保持原有特性]  ===  绝对定位这种东西，显然没有流动的气质，尤其拿来定位后（虽然有时候在relative的庇护下也有一定的流动性）。流动性布局很强调不定宽，不定高，活用标签自身属性，顺其自然，最少干预。但是由于absolute属性（尤其是带有left/top值）的破坏性，需要设定一个高度值（或是足以撑开高度的值）。  想重构高质量的页面，少用绝对定位布局！  ===  position:relative天生可以限制position:absolute在left/top翅膀下瞬间移动的能力；否则top值会直接作用相对于浏览器的边际（body）； | | |
| 包裹性 | <**style**>  **span** {  height: 80px;  width: 200px; **/\*width由无效变为有效\*/** }  **div** {  height: 80px;  background-color: #f0f3f9;  }  .abs {  position: absolute;**/\*\*/** }  .fl {  float: left; **/\*\*/** } </**style**> | | |
| abs的包裹性 | <**span**>span 添加position:absolute前 </**span**> <**br**/> <**br**/> <**br**/> <**span** class="abs">span 添加position:absolute后 </**span**> <**br**/> <**br**/> <**br**/> <**div**>添加position:absolute前</**div**> <**br**/> <**br**/> <**br**/> <**div** class="abs">添加position:absolute后</**div**> | | |
|  | | |
| fl的包裹性 | <**span**>span 添加position:absolute前 </**span**> <**br**/> <**br**/> <**br**/> <**span** class="fl">span 添加position:absolute后 </**span**> <**br**/> <**br**/> <**br**/> <**div**>添加position:absolute前</**div**> <**br**/> <**br**/> <**br**/> <**div** class="fl">添加position:absolute后</**div**> | | |
|  | | |
| 破坏性 | | abs与float具体对比  > abs作用强度大于float left，abs会遮盖float left的存在 | |
| 正常 | | <**style**>  .fl {  float: left; **/\*\*/** }  .abs {  position: absolute; **/\*\*/** } </**style**> |  |
| <**img** class="" src="images/1.jpg"/> <**img** class="" src="images/2.jpg"/> 无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、 |
| fl | | <**img** class="fl" src="images/1.jpg"/> <**img** class="fl" src="images/2.jpg"/> 无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、 |  |
| <**img** class="fl" src="images/1.jpg"/> <**img** class="" src="images/2.jpg"/> 无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、 |  |
| <**img** class="" src="images/1.jpg"/> <**img** class="fl" src="images/2.jpg"/> 无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、 |  |
| abs | | <**img** class="abs" src="images/1.jpg"/> <**img** class="abs" src="images/2.jpg"/> 无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、 |  |
| <**img** class="abs" src="images/1.jpg"/> <**img** class="" src="images/2.jpg"/> 无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、 |  |
| <**img** class="" src="images/1.jpg"/> <**img** class="abs" src="images/2.jpg"/> 无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、 |  |
| fl+abs | | <**img** class="fl abs" src="images/1.jpg"/> <**img** class="fl abs" src="images/2.jpg"/> 无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、 |  |
| <**img** class="" src="images/1.jpg"/> <**img** class="fl abs" src="images/2.jpg"/> 无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、 |  |
|  | | <**img** class="" src="images/1.jpg"/> <**img** class="" src="images/2.jpg"/> <**span** class="">  无上甚span.abs </**span**> <**div** class="">  无上甚div.abs </**div**> |  |
| <**img** class="fl" src="images/1.jpg"/> <**img** class="fl" src="images/2.jpg"/> <**span** class="abs">  无上甚span.abs </**span**> <**div** class="abs">  无上甚div.abs </**div**> |  |

abs滥用

|  |  |
| --- | --- |
|  | 在我看来，很多网站，position:absolute都有滥用的嫌疑，不仅仅是position:absolute滥用，其left,top属性值也滥用，relative属性值滥用，z-index也滥用。有的属性滥用就是单纯的多些代码而已；而有些其实可以不使用的地方也使用的话会给后来的扩展和维护造成很大的麻烦的。  relative/absolute/left/top/z-index这5个关系紧密，无论将哪一点都会牵扯到其他。为了避免讲述的混乱，我尽量只讲某一个。例如，这里只讲absolute属性。  absolute经常被一些新手拿来大肆建房子做布局也是可以理解的，貌似什么时候我看过一篇文章说，absolute属性的出现本来是想把页面搞得像photoshop那样，一个图层一个图层覆盖似的。但是，页面的发展显然与这个背道而驰，毕竟页面是活的。确实，像photoshop，把一个个层搞出来，使用absolute属性，以及一定坐标值就可以定位出来了。不需要考虑什么间距啊，margin啊，一些IE下乱七八糟的bug啊什么的，有一了百了的畅快感。尤其对于页面制作不熟悉的新手，这种绝对定位布局可谓屡试不爽。  还有时候，遇到有些复杂的布局，项目经理又催得紧，懒得再去折腾，干脆，直接，绝对定位，先把效果做出来再说。因为绝对定位毕竟有飞翔的能力，例如想在黄浦江中间建个灯塔，直接告知坐标，飞过去，灯塔一插，搞定了，很省事的。  看上去省事，其实是自掘坟墓。所谓“本属天上人，勿管凡间事”，对于普通的页面布局，不到万不得已不要使用absolute进行定位。  对普通的layout，如果动不动就使用absolute属性，我个人是比较深恶痛疾的。  首先自己是个流体布局控，绝对定位这种东西，显然没有流动的气质，尤其拿来定位后（虽然有时候在relative的庇护下也有一定的流动性）。流动性布局很强调不定宽，不定高，活用标签自身属性，顺其自然，最少干预。但是由于absolute属性（尤其是带有left/top值）的破坏性，会导致高宽塌陷，于是，不得已，需要设定一个高度值（或是足以撑开高度的值），例如新浪微博导航就是绝对定位，于是，导航外标签必须定高，否则，下面的元素会上来发生重叠 |

absolute与负margin布局对比

|  |  |  |
| --- | --- | --- |
| 说明文字在大背景中的定位 | 负margin布局  > ？外层元素的高度塌陷了？ | abs布局 |
| <**div** style="position: relative;">  <**img** src="images/1.jpg" alt=""/>  <**div** style="margin: -60px 0 0 30px;">说明</**div**> </**div**> 12345678910123456789 | <**div** style="position: relative">  <**img** src="images/1.jpg" alt=""/>  <**div** style="position: absolute;bottom: 20px;left: 20px;">说明</**div**> </**div**> |
|  |  |
| 使用absolute定位估计是大部分页面重构人员的首选。但是，绝对定位之“绝对”一词本身就意味着其在扩展性方面的表现受限。（外层元素relative，内层元素absolute，top left需要严格计算）  如果我们要求更高，可以使用CSS代码量更少，扩展性更强的margin布局。 | |
| 更多margin使用案例：  CSS 相对/绝对(relative/absolute)定位系列（二）  <http://www.zhangxinxu.com/wordpress/2010/12/css-%E7%9B%B8%E5%AF%B9%E7%BB%9D%E5%AF%B9relativeabsolute%E5%AE%9A%E4%BD%8D%E7%B3%BB%E5%88%97%EF%BC%88%E4%BA%8C%EF%BC%89/>  个人认为无可比性。 | |

将absolute元素置于body

|  |  |
| --- | --- |
|  | 在采用absolute进行布局时，一般会将其深深地嵌入到dom中，通过left top relative display进行控制；如下拉框实现；  越深的DOM元素，造成了回流越强；因为父标签需要relative限制，增加了CSS代码的消耗量；隐藏元素头部加载，延迟了页面的呈现速度；每个下拉几乎都要重新定位，其重用性受限。  如果是我，一定会把这些隐藏的绝对定位元素放在在body标签内部，且最底部加载，以提高页面的呈现速度，甚至根本就不加载。此做法就是mtime时光网下拉导航的做法，是推荐的做法。绝对元素的定位不是通过当前触发元素的relative限制，而是在body大环境中，直接通过触发元素的偏移值设定位置，是最直接最高效的做法。  在body标签下自由驰骋，这才是绝对定位元素真正应该呆的地方。真正有才华的元素应该放在更大的舞台上，限制在蹩脚的角落里注定要埋没的。我之前写过一个名为”[powerFloat](http://www.zhangxinxu.com/wordpress/?p=1328)“的万能浮动插件，就是希望通过降低使用的学习成本，让大家都把绝对定位元素解放到body标签下。 |

|  |  |
| --- | --- |
|  | CSS 相对/绝对(relative/absolute)定位系列（三）  <http://www.zhangxinxu.com/wordpress/2011/03/css-%E7%9B%B8%E5%AF%B9%E7%BB%9D%E5%AF%B9relativeabsolute%E5%AE%9A%E4%BD%8D%E7%B3%BB%E5%88%97%EF%BC%88%E4%B8%89%EF%BC%89/>  一、absolute正业之元素隐藏  二、absolute与等高布局  三、absolute属性与IE6/IE7之间的误会 |

absolute与元素隐藏

|  |  |
| --- | --- |
| 说明 | 元素隐藏与显示是我们在页面制作与交互效果实现中非常常见的，如果您只是使用display:none与display:block/inline来实现DOM元素的显隐控制，那你就out了。就元素的显示与隐藏实现，使用display在有些时候算是比较糟糕的方法了。  控制元素显隐的方法很多，但是本文不是讲元素显隐控制的，所以，只讲与absolute相关的一些方法。absolute属性相关的隐藏方法，我知道的有三种，分别如下：  .hidden{  position:absolute;  top:-9999em;  }  .hidden{  position:absolute;  visibility:hidden;  }  .hidden{  position:absolute;  clip: rect(1px 1px 1px 1px); /\* IE6, IE7 \*/  clip: rect(1px, 1px, 1px, 1px);  }  使用absolute属性控制DOM元素的显隐有三个关键点：页面可用性，回流与渲染，配合JavaScript的控制。 |
| 1、可用性隐藏 | 所谓可用性隐藏，就是兼顾屏幕阅读器这类互联网阅读辅助设备的隐藏方式。Yahoo! 可用性实验室成员[Ted Drake](http://yaccessibilityblog.com/library/author/ted/)就不同隐藏方法下屏幕阅读器的可用性问题作为测试，结果发现下面两种隐藏方式屏幕阅读器是读不了的。  .completelyhidden {  display:none;  }  .visibilityhidden {  visibility:hidden;  }  You don’t want to show those hidden panels to any user. Use display:none for the hidden panels.  Screen readers will also ignore sections with visibility:hidden.  所以，从可用性角度而言，像“选项卡内容”，“更多收起展开”这类元素隐藏与显示就不推荐使用display:none， 或者是position:absolute + visibility:hidden。  隐藏内容其实都是有用的信息，对于像盲人这类需要借助屏幕阅读器的用户无法知道这些信息了。 |
| 如果希望隐藏内容能够被辅助阅读设备识别，就不能使用display:none以及visibility:hidden隐藏元素。可以使用模拟隐藏的隐藏方法，又称可用性隐藏。就是下面两种隐藏方法。  .hidden{  position:absolute;  top:-9999em;  }  .hidden{  position:absolute;  clip: rect(1px 1px 1px 1px); /\* IE6, IE7 \*/  clip: rect(1px, 1px, 1px, 1px);  }  但是，如果你是希望完全隐藏的，那就可以使用display:none或visibility:hidden。  额外说明：如果隐藏元素含有链接元素或是可获得焦点的控件元素，但是又是使用的可用性隐藏。这些隐藏的链接与控件也是可以响应键盘焦点Tab切换的，但是这会让键盘使用用户产生不解与疑惑的。所以，从某种意义说，某些情况下，要兼顾屏幕阅读器用户和键盘用户有时候是不可兼得的。 |
| 2、回流与渲染 | 早先时候我曾翻译过两篇关于回流与重绘的文章，“[最小化浏览器中的回流(reflow)](http://www.zhangxinxu.com/wordpress/?p=311)”以及“[回流与重绘：CSS性能让JavaScript变慢？](http://www.zhangxinxu.com/wordpress/?p=600)”。  我自己是没测过。不过根据上面这两篇文章的说法，以及一位口碑前端前辈的说法，使用absolute隐藏于显示元素是会产生重绘而不会产生强烈的回流。而使用display:none不仅会重绘，还会产生回流，DOM影响范围越广，回流越强烈。所以，就JavaScript交互的呈现性能上来讲，使用absolute隐藏是要优于display相关隐藏的。 |
| 3、配合JavaScript的控制 | 说到元素的显示与隐藏，免不了与JavaScript的交互。例如display相关的隐藏于显示，就是display:block/inline/inline-block/...与display:none。 要让元素隐藏，很简单，直接：dom.style.display = "none";  但是，如果要显示隐藏的元素，咋办呢？因为不同的标签所处的display水平是不一样的，于是，我们很难有一个简单的统一的显示方法。  例如，下面的代码可能使用于div, p标签，但是对于span等inline水平的元素，可能就会嗝屁了（原本单行显示结果换行）。  dom.style.display = "block";  况且，随着浏览器的不断进步，以后类似于display:table-cell，display:list-item会越来越多的使用。再想通过display实现通用的显隐方法难度又会增大些。  这就是使用display属性控制元素显隐的局限性。顺带一提的是jQuery的显隐方法show()/hide()/toggle()就是基于display的，其会存储元素先前的display属性值，于是元素再显示的时候就可以准确地显示出之前的display值了。 |
| 而使用绝对定位实现的一些元素隐藏方法的控制就相对简单很多的。  例如：position:absolute + visibility:hidden方法，当我们要让元素（原本非绝对定位元素）显示的时候，我们需要设置：  dom.style.position = "static";  dom.style.visibility = "visible";  而类似的position:absolute + top:-999em方法，当我们要让元素（原本非绝对定位元素）显示的时候，我们只需要设置：  dom.style.position = "static";  而无需担心原本标签的是inline水平还是block水平。所以，就显隐的JavaScript控制上来讲，absolute相关方法要比display略胜一筹。 |
|  | 结合上面三点讨论，我们可以看出，当前占据主流的display:block/none控制元素显示与隐藏的方法其实是诸多方面有弊端的方法，有拿着鸡毛当令箭的意味。实际上，这种活（元素显隐）交给absolute属性更合适，控制元素显示与隐藏才是absolute属性的正业所在。 |

absolute与左右内容不同高，背景相同布局

|  |  |
| --- | --- |
| absolute实现等高布局 | 拿简单的两栏布局举例，左栏与右栏有不同的背景色，且中间隔边框线分隔，如何实现？  因为随着内容的不同，有可能左侧栏高度较高，也有可能是右侧栏高度较高。所以，要实现无缝的填色，定高不行不通的，置高度不理显然也不行，此时解决方法就是让左右两栏等高。  我较早的时候写过一篇名为“[纯CSS实现侧边栏/分栏高度自动相等](http://www.zhangxinxu.com/wordpress/?p=694)”的小tip，其实现原理如下：  margin-bottom:-3000px;  padding-bottom:3000px;  后来在“[我所知道的几种display:table-cell的应用](http://www.zhangxinxu.com/wordpress/?p=1187)”一文中也提过使用display:table-cell实现等高布局。 |
| position:absolute的元素无宽度，无高度。  正好，我们可以利用该特性来实现等高布局所需要的效果——如等高的背景色、边框效果等  > 左边区域absolute |
| <**style**>  .equal\_height {  width: 100%;  height: 999em;  position: absolute;  left: 0;  top: 0;  }  .out\_box {  width: 66%;  margin-left: auto;  margin-right: auto;  border: 1px solid #ccc;  background-color: #f5f5f5;  overflow: hidden;  position: relative;  }  .left\_box {  width: 25%;  float: left;  position: relative;  }  .right\_box {  width: 74.6%;  float: right;  }  .left\_box\_bg {  background-color: #fff;  border-right: 1px solid #ccc;  }  .left\_con {  padding: 1em;  position: relative;  z-index: 1;  }  .right\_con {  padding: 1em;  }  .out\_box **img** {  display: block;  }  .btn\_box {  width: 66%;  margin: 1em auto 3em;  } </**style**> <**div** class="out\_box">  <**div** class="left\_box">  <**div** class="equal\_height left\_box\_bg"></**div**>  <**div** id="leftCon" class="left\_con">  <**img** src="http://image.zhangxinxu.com/image/study/s/s128/mm2.jpg"/>  </**div**>  </**div**>  <**div** class="right\_box">  <**div** id="rightCon" class="right\_con">  <**img** id="rightImage" src="http://image.zhangxinxu.com/image/study/s/s128/mm5.jpg"/>  </**div**>  </**div**> </**div**> <**div** class="btn\_box">  <**button** id="btnLeftSide">左侧栏高度增加</**button**>  <**button** id="btnRightSide">右侧栏高度增加</**button**> </**div**>  <**script**>  $("#btnLeftSide").bind("click", **function** () {  $("#leftCon").append('<img src="http://image.zhangxinxu.com/image/study/s/s128/mm2.jpg" />');  });  $("#btnRightSide").bind("click", **function** () {  $("#rightCon").append('<img src="http://image.zhangxinxu.com/image/study/s/s128/mm5.jpg" />');  }); </**script**> |
| 其中，实现等高效果的核心CSS代码如下：  .equal\_height{width:100%; height:999em; position:absolute; left:0; top:0;}  同时，满足以下一些条件：  高度999em的绝对定位层位于侧栏容器内，侧栏position为relative  该栏实际元素内容用一个与absolute绝对定位层为兄弟关系的标签层包裹，position为relative，z-index值1或其他  左右栏的父标签需设置overflow:hidden，同时为了兼容IE6/7，需设置position为relative  以上条件对应下图标注：  绝对定位等高布局实现条件标注 张鑫旭-鑫空间-鑫生活  原理很简单：由于绝对定位元素无高度的特性无宽度的特性，我们可以伪造一个高度足够高的绝对定位层（设置背景色，边框等属性），同时设置父标签溢出隐藏，那么其多出来的高度酒不会显示了，也就实现了看上去的等高布局效果了。具体细节可参见[demo页面](http://www.zhangxinxu.com/study/201103/absolute-equal-height-layout.html" \t "http://www.zhangxinxu.com/wordpress/2011/03/css-%e7%9b%b8%e5%af%b9%e7%bb%9d%e5%af%b9relativeabsolute%e5%ae%9a%e4%bd%8d%e7%b3%bb%e5%88%97%ef%bc%88%e4%b8%89%ef%bc%89/_blank)中的代码展示，相信很好理解的。 |

absolute对前后元素的影响，以及在ie6/7中的不同表现

|  |  |  |
| --- | --- | --- |
| 前元素abs化，遮蔽原图层 | <**style**>  .box {  width: 300px;  }  .abs {  position: absolute;  }  .ml140 {  margin-left: 140px;**/\*\*/** } </**style**> | > 当元素处在abs元素后面，将占据abs元素在基础图层中的位置；  > 若想从屏幕俯视页面时，上图层的abs元素不遮蔽基础图层中的元素，可通过margin对基础图层的元素进行偏移；  > 注意margin对line元素与block元素的不同效果 |
|  | <**div** class="box">  <**span** class="abs"><**img** src="http://image.zhangxinxu.com/image/study/s/s128/mm1.jpg"/></**span**>  <**div**>某豆友："昨天被车撞了，还没等司机下车，我就赶紧跑了，我这样做对吗？"。楼下众豆友："LZ真是捡了条命啊！！！！！！！！！！"</**div**> </**div**> |  |
| <**div** class="box">  <**span** class="abs"><**img** src="http://image.zhangxinxu.com/image/study/s/s128/mm1.jpg"/></**span**>  <**div** class="ml140">某豆友："昨天被车撞了，还没等司机下车，我就赶紧跑了，我这样做对吗？"。楼下众豆友："LZ真是捡了条命啊！！！！！！！！！！"</**div**> </**div**> |  |
|  | <**div** class="box"><**div** class="ml140">某豆友："昨天被车撞了，还没等司机下车，我就赶紧跑了，我这样做对吗？"。楼下众豆友："LZ真是捡了条命啊！！！！！！！！！！"</**div**> </**div**> | 可用作两列布局 |
| <**div** class="box">  <**span** class="ml140">某豆友："昨天被车撞了，还没等司机下车，我就赶紧跑了，我这样做对吗？"。楼下众豆友："LZ真是捡了条命啊！！！！！！！！！！"</**span**> </**div**> | 可用作段落缩进 |
| +[IE6/7下margin与absolute元素重叠](http://www.zhangxinxu.com/study/201104/margin-absolute-overlap-2.html" \t "http://www.zhangxinxu.com/wordpress/2011/03/css-%e7%9b%b8%e5%af%b9%e7%bb%9d%e5%af%b9relativeabsolute%e5%ae%9a%e4%bd%8d%e7%b3%bb%e5%88%97%ef%bc%88%e4%b8%89%ef%bc%89/_blank) | IE6/IE7下的这个margin定位元素绝对定位元素会重叠；  此问题出现的原因与下面浮动与绝对定位元素重叠有着某些类似的原因，因为问题的出现都与绝对定位元素所在的标签水平有关：上述demo中，absolute属性所在的标签是div标签，属于block水平的元素。如下截图：  使用的div标签 张鑫旭-鑫空间-鑫生活  要是我们把这里的block水平的div元素修改成inline水平的span标签，则重叠的问题就没有了。如下： |  |
| <**div** class="box">  <**span** class="abs">  <**img** src="http://image.zhangxinxu.com/image/study/s/s128/mm1.jpg"/>  </**span**>  <**div** class="ml140">某豆友："昨天被车撞了，还没等司机下车，我就赶紧跑了，我这样做对吗？"。楼下众豆友："LZ真是捡了条命啊！！！！！！！！！！"</**div**> </**div**> | span标签下的重叠问题的修复 张鑫旭-鑫空间-鑫生活 |
| float元素与absolute元素（line/block） | <**img** class="fl" src="images/1.jpg"/> <**img** class="fl" src="images/2.jpg"/> <**div** class="">  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、 </**div**> |  |
| <**img** class="fl" src="images/1.jpg"/> <**img** class="fl" src="images/2.jpg"/> <**div** class="abs">  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、 </**div**> |  |
| <**img** class="fl" src="images/1.jpg"/> <**img** class="fl" src="images/2.jpg"/> <**span** class="">  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、 </**span**> |  |
| <**img** class="fl" src="images/1.jpg"/> <**img** class="fl" src="images/2.jpg"/> <**span** class="abs">  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、  无上甚深微妙法、 百千万劫难遭遇、 我今见闻得受持、 愿解如来真实义、 </**span**> |  |
| +ie6 7异常 | 前面一个标签是浮动元素，后面的是block水平的绝对定位元素，结果IE8+，以及现代浏览器文字与图片重叠；但是IE6/IE7浏览器确是并排显示的。  <**style**>  .box {  overflow: hidden;  \_zoom: 1;  }  .l {  float: left;  }  .abs {  position: absolute;  } </**style**> <**div** class="box">  <**img** class="l" src="http://image.zhangxinxu.com/image/study/s/s128/mm1.jpg"/>  <**div** class="abs">呦喝，这不是张含韵小姐吗？</**div**> </**div**>  如下截图示意：  IE8下文字与图片重叠 张鑫旭-鑫空间-鑫生活 IE6下图文并排 张鑫旭-鑫空间-鑫生活  您可以狠狠地点击这里：[浮动元素绝对定位元素重叠demo](http://www.zhangxinxu.com/study/201103/absolute-float-mistake.html" \t "http://www.zhangxinxu.com/wordpress/2011/03/css-%e7%9b%b8%e5%af%b9%e7%bb%9d%e5%af%b9relativeabsolute%e5%ae%9a%e4%bd%8d%e7%b3%bb%e5%88%97%ef%bc%88%e4%b8%89%ef%bc%89/_blank)  为何现代浏览器以及IE8+浏览器下浮动的图片与绝对定位的文字会重叠，而IE6/7却是并列显示？这是由于IE6/IE7浏览器将inline水平标签元素和block水平的标签元素没有加以区分一视同仁渲染了。我在前面已经多次提到，应用了绝对定位属性的元素具有包裹性，等同于没有高度与宽度的inline-block元素。  上面斜体加粗的这个结论实际上说得不够严谨，在IE6/IE7浏览器下，上面的话是没错的；在所有浏览器下，对于inline水平的元素，上面的话也是没错的；但是在现代浏览器下，对于block水平的元素，上面的结论就有商榷之处。实际上，按照正确的绝对定位渲染，像div, p这类block水平标签并未完全inline-block化。inline-block化的元素有三大特性：包裹性；高宽可定义；图文混排。然而，div, p这类标签应用了position:absolute后，在非IE6/7浏览器下，只有包裹性和高宽可定义这两个特性，但并不支持图片混排，也就是与图片文字在一起的时候会换行。  下面是举例字，验证上面的结论。首先是这么句话：“对于inline水平的元素，上面的话也是没错的”。这句话的意思其实是，如果是inline水平的元素，上面的那个示例就不会有兼容性问题了，于是我们把应用了abs类名的div标签改成span，如下HTML代码：  <div class="box">  <img class="l" data-src="http://image.zhangxinxu.com/image/study/s/s128/mm1.jpg" />  <span class="abs">呦喝，这不是张含韵小姐吗？</span>  </div>  结果如下图：  IE8下浮动元素与绝对定位元素并排显示 张鑫旭-鑫空间-鑫生活  不仅IE8浏览器，Firefox/Chrome等先前重叠的现在都并排显示了。  您可以狠狠地点击这里：[浮动与inline水平绝对定位元素不重叠demo](http://www.zhangxinxu.com/study/201103/absolute-float-mistake-span.html" \t "http://www.zhangxinxu.com/wordpress/2011/03/css-%e7%9b%b8%e5%af%b9%e7%bb%9d%e5%af%b9relativeabsolute%e5%ae%9a%e4%bd%8d%e7%b3%bb%e5%88%97%ef%bc%88%e4%b8%89%ef%bc%89/_blank)  下面再来验证这个结论：“现代浏览器下block水平元素absolute化后不支持图片混排”。也是很简单的，我们可以把最上面重叠的那个例子的图片的浮动属性干掉，也就是如下的HTML代码：  <div class="box">  <img data-src="http://image.zhangxinxu.com/image/study/s/s128/mm1.jpg" />  <div class="abs">呦喝，这不是张含韵小姐吗？</div>  </div>  结果如下图： 而在IE6/IE7浏览器下，依旧是并排显示滴，如下图：  IE8下绝对定位元素与图片换行 张鑫旭-鑫空间-鑫生活IE6/IE7下依旧并排显示 张鑫旭-鑫空间-鑫生活  OK，现在应该很好理解最上面为何在现代浏览器下图片文字重叠而IE6/IE7下并排显示了。  在“[CSS float浮动的深入研究、详解及拓展](http://www.zhangxinxu.com/wordpress/?p=583)”系列中多次阐述了浮动元素的“无高度”特性，所以，当图片应用了float:left属性后，图片所占据的高度丢失，于是，原本换行显示在下面的文字就提上去了，于是就形成了重叠，如下图标示：  浮动与绝对定位重叠图示 张鑫旭-鑫空间-鑫生活 | |

absolute绝对定位的非绝对定位用法

|  |  |  |
| --- | --- | --- |
|  | absolute绝对定位的非绝对定位用法  <http://www.zhangxinxu.com/wordpress/2010/01/absolute%E7%BB%9D%E5%AF%B9%E5%AE%9A%E4%BD%8D%E7%9A%84%E9%9D%9E%E7%BB%9D%E5%AF%B9%E5%AE%9A%E4%BD%8D%E7%94%A8%E6%B3%95/>  一、absolute绝对定位的流行用法  二、absolute元素的margin定位  三、绝对定位元素的非绝对定位 | |
| abs特性 | 其一，无定位值的absolute元素可以使用margin定位；  absolute定位与margin定位其实是没有什么冲突的，无论absolute元素时候设置了left/top值，其margin属性值都是可以起作用的；absolute的top/left建立于元素整体盒，margin、padding都在此盒中；  其二，无定位值的absolute元素是个没有实际宽度的浮动元素。  没有设置定位值的absolute元素是个普通又不普通的元素，普通之处在于其依旧在DOM tree中，对margin等属性敏感；不普通在于其实际的高宽都丢失了。这非常类似于浮动(float)，浮动的本质就是“包裹与破坏”，破坏高度，浮动元素的实际占据高度为0（具体点[这里](http://www.zhangxinxu.com/wordpress/?p=583" \t "http://www.zhangxinxu.com/wordpress/2010/01/absolute%E7%BB%9D%E5%AF%B9%E5%AE%9A%E4%BD%8D%E7%9A%84%E9%9D%9E%E7%BB%9D%E5%AF%B9%E5%AE%9A%E4%BD%8D%E7%94%A8%E6%B3%95/_blank)）；而absolute元素（无定位值）也是“包裹与破坏”，只是其“破坏”比float更加凶猛，不仅实际的高度没有，连实际的宽度也没有。  说句不严谨的结论：绝对定位元素就是个比浮动元素更加变态的近亲。  我们可以使用这两个特性实现一些特别的定位效果。两个例子，一是文字投影效果，二是自适应布局。 | |
| 文字的阴影效果 | CSS3中有text-shadow属性可以实现文字阴影效果，但是IE对CSS3的支持不佳，所以我们需要想替他的方法实现，absolute+margin是个很好很简单的方法，见下面的效果图，截自Firefox3.5：    相关代码如下：  <**style**>  .zxx\_show {  padding: 20px;  color: yellow;  font-size: 14px;  }  .zxx\_text {  position: absolute;  margin: -5px 0 0 -5px;  color: red;  } </**style**> <**div** class="zxx\_show">  <**span** class="zxx\_text">这是一段用来测试的文字，看看是否有投影效果~~</**span**>  这是一段用来测试的文字，看看是否有投影效果~~ </**div**>  您可以狠狠地点击这里：[文字投影效果demo](http://www.zhangxinxu.com/study/201001/absolute-margin-text-shadow.html" \t "http://www.zhangxinxu.com/wordpress/2010/01/absolute%e7%bb%9d%e5%af%b9%e5%ae%9a%e4%bd%8d%e7%9a%84%e9%9d%9e%e7%bb%9d%e5%af%b9%e5%ae%9a%e4%bd%8d%e7%94%a8%e6%b3%95/_balnk)  绝对定位元素需要写在投影文字之前，因为absolute元素实际占据的高度宽度都是0，所以文字可以自然而然的在其下方显示，由于下方的文字颜色较浅，于是便形成投影效果。此方法很精妙，可惜有小小兼容性差异，IE6下投影方向是水平朝右，IE7是右上方向，需要hack修复。 | |
| 自适应 | 没有定位值的absolute元素是个更加变态的float元素，所以浮动元素干的某些事情absolute元素也能做，例如自适应布局。  使用的float自适应布局，其实将float属性替换成absolute，效果一样的;  absolute绝对定位的非绝对定位应用肯定还有其他，只要记住无定位值的absolute元素就是个连实际宽度也没有的float浮动元素就可以了，然后利用这个特性，可以实现一些意想不到的效果；  由于头像的宽度固定，所以对于描述标签，我们可以使用margin或是padding撑开一段距离，头像使用无定位值的absolute定位，这样就实现了头像与描述的自适应布局效果了。 | |
| <**style**>  .mini-img{  height: 100px;**/\*图片的宽高自适应变换可以实现某些特性\*/** }   .out {  height: 120px;**/\*width不定\*/** padding: 10px;**/\*float:left;\*//\*0->1\*/**  background: #f0f3f9;  }   .abs {  position: absolute;  }   .rightDiv {  padding-left: 100px;  } </**style**> <**div** class="out">  <**a** href="#" class="abs">  <**img** class="mini-img" src="images/1.jpg"/>  </**a**>  <**p** class="rightDiv">刘亦菲，1987年8月25日出生于湖北武汉，华语影视女演员、流行乐歌手，毕业于北京电影学院2002级表演系本科班。200...</**p**> </**div**> |  |
| position为absolute的元素如果没有设置left, top等值与left:0;top:0;的的效果为何不一样？相信您应该知道答案了，例如一个div中有个absolute属性元素，其没有left或是top值，其会像个普通的inline-block属性元素一样静静地呆在这个div里面，但是一旦设置了left:0;top:0;对不起，这个absolute元素立马变身，直接从DOM tree里面脱离，独立于文档流，**结果相对于最近的relative属性的祖先标签定位（如果没有，就body定位）**。由于我们平时使用absolute都离不开left，top之类的值，所以才会概念不清。  每个CSS属性都有一段故事，需要我们静下心来发现，体会。 | |

relative

|  |  |
| --- | --- |
|  |  |
|  | relative翻译成中文称相对定位，设置了这个属性后，元素会根据top，left，bottom，right进行偏移，关键点是它原本的空间仍然保留。  div.relative相对定位，并且left设置为20px，left设置为50px，其相对于父元素进行偏移，并且原本的空间也占据着，下面的元素并不会顶替上去。  一个元素被设置为abs绝对定位时，会脱离文档流，然后相对其包含块进行偏移。一般来说，我们会将一个元素设置为relative来作为absolute元素的包含块。  ===  包含块：  在详解CSS float属性中我们提到包含块这个概念。对于position属性也有包含块这个属性，它要分几种情况来讨论：  1.根元素的包含块，根元素一般是html元素，有些浏览器会使用body作为根元素，大多数浏览器，初始包含块是一个视窗大小的矩形  2.非根元素的包含块，如果该元素的position是relative或static，它的包含块是最近的块级框，表的单元格或行内块的内容边界  我们举例进行说明：   |  |  | | --- | --- | | <**style**>   **div** {  background: #0094ff;  width: 250px;  height: 100px;  }  .relative {  background: #ff6a00;  position: relative;  width: 200px;  height: 100px;  top: 20px;  left: 50px;  } </**style**> <**div**>  我是父级元素的内容  <**div** class="relative">  相对定位元素  </**div**> </**div**> | 红色框指相对定位元素的包含块 |   这是相对定位元素的包含块，为最近的块级框，表的单元格或行内块的内容边界，相对定位元素相对于其包含块进行偏移，我们可以简单理解为相对于其原来的位置进行偏移。  3.非根元素的包含块，如果该元素的position是absolute，则包含块为最近的position不是static的祖先元素。  简单来说，它的包含块会从父级元素一直向上查找，找到第一个position不是static的元素为止。  偏移属性  前面的例子中已经涉及到偏移属性，它指的是元素相对于其包含块的偏移，我们称其为偏移属性，分别是top，bottom，left，right，依次代表上下左右。他们的值可以是具体的数值，也可以是百分比。如果是百分比，top和bottom是相对于包含块高度的百分比，left和right是相对于宽度的百分比。它们也可以设置负值，即有可能将元素移动到包含块的外边。 |
|  | CSS 相对/绝对(relative/absolute)定位系列（四）  <http://www.zhangxinxu.com/wordpress/2011/08/css%E7%9B%B8%E5%AF%B9%E5%AE%9A%E4%BD%8Drelative%E7%BB%9D%E5%AF%B9%E5%AE%9A%E4%BD%8Dabsolute%E7%B3%BB%E5%88%97%EF%BC%88%E5%9B%9B%EF%BC%89/>  前三篇([1](http://www.zhangxinxu.com/wordpress/?p=1287), [2](http://www.zhangxinxu.com/wordpress/?p=1330), [3](http://www.zhangxinxu.com/wordpress/?p=1528))都是介绍的absolute属性，终于，轮到本文讲讲relative相对定位。  三、relative属性的特殊才能 |

relative针对absolute的常用方式

|  |  |  |
| --- | --- | --- |
| 通过外部元素relative限制内部absolute | 前文提到，魔鬼兄弟浮动因为小时候被巨雷袭击为折翼恶魔从小陨落人间，因为其某些建筑方便的才能就被人类用来布局建造房子。同时聪明的凡人们发现其兄弟absolute绝对定位也有布局建房子的能力。但是，absolute羽翼尚存，过于强大，比较难以驾驭，所以凡人们就大肆请relative属性使用限魔大法来约束管制absolute元素，以帮助人们建造合乎他们要求的房子。  那么relative定位是如何限制absolute定位的呢？  在没有relative的情况下，absolute添加left/top等属性变身后可以发生瞬移，例如：  { position: absolute; left: 20px; top: 20px; }  此时absolute所在元素就会迅速定位至距离浏览器左侧20像素，顶部20像素的地方，具体效果可参见下demo。  您可以狠狠地点击这里：[relative属性之限魔大法demo](http://www.zhangxinxu.com/study/201108/css-relative-skill-limit.html" \t "http://www.zhangxinxu.com/wordpress/2011/08/css%e7%9b%b8%e5%af%b9%e5%ae%9a%e4%bd%8drelative%e7%bb%9d%e5%af%b9%e5%ae%9a%e4%bd%8dabsolute%e7%b3%bb%e5%88%97%ef%bc%88%e5%9b%9b%ef%bc%89/_blank)  默认效果如下，图片在浏览器的左上角，即left/top的计算是相对于浏览器的。  但是，一旦图片的父标签应用position:relative之后（点击demo页中的按钮），“限魔大法”就开始生效了，absolute的瞬移能力被限制了，被限制在了凡间，小小的屋子里面。  俨然一个活生生的父亲管束孩子的例子！ | |
| 关于获焦点显示子节点的实现方式：  下拉显示的实现就是使用的绝对定位，且需要深深地嵌入到DOM中的；  将默认隐藏的绝对定位元素被relative限制在很深的DOM节点中，弊处相当多：首先增加了DOM的复杂度，不利于维护；越深的DOM元素，造成了回流越强；  因为父标签需要relative限制，增加了CSS代码的消耗量；  隐藏元素头部加载，延迟了页面的呈现速度；  每个下拉几乎都要重新定位，其重用性受限。  建议做法：  将隐藏的绝对定位元素放在在body标签内部，且最底部加载，以提高页面的呈现速度，甚至根本就不加载。  绝对元素的定位不是通过当前触发元素的relative限制，而是在body大环境中，直接通过触发元素的偏移值设定位置，是最直接最高效的做法。  在body标签下自由驰骋，这才是绝对定位元素真正应该呆的地方。真正有才华的元素应该放在更大的舞台上，限制在蹩脚的角落里注定要埋没的。我之前写过一个名为”[powerFloat](http://www.zhangxinxu.com/wordpress/?p=1328)“的万能浮动插件，就是希望通过降低使用的学习成本，让大家都把绝对定位元素解放到body标签下。 | |
| 写页面，如果元素间的牵连甚广，例如很多个内部元素的定位要依托于外部标签，则该外部标签就像是抚养了一大堆孩子的妈妈，有了责任，有了牵连，其自身行动就会限制，不能随意改动，不能把妈妈的身份扔掉一走了之。显然，该标签后期的可维护性就糟糕了！这就不难理解我所说的：最小化relative影响原则可以让你后期更从容地修改元素的样式，或是结构等。  我在[系列二](http://www.zhangxinxu.com/wordpress/?p=1330)中浓烟重墨地表述“absolute可以一个人战斗”的观点。  定位这东西，margin或float定位往往更具有适应性，还轮不到absolute；就算我们确实应该使用absolute绝对定位，在绝大多数情况下，我们是不需要多余同时又不利于维护，留下潜在隐患的relative属性的；就算我们要使用relative相对定位来限制absolute在小范围内的定位，也不应该把relative属性放在最外层的标签上。 | |
| demo | 举个例子说说“就算我们要使用relative相对定位”这种情况下，如何保证相对定位的最小化影响。  提问，对于下图这个效果，右下角的绿色叉叉你如何绝对定位？  我想大多数的人可能会采用下面这种实现方式，即设置框框div的position:relative，然后大大的绿色钩钩所在的标签在该div内部绝对定位，如下代码： | |
| <**style**>  .test {  width: 25em;  margin: 2em auto;  }  **/\*外层relative限制内层absolute\*/** .box {  padding: 2em;  border: 1px solid #beceeb;  border-radius: 2px;  background-color: #f0f3f9;  position: relative;  }  **/\*abs 定位\*/** .ok {  color: green;  font-size: 6em;  position: absolute;  right: -11px;  bottom: -.5em;  } </**style**> <**div** class="test">  <**div** class="box">  CSS relative相对定位的最小化影响原则  <**strong** class="ok">√</**strong**>  </**div**> </**div**> |  |
| 上面的结构可以说非常简单了。但是，我们需要意识到，文字的定位其实并不需要父标签position属性值为relative，这就不符合相对定位的最小化影响原则了。  下面来看看最小化影响原则下的例子：  您可以狠狠地点击这里：[遵循最小化影响原则实现demo](http://www.zhangxinxu.com/study/201108/css-relative-mini-effect-rule.html" \t "http://www.zhangxinxu.com/wordpress/2011/08/css%e7%9b%b8%e5%af%b9%e5%ae%9a%e4%bd%8drelative%e7%bb%9d%e5%af%b9%e5%ae%9a%e4%bd%8dabsolute%e7%b3%bb%e5%88%97%ef%bc%88%e5%9b%9b%ef%bc%89/_blank) | |
| <**style**>  .box {  width: 25em;  margin: 2em auto;  }  .content {  padding: 2em;  border: 1px solid #beceeb;  border-radius: 2px;  background-color: #f0f3f9;  }  .rel {  position: relative; **/\*由最外层切换至偏外层\*/** }  .ok {  color: green;  font-size: 6em;  position: absolute;  right: -10px;  top: -1em;  } </**style**> <**div** class="box">  <**div** class="content">CSS relative相对定位的最小化影响原则</**div**>  <**div** class="rel"><**strong** class="ok">√</**strong**></**div**> </**div**> |  |
|  | 将需要绝对定位的元素单独放在relative属性的标签下，于是，relative相对定位就不会影响任何其他元素，仅仅是其内部的绝对定位元素。于是，上面的文字内容div还是那个普普通通的文字内容div，以后要改动什么东西就可以放心大胆的改，而不需要担心扔掉那个属性或是布局变了，里面原来绝对定位的元素位置偏移掉了。  这种牺牲一个标签增强扩展性和易维护性的做法在“[CSS宽度分离准侧](http://www.zhangxinxu.com/wordpress/?p=1463)”中就有所展现。  定位这个东西，如果要使用相对定位属性或是绝对定位属性，我个人比较推荐如下组合：absolute+margin（左上角元素定位，作用于当前元素）、float+relative（右上角元素定位，作用于当前元素）和relative+absolute（右下角元素定位，直接父标签+当前定位元素）。  而目前web届大肆使用的外层div层设置relative属性，里面一些absolute元素定位的方法是不推荐的。 | |

relative其他特性

|  |  |
| --- | --- |
| 与 left | relative元素能够响应left设置，但是本身的位置还是保留的。当left的值设置的过大可以用来控制元素的显示隐藏，且不会影响前后元素的布局。  <**style**>  .rel {  position: relative;  } </**style**> <**img** src="http://image.zhangxinxu.com/image/study/s/s128/mm1.jpg"/> <**img** class="rel" src="http://image.zhangxinxu.com/image/study/s/s128/mm2.jpg"/> <**img** src="http://image.zhangxinxu.com/image/study/s/s128/mm3.jpg"/> <**p**>上面第二张图片position属性为relative</**p**> <**input** type="button" id="button" value="点击我-展示幻影瞬移"/> <**script**>  $("#button").click(**function** () {  $("img").css("left", "-999em");  }); </**script**> |
| 与 z-index | position:relative虽是凡夫肉体，但是却有只有神魔才有的“纵云梯”技能，亦称“垂直升空”的本事。借助z-index这把御剑，就能直接腾空9万里，越于其他凡人之上。且其纵云梯的技能和纯魔鬼血统的position:absolute的技能是平起平坐的。如果两者纵云梯的高度一致（z-index值一样），谁后发制人谁就是上面显示。  relative属性要借助z-index采能实现任意高度的纵云梯，但是，实际上，即使没有z-index这把御剑，在凡人之中，其纵云之气已经完全震慑周围。例如，凡人标签在margin负值下，后者会覆盖前者，如下图所示：  但是，普普通通的元素一旦应用了position:relative，其霸气就会外露出来，立马显示在其他元素之上。  <**style**>  **img** {  margin-right: -3em;  box-shadow: 0 0 5px rgba(0, 0, 0, .75);  } </**style**> <**img** id="target" src="http://image.zhangxinxu.com/image/study/s/s128/mm1.jpg"/> <**img** src="http://image.zhangxinxu.com/image/study/s/s128/mm2.jpg"/> <**img** src="http://image.zhangxinxu.com/image/study/s/s128/mm3.jpg"/> <**p**>  <**input** type="button" id="button" value="点击我第一张图片应用position:relative"/> </**p**> <**script**>  $("#button").click(**function** () {  $("#target").css("position", "relative");  }); </**script**>  如下图所示：  margin负值下后者覆盖前者 张鑫旭-鑫空间-鑫生活引用position:relative后霸气外露 张鑫旭-鑫空间-鑫生活 |
|  |