|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | http://www.w3cfuns.com/data/attachment/forum/201405/01/162609gf5x5qnb5ibinnkk.jpg  **用CSS3绘制小新.jpg** (45.7 KB, 下载次数: 1)  [下载附件](http://www.w3cfuns.com/forum.php?mod=attachment&aid=MTU2MTd8NTllZDZmODh8MTQxMDE0NjEzOHw1NDUyMzE4fDU2MDAwNTc%3D&nothumb=yes)  [保存到图库](javascript:;)  2014-5-1 16:26 上传    本文将讲述 CSS 中最核心的几个概念，包括：盒模型、position、float等。这些是 CSS 的基础，也是最常用的几个属性，它们之间看似独立却又相辅相成。为了掌握它们，有必要写出来探讨一下，如有错误欢迎指正。  **元素类型**  HTML 的元素可以分为两种：   * 块级元素（block level element） * 内联元素（inline element 有的人也叫它行内元素）     **两者的区别在于以下三点：**   1. 块级元素会独占一行（即无法与其他元素显示在同一行内，除非你显示修改元素的 display 属性），而内联元素则都会在一行内显示。 2. 块级元素可以设置 width、height 属性，而内联元素设置无效。 3. 块级元素的 width 默认为 100%，而内联元素则是根据其自身的内容或子元素来决定其宽度。     最常见块级元素应该是 <div> 吧，内联元素有 <span> <a> <img> 等等，完整的元素列表可以谷歌一下。    具体来说一下吧,   1. .example { 2. width: 100px; 3. height: 100px; 4. }   复制代码  我们为 <div> 设置上面的样式，是有效果的，因为其是块级元素，而对 <span> 设置上面的样式是没用的。要想让 <span> 也可以改变宽高，可以通过设置 display: block; 来达到效果。当 display 的值设为 block 时，元素将以块级形式呈现；当 display 值设为 inline 时，元素将以内联形式呈现。    若既想让元素在行内显示，又能设置宽高，可以设置：   1. display: inline-block;   复制代码  inline-block 在我看来就是让元素对外呈内联元素，可以和其他元素共处与一行内；对内则让元素呈块级元素，可改变其宽高。  HTML 代码是顺序执行的，一份无任何 CSS 样式的 HTML 代码最终呈现出的页面是根据元素出现的顺序和类型排列的。块级元素就从上到下排列，遇到内联元素则从左到右排列。这种无样式的情况下，元素的分布叫**普通流**，元素出现的位置应该叫**正常位置**（这是我瞎起的），同时**所有元素会在页面上占据一个空间**，空间大小由其盒模型决定。  **盒模型**  页面上显示的每个元素（包括内联元素）都可以看作一个盒子，即盒模型( box model )。请看Chrome DevTools 里的截图：    http://www.w3cfuns.com/data/attachment/forum/201405/01/163808eqzyqq9cunyspncv.jpg    可以显而易见的看出盒模型由 4 部分组成。从内到外分别是：   1. content -> padding -> border -> margin   复制代码  按理来说一个元素的宽度（高度以此类推）应该这样计算：   1. 总宽度 = margin-left + border-left + padding-left + width + padding-right + border-right + margin-right   复制代码  但是不同浏览器（你没有猜错，就是那个与众不同的浏览器）对宽度的诠释不一样。符合 W3C 标准的浏览器认为一个元素的宽度只等于其 content 的宽度，其余都要额外算。于是你规定一个元素：   1. .example { 2. width: 200px; 3. padding: 10px; 4. border: 5px solid #000; 5. margin: 20px; 6. }   复制代码  则他最终的宽度应为：  宽度 = width(200px) + padding(10px \* 2) + border(5px \* 2) + margin(20px \* 2) = 270px;  而在 IE（低于IE9） 下，最终宽度为：  宽度 = width(200px) + margin(20px \* 2) = 240px;  我个人觉得 IE 的更符合人类思维，毕竟 padding 叫内边距，边框算作额外的宽度也说不下去。W3C 最后为了解决这个问题，在 CSS3 中加了 box-sizing 这个属性。当我们设置 box-sizing: border-box; 时，border 和 padding 就被包含在了宽高之内，和 IE 之前的标准是一样的。所以，为了避免你同一份 css 在不同浏览器下表现不同，最好加上：   1. \*, \*:before, \*:after { 2. -moz-box-sizing: border-box; 3. -webkit-box-sizing: border-box; 4. box-sizing: border-box; 5. }   复制代码  这里还有两种特殊情况：   * 无宽度 —— 绝对定位（position: absolute;） 元素 * 无宽度 —— 浮动（float） 元素   它们在页面上的表现均**不占据空间**（脱离普通流，感觉像浮在页面上层一样，移动它们不影响其他元素的定位）。这就涉及到另外两个核心概念 position 和 float。  **position**  position 这个属性决定了元素将如何定位。它的值大概有以下五种：   |  |  | | --- | --- | | position 值 | 如何定位 | | static | position的**默认值**。元素将定位到它的正常位置（上文提到过），其实也就相当于没有定位。元素在页面上**占据**位置。**不能**使用 top right bottom left 移动元素位置。 | | relative | 相对定位，相对于元素的正常位置来进行定位。元素在页面**占据**位置。**可以**使用 top right bottom left 移动元素位置。 | | absolute | 绝对定位，相对于*最近一级的* **定位不是 static 的**父元素来进行定位。元素在页面**不占据**位置。 **可以**使用 top right bottom left 移动元素位置。 | | fixed | 绝对定位，相对于**浏览器窗口**来进行定位。其余和 absolute 一样，相当于一种特殊的 absolute。 | | inherit | 从父元素继承 position 属性的值。 |     每个网页都可以看成是由一层一层页面堆叠起来的，如下图所示。  http://www.w3cfuns.com/data/attachment/forum/201405/01/163829pm3ir22huj8819oi.gif    position 设置为 relative 的时候，元素依然在普通流中，位置是正常位置，你可以通过 left right 等移动元素。会影响其他元素的位置。    而当一个元素的 position 值为 absolute 或 fixed 的时候，会发生三件事：   1. 把该元素往 Z 轴方向移了一层，**元素脱离了普通流，所以不再占据原来那层的空间**，还会覆盖下层的元素。 2. 该元素将**变为块级元素**，相当于给该元素设置了 display: block;（给一个内联元素，如 <span> ，设置 absolute 之后发现它可以设置宽高了）。 3. 如果该元素是块级元素，元素的宽度由原来的 width: 100%（占据一行），变为了 auto。   由此观之，当 position 设置为 absolute 或 fixed，就没必要设置 display 为 block 了。而且如果你不想覆盖下层的元素，可以设置 z-index 值 达到效果。  **float**  float 顾名思义，就是把元素浮动，它的取值一共有四个：left right none inherit，光看名字就懂了，无需多言。  最初的 float 只是用来实现**文字环绕图片**的效果，仅此而已。而现在 float 的应用已不止这个，前辈们也是写了无数博文来深入浅出的讲解它。  经验分享：  [CSS浮动(float,clear)通俗讲解](http://www.w3cfuns.com/article-5599892-1-1.html) 篇幅不长，通俗易懂，可以看完这篇文章再回过头来看本文。    我就不班门弄斧写原理了，只说说 float 的几个要点就行了：   1. 只有左右浮动，没有上下浮动。 2. 元素设置 float 之后，它会**脱离普通流**（和 position: absolute; 一样），不再占据原来那层的空间，还会覆盖下一层的元素。 3. 浮动不会对该元素的上一个兄弟元素有任何影响。 4. 浮动之后，该元素的下一个兄弟元素会紧贴到该元素之前没有设置 float 的元素之后（很好理解，因为该元素脱离普通流了，或者说不在这一层了，所以它的下一个元素当然要补上它的位置）。 5. 如果该元素的下一个兄弟元素中有内联元素（通常是文字），则会围绕该元素显示，形成类似「文字围绕图片」的效果。（可参考[CSS float浮动的深入研究、详解及拓展(一)](http://www.zhangxinxu.com/wordpress/2010/01/css-float%E6%B5%AE%E5%8A%A8%E7%9A%84%E6%B7%B1%E5%85%A5%E7%A0%94%E7%A9%B6%E3%80%81%E8%AF%A6%E8%A7%A3%E5%8F%8A%E6%8B%93%E5%B1%95%E4%B8%80/)中的讲解） 6. 下一个兄弟元素如果也设置了同一方向的 float，则会紧随该元素之后显示。 7. 该元素将**变为块级元素**，相当于给该元素设置了 display: block;（和position: absolute; 一样）。     这里还有个东西，就是广为人知的——**清除浮动**。具体的方法五花八门，可以看这篇：[那些年我们一起清除过的浮动](http://www.w3cfuns.com/article-5596578-1-1.html)，我就不多说了。 | |