1. 层叠样式表(英文全称：Cascading Style Sheets)是一种用来表现HTML（标准通用标记语言的一个应用）或XML（标准通用标记语言的一个子集）等文件样式的计算机语言。CSS不仅可以静态地修饰网页，还可以配合各种脚本语言动态地对网页各元素进行格式化。

CSS 能够对网页中元素位置的排版进行像素级精确控制，支持几乎所有的字体字号样式，拥有对网页对象和模型样式编辑的能力。

1. DIV元素是用来为HTML（标准通用标记语言下的一个应用）文档内大块（block-level）的内容提供结构和背景的元素。<div> 标签定义 HTML 文档中的分隔（DIVision）或部分（section）。
2. 父元素与子元素

|  |
| --- |
| <div class="parent">      <div class="children1">          <div class="grandchildren"></div>      </div>      <div class="children2"></div>  </div> |

其中parent是children1、children2的父级元素

children1是grandchildren的父级元素

而parent是grandchildren的祖先元素(只要是上级关系都是祖先元素)

grandchildren为parent的子孙元素(后代元素)

一个元素的上一级即父元素，下一级为子元素，下N级(N>0)为后代元素

1. 关于margin

**定义和用法**

margin 简写属性在一个声明中设置所有外边距属性。该属性可以有 1 到 4 个值。

**说明**

这个简写属性设置一个元素所有外边距的宽度，或者设置各边上外边距的宽度。

块级元素的垂直相邻外边距会合并，而行内元素实际上不占上下外边距。行内元素的的左右外边距不会合并。同样地，浮动元素的外边距也不会合并。允许指定负的外边距值，不过使用时要小心。

**注释：**允许使用负值。

**例子 1**

margin:10px 5px 15px 20px;

* 上外边距是 10px
* 右外边距是 5px
* 下外边距是 15px
* 左外边距是 20px

**例子 2**

margin:10px 5px 15px;

* 上外边距是 10px
* 右外边距和左外边距是 5px
* 下外边距是 15px

**例子 3**

margin:10px 5px;

* 上外边距和下外边距是 10px
* 右外边距和左外边距是 5px

**例子 4**

margin:10px;

* 所有 4 个外边距都是 10px

|  |  |
| --- | --- |
| **默认值：** | 0 |
| **继承性：** | no |
| **版本：** | CSS1 |
| **JavaScript 语法：** | *object*.style.margin="10px 5px" |

**可能的值**

|  |  |
| --- | --- |
| **值** | **描述** |
| auto | 浏览器计算外边距。 |
| *length* | 规定以具体单位计的外边距值，比如像素、厘米等。默认值是 0px。 |
| *%* | 规定基于父元素的宽度的百分比的外边距。 |
| inherit | 规定应该从父元素继承外边距。 |

1. 行内元素和块级元素

**一、行内元素与块级元素**

|  |  |
| --- | --- |
| **块级元素列表** | |
| <address> | 定义地址 |
| <caption> | 定义表格标题 |
| <dd> | 定义列表中定义条目 |
| <div> | 定义文档中的分区或节 |
| <dl> | 定义列表 |

|  |
| --- |
|  |
| <dt> | 定义列表中的项目 |
| <fieldset> | 定义一个框架集 |
| <form> | 创建 HTML 表单 |
| <h1> | 定义最大的标题 |
| <h2> | 定义副标题 |
| <h3> | 定义标题 |
| <h4> | 定义标题 |
| <h5> | 定义标题 |
| <h6> | 定义最小的标题 |
| <hr> | 创建一条水平线 |
| <legend> | 元素为 fieldset 元素定义标题 |
| <li> | 标签定义列表项目 |
| <noframes> | 为那些不支持框架的浏览器显示文本，于 frameset 元素内部 |
| <noscript> | 定义在脚本未被执行时的替代内容 |
| <ol> | 定义有序列表 |
| <ul> | 定义无序列表 |
| <p> | 标签定义段落 |
| <pre> | 定义预格式化的文本 |
| <table> | 标签定义 HTML 表格 |
| <tbody> | 标签表格主体（正文） |
| <td> | 表格中的标准单元格 |
| <tfoot> | 定义表格的页脚（脚注或表注） |
| <th> | 定义表头单元格 |
| <thead> | 标签定义表格的表头 |
| <tr> | 定义表格中的行 |

|  |  |
| --- | --- |
| **行内元素列表** | |
| <a> | 标签可定义锚 |
| <abbr> | 表示一个缩写形式 |
| <acronym> | 定义只取首字母缩写 |
| <b> | 字体加粗 |
| <bdo> | 可覆盖默认的文本方向 |
| <big> | 大号字体加粗 |
| <br> | 换行 |
| <cite> | 引用进行定义 |
| <code> | 定义计算机代码文本 |
| <dfn> | 定义一个定义项目 |
| <em> | 定义为强调的内容 |
| <i> | 斜体文本效果 |
| <img> | 向网页中嵌入一幅图像 |
| <input> | 输入框 |
| <kbd> | 定义键盘文本 |
| <label> | 标签为 input 元素定义标注（标记） |
| <q> | 定义短的引用 |
| <samp> | 定义样本文本 |
| <select> | 创建单选或多选菜单 |
| <small> | 呈现小号字体效果 |
| <span> | 组合文档中的行内元素 |
| <strong> | 语气更强的强调的内容 |
| <sub> | 定义下标文本 |
| <sup> | 定义上标文本 |
| <textarea> | 多行的文本输入控件 |
| <tt> | 打字机或者等宽的文本效果 |
| <var> | 定义变量 |

|  |  |
| --- | --- |
| **可变元素素列表--可变元素为根据上下文语境决定该元素为块元素或者内联元素** | |
| <button> | 按钮 |
| <del> | 定义文档中已被删除的文本 |
| <iframe> | 创建包含另外一个文档的内联框架（即行内框架） |
| <ins> | 标签定义已经被插入文档中的文本 |
| <map> | 客户端图像映射（即热区） |
| <object> | object对象 |
| <script> | 客户端脚本 |

二、行内元素与块级函数的三个区别

1.行内元素与块级元素直观上的区别

    行内元素会在一条直线上排列，都是同一行的，水平方向排列

    块级元素各占据一行，垂直方向排列。块级元素从新行开始结束接着一个断行。

2.块级元素可以包含行内元素和块级元素。行内元素不能包含块级元素。

3.行内元素与块级元素属性的不同，主要是盒模型属性上

行内元素设置width无效，height无效(可以设置line-height)，margin上下无效，padding上下无效

三、行内元素转换为块级元素

        display:block (字面意思表现形式设为块级)

1. 元素居中

1.使用自动外边距实现居中  
　　CSS中首选的让元素水平居中的方法就是使用margin属性—将元素的margin-left和margin-right属性设置为auto即可。在实际使用中，我们可以为这些需要居中的元素创建一个起容器作用的div。需要特别注意的一点就是，必须为该容器指定宽度：    
　　div#container {

　　margin-left: auto;

　　margin-right: auto;

　　width: 168px; //宽度为168像素

　　}

　　在大多数主流浏览器中，这种方法都非常有效，即使是Windows平台上的IE6，在其标准兼容模式（compliance mode）下也能够正常显示。但不幸的是，在更低版本的IE中，这种设置却并不能实现居中效果。所以若想在实际项目中使用这种方法，那么就要确保用户的IE浏览器版本不低于6.0。   
　　尽管在支持上不尽如人意，但大多数设计师都建议尽可能地使用这种方法。该方法也被认为是在各种用CSS实现元素水平居中方法中最正确、最合理的一种。

2.使用text-align实现居中  
　　另一种实现元素居中的方法是使用text-align属性，将该属性值设置为center并应用到body元素上即可。这种做法是彻头彻尾的hack（CSS hack是通过在[CSS样式](http://www.w3cschool.cn/codecamp/style-the-html-body-element.html)中加入一些特殊的符号，让不同的浏览器识别不同的符号（什么样的浏览器识别什么样的符号是有标准的，CSS hack就是让你记住这个标准），以达到应用不同的[CSS](http://www.w3cschool.cn/css)样式的目的），但它却能兼容大多数浏览器，所以在某些情况下也自然必不可少。  
　　之所以说它是hack，是因为这种方法并没有将文本属性应用到文本上，而是应用到了作为容器的元素上。这也给我们带来了额外的工作。在创建好布局必须的div之后，我们要按照如下代码为body应用text-align属性：

　　body{

　　text-align:center;

　　}

　　之后会出现什么问题吗？body的所有子孙元素都会被居中显示。  
　　因此，我们就需要用再写一条规则，让其中的文本回到默认的居左对齐：

　　p {  
　　text-align:left;  
　　}

　　可以想象这条附加的规则将带来一些不便。另外，真正完全遵循标准的浏览器并不会改变容器的位置，而只会让其中的文字居中显示。

关于text-align：text-align 属性规定元素中的文本的水平对齐方式。

该属性通过指定行框与哪个点对齐，从而设置块级元素内文本的水平对齐方式。通过允许用户代理调整行内容中字母和字之间的间隔，可以支持值 justify；不同用户代理可能会得到不同的结果。

　3.组合使用自动外边距和文本对齐  
　　因为文本对齐居中方式有着良好的向下兼容性，且自动外边距方式也被大多数当代浏览器支持，所以很多设计师将二者组合起来使用，以期让居中效果得到最大限度的跨浏览器支持：

　　body {  
　　text-align: center;  
　　}  
　　#container {  
　　margin-left: auto;  
　　margin-right: auto;  
　　border: 1px solid red;  
　　width: 168px;  
　　text-align: left  
　　}  
 vertical-align:middle实现行内元素的水平垂直居中对齐

　　可是这始终是个hack，无论如何也算不上完美。我们还是需要为居中容器中的文本编写附加的规则，但至少在各个浏览器中看起来都不错。  
　　4.负外边距解决方案（水平，垂直都可以用这个方法）  
　　负外边距解决方案远不是仅仅为元素添加负外边距这么简单。这种方法需要同时使用绝对定位和负外边距两种技巧。  
　　下面是该方案的具体实现方法。首先，创建一个包含居中元素的容器，然后将其绝对定位于相对页面左边边缘50%的位置。这样，该容器的左外边距将从页面50%宽度的位置开始算起。  
　　然后，将容器的左外边距值设置为负的容器宽度的一半。这样即可将该容器固定在页面水平方向的中点。

#container {  
　　background: #ffc url(mid.jpg) repeat-y center;  
　　position: absolute;  
　　left: 50%;  
　　width: 760px;  
　　margin-left: -380px;  
　　}margin-top也一样