认识CSS，为网页添加样式

CSS全称：Cascading Style Sheets 层叠样式表主要用于定义HTML内容在浏览器内显示的样式，如文字大小、颜色、加粗等。

实例代码：

```
<style type="text/css">
  p{
    font-size:12px;
    color:red;
    font-weight:bold;
  }
</style>
```

`style`嵌套在`head`标签中，`style`内嵌css代码，`p`的位置代表选择器，表示网页内要添加样式的内容；`{ }`内表示声明，由属性和值组成，以冒号分割，多条声明以分号间隔。

使用**css**代码的优势：

通过定义某个样式，可以让不同网页位置的文字有同一的字体、字号或着色等；

CSS的注释

css代码的注释：`/*注释内容*/`；

实例代码：

```
/*设置段落默认格式*/
p{
  font-size:12px;
  color:red;
}
```

注：多个声明之间由分号分隔，每个声明最好独占一行，便于阅读、修改。

CSS样式的基本知识

css代码根据插入位置的不同有三种型式：内联式、嵌入式和外部式。

内联式**css**样式表

将css代码直接写在现有的html标签中。

实例代码：

```
<p style="color:red; font-size=14px">这里是红色的文字</p>
```

注：将css代码写在开始标签的后面，不能写在结束标签中；css代码写在`""`中间，多条声明用`;`隔开。

嵌入式**css**样式表

由于内联式css代码必须写在html标签内，如果有多个内容需要做同样的样式，需要在每个内容的html标签中写入相同的css代码。可以使用嵌入式**css**代码来解决这个问题，将css样式代码写在`head`标签中的`<style type="text/css">···</style>`之间。

实例代码：

```
<style type="text/css">
    span{
  color:red;
  font-size:13px;
}
</style>
```

注：嵌入式css代码必须写在`<style>···</style>`之间，并且在`head`标签中。

外部式**css**样式表

简称外联式，将css代码写在一个单独的外部文件中，以`.css`为扩展名，在`head`标签内用`<link />`标签将css样式文件链接到html文件内。

实例代码：

```
<link href="base.css" rel="stylesheet" type="text/css" />
```

注：

1. css样式文件以有意义的单词命名；
2. `rel="stylesheet" type="text/css"`为固定写法，不能修改；
3. `link`标签一般写在`head`标签内，但是不能写在`style`标签内。

优先级

`内联式 > 嵌入式 > 外部式`:

`嵌入式 > 外部式`的前提：嵌入式css样式的位置在外部式的后面。

```
<link href="base.css" rel="stylesheet" type="text/css" />  /*在style标签之前*/<style type="text/css">span{  color:red;}</style>
```

总结：就近原则，css样式代码离被设置的元素越近，优先级越高。

`内联式 > 嵌入式 > 外部式`成立的大前提：内联式、嵌入式、外部式样式表中css样式是在权值相同的情况下。

CSS选择器

选择器定义

每一条css样式定义由两个部分组成，选择器和样式。

```
选择器{
  样式;
}
```

选择器：`{ }`之前的部分，指明了`{  }`中的样式作用的对象（网页的相应标签元素）。

标签选择器

指html代码中的标签，例如`body`、`h1`、`p`、`img`等。

实例代码：

```
p{
  font-size:12px;
  line-height:1.6em;
}
```

设置`p`便签为12px字号，行间距1.6em。

类选择器

实例代码：

```
.类选择器名称{css样式代码;}
```

注：英文圆点开头；类选择器名称任意起名，不要中文。

使用方法：

第一步：使用合适的标签将要修饰的内容标记；

```
<span>记忆的留念</span>
```

第二步：使用`class="类选择器名称"`为标签设置一个类；

```
<span class="fox">记忆的留念</span>
```

第三不：设置类选择器css样式；

```
.stress{color: red;}  /*放在style标签内，注意stress前面的英文圆点号*/
```

ID选择器

在很多方面`id选择器`类似`类选择器`，有以下区别：

1. 为标签设置`id="id名称"`而非`class="class名称"`。
2. id选择符前面的是#号，而非英文圆点号。

实例代码：

```
<span id="myCLass">公开课</span>
```

在`style`标签内的css代码：

```
<style type="text/css">
    #myClass{
  color:red;
}
</style>
```

注：**id**绝对不能重复使用

类选择器与**ID**选择器区别

相同点：可以应用于任何元素；

不同点：

1. 在一个HTML文档中，id选择器只能在文档中使用一次。类选择器可以多次使用。

    正确代码：

```
        <p>三年级时，我还是一个<span class="stress">胆小如鼠</span>的小女孩，上课从来不敢回答老师提出的问题，生怕回答错了老师会批评我。就一直没有这个<span class="stress">勇气</span>来回答老师提出的问题。</p>
```

    错误代码：

```
        <p>三年级时，我还是一个<span id="stress">胆小如鼠</span>的小女孩，上课从来不敢回答老师提出的问题，生怕回答错了老师会批评我。就一直没有这个<span id="stress">勇气</span>来回答老师提出的问题。</p>
```

2. 类选择器可以使用词列表方法为一个元素同时设置多个样式。id选择器不能使用ID词列表。

    正确代码：

```
.stress{
    color:red;
}
.bigsize{
    font-size:25px;
}
<p>到了<span class="stress bigsize">三年级</span>下学期时，我们班上了一节公开课...</p>
```

    作用：为“三年级”三个文字设置文本颜色为红色，并且字号为25px

    错误代码：

```
#stressid{
    color:red;
}
#bigsizeid{
    font-size:25px;
}


<p>到了<span id="stressid bigsizeid">三年级</span>下学期
```

```
    时，我们班上了一节公开课...</p>
```

没有作用。

子选择器

用于选定指定标签元素的第一代子标签，使用`>`符号。

```
1  .food>li{border:1px solid red;}
```

使`class="food"`下的子元素`li`加入红色实线边框。

包含（后代）选择器

相较于子选择器只作用于选择标签元素的第一（直接）后代，包含选择器作用于选择标签的所有后代元素。用空格取代子选择器的`>`。

```
1  .first  span{color:red;}
```

总结：`>`代表自选择器，作用与元素的第一代后代；空格作用于元素的所有后代。

通用选择器

功能最强大的选择器，使用`*`指定，作用是匹配html中所有元素标签。

```
1  *{
2    color：red；
3  }
```

作用：所有元素的内容颜色都改为红色。

伪类选择符

伪类选择符允许给html不存在的标签（标签的某种状态）设置样式。比如html中一个标签元素的鼠标滑过的状态来设置字体或颜色。

```
1  a:hover{color:red;}
```

注：使`a`标签中的内容在鼠标滑过的状态下，字体颜色变为红色。

`:hover`可以放在任何标签上，只是兼容性不是太好，最常用的还是`a:hover`。

分组选择符

需要为html中多个标签设置同一个样式时，可以使用分组选择符`,`，在标签名之间用逗号分隔。

```
h1,span,h2{
  color:blue;
}
```

CSS的继承、层叠和特殊性

继承

css的某些样式具有继承性，允许样式不经应用于某个特定的html标签元素，而且应用于其后代。

```
p{color:red;}

<p>三年级时，我还是一个<span>胆小如鼠</span>的小女孩。</p>
```

注：颜色样式，红色不仅应用于`p`标签，还应用于`p`标签的所有元素（`span`等）。

但某些样式并不具有继承性：

```
p{border:1px solid red;}

<p>三年级时，我还是一个<span>胆小如鼠</span>的小女孩。</p>
```

注：border样式不具有继承性。

特殊性

有时会给同一个元素通过不同的方式（内联式、类选择器、id选择器等）设置了不同的属性，浏览器根据权值来判断使用权值高的css样式。

权值的规则：标签的权值为1；类选择符的权值为10；id选择符的权值为100。

```
p{color:red;} /*权值为1*/
p span{color:green;} /*权值为1+1=2*/
.warning{color:white;} /*权值为10*/
p span.warning{color:purple;} /*权值为1+1+10=12*/
#footer .note p{color:yellow;} /*权值为100+10+1=111*/
```

注：继承也有权值，值很小，可以理解为继承的权值最低。

层叠

层叠用来解决在html文件中对于同一个元素可以有多个css样式存在并且这多个css样式具有相同的权值的问题。

当有相同权值的样式存在时，根据这些css样式的前后顺序来解决，处于最后面的**css**样式会被应用（就近原则）。

```
p{color:red;}
p{color:green;}
<p class="first">三年级时，我还是一个<span>胆小如鼠</span>的小女孩。</p>
```

注：最后文本会被设置为绿色，理解为后面的样式会覆盖前面的。按照就近原则：可以理解`内联式>嵌入式>外部式`，并且要求`link`标签放在`style`之前。

重要新（！**important**）

在做网页时需要为某些样式设置最高权值，使用`!important`来解决。

```
p{color:red!important;}
p{color:green;}
<p class="first">三年级时，我还是一个<span>胆小如鼠</span>的小女孩。</p>
```

注：`p`段落中的文字会显示为红色。`!important`要写在`;`前面。

注：当网页制作者不设置css样式时，浏览器会按照自己的一套样式来显示网页，用户也可以在浏览器中设置自己习惯的样式。这时样式优先级：`浏览器默认样式<网页制作者样式<用户自己设置的样式`，`!important`样式优先级权值高于用户自己设置的样式。

CSS格式化排版

文字排版--字体

利用css可以为网页中的文字设置字体、颜色、字号等样式属性。

```
body{font-family:"宋体";}
```

注：网页中的文字设置为宋体。不要设置不常用字体，如果电脑中没有安装设置的字体，浏览器显示默认的字体。现在的网页一般设置为微软雅黑`body{font-family:"Microsoft Yahei"}`，用英文的兼容性好一点。

字号、颜色

使用`font-size`和`color`来设置字体的颜色和自号。

```
body{
  font-size:20px;
  color:blue;
}
```

注意在`` `body ``中，将自号设置为20个像素，颜色设置为蓝色。

粗体

`font-weight`来指定字体的宽度。

```
p span{font-weight:bold;}
```

使用`font-weight`来实现粗体，不要`h`标签和`strong`来实现粗体。

斜体

`font-style:italic;`来实现斜体。

```
p a{font-style:italic;}

<p>三年级时，我还是一个<a>胆小如鼠</a>的小女孩。</p>
```

下划线

为文字设置下划线，可以在视觉上强调文字。

```
p a{text-decoration:underline;}

<p>三年级时，我还是一个<a>胆小如鼠</a>的小女孩。</p>
```

注：使用`text-decoration:underline;`实现。

删除线

在网页中实现删除线，`text-decoration:line-through`。

```
  .oldPrice{text-decoration:line-through;}
```

在`oldPrice`的文本中添加删除线。

缩进

中文文字习惯在段前空两个字符的空白，采用`p{text-indent:2em}`实现。

```
p{text-indent:2em;}
<p>1922年的春天，一个想要成名名叫尼克卡拉威（托比?马奎尔Tobey Maguire 饰）的作家，离开了美国中西部，来到了纽约。那是一个道德感渐失，爵士乐流行，走私为王，股票飞涨的时代。为了追寻他的美国梦，他搬入纽约附近一海湾居住。</p>
```

注：2em指文字大小的两倍。

行间距（行高）

行高属性：`line-height`

```
p{line-height:1.5em;}
<p>菲茨杰拉德，二十世纪美国文学巨擘之一，兼具作家和编剧双重身份。他以诗人的敏感和戏剧家的想象为"爵士乐时代"吟唱华丽挽歌，其诗人和梦想家的气质亦为那个奢靡年代的不二注解。</p>
```

注：行高为文字大小的1.5倍。

中文字间距、字母间距

使用`letter-spacing`实现网页排版中的文字间隔或者字母间隔。

```
h1{
    letter-spacing:50px;
}
...
<h1>了不起的盖茨比</h1>
```

注：在文本为英文是，设置英文字母间距。

单词间距：`word-spacing`

```
h1{
    word-spacing:50px;
}
...
<h1>welcome to imooc!</h1>
```

对齐

为块元素中的文本、图片设置居中样式，使用`text-align:center`实现。

```
h1{
    text-align:center;
}
<h1>了不起的盖茨比</h1>
```

注：居左left，居右right。

CSS盒模型

元素分类

在css中，html中的标签元素大体被分为三种不同的类型：块状元素、内联元素（行内元素）、内联块状元素。

常用的块状元素：

`<div>、<p>、<h1>···<h6>、<ol>、<ul>、<dl>、<address>、<blockquote>、<form>`

内联元素：

`<a>、<span>、<br>、<i>、<em>、<strong>、<label>、<q>、<var>、<cite>、<code>`

内联块状元素：

`<img>、<input>`

块级元素

在html中，`<div>、<p>、<h1>···<h6>、<ol>、<ul>、<dl>、<address>、<blockquote>、<form>`为块级元素（block），可以将内联元素转换为块级元素。

```
a{dispaly:block;}
```

注：将内联元素`<a>`转换为块状元素。

块状元素的特点：

1. 每个块级元素都从新的一行开始，并且其后的元素另起一行；
2. 元素的高度、宽度、行高以及顶和底边距都可以设置；
3. 元素宽度在不设置情况下，是它本身父容器的100%（和父元素宽度一

致），除非设定一个宽度。

内联元素

在html中，`<a>、<span>、<br>、<i>、<em>、<strong>、<label>、<q>、<var>、<cite>、<code>`就是内联元素（inline），块状元素可以转换为内联元素。

```
div{
  display:inline;
}
```

内联元素特点：

1. 和其他元素在同一行上；
2. 元素的高度、宽度及顶部、底部边距不可设置；
3. 元素的宽度就是它所包含文字或图片的宽度，不可改变。

内联块状元素

同时具备内联元素与块状元素的特点，`<img>、<input>`是内联块状元素（inline-block）。

```
p{display:inline-block;}
```

inline-block特点：

1. 和其他元素在同一行上；
2. 元素的高度、宽度、顶部和底部边距都可以设置。

盒模型-边框

盒子模型的边框是围绕着内容和补白的线，这条线的粗细、样式和颜色三个属性可以设置。

```
div{
    border-width:2px;
    border-style:solid;
    border-color:red;
}
```

注：设置`div`标签的边框粗细2px，样式为实心，颜色为红色，可以合写为下列型式。

```
div{
    border:2px  solid  red;
}
```

注意：

1. border-style（边框样式）：dashed（虚线）、solid（实现）、dotted（点线）；
2. border-color（边框颜色）：可以设置为16进制颜色；

   ```
   border-color:#888;//前面的井号不要忘掉。
   ```

3. border-width（边框宽度）：thin、 medium、thick（不常用），最常用的是像素px。

四个方向的边框：

css允许只为一个方向的边框设置样式：

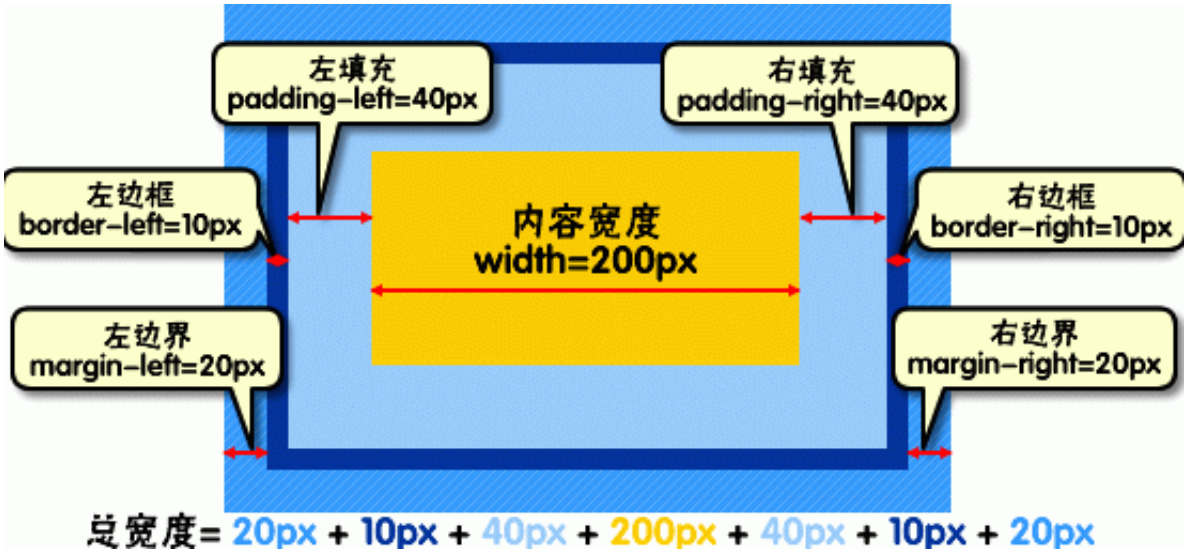
```
div{border-bottom:1px solid red;}
```

同理可以修改其他三边：

```
border-top:1px solid red;
border-right:1px solid red;
border-left:1px solid red;
```

高度和宽度

盒模型的高度（height）和宽度（width）在css定义内值得是填充以里的内容范围，一个`元素的实际宽度=左边界+左边框+左填充+内容宽度+右填充+右边框+右边界`。元素高度同理。



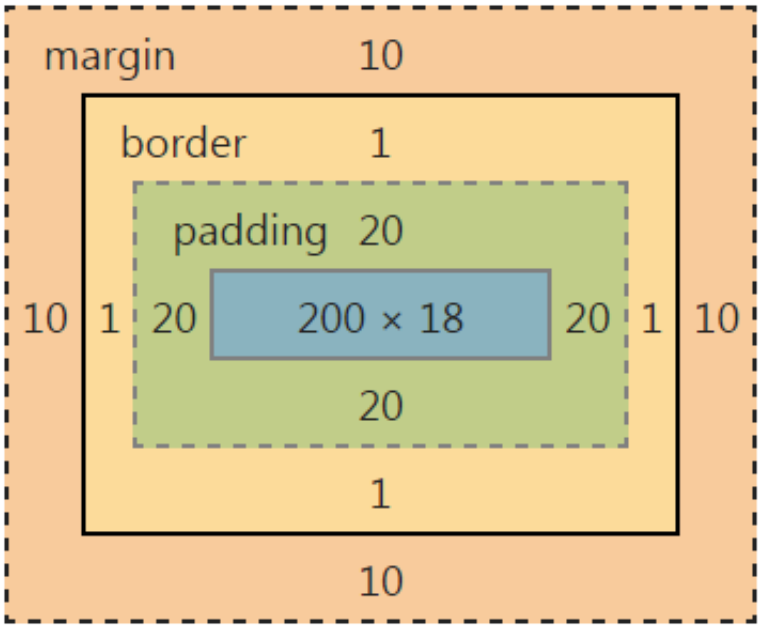
css代码：

```
div{
    width:200px;
    padding:20px;
    border:1px solid red;
    margin:10px;
}
```

html代码：

```
<body>
   <div>文本内容</div>
</body>
```

元素的实际长度为：10px+1px+20px+200px+20px+1px+10px=262px。



注：padding是内边距、margin是外边距。

填充**padding**

元素内容与边框之间可以设置距离，称之为填充。填充可以分为上、右、下、左（顺时针）。

```
div{padding:20px 10px 15px 30px;}
```

顺序不能乱，展开：`padding`指内边距。

```
div{
   padding-top:20px;
   padding-right:10px;
   padding-bottom:15px;
   padding-left:30px;
}
```

边界**margin**

元素与其他元素之间的距离用`margin`来设置，边界也分为上、右、下、左。

```
div{margin:20px 10px 15px 30px;}
```

展开：

```
div{
    margin-top:20px;
    margin-right:10px;
    margin-bottom:15px;
    margin-left:30px;
}
```

上下左右边界都是10px：

```
div{ margin:10px;}
```

上下边界10px，左右20px：

```
div{ margin:10px 20px;}
```

注：padding在边框内部，margin在边框外部。

CSS布局模型

布局模型和盒模型都是css最基本、最核心的概念。布局模型建立在盒模型的基础之上，但是不同于**css**布局样式或者**css**布局模板。css布局模板是布局模型的外在表现形式。

css包含三种基本的布局模型：`Flow、Layer、Float`

流动模型Flow、浮动模型Float、层模型Layer。

流动模型

流动模型Flow是默认的网页布局模式，默认状态下html网页元素都是根据流动模型来分布网页内容。

特点：

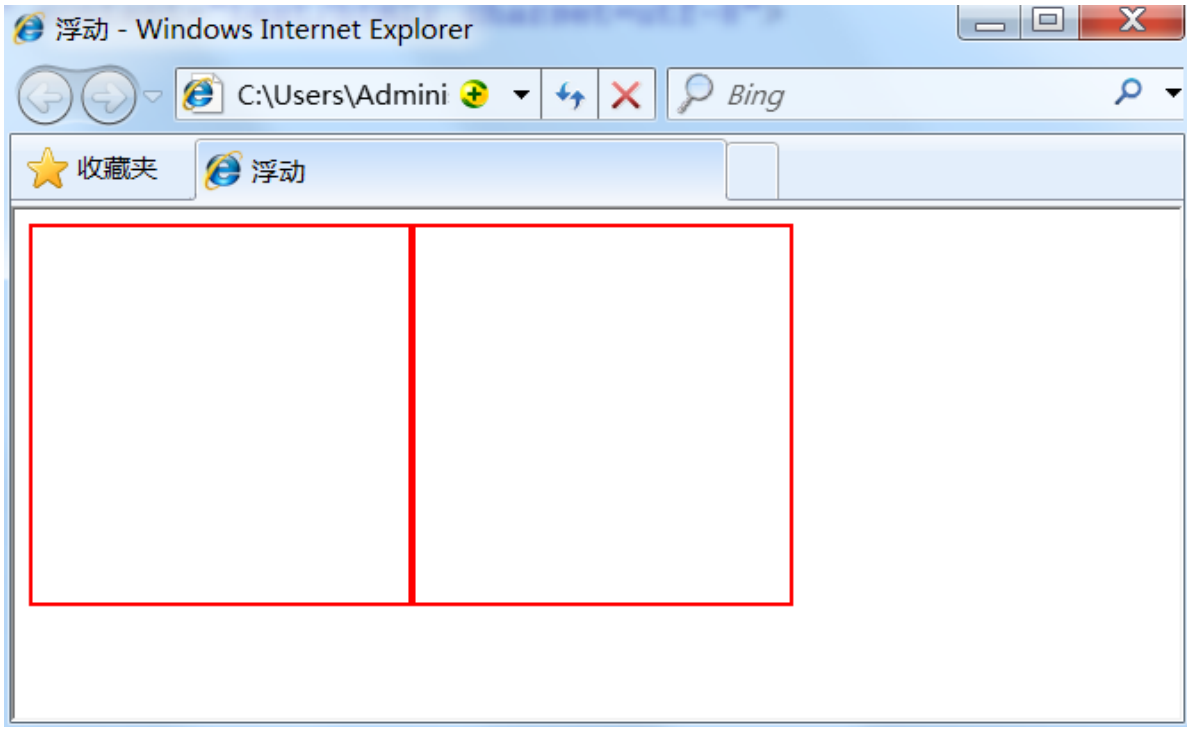
1. 块状元素都会在所处的包含元素内自上而下按顺序垂直延伸分布，默认状况下块状元素的宽度都是100%。实际上，块状元素都会以行的形式占据位置（独占一行）。
2. 在流动模型下，内联元素都会在所处的包含元素内从左到右水平分布显示（不独占一行）。

浮动模型

在流动模型Flow中，块状元素总是独占一行。设置元素浮动可以让两个块状元

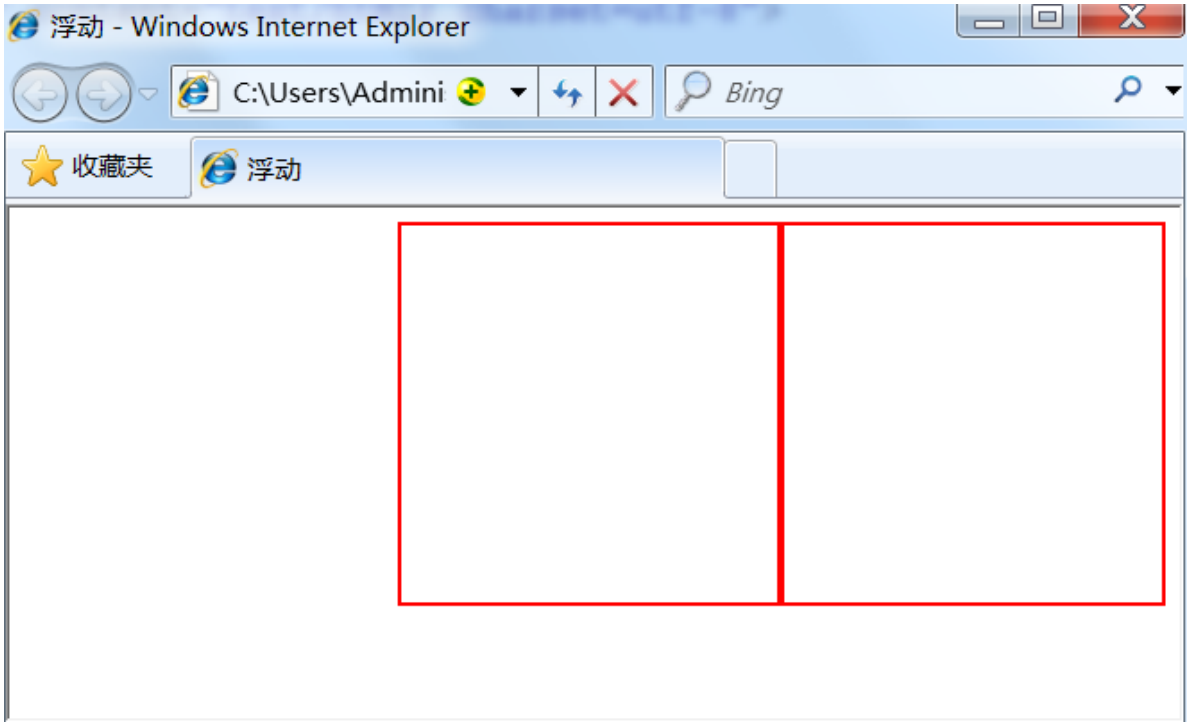
素并排显示。任何元素在默认情况下是不能浮动的，但可以用css定义为浮动。

```
div{
    width:200px;
    height:200px;
    border:2px red solid;
    float:left;
}
<div id="div1"></div>
<div id="div2"></div>
```



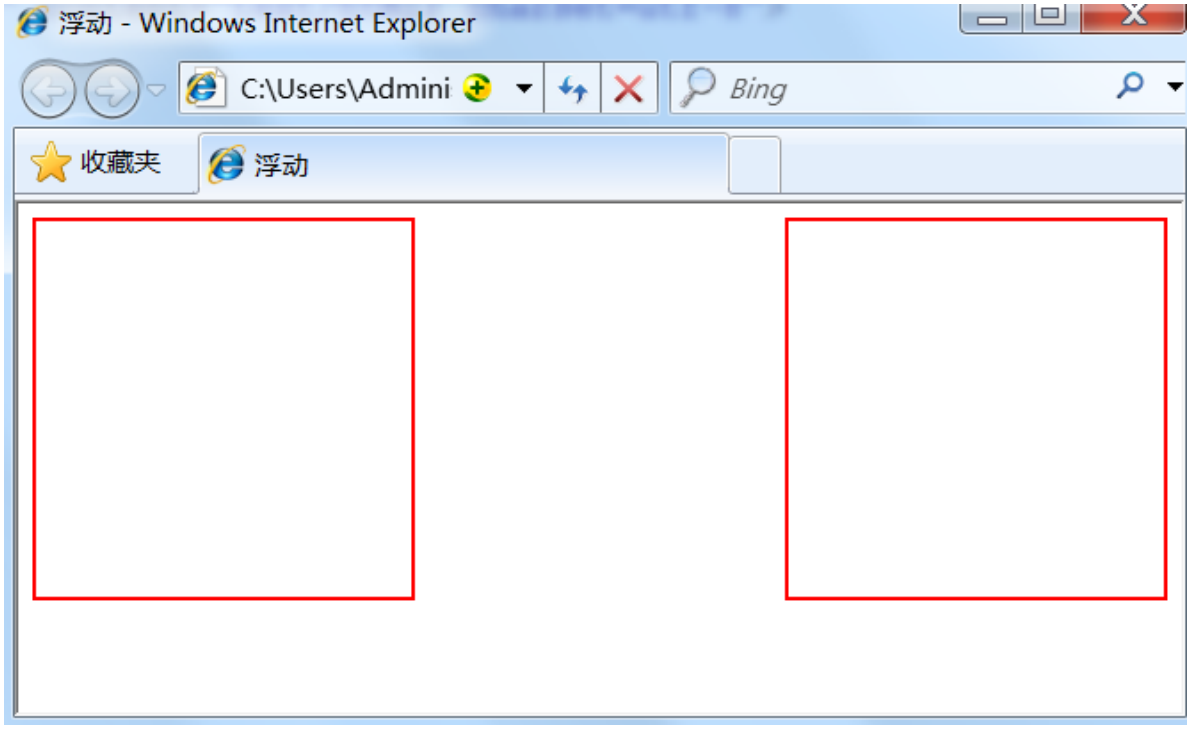
两个元素右浮动也可以实现一行显示：

```
div{
    width:200px;
    height:200px;
    border:2px red solid;
    float:right;
}
```



两个元素一左一右显示：

```
div{
    width:200px;
    height:200px;
    border:2px red solid;
}
#div1{float:left;}
#div2{float:right;}
```



层布局模型

层布局模型Layer类似PS等图形处理软件中图层的概念，每个图层能够精确定位操作。但是在网页设计领域，由于网页大小的活动性，层布局不太受追捧。CSS定义了一组定位（positioning）属性来支持层布局模型。

层模型有三种形式：

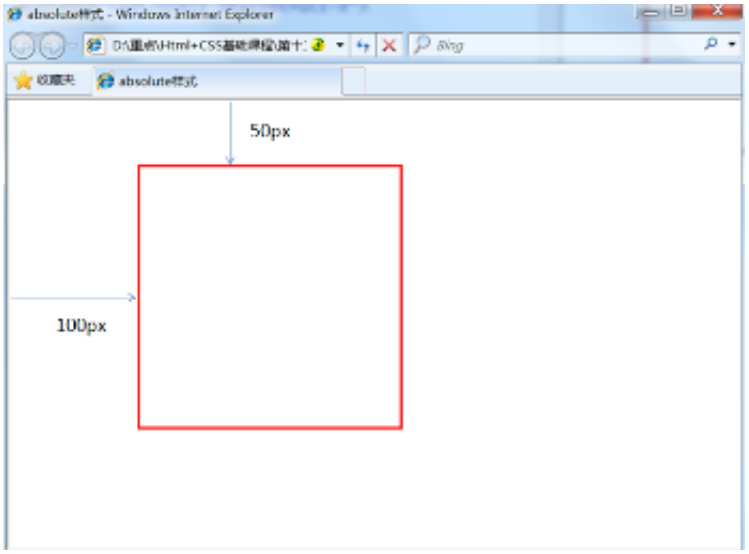
1. 绝对定位：`position:absolute`
2. 相对定位：`position:relative`
3. 固定定位：`position: fixed`

绝对定位

元素在层模型中设置绝对定位，使用`positon:absolute`语句，将元素从文档流中拖出来，然后使用left、right、top、bottom属性性对于其最接近的一个具有定位属性的父包含块进行定位（如果不存在，则相对于body，即浏览器窗口）。

```
div{
    width:200px;
    height:200px;
    border:2px red solid;
    position:absolute;
    left:100px;
    top:50px;
}
<div id="div1"></div>
```

注：实现`div`元素相对于浏览器窗口向右移100个像素，向下移50个像素。



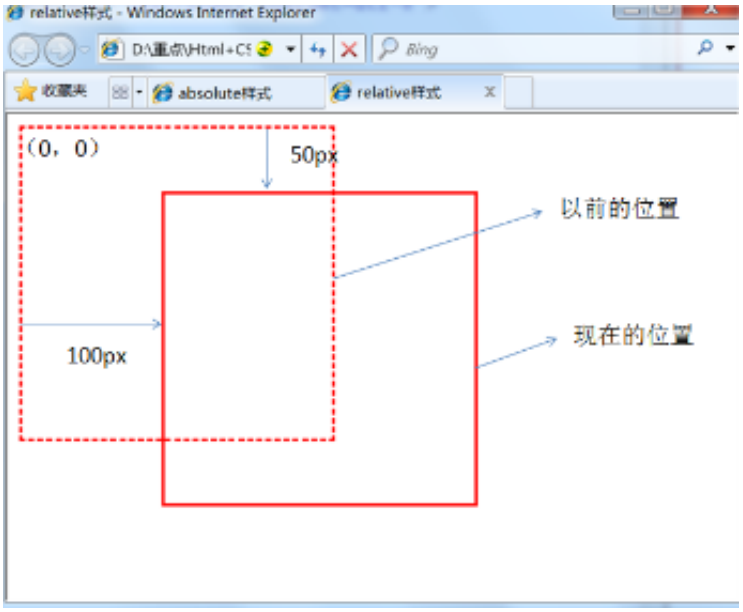
相对定位

通过`position:relative`为层模型中的元素设置相对定位，通过left、right、top、bottom属性确定元素在正常文档流中的偏移位置。相对定位完成的过程：首先按static（float）方式生成一个元素（并且元素向层一样浮动起来），然后相对于以前的位置移动，移动的方向和幅度由left、right、top、bottom属性确定，偏移前的位置保持不动。

```
#div1{
    width:200px;
    height:200px;
    border:2px red solid;
    position:relative;
    left:100px;
    top:50px;
}

<div id="div1"></div>
```

注：相较于以前位置向下移动50px，向右移动100px。



固定定位

通过`position:fixed`实现固定定位，相对移动坐标是视图（屏幕内的网页窗口）本身。由于视图本身是固定的，不会随浏览器窗口的滚动条滚动而变化，除非你在屏幕中移动浏览器窗口的屏幕位置，或者改变浏览器窗口的显示大小。

固定定位的元素会始终位于浏览器窗口内视图的某个位置，不会受文档流动影响。与`background-attachment:fixed`属性功能相同。

```
#div1{
    width:200px;
    height:200px;
    border:2px red solid;
    position:fixed;
    left:100px;
    top:50px;
}
<p>文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本文本。</p>
....
```

实现相对于浏览器视图向右移动100px，向下移动50px。并且拖动滚动条时位置固定不变。

Relative和Absolute组合使用

绝对定位方法：使用`position:absolute`可以实现被设置元素相对于浏览器（**body**）设置定位后，可以通过`position:relative`来设置相对于其他元素进行定位。但必须遵循下列规定：

1. 参照定位的元素必须是相对定位元素的前辈元素：

```
        <div id="box1"><!--参照定位的元素-->
            <div id="box2">相对参照元素进行定位</div><!--相对定位元素-->
        </div>
```

2. 参照定位元素必须加入`position:relative`；

```
#box1{
    width:200px;
    height:200px;
    position:relative;
}
```

3. 定位元素加入`position:absolute`后，可以使用left、right、top、bottom来进行偏移定位：

```
#box2{
    position:absolute;
    top:20px;
    left:30px;
}
```

注：box2可以相对于父元素box1定位（这里的参照物不再是浏览器，可以自由设置）

CSS代码缩写，占用更少带宽

盒模型代码缩写

讲盒模型时，外边距(margin)、内边距(padding)、和边框(border)设置上下左右四个方向的边距按照顺时针方向：上、右、下、左。

具体应用在**margin**：

```
margin:10px 15px 12px 14px;
/*上设置为10px、右设置为15px、下设置为12px、左设置为14px*/
```

通常有以下三种缩写：

1. 如果top、right、bottom、left值相同：

```
        margin:10px 10px 10px 10px;
```

可以缩写为：

```
        margin:10px;
```

2. 如果bottom和top值相同，left和right值相同：

```
margin:10px 20px 10px 20px;
```

可以缩写为：

```
margin:10px 20px;
```

3. 如果left和right值相同：

```
margin:10px 20px 30px 20px;
```

可以缩写为：

```
margin:10px 20px 30px;
```

注意：padding和margin的缩写形式一致。

颜色值缩写

颜色的css样式同样可以缩写，当设置的颜色是16进制值时，如果两位相同的值可以缩写一半。

```
p{color:#000000;}
```

可以缩写为：

```
p{color: #000;}
```

```
p{color: #336699;}
```

可以缩写为：

p{color: #369;}

字体缩写

字体css样式的缩写：

```
body{
    font-style:italic;
    font-variant:small-caps;
    font-weight:bold;
    font-size:12px;
    line-height:1.5em;
    font-family:"宋体",sans-serif;
}
```

可以缩写为一句：

```
body{
    font:italic  small-caps  bold  12px/1.5em  "宋体",sans-serif;
}
```

注意：

1. 使用这种简写至少要指定font-size、font-family属性，其他属性（如 font-weight、font-style、font-varient、line-height）如果没有指定值会使用默认值。
2. 在缩写时 font-size 与 line-height 中间要加入“/”斜杠。

   一般情况下对于中文网站，英文较少，比较常用：

   设置字号、行间距、中文字体、英文字体。

   ```
   body{
       font:12px/1.5em  "宋体",sans-serif;
   }
   ```

单位和值

颜色

在网页中颜色设置很重要，包括字体颜色（color）、背景颜色（background-color）、边框颜色（border）等。

设置颜色的方法：

1. 英文命令颜色：red、blue、pink…

   ```
   p{color:red;}
   ```

2. RGB颜色：与PS中的RGB颜色一致，R（red）、G（green）、B（blue）三种颜色的比例来配色。

   ```
   p{color:rgb(133,45,200);}
   ```

   每一项的值可以是0~255之间的整数，也可以是0%-100%的百分数。

3. 十六进制颜色：运用比较普遍的方法。原理是RGB设置，但是每一项的值由0-255编程了00-ff。只是用16进制来表示RGB配色的数值。

```
1  p{color:#00ffff;}
```

配色表：

长度值

长度单位目前常用的：像素px、文本宽度的倍数em、%百分比。都是相对单位。

像素

像素指显示器上的小点（CSS规范中假设90像素=1英寸）。实际情况是浏览器会使用显示器的实际像素值，大多数设计倾向使用像素px作为单位。

em

数值上等于本元素字体给定的`font-size`值，如果元素`font-size`值为20px，

有`1em=20px`。

```
p{font-size:12px;text-indent:2em;}
```

实现代码首行缩进40px。注意：给`font-size`设置单位为em时，计算的标准以p的父元素的`font-size`为基础。

百分比

```
p{font-size:12px;line-height:130%}
```

设置行高（行间距）为字体的130%。

CSS样式设置技巧

行内元素-水平居中

被设置元素为文本、图片等行内元素时，水平居中通过给父元素设置`text-align`来实现。

```
<body>
  <div class="txtCenter">我是文本，哈哈，我想要在父容器中水平居中显示。</div>
</body>
```

```
<style>
  div.txtCenter{
    text-align:center;
  }
</style>
```

块元素-水平居中

当被设置元素是块元素时`text-align:center`不起作用，块状元素分为定宽和不定宽两种。

定宽：

对于定宽块状元素可以通过设置左右**margin**值为：**auto**来实现居中。

```
<body>
  <div>我是定宽块状元素，哈哈，我要水平居中显示。</div>

</body>
```

```
<style>
div{
    border:1px solid red;/*为了显示居中效果明显为 div 设置了边框 */

    width:500px;/*定宽*/
    margin:20px auto;/* margin-left 与 margin-right 设置为 auto */
}
```

可以写为：

```
{
  margin-left:auto
  margin-right:auto
  }
```

注意：元素的上下margin值可以随意设置。

理解：行内元素如文本、图片等设置`text-align:center`是将元素标签框以默认样式显示（没有居中），标签匡中的内容居中显示；定宽块状元素，是将制定宽度的标签框居中放置，其中的内容在标签框内不居中显示。

不定宽块状元素

在实际工作中会遇到为“不定宽度的块状元素”设置居中，类似：网页上的分页导航，分页数量不确定所以不能通过设置宽度来限制它的弹性。为不定宽度块状元素设置居中的三种方法：

1. 加入`table`标签；
2. 设置`display:inline`方法；
3. 设置`position:relative`和`left:50%`

加入`table`标签：

1. 为需要设置居中的元素外面加入一个`table`标签（包括`<tbody>、<tr>、<td>`）；
2. 为这个table设置左右margin居中（与定宽块相同）。

```
<style>
table{
    margin:0 auto;
}

ul{list-style:none;margin:0;padding:0;}
li{float:left;display:inline;margin-right:8px;}
</style>
```

```
<div>
<table>
  <tbody>
    <tr><td>
    <ul>
        <li><a href="#">1</a></li>
        <li><a href="#">2</a></li>
        <li><a href="#">3</a></li>
    </ul>
    </td></tr>
  </tbody>
</table>
</div>
```

将块级元素为：**`display:inline`**：将块级元素（独占一行，其后内容另起一行）转换为内联元素后，使用`text-align:center`来实现居中效果。

```
<body>
<div class="container">
    <ul>
        <li><a href="#">1</a></li>
        <li><a href="#">2</a></li>
        <li><a href="#">3</a></li>
    </ul>
</div>
</body>
```

```
style>
.container{
    text-align:center;
}
.container ul{
    list-style:none;
    margin:0;
    padding:0;
    display:inline;
}
.container li{
    margin-right:8px;
    display:inline;
}
</style>
```

注：相比于table方法，不用增加无语义标签，简化了嵌套深度，但是将块状元素变为行内元素后，少了设定长度值等功能。

通过给父元素设置**float**，然后父元素设置`position:relative`和`left:50%`，子元素设置`position:relative`和`left:50%`实现水平居中。

```
<body>
<div class="container">
    <ul>
        <li><a href="#">1</a></li>
        <li><a href="#">2</a></li>
        <li><a href="#">3</a></li>
    </ul>
</div>
</body>
```

```
<style>
.container{
    float:left;
    position:relative;
    left:50%
}

.container ul{
    list-style:none;
    margin:0;
    padding:0;

    position:relative;
    left:-50%;
}
.container li{float:left;display:inline;margin-right:8px;}
</style>
```

注：这种方法可以保留块级元素以`display:block`显示，不增加无语义标签，不增加嵌套深度，但是设置`position:relative`带来了一定的副作用。

总结：三种方法各有优点，视情况使用。