

1. 文本格式化

本章重要解说如何实现文本格式化，涉及控制文本字体（如字体大小、字体样式和字体系列等）和设立文本格式（如文本颜色、文本排列和文本缩进等）。

1.1. 文本格式化

1.1.1. 控制字体

CSS 字体属性定义文本字体系列、大小、加粗、风格（如斜体）等，这些属性会直接影响字体及其外观。惯用属性有：

1、指定字体（font-family）

可以使用 font-family 属性指定文本字体，语法如下：

```
1. font-family: name/inherit;
```

如果需要指定字体，则设立值为 name，即首选字体名称。如果字体名称有各种单词，即中间有空格，则需要将字体名称用一对单引号或者双引号包围起来。如果字体名称包括非 ASCII 字符，就必要声明样式表编码。

这个属性最大问题在于，如果顾客机器上并没有安装所需要字体，则会显示默认字体。因而，在指定字体时，最佳同步指定代替字体。

咱们可觉得 font-family 属性指定各种字体，且各种字体之间用逗号隔开，这样可觉得页面指定一种字体列表。如果顾客机器没有第一种字体，则浏览器会查找字体列表中下一种字体作为代替字体显示。如果找遍了字体列表还是没有可以使用字体，浏览器会使用默认字体显示页面。

咱们可以结合特定字体和通用字体系列来指定字体，如：

```
1. h1 {font-family: Georgia, serif;}
```

如果顾客机器上没有安装 Georgia，但安装了 Times 字体（serif 字体系列中一种字体），浏览器就也许对 <h1> 元素使用 Times。尽管 Times 与 Georgia 并不完全匹配，但至少足够接近。

2、字体大小 (font-size)

font-size 属性用来设立文本大小。如果没有规定字体大小，普通文本（例如段落）默认大小是 16 像素（16px=1em）。

可以使用各种方式指定这个属性值：绝对大小、相对大小、长度单位和比例。其中，如果使用长度单位来设立字体大小，还可以分别使用相对单位和绝对单位。

长度是指采用一种长度单位来表达字体大小。可使用绝对单位如表 - 1 所示。

表 - 1 长度绝对单位

单位	说明
pt	1 点（也称为磅，1 pt 等于 1/72 英寸）
pc	12 点（1 pc 等于 12 pt）
in	英寸
cm	厘米
mm	毫米

图 - 1 给出了使用长度绝对单位来设立字体大小效果。

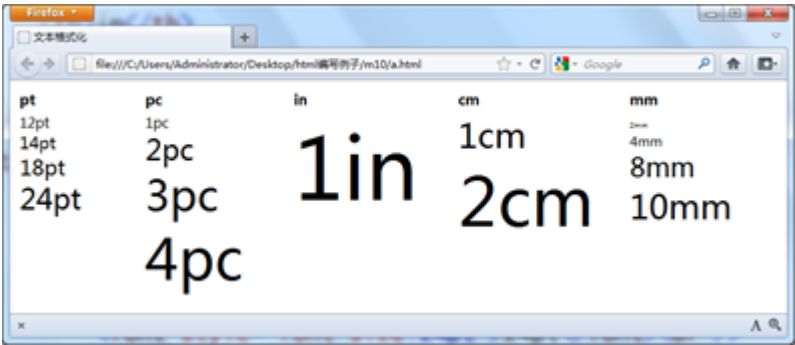


图-1

也可以使用相对单位长度来设立字体大小，可使用相对单位如表 - 2 所示。

表 - 2 长度相对单位

单位	说明
px	像素，计算机屏幕上的一个点
em	em 单位相对于参考元素的字体尺寸，1em 等于当前的字体尺寸，2em 等于当前字体尺寸的两倍
ex	小写字母 x 的高度，取决于不同字体的类型（不同的字体有不同的比例，且 x 的高度通常是字体尺寸的一半）

如果需要使用相对单位来设立大小，咱们可以使用 px，表达像素。通过像素设立文本大小，可以对文本大小进行完全控制。例如，咱们可以这样设立：

```
1. h1 {font-size:60px;}
2. h2 {font-size:40px;}
3. p {font-size:14px;}
```

font-size 属性值还可以设立为 “inherit”，规定应当从父元素继承字体尺寸。

3、字体加粗 (font-weight)

font-weight 属性用于设立文本粗细，惯用于实现将显示元素文本中所用字体加粗。

该属性也许值如表 - 3 所示。

表 - 3 font-weight 属性取值

值	说明
normal	定义标准的字符，为默认值
bold	定义粗体字符
bolder	定义更粗的字符
lighter	定义更细的字符
100、200、300、400、500、600、700、800、900	定义由粗到细的字符（400 等同于 normal，而 700 等同于 bold）
inherit	规定应该从父元素继承字体的粗细

使用 bold 核心字可以将文本设立为粗体。

核心字 100 ~ 900 为字体指定了 9 级加粗度。如果一种字体内置了这些加粗级别，那么这些数字就直接映射到预定义级别，100 相应最细字体变形，900 相应最粗字体变形。数字 400 等价于 normal，而 700 等价于 bold。

如果将元素加粗设立为 bolder，浏览器会设立比所继承值更粗一种字体加粗。与此相反，核心词 lighter 会导致浏览器将加粗度下移而不是上移。

例如，咱们可以这样定义样式规则：

```
1. p.normal {font-weight:normal;}
2. p.thick {font-weight:bold;}
3. p.thicker {font-weight:900;}
```

然后查看如下代码：

```
1. <p class="normal">font-weight:normal</p>
2. <p class="thick">font-weight:bold</p>
3. <p class="thicker">font-weight:900</p>
```

上述代码在浏览器中显示效果如图-2 所示。

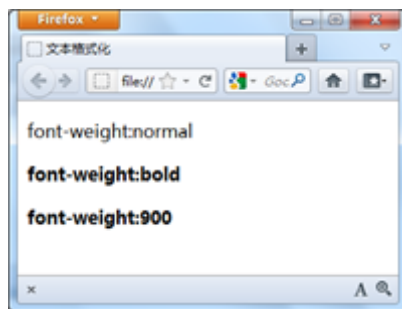


图-2

其中，bold 是最惯用值，也会遇到使用 normal 状况，特别是在大量加粗文本中创立不同效果文本时。

1.1.2. 控制文本格式

CSS 除了可以设立字体，还可以定义文本外观。通过文本属性，可以变化文本颜色、字符间距、对齐文本、装饰文本、对文本进行缩进等。

1、文本颜色 (color)

color 属性用于设立文本颜色，即元素前景色。这个颜色还会应用到元素所有边框，除非被 border-color 或此外某个边框颜色属性覆盖。

没有设立 color 属性文本将使用浏览器默认颜色显示。如果需要使用该属性设立文本颜色，该属性值可以是颜色名称、rgb 值或者十六进制数，其默认值取决于浏览器。

例如，咱们可以这样设立：

```
1. p {color: rgb(255, 255, 0);}
```

或者

```
1. p {color: #FFFF00;}
```

2、文本排列 (text-align)

`text-align` 是一种基本属性，用于设立一种元素中文本行互相之间对齐方式。

该属性通过指定行框与哪个点对齐，从而设立块级元素内文本水平对齐方式。该属性也许取值如表 - 4 所示。

表 - 4 `text-align` 属性取值

值	说明
<code>left</code>	将文本与包含它的元素的左边框对齐
<code>right</code>	将文本与包含它的元素的右边框对齐
<code>center</code>	将文本居中放置在包含它的元素中
<code>justify</code>	将文本的宽度扩展为包含的元素的整个宽度 ,从而实现两端对齐的效果

前 3 个值相称直接，不用额外解释。

`text-align` 属性还也许取值 `justify`，咱们称为两端对齐。在两端对齐文本中，文本行左右两端都放在父元素内边界上。然后，调节单词和字母间间隔，使各行长度正好相等。这种两端对齐排列方式经惯用于打印。

例如，咱们可以这样定义样式规则：

```
1. td.leftAlign {text-align:left;}
2. td.rightAlign {text-align:right;}
3. td.centerAlign {text-align:center;}
4. td.justifyAlign {text-align:justify;}
```

然后，查看如下代码：

```
1. <table border="1">
2.     <tr>
```

```

3.      <td class="leftAlign">With sunshine, water, and careful
        tending, roses will bloom several times in a season.</td>
4.      </tr>
5.      <tr>
6.      <td class="rightAlign">With sunshine, water, and careful
        tending, roses will bloom several times in a season.</td>
7.      </tr>
8.      <tr>
9.      <td class="centerAlign">With sunshine, water, and careful
        tending, roses will bloom several times in a season.</td>
10.     </tr>
11.     <tr>
12.     <td class="justifyAlign">With sunshine, water, and careful
        tending, roses will bloom several times in a season.</td>
13.     </tr>
14. </table>

```

上述代码在浏览器中显示效果如图-3 所示。

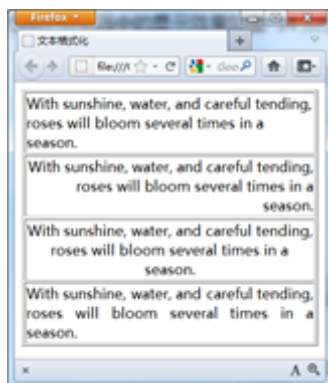


图-3

由图 - 3 可以看出，居左或者居右排列后，此外一端会显示为锯齿状（使用默认字宽和间隔，以单词为单位换行）；而如果使用 justify 则会显示为两端对齐效果。

3、文字修饰 (text-decoration)

text-decoration 属性用于对文本进行修饰。它容许对文本设立某种效果，如加下划线、上划线或者闪烁等。

text-decoration 属性也许取值如表 - 5 所示。

表 - 5 text-decoration 属性取值

值	说明
none	定义标准的文本，为默认值
underline	定义文本下的一条线（下划线）
overline	定义文本上的一条线（上划线）
line-through	定义穿过文本下的一条线（类似于删除线）
blink	定义闪烁的文本

underline 会对元素加下划线;而 overline 作用正好相反，会在文本顶端画一种上划线；值 line-through 则在文本中间画一种贯穿线;blink 会让文本闪烁。

还可以在一种规则中结合各种装饰，只需要为 text-decoration 属性设立各种值，且各种值之间用空格隔开。

例如，如果一种段落中文本既有下划线，又有上划线，咱们可以这样定义：

```
1. p {text-decoration:underline overline;}
```

none 值会关闭原本应用到一种元素上所有装饰。普通，无装饰文本是默认外观，但某些元素例外。例如，链接默认地会有下划线。因而，如果但愿去掉超链接下划线，可以使用如下 CSS 来做到这一点：

```
1. a {text-decoration: none;}
```

如果显式地用这样一种规则去掉链接下划线，那么它与正常文本之间在视觉上唯一差别也许就是颜色（浏览器往往会为链接设立默认颜色）。

blink 值会创立闪烁文本，但是这普通被以为是不赞成使用方式，大某些浏览器也不支持它。

4、行高 (line-height)

line-height 属性用于设立行间距。

当解决大量文本时，增长文本行之间垂直空间量可以提高文档可阅读性，这种空间量称为行间距。在 Web 页面中增长行间距是非常有用。例如，如果文本行之间存在更多空间，当达到一行末尾之后，更容易发现下一行起点。

line-height 属性会影响行框布局。在应用到一种块级元素时，它定义了该元素中基线之间最小距离。

行间距是 line-height 与 font-size 计算值之差。行间距会被分为两半，分别加到一种文本行内容顶部和底部。而可以包括这些内容最小框就是行框。例如，咱们可以这样定义样式规则：

```
1. p {border:1px solid red;}
2. p.smallLength {line-height: 10px;}
3. p.bigLength {line-height: 30px;}
```

然后，查看如下代码：

```
1. <p>这是拥有原则行高段落。在大多数浏览器中默认行高大概是 20px。这是拥有
   原则行高段落。</p>
```

2. `<p class="smallLength">length=10px。这个段落拥有更小行高。这个段落拥有更小行高。</p>`
3. `<p class="bigLength">length=30px。这个段落拥有更大行高。这个段落拥有更大行高。</p>`

上述代码在浏览器中显示效果如图-4 所示。

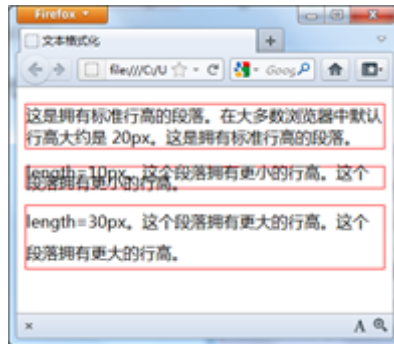


图-4

5、首行文本缩进 (text-indent)

把 Web 页面上段落第一行缩进，这是一种最惯用文本格式化效果。CSS 提供了 text-indent 属性，该属性可以以便地实现文本缩进。

text-indent 属性缩进元素中首行文本，虽然用该属性可以让元素第一行缩进一种给定距离。这个属性最常用用途是将段落首行缩进，下面规则会使段落首行缩进 2 em :

1. `p.first {text-indent: 2em;}`

可觉得块级元素应用 text-indent 属性，但无法将该属性应用于行内元素，且图像之类替代元素上也无法应用 text-indent 属性。但是，如果一种块级元素（例如段落）首行中有一种图像，它会随该行别的文本移动。

`text-indent` 属性除了可以使用长度单位，还涉及比例值。百分数是相对于缩进元素父元素宽度。即如果将缩进值设立为 20%，所影响元素第一行会缩进其父元素宽度 20%。

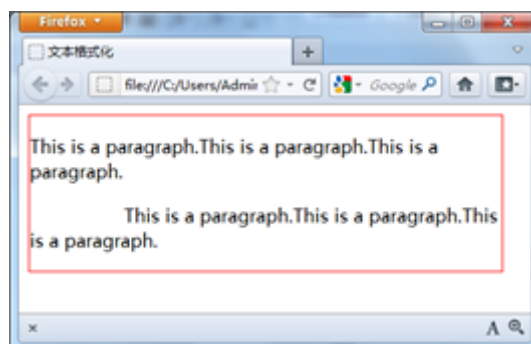
例如，咱们可以这样定义样式规则（为了更好观测父元素边界，咱们定义了 `<div>` 元素边框）：

```
1. div {  
2.   width: 400px;  
3.   border: 1px solid red;}  
4. p.indent {text-indent: 20%;}
```

然后，查看如下代码：

```
1. <div>  
2.     <p>This is a paragraph.This is a paragraph.This is a paragraph.</p>  
3.     <p class="indent">  
4.         This is a paragraph.This is a paragraph.This is a paragraph.  
5.     </p>  
6. </div>
```

上述代码在浏览器中显示效果如图-5 所示（缩进值是父元素 20%，即 80 个像素）：



2. 表格样式

2.1. 表格惯用样式属性

2.1.1. 表格惯用样式属性

对于表格而言，它可以使用前面章节中讲述诸多 CSS 样式（如背景色、边框和字体等），例如，可以使用 `width` 属性、`height` 属性设立单元格大小；可以使用 `border` 属性设立单元格边框；可以使用 `padding` 属性控制表格中内容与边框距离；可以使用 `background` 属性设立表格或者单元格背景色以及背景图像；还可以使用文本格式化有关样式属性来定义表格中文本。

需要注意的是，除了 `background-color` 属性和 `height` 属性之外，最佳避免将这些属性用于 `<tr>` 元素。由于这些属性用于表行时，浏览器对它们支持并没有像它们用于单个单元格时那么好。

2.1.2. 垂直方向对齐

`vertical-align` 属性用于设立元素垂直对齐方式，当操作内联元素时（例如图像或者普通文本），该属性非常有用。在表单元格中，这个属性会设立单元格框中单元格内容对齐方式。

`vertical-align` 属性也许取值有诸多，但是该属性在用于表格单元格中内容垂直对齐方式时可取值如表 - 6 所示。

表 - 6 `vertical-align` 属性取值

值	说明
top	把元素的顶端与行中最高元素的顶端对齐
middle	把此元素放置在父元素的中部
bottom	把元素的顶端与行中最低的元素顶端对齐

`vertical-align` 属性在用于表格单元格中内容垂直对齐方式时，可以设立为顶部 `top`、中部 `middle` 和底部 `bottom` 三个值。

例如，咱们定义如下样式规则：

```
1. .top {vertical-align:top;}
2. .middle {vertical-align:middle;}
3. .bottom {vertical-align:bottom;}
4. td {width:200px;height:80px;border:1px solid red;}
```

然后，在页面主体中添加如下代码：

```
1. <table>
2.     <tr>
3.         <td>some text</td>
4.         <td class="top">some text</td>
5.     </tr>
6.     <tr>
7.         <td class="middle">some text</td>
8.         <td class="bottom">some text</td>
9.     </tr>
10.</table>
```

上述代码在浏览器中显示效果如图-6 所示。

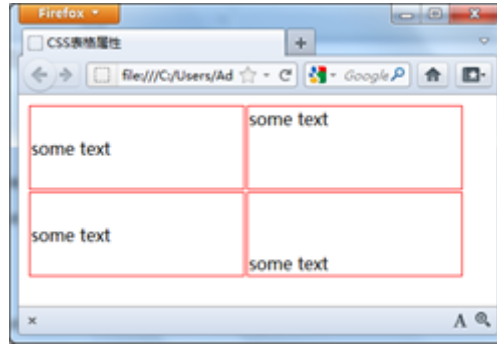


图-6

表格单元格中内容垂直对齐方式默认值为 middle，因而，图 - 1 中表格第一列两个单元格中文本都是垂直居中显示；而设立了 top 值和 bottom 值后来，可以设立单元格中内容位于单元格顶部或者底部显示。

此外，默认状况下，单元格中文本在水平方向上居左显示，可以使用 text-align 属性来修改其水平对齐方式。

2.2. 表格特有样式属性

除了上一节中讲述那些属性以外，尚有某些属性是只与表格有关。这些仅与表格有关样式属性可以极大改进表格外观。

2.2.1. 边框合并

从在前面示例中不难发现，如果设立了单元格边框，相邻单元格边框会单独显示，类似于双线边框（见图-6）。如果需要合并相邻边框，则可以使用 border-collapse 属性。

border-collapse 属性设立与否将表格边框折叠为单一边框，即与否被合并为一种单一边框，还是像在原则 HTML 中那样分开显示。

border-collapse 属性也许取值如表 - 7 所示。

表 - 7 border-collapse 属性取值

值	说明
separate	边框会被分开，为默认值，且不会忽略 border-spacing 和 empty-cells 属性
collapse	如果可能，边框会合并为一个单一的边框，并且会忽略 border-spacing 和 empty-cells 属性
inherit	规定应该从父元素继承 border-collapse 属性的值

border-collapse 属性值如果设立为 separate 值或者不设立，浏览器会独立显示每一种单元格边框，虽然两个相邻单元格具备不同类型边框。

border-collapse 属性值如果设立为 collapse 值，则会对边框进行合并，即会基于一组内置复杂规则来决定显示哪一种边框。普通状况下，浏览器会对边框进行折叠。

例如，咱们定义如下样式规则：

```
1. table {border:2px dotted black;}
2. td {width:200px;height:50px;}
3. table.separate {border-collapse:separate;}
4. table.collapse {border-collapse:collapse;}
5. td.solid {border:3px solid silver;}
6. td.dashed {border:3px dashed gray;}
```

然后，在页面主体中添加如下代码：

```
1. border-collapse:separate
2. <table class="separate">
3.     <tr>
4.         <td class="solid">第 1 行第 1 列</td>
5.         <td class="dashed">第 1 行第 2 列</td>
```

```

6.      </tr>
7. <tr>
8.      <td class="dashed">第 2 行第 1 列</td>
9.      <td class="solid">第 2 行第 2 列</td>
10.     </tr>
11. </table>
12. <br />
13. border-collapse:collapse
14. <table class="collapse">
15.     <tr>
16.         <td class="solid">第 1 行第 1 列</td>
17.         <td class="dashed">第 1 行第 2 列</td>
18.     </tr>
19. <tr>
20.         <td class="dashed">第 2 行第 1 列</td>
21.         <td class="solid">第 2 行第 2 列</td>
22.     </tr>
23. </table>

```

上述代码在浏览器中显示效果如图-7 所示。

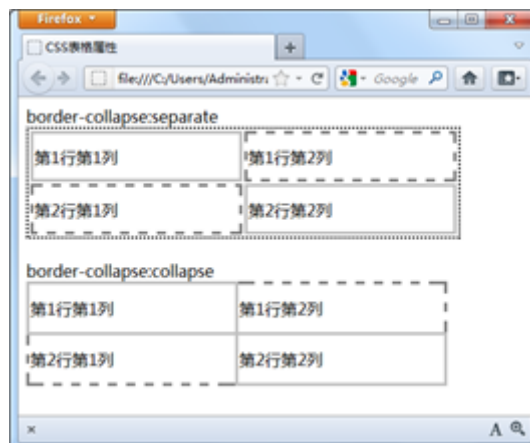


图-7

为了更好查看效果，图-7 中相邻单元格使用了不同样式边框。图-7 中第一种表格 `border-collapse` 属性使用了 `separate` 值，则表格边框、各单元格边框都独立显示，虽然相邻单元格边框样式不同。

第二个表格 `border-collapse` 属性使用了 `collapse` 值，相邻边框则会发生合并，边框会互相折叠。由图可见，实线边框优先级高于虚线边框。

2.2.2. 边框边距

由前面示例不难发现，在表格中单元格之间存在一定间距，如果但愿控制这个间距，则可以使用 `border-spacing` 属性。

`border-spacing` 属性设立相邻单元格边框间距离，但是仅限于分隔单元格边框，即 `border-collapse` 属性为 `separate` 值状况下，也称为边框分离模式。

该属性值可以是长度单位或者单词 `inherit`。如果设立为长度，则可以使用 `px`、`cm` 等单位，但是不容许使用负值；如果设立为 `inherit` 值，表达规定应当从父元素继承 `border-spacing` 属性值。

设立 `border-spacing` 属性值为长度值时，可觉得该属性指定一种或者两个值。

如果指定一种值，则该值同步应用于水平和垂直间距；如果指定两个值，那么第一种值指定水平间距，第二个值指定垂直间距，且两个值之间用空格隔开。

例如，咱们定义如下样式规则：

```
1. table {border:2px dotted gray;}
2. td {
```

```
1. background-color:#f0f0f0;
2. width:200px;
3. height:50px;
4. border-collapse:separate;
5. border:1px solid black;}
6. table.singleSpacing {border-spacing:5px;}
7. table.doubleSpacing {border-spacing:10px 20px;}
```

然后，在页面主体中添加如下代码：

```
1. <table class="singleSpacing">
2.     <caption>设立一种值</caption>
3.     <tr>
4.         <td>第 1 行第 1 列</td>
5.         <td>第 1 行第 2 列</td>
6.     </tr>
7.     <tr>
8.         <td>第 2 行第 1 列</td>
9.         <td>第 2 行第 2 列</td>
10.    </tr>
11. </table>
12. <br />
13. <table class="doubleSpacing">
14.     <caption>设立两个值</caption>
15.     <tr>
16.         <td>第 1 行第 1 列</td>
17.         <td>第 1 行第 2 列</td>
```

```
18.     </tr>
19.     <tr>
20.         <td>第 2 行第 1 列</td>
21.         <td>第 2 行第 2 列</td>
22.     </tr>
23. </table>
```

上述代码在浏览器中显示效果如图-8 所示。



图-8

由图-8 可以看出，第一种表 border-spacing 属性只设立了一种值为 5px，则单元格边框之间垂直和水平间隔均为 5 像素；而第二个表 border-spacing 属性设立了两个值，则单元格边框之间水平间距为 10 像素，而垂直间距较大，为 20 像素。

需要注意的是，为了尽量可以在各浏览器中得到一致显示效果，最佳为 <table> 元素设立 border-spacing 属性，而不是单元格等其它元素。

3. 浮动定位

3.1. 定位概述

3.1.1. 定位概述

CSS 为定位提供了某些属性，运用这些属性，可以建立列式布局，还可以将布局一某些与另一某些重叠，这样可以完毕近年来普通需要使用各种表格才干完毕任务。

定位基本思想很简朴，即可以定义元素框相对于其正常位置应当浮现位置，或者相对于父元素、另一种元素甚至浏览器窗口自身位置。显然，这个功能非常强大，并且各浏览器对 CSS2 中定位支持远胜于对其他方面支持。

CSS 有三种基本定位机制：普通流定位、浮动和绝对定位。

其中，使用 position 属性和偏移属性可以实现普通流定位（涉及相对定位）和绝对定位（涉及固定定位）；使用 float 属性可以实现浮动定位。其她属性为辅助属性。

在背面章节中，咱们会详细解说普通流定位、相对定位、绝对定位和浮动。

3.1.2. 普通流定位

默认状况下，通过使用称为普通流方式在页面中布局元素。

在普通流中定位，页面中块级元素框从上到下一一种接一种地排列，且每一种块级元素都会出当前一种新行中（例如 <p> 元素、<div> 元素），元素框之间垂直距离是由框垂直外边距计算出来。

内联元素将在一行中从左到右排列水平布置，不需要从新行开始。可以使用水平内边距、边框和外边距调节它们间距。但是，垂直内边距、边框和外边距不影响行内框高度。由一行形成水平框称为行框（Line Box），行框高度总是足以容纳它包括所有行内框。但是，设立行高可以增长这个框高度。

为了更好理解普通流定位模式，咱们定义如下样式规则：

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：

<https://d.book118.com/436110055134010104>