

# 层叠样式表CSS（中）

(Cascade Style Sheet)

2019.8.1

isszym [sysu.edu.cn](http://sysu.edu.cn)

# 目录

- [CSS 属性组](#)
- [字体](#)
- [文本颜色和字号](#)
- [文本修饰](#)
- [格式化段落](#)
- [盒模型概述](#)
- [背景设计](#)
- [修饰边框](#)
- [列表样式](#)
- [表格样式](#)

文本  
格式  
化

盒  
模  
型

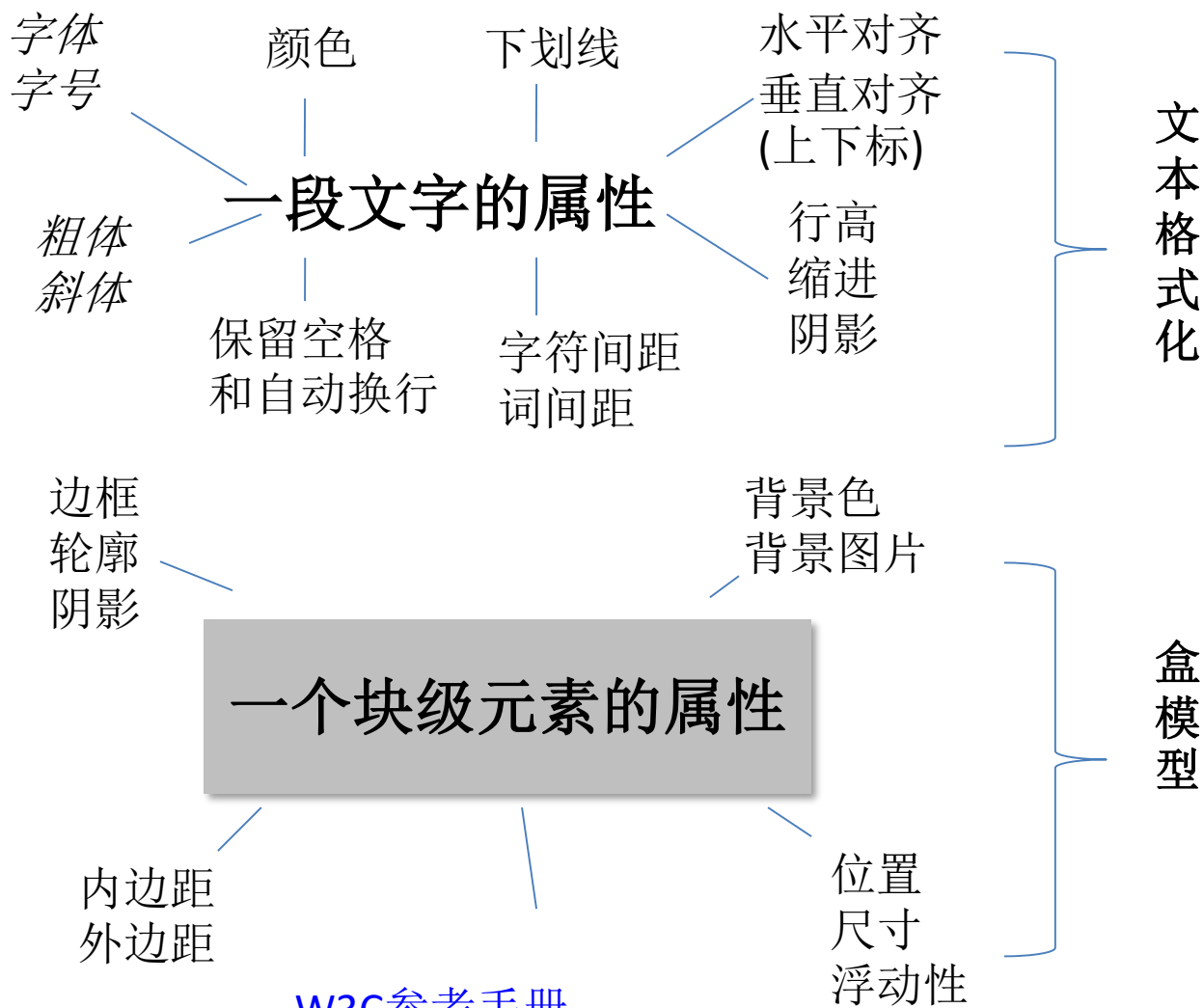
- [设置尺寸与溢出](#)
- [设置浮动元素](#)
- [使用定位技术](#)
- [垂直对齐](#)
- [变换、过渡和动画](#)
- [附录1、140种命名颜色](#)
- [附录2、Web安全色\(216种\)](#)
- [附录3、CSS常用属性](#)
- [附录4、一些属性的取值](#)
- [附录5、transform取值](#)
- [附录6、width和height总结](#)

布  
局

CSS3

# CSS 属性组

- 文本
- 字体
- 背景
- 边框和轮廓
- 外边距
- 内边距
- 尺寸
- 定位
- 打印
- 表格
- 列表
- 内容生成



[W3C参考手册](#)

# 文本格式化

- ❑ 字体: font-family, @font-face
- ❑ 文本颜色和字号: color, font-size
- ❑ 文本修饰: font-style, font-weight, text-transform  
font-variant, text-decoration, text-shadow
- ❑ 格式化段落: letter-spacing, word-spacing, line-height  
text-indent, text-align, white-space  
word-wrap, word-break

# 字体

- 如何使用字体？

```
p {  
  font-family: 宋体, " Times New Roman", Georgia, Serif, Arial;  
}
```

如果系统中没有前面字体的话，就选择后面的字体，都没有的话，就是用系统默认的字體。字体名称有空格要使用引号进行引用。

- 有哪些可用的Web字体？

EOT字体 (Embedded Open Type,.eot)	只有IE使用
TrueType字体(.ttf)和OpenType字体(.otf)	主流浏览器支持
WOFF字体(.woff) Web Open Font Format	为Web而设计的字体，是TrueType字体和OpenType字体的压缩版
SVG字体 (Scalable Vector Graphic)矢量图	基本都不支持

- 如何引入新的Web字体？

```
@font-face {  
    font-family: 'MySans-serif';  
    src: url('font/MySans-serifRegular.ttf');    /*定义标准字体*/  
}  
@font-face {  
    font-family: 'MySans-serif';  
    src:url('font/MySans-serifBold.ttf');        /*定义粗体字体*/  
}  
@font-face {  
    font-family: 'MySans-serif';  
    src:url('font/MySans-serifItalic.ttf');  
}  
  
p {  
    font-family:MySans-serif; /*引用*/  
}
```

\* 如果没有定义斜体和粗体字库，系统会自动生成，自动生成的字体会稍微难看一些。

具体格式： <https://developer.mozilla.org/en/css/@font-face>

# • FontAwesome字符集

Font Awesome 是一套专门为 Twitter bootstrap 设计的图标字体库。这套图标字体集几乎囊括了网页中可能用到的所有图标，除了包括 Twitter Bootstrap 的默认图标外，还有社交网络图标、Web 应用程序图标和编辑器图标等等，可以免费用于商业项目。

## font-awesome.css

```
@font-face {
  font-family: 'FontAwesome';
  src: url('../fonts/fontawesome-webfont.eot?v=4.7.0'); /* 首选字体*/
  src: url('../fonts/fontawesome-webfont.eot?#iefix&v=4.7.0') /* 可选字体*/
      format('embedded-opentype'),
  url('../fonts/fontawesome-webfont.woff2?v=4.7.0')
      format('woff2'),
  url('../fonts/fontawesome-webfont.woff?v=4.7.0')
      format('woff'),
  url('../fonts/fontawesome-webfont.ttf?v=4.7.0')
      format('truetype'),
  url('../fonts/fontawesome-webfont.svg?v=4.7.0
      #fontawesomeregular')
      format('svg');
  font-weight: normal;
  font-style: normal;
}

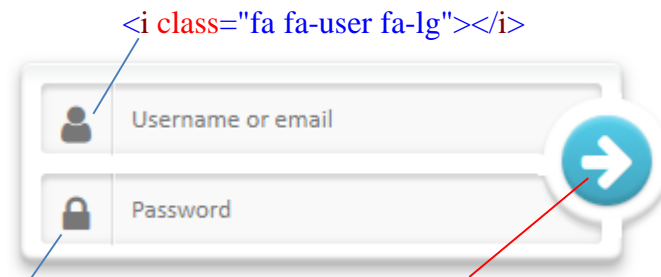
.fa-user:before { content: "\f007"; }
.fa-lock:before { content: "\f023"; }
.fa-arrow-right:before { content: "\f061"; }
```

unicode码

```
.fa {
  display: inline-block;
  font: normal normal normal 14px/1 FontAwesome;
  font-size: inherit;
  text-rendering: auto;
}

.fa-lg {
  font-size: 1.33333333em;
  line-height: 0.75em;
  vertical-align: -15%;
}

.fa-2x { font-size: 2em; }
```



\* 如果字库存于本地: `src: local("FontAwesome")`

`<i class="fa fa-lock fa-lg"></i>`

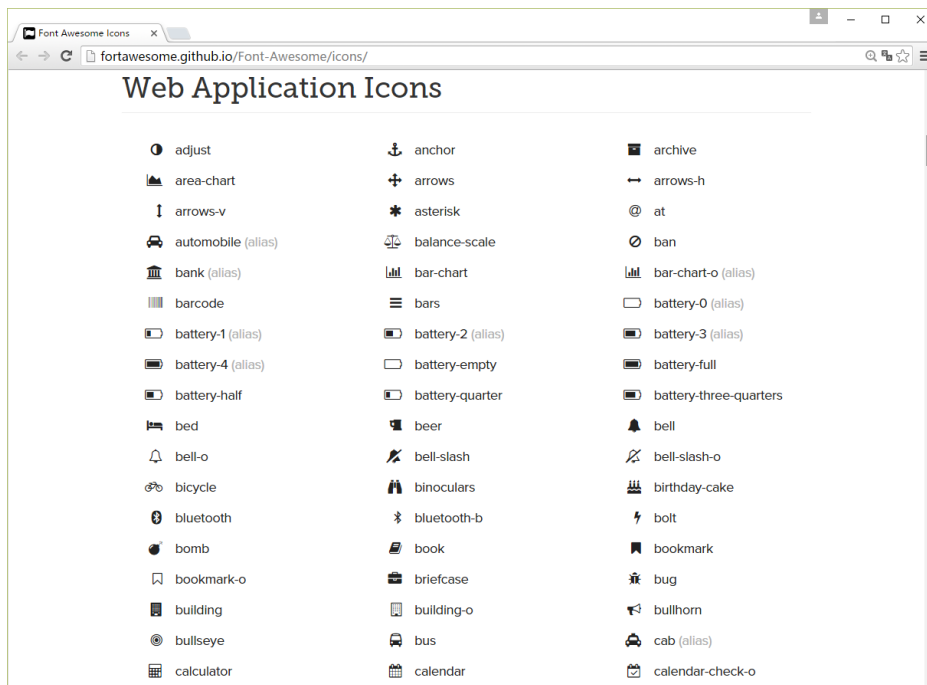
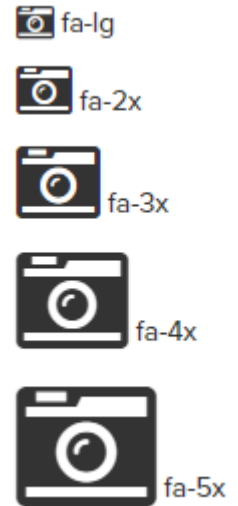
`<i class="fa fa-arrow-right fa-lg"></i>`

\* format用于说明字库类型，未指明format按照扩展名识别

```

<link rel="stylesheet" type="text/css" href="css/ font-awesome.css " />
<form class="form-1">
  <p class="field">
    <input type="text" name="login" placeholder="Username or email" />
    <i class="fa fa-user fa-lg"></i>
  </p>
  <p class="field">
    <input type="password" name="password" placeholder="Password" />
    <i class="fa fa-lock fa-lg"></i>
  </p>
  <p class="submit">
    <button type="submit" name="submit"><i class="fa fa-arrow-right fa-lg"></i></button>
  </p>
</form>

```



[awesome.html](#)

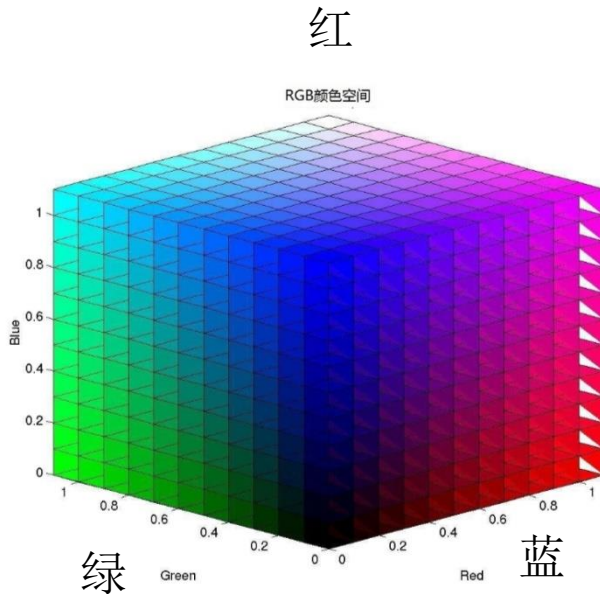
[参考1](#)

[参考2](#)



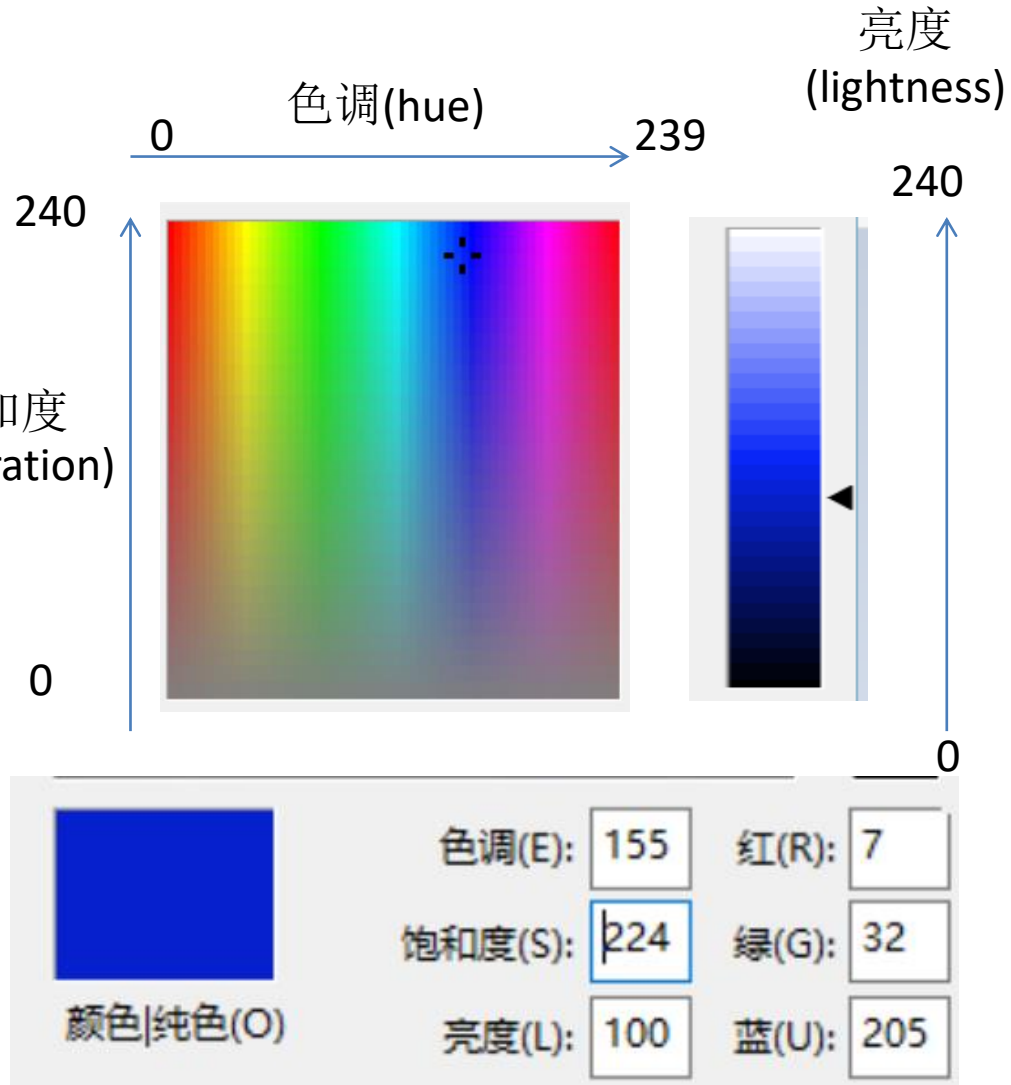
# 文本颜色和字号

- 颜色分量



显示三原色：红绿蓝  
美术三原色：红黄蓝

windows 画图



## • 如何给文本添加颜色

\*颜色不区分大小写

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

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

```
p {  
  color: #3E8988;  
}
```

---

```
span {  
  color: rgb(50%,60%,100%);  
}
```

```
span {  
  color: rgba(0,0,255,0.5);  
}
```

---

```
p {  
  color: hsla(300,100%,50%,0.3);  
}
```

color: #CEE; ↔ color: #CEE~~EE~~;

h-hue(色调) 0~360 红橙黄绿青蓝紫

s-saturation(饱和度) 即颜色的纯度

l-lightness(亮度)

a-alpha (不透明度)

0-红 50-橙 100-黄 150-绿... (每51一级, 0和360都表示红色)

0%-全灰 100%-纯色

0%全黑 100% 全白

取值0~1. 0-完全透明, 0.5-半透明, 1-完全不透明

\* **currentColor**表示当前字体颜色 **transparent**表示透明颜色。 **border:currentColor 1px solid;**

\* **opacity**用来设置整个元素(文字和背景)的透明度, 比如, **div {opacity:0.5}**, 而alpha值设置各种带颜色属性的透明度: 文字, 背景和边框等。 **Web安全颜色**和**命名颜色**见附录。

<http://202.116.76.22:8080/13mo/html5/colorRGB.html> <http://202.116.76.22:8080/13mo/html5/colorHSL.html>

- 如何修改字号

```
p {  
  font-size: 12px;  
}
```

像素

```
p {  
  font-size: 1.2em;  
}
```

一个字的大小

```
p {  
  font-size: 120%;  
}
```

如果基准字号(继承而来)为10px，上述三种表达是一个效果。

```
p {  
  font-size: large;  
}
```

Keywords: xx-small, x-small, small, medium, large, x-large, xx-large;

表示比基准字号大

px	像素(pixel)。屏幕上的点数。	height:12px
em	相对于当前对象内文本的字体尺寸	font-size:1.2em
ex	字符高度的相对尺寸(倍数)	font-size:1.2ex
pt	点/磅(point)，等于1/72英寸，是印刷行业常用单位，	font-size:12pt
pc	12 点活字 (1 pc 等于 12 pt)	font-size:12pc
in	in-英寸,cm-厘米,mm-毫米	height:12in
rem	root em, 相对于根元素(html)的 字体尺寸。	font-size:1.2rem

# 文本修饰

- 如何修饰文本和字母

```
li {  
  font-style: italic;  
}
```

斜体字

```
li {  
  font-style: normal;  
}
```

正常字体

---

```
li {  
  font-weight: bold;  
}
```

粗体字

```
li {  
  font-weight: normal;  
}
```

正常字体

```
li {  
  font-weight: 800;  
}
```

Keywords: normal (默认), bold (粗体), bolder (更粗), lighter (更细)  
100, 200, 300, 400(normal), 500, 600, 700(bold), 800, 900

```
em {  
    font-variant: small-caps; //大写字母小型化 ABCDE  
}
```

font-family  
font-size  
font-style  
font-weight  
font-variant  
color

---

```
li {  
    text-transform: uppercase;  
}
```

Keywords: uppercase(大写), lowercase (小写), capitalize (首字母大写), none

---

```
em {  
    text-decoration: underline;  
}
```

Keywords: underline(下划线) overline(上划线) line-through(删除) none

- 如何修改字母和单词间距

`<p style="letter-spacing: 0.5em;">`

abcdefghijkl <br>

中山大学计算机科学系

`</p>`

a b c d e f g h i j k

中 山 大 学 计 算 机 科 学 系

每个字符之间间隔半个字

`letter-spacing: -0.2em;`

abcdefghijkl

中山大学计算机科学系

`letter-spacing: 0em;`

abcdefghijkl

中山大学计算机科学系

`<p style="word-spacing: 2em;">`

abcd efg hijk <br>

中山大学 计算机 科学系

`</p>`

abcd efg hijk

中山大学 计算机 科学系

- 如何给文本添加阴影？

`<p style="text-shadow: -4px 5px 3px #555555 ;">`

abcd efg hijk `<br>`

中山大学 计算机 科学系

`</p>`

abcd efg hijk

中山大学 计算机 科学系

水平偏移量(-4px)  
`x-offset`

垂直偏移量(5px)  
`y-offset`

阴影模糊度(3px)  
`blur-radius`

阴影颜色(#555555)  
`color`

`<p style="text-shadow: -4px 5px 1px #555555, 4px -5px 1px #550000;">`

abcd efg hijk `<br>`

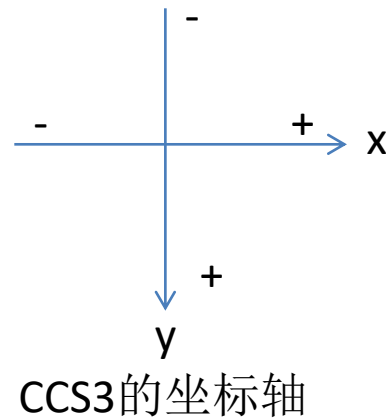
中山大学 计算机 科学系

`</p>`

abcd efg hijk

中山大学 计算机 科学系

`text-shadow: none`(默认)



# 格式化段落

## • 如何设定行高？

`<p style="line-height: 2em;">`      `font-size line-height font-family`  
`font: 100%/1.2 Georgia, "Times New Roman", serif;`

为了美化网页，原来直接使用HTML的表现元素或表现属性，例如：设置文字字体采用元素font，设置文字颜色使用元素color，文字对中使用元素center，设置整个网页默认的文字颜色和背景颜色可以使用body元素的text 属性和bgcolor属性。这些现在已经不建议用了，主要是为了防止这些表现元素和属性会淹没网页的实际内容，会使网页设计变得非常杂乱。

`</p>`

为了美化网页，原来直接使用HTML的表现元素或表现属性，例如：设置文字字体采用元素font，设置文字颜色使用元素color，文字对中使用元素center，设置整个网页默认的文字颜色和背景颜色可以使用body元素的text 属性和bgcolor属性。这些现在已经不建议用了，主要是为了防止这些表现元素和属性会淹没网页的实际内容，会使网页设计变得非常杂乱。

`line-height: 2em`

\* em按照当前字号定义行高。1em就是当前高度。还可以使用 px, percentage。如果没加单位，`line-height: 1.2`，实际使用的是em。文字显示注意行高产生的空白。



## • 如何设置首行缩进？

`<p style="text-indent: 2em;">`

还可以使用px, percentage

为了美化网页，原来直接使用HTML的表现元素或表现属性，例如：设置文字字体采用元素font，设置文字颜色使用元素color，文字对中使用元素center，设置整个网页默认的文字颜色和背景颜色可以使用body元素的text 属性和bgcolor属性。这些现在已经不建议用了，主要是为了防止这些表现元素和属性会淹没网页的实际内容，会使网页设计变得非常杂乱。

`</p>`

首行缩进2个字

为了美化网页，原来直接使用HTML的表现元素或表现属性，例如：设置文字字体采用元素font，设置文字颜色使用元素color，文字对中使用元素center，设置整个网页默认的文字颜色和背景颜色可以使用body元素的text 属性和bgcolor属性。这些现在已经不建议用了，主要是为了防止这些表现元素和属性会淹没网页的实际内容，会使网页设计变得非常杂乱。

如何实现悬挂格式？

## • 如何设置文本(内容)对齐?

`<p style="text-align: center; ">`

这些现在已经不建议用了，主要是为了防止这些表现元素和属性会淹没网页的实际内容，会使网页设计变得非常杂乱。

`</p>`

这些现在已经不建议用了，主要是为了防止这些表现元素和属性会淹没网页的实际内容，会使网页设计变得非常杂乱。

`text-align: left;`

这些现在已经不建议用了，主要是为了防止这些表现元素和属性会淹没网页的实际内容，会使网页设计变得非常杂乱。

`text-align: right;`

这些现在已经不建议用了，主要是为了防止这些表现元素和属性会淹没网页的实际内容，会使网页设计变得非常杂乱。

`<p style="text-align: justify; ">`

abc defg abc defg hijklmn opqrst  
uvwxyzabc defg defg hijklmn  
opqrstuvwxyzabc defg efg abc defg  
hijklmn opqrst uvwxyzabc defg defg  
hijklmn opqrsefg abc defg hijklmn  
opqrst uvwxyzabc defg defg hijklmn  
opqrsefg abc defg hijklmn opqrst  
uvwxyzabc defg defg hijklmn opqrsefg  
abc defg hijklmn opqrst uvwxyzabc  
defg defg hijklmn opqrs

abc defg abc defg hijklmn opqrst uvwxyzabc defg defg hijklmn  
opqrstuvwxyzabc defg efg abc defg hijklmn opqrst uvwxyzabc defg defg  
hijklmn opqrsefg abc defg hijklmn opqrst uvwxyzabc defg defg hijklmn  
opqrsefg abc defg hijklmn opqrst uvwxyzabc defg defg hijklmn opqrsefg abc  
defg hijklmn opqrst uvwxyzabc defg defg hijklmn opqrs

两端对齐

其他取值: start,end,match-parent

`</p>`

属性text-rendering可以设置连笔模式

## • 如何设置段落之间的距离？

`<p>` 为了美化网页，原来直接使用HTML的表现元素或表现属性，例如：设置文字字体采用元素font，设置文字颜色使用元素color，文字对中使用元素center，设置整个网页默认的文字颜色和背景颜色可以使用用body元素的text 属性和bgcolor属性。

`</p>`

`<p>` 这些现在已经不建议用了，主要是为了防止这些表现元素和属性会淹没网页的实际内容，会使网页设计变得非常杂乱。

`</p>`

为了美化网页，原来直接使用HTML的表现元素或表现属性，例如：设置文字字体采用元素font，设置文字颜色使用元素color，文字对中使用元素center，设置整个网页默认的文字颜色和背景颜色可以使用用body元素的text 属性和bgcolor属性。

这些现在已经不建议用了，主要是为了防止这些表现元素和属性会淹没网页的实际内容，会使网页设计变得非常杂乱。



```
p {  
    margin-top: 0;  
    margin-bottom: 0;  
}
```

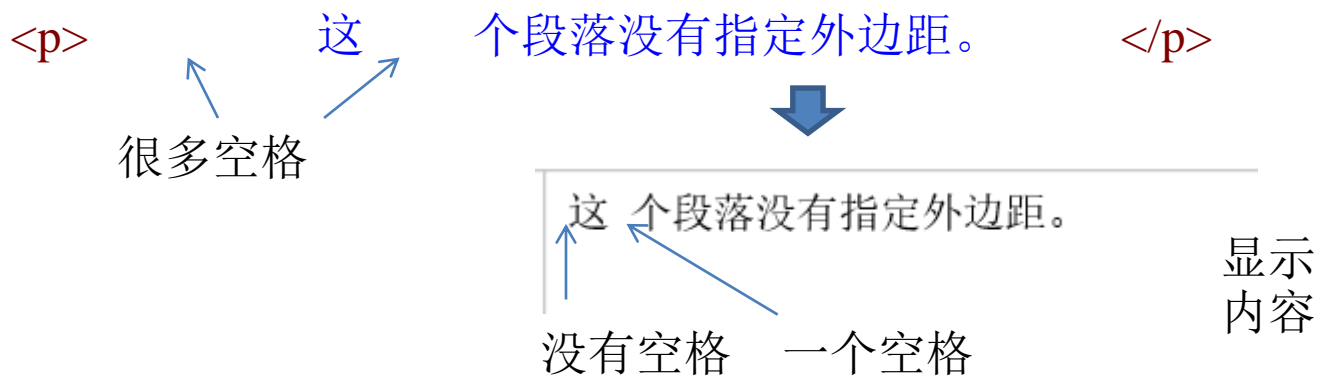
为了美化网页，原来直接使用HTML的表现元素或表现属性，例如：设置文字字体采用元素font，设置文字颜色使用元素color，文字对中使用元素center，设置整个网页默认的文字颜色和背景颜色可以使用用body元素的text 属性和bgcolor属性。  
这些现在已经不建议用了，主要是为了防止这些表现元素和属性会淹没网页的实际内容，会使网页设计变得非常杂乱。

\* 后面还会讲margin

## • 如何处理空白符和断句

P元素和div元素会自动删除文本前后的空白，并把文本中间连续多个空白合并为一个。

默认样式



加上样式

```
p {  
  white-space: pre;  
}
```

保存源码中的空白和回车



加上自动换行

```
p {  
  white-space: pre-wrap;  
}
```

自动换行

这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。

也不保留源码中的空白和回车

```
p {  
  white-space: nowrap; /*nowrap不自动换行 normal 恢复正常 */  
}
```

这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。

不自动换行(nowrap)

**word-wrap:** normal | hyphenate | break-word; /\*只在空格和连字符处断句|同上|可以在单词内断句\*/  
**word-break:** normal | break-all | keep-all; /\*默认|可以在单词内断句|只在空格和连字符处断句\*/

- word-wrap取值hyphenate时与word-break取值keep-all功能相同。
- 属性hyphens:none|manual|auto也可以用于处理断句问题。
- 只在空格和连字符处断句时会造成长单词溢出盒子。

# 盒模型(box model)

- ❑ 概述: margin(-TopRightBottomLeft), padding(-TRBL), border(-TRBL+style|color|width), box-sizing width, height, display, visibility
- ❑ 设置背景色或背景图: background-color, background-image, background-repeat, background-clip, background-origin, background-attachment, background-size, background-position; linear-gradient, repeating-linear-gradient, radial-gradient, repeating-radial-gradient
- ❑ 修饰边框: border-radius, box-shadow, border-image
- ❑ 列表样式: list-style-type, list-style-position, list-style-image
- ❑ 表格样式: border-collapse, border-spacing, empty-cells, table-layout
- ❑ 垂直对齐: vertical-align
- ❑ 宽度和高度: auto, percentage, length (px, em)

# • 概述

```
<div style="border:4px solid black">  
<div>
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent aliquam, justo convallis luctus rutrum, erat nulla fermentum diam, at nonummy quam ante ac quam. </div>

```
<div>
```

Maecenas urna purus, fermentum id, molestie in, commodo porttitor, felis. Nam blandit quam ut lacus. Quisque ornare risus quis ligula.

```
</div>
```

```
</div>
```



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent aliquam, justo convallis luctus rutrum, erat nulla fermentum diam, at nonummy quam ante ac quam.

Maecenas urna purus, fermentum id, molestie in, commodo porttitor, felis. Nam blandit quam ut lacus. Quisque ornare risus quis ligula.

```
<div style="border:4px solid black">  
<div style="border:4px solid red">
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent aliquam, justo convallis luctus rutrum, erat nulla fermentum diam, at nonummy quam ante ac quam. </div>

```
<div style="border:4px solid blue">
```

Maecenas urna purus, fermentum id, molestie in, commodo porttitor, felis. Nam blandit quam ut lacus. Quisque ornare risus quis ligula.

```
</div>
```

```
</div>
```

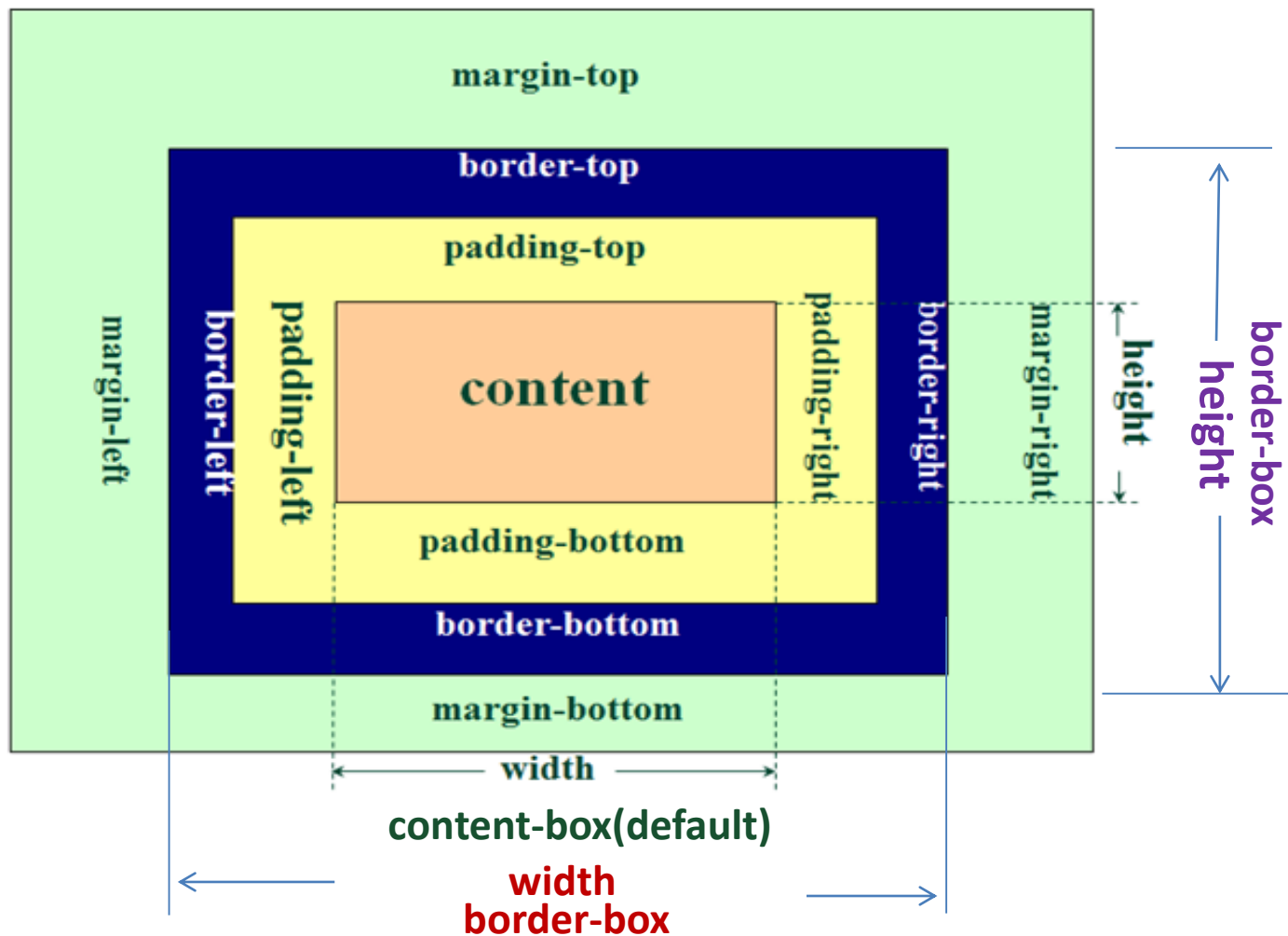


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent aliquam, justo convallis luctus rutrum, erat nulla fermentum diam, at nonummy quam ante ac quam.

Maecenas urna purus, fermentum id, molestie in, commodo porttitor, felis. Nam blandit quam ut lacus. Quisque ornare risus quis ligula.

如何使边界与内容之间、边界和边界之间留下空白？

盒模型包含内容(content)、内边距(padding)、边框(border)、外边距(margin)



- margin是边框之外的空白区，用于与其他元素的border、padding或content保持一定的距离。
- padding是内容和其边框线之间的空白。背景颜色可以渗入padding区但是不能渗入margin区。
- width和height默认指内容的宽度和高度。
- **box-sizing**: **border-box** | **content-box**(默认);  
Firefox: -moz-box-sizing Safari或chrome或android:-webkit-box-sizing IE: -ms-box-sizing



# • border的使用

border有三个属性:

border-style (边界线型)

border-width (边界宽度)

border-color (边界颜色)

border可以分边定义样式, 例如:

border-left-style (左边界线型)

border-left-width (左边界宽度)

border-left-color (左边界颜色)

边界线型 边界宽度 边界颜色

`<div style="border: solid 4px blue">`

`border: currentColor;`

Maecenas urna purus, fermentum id, molestie in, commodo porttitor, felis. Nam blandit quam ut lacus. Quisque ornare risus quis ligula.

`</div>`

Maecenas urna purus, fermentum id, molestie in, commodo porttitor, felis. Nam blandit quam ut lacus. Quisque ornare risus quis ligula.

style的取值:

外嵌      none, hidden, dotted, dashed, solid, double, groove, ridge, inset, outset, inherit

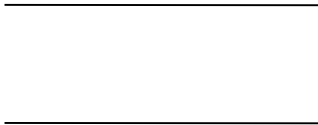
槽      脊      内嵌

- 简化border的设置

```
border-top: 2px solid black;
border-left: 2px solid black;
border-right: 2px solid black;
border-bottom: 3px dotted red;
```



```
border: 2px solid black;
border-bottom: 3px dotted red;
```



```
border: 1px solid black;
border-left: none;
```



```
border: 1px solid;
border-color: black red green blue;
/*Top, Right, Bottom, Left*/ TRouBLe
```



```
border: 1px solid blue;
border-left-color: red;
```



```
width: 400px; height: 200px;
border: 50px solid;
border-color: black transparent green red;
```

## • margin和padding的使用方法

margin和padding都可以分边单独定义，例如：

```
div { margin-top:10px;margin-right:20px;  
      margin-bottom:30px;margin-left:40px; }
```

也可以简写为：

```
div { margin:10px 20px 30px 40px }
```

margin:10px

margin:10px 20px

margin: 0 auto

margin:10px 20px 30px

如果没有边框和背景颜色，我们很难区分两块内容之间的空白是由padding还是margin造成的。

单位

em

px

percentage

响应式Web设计

当前字号

像素

比例，参照父窗口大小（border也一样）

- margin-边距重叠(margin collapsing)

```
<div style="margin-bottom:10px">  
  Lorem ipsum ...  
</div>  
<div style="margin-top:20px">  
  Maecenas urna ...  
</div>
```



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent aliquam, justo convallis luctus rutrum, erat nulla fermentum diam, at nonummy quam ante ac quam.

20px

Maecenas urna purus, fermentum id, molestie in, commodo porttitor, felis. Nam blandit quam ut lacus. Quisque ornare risus quis ligula.

```
<div style="border:solid 1px blue">  
  <div style="margin-top:10px">  
    <div style="margin-top:20px">  
      Lorem ipsum ...  
    </div>  
  </div>  
</div>
```



20px  
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent aliquam, justo convallis luctus rutrum, erat nulla fermentum diam, at nonummy quam ante ac quam.

collapsing: 坍塌, 折叠, 重叠

一个元素的margin会与其他元素（非float元素）的margin合并到一起，引起margin重叠现象。浮动元素的margin不会重叠。解释：考虑两个元素之间留空时不考虑兄弟元素或子元素的margin。

- margin负值

margin折叠：两正值和两负值都取绝对值大的，一正一副取和值

style="border:solid 1px black;border-bottom:none;width:300px"

<h5>EXTRA EXTRA EXTRA EXTRA</h5>

<h1>Main Content Main ContentMain</h1>

style="margin-top:-28px"

<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent aliquam, justo convallis luctus rutrum, erat nulla fermentum diam, at nonummy quam ante ac quam. Maecenas urna purus, fermentum id, molestie in, commodo porttitor, felis.</p>

<h2>H2 level heading</h2>

<p> The background color on this div will only show for the length of the content. If you'd like a dividing line instead, place a border on the left side of the #mainContent div if it will always contain more content.</p>

style="margin-top:-24px;border-top:solid 1px black;"

# • 块级元素和内联元素

- 块级元素默认不与其它显示在同一行，而内联元素则可以。
- 浏览器把块级元素和内联元素都当成盒子(box)处理，可以设置字体、颜色、背景，边框线。
- 块级可以设置height、width、margin和padding，除了img元素，内联元素只能设置左右的margin和padding，其它都不行。img元素可以设置高度和宽度以及上下的margin。
- 块级元素和内联元素可以相互转换：

`display: block;`

把元素变为块级元素

`display: inline;`

把元素变为内联元素

`display: inline-block;`

使元素保持为内联元素，但是可以像块级元素一样设置height、width、上下margin和padding。

## 例 1

<p> 这是一个没有指定<span>外边距的段落。 这是一个没有指定外边距的段落。  
</span> 这是一个没有指定外边距的段落。  
这是一个没有指定外边距的段落。 </p>

```
span {  
    height: 40px;  
    border: 1px solid black;  
}
```

这是一个没有指定外边距的段落。 这是一个没有指定外边距的段落。 这是一个没有指定外边距的段落。 这是一个没有指定外边距的段落。

## 例 2

```
span {  
    display: block;  
    height: 40px;  
    border: 1px solid black;  
}
```

这是一个没有指定外边距的段落。 这是一个没有指定外边距的段落。

这是一个没有指定外边距的段落。 这是一个没有指定外边距的段落。

例 3 `span {`  
    `display: inline-block;`  
    `height: 40px;`  
    `border: 1px solid black;`  
`}`

这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。

例 4

`span {`  
    `display: none;`      `visibility: hidden;` // 其它取值: visible, collapse  
`}`

元素不显示，完全不占用空间

`hidden`的元素在文档流依然占据原来的空间

靠`display:inline|inline-block`就可以完成布局了吗？



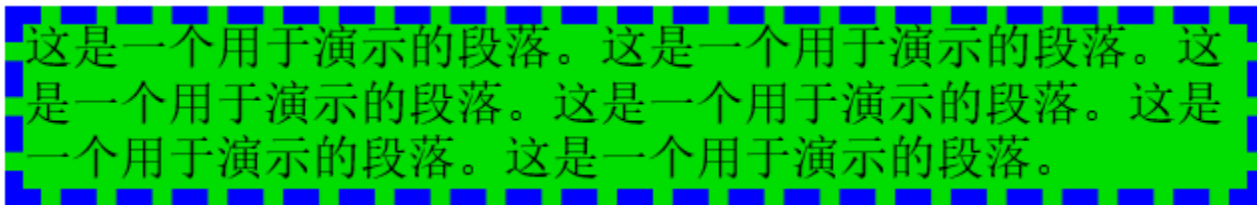
# 设置背景色或背景图

- 定义背景色

```
div { background-color: #0000FF; }      /* 蓝色背景 */  
div { background-color: rgba(0,0,255,0.5); /* 半透明 */ }
```

- 当边框设置为dashed，背景色默认会渗入虚线内

```
div {  
    border: 6px dashed blue;  
    background-color: #0D0;  
}
```



这是一个用于演示的段落。这是一个用于演示的段落。这是一个用于演示的段落。这是一个用于演示的段落。这是一个用于演示的段落。这是一个用于演示的段落。这是一个用于演示的段落。这是一个用于演示的段落。

## • 其它背景属性

```
<html><head>  
<style type="text/css">
```

```
div { width: 1000px;
```

```
height: 800px;
```

```
padding: 40px;
```

```
border: dashed 10px blue;
```

```
background-image: url('sysu.png'); /* 使用的背景图像 */
```

```
background-origin: padding-box; /* 填充的第一个图的位置: border-box (默认),content-box */
```

```
background-repeat: no-repeat; /* 重复填充整个区域: repeat(默认),repeat-x, repeat-y*/
```

```
background-clip: padding-box; /* 填充范围: border-box (默认),content-box */
```

```
background-position :center left; /* 背景图的对齐位置 (水平、垂直), 取值见附录*/
```

```
background-attachment: fixed; /* 背景图是否随区域内容滚动: scroll(默认),fixed */
```

```
background-size: cover; /* 背景图大小: cover|contain|percentage|length (宽 高) */
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<div>这是内容, 这是内容, ... </div>
```

```
</body></html>
```

background-position :-150px -50px; //宽度 高度

background-position :50% 80%; //左上角0% 0%, 右下角100% 100% fixed时有效

background-position-x,background-position-y

组合方式: background: url('sysu.png') #CCC repeat-x bottom;

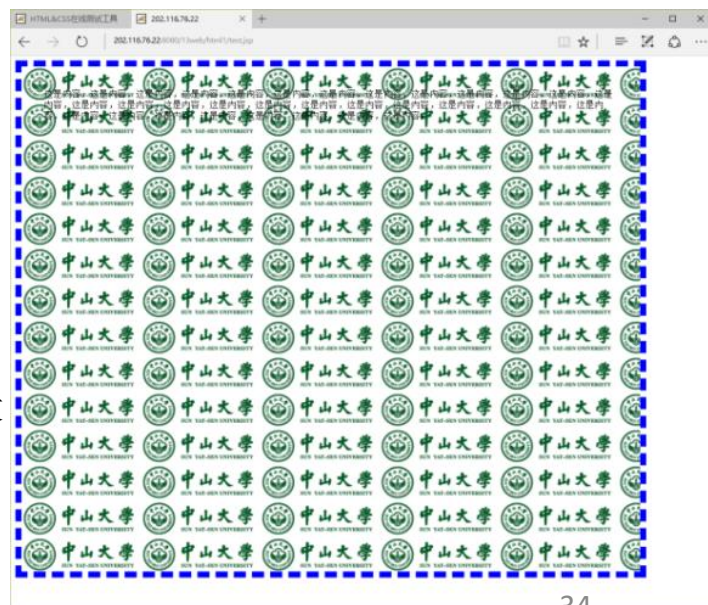
\* clip-path可以剪切出不同形状的背景

### background-size:

**cover**缩放背景图到完全覆盖整个元素并最小化, 超出范围的部分看不到。

**contain**把背景图完全纳入整个背景区并最大化, 背景区中可能会留空白。它与cover都会保持图像的高宽比不变。

**percentage**和**length**设置背景图的高度(宽度)时会等比例缩放宽度(高度)。percentage是参照当前背景区的比例。

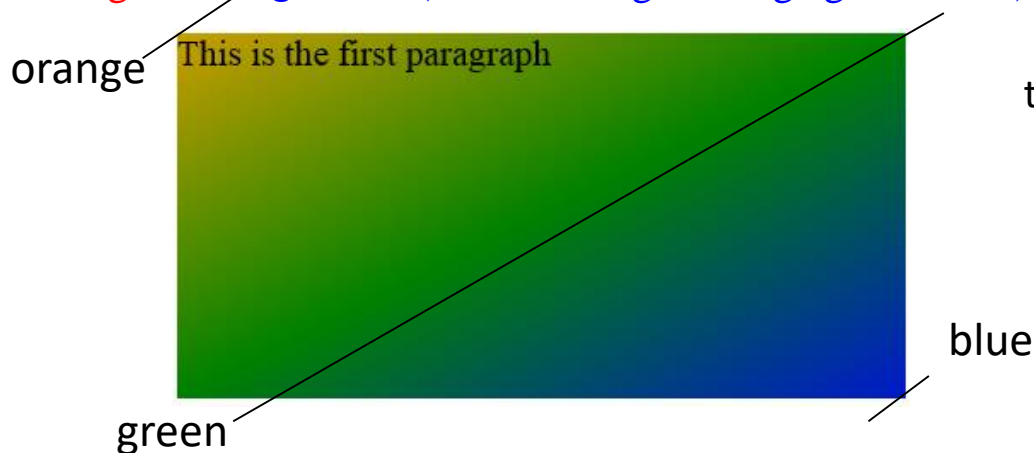


参考1 参考2

- 渐变

[参考](#)

`background-image:-webkit-linear-gradient(to bottom right,orange,green, blue); /* 安卓 */`  
`background-image:linear-gradient(to bottom right,orange,green, blue); /* 其它 */`



to bottom right 在这里近似 115deg

`background:linear-gradient(90deg,orange,green,blue); // 组合分量`

90deg 同 to right



其它颜色方式: `rgba(0,0,255,0.3)`  
`#DDD`

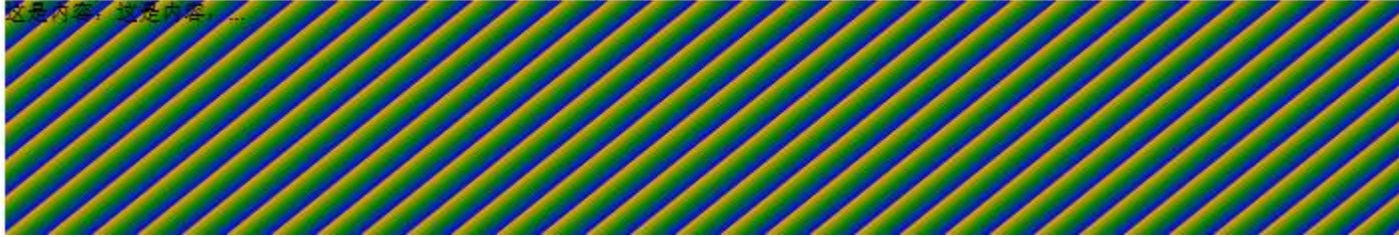
`background: blue; // 放在前面, 用于不支持 linear-gradient 的浏览器`

`background: linear-gradient(to right, orange 0% ,green 10%,blue 100%); /* 百分数(或px)为位置 */`



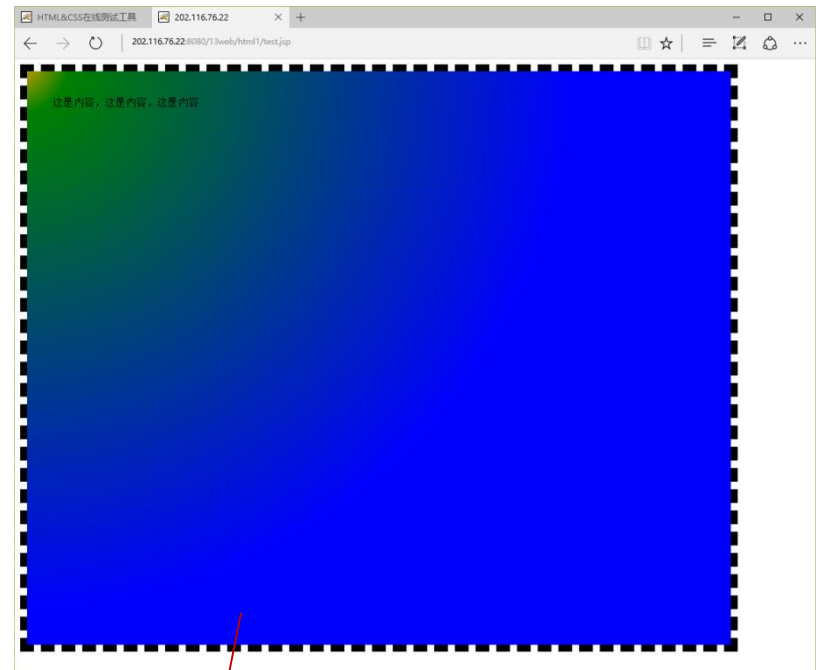
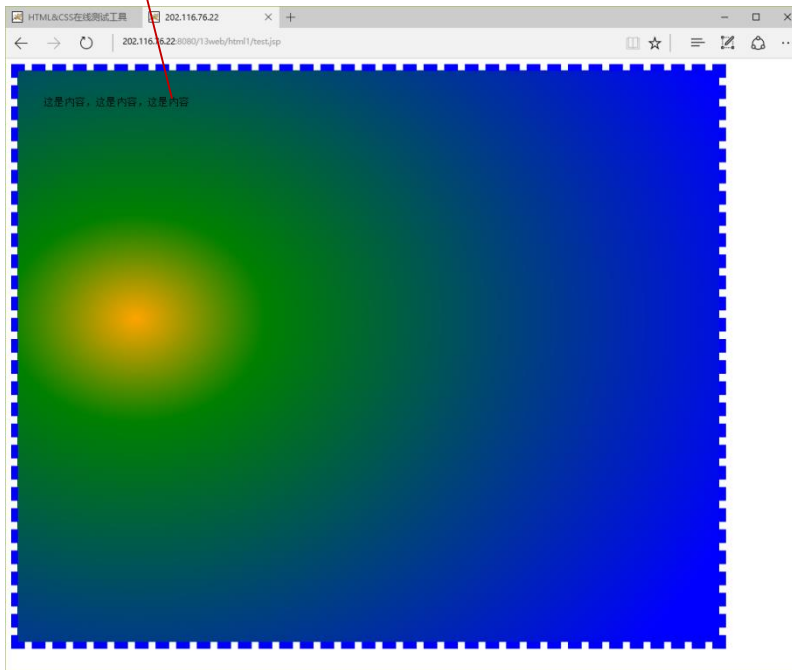
green

**background:** repeating-linear-gradient(to bottom right, orange 10px,green 20px,blue 30px);



范围:width height    中心: x, y    中心颜色--->中间颜色--> 最外颜色

**background-image:** radial-gradient(1000px 800px at 200px 400px, yellow,green 20%,blue 100%);

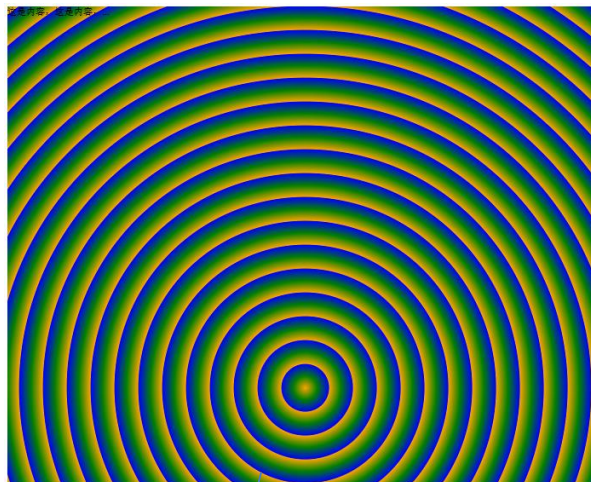
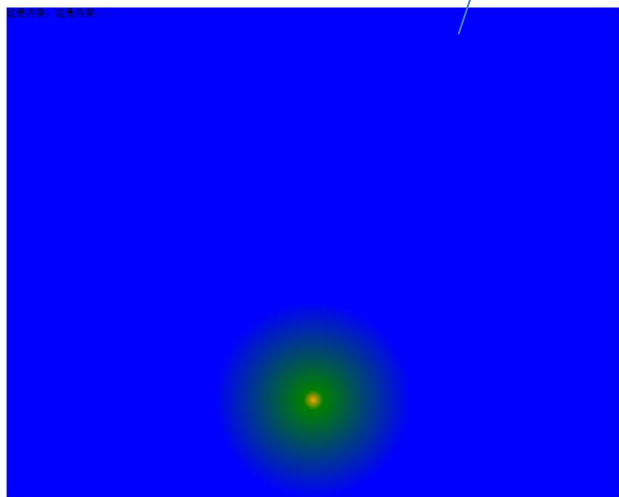


**background:** radial-gradient(70% 90% at top left, orange 0% ,green 10%,blue 100%);

**background:** -ms-radial-gradient(50% 80%,ellipse closest-side, orange 0% ,green 10%,blue 100%);

圆心 (或center center或10px 100px)

半径 (或circle closest-side或farthest-side  
或closest corner或farthest-corner  
或20% 80%或60px 100px)



**background:** -ms-repeating-radial-gradient(50% 80%,circle closest-side, orange 0 ,green 20px,blue 40px);

```
div:before {content:"abcde"; background-image: url(" sysu.png"); display:block}
```

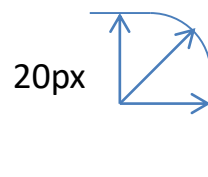
```
div:before {content: url(" sysu.png");}
```

```
div{background:url(" sysu.png ") left repeat-y, url(" sysu.png ") right repeat-y;} /* 多重背景 */
```

# 修饰边框

- 创建圆角

```
p {  
    padding: 20px;  
    margin: 20px;  
    border: 1px solid blue;  
    border-radius: 20px;           /* 四个圆角的半径 */  
}
```



<p>这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。</p>



这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。



```
p {  
    padding: 20px;  
    margin: 20px;  
    border: 1px solid black;  
    border-radius: 0px 20px 10px 0px;    /* 四个圆角: top-left, top-right,  
                                           bottom-right, bottom-left */  
}
```



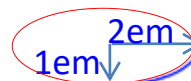
这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。

最小样式属性:

```
border-top-right-radius: 20px;
```

```
p {  
    padding: 20px;  
    margin: 20px;  
    border: 1px solid black;  
    /* x方向和y方向半径 */  
    border-radius: 2em/1em;  
}
```

这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。





- 添加阴影

```
p {  
  padding: 20px;  
  margin: 20px;  
  border: 1px solid black;  
  box-shadow: -6px 8px 4px #808080;  
}
```

模糊半径(模糊度, 可选)  
阴影颜色  
水平偏移量  
垂直偏移量

这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。

水平偏移量: -6px

垂直偏移量: 8px

嵌入式阴影

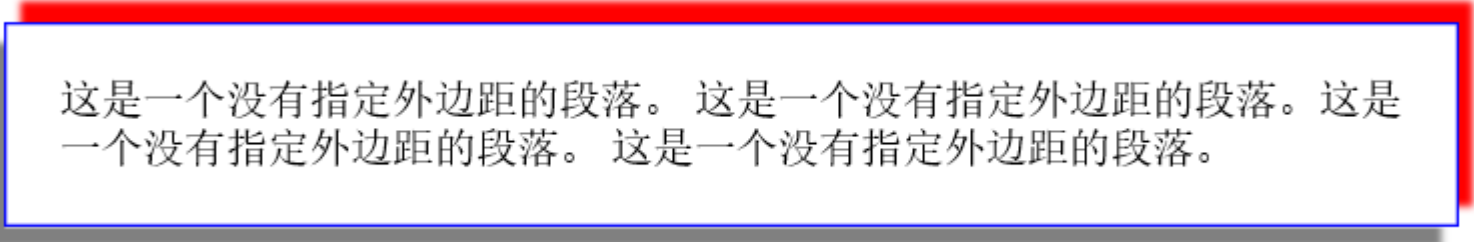
```
box-shadow: inset -6px 8px 4px #808080; /*默认为outset阴影*/
```

这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。

格式 `box-shadow: inset h-shadow v-shadow blur spread color; /*spread (可选) 为阴影距离*/`

## 多重阴影

**box-shadow:** -6px 8px 4px #808080, 6px -8px 4px red;



这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。这是一个没有指定外边距的段落。

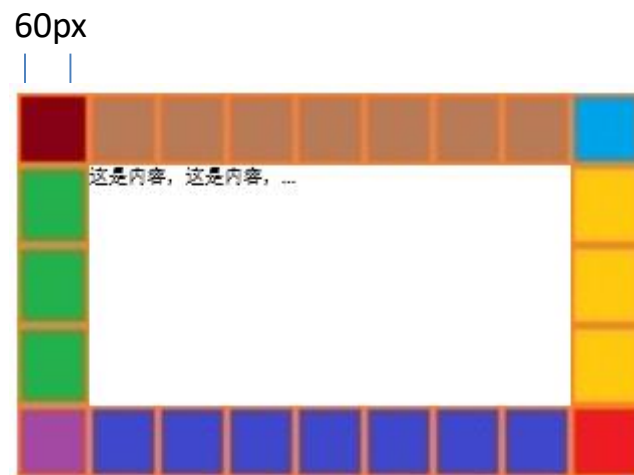
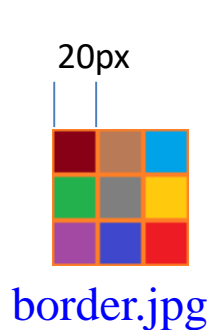
不同浏览器画出的阴影会有差别。

Android采用: -webkit-box-shadow: -6px 8px 4px #808080

- 用图像做边界

```
div {
  width: 400px;
  height: 200px;
  border: solid 60px;
  border-image: url('border.jpg') 20 20 20 20 round;
}
```

<div>这是内容，这是内容，... </div>



\* 4个20表示四条边（TRBL）的截取片断范围. Round（上图）表示重复片段，stretch（下图）只采用一个片段。



**border-image: url('border.jpg') 20 20 20 20 stretch;**

\*有的浏览器支持四个方向不同属性stretch stretch round round

# 列表样式

```
ol {  
  list-style-type: decimal;  
}
```

decimal: 1, 2, 3, ...  
decimal-leading-zero: 01, 02, ...  
upper-alpha: A, B, C, ...  
lower-alpha: a, b, c, ...  
upper-roman: I, II, III, ...  
lower-roman: i, ii, iii, ...

**none**

`list-style-position: inside;` —————

`list-style-position: outside;` —————

`list-style-image: url(bullet.gif);`

```
ul {  
  list-style-type: disc;  
}
```

disc: 实心圆点  
square: 实心方块  
circle: 空心圆

• abcd efgh ijkl abcd efgh ijkl abcd  
efgh ijkl abcd  
• abcd efgh ijkl abcd efgh ijkl abcd  
efgh ijkl abcd

• abcd efgh ijkl abcd efgh ijkl abcd  
efgh ijkl abcd  
• abcd efgh ijkl abcd efgh ijkl abcd  
efgh ijkl abcd

# 表格样式

## 表格属性

### table 无边框

第一行第一列 第一行第二列 第一行第三列 第一行第四列  
第二行第一列 第二行第二列 第二行第三列 第二行第四列  
第三行第一列 第三行第二列 第三行第三列 第三行第四列

没有边框

## 表格属性

### table 单元有边框

第一行第一列	第一行第二列	第一行第三列	第一行第四列
第二行第一列	第二行第二列	第二行第三列	第二行第四列
第三行第一列	第三行第二列	第三行第三列	第三行第四列

td {border:solid 1px black}

## 表格属性

### table 表和单元有边框

第一行第一列	第一行第二列	第一行第三列	第一行第四列
第二行第一列	第二行第二列	第二行第三列	第二行第四列
第三行第一列	第三行第二列	第三行第三列	第三行第四列

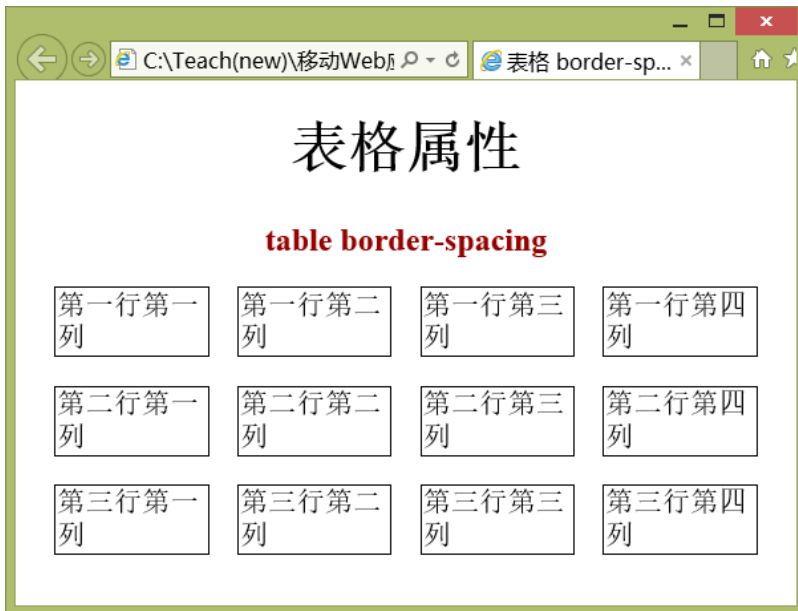
table, td {border:solid 1px black}  
表格对中: table {margin:0 auto}

## 表格属性

### table 合并边框

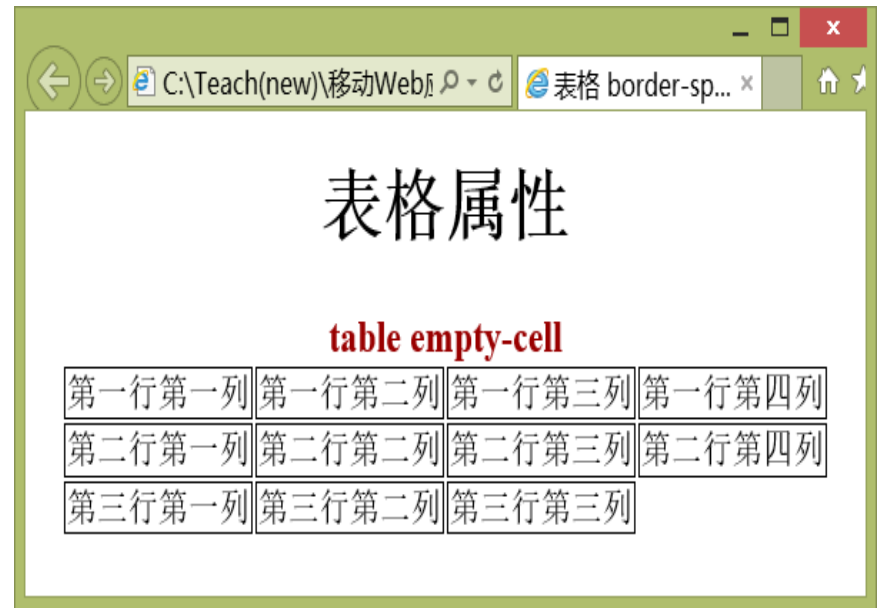
第一行第一列	第一行第二列	第一行第三列	第一行第四列
第二行第一列	第二行第二列	第二行第三列	第二行第四列
第三行第一列	第三行第二列	第三行第三列	第三行第四列

td {border:solid 1px black}  
table {border-collapse:collapse}  
\* border-collapse的另一个取值:separate



加大单元格之间的距离

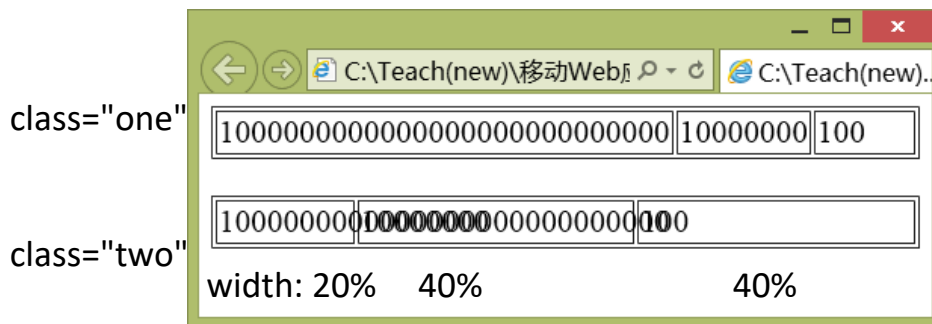
```
table {border-spacing:1em}
td {border:solid 1px black}
```



是否显示空单元格（没有内容）

```
td {border:solid 1px black;empty-cells:hide}
```

\* empty-cells的另一个取值为show(默认)



```
table {border:solid 1px black;} 列宽度由单元格内容设定
table.one
```

```
{
  table-layout: automatic;width:100%
}
```

列宽度由单元格宽度和表格宽度设定，单元格的宽度之和超出表格宽度之后，单元格将重叠起来。

```
table.two
{
  table-layout: fixed;width:100%
}
```

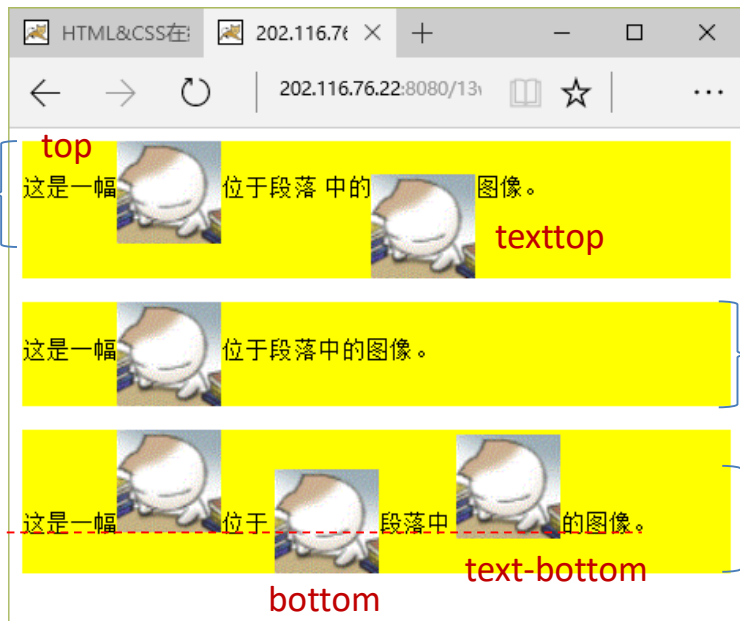
# 垂直对齐

采用vertical-align可以对处于同一行上的对象进行垂直对齐。

vertical-align: baseline, top, bottom, text-top, text-bottom, middle, %, length, sub, super  
/\*具体解释见附录\*/

```
<html>
<head>
  <style type="text/css">
    img.top { vertical-align: top; }
    img.texttop { vertical-align: text-top; }
    img.middle { vertical-align: middle; }
    img.baseline { vertical-align: baseline; }
    img.bottom { vertical-align: bottom; }
    img.textbottom { vertical-align: text-bottom; }
    p { line-height: 4em; background: yellow }
  </style>
</head>
<body>
  <p>这是一幅&ltimg class="top" src="i/cute.gif" />位于段落
    中的&ltimg class="texttop" src="i/cute.gif" />图像。</p>
  <p>这是一幅&ltimg class="middle" src="i/cute.gif" />
    位于段落中的图像。</p>
  <p>这是一幅&ltimg class="baseline" src="i/cute.gif" />位于
    段落中
    的图像。</p>
</body>
</html>
```

line-height: 4em



\*一边对齐之后，图片另一边的高度会改变行高。  
\*text-bottom要比文字低一点(2个像素)，用于做下划线的。

下标: vertical-align: sub  
上标: vertical-align: super

\* 一段文字(span)只有采用自然行高(未设置line-height)时行中垂直对齐才有效。

# 布局(layout)

- 设置尺寸: height,width,box-sizing,max-width,min-width,max-height, min-height
- 溢出: overflow, overflow-x, overflow-y
- 浮动元素: float, clear
- 定位技术: position
- 放置层次: z-index



# 设置尺寸与溢出

设置盒模型的高度和宽度：

```
height: 60px;
height: auto;      /* 默认（包裹内容） */
height: 6em;        /* 以当前文字大小为参照 */
width: 30%;         /* 以父元素或浏览器客户区的宽度为参照 */
width: 6rem;        /* 以根元素html的字体宽度为参照 */
```

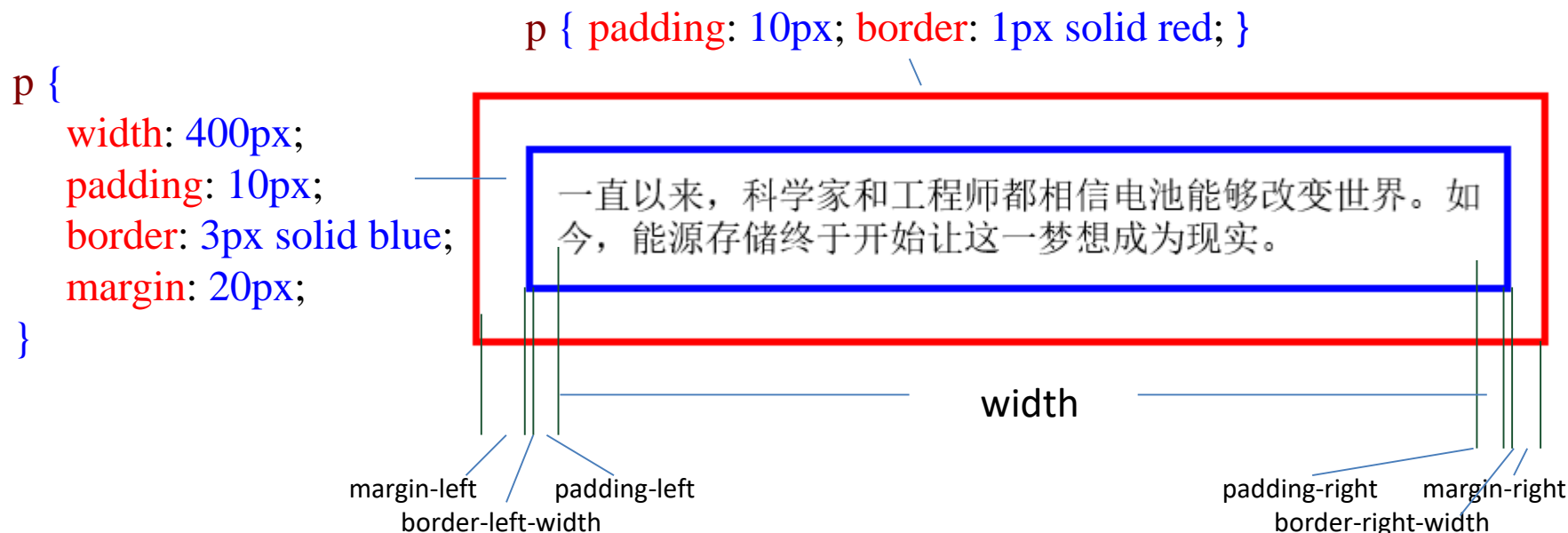


设置根标签html文字大小后（浏览器默认为16px），rem是该大小的倍数。

```
html {
    font-size: 20px;
}

p {
    width: 2rem; /*40px*/
}
```

height和width默认只是内容的高度和宽度，并非盒子的整个高度和宽度。



整个盒子的宽度 = border-left-width + padding-left + width  
+ padding-right + border-right-width

\* 这里盒子的宽度不考虑margin

上面的例子的盒子宽度=?      3px+10px+400px+10px+3px=426px

如果希望width包含盒子的宽度，就要采用CSS3的样式box-sizing。如果box-sizing取值border-box，则width就是盒子的宽度（不含margin）。

box-sizing {  
border-box: width包含内容宽度、左右padding和左右border-width  
content-box: width只包含内容宽度（默认）

width采用percentage，是指则其宽度会随着父元素宽度的变化而变化。如果使用很大的桌面显示器，内容可能会变得太宽而难以阅读。这时可以使用样式max-width（最大宽度）进行控制。如果把内容宽度变得太小，也会使人难以阅读。这时可以使用样式min-width（最小宽度）进行控制。

```
p {  
  width: 40%;  
  max-width: 1000px;  
  min-width: 200px;  
}
```

width: 500px; /\* older browsers \*/  
width: calc(50% - 20px); /\* calc可用于计算 \*/  
/\* -webkit-, -moz- \*/

整个父元素宽度的50%      左右padding的宽度，没有border

高度定义与宽度类似。如果宽度受限且自动折行，内容很多时将一直向下延伸。如果高度也受限，内容就可能会溢出边框，溢出部分会覆盖下面的元素。这可以通过设置样式`overflow`为`hidden`和`scroll`来解决。

```
p {  
  width: 80%;  
  max-height: 60px;  
  border: 3px solid blue;  
}
```

一直以来，科学家和工程师都相信电池能够改变世界。如今，能源存储终于开始让这一梦想成为现实。

`overflow:scroll;` 显示滚动条

一直以来，科学家和工程师都相信电池能够改变世界。如今，

`overflow:hidden;` 不显示溢出部分

一直以来，科学家和工程师都相信电池能够改变世界。如今，能源存储终于开始让这一梦想

`overflow:auto;` 溢出时才显示滚动条

一直以来，科学家和工程师都相信电池能够改变世界。如今，能源存储终于开始让

`overflow:visible;` 全部显示出来（默认）

一直以来，科学家和工程师都相信电池能够改变世界。如今，能源存储终于开始让这一梦想成为现实。

单独控制x方向和y方向溢出：`overflow-x,overflow-y`。取值与`overflow`相同。

`text-overflow: clip|ellipsis|string;` /\* 剪切掉(默认)|用省略号代替|用字符串代替溢出内容\*/

# 设置浮动元素

[参考](#)

- 块级元素默认从上到下依次摆放，这是普通文档流(document flow)的显示方式。如果希望块级元素并列显示，可以把它们定义为浮动元素（`float: left|right|none|inherit`），让它们脱离普通文档流。
- 设置浮动属性是目前用得最多的多列布局方法。不过最初设计浮动时并不是为了布局的，而是为了实现文字矩形环绕的特效。下面右图采用 `shape-outside` 实现任意形状的环境。

埃菲尔铁塔

`float:right`

埃菲尔铁塔的结构体系既直观又简洁:底部是分布在每边128米长底座上的4个巨型倾斜柱墩,倾角54度,由57.63米高度处的第一层平台联系支承:第一层平台和115.73米高度处的第二层平台之间是4个微曲的立柱;再向上4个立柱转化为几乎垂直的、刚度很大的方尖塔,其间在276.13m高度处设有第三层平台;在300.65米高度处是塔顶平台,布置有电视天线。铁塔总重10000吨,承担这些重量的是4个坚固的直伸至下卧持力土层的沉箱基础。在1884年6月的时候,埃菲尔设计事务所的两位主要工程师Emile Nougier和Maurice Koechlin就有了设计一座超高塔的思想。初步的设计像一个巨大的塔架,有四个由格构梁架构成的支腿,分立支撑在基础上并在塔顶收在一起,其间布置等间距横梁联系。为了更加合意公众的评价,Nougier和Koechlin邀请建筑师Stephen Sauvestre对结构的表现形式作进一步处理。Sauvestre提议用石砌台座整饰塔脚,用具有纪念性的拱结构连接四个塔柱与第一层平台,每层平台设大型玻璃墙大厅,顶部采用球状造型等装饰手法来亮化结构。在其后的设计中工程师们进行了进一步的完善,像巨型拱之类的一些设计思想得到保留,并给铁塔以极富性格的表现。另外,横梁的设置得到简化并与塔柱优化以后的线形相协调,使铁塔结构受力更加简捷,形式更加优美,高扬的动势更加强烈。



埃菲尔铁塔

`float:right`

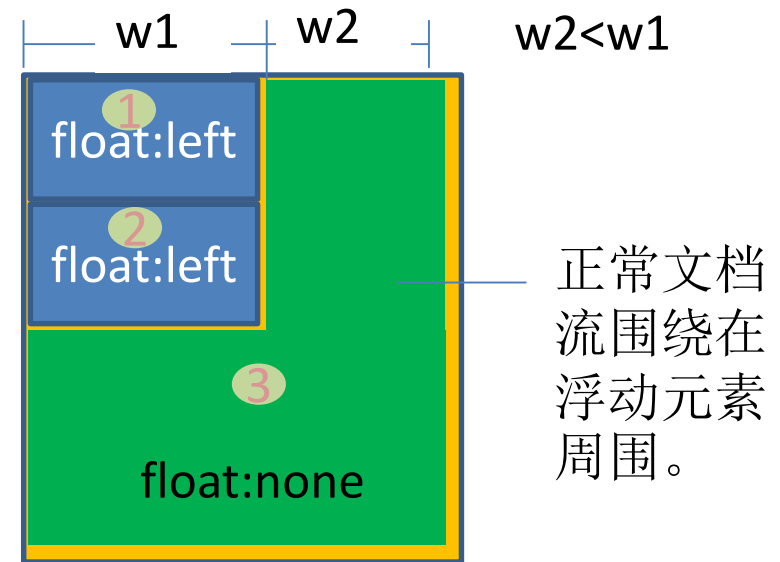
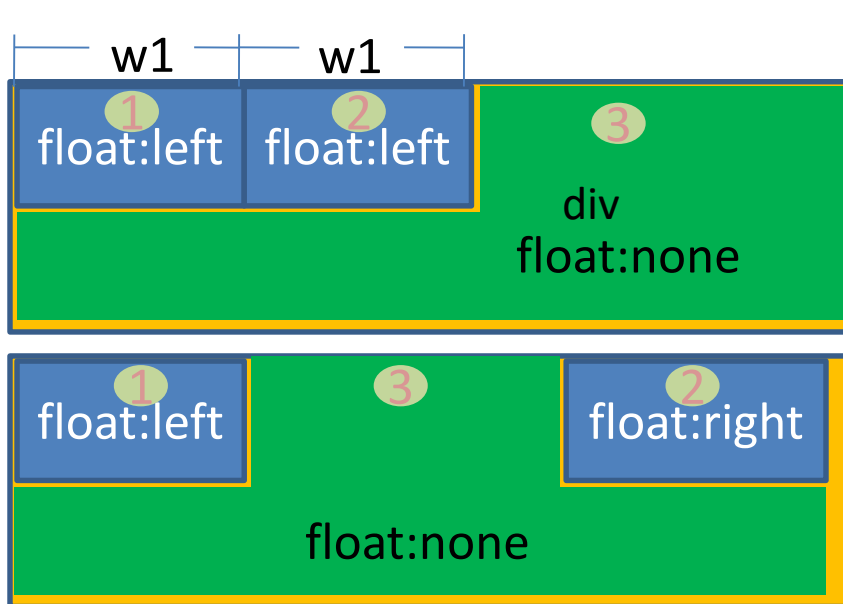
埃菲尔铁塔的结构体系既直观又简洁:底部是分布在每边128米长底座上的4个巨型倾斜柱墩,倾角54度,由57.63米高度处的第一层平台联系支承:第一层平台和115.73米高度处的第二层平台之间是4个微曲的立柱;再向上4个立柱转化为几乎垂直的、刚度很大的方尖塔,其间在276.13m高度处设有第三层平台;在300.65米高度处是塔顶平台,布置有电视天线。铁塔总重10000吨,承担这些重量的是4个坚固的直伸至下卧持力土层的沉箱基础。在1884年6月的时候,埃菲尔设计事务所的两位主要工程师Emile Nougier和Maurice Koechlin就有了设计一座超高塔的思想。初步的设计像一个巨大的塔架,有四个由格构梁架构成的支腿,分立支撑在基础上并在塔顶收在一起,其间布置等间距横梁联系。为了更加合意公众的评价,Nougier和Koechlin邀请建筑师Stephen Sauvestre对结构的表现形式作进一步处理。Sauvestre提议用石砌台座整饰塔脚,用具有纪念性的拱结构连接四个塔柱与第一层平台,每层平台设大型玻璃墙大厅,顶部采用球状造型等装饰手法来亮化结构。在其后的设计中工程师们进行了进一步的完善,像巨型拱之类的一些设计思想得到保留,并给铁塔以极富性格的表现。另外,横梁的设置得到简化并与塔柱优化以后的线形相协调,使铁塔结构受力更加简捷,形式更加优美,高扬的动势更加强烈。



```
clip-path: polygon(0 0, 100% 0, 100% 100%, 30% 100%);  
shape-outside: polygon(0 0, 100% 0, 100% 100%, 30% 100%);
```

\* [clip-path](#)可以剪切出不同形状的背景, [shape-outside](#)实现任意形状环绕 (IE、Edge不支持)

- 浮动元素一般需要定义宽度，并可以浮动到左边或右边。没有定义宽度的浮动元素会尽量占用父元素的宽度，内容不够时，与内容同宽。
- 如果前面的浮动元素摆放后留下的宽度足够，紧跟着的浮动元素可以与它并排放置。如果前面的是非浮动元素，紧跟着的浮动元素只能排在它的下面。
- 在布局非浮动元素时要像前面的浮动元素不存在一样，只是要求其内容不能与浮动元素的盒模型重叠，而其它部分(padding、margin和border)则可以与浮动元素重叠。
- 如果内联元素定义了float属性，它们会变为块级元素。



\* `display:inline-block`也可以实现div的并排摆放，可以用吗？内联元素不能包含块级元素

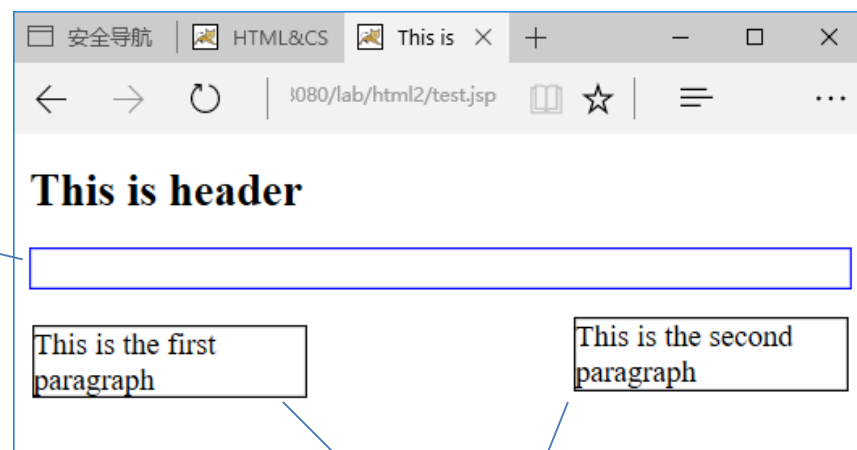
- 非浮动元素包含浮动元素时，会出现什么情况？

```

<!DOCTYPE html>
<html lang="zh-cn">
  <head>
    <meta charset="utf-8"/>
    <title>This is my title</title>
    <style>
      #c1 {border: solid 1px blue; padding-top: 20px}
      #p1, #p2 {width: 9em; border: solid 1px}
      #p1 {float: left; margin-top: 20px}
      #p2 {float: right;}
    </style>
  </head>
  <body>
    <h1>This is header</h1>
    <div id="c1">
      <p id="p1">This is the first paragraph</p>
      <p id="p2">This is the second paragraph</p>
    </div>
  </body>
</html>

```

父元素  
(div)  
高度塌陷



子元素 (p)

div 增加子元素 `<p style="clear:both">`  
 如果让浮动元素包含非浮动元素或  
 浮动元素，会怎么样？

clear: **both** | left | right  
 两边无浮动元素



## • 三列式布局



(1)

(3)

(2)

(4)

float:left  
左边区

float:none  
主区

float:right  
右边区

```
<!DOCTYPE html>
<html>
<head> <title>3列式布局</title>
<style>
    #container { width: 100%; background:aqua;}
    #first {
        float: left;width: 7em;padding: 0.5em;
    }
    #second {
        float: right;width: 7em;padding: 0.5em;
    }
    #third {
        border: 4px solid blue;
        background:red;
    }
    #fourth {
        border: 4px solid green;
        background: yellow;
    }
</style>
</head>
<body>
    <div id="container">
        <div id="first"> (1) 蒂姆进入大学... </div>
        <div id="second"> (2) 在这里年轻的... </div>
        <div id="third"> (3) 1989年仲夏之夜... </div>
        <div id="fourth"> (4) 在今天作为Web... </div>
    </div>
</body>
</html>
```



解决方法：（1）主区增加overflow: hidden



（2）边区添加背景 background: #CCC



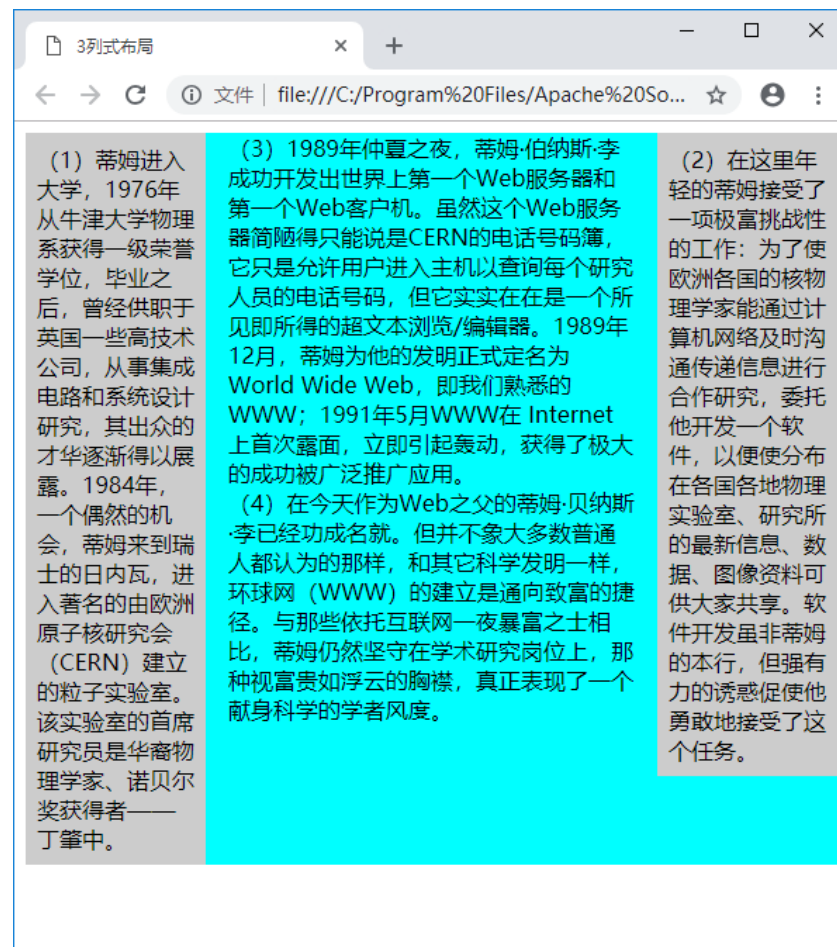
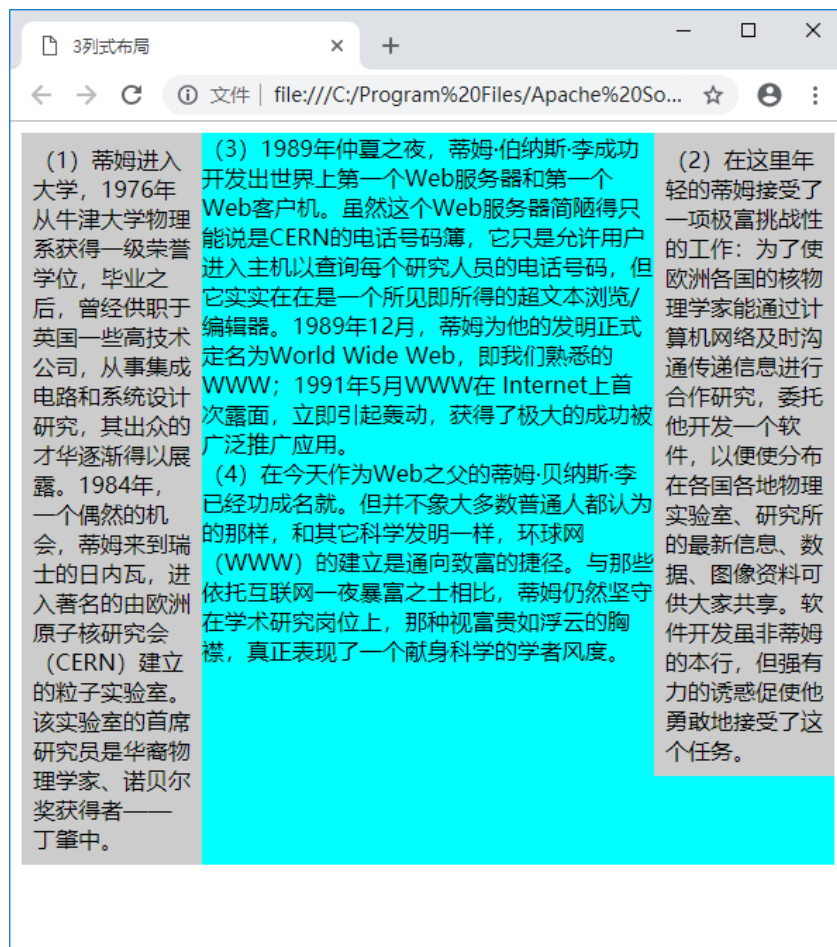
如何要让container（青色）的高度包含浮动框，即清除浮动？

- (1) 在container的最后添加一个空子元素：<div style="clear:both"></div>
- (2) #container:after {content:"";clear:both;display:block}
- (3) #container {overflow:hidden}
- (4) #container {height:400px}

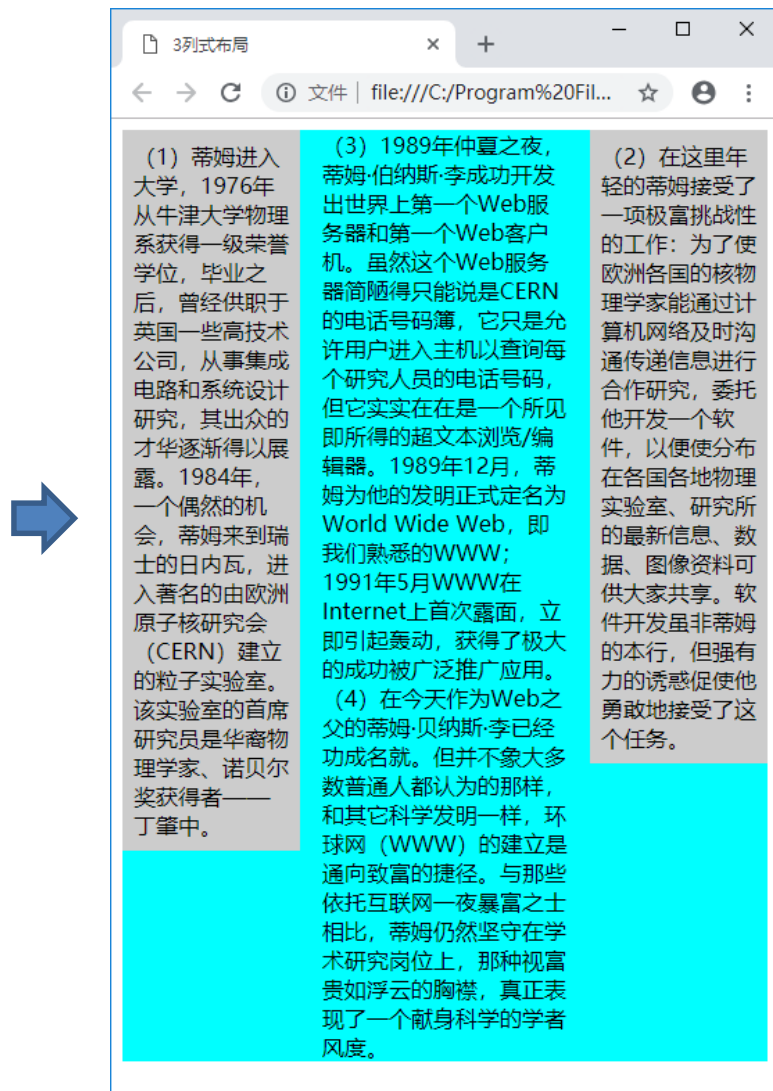
任何元素布局时一定会包含其非浮动的子元素，overflow:hidden元素(BFC元素)还会包含其浮动子元素。有关BFC元素见CSS(下)

去掉主区(3)(4)的边框和背景色，得到下图。  
如何增加主区(3)(4)与边区(1)(2)之间的空白？

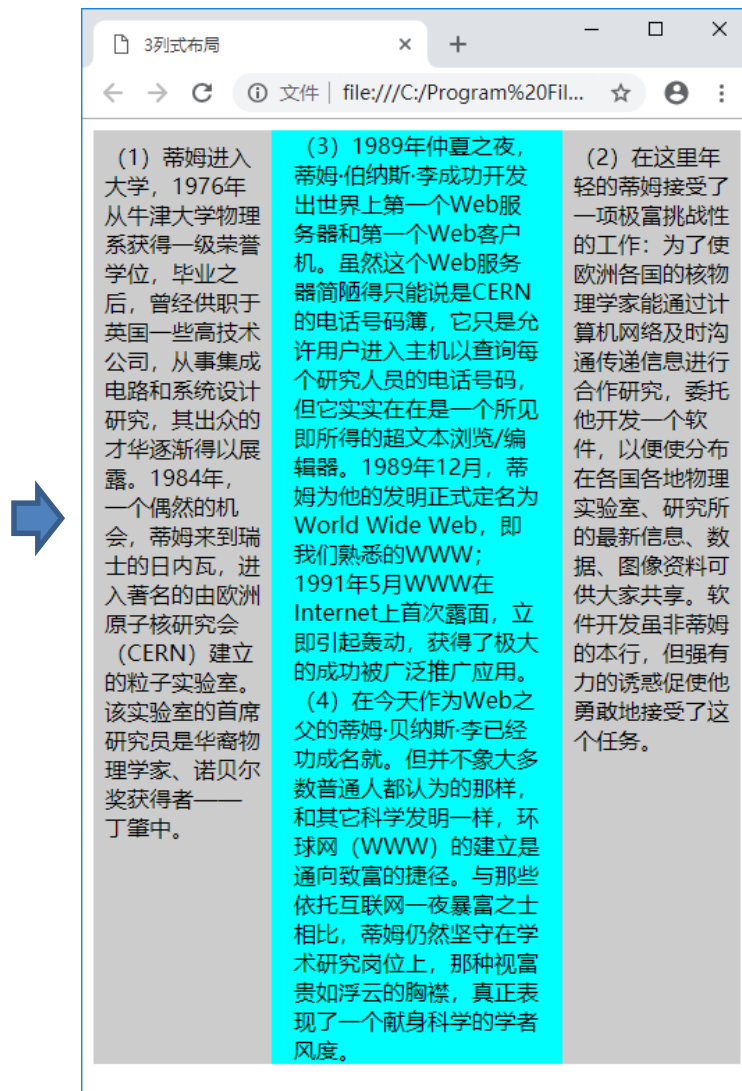
方法：(3) (4) 设置左右margin  
`margin-left: 9em;`  
`margin-right: 9em;`



## 窗口变窄后



## 如何让边区背景颜色一直到底？



- 方法：
- (1) 背景图(repeat-y):固定边区列宽，多重背景 left right。
  - (2) 采用表格模式：直接采用一个一行三列的表格。
  - (3) 通过把边区的height设置为100%
  - (4) 通过Javascript设置边区的高度
  - (5) 对边区进行设置：margin:-10000px;padding:10000px;overflow:hidden
  - \* (3)~(5)在某些浏览器上不一定有效

# 使用定位技术

- CSS定位机制有三种，分别是普通文档流、浮动和定位。如果想把多个元素重叠起来，例如，两个- position样式主要有四个取值：

absolute      相对于最近的非static的祖先元素进行定位

relative      相对于自己在正常文档流的位置进行定位

fixed      相对于浏览器窗口进行定位

static      采用普通文档流定位（默认）

position为relative的元素在正常文档流中仍然占据原有位置，而position为absolute和fixed的元素则会脱离正常文档流。

非static元素采用四个样式属性top、left、bottom、right进行设置。它们的取值均可以为负数。对于static元素，这四个样式属性不起作用。

position还有一个取值是sticky，它表现为relative和fixed的综合，平时表现为relative当超出阈值时表现为fixed，可以保证一些内容不会被滚出视口。[参考](#)

浮动元素也可以进行相对定位，就是相对于浮动后的位置移动。

基准(static)  
(原位置)



The first publicly available description of HTML was a document called "HTML Tags", first mentioned on the Internet by Tim Berners-Lee in late 1991.

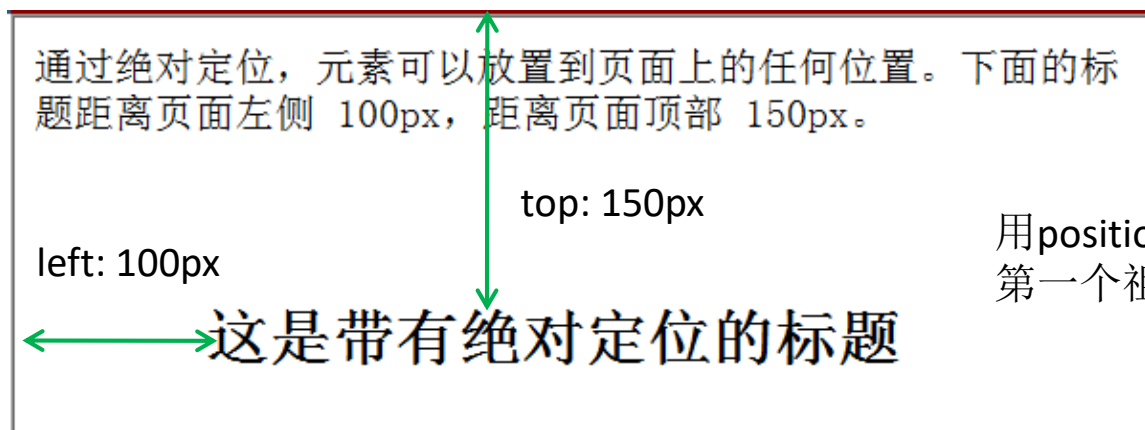
position:relative; left:-20px;

The first publicly available description of HTML was a document called "HTML Tags", first mentioned on the Internet by Tim Berners-Lee in late 1991.

position:relative; left:20px;

The first publicly available description of HTML was a document called "HTML Tags", first mentioned on the Internet by Tim Berners-Lee in late 1991.

position:absolute; left:100px; top:150px;



基准  
(祖先元素)

用position定位(非static)的  
第一个祖先元素

position:fixed; 相对浏览器窗口，不会随页面滚动。



多个对象重叠在一起时，谁在上层谁在下层呢？默认的是按照文档中出现的先后次序进行叠加，最后出现的放在在最上面。通过设置z-index的值可以改变元素的叠加次序，设置的值越大越在上层，值越小（可以为负值）越在底层。元素的z-index的默认值为0。

```
img.x
{
  position:absolute;
  left:0px;
  top:0px;
  z-index:1
}
```

## 正文

<h1>这是一个标题</h1>  
  
<p>默认的 z-index 是 0。z-index 1 拥有更高的优先级。</p>

z-index越大越显示  
在最上层

margin取负值形成  
的重叠z-index无效



# 一个标题

dex 是 0。Z-index 1 拥有  
{。

# 变换、过渡和动画

- ❑ 变换: `transform(skew,scale,rotate,translate,rotate3d), transform-origin, backface-visibility, perspective,perspective-origin`
- ❑ 过渡 `transition(property,duration,delay,timing-function)`
- ❑ 动画 `animation(name,duration,timing-function,direction, fill-mode,play-state)`

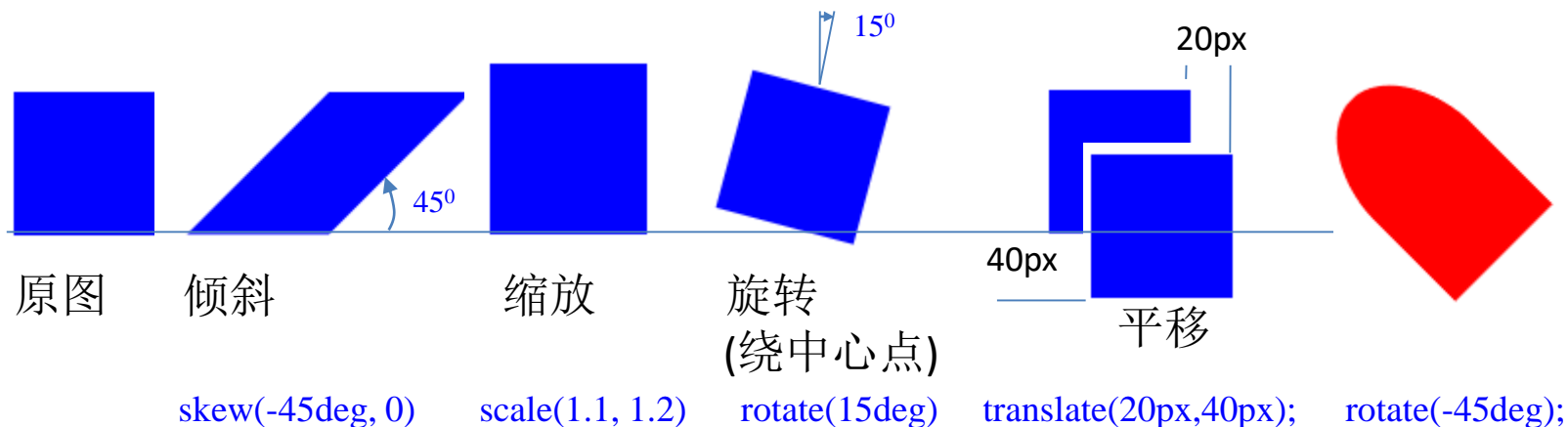
# 变换(transform)

## • 2D变换

[参考](#)

采用transform可以对块级元素进行2D的倾斜、平移、缩放、旋转变换。

<code>div{ transform: skew(-45deg, 0); }</code>	<code>/* 整个盒模型的倾斜度(水平, 垂直) */</code>
<code>div{ transform: scale(1.1, 1.2); }</code>	<code>/* 水平放大1.1倍, 垂直放大1.2倍 */</code>
<code>div{ transform: rotate(15deg); }</code>	<code>/* 顺时针方向旋转15度*/</code>
<code>div{ transform: translate(20px, 40px); }</code>	<code>/* 移位(水平, 垂直) */</code>



\* 组合: `transform: skew(0deg, -5deg) scale(1.1, 1.2)`



# • 3D变换

每种变换都有3D变换，即沿x，y和z轴的变换：

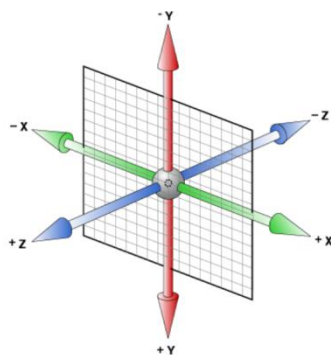
[Demos](#)

rotateX(15deg)  
rotateY(45deg)  
rotateZ(250deg)

沿任意轴

``

从通过原点和(1, -1, 1)的直线作为轴，逆时针旋转60度。



translateX, translateY, translateZ

其它组合模式：`transform: translate3d (10px, 20px, 1px) scale3d (1.5, 2, 3);`

`transform-origin: right bottom;` /\* 改变变换的原点。right bottom等同于 100% 100%\*/

`transform-style: preserve-3d;` /\* 3D变换。flat（默认）--2D变换 \*/

`backface-visibility: hidden|visible` /\* 3D背面是否可见 \*/

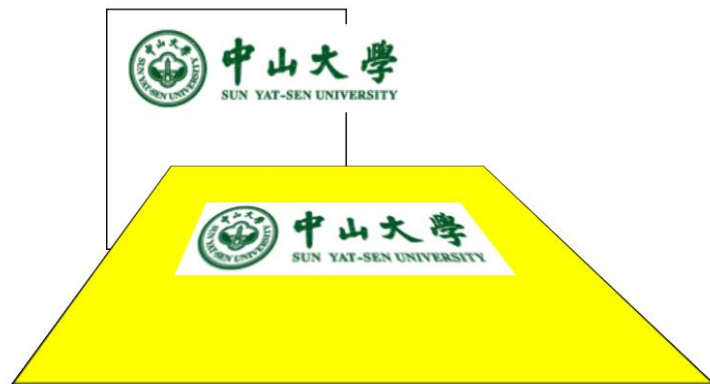
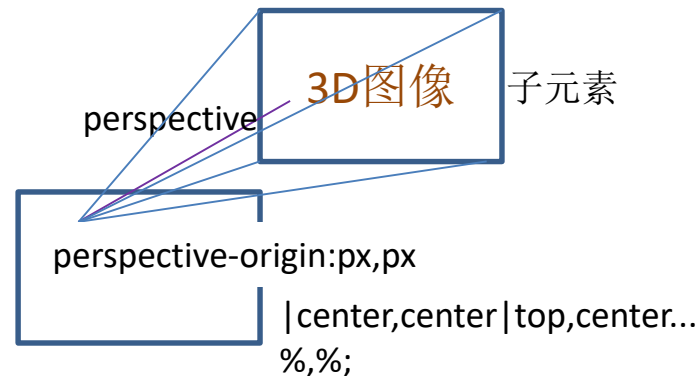
[测试](#)

# • 透视

`transform: perspective(400);` /\* 当一个元素定义了 `perspective` 属性时，其子元素会获得透视效果。400(px)为距离视图的距离，距离越近，3D效果越好。\*/

```
<!DOCTYPE html>
<html><head>
  <style>
    #div1 {
      height: 150px;
      width: 150px;
      margin: 100px;
      padding: 10px;
      border: 1px solid black;
      perspective: 150;           /* 设置观看的距离*/
      -webkit-perspective: 150;  /* Safari and Chrome */
      perspective-origin: 80% 80%; /* 设置观看的角度*/
      -webkit-perspective-origin: 80% 80%;
    }
    #img2 {
      padding: 50px;
      border: 1px solid black;
      background-color: yellow;
      transform: rotateX(45deg);
      -webkit-transform: rotateX(45deg); /* Safari and Chrome */
    }
  </style>
</head>
<body>
  <div id="div1">
    
    
  </div>
</body></html>
```

`perspective(400);`



[demo](#)



\* transform的各种演示见附录  
[测试](#)

# 过渡 (transition)

[参考](#)

[transition-property](#)

[transition-duration](#)

[transition-timing-function](#)

[transition-delay](#)

设置过渡效果的 CSS 属性的名称。 none | all | width...

完成过渡效果需要多少秒或毫秒。

速度效果的速度曲线。 取值： linear | ease | ease-in...

定义过渡效果何时开始(秒或毫秒)，默认值为0。

```
<!DOCTYPE html><html><head>
```

```
<style>
```

```
div {
```

```
width:100px;
```

```
height:100px;
```

```
background:blue;
```

```
transition-property: width;
```

```
transition-duration: 2s;
```

```
}
```

```
div:hover{
```

```
width:300px;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<div></div>
```

```
<p>请把鼠标指针移动到蓝色的 div 元素上，就可以看到过渡效果。</p>
```

```
</body></html>
```

\* **transition-timing-function**取值:

linear 匀速

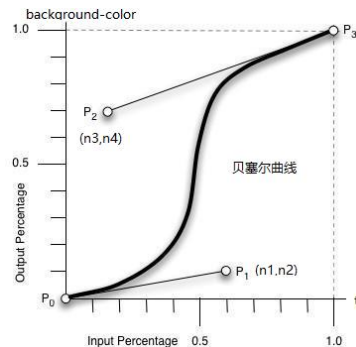
ease 从快开始逐渐变慢 (默认)

ease-in 以慢速开始逐渐变快

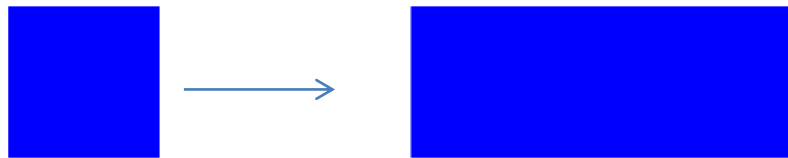
ease-out 从快开始不断增快突然减慢

ease-in-out 以慢速开始和结束

cubic-bezier(n,n,n,n) n取值 0 到 1。



\* 在鼠标悬停之后实际效果就是宽度在两秒内从100px逐渐变为300px (也可以用于背景色等)



[测试](#)

\* 多重过渡:

**transition-property:**opacity,height,width,background-color;

**transition-duration:** 3s;

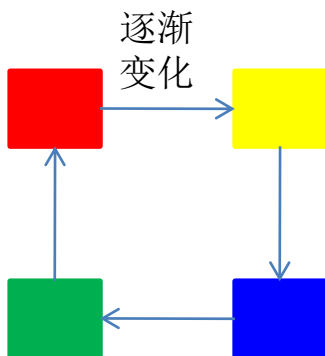
**transition-timing-function:**ease-in,ease,linear; 轮流给每个属性

\* 组合: **transition:** all 3s ease-out; all--所有属性, 可以省略  
**-webkit-transition:** all 3s ease-out;

# 动画 (Animation)

[参考](#)

```
<!DOCTYPE html>
<html>
<head>
<style>
...
</style>
</head>
<body>
<p><b>注释: </b>本例在某些浏览器中可能无效。</p>
<div></div>
</body>
</html>
```



```
div {
  margin:80px auto;
  width:100px;
  height:100px;
  background:red;
  position:relative;
  animation-name:myfirst; /* 动画名称 */
  animation-duration:5s; /* 每个循环持续时间 5s */
  animation-timing-function:linear; /* 平均分配时间, 参见上页 */
  animation-delay:2s; /* 延迟2s后开始 */
  animation-iteration-count:infinite; /* 一直循环. 或20-循环20次 */
  animation-direction:alternate; /* 每次循环改变方向 */
  animation-fill-mode: backwards; /* 延迟期间使用0%位置的帧 */
  animation-play-state:running; /* 运行(默认). 暂停-paused */
}
@keyframes myfirst { /* 五个样式变化 */
  0% {background:red; left:0px; top:0px;} /* 用from也可以 */
  25% {background:yellow; left:200px; top:0px;}
  50% {background:blue; left:200px; top:200px;}
  75% {background:green; left:0px; top:200px;}
  100% {background:red; left:0px; top:0px;} /* to */
}
```

[测试](#)

\* 组合定义 **animation:** *myfirst 5s linear 2s infinite alternate;*

\* 相同帧可以合并: 0%, 100% {background:red; left:0px; top:0px;}

\* 播放方向: **animation-direction:** *back|forth|alternate;*

\* 静止时使用样式: **animation-fill-mode:** *none(默认)|backwards|forwards|both*

\* 多重动画: **animation:** *anim1 10s , anim2 20s, anim3 30s;*

*/\* 每次都向后(0%到100%)|向前|向前向后交替 \*/*  
*/\* 原有样式|0%帧|100%帧|0%和100%叠加 \*/*  
*/\* 也可以animation-name中写多个名字, 下面属性用逗号隔开不同的值\*/*

# 附录1、140种命名颜色

颜色	名称	中文名称	RGB 值	颜色	名称	中文名称	RGB 值	颜色	名称	中文名称	RGB 值
	LightPink	浅粉红	#FFB6C1		LightCyan	淡青色	#E0FFFF		FloralWhite	花的白色	#FFFAF0
	Pink	粉红	#FFC0CB		PaleTurquoise	弱绿宝石	#AFEEEE		OldLace	旧蕾丝	#FDF5E6
	Crimson	猩红 (深红)	#DC143C		Cyan	青色	#00FFFF		Wheat	小麦色	#F5DEB3
	LavenderBlush	淡紫红	#FFF0F5		Aqua	水色	#00FFFF		Moccasin	鹿皮鞋	#FFE4B5
	PaleVioletRed	弱紫罗兰红	#DB7093		DarkTurquoise	暗绿宝石	#00CED1		Orange	橙色	#FFA500
	HotPink	热情的粉红	#FF69B4		DarkSlateGray	暗石板灰	#2F4F4F		PapayaWhip	番木瓜	#FFEFD5
	DeepPink	深粉红	#FF1493		DarkCyan	暗青色	#008B8B		BlanchedAlmond	发白的杏仁色	#FFEBCD
	MediumVioletRed	中紫罗兰红	#C71585		Teal	水鸭色	#008080		NavajoWhite	土著白	#FFDEAD
	Orchid	兰花紫	#DA70D6		MediumTurquoise	中绿宝石	#48D1CC		AntiqueWhite	古董白	#FAEBD7
	Thistle	薊	#D8BFD8		LightSeaGreen	浅海洋绿	#20B2AA		Tan	茶色	#D2B48C
	Plum	李子紫	#DDA0DD		Turquoise	绿宝石	#40E0D0		BurlyWood	硬木色	#DEB887
	Violet	紫罗兰	#EE82EE		Aquamarine	宝石碧绿	#7FFFD4		Bisque	陶坯黄	#FFE4C4
	Magenta	洋红 (品红 玫瑰红)	#FF00FF		MediumAquamarine	中宝石碧绿	#86CDAA		DarkOrange	深橙色	#FF8C00
	Fuchsia	灯笼海棠 (紫红色)	#FF00FF		MediumSpringGreen	中春绿色	#00FA9A		Linen	亚麻布	#FAFAD2
	DarkMagenta	深洋红	#8B008B		MintCream	薄荷奶油	#F5FFFA		Peru	秘鲁	#CD853F
	Purple	紫色	#800080		SpringGreen	春绿色	#00FF00		PeachPuff	桃肉色	#FFDAB9
	MediumOrchid	中兰花紫	#BA55D3		MediumSeaGreen	中海洋绿	#3CB371		SandyBrown	沙棕色	#F4A460
	DarkViolet	暗紫罗兰	#9400D3		SeaGreen	海洋绿	#2E8B57		Chocolate	巧克力	#D2691E
	DarkOrchid	暗兰花紫	#9932CC		Honeydew	蜜瓜色	#F0FF00		SaddleBrown	马鞍棕色	#8B4513
	Indigo	靛青 (紫兰色)	#4B0082		LightGreen	淡绿色	#90EE90		Seashell	海贝壳	#FFF5EE
	BlueViolet	蓝紫罗兰	#8A2BE2		PaleGreen	弱绿色	#98FB98		Sienna	黄土赭色	#A0522D
	MediumPurple	中紫色	#9370DB		DarkSeaGreen	暗海洋绿	#8FBC8F		LightSalmon	浅鲑鱼肉色	#FFA07A
	MediumSlateBlue	中板岩蓝	#7B68EE		LimeGreen	闪光深绿	#32CD32		Coral	珊瑚	#FF7F50
	SlateBlue	板岩蓝	#6A5ACD		Lime	闪光绿	#00FF00		OrangeRed	橙红色	#FF4500
	DarkSlateBlue	暗板岩蓝	#483D8B		ForestGreen	森林绿	#228B22		DarkSalmon	深鲜肉 (鲑鱼)色	#E9967A
	Lavender	薰衣草淡紫	#E6E6FA		Green	纯绿	#008000		Tomato	番茄红	#FF6347
	GhostWhite	幽灵白	#F8F8FF		DarkGreen	暗绿色	#006400		MistyRose	薄雾玫瑰	#FFE4E1
	Blue	纯蓝	#0000FF		Chartreuse	查特酒绿 (黄绿色)	#7FFF00		Salmon	鲜肉 (鲑鱼)色	#FA8072
	MediumBlue	中蓝色	#0000CD		LawnGreen	草坪绿	#7CFC00		Snow	雪	#FFFAFA
	MidnightBlue	午夜蓝	#191970		GreenYellow	绿黄色	#ADFF2F		LightCoral	淡珊瑚色	#F08080
	DarkBlue	暗蓝色	#00008B		DarkOliveGreen	暗橄榄绿	#556B2F		RosyBrown	玫瑰棕色	#BC8F8F
	Navy	海军蓝	#000080		YellowGreen	黄绿色	#9ACD32		IndianRed	印度红	#CD5C5C
	RoyalBlue	皇家蓝 (宝蓝)	#4169E1		OliveDrab	橄榄褐色	#6B8E23		Red	纯红	#FF0000
	CornflowerBlue	矢车菊蓝	#6495ED		Beige	米色 (灰棕色)	#F5F5DC		Brown	棕色	#A52A2A
	LightSteelBlue	亮钢蓝	#B0C4DE		LightGoldenrodYellow	亮菊黄	#FADFAD		FireBrick	耐火砖	#B22222
	LightSlateGray	亮石板灰	#778899		Ivory	象牙	#FFFFF0		DarkRed	深红色	#8B0000
	SlateGray	石板灰	#708090		LightYellow	浅黄色	#FFFFE0		Maroon	栗色	#800000
	DodgerBlue	道奇蓝	#1E90FF		Yellow	纯黄	#FFFF00		White	纯白	#FFFFFF
	AliceBlue	爱丽丝蓝	#F0F8FF		Olive	橄榄	#808000		WhiteSmoke	白烟	#F5F5F5
	SteelBlue	钢蓝 (铁青)	#4682B4		DarkKhaki	深卡叽布	#BDB76B		Gainsboro	康斯博罗灰色	#DCDCDC
	LightSkyBlue	亮天蓝色	#87CEFA		LemonChiffon	柠檬绸	#FFFACD		LightGray	浅灰色	#D3D3D3
	SkyBlue	天蓝色	#87CEEB		PaleGoldenrod	灰菊黄	#EEE8AA		Silver	银灰色	#C0C0C0
	DeepSkyBlue	深天蓝	#00BFFF		Khaki	卡叽布	#F0E68C		DarkGray	深灰色	#A9A9A9
	LightBlue	亮蓝	#ADD8E6		Gold	金色	#FFD700		Gray	灰色	#808080
	PowderBlue	火药青	#B0E0E6		Corn silk	玉米丝色	#FFF8DC		DimGray	暗淡的灰色	#696969
	CadetBlue	军服蓝	#5F9EAO		Goldenrod	金菊黄	#DAA520		Black	纯黑	#000000
	Azure	蔚蓝色	#F0FFFF		DarkGoldenrod	暗金菊黄	#8B860B				

# 附录2、Web安全色(216种)

#000000	#000033	#000066	#000099	#0000CC	#0000FF
#003300	#003333	#003366	#003399	#0033CC	#0033FF
#006600	#006633	#006666	#006699	#0066CC	#0066FF
#009900	#009933	#009966	#009999	#0099CC	#0099FF
#00CC00	#00CC33	#00CC66	#00CC99	#00CCCC	#00CCFF
#00FF00	#00FF33	#00FF66	#00FF99	#00FFCC	#00FFFF
#330000	#330033	#330066	#330099	#3300CC	#3300FF
#333300	#333333	#333366	#333399	#3333CC	#3333FF
#336600	#336633	#336666	#336699	#3366CC	#3366FF
#339900	#339933	#339966	#339999	#3399CC	#3399FF
#33CC00	#33CC33	#33CC66	#33CC99	#33CCCC	#33CCFF
#33FF00	#33FF33	#33FF66	#33FF99	#33FFCC	#33FFFF
#660000	#660033	#660066	#660099	#6600CC	#6600FF
#663300	#663333	#663366	#663399	#6633CC	#6633FF
#666600	#666633	#666666	#666699	#6666CC	#6666FF
#669900	#669933	#669966	#669999	#6699CC	#6699FF
#66CC00	#66CC33	#66CC66	#66CC99	#66CCCC	#66CCFF
#66FF00	#66FF33	#66FF66	#66FF99	#66FFCC	#66FFFF

#990000	#990033	#990066	#990099	#9900CC	#9900FF
#993300	#993333	#993366	#993399	#9933CC	#9933FF
#996600	#996633	#996666	#996699	#9966CC	#9966FF
#999900	#999933	#999966	#999999	#9999CC	#9999FF
#99CC00	#99CC33	#99CC66	#99CC99	#99CCCC	#99CCFF
#99FF00	#99FF33	#99FF66	#99FF99	#99FFCC	#99FFFF
#CC0000	#CC0033	#CC0066	#CC0099	#CC00CC	#CC00FF
#CC3300	#CC3333	#CC3366	#CC3399	#CC33CC	#CC33FF
#CC6600	#CC6633	#CC6666	#CC6699	#CC66CC	#CC66FF
#CC9900	#CC9933	#CC9966	#CC9999	#CC99CC	#CC99FF
#CCCC00	#CCCC33	#CCCC66	#CCCC99	#CCCCCC	#CCCCFF
#CCFF00	#CCFF33	#CCFF66	#CCFF99	#CCFFCC	#CCFFFF
#FF0000	#FF0033	#FF0066	#FF0099	#FF00CC	#FF00FF
#FF3300	#FF3333	#FF3366	#FF3399	#FF33CC	#FF33FF
#FF6600	#FF6633	#FF6666	#FF6699	#FF66CC	#FF66FF
#FF9900	#FF9933	#FF9966	#FF9999	#FF99CC	#FF99FF
#FFCC00	#FFCC33	#FFCC66	#FFCC99	#FFCCCC	#FFCCFF
#FFFF00	#FFFF33	#FFFF66	#FFFF99	#FFFFCC	#FFFFFF

# 附录3、CSS常用属性

[参考1](#)  
[参考2](#)

## • CSS 字体属性（Font）

属性	描述	CSS
<a href="#">font-family</a>	规定文本的字体系列。依次选择，直到支持的一个。font-family:"Times New Roman",Georgia,Serif;	1
<a href="#">font-size</a>	规定文本的字体尺寸。1.2em,120%,12px (原来为10px)。关键字取值：xx-small, x-small, small, medium, large, x-large, xx-large（与默认尺寸比较）	1
<a href="#">font-style</a>	规定文本的字体样式。normal,italic,oblique (倾斜)	1
<a href="#">font-variant</a>	规定文本的字体样式。normal,small-caps (小型大写)	1
<a href="#">font-weight</a>	规定字体的粗细。normal, bold, bolder, lighter, 100, 200, 300,400(normal),500,600,700(bold),800,900	1
<a href="#">font</a>	在一个声明中设置所有字体属性。font:italic bold 12px/20px arial,sans-serif; *12px-font-size, 20px-line height	1

@font-face {font-family: "self font"; src:url('fonts/self.ttf'); 用于自动下载字体。

## • CSS 列表属性（List）

属性	描述	CSS
<a href="#">list-style-image</a>	将图象设置为列表项标记。list-style-image: url("/i/arrow.gif");	1
<a href="#">list-style-position</a>	设置列表项标记的放置位置。inside,outside	1
<a href="#">list-style-type</a>	设置列表项标记的类型。disc   circle   square   decimal   decimal-leading-zero   lower-roman   upper-roman   lower-greek   lower-latin   upper-latin   armenian   georgian   none   inherit	1
<a href="#">list-style</a>	在一个声明中设置所有的列表属性。 list-style:square inside url('/i/arrow.gif');	1



## • CSS 文本属性 (Text)

属性	描述	CSS
<a href="#">color</a>	设置文本的颜色。color:red ;#0000FF; rgb(0,0,255); rgb(50%,0,50%);	1
<a href="#">letter-spacing</a>	设置字符间距。	1
<a href="#">line-height</a>	设置行高。line-height:1.2em; 120%;12px; (原来为10px)	1
<a href="#">text-align</a>	规定文本的水平对齐方式。left,right,center,justify(两端对齐)	1
<a href="#">text-decoration</a>	规定添加到文本的装饰效果。none,underline,overline(上划线),line-through(删除线),blink(闪烁)	1
<a href="#">text-indent</a>	规定文本块首行的缩进。text-indent:2em;	1
<a href="#">text-transform</a>	控制文本的大小写。none, uppercase,lowercase, capitalize(首字母大写)	1
<a href="#">white-space</a>	规定如何处理元素中的空白符。normal 合并空白符忽略换行符, pre 保留空白符, nowrap不换行, pre-wrap 保留空白和正常换行符(自动换行), pre-line 合并空白符序列但是保留换行符。	1
<a href="#">word-spacing</a>	设置单词间距。	
text-shadow	规定添加到文本的阴影效果。text-shadow:-4px 4px 3px #999999; 阴影水平偏移量, 垂直偏移量, 模糊度, 颜色	

## • CSS 表格属性 (Table)

属性	描述
<a href="#">border-collapse</a>	设置是否把表格边框合并为单一的边框。取值:collapse, separate。
<a href="#">border-spacing</a>	设置分隔单元格边框的距离。
<a href="#">caption-side</a>	设置表格标题的位置。取值: top, bottom
<a href="#">empty-cells</a>	设置是否显示表格中的空单元格的边框。取值: show, hide
<a href="#">table-layout</a>	设置显示单元、行和列的算法。automatic(默认。列宽度由单元格内容设定), fixed(列宽由表格宽度和列宽度设定)



## • CSS 边框属性（Border 和 Outline）

属性	描述	CSS
<a href="#">border-color</a>	设置四条边框的颜色。up,right,down,left	1
<a href="#">border-style</a>	设置四条边框的样式。取值：dotted, solid, double, dashed, ... 更多的值见附录。	1
<a href="#">border-width</a>	设置四条边框的宽度。thin medium thick 数值	1
<a href="#">border-left</a>	在一个声明中设置所有的左边框属性。right, top, bottom与此类似	1
<a href="#">border-left-color</a>	设置左边框的颜色。	2
<a href="#">border-left-style</a>	设置左边框的样式。	2
<a href="#">border-left-width</a>	设置左边框的宽度。	1
<a href="#">border</a>	在一个声明中设置所有的边框属性。border:5px solid red;	1
border-radius	设置圆角。border-radius: 0 30px 10px 5px; 左上角 右上角 右下角 左下角 （半径）border-top-left-radius, border-top-right-radius,...	3
box-shadow	设置阴影。border-shadow: -4px 6px 8px #000000 (水平偏移, 垂直偏移, 半径, 颜色)	3

\* 前三个属性都可以取1~4个值: 1个值表示所有项取值相同, 2个值的down,left取up和right的值, 有3个值时left取right的值。

## • 背景（background）

属性	描述	CSS
<a href="#">background</a>	在一个声明中设置所有的背景属性。	1
<a href="#">background-attachment</a>	设置背景图像是否固定或者随着页面的其余部分滚动。	1
<a href="#">background-color</a>	设置元素的背景颜色。	1
<a href="#">background-image</a>	设置元素的背景图像。	1
<a href="#">background-position</a>	设置背景图像的开始位置。	1
<a href="#">background-repeat</a>	设置是否及如何重复背景图像。	1
<a href="#">background-clip</a>	规定背景的绘制区域。	3
<a href="#">background-origin</a>	规定背景图片的定位区域。	3
<a href="#">background-size</a>	规定背景图片的尺寸。	3

## • CSS 边距属性（Margin&Padding）

属性	描述	CSS
<a href="#">margin-bottom</a>	设置元素的下外边距。	1
<a href="#">margin-left</a>	设置元素的左外边距。	1
<a href="#">margin-right</a>	设置元素的右外边距。	1
<a href="#">margin-top</a>	设置元素的上外边距。	1
<a href="#">margin</a>	在一个声明中设置所有外边距属性。 p { margin:2cm 4cm 3cm 4cm; }      up, right, down, left （TRouBLE）	
<a href="#">padding-bottom</a>	设置元素的下内边距。	1
<a href="#">padding-left</a>	设置元素的左内边距。	1
<a href="#">padding-right</a>	设置元素的右内边距。	1
<a href="#">padding-top</a>	设置元素的上内边距。	1
<a href="#">padding</a>	在一个声明中设置所有内边距属性。 p { padding:2cm 4cm 3cm 4cm; } 可取1, 2, 3个值      up, right, down, left （TRouBLE）	

## • CSS 尺寸属性（Dimension）

属性	描述	CSS
<a href="#">height</a>	设置元素高度。50%，12px	1
<a href="#">max-height</a>	设置元素的最大高度。	2
<a href="#">max-width</a>	设置元素的最大宽度。	2
<a href="#">min-height</a>	设置元素的最小高度。	2
<a href="#">min-width</a>	设置元素的最小宽度。	2
<a href="#">width</a>	设置元素的宽度。	1
box-sizing	取值:content-box,padding-box,border-box	3

- CSS 定位属性（Positioning）

属性	描述	CSS
<a href="#">position</a>	规定元素的定位类型，取值：static, fixed, relative, absolute。	2
<a href="#">float</a>	规定框是否应该浮动。none, left, right	1
<a href="#">clear</a>	规定元素的哪一侧不允许其他浮动元素。none(都允许), left, right, both。	2
<a href="#">display</a>	规定元素应该生成的框的类型。none不显示，block按块级元素显示(换行)，inline按内联元素显示(不换行)，inline-block内联块(块元素但是不换行)	1
<a href="#">visibility</a>	规定元素是否可见。visible, hidden, collapse (IE8不支持) collapse可删除表格行或列，但它不会影响表格布局。用在其他的元素上，会呈现为“hidden”。	2

<a href="#">vertical-align</a>	设置元素的垂直对齐方式。取值：baseline, sub, super, top, text-top, text-bottom, middle, bottom。具体解释见后面	1
<a href="#">z-index</a>	设置元素的堆叠顺序。默认值为0，可以为负数。值越大越顶层，越小越底层。	2
<a href="#">Left</a> , <a href="#">top</a> <a href="#">right</a> , <a href="#">bottom</a>	<b>left:</b> 设置定位元素左外边距边界与其包含块左边界之间的偏移。 <b>top:</b> 上外边距边界与其包含块上边界 <b>right:</b> 右外边距边界与其包含块右边界 <b>bottom:</b> 下外边距边界与其包含块下边界	2
<a href="#">overflow</a> <a href="#">overflow-x</a> <a href="#">overflow-y</a>	规定当内容溢出元素框时发生的事情。visible, hidden, scroll, auto(有隐藏时自动显示滚动条)。 <a href="#">overflow-x</a> 和 <a href="#">overflow-y</a> 水平和垂直方向溢出处理	2
<a href="#">cursor</a>	规定要显示的光标的类型(形状)。span.help {cursor:help;} 其它形状见后面。	2
<a href="#">clip</a>	剪裁绝对定位元素。img { position:absolute; clip:rect(0px,60px,200px,0px); }	

- 动画属性

## CSS3 动画属性 ( Animation )

属性	描述	CSS
<a href="#"><u>@keyframes</u></a>	规定动画。	3
<a href="#"><u>animation</u></a>	所有动画属性的简写属性，除了 animation-play-state 属性。	3
<a href="#"><u>animation-name</u></a>	规定 @keyframes 动画的名称。	3
<a href="#"><u>animation-duration</u></a>	规定动画完成一个周期所花费的秒或毫秒。	3
<a href="#"><u>animation-timing-function</u></a>	规定动画的速度曲线。	3
<a href="#"><u>animation-delay</u></a>	规定动画何时开始。	3
<a href="#"><u>animation-iteration-count</u></a>	规定动画被播放的次数。	3
<a href="#"><u>animation-direction</u></a>	规定动画是否在下一周期逆向地播放。	3
<a href="#"><u>animation-play-state</u></a>	规定动画是否正在运行或暂停。	3
<a href="#"><u>animation-fill-mode</u></a>	规定对象动画时间之外的状态。	3

# 附录4、属性取值

- vertical-align 属性值

值	描述
<b>baseline</b>	默认。元素放置在父元素的基线上。
<b>sub</b>	垂直对齐文本的下标。
<b>super</b>	垂直对齐文本的上标
<b>top</b>	把元素的顶端与行中最高元素的顶端对齐
<b>text-top</b>	把元素的顶端与父元素字体的顶端对齐
<b>middle</b>	把此元素放置在父元素的中部。
<b>bottom</b>	把元素的顶端与行中最低的元素的顶端对齐。
<b>text-bottom</b>	把元素的底端与父元素字体的底端对齐。
<b>length</b>	
<b>%</b>	使用 "line-height" 属性的百分比值来排列此元素。允许使用负值。
<b>inherit</b>	规定应该从父元素继承 vertical-align 属性的值。

- background-position取值

值	描述
top left top center top right	其它取值: center left, center center, center right, bottom left, bottom center, bottom right。如果您仅规定了一个关键词, 那么第二个值将是"center"。默认值: 0% 0%。
x% y%	第一个值是水平位置, 第二个值是垂直位置。左上角是 0% 0%。右下角是 100% 100%。如果您仅规定了一个值, 另一个值将是 50%。
xpos ypos	第一个值是水平位置, 第二个值是垂直位置。左上角是 0 0。单位是像素 (0px 0px) 或任何其他 CSS 单位。如果您仅规定了一个值, 另一个值将是50%。您可以混合使用 % 和 position 值。

## •边框(线)样式值

值	描述
none	定义无边框。
hidden	与 "none" 相同。不过应用于表时除外，对于表，hidden 用于解决边框冲突。
dotted	定义点状边框。在大多数浏览器中呈现为实线。
dashed	定义虚线。在大多数浏览器中呈现为实线。
solid	定义实线。
double	定义双线。双线的宽度等于 border-width 的值。
groove	定义 3D 凹槽边框。其效果取决于 border-color 的值。
ridge	定义 3D 垄状边框。其效果取决于 border-color 的值。
inset	定义 3D inset 边框。其效果取决于 border-color 的值。
outset	定义 3D outset 边框。其效果取决于 border-color 的值。
inherit	规定应该从父元素继承边框样式。

## • [position](#) 属性取值

position 属性规定元素的定位类型。

值	描述
absolute	从正常文档流移出，相对于 position 设定为非static的第一个父元素进行定位。元素的位置通过 "left", "top", "right" 以及 "bottom" 属性进行规定。
fixed	相对于浏览器窗口进行定位，不会随页面滚动。元素的位置通过 "left", "top", "right" 以及 "bottom" 属性进行规定。
relative	相对于其正常位置进行定位。 因此, "left:20" 会向元素的 LEFT 位置添加 20 像素。
static	默认值。没有定位，元素出现在正常的流中（忽略 top, bottom, left, right 或者 z-index 声明）。
inherit	规定应该从父元素继承 position 属性的值。

- 光标([cursor](#))的属性值

值	描述
url	需使用的自定义光标的 URL。注释：请在此列表的末端始终定义一种普通的光标，以防没有由 URL 定义的可用光标。
default	默认光标（通常是一个箭头）
auto	默认。浏览器设置的光标。
crosshair	光标呈现为十字线。
pointer	光标呈现为指示链接的指针（一只手）
move	此光标指示某对象可被移动。
e-resize	此光标指示矩形框的边缘可被向右（东）移动。
ne-resize	此光标指示矩形框的边缘可被向上及向右移动（北/东）。
nw-resize	此光标指示矩形框的边缘可被向上及向左移动（北/西）。
n-resize	此光标指示矩形框的边缘可被向上（北）移动。
se-resize	此光标指示矩形框的边缘可被向下及向右移动（南/东）。
sw-resize	此光标指示矩形框的边缘可被向下及向左移动（南/西）。
s-resize	此光标指示矩形框的边缘可被向下移动（南/西）。
w-resize	此光标指示矩形框的边缘可被向左移动（西）。
text	此光标指示文本。
wait	此光标指示程序正忙（通常是一只表或沙漏）。
help	此光标指示可用的帮助（通常是一个问号或一个气球）。

cursor:help;

## • display 属性值

值	描述
none	此元素不会被显示。
block	此元素将显示为块级元素，此元素前后会带有换行符。
inline	默认。此元素会被显示为内联元素，元素前后没有换行符。
inline-block	内联块元素。（CSS2.1 新增的值）
list-item	此元素会作为列表显示。
run-in	此元素会根据上下文作为块级元素或内联元素显示。
compact	CSS 中有值 compact，不过由于缺乏广泛支持，已经从 CSS2.1 中删除。
marker	CSS 中有值 marker，不过由于缺乏广泛支持，已经从 CSS2.1 中删除。
table	此元素会作为块级表格来显示（类似 <table>），表格前后带有换行符。
inline-table	此元素会作为内联表格来显示（类似 <table>），表格前后没有换行符。
table-row-group	此元素会作为一个或多个行的分组来显示（类似 <tbody>）。
table-header-group	此元素会作为一个或多个行的分组来显示（类似 <thead>）。
table-footer-group	此元素会作为一个或多个行的分组来显示（类似 <tfoot>）。
table-row	此元素会作为一个表格行显示（类似 <tr>）。
table-column-group	此元素会作为一个或多个列的分组来显示（类似 <colgroup>）。
table-column	此元素会作为一个单元格列显示（类似 <col>）
table-cell	此元素会作为一个表格单元格显示（类似 <td> 和 <th>）
table-caption	此元素会作为一个表格标题显示（类似 <caption>）
inherit	规定应该从父元素继承 display 属性的值。



# 附录5、transform取值

**transform:** none | *transform-functions*;

值	描述	测试
none	定义不进行转换。	<a href="#">测试</a>
matrix( <i>n,n,n,n,n,n</i> )	定义 2D 转换，使用六个值的矩阵。	<a href="#">测试</a>
matrix3d( <i>n,n,n,n,n,n,n,n,n,n,n,n,n,n,n,n</i> )	定义 3D 转换，使用 16 个值的 4x4 矩阵。	
translate( <i>x,y</i> )	定义 2D 转换。	<a href="#">测试</a>
translate3d( <i>x,y,z</i> )	定义 3D 转换。	
translateX( <i>x</i> )	定义转换，只是用 X 轴的值。	<a href="#">测试</a>
translateY( <i>y</i> )	定义转换，只是用 Y 轴的值。	<a href="#">测试</a>
translateZ( <i>z</i> )	定义 3D 转换，只是用 Z 轴的值。	
scale( <i>x,y</i> )	定义 2D 缩放转换。	<a href="#">测试</a>
scale3d( <i>x,y,z</i> )	定义 3D 缩放转换。	
scaleX( <i>x</i> )	通过设置 X 轴的值来定义缩放转换。	<a href="#">测试</a>
scaleY( <i>y</i> )	通过设置 Y 轴的值来定义缩放转换。	<a href="#">测试</a>
scaleZ( <i>z</i> )	通过设置 Z 轴的值来定义 3D 缩放转换。	
rotate( <i>angle</i> )	定义 2D 旋转，在参数中规定角度。	<a href="#">测试</a>
rotate3d( <i>x,y,z,angle</i> )	定义 3D 旋转。	
rotateX( <i>angle</i> )	定义沿着 X 轴的 3D 旋转。	<a href="#">测试</a>
rotateY( <i>angle</i> )	定义沿着 Y 轴的 3D 旋转。	<a href="#">测试</a>
rotateZ( <i>angle</i> )	定义沿着 Z 轴的 3D 旋转。	<a href="#">测试</a>
skew( <i>x-angle,y-angle</i> )	定义沿着 X 和 Y 轴的 2D 倾斜转换。	<a href="#">测试</a>
skewX( <i>angle</i> )	定义沿着 X 轴的 2D 倾斜转换。	<a href="#">测试</a>
skewY( <i>angle</i> )	定义沿着 Y 轴的 2D 倾斜转换。	<a href="#">测试</a>
perspective( <i>n</i> )	为 3D 转换元素定义透视视图。	<a href="#">测试</a>

[http://www.w3school.com.cn/cssref/pr\\_transform.asp](http://www.w3school.com.cn/cssref/pr_transform.asp)

属性	描述	CSS
<a href="#"><u>transform</u></a>	向元素应用 2D 或 3D 转换。	3
<a href="#"><u>transform-origin</u></a>	允许你改变被转换元素的位置。	3
<a href="#"><u>transform-style</u></a>	规定被嵌套元素如何在 3D 空间中显示。	3
<a href="#"><u>perspective</u></a>	规定 3D 元素的透视效果。	3
<a href="#"><u>perspective-origin</u></a>	规定 3D 元素的底部位置。	3
<a href="#"><u>backface-visibility</u></a>	定义元素在不面对屏幕时是否可见。	3

# 附录6、width和height总结

- width取值：px, percentage, auto（默认）
- 对于块级元素（也称为静态块级元素），width限制了内容的宽度。width设置为auto时，内容很多时，会尽量占满父元素的宽度并自动换行（wrap）。width:100%是填满整个父元素的宽度。
- 对于内联元素，width不起作用，元素宽度为包裹元素内容的宽度。
- 对于浮动元素、表格元素、内联元素、内联块级元素和绝对定位的元素(left和right为auto)，width设置为auto(默认)是为包裹内容的宽度。
- 对于绝对定位的块级元素，如果width设置为auto，left和right不为auto，则该元素会被伸缩，与width:100%类似。
- height与width类似，但是对于静态块级元素，height:auto会包裹元素内容。