# CSS(层叠样式表)

## CSS位置

### 引用内部样式

<head>

<meta charset=”utf-8”/>

<style type=”text/css”>

CSS代码

</style>

</head>

### 引用外部样式

样式引用先base.css，然后对应相应页面的css

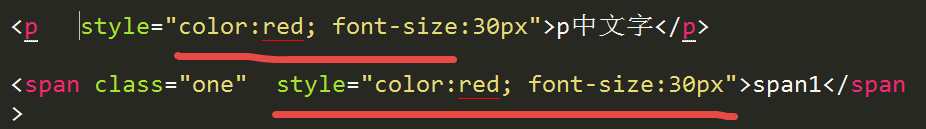
<head>

<meta charset=”utf-8”/>

<link rel=”stylesheet” href=”base.css”/>

</head>

### 行内样式



## CSS选择器

### ☞基础选择器

#### 标签选择器

标签名{属性:值; 属性:值;} p{ }

#### 类选择器

.类名{属性:值;} .className{ }

类可以由多个标签使用

一个标签可以使用多个类，类名之间用空格分开

#### id选择器

#id名{属性:值;} #idName{ }

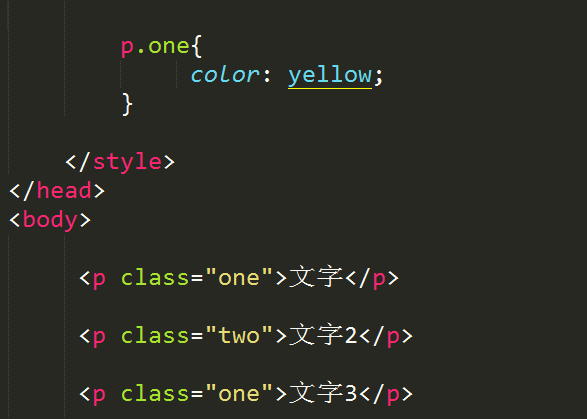
id只能单个标签使用

#### 通配符选择器

\*{属性:值;}

### ☞复合选择器

#### 标签指定式选择器





#### 后代选择器

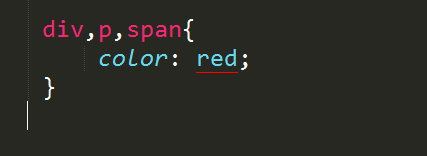
特点：

标签之间必须属于嵌套关系。

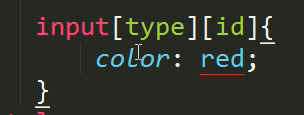
用法：

选择器 +空格+ 选择器{属性:值; 属性:值;...}

#### 并集选择器



#### 属性选择器



#### 子代选择器

不同于后代选择器，子代选择器只包括第一代子类



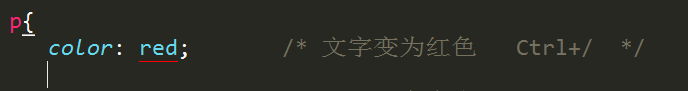
## CSS相关属性

### 公共属性

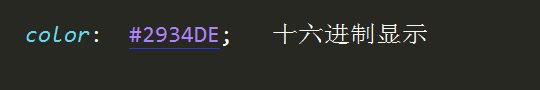
|  |  |  |
| --- | --- | --- |
| 属性 | 赋值 | 介绍 |
| **color** | color: red; | 前景色（文字颜色） |
| **font-size** | font-size: 20px; | 设置文字大小 |
| **width** | width: 30px; | 设置宽度 |
| **Height** | height:30px; | 设置高度 |
| **Background-color** | background-color:red; | 设置背景颜色 |
| **Text-align** | text-align:center; left|right; | 文字居中(在块级元素也可以文本居中) |
| **Text-indent** | text-indent: 2em； | 首行缩进 |
| **Margin** | Margin: 0 auto; | 盒子居中显示 |

### color

颜色的显示方式：

☞ 直接设置对应颜色的名称

☞ 通过十六进制显示

以#开头 0-f 

☞通过rgb方式显示

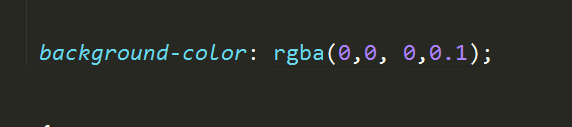
R: 代表的是红色（red） 0-255

G: 代表的是绿色（green） 0-255

B: 代表的是蓝色（blue） 0-255

☞rgba

a: 设置透明度 alpha 0-1



### font

Font-size: 设置文字大小

Font-family: 设置文字字体

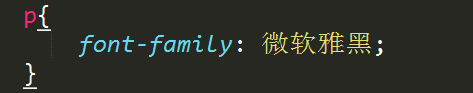
Font-weight: 设置文字加粗

Font-style: 设置文字斜体

Line-height: 行高

#### Font-family

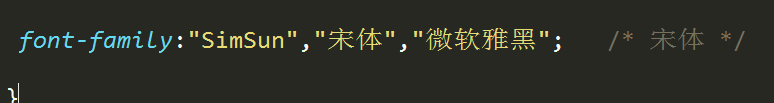
☞ 直接设置汉字



☞ 设置对应字体的英文



☞可以设置unicode编码

Font-family可以设置多个值；

#### Font-weight

取值方式：

☞ 直接设置数字 100-900

☞ bold (字体加粗) 效果与 700-900

☞ normal(文字字体正常显示)

注意：文字加粗显示，推荐使用 font-weight:700;

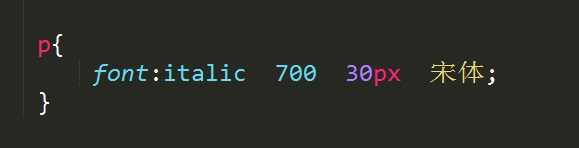
#### Font-style(斜体)

☞ italic 文字斜体显示

☞ normal 文字正常显示

#### Font属性联写

font: font-style font-weight font-size/line-height font-family



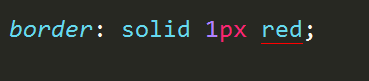
注意：

☞ font属性联写必须有 font-size 和font-family(其他属性可以不写)

☞ font-size 和font-family的顺序不能换

### border

#### 属性联写



（属性联写的时候没有先后顺序限制，边框颜色可以不写，边框宽度可以不写）

### 背景

background-color：背景颜色

background-image：背景图片

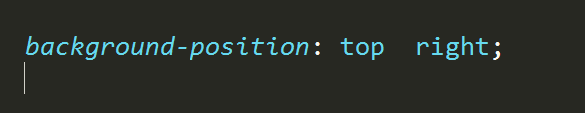
（注意：设置背景图片的时候一定要设置宽度和高度）

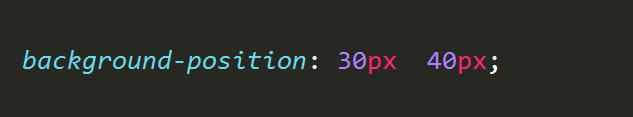
background-repeat：背景平铺

（repeat|no-repeat|repeat-x|repeat-y）

background-position：背景位置

（left| right| top| bottom| center）

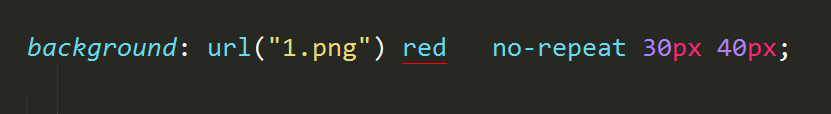




background-attachment：背景是否固定

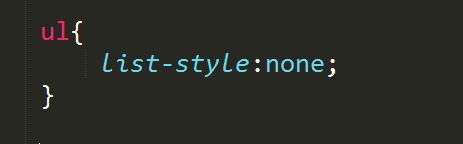
（scroll|fixed）

属性联写：无先后顺序



### 列表

去掉列表前的显示方式：



### 鼠标样式

Cursor: pointer 鼠标变成小手

Cursor: default; 小白

Cursor : move; 移动

Cursor : text ; 文本输入

### Border-radius

圆角矩形：Border-radius: 左上 右上 右下 左下； 顺时针

border-radius: 7px 7px 7px 0;



## CSS属性简写

background：背景颜色 背景图片 是否重复

* background:gray url(img/XXX) no-repeat;
* background:url(img/XXX) repeat-y;
* background:gray;

border：边框粗细 边框类型 边框颜色

* border:1px solid #ccc;
* border:1px solid;(只有颜色可以省略，颜色默认黑色)

font:斜体 加粗 字号/行高 字体(包含备用字体)

* font:italic bold 20px/35px arial,"微软雅黑"；

(可以省略一些,但是要注意顺序问题)

margin/padding: 上 右 下 左

* 上 左右 下
* 上下 左右
* 上下左右

color

* 名称
* rgb(255,255,255);
* #FFFFFF;
* rgba(255,255,255,0.5) (a:透明度)

## HTML标签分类

### 块级元素

可以嵌套任意元素

（p 不能放 div）

典型代表：

div,p,li,h1...

特点：

☞元素自己独占一行显示（与宽度无关）

☞可以设置宽度和高度

☞当嵌套一个块级元素，子元素如果不设置宽度，那么该子元素的宽度为父元素的宽度。

### 行级元素

行内元素 只 嵌套元素 b u span i s

（a 无所不能： a里面不能放 a 、input 等）

典型代表:

Span， a, font, strong...

特点：

☞元素在一行上显示

☞不能直接设置宽度和高度

### 行内块元素

典型代表:

Image, input(表单控件)

特点：

☞ 元素在一行上显示

☞ 可以设置宽度和高度

行内块之间默认存在间隙，可以使用浮动解除间隙

### 元素类型的转换

Display: inline 将元素转化为行内元素

Display：inline-block 将元素转化行内块元素

Display: block 将元素转化为块元素



## CSS特性

### 层叠性

样式的层叠性与样式的调用顺序没有关系，与样式的定义顺序有关

### 继承性

继承性发生的前提是标签之间属于一种嵌套关系

☞文字颜色可以实现继承

☞文字大小可以实现继承

☞字体可以实现继承

☞行高可以实现继承

与文字有关的属性都可以 实现继承

特殊情况：

<a href=“#”></a> 不能继承父元素中的文字颜色

<h1></h1> 标题标签不能继承父元素中的文字大小

### 优先级

默认样式<标签选择器<类选择器<id选择器<行内样式<!important

0 1 10 100 1000 1000以上

（继承的权重为0，即用class选择器定义子元素样式不会被其父元素的id选择器效果抵消）

（权重会叠加，即使用复合选择器会将每个选择器的权重叠加）

## CSS伪类/伪元素（Pseudo-classes/elements）

伪类：定义元素不同状态的样式

伪元素：定义元素内部文本的样式

### 以a标签为例的伪类：

a标签由 4 种状态：

* 正常状态（默认状态）

a{ } 如果链接没有指向的地址，就显示a的颜色

a:link{ } 如果链接定义了指向的地址，就显示a:link状态

* 鼠标放在链接上的状态

a:hover{ }

* 鼠标点击后状态

a:visited{ }

* 获取焦点(表单状态)

a:focus{ }

* 鼠标正在点击的状态（激活状态）

a:active{ }

a的属性： text-decoration=“None|underline|line-through“

### 其他伪类

:first-child 伪类--来选择父元素的第一个子元素

:lang 伪类--使你有能力为不同的语言定义特殊的规则





### 伪元素



## 浮动

### Float: left| right

特点：

☞ 设置了浮动的元素不占原来的位置（脱标：脱离标准流）

☞ 可以让块级元素在一行上显示

☞ 浮动可以行内元素转化为行内块元素

（模式转换的过程中，能用display就用display，因为不会造成脱标）

作用：

☞ 浮动用来解决文字图片环绕问题

☞ 制作导航栏

☞ 网页布局

### 清除浮动

☞ 清除浮动不是删除浮动

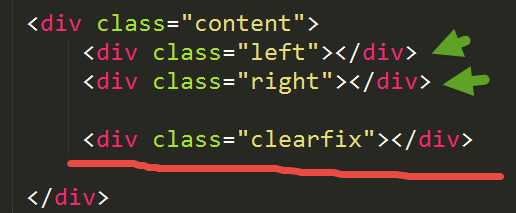
☞ 清除浮动指的是清除浮动的影响

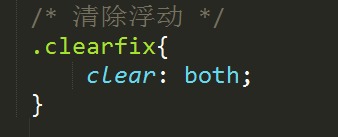
注意：当子元素设置了浮动，父元素没有高度的时候，造成页面布局混乱。这种情况下进行清除浮动。

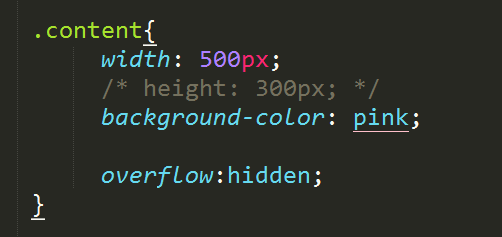
#### 清除浮动的方式

1、clear: left | right | both

clear: both 在要清除浮动的元素后面添加一个标签

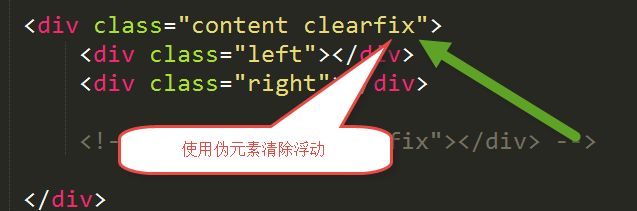




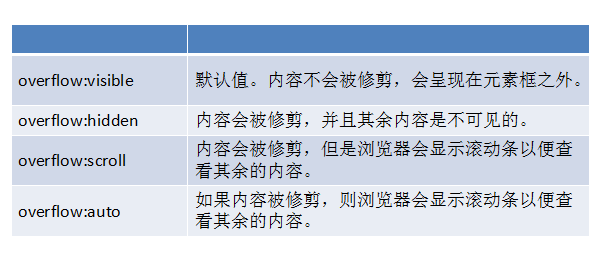
2、给父盒子设置overflow:hidden 

如果父盒子中有定位的元素，一般不推荐使用该种方式清除浮动。

3、使用伪元素清除浮动

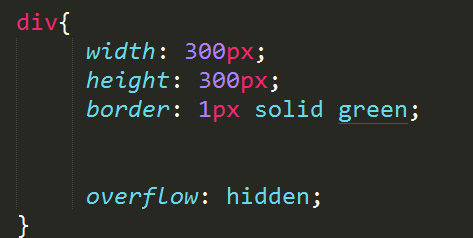


### overflow



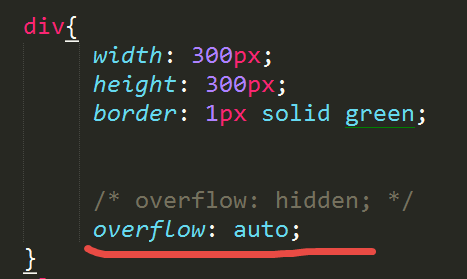
☞ hidden

将超出部分进行隐藏

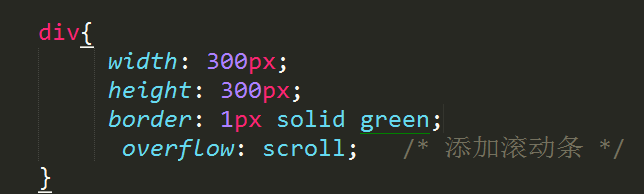


☞auto

如果内容超出盒子，那么给盒子设置滚动条，如果内容没有超出盒子，那么不显示滚动条。



☞scorll

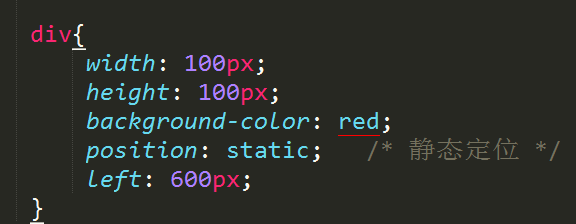


## 定位

方位名称：Left， right ， top, bottom

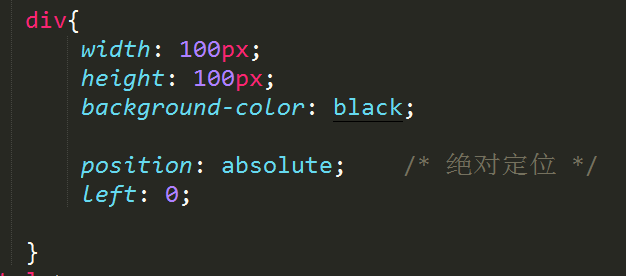
### 静态定位（static）

Position： static



（静态定位就是元素标准流的显示方式，即默认显示效果。）

### 绝对定位（absolute）（看脸型）

Position：absolute;

特点：

☞当给一个单独的元素设置绝对定位，以浏览器左上角（body）为基准设置定位的。

☞当盒子发生嵌套关系的时候，如果父盒子没有设置定位，子盒子设置定位以浏览器左上角为基准设置定位。

☞当盒子发生嵌套关系的时候，如果父盒子设置定位，子盒子设置定位以父盒子左上角为基准设置定位。

☞给盒子设置了绝对定位，该盒子不占位置（脱标）

☞给行内元素设置绝对定位后，该元素转化为了行内块元素

（注意：元素设置了绝对定位后，通过具体的方位名称可以随便设置元素的位置。）

### 相对定位（relative）（自恋型）

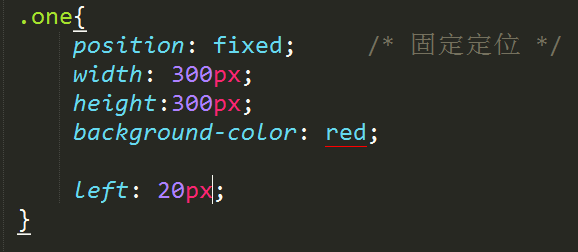
☞ 元素设置了相对定位后占原来的位置

☞ 设置相对定位以自己的位置为参照设置位置

☞ 相对定位不能进行元素的模式转换

☞ 子绝父相（子元素设置绝对定位，父元素设置相对定位）

### 固定定位（fixed）（只看浏览器）



☞固定定位不占位置（脱标）

☞将行内元素转化为行内块元素

### z-index

只有 定位的盒子 （除了static） 才有 z-index

如果都是定位-绝对定位 他们默认的z-index 是 0

最后的一个：靠上

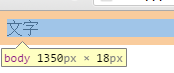
## 盒子模型

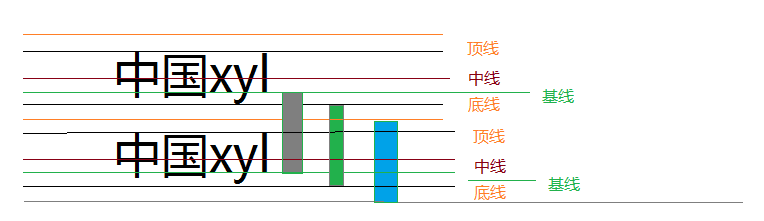
### 浏览器默认文字

默认大小 16px

默认行高 18px（inline-height）

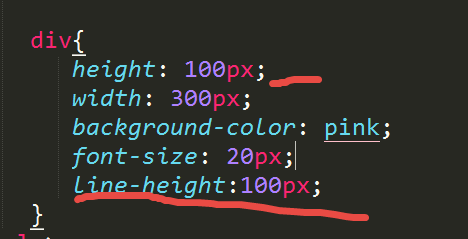
行高=文字大小+上间距+下间距





### 行高作用

当行高值为父容器的高度时，可以让父容器中的文字垂直显示



#### 行高单位的影响

1、单独给一个元素设置行高

|  |  |  |  |
| --- | --- | --- | --- |
| 行高单位 | 赋值 | 文字大小 | 行高值 |
| px | 20px | 20px | 20px |
| em | 2em | 20px | 40px |
| % | 120% | 20px | 24px |
| 不带单位 | 2 | 20px | 40px |

总结：

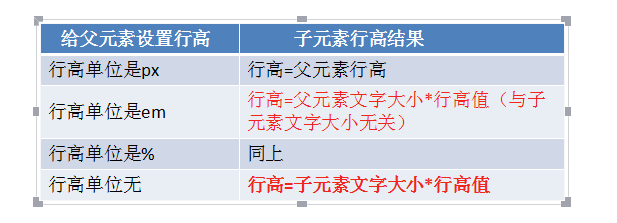
当给一个独立的元素设置行高值的时候，除了以px为单位的行高值与文字大小无关，其他都与文字大小有关（与文字大小的积）

2、给父元素设置行高值，子元素的行高问题

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 行高单位 | 设置行高 | 父文字 | 子文字 | 行高 |
| Px | 20px | 20px | 30px | 20px |
| em | 2em | 20px | 30px | 40px |
| % | 120% | 20px | 30px | 24px |
| 不带单位 | 2 | 20px | 30px | 60px |

3、行高可以实现继承

当父元素设置的行高值除不带单位情况下，都是先与父元素的文字大小相乘最后的结果，被子元素继承。



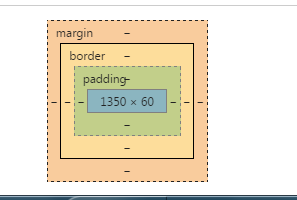
### 盒子模型

#### 盒子组成

☞border(边框)

☞内边距（padding）

☞外边距（margin）



#### 边框（border）

☞Border-width: 边框宽度

☞border-style: 边框样式

◆solid 边框为实线

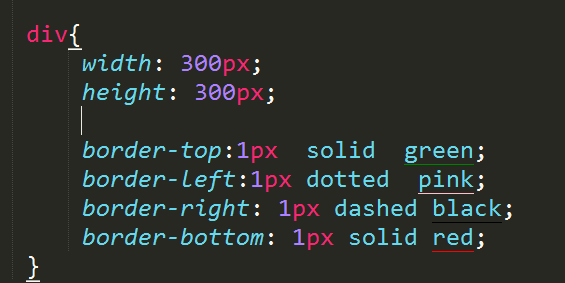
◆dotted 点线

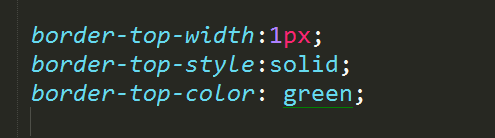
◆dashed 虚线

☞border-color: 边框颜色

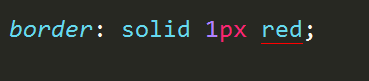


单独设置border各部分



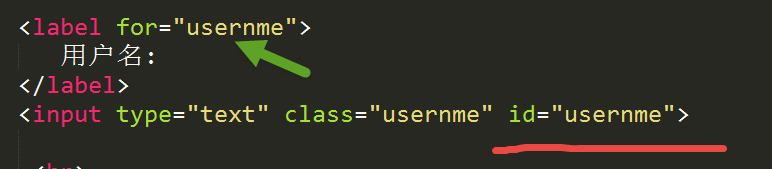


属性联写

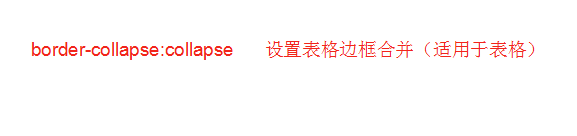


（属性联写的时候没有先后顺序限制，边框颜色可以不写，边框宽度可以不写）

表单优化



表格单元格合并



#### 内边距（padding）

设置内容距离盒子边框之间的距离

1、构成

Padding-left: 左边距

Padding-right: 右边距

Padding-top: 上边距

Padding-bottom: 下边距

2、属性联写

Padding: 10px; 上，右，下，左的距离为10px

Padding: 10px 20px; 上下10px 左右20px

Padding: 10px 20px 30px 上10px 左右20px 下30px

Padding: 10px 20px 30px 40px; 上， 右 ， 下， 左

#### 盒子大小计算

☞ 边框可以影响盒子大小

☞ 内边距影响盒子大小

宽度=内容宽度+左右边框+左右内边距

注意：

以后进行页面盒子布局的实现，如果给盒子设置了内边距，对应的要将内容宽度或者高度减去相应的值。

继承的盒子在父盒子宽度范围内，padding值不会影响该盒子大小。

#### 外边距（margin）

1、构成

Margin-left：

Margin-right：

Margin-top：

Margin-bottom：

2、属性联写

Margin: 10px 上 右 下 左 10px

Margin:10px 20px 上下10px 左右20px

Margin: 10px 20px 30px 上10px 左右20px 下30px

Margin: 10px 20px 30px 40px 上右下左

3、

当两个盒子垂直显示的时候，外边距以设置的最大值为准（外边距合并的第一种情况）

☞外边距塌陷

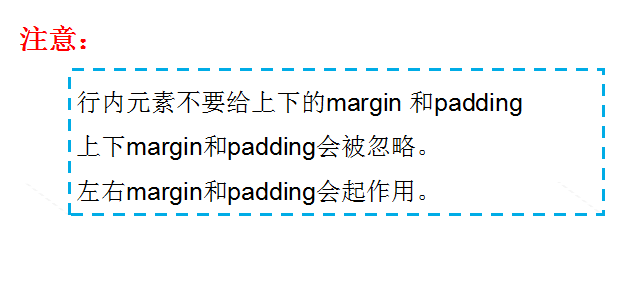
◆给父盒子设置边框

◆给父盒子设置overflow:hidden;

☞给父元素设置了overflow：hiddenh会触发bfc

4、Bfc：格式化上下文

http://www.w3cplus.com/css/understanding-bfc-and-margin-collapse.html



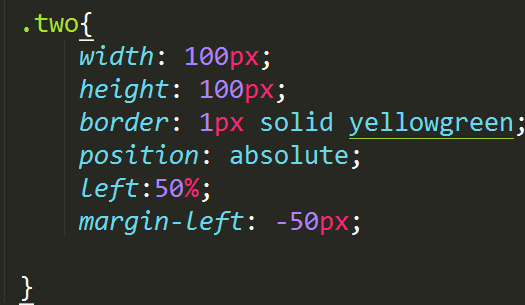
### 盒子居中的方法

标准流下的盒子居中显示 Margin:0 auto;

定位的盒子居中显示

◆先走父盒子宽度的一半： left:50%;

◆往回走自己宽度的一半



## CSS初始化

@charset "utf-8";

/\*CSS初始化\*/

/\*part-1:默认设置\*/

html,body,

ul,li,ol,dl,dd,dt,

p,h1,h2,h3,h4,h5,h6,

form,fieldset,legend,img {

margin: 0;

padding: 0;

}

h1,h2,h3,h4,h5,h6 {

text-decoration: none;

font-weight: normal;

font-size: 100%;

}

s,i,em {

font-style: normal;

text-decoration: none;

}

ul,ol {

list-style: none;/\*列表前面没有点\*/

}

fieldset,img,input,button {

border: none;

padding: 0;

margin: 0;

outline-style: none;

}

select,input {

vertical-align: middle;

}

textarea {

resize: none; /\*防止拖动\*/

}

img {/\* 去掉图片低测默认的3像素空白缝隙\*/

border: 0;

vertical-align: middle;

}

table {

border-collapse: collapse;

}

/\*part-2:自定义样式\*/

/\*默认字体\*/

body,div,p,span {

font: bold 16px 微软雅黑, "Helvetica Neue", Helvetica, Arial, sans-serif;

}

input {

padding-top: 0;

padding-bottom: 0;

font-family: "SimSun", "宋体";

}

select,input,textarea {

font-size: 12px;

margin: 0;

}

body {

font: 12px/150% Arial, Verdana, "\5b8b\4f53";

/\*color: #666;\*/

/\*background: #fff;\*/

}

/\*默认链接样式\*/

a {

text-decoration: none;

/\*color: #666;\*/

}

a:hover {

/\*color: #ad0711;\*/

}

/\*part-3:自定义类样式\*/

/\*清除浮动方法1：伪元素\*/

.clearfix:after {

content:””;

Visibility:hidden;

Display:block;

Height:0;

Clear:both;

}

.clearfix{

Zoom:1;

}

/\*清除浮动方法2：双伪元素\*/

/\*.clearfix:before,

.clearfix:after {

content: "";

display: table;

}

.clearfix:after {

clear: both;

}

.clearfix {

\*zoom: 1; /\*兼容IE/7/6\*/

}\*/

/\*背景颜色\*/

.bgc-red {

background-color: #ad0711;

}

/\*颜色\*/

.col-red {

color: #C81623!important;

}

/\*公共类\*/

.w {

/\*版心 提取 \*/

width: 1210px;

margin: 0 auto;

}

.fl {

float: left

}

.fr {

float: right

}

.al {

text-align: left

}

.ac {

text-align: center

}

.ar {

text-align: right

}

.hide {

display: none

}

## 补充知识

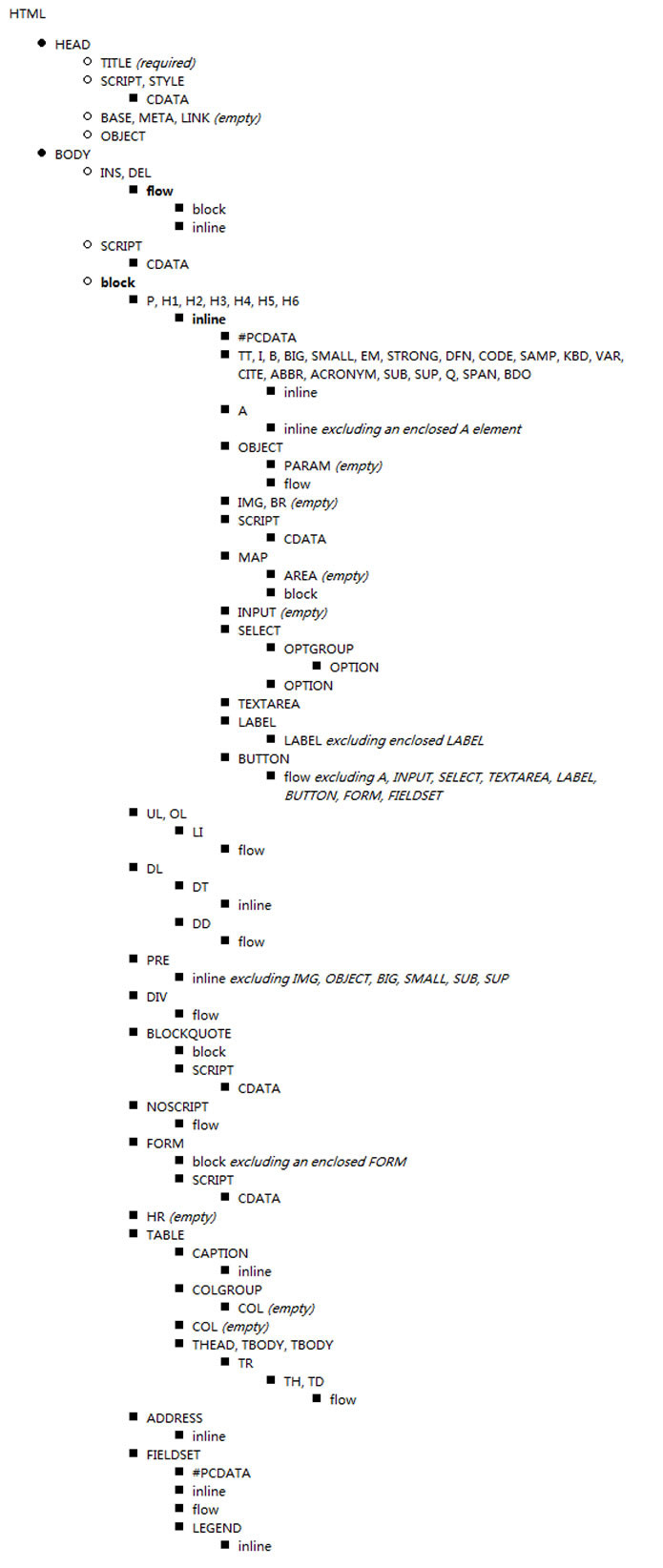
### 标签包含规范

☞ div可以包含任何标准流下的元素

☞ p标签中不能包含div 和 标题标签 及列表标签。

☞ 标题标签可以包含其他标签。

☞ 行内元素最好不要包含其他标签。



### 规避脱标流

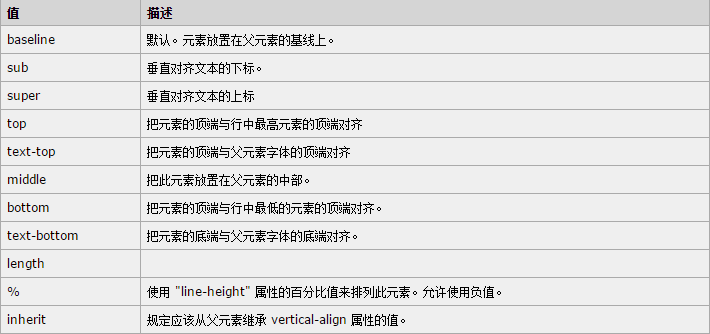
☞网页布局过程中，能用表流布局就用标准流布局。

☞标准流不能解决用浮动

☞浮动不能解决用定位

■使用margin-left| margin-right 取值为auto可以将盒子自动冲到另一边。



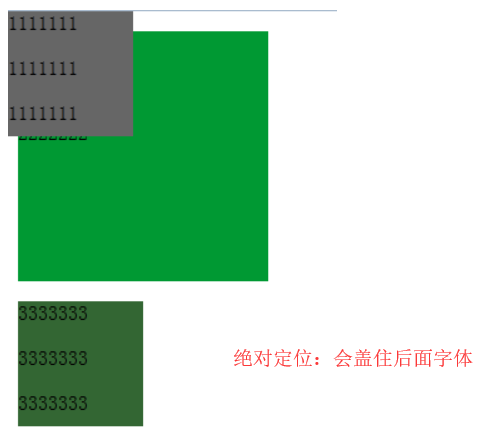


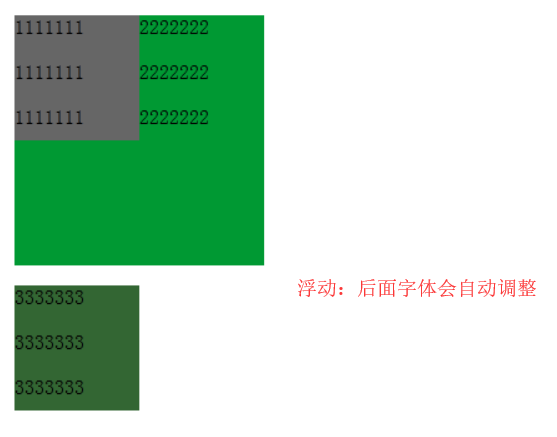
### 浮动和绝对定位的区别

float：**浮动的框可以向左或向右移动，直到它的外边缘碰到包含框或另一个浮动框的边框为止。由于浮动框不在文档的普通流中，所以文档的普通流中的块框表现得就像浮动框不存在一样。**

absolute：**设置为绝对定位的元素框从文档流完全删除，并相对于其包含块定位，包含块可能是文档中的另一个元素或者是初始包含块。元素原先在正常文档流中所占的空间会关闭，就好像该元素原来不存在一样。元素定位后生成一个块级框，而不论原来它在正常流中生成何种类型的框。**

对于浮动float来说，它仅仅是浮动框脱离标准流，并不是去安全的脱离标准流，而绝对定位absolute是将元素框完全的脱离标准流从文档流中删除：





### CSS可见性

☞overflow: hidden 将超出部分进行隐藏

☞display: none 直接将元素隐藏

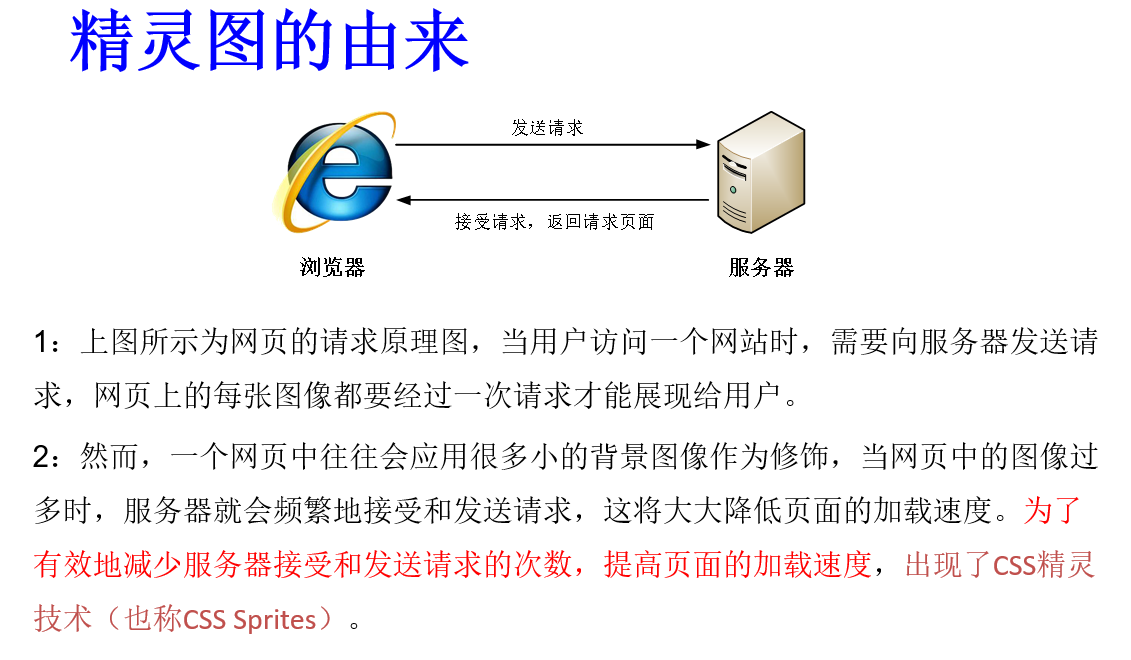
display：block 将元素显示(与js配合更搭)

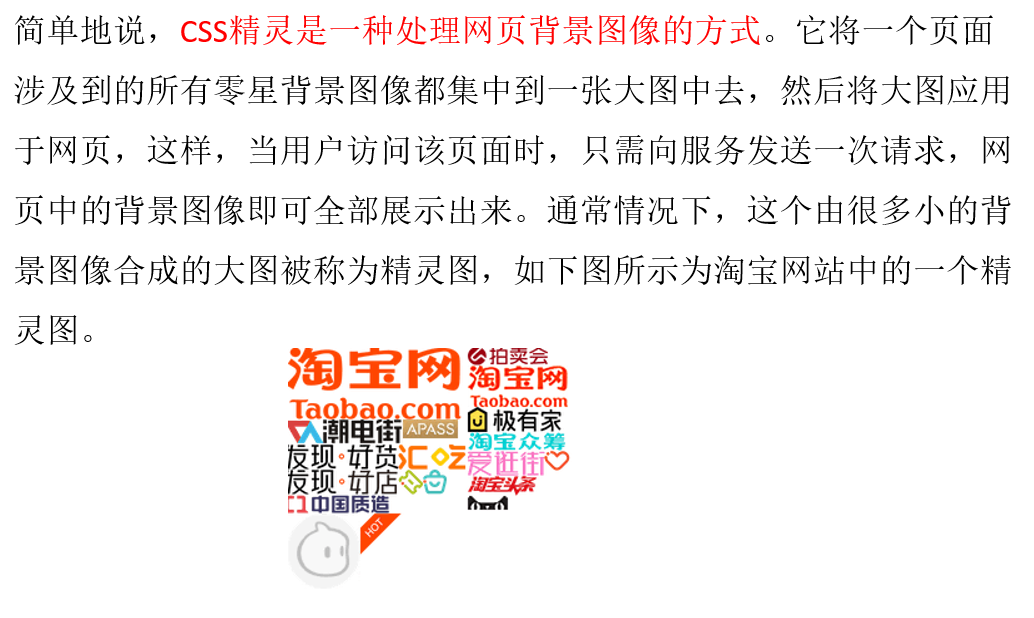
☞visibility:hidden 将元素隐藏

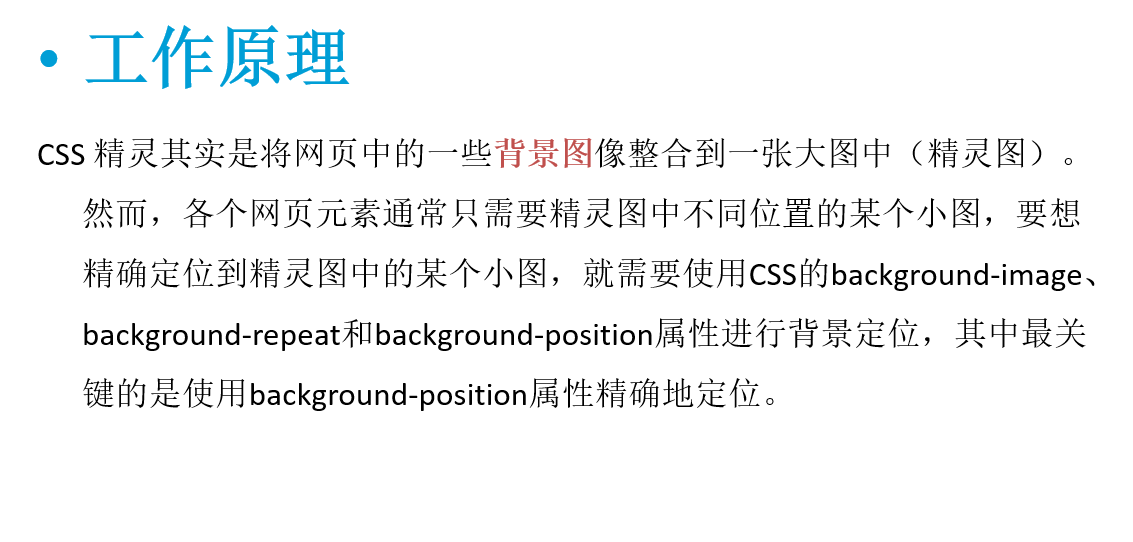
display: none; 将元素隐藏后不占位置

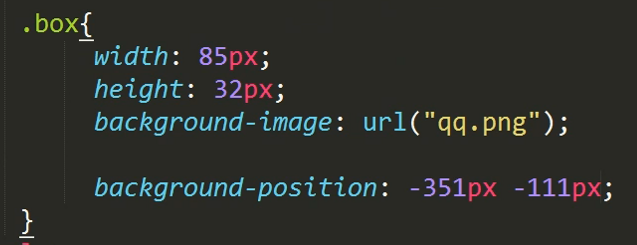
visibility: hidden; 将元素隐藏后占原来的位置

### 精灵图

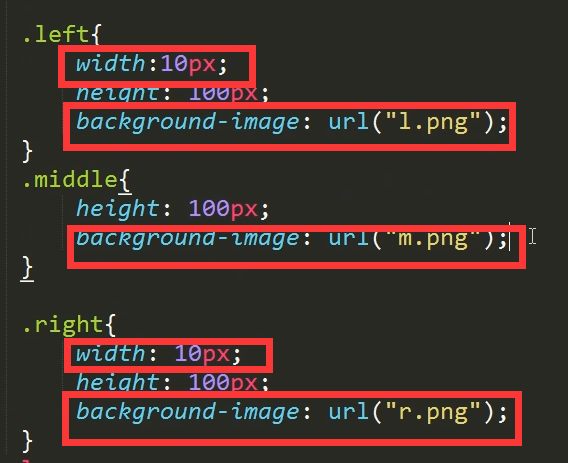


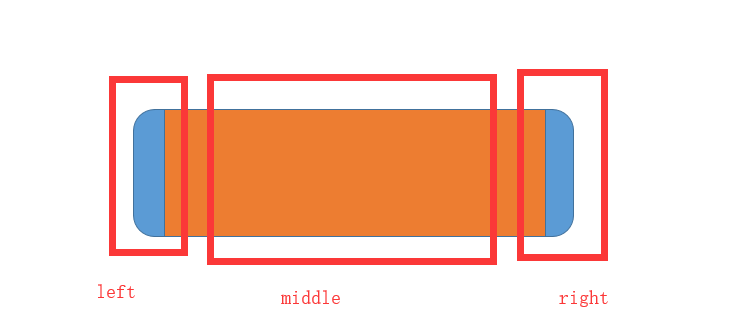






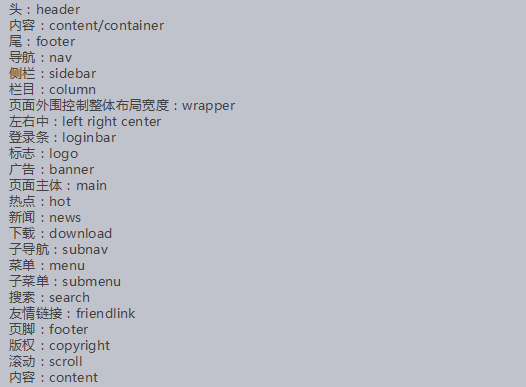
### 滑动门





中间部分不设置宽度，由文字撑开

### 常见英文命名



### 媒体类型

@media 规则

@media 规则允许在相同样式表为不同媒体设置不同的样式。

在下面的例子告诉我们浏览器屏幕上显示一个 14 像素的 Verdana 字体样式。但是如果页面打印，将是 10 个像素的 Times 字体。请注意，font-weight 在屏幕上和纸上设置为粗体：

@media screen

{

p.test {font-family:verdana,sans-serif;font-size:14px;}

}

@media print

{

p.test {font-family:times,serif;font-size:10px;}

}

@media screen,print

{

p.test {font-weight:bold;}

}



### 计数器

使用计数器自动编号

CSS 计数器根据规则来递增变量。

CSS 计数器使用到以下几个属性：

* counter-reset - 创建或者重置计数器
* counter-increment - 递增变量
* content - 插入生成的内容
* counter() 或 counters() 函数 - 将计数器的值添加到元素

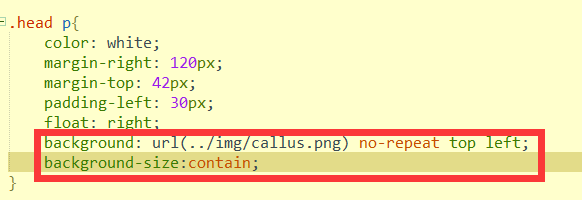
## 布局常见问题

### 外边距塌陷

父盒子未设置高度，子盒子使用margin-top会造成外边距塌陷

解决办法：父盒子设置overflow：hidden；

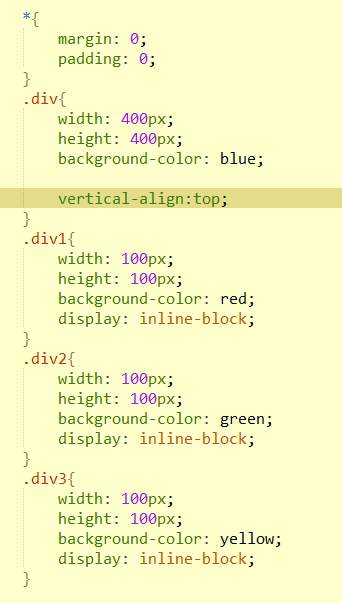
### 将logo作为背景图显示

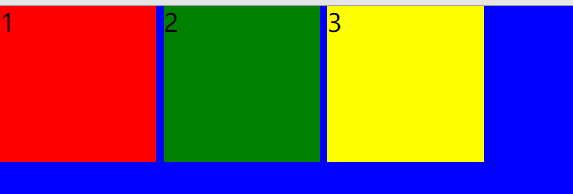


### ul无序列表默认样式更改



### 使用display: inline-block;会导致边距



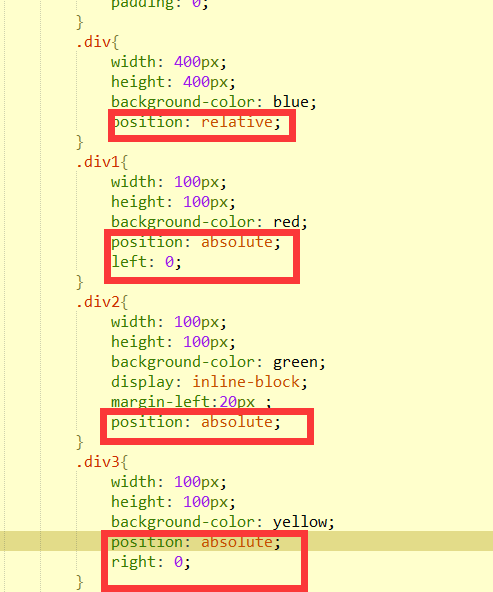


解决办法：margin-left设置为负值

margin-left: -10px;

产生距离的原因？？？

### 解决悬浮问题



### 图片水平+垂直居中

父盒子{

text-align：center；

}

img{

vertical-align：middle；

}

### 清除浮动

在设置浮动的div中加上 .clearfix

.clearfix:after{

content:””;

line-height:0;

height:0;

display:block;

clear:both;

visibility:hidden;

}

.clear{//为了兼容ie浏览器

zoom：1；

}

### 背景半透明

Background: rgba(0,0,0,0.5);

Opacity: 0.5; 让盒子半透明，里面的内容也半透明（效果不好，不建议）

### div随着页面大小动态居中

position：absolute；

left：50%；

margin-left：-盒子的宽度；

### 上下左右样式

**左右：**

使用宋体格式的符号<>

**上下：**

使用∧∨：样式有点不够自然，不建议使用

使用◇

html：

<span>◇</span>

css：

span{

display: inline-block;

width: 25px;

height: 25px;

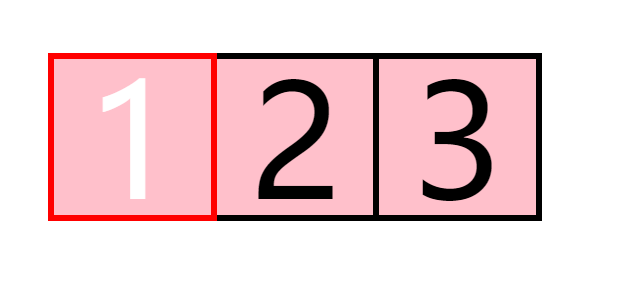
font-size: 37px;

color: white;

overflow: hidden;

}

### 实现并列框选择



html：

<ul>

<li>1</li>

<li>2</li>

<li>3</li>

</ul>

css：

ul{

list-style: none;

margin: 50px 50px;

font-size: 25px;

line-height: 25px;

}

li{

width: 25px;

height: 25px;

float: left;

background-color: pink;

border:solid 1px black;

text-align: center;

margin-left: -1px;

}

li:hover{

position:relative; /\*防止边框效果被遮挡\*/

/\*z-index:10;\*/ /\*预防里面有定位元素\*/

border:solid 1px red;

color: white;

}