

CSS

层叠样式表，负责布局美化网页，html专注于网页结构。

▼ 引入方式

样式表	优点	缺点	使用情况	控制范围
行内样式表	书写方便，权重高	结构样式混写	较少	控制一个标签
内部样式表	部分结构和样式相分离	没有彻底分离	较多	控制一个页面
外部样式表	完全实现结构和样式相分离	需要引入	最多，吐血推荐	控制多个页面

▪ 行内样式表

标签内部直接写style = "CSS属性：属性值；... "，适用于修改一些简单的样式。

▪ 内部样式表（嵌入式）

▪ 外联样式表

使用<link>标签引入.css文件

```
<link rel = "stylesheet" href = "样式表路径">
```

▪ 属性书写顺序

建议遵循以下顺序：

1. 布局定位属性：display / position / float / clear / visibility / overflow (建议 display第一个写，毕竟关系到模式)
2. 自身属性：width / height / margin / padding / border / background
3. 文本属性：color / font / text-decoration / text-align / vertical-align / white-space / break-word
4. 其他属性 (CSS3) : content / cursor / border-radius / box-shadow / text-shadow / background:linear-gradient...

▼ 元素显示模式

元素模式	元素排列	设置样式	默认宽度	包含
块级元素	一行只能放一个块级元素	可以设置宽度高度	容器的100%	容器级可以包含任何标签
行内元素	一行可以放多个行内元素	不可以直接设置宽度高度	它本身内容的宽度	容纳文本或则其他行内元素
行内块元素	一行放多个行内块元素	可以设置宽度和高度	它本身内容的宽度	

▪ 块元素

常见的块元素：

<h1-h6>、<p>、、、、<div>

1. 独占一行
2. 高度宽度以及内外边距都可以控制
3. 宽度默认是容器（父级）的100%
4. 是一个可以放行内元素或者块元素的盒子

注意：文字类标签内不能够使用块元素。

▪ 行内元素

常见的行内元素：

<a>、、、、<ins>、<u>

1. 行内元素只能容纳行内元素或者文本
2. 行内元素一行可以多个
3. 行内元素宽度是它本身内容的宽度
4. 行内元素无法直接设置宽高

注意：链接里面不能再放链接。

▪ 行内块元素

同时兼具行内元素和块元素的特点。

1. 一行可以有多个，之间有空白间隔
2. 默认宽度是其本身内容的宽度
3. 宽度和高度可以进行编辑

▪ 显示模式的转换

- 转换为块元素 : `display:block`
- 转换为行内元素 : `display:inline;`
- 转换为行内块 : `display:inline-block;`

▼ 文本

▼ 文本样式

▪ 文本颜色 color

文本颜色的属性值可以使用预定义颜色（blue, pink等等），也可以使用十六进制颜色值（#FF0000），或者RGB值（rgb(0,0,0)）

▪ 对齐文本 text-align

设置元素内文本对齐的方式，左对齐left，居中对齐center，右对齐right。（默认靠左对齐）

- 文本装饰 **text-decoration**

为文本添加删除线，下划线，上划线等等。属性值有：none, underline, overline, line-through

- 文本缩进 **text-indent**

设置元素内文本首行锁进大小，通常使用em作为单位，em是相对单位，1em是相对于当前元素的1个文字大小，如果当前元素没有设置，则按照父级元素一个文字大小进行缩进。

- 行间距 **line-height**

实际调整的是行间距中的上下间距两部分。

▼ 字体样式

- 字体系列 **font-family**

可以设置多个字体，每种字体之间用英文逗号间隔，如果字体名由空格则字体名用引号进行包裹，字体名支持中文，设置多种字体的时候按照浏览器兼容性依次进行显示。

- 字体大小 **font-size**

设置字体的大小，可以给body设置字体大小以设置整个网页的字体大小，字体的单位是px，即像素，谷歌浏览器的默认大小是16像素。

- 字体粗细 **font-weight**

100-900 设置字体的粗细，也可以用normal（相当于400， 默认不加粗）或者bold（相当于700）来设置。

- 字体风格 **font-style**

用来设置字体是否倾斜，默认为 normal 即不倾斜，可以更改为italic倾斜。通常我们会用这个属性来修正本来倾斜的字体，例如``标签。

- 字体的复合属性

综合书写字体样式。

```
body { font : font-style font-weight font-size/line-height  
       font-family; }
```

该顺序不能更改，不需要设置的属性可以不写（显示默认值），但是font-family和font-size必须保留，否则综合书写将不起作用。

- 文字阴影

值	描述
<i>h-shadow</i>	必需。水平阴影的位置。允许负值。
<i>v-shadow</i>	必需。垂直阴影的位置。允许负值。
<i>blur</i>	可选。模糊的距离。
<i>color</i>	可选。阴影的颜色。参阅 CSS 颜色值 。

`text-shadow: h-shadow v-shadow blur color;`

- ▼ 选择器

- ▼ 基础选择器

- 标签选择器

把某一类标签选择出来，并不能差异化显示。

`标签 { }`

- 类选择器

将选定的标签定义为某个类，然后对该类进行操作。

`.类 { }`

一个标签可以定义多个类名，用空格隔开。eg. `class = "类名1 类名2 ..."`

- ID选择器

`#id名 { }`

ID选择器与class选择器的区别在于，ID选择器只能调用一次，class可以重复调用。通常ID选择器用与页面上唯一的元素，与JS搭配使用。

- 通配符选择器

选择页面中的所有标签，特殊情况下使用。

`* { }`

- ▼ 复合选择器

对基础选择器的组合

- 后代选择器

`元素1 元素2 { 样式声明 }`

又被称为包含选择器，选择父元素里面的子元素。

- 子选择器

`元素1 > 元素2 { 样式声明 }`

子选择器只选择与父级元素最近的一级元素。

- 并集选择器

`元素1, 元素2 { 样式声明 }`

可以同时选择多个元素设定同一种样式。

▼ 伪类选择器

主要作用是向选择器添加特殊效果，主要特点是冒号。

▼ 链接伪类选择器

尽可能按照LVHA的顺序以避免出错。

- `a:link`

选择未被访问的链接

- `a:visited`

选择已被访问的链接

- `a:hover`

选择鼠标悬停的链接

- `a:active`

选择鼠标按下未弹起的链接

- `:focus伪类选择器`

通常用于选取获得了光标焦点的表单元素

`input:focus { 样式声明 }`

▼ 背景

- 背景颜色 `background-color`

默认是`transparent`

- 背景图片 `background-image`

背景图片非常利于控制位置，小logo和大背景都很常用。

属性值默认`none`，添加背景用`url(背景图片路径)`。

- 背景平铺

`background-repeat: repeat | no-repeat | repeat-x | repeat-y`

- 背景位置

`background-position: length | position`

`length`: 百分数、由浮点数字和单位标识符组成的长度值

`position`: top | bottom | left | right | center

- 背景附着

背景图片是否滚动 `background-attachment: scroll | fixed`

- 背景复合写法

`background:` 颜色 背景图片地址 平铺 是否滚动 图片位置

顺序并没有严格要求，一般按照以上顺序书写，空格隔开。

- 背景颜色半透明

`background: rgba (0,0,0,x)`

▼ 三大特性

选择器相同执行层叠性，选择器不同执行优先级。

- 层叠性

样式冲突，就近原则。

- 继承性

主要继承文字样式，text- font- line- color 等等

注意行高的继承可以使用浮点数让行高根据自己当前文字大小来实时调整。

- 优先级

继承/通配符 0、0、0、0

标签选择器 0、0、0、1

类选择器 0、0、1、0

ID选择器 0、1、0、0

行内样式表 1、0、0、0

! important 无穷大

(! important直接添加在样式后面)

注意：继承的权重为零，链接a被系统指定了默认样式，所以不执行继承的样式。

▼ 页面布局

页面布局基本过程：

1. 准备好相关的网页元素，网页元素基本上都是BOX
2. 利用CSS设置好盒子样式，将元素盒子摆放到合适的位置
3. 往盒子里面装内容

三种传统布局方式：普通流（标准流）；浮动；定位；

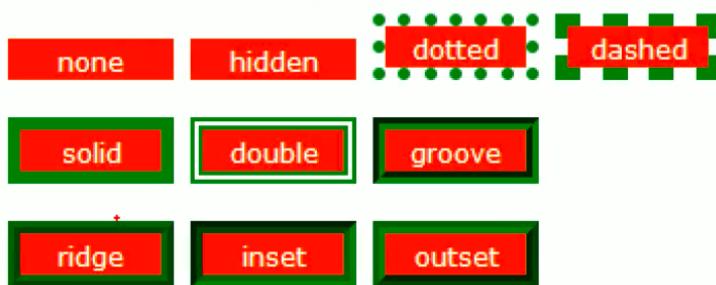
- 页面布局整体思路

1. 必须确定页面的版心（可视区），我们测量可得知。
2. 分析页面中的行模块，以及每个行模块中的列模块。其实页面布局第一准则。
3. 一行中的列模块经常浮动布局，先确定每个列的大小，之后确定列的位置。页面布局第二准则。
4. 制作HTML结构。我们还是遵循，先有结构，后有样式的原则。结构永远最重要。
5. 所以，先理清楚**布局结构**，再写代码尤为重要。这需要我们多写多积累。

- ▼ 盒子模型

由边框border、内边距padding、外边距margin、实际内容content四部分组成；

▼ 边框border



边框由边框颜色边框样式和边框宽度（粗细）组成；

border: border-color | border-width | border-style;

border-style: solid | dashed | dotted;

border-width: (px为单位)；

border-color: 颜色；

复合型写法：

border: 1px solid red; 三个属性没有顺序要求；

边框分开写：

border-(top bottom left right): +属性；

注意：边框具有层叠性；

border-collapse: collapse; 该属性表示相邻边框合并到一起，用于绘制表格细线边框；

注意：边框是会影响盒子大小的，设置盒子大小的时候注意将边框粗细考虑在内；

- 圆角边框

border-radius: length;

- 内边距padding

值的个数	表达意思
padding: 5px;	1个值，代表上下左右都有5像素内边距；
padding: 5px 10px;	2个值，代表上下内边距是5像素 左右内边距是10像素；
padding: 5px 10px 20px;	3个值，代表上内边距5像素 左右内边距10像素 下内边距20像素；
padding: 5px 10px 20px 30px;	4个值， 上是5像素 右10像素 下20像素 左是30像素 顺时针

padding-(top bottom left right): +px为单位的数值；

注意：

padding也会影响盒子的大小，例如添加内边距会撑大盒子；

如果盒子本身没有指定宽度和高度，那么padding就不会影响盒子的大小；

▼ 外边距margin

控制盒子与盒子之间的距离；

margin-(top bottom right left): +以px为单位的数值；

简写形式与padding的规则相同；

应用：

「外边距可以让块级盒子水平居中，但必须满足两个条件：

1. 盒子必须指定了宽度；2. 盒子左右外边距都设置为auto；

margin: 0 auto;」

注意：如果要让行内元素或者行内块元素居中对齐，可以给其父级元素添加 text-align: center;

▪ 外边距的合并问题

1. 相邻块元素垂直外边距的合并

当两个有上下外边距的块元素垂直相邻时，页面不会显示两个上下外边距之和，而会选择较大的一个显示，解决办法是只给一个盒子设置外边距；

2. 嵌套块元素垂直外边距的塌陷

两个有嵌套关系的块元素都有上外边距的时候，父元素会塌陷较大的上外边距；

解决办法：给父级元素添加上边框；给父级元素添加上内边距；使用overflow: hidden;

▪ 清除内外边距

不同的浏览器给不同的页面元素自带默认的内外边距，因此在布局前要清清除网页元素默认的内外边距；

```
* {  
    margin: 0;  
    padding: 0;  
}
```

这句代码通常为CSS的第一句代码；

行内元素尽量只设置左右外边距，由于兼容性问题上下外边距设置了也不一定有用；

▪ 盒子阴影

值	描述
<i>h-shadow</i>	必需。水平阴影的位置。允许负值。
<i>v-shadow</i>	必需。垂直阴影的位置。允许负值。
<i>blur</i>	可选。模糊距离。
<i>spread</i>	可选。阴影的尺寸。
<i>color</i>	可选。阴影的颜色。请参阅 CSS 颜色值。
<i>inset</i>	可选。将外部阴影 (<i>outset</i>) 改为内部阴影。

box-shadow: h-shadow V-shadow blur spread color inset;

盒子阴影不占用空间；

▼ 浮动

多个块级元素纵向排列找标准流，多个块级元素横向排列找浮动；

float属性用于创建浮动框，将其移动到一边，直到左边缘或者右边缘触及到包含块或另一个浮动框的边缘；

语法：选择器 { float: none; /left; /right; }

特性：

1. 脱标：浮动元素会脱离标准流；浮动的盒子不再保留原来的位置；
2. 浮动的元素会一行内显示并且顶部对齐
3. 浮动元素会具有行内块的特性；块级元素没有设置宽度的时候默认和父级元素一样宽；添加浮动后大小由内容决定；

通常先用标准流的父级元素排列上下位置，之后内部的子元素采取浮动排列左右位置；先设置盒子大小，再设置盒子位置；

浮动的盒子只会影响这个浮动盒子后面的标准流，前面的不会影响到；一般都是全浮（兄弟盒子）；

▼ 清除浮动

属性值	描述
left	不允许左侧有浮动元素（清除左侧浮动的影响）
right	不允许右侧有浮动元素（清除右侧浮动的影响）
both	同时清除左右两侧浮动的影响

父级盒子不方便设置高度，内部子盒子又要设置浮动的时候，为了消除浮动对后面标准流造成的影响，则需要清除浮动；

基本语法：选择器 {clear: 属性值; }

▪ 额外标签法

通俗易懂但是添加许多无用标签，结构性较差；

▪ 父级添加overflow属性

给父级盒子设置overflow属性，属性值设定为hidden、scroll、auto都可以；这种方法代码简洁，但是会隐藏掉溢出的部分；

- 父级添加after伪元素

```
.clearfix:after {  
    content: "";  
    display: block;  
    height: 0;  
    clear: both;  
    visibility: hidden;  
}  
.clearfix { /* IE6、7 专有 */  
    *zoom: 1;  
}
```

没有添加多余标签结构简单，但是需要照顾低版本的浏览器；

- 父级添加双伪元素

```
.clearfix:before, .clearfix:after {  
    content:"";  
    display:table;  
}  
.clearfix:after {  
    clear:both;  
}  
.clearfix {  
    *zoom:1;  
}
```

▼ 定位

▼ 为什么需要定位

1. 浮动可以让多个块级盒子一行没有缝隙排列显示，经常用于横向排列盒子。
2. 定位则是可以让盒子自由的在某个盒子内移动位置或者固定屏幕中某个位置，并且可以压住其他盒子。

- 定位的组成

定位：将盒子**定**在某一个**位**置，所以**定位也是在摆放盒子，按照定位的方式移动盒子。**

定位 = 定位模式 + 边偏移。

定位模式用于指定一个元素在文档中的定位方式。**边偏移**则决定了该元素的最终位置。

- 定位的模式

定位模式决定元素的定位方式，它通过CSS的position属性来设置，其值可以分为四个：

值	语义
static	静态定位
relative	相对定位
absolute	绝对定位
fixed	固定定位

- 边偏移

边偏移就是定位的盒子移动到最终位置。有top、bottom、left和right 4个属性。

边偏移属性	示例	描述
top	top: 80px	顶端偏移量，定义元素相对于其父元素上边线的距离。
bottom	bottom: 80px	底部偏移量，定义元素相对于其父元素下边线的距离。
left	left: 80px	左侧偏移量，定义元素相对于其父元素左边线的距离。
right	right: 80px	右侧偏移量，定义元素相对于其父元素右边线的距离

只有定位才有这四个属性

- ▼ 定位模式

- 静态定位

静态定位是元素的默认定位方式，无定位的意思。

语法：

```
选择器 { position: static; }
```

- 静态定位按照标准流特性摆放位置，它没有边偏移
- 静态定位在布局时很少用到

▪ 相对定位

相对定位是元素在移动位置的时候，是相对于它原来的位置来说的（自恋型）。

语法：

```
选择器 { position: relative; }
```

相对定位的特点：（务必记住）

1. 它是相对于自己原来的位置来移动的（移动位置的时候参照点是自己原来的位置）。
2. 原来在标准流的位置继续占有，后面的盒子仍然以标准流的方式对待它。（不脱标，继续保留原来位置）

▼ 绝对定位

绝对定位是元素在移动位置的时候，是相对于它祖先元素来说的（拼爹型）。

语法：

```
选择器 { position: absolute; }
```

绝对定位的特点：（务必记住）

1. 如果没有祖先元素或者祖先元素没有定位，则以浏览器为准定位（Document文档）。
2. 如果祖先元素有定位（相对、绝对、固定定位），则以最近一级的有定位祖先元素为参考点移动位置。
3. 绝对定位不再占有原先的位置。（脱标）

▪ 绝对定位盒子水平居中

1. 绝对定位的盒子居中

加了绝对定位的盒子不能通过 margin: 0 auto 水平居中，但是可以通过以下计算方法实现水平和垂直居中。

- ① left: 50%; 让盒子的左侧移动到父级元素的水平中心位置。
- ② margin-left: -100px; 让盒子向左移动自身宽度的一半。

▼ 固定定位

固定定位是元素**固定于浏览器可视区的位置**。主要使用场景：可以在浏览器页面滚动时元素的位置不会改变。

语法：

```
选择器 { position: fixed; }
```

固定定位的特点：(务必记住)

1. 以浏览器的可视窗口为参照点移动元素。
 - 跟父元素没有任何关系
 - 不随滚动条滚动。
2. 固定定位**不在占有原先的位置**。

固定定位也是脱标的，其实固定定位也可以看做是一种特殊的绝对定位。

▪ 小技巧

固定定位小技巧： 固定在版心右侧位置。

小算法：

1. 让固定定位的盒子 left: 50%. 走到浏览器可视区（也可以看做版心）的一半位置。
2. 让固定定位的盒子 margin-left: 版心宽度的一半距离。 多走 版心宽度的一半位置
就可以让固定定位的盒子贴着版心右侧对齐了。

▪ 粘性定位

粘性定位可以被认为是相对定位和固定定位的混合。 Sticky 粘性的

语法：

```
选择器 { position: sticky; top: 10px; }
```

粘性定位的特点：

1. 以浏览器的可视窗口为参照点移动元素（固定定位特点）
2. 粘性定位**占有原先的位置**（相对定位特点）
3. 必须添加 top、left、right、bottom 其中一个才有效
+ 跟页面滚动搭配使用。兼容性较差，IE不支持。

▪ 子绝父相

弄清楚这个口诀，就明白了绝对定位和相对定位的使用场景。

这个“子绝父相”太重要了，是我们学习定位的口诀，是定位中最常用的一种方式这句话的意思是：**子级是绝对定位的话，父级要用相对定位。**

- ① 子级绝对定位，不会占有位置，可以放到父盒子里面的任何一个地方，不会影响其他的兄弟盒子。
- ② 父盒子需要加定位限制子盒子在父盒子内显示。
- ③ 父盒子布局时，需要占有位置，因此父亲只能是相对定位。

这就是子绝父相的由来，所以**相对定位经常用来作为绝对定位的父级**。

总结：因为父级需要占有位置，因此是相对定位，子盒子不需要占有位置，则是绝对定位

当然，子绝父相不是永远不变的，如果父元素不需要占有位置，**子绝父绝**也会遇到。

▪ 定位叠放次序

在使用定位布局时，可能会出现盒子重叠的情况。此时，可以使用 **z-index** 来控制盒子的前后次序 (z轴)

语法：

```
选择器 { z-index: 1; }
```

- 数值可以是正整数、负整数或 0，默认是 auto，数值越大，盒子越靠上
- 如果属性值相同，则按照书写顺序，后来居上
- 数字后面不能加单位
- 只有定位的盒子才有 z-index 属性

▼ 定位的特殊性

绝对定位和固定定位也和浮动类似。

1. 行内元素添加绝对或者固定定位，可以直接设置高度和宽度。
2. 块级元素添加绝对或者固定定位，如果不给宽度或者高度，默认大小是内容的大小。

3. 脱标的盒子不会触发外边距塌陷

浮动元素、绝对定位(固定定位)元素的都不会触发外边距合并的问题。

4. 绝对定位（固定定位）会完全压住盒子

浮动元素不同，只会压住它下面标准流的盒子，但是不会压住下面标准流盒子里面的内容（图片）

但是绝对定位（固定定位）会压住下面标准流所有的内容。

浮动之所以不会压住文字，因为浮动产生的目的最初是为了做文字环绕效果的。文字会围绕浮动元素

港剧时代

1981年，刘德华考进第10期无线电视艺员训练班^[29]；同年，出演个人首部电视剧《江湖再见》，在剧中饰演以赈济妇女为主的小龙混阿龙，该剧获得美国电视节电视剧特别奖^[30]。

1982年，刘德华以甲级成绩从艺员训练班毕业后正式签约TVB^[31]，同年在喜剧《花艇小英雄》中饰演珠宝仔钱铁丁；12月，与叶德娴搭档主演时装喜剧《错觉》，凭借卧底警察江大伟一角获得关注^[32]。

1983年，主演金庸武侠剧《神雕侠侣》，在剧中饰演外貌俊俏、倜傥不羁的杨过^[33]，该剧在香港播出后取得62点的收视纪录；同年，与黄日华、梁朝伟、苗侨伟、汤镇业组成“无线五虎将”^[34]。

1984年，与赵雅芝合作主演古装武侠剧《魔域桃源》，在剧中饰演资质出众、武功高强的傅青云^[35]；同年，与梁朝伟共同主演金庸武侠剧《鹿鼎记》，在剧中饰演英明果断的康熙^[36]。

1985年，在古装武侠剧《杨家将》中饰演骁勇善战的杨六郎^[37]；同年，TVB向刘德华提出加签五年的合约，刘德华因拒绝而被TVB雪藏400天^[38-39]。1986年，在邵逸夫的调解下，刘德华与TVB和解并签下合约；同年，主演古装剧《真命天子》。1988年，在出演了武侠剧《天龙八部》后，刘德华将演艺事业的重心转向影坛^[39]。



83年刘德华饰王莲斯《神…

▪ 定位总结

定位模式	是否脱标	移动位置	是否常用
static 静态定位	否	不能使用边偏移	很少
relative 相对定位	否(占有位置)	相对于自身位置移动	常用
absolute 绝对定位	是(不占有位置)	带有定位的父级	常用
fixed 固定定位	是(不占有位置)	浏览器可视区	常用
sticky 粘性定位	否(占有位置)	浏览器可视区	当前阶段少

▼ 页面元素的显示与隐藏

display 显示隐藏元素但是不留位置

visibility 显示隐藏元素 但是保留原来的位置

overflow 溢出显示隐藏 但是只是对于溢出的部分处理

▪ display

display 属性用于设置一个元素应如何显示。

- display: none ; 隐藏对象
- display : block ; 除了转换为块级元素之外，同时还有显示元素的意思

display 隐藏元素后，不再占有原来的位置。

后面应用及其广泛，搭配 JS 可以做很多的网页特效。

▪ visibility

visibility 属性用于指定一个元素应可见还是隐藏。

- visibility : visible; 元素可视
- visibility : hidden; 元素隐藏

visibility 隐藏元素后，继续占有原来的位置。

如果隐藏元素想要原来位置，就用 visibility : hidden

如果隐藏元素不想要原来位置，就用 display : none (用处更多 重点)

▪ overflow

overflow 属性指定了如果内容溢出一个元素的框（超过其指定高度及宽度）时，会发生什么。

属性值	描述
visible	不剪切内容也不添加滚动条
hidden	不显示超过对象尺寸的内容，超出的部分隐藏掉
scroll	不管超出内容否，总是显示滚动条
auto	超出自动显示滚动条，不超出不显示滚动条

一般情况下，我们都不想让溢出的内容显示出来，因为溢出的部分会影响布局。

但是如果有定位的盒子，请慎用 overflow:hidden 因为它会隐藏多余的部分。

▪ 页面布局总结

通过盒子模型，清楚知道大部分html标签是一个盒子。

通过CSS浮动、定位可以让每个盒子排列成为网页。

一个完整的网页，是标准流、浮动、定位一起完成布局的，每个都有自己的专门用法。

1. 标准流



可以让盒子上下排列或者左右排列，**垂直的块级盒子显示就用标准流布局。**

2. 浮动

可以让多个块级元素一行显示或者左右对齐盒子，**多个块级盒子水平显示就用浮动布局。**

3. 定位

定位最大的特点是有层叠的概念，就是可以让多个盒子前后叠压来显示。**如果元素自由在某个盒子内移动就用定位布局。**

▼ CSS高级

▼ 精灵图

▪ 为什么要使用精灵图

一个网页中往往应用很多小的背景图像作为修饰，当网页中的图像过多时，服务器就会频繁地接收和发送请求图片，造成服务器请求压力过大，这将大大降低页面的加载速度。

因此，**为了有效地减少服务器接收和发送请求的次数，提高页面的加载速度**，出现了**CSS精灵技术**（也称CSS Sprites、CSS 雪碧）。

核心原理：将网页中的一些小背景图像整合到一张大图中，这样服务器只需要一次请求就可以了。

▪ 精灵图的使用

使用精灵图核心总结：

1. 精灵图主要针对于小的背景图片使用。
2. 主要借助于**background-position**。
3. 一般情况下精灵图都是负值。（千万注意网页中的坐标：x轴右边走是正值，左边走是负值，y轴同理。）

▼ 字体图标

▪ 字体图标的产生

字体图标使用场景：主要用于显示网页中通用、常用的一些小图标。

精灵图是有诸多优点的，但是缺点很明显。

1. 图片文件还是比较大的。
2. 图片本身放大和缩小会失真。
3. 一旦图片制作完毕想要更换非常复杂。

此时，有一种技术的出现很好的解决了以上问题，就是**字体图标iconfont**。

字体图标可以为前端工程师提供一种方便高效的图标使用方式，**展示的是图标，本质属于字体**。

▪ 字体图标的优势

- 轻量级：一个图标字体要比一系列的图像要小。一旦字体加载了，图标就会马上渲染出来，减少了服务器请求
- 灵活性：本质其实是文字，可以很随意的改变颜色、产生阴影、透明效果、旋转等
- 兼容性：几乎支持所有的浏览器，请放心使用

注意：字体图标不能替代精灵技术，只是对工作中图标部分技术的提升和优化。

▪ 总结

1. 如果遇到一些结构和样式比较简单的小图标，就用字体图标。
2. 如果遇到一些结构和样式复杂一点的小图片，就用精灵图。 |

▼ CSS三角

```
div {  
    width: 0;  
    height: 0;  
    line-height: 0;  
    font-size: 0; +  
    border: 50px solid transparent;  
    border-left-color: pink;  
}
```

- 特殊三角



代码：

```
width: 0;  
height: 0;  
border-color: transparent red transparent transparent;  
border-style: solid;  
border-width: 22px 8px 0 0;
```

▼ 用户界面样式

- 鼠标样式cursor

```
li {cursor: pointer; }
```

设置或检索在对象上移动的鼠标指针采用何种系统预定义的光标形状。

属性值	描述
default	小白 默认
pointer	小手
move	移动
text	文本
not-allowed	禁止

- 轮廓线

给表单添加 outline: 0; 或者 outline: none; 样式之后，就可以去掉默认的蓝色边框。

```
input {outline: none; }
```

- 防止拖拽文本域

实际开发中，我们文本域右下角是不可以拖拽的。

```
textarea{ resize: none; }
```

文本域尽量单独放在一行

- ▼ vertical-align

```
vertical-align : baseline | top | middle | bottom
```

- 应用

图片、表单都属于行内块元素，默认的vertical-align是基线对齐。



此时可以给图片、表单这些行内块元素的vertical-align属性设置为middle就可以让文字和图片垂直居中对齐了。

- 图片底部空白BUG

bug：图片底侧会有一个空白缝隙，原因是行内块元素会和文字的基线对齐。

主要解决方法有两种：

1. 给图片添加vertical-align:middle | top| bottom等。（提倡使用的）
2. 把图片转换为块级元素 display:block;

- ▼ 溢出文字省略号

- 单行文本需满足三个条件

```
/*1. 先强制一行内显示文本*/
white-space: nowrap; (默认 normal 自动换行)
/*2. 超出的部分隐藏*/
overflow: hidden;
/*3. 文字用省略号替代超出的部分*/
text-overflow: ellipsis;
```

- 多行文本有兼容性问题

多行文本溢出显示省略号，有较大兼容性问题，适合于webkit浏览器或移动端（移动端大部分是webkit内核）

```
overflow: hidden;
text-overflow: ellipsis;
/* 弹性伸缩盒子模型显示 */
display: -webkit-box;
/* 限制在一个块元素显示的文本的行数 */
-webkit-line-clamp: 2;
/* 设置或检索伸缩盒对象的子元素的排列方式 */
-webkit-box-orient: vertical;
```

▼ 常见布局技巧

- Margin负值的运用

- 1.让每个盒子margin往左侧移动 -1px 正好压住相邻盒子边框
- 2.鼠标经过某个盒子的时候，提高当前盒子的层级即可（如果没有定位，则加相对定位（保留位置），如果有定位，则加z-index）

消除相邻盒子的边框变粗问题

- 浮动元素实现文字环绕图片效果
- 行内块元素适用于一排有间隔的小盒子
- CSS初始化

不同浏览器对有些标签的默认值是不同的，为了消除不同浏览器对HTML文本呈现的差异，照顾浏览器的兼容，我们需要对CSS初始化

简单理解：CSS初始化是指重设浏览器的样式。（也称为CSS reset）

每个网页都必须首先进行CSS初始化。

▼ CSS3

- 新增的CSS3特性有兼容性问题，ie9+才支持
- 移动端支持优于PC端
- 不断改进中
- 应用相对广泛
- 现阶段主要学习：新增选择器和盒子模型以及其他特性

▼ CSS3新增选择器

▪ 属性选择器

选择符	简介
E[att]	选择具有 att 属性的 E 元素
E[att="val"]	选择具有 att 属性且属性值等于 val 的 E 元素
E[att^="val"]	匹配具有 att 属性且值以 val 开头的 E 元素
E[att\$="val"]	匹配具有 att 属性且值以 val 结尾的 E 元素
E[att*="val"]	匹配具有 att 属性且值中含有 val 的 E 元素

属性选择器可以根据元素的特定属性来进行选择，而不用借助于ID或者类；

▼ 结构伪类选择器

选择符	简介
E:first-child	匹配父元素中的第一个子元素 E
E:last-child	匹配父元素中最后一个 E 元素
E:nth-child(n)	匹配父元素中的第 n 个子元素 E
E:first-of-type	指定类型 E 的第一个
E:last-of-type	指定类型 E 的最后一个
E:nth-of-type(n)	指定类型 E 的第 n 个

结构伪类选择器主要根据文档结构选择元素，常用于根据父级选择子级；前三个不论type，先排序；后三个先选指定元素，再给指定元素排序；

- E: nth-child(n)

nth-child (n) 选择某个父元素的一个或多个特定的子元素

- n 可以是数字，关键字和公式
- n 如果是数字，就是选择第 n 个子元素，里面数字从1开始...
- n 可以是关键字：even 偶数，odd 奇数
- n 可以是公式：常见的公式如下（如果n是公式，则从0开始计算，但是第 0 个元素或者超出了元素的个数会被忽略）

公式	取值
2n	偶数
2n+1	奇数
5n	5 10 15 ...
n+5	从第5个开始（包含第五个）到最后
-n+5	前5个（包含第5个）...

- 总结

- 结构伪类选择器一般用于选择父级里面的第几个孩子
- nth-child 对父元素里面所有孩子排序选择（序号是固定的）先找到第n个孩子，然后看看是否和E匹配
- nth-of-type 对父元素里面指定子元素进行排序选择。先去匹配E，然后再根据E找第n个孩子
- 关于 nth-child (n) 我们要知道 n 是从 0 开始计算的，要记住常用的公式
- 如果是无序列表，我们肯定用 nth-child 更多
- 类选择器、属性选择器、伪类选择器，权重为 10。

▼ 伪元素选择器

伪元素选择器可以帮助我们利用CSS创建新标签元素，而不需要HTML标签，从而简化HTML结构。

选择符	简介
::before	在元素内部的前面插入内容
::after	在元素内部的后面插入内容

注意：

- before 和 after 创建一个元素，但是属于行内元素
- 新创建的这个元素在文档树中是找不到的，所以我们称为**伪元素**
- 语法： element::before {}
- before 和 after 必须有 content 属性
- before 在父元素内容的前面创建元素，after 在父元素内容的后面插入元素
- 伪元素选择器和标签选择器一样，权重为 1

- 伪元素字体图标

```
p::before {  
    position: absolute;  
    right: 20px;  
    top: 10px;  
    content: '\ue91e';  
    font-size: 20px;  
}
```

- CSS3盒子模型

CSS3 中可以通过 **box-sizing** 来指定盒模型，有2个值：即可指定为**content-box**、**border-box**，这样我们计算盒子大小的方式就发生了改变。

可以分成两种情况：

1. **box-sizing: content-box** 盒子大小为 width + padding + border (以前默认的)
2. **box-sizing: border-box** 盒子大小为 width

如果盒子模型我们改为了**box-sizing: border-box**，那padding和border就不会撑大盒子了(前提padding和border不会超过width宽度)

- ▼ CSS3其他特性

- 滤镜filter

filter: 函数(); 例如：`filter: blur(5px);` blur模糊处理 数值越大越模糊

- calc函数

calc() 此CSS函数让你在声明CSS属性值时执行一些计算。

```
width: calc(100% - 80px);
```

括号里面可以使用 `+ - * /` 来进行计算。

- ▼ 过渡特性

▪ 简介

过渡 (transition)是CSS3中具有颠覆性的特征之一，我们可以在不使用 Flash 动画或 JavaScript 的情况下，当元素从一种样式变换为另一种样式时为元素添加效果。

过渡动画：是从一个状态 慢慢的过渡到另外一个状态

可以让我们页面更好看，更动感十足，虽然 低版本浏览器不支持 (ie9以下版本) 但是不会影响页面布局。

我们现在经常和 :hover 一起 搭配使用。

如果写多个过渡用逗号分隔；

▪ 使用

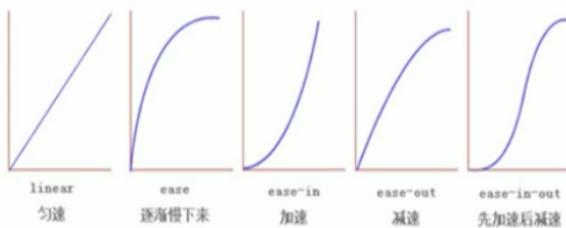
```
transition: 要过渡的属性 花费时间 运动曲线 何时开始;
```

1. 属性：想要变化的 css 属性，宽度高度 背景颜色 内外边距都可以。如果想要所有的属性都变化过渡，写一个all 就可以。

2. 花费时间：单位是 秒 (必须写单位) 比如 0.5s

3. 运动曲线：默认是 ease (可以省略)

4. 何时开始：单位是 秒 (必须写单位) 可以设置延迟触发时间 默认是 0s (可以省略)



记住过渡的使用口诀：谁做过渡给谁加

▼ CSS3的2D转换

转换 transform：旋转，缩放，位移；

- 位移translate

2D移动是2D转换里面的一种功能，可以改变元素在页面中的位置，类似**定位**。

1. 语法

```
transform: translate(x, y); 或者分开写  
transform: translateX(n);  
transform: translateY(n);
```

2. 重点

- 定义 2D 转换中的移动，沿着 X 和 Y 轴移动元素
- translate最大的优点：不会影响到其他元素的位置
- translate中的百分比单位是相对于自身元素的translate:(50%,50%);
- 对行内标签没有效果

- ▼ 旋转rotate

2D旋转指的是让元素在2维平面内顺时针旋转或者逆时针旋转。

1. 语法

```
transform:rotate(度数)
```

2. 重点

- rotate里面跟度数，单位是deg 比如 rotate(45deg)
- 角度为正时，顺时针，负时，为逆时针
- 默认旋转的中心点是元素的中心点

- 设置旋转中心点

1. 语法

```
transform-origin: x y;
```

2. 重点

- 注意后面的参数 x 和 y 用空格隔开
- x y 默认转换的中心点是元素的中心点 (50% 50%)
- 还可以给x y 设置 像素 或者 方位名词 (top bottom left right center)

- 缩放scale

缩放，顾名思义，可以放大和缩小。只要给元素添加上了这个属性就能控制它放大还是缩小。

1. 语法

```
transform:scale(x,y);
```

2. 注意

- 注意其中的x和y用逗号分隔
 - transform:scale(1,1) : 宽和高都放大一倍，相对于没有放大
 - transform:scale(2,2) : 宽和高都放大了2倍
 - transform:scale(2) : 只写一个参数，第二个参数则和第一个参数一样，相当于 scale(2,2)
 - transform:scale(0.5,0.5) : 缩小
 - scale缩放最大的优势：可以设置转换中心点缩放，默认以中心点缩放的，而且不影响其他盒子
- 2D转换综合书写

注意：

1. 同时使用多个转换，其格式为：transform: translate() rotate() scale() ...等，
2. 其顺序会影响转换的效果。（先旋转会改变坐标轴方向）
3. 当我们同时有位移和其他属性的时候，记得要将位移放到最前

▪ 2D转换总结

- 转换transform 我们简单理解就是变形有2D 和 3D 之分
- 我们暂且学了三个分别是 位移 旋转 和 缩放
- 2D 移动 translate(x, y) 最大的优势是不影响其他盒子，里面参数用%，是相对于自身宽度和高度来计算的
- 可以分开写比如 translateX(x) 和 translateY(y)
- 2D 旋转 rotate(度数) 可以实现旋转元素 度数的单位是deg
- 2D 缩放 scale(x,y) 里面参数是数字不跟单位 可以是小数 最大的优势 不影响其他盒子
- 设置转换中心点 transform-origin : x y; 参数可以百分比、像素或者是方位名词
- 当我们进行综合写法，同时有位移和其他属性的时候，记得要将位移放到最前

▼ CSS3的3D转换

▪ 页面中的三维坐标系

- x轴：水平向右 注意：x 右边是正值，左边是负值
- y轴：垂直向下 注意：y 下面是正值，上面是负值
- z轴：垂直屏幕 注意：往外面是正值，往里面是负值

▪ 透视

6.3 透视 perspective

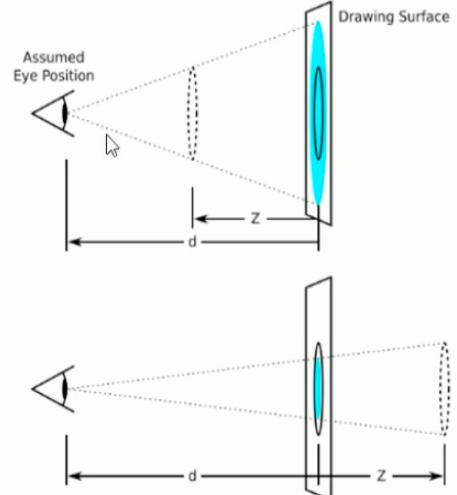
在2D平面产生近大远小视觉立体，但是只是效果二维的

- 如果想要在网页产生3D效果需要透视（理解成3D物体投影在2D平面内）。
- 模拟人类的视觉位置，可认为安排一只眼睛去看
- 透视我们也称为视距：视距就是人的眼睛到屏幕的距离
- 距离视觉点越近的在电脑平面成像越大，越远成像越小
- 透视的单位是像素

透视写在被观察元素的父盒子上面的

d：就是视距，视距就是一个距离人的眼睛到屏幕的距离。

z：就是 z轴，物体距离屏幕的距离，z轴越大（正值）我们看到的物体就越大。



▪ 位移translate3d

3D移动在2D移动的基础上多加了一个可以移动的方向，就是z轴方向。

- transform:translateX(100px)：仅仅是在x轴上移动
- transform:translateY(100px)：仅仅是在Y轴上移动
- transform:translateZ(100px)：仅仅是在Z轴上移动（注意：translateZ一般用px单位）
- transform:translate3d(x,y,z)：其中 x、y、z 分别指要移动的轴的方向的距离

- 旋转rotate3d

3D旋转指可以让元素在三维平面内沿着x轴，y轴，z轴或者自定义轴进行旋转。

语法

- transform:rotateX(45deg) : 沿着x轴正方向旋转 45度
- transform:rotateY(45deg) : 沿着y轴正方向旋转 45deg
- transform:rotateZ(45deg) : 沿着Z轴正方向旋转 45deg
- transform:rotate3d(x,y,z,deg) : 沿着自定义轴旋转 deg为角度 (了解即可)

旋转方向参考左手法则；

- 3D呈现 transform-style

- 控制子元素是否开启三维立体环境。。
- transform-style: flat 子元素不开启3d立体空间 默认的
- transform-style: preserve-3d; 子元素开启立体空间
- 代码写给父级，但是影响的是子盒子
- 这个属性很重要，后面必用

▼ CSS3动画

- 意义

动画 (animation) 是CSS3中具有颠覆性的特征之一，可通过设置多个节点来精确控制一个或一组动画，常用来实现复杂的动画效果。

相比较过渡，动画可以实现更多变化，更多控制，连续自动播放等效果。

▼ 使用

- 定义动画

```
@keyframes 动画名称 {  
    0%{  
        width:100px;  
    }  
    100%{  
        width:200px;  
    }  
}
```

▪ 元素调用

```
div {  
    width: 200px;  
    height: 200px;  
    background-color: aqua;  
    margin: 100px auto;  
    /* 调用动画 */  
    animation-name: 动画名称;  
    /* 持续时间 */  
    animation-duration: 持续时间;  
}
```

▪ 动画序列

- 0% 是动画的开始，100% 是动画的完成。这样的规则就是动画序列。
- 在 `@keyframes` 中规定某项 CSS 样式，就能创建由当前样式逐渐改为新样式的动画效果。
- 动画是使元素从一种样式逐渐变化为另一种样式的效果。您可以改变任意多的样式任意多的次数。
- 请用百分比来规定变化发生的时间，或用关键词 "from" 和 "to"，等同于 0% 和 100%。

▪ 常用动画属性

属性	描述
<code>@keyframes</code>	规定动画。
<code>animation</code>	所有动画属性的简写属性，除了 <code>animation-play-state</code> 属性。
<code>animation-name</code>	规定 <code>@keyframes</code> 动画的名称。（必须的）
<code>animation-duration</code>	规定动画完成一个周期所花费的秒或毫秒，默认是0。（必须的）
<code>animation-timing-function</code>	规定动画的速度曲线，默认是“ease”。
<code>animation-delay</code>	规定动画何时开始，默认是0。
<code>animation-iteration-count</code>	规定动画被播放的次数，默认是1，还有 <code>infinite</code>
<code>animation-direction</code>	规定动画是否在下一周期逆向播放，默认是“normal”， <code>alternate</code> 逆播放
<code>animation-play-state</code>	规定动画是否正在运行或暂停。默认是“running”，还有“pause”。
<code>animation-fill-mode</code>	规定动画结束后状态，保持 <code>forwards</code> 回到起始 <code>backwards</code>

▪ 动画简写

animation : 动画名称 持续时间 运动曲线 何时开始 播放次数 是否反方向 动画起始或者结束的状态;

```
animation: myfirst 5s linear 2s infinite alternate;
```

- 简写属性里面不包含 animation-play-state
- 暂停动画 : animation-play-state: paused; 经常和鼠标经过等其他配合使用
- 想要动画走回来 , 而不是直接跳回来 : animation-direction : alternate
- 盒子动画结束后 , 停在结束位置 : animation-fill-mode : forwards

▪ 速度曲线细节

animation-timing-function : 规定动画的速度曲线 , 默认是 "ease"

值	描述
linear	动画从头到尾的速度是相同的。匀速
ease	默认。动画以低速开始 , 然后加快 , 在结束前变慢。
ease-in	动画以低速开始。
ease-out	动画以低速结束。
ease-in-out	动画以低速开始和结束。
steps()	指定了时间函数中的间隔数量 (步长)

- 浏览器私有前缀

浏览器私有前缀是为了兼容老版本的写法，比较新版本的浏览器无须添加。

1. 私有前缀

- -moz- : 代表 firefox 浏览器私有属性
- -ms- : 代表 ie 浏览器私有属性
- -webkit- : 代表 safari、chrome 私有属性
- -o- : 代表 Opera 私有属性

2. 提倡的写法

```
-moz-border-radius: 10px;  
-webkit-border-radius: 10px;  
-o-border-radius: 10px;  
border-radius: 10px;
```