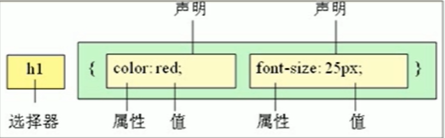
# 一、css基本概述

1）<style></style>

Css主要的构成部分：选择器以及一条或者多条声明

Font-size：12px 修改文字大小

Color：red 修改颜色



属性和属性值以键值对出现

&lt;表示<

&rt;表示>

&nbsp；表示空格

2、选择器分类

选择器分为基础选择器和复合选择器

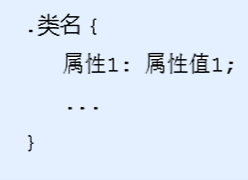
1、基础选择器：标签选择器，类选择器、id选择器和通配符选择器

1. 标签选择

指标签名作为选择器

优点能迅速的把同类型的标签设置样式

1. 类选择器

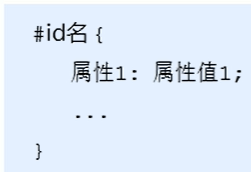


谁想使用谁调用 clss类调用 Clss=”类名” <div class="red">啦啦啦</div>

多类名使用方式

<div class="red" font35>啦啦啦</div>

1. id选择器

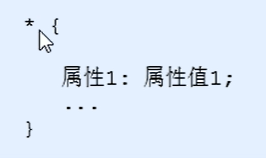


 Id选择器 id=“类名”<div id="green">啦啦啦</div>

与类选择器区别：Id选择不能被别人使用 是唯一的

1. 通配符选择器

将所有的标签都改为规定样式 不需要调用



1. 复合选择器

选择器1，

选择器2 {

}

## css字体属性

**1）Font-family**： “宋体”，微软雅黑； 多个字体提升兼容性

**2）Body { Font-size： 20px；}** 定义文字大小

标题标签比较特殊，需要单独指定文字大小

**3）Font-weight：700**；定义字体大小（后面不要跟单位）

700加粗 400正常

**4）Font-style：normal** 默认值 italic倾斜 文字样式

**5）字体的复合属性**

 div {

     font: italic 700 16px/28px 'microsoft yahei';

 }

**必须按照上面书法顺序，不能更改顺序**

不需要的属性可以省略（取默认值），但必须保留font-size字号和font-family字体属性，否则不起作用



## 4、css文本属性

属性可定义文本的外观；比如文本的颜色、对齐文本、装饰文本、文本的缩进、行间距等

1. **文本的颜色**

Div {

Color：red；

}

Rgb（255，0，0）

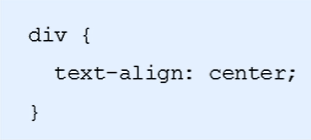
#ff0000

十六进制使用最多

1. **对齐文本**

Text-align：center设置元素内容水平对齐方式（行内元素和行内块元素）

line-height：像素 设置元素内容垂直对齐方式 让文字的行高等于盒子的高度



Left默认值 right

1. **装饰文本**

Text-decoration：属性



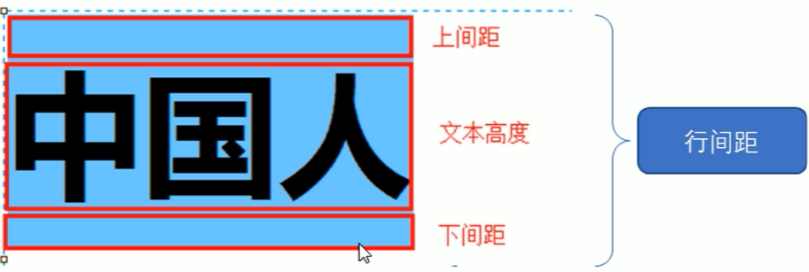
1. **文本缩进** （首行缩进）

Text-indent： 数值px/em；

Em：是缩进当前元素文字大小距离

1. **行间距设置行间的距离（行高：文字大小+上下间距）** 改变上下间距

Line-height：数值px；





## 5、css引入方式

1、**行内样式表（行内式）**2、**内部样式表（嵌入式）**3、**外部样式表（链接式）**

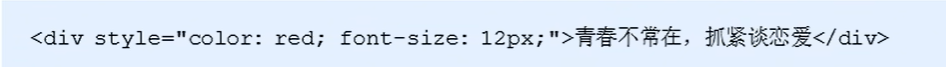
1. **内部样式表（嵌入式）**

是写到html的页面内部，是将所有的css代码抽取出来，单独放到<style>标签中



1. **行内样式表（行内式）**

直接在标签内部改写



1. **外部样式表（链接式）**

适用于样式比较多的情况 是开发中用的最多的

新建一个css文件

用link引入html

<link rel="stylesheet" href="文件路径">

# 二、css的复合选择器与特性

1. **后代选择器（重要**）：又称为包含选择器，可以选择父元素里面的子元素

 ol li {

        color: pink;

    }

样式规则： 选择器1 选择器2 ……{ 样式声明 }

元素二被修改

1. **子元素选择器（子选择器）**只能选择作为某元素的最近一级子元素。就是亲儿子元素

样式规则： 选择器1>选择器2 { 样式声明 }

\*\*元素二必须是亲儿子

1. **并集选择器（重要）**：可以选择多组标签，同时为他们定义相同的样式。通常用于集体声明

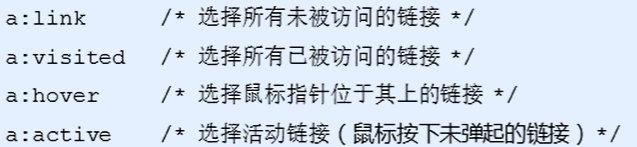
样式规则：选择器1，选择器2 ，……{ 样式声明 }

选择器1和选择器2的元素都被修改

\*\*并集选择器喜欢竖着写

1. **伪类选择器：**用于向某些选择器添加特殊效果，比如给链接添加特殊效果，或者选择第一个，第n个元素

**4.1）链接伪类选择器**



Lv哈哈哈

\*\*主要用于链接

\*\*最好必须按照顺序

\*\*给链接指定样式必须单独

**4.2）focus伪类选择器**

：focus用于选取获得焦点的表单元素

只要是<input>表单元素获取焦点就采用样式

Input：focus {

Background-color：pink；

}

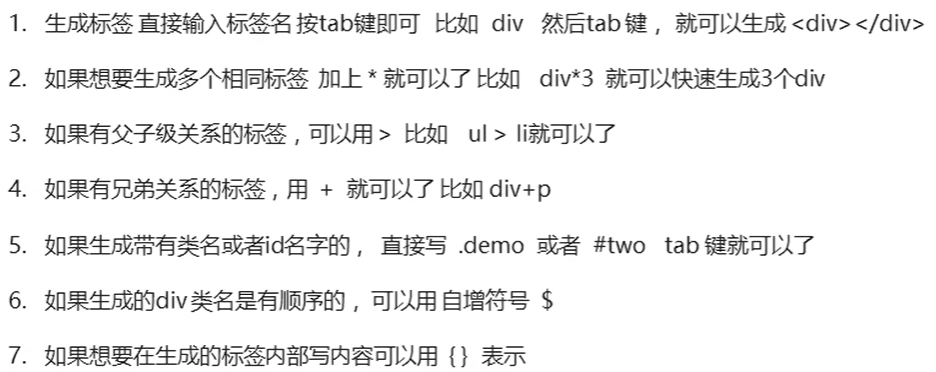
1. **兄弟选择器：**选择符合条件的**相邻兄弟所有元素**

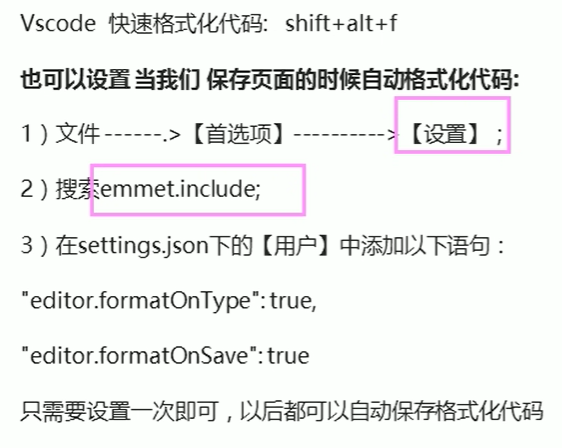
样式规则：选择器1+选择器2

6）**通用兄弟选择器：**符合条件的**所有兄弟选择**

样式规则：选择器1~选择器2

## 2、Emmet语法





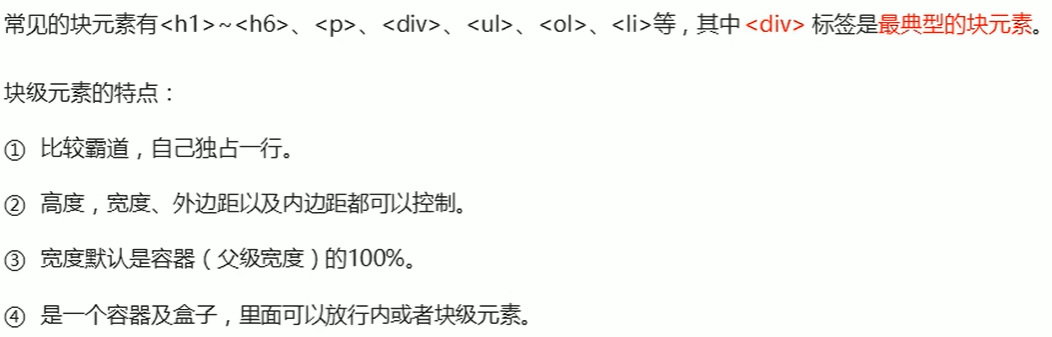
Shift+alt在多行输入文字

## css的元素显示模式

元素的显示模式就是元素（标签）以什么方式进行显示，比如<div>自己占一行，比如一行放很多<span>

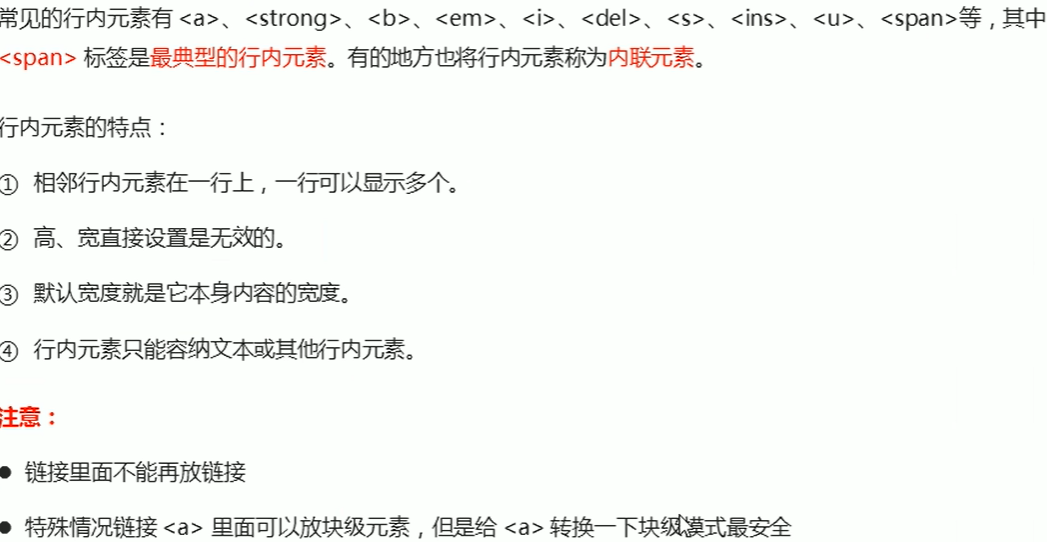
**Html元素一般分为块元素和行元素**

1. **块元素**

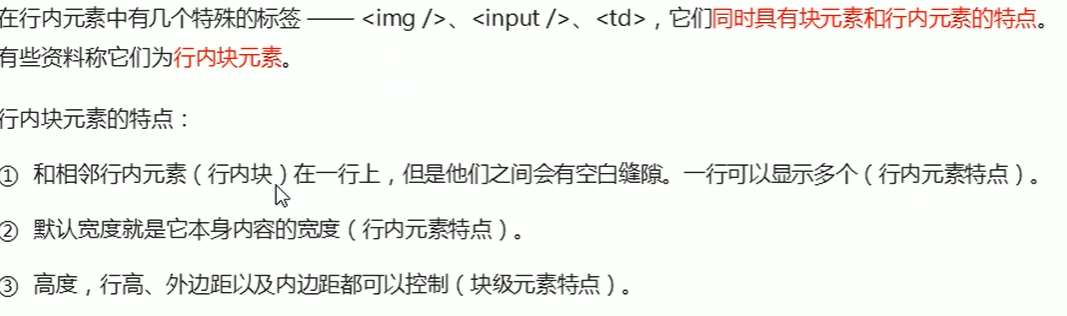


**注意：文字类的元素內不能使用块级元素，比如<p>标签<h1……h6>标签**

**2）行内元素**



**3）行内块元素**



**4）元素显示模式转换**

简单理解：一个模式的元素需要转换成另一种模式的特性比如想要增加链接<a>的触发范围

Display：block转换为块级元素 写在文本属性里面

Display：inline 转换为行内元素 写在文本属性里面

Display：inline-block 换成为行内块元素 写在文本属性里面

## css背景

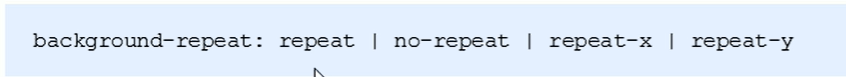
1. **背景颜色**

-color：transparent（默认透明）

1. **背景图片**

background-image: url(地址)/none;

1. **背景平铺**

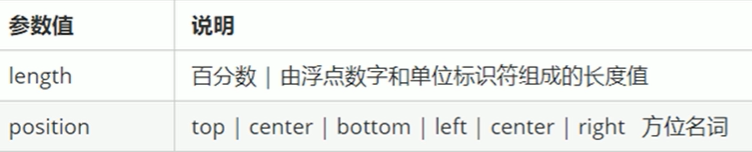


1. **背景图片位置**

Background-position：x y；

x和y为方位名词不分前后

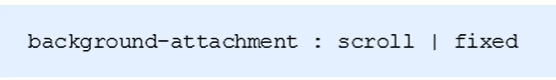
如果省略y，默认居中对齐



Background-position：xpx ypx；

如果指定一个数值，那么该数值一定是x坐标，另一个默认居中

1. **背景图像固定（背景附着）**



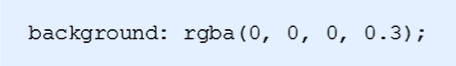
Scroll：背景图像是随着对象内容滚动

Fixed：背景图像固定

背景复合写法



1. **背景色半透明**





1. **背景图片大小**

Background-size：宽 高；

单位：长度px | 百分比（相对于父盒子的百分比） | cover(长款一直拉伸直到把整个盒子覆盖) | contain(高度和宽度等比例拉伸，当宽度或者高度铺满div盒子就不再拉伸了)

只写一个参数 肯定是宽度 高度会等比例拉伸

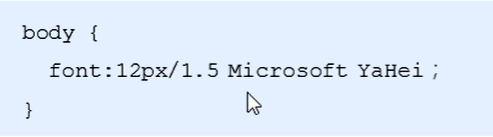
## 4、背景色线性渐变

 background: -webkit-linear-gradient(起始方向，颜色1，颜色2，……); 默认是从上到下

背景颜色必须添加浏览器私有前缀

## 5、css三个特性：层叠性、继承性、优先级

1. **层叠性**解决样式冲突，遵循就近原则；样式不冲突不层叠
2. **继承性**子标签会继承父标签的某些样式，如文本颜色和字号行高。简单理解：子承父业



1.5是当前元素大小的1.5倍行高

1. **优先级**

选择器相同，则执行层叠性

选择器不同，则根据选择器权重执行



\*\*继承的权重是0，如果该元素没有直接选中，不管父元素权重多高，子元素都是0

权重叠加：如果是复合选择器，则会有权重叠加，需要计算权重

# 三、CSS盒子模型



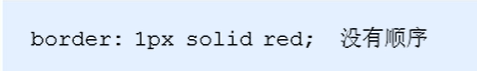
**页面布局三大核心：盒子模型，浮动和定位**

## 1、边框border



Style：Solid 实线边框 dashed 虚线边框 dotted点线边框

边框的复合型写法：



Border-top/bottom/right/left：属性px

Border-collapse：collapse 合并相邻的边框

## 2、内边距

Padding-top/bottom/right/left：属性px

复合写法



padding坏处会撑开盒子大小 不指定宽度就不会撑大

Padding好处可以根据文字数量调整单元格大小 （新浪导航栏）

加padding撑开盒子

## 3、外边距

3.2）Margin-left/right/top/bottom：属性px

Margin和padding的简写意义一致

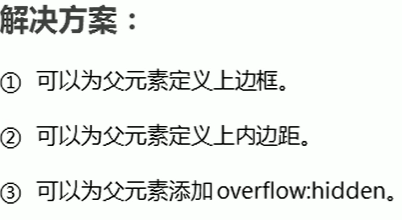
3.2）外页边距可以让**块盒子**水平居中，**但必须满足两个条件**：盒子必须有宽度，左右外边距都设置为auto（自动）margin：0 auto；

3.3）行内元素或者行内块元素水平居中给其父元素添加text-align：center即可

两个盒子之间距离可添加Margin来分开

3.4）对于两个嵌套关系的块元素，父元素有上外边距同时子元素也有上外边距，此时父元素会塌陷较大的外边距值

浮动的盒子不会有外边距塌陷的问题



## 4、清除内外边距

很多网页元素都默认带有内外边距

\* {

        padding: 0;

        margin: 0;

Box-sizing：border-box

}

Css的第一行代码

## 5、圆角边框

1、Border-radius：length； length表数值px,可以写四个

Radius 圆的半径

正方形设置为圆形将Border-radius：设置为长或者宽的百分之50%

长方形设置圆角矩形将Border-radius：设置为高度的百分之50%



顺序不能颠倒

## 6、盒子阴影

1. box-shadow：属性



默认outset不应该写 内阴影inset 影子不占用空间

1. 原来盒子没有影子，当我们鼠标经过有影子

Box：hover

## 7、文字阴影

Text-shadow：属性



# 四、网页浮动

浮动产生是为了做文字环绕效果

传统网页布局的三种方式：普通流（标准流/文档流）、浮动、定位；

所谓标准流：就是标签按照规定默认方式排列

网页布局的第一准则：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

**1）Float:属性值**（none left right）



**2）浮动的特性（重难点）**

1. 浮动元素会脱离标准（脱标） \*浮动的盒子不在保留原来的位置
2. 浮动的元素会一行内显示并且元素**顶部对齐 \***如果宽度装不下，多出来的盒子另起一行
3. 浮动的元素会具有行内块元素的特性 \*行内元素块级元素都可添加浮动

\*如果行内元素有了浮动，就不需要转换为行内块元素就可以直接给高度和宽度

如果块级盒子没有设置宽度，默认宽度和父级一样宽，添加浮动后，根据内容大小决定

行内元素同理

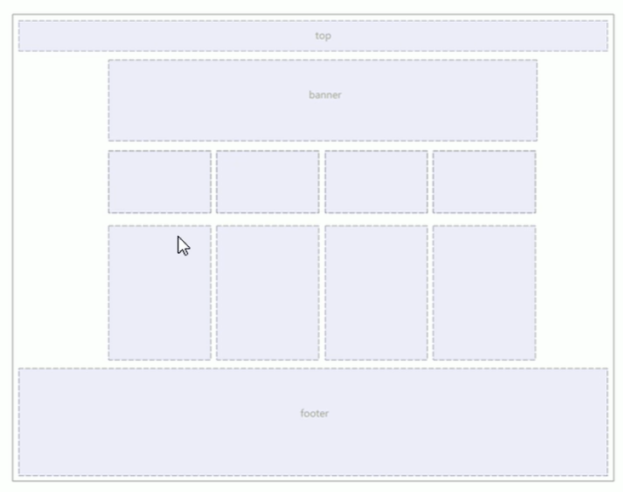
浮动只会影响后面的盒子位置，不会影响前面的盒子

浮动的元素不会压住下面标准流的文字和图片，但会压住盒子

1. **浮动元素经常和标准流父级搭配使用**

先用标准流的父元素排列上下位置，之后内部子元素采取浮动排序左右位置，符合网页布局第一准则

1. **常见的网页布局**



4.1）浮动和标准流的父盒子搭配

4.2）一个元素浮动，理论上其余的兄弟元素也要浮动

\*\*浮动的盒子只会影响浮动盒子后面的标准流，不会影响前面的标准流

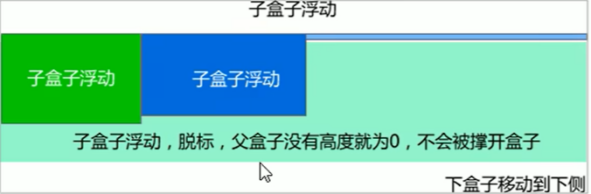
## 2、清除浮动

**本质清除浮动元素造成的影响**

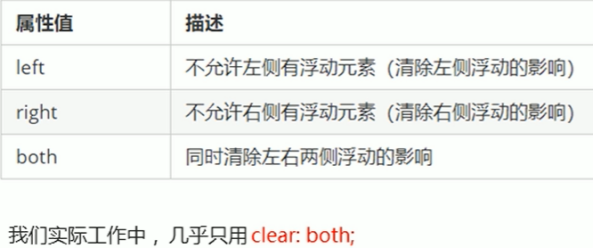
**如果父盒子本身有高度则不需要清除浮动**

为什么清除浮动？

由于父盒子很多情况下，不方便给高度，但是盒子浮动又不占有位置，最后父盒子的高度为0，就会影响下面的标准流盒子



清除浮动语法：clear：属性



5.1）清除浮动方法

1、额外标签法也称隔墙法，W3C推荐的做法

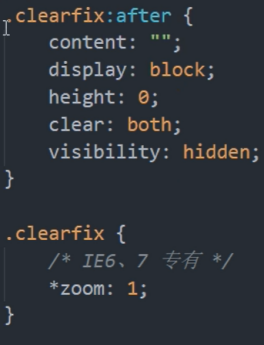
在最后一个标签后面在添加一个标签（块级元素），为其加上css属性clear：both

1. 父级添加overflow属性

属性值为hidden、auto、scroll

缺点：无法显示溢出部分

1. 父级添加after伪元素



代表网站：百度、网易

1. 父级添加双伪元素

.clearfix:before,.clearfix:after {

    content:"";

    display:table;

  }

  .clearfix:after {

    clear:both;

  }

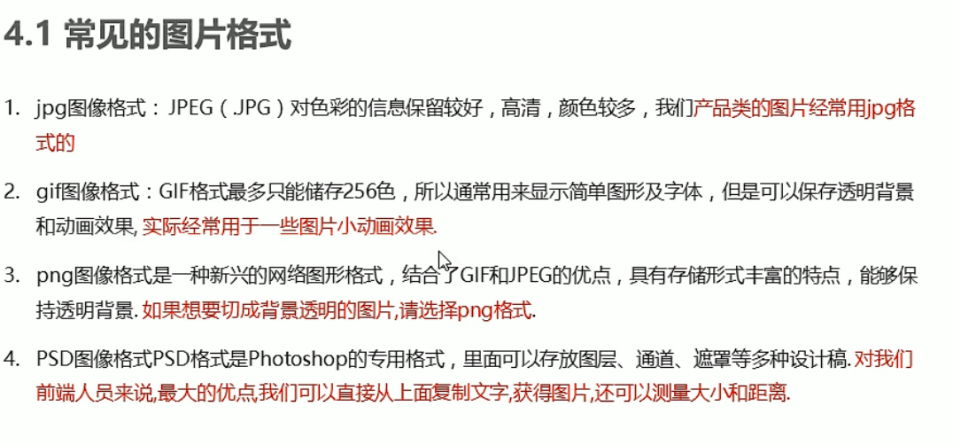
  .clearfix {

     \*zoom:1;

  }

## 3、Ps切图

**1、常见的图片格式**



2、图片的切图

Ctrl+e合并图层

利用ps切图工具->文件->导出->存储为web所用格式->所选格式

Png可以存储透明背景 jpeg不可以

# 五、定位

## 1、为什么要学习定位

可以让某个元素可以自由的在一个内移动位置，并且压住其他盒子

## 2、定位的组成

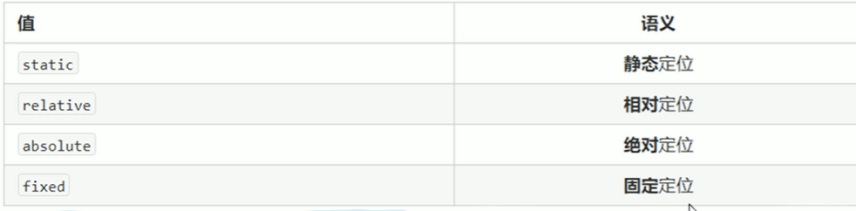
定位=定位模式+边偏移

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

2.1）定位模式

通过css的position的属性值来设置：



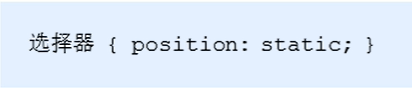
2.2）边偏移

边偏移就是定位的盒子移动到最终位置。有top、bottom、left和right四个属性



## 2、静态定位static

默认定位方式，无定位的意思

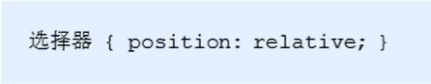


静态定位按照标准流特征摆放位置，他没有边偏移

静态定位很少使用

## 3、相对定位relative（重要）

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



相对定位特点

1. 他是相对于自己原来的位置来移动的
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他（不脱标）

## 4、绝对定位absolute（重要）

绝对定位是元素在移动位置的时候，是相对于它的祖先元素来说的

绝对定位特点

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. 如果祖父元素有定位（相对、绝对、固定定位），则以最近一级的有定位祖父元素为参考点移动位置

3、绝对定位不在占有原先的位置（在浮动之上）

## 5、子绝父相

意思：子级使用绝对定位，父级则需要相对定位

子集绝对定位，不会占有位置可以放到父盒子里面的任何一个地方，不会影响其他兄弟盒

父盒子需要加定位限制盒子在父盒子内显示

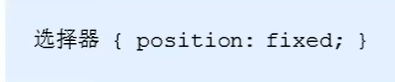
总结：因为父级需要占有位置，因此是相对定位，子盒子不需要占有位置，则是绝对定位

轮播图

## 6、固定定位fixed（重要）

固定定位是元素固定于阅览器可视区固定位置

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



固定定位的特点：

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

不随着滚动条的滚动而滚动

2、固定定位不占有原来位置

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**

1. 固定定位小技巧：固定在版心右侧位置

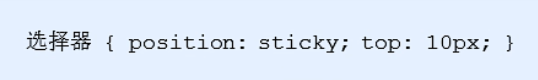
先让固定的盒子left：50%走到阅览器可视区的一半

在让固定的盒子margin-left：50%像素，版心宽度的一半距离

## 7、粘性定位sticky（了解）

粘性定位可以被认为是相对定位和固定定位的混合

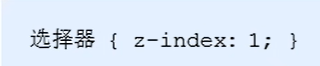
使用场景：浏览器右侧元素，一开始没有固定 随着滚动条滚动开始固定



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（相对定位的特点）
3. 必须添加边偏移的top、left、right、bottom其中一个才有效

## 定位的叠放顺序



数值可以是整数复数和0，默认auto，数值越大，盒子越靠前

不能加单位

只有加定位的盒子才有z-index属性

定位拓展

1. **绝对定位盒子居中**

加了绝对定位的盒子不能通过 margin：0auto 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。

left： 50% ：让盒子的左侧移动到父级元素的水平中心位置。

margin—left：—100px； ：让盒子向左移动自身宽度的一半。

**2、绝对定位和固定定位也和浮动类似**

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

块级元素添加绝对或者固定定位，如果不给宽度或者高度。默认大小是内容大小

1. **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（包括文字和图片）**

## 9、元素的显示与隐藏

类似网站广告，当我们点关闭就不见了，但是我们重新刷新页面，会重新出现

本质：让一个元素在页面中隐藏或者显示出来

1. **display显示隐藏**

Display属性用于设置一个元素应如何显示

Display：none； 隐藏对象 隐藏元素之后不在占有原来位置

Display：block； 除了转换为块级元素之外，同时还有显示元素的意思

后面应用广泛，搭配js可以做很多的网页特效

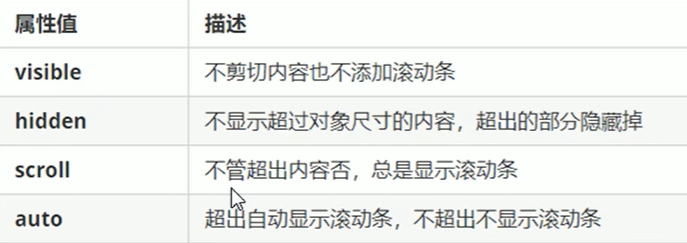
1. **visibility可见性**

Visibility：inherit继承上一个父对象的可见性（默认）/visible元素可视/hidden元素隐藏/

visibility隐藏元素之后，继续占有原来的位置

1. **overflow溢出隐藏**

Overflow：属性值



**但如果有定位的盒子，请慎用overflow：hidden因为他会隐藏多余部分**

# css高级

## 精灵图

1. **为什么要学习精灵图**

一个网页中往往会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。

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 sqrits Css 雪碧）

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

1. **精灵图（sprites）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、主要借助于背景位置来实现：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。

1. **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

**精灵图缺点：**

1. 图片文件比较大
2. 图片本身放大和缩小会失真
3. 一但图片制作完毕想要更换非常复杂

**字体图标 Iconfont**

字体图标展示的是图标，本质是属于字体

**字体图标优点：**

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：如果遇到遇到一些结构样式比较简单的小图标，就用字体图标

## 2、Iconfont的引入

1、选择字体并下载

2、将下载文件中的 fonts 文件夹复制到项目根目录下

3、字体声明，将 style.css 文件中的开头的字体声明代码赋值到 html 中

4、给 span 声明字体：

span {

    font-family: 'icomoon';

}

1. 打开 demo.html，复制页面中的方框图标到 html 代码中即可

**字体图标的追加**

把压缩包里的selection.Json从新上传，从新下载压缩包

## 3、css三角制作.

Box {

width: 0;

height: 0;

border: 10px（10px表示三角形的高） solid transparent;

border-left-color: black;

/\* 照顾兼容性 \*/

line-height: 0;

font-size: 0;}

}



## 4、css用户界面样式

**什么是界面样式？**

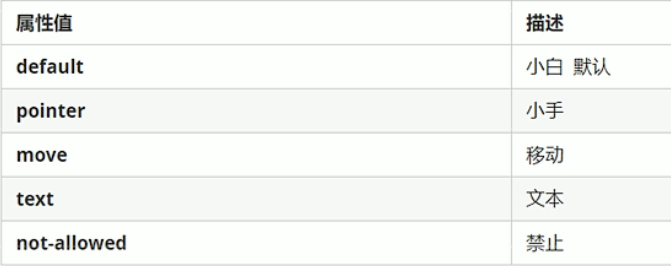
界面样式，就是更改一些用户操作样式，以便提高更好的用户体验

1. **更改用户的鼠标样式cursor**

**Li {**

**Cursor：pointer**

**}**



1. **表单（input）轮廓线outline**

给表单添加**outline：0；**

或者**outline：none；** 就可以去掉默认的蓝色边框

1. **防止拖拽文本区域resize**

**Textarea {**

**Resize：none**

**}**

## vertical-align属性应用

1）Css的vertical-align属性使用场景：经常用于设置图片或者表单（行内块元素）的文字垂直对齐

只针对行内元素或者行内块元素有效





**2）解决图片底部默认空白间隙问题**

原因是行内块元素会和文字的基线对齐 所以有空白间隙

2.1）给图片添加vertical-align：middle|top|bottom等 （提倡使用）

2.2）把图片转换为块级元素：display：block

# 溢出文字省略号显示

## 单行文本溢出显示省略号

1. 先强制一行内显示文本

white-space: nowrap;

1. 超出的部分隐藏

overflow: hidden;

1. 文字用省略号替代超出的部分

text-overflow: ellipsis;

## 多行文本溢出显示省略号

有兼容性问题，适合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: 3;

/\* 设置活检索伸缩盒对象的子元素的排列 \*/

-webkit-box-orient: vertical;

# 常见布局技巧

## margin负值运用

1）解决并排盒子之间的边框宽度加倍问题。 原理：让每个盒子压住前面的盒子，边框叠加

margin-left：-1px；

1. 鼠标经过时边框变颜色

如果盒子没有定位则鼠标经过添加相对定位即可

position: relative;

如果盒子有定位，则利用z-index提高层级

z-index: 1;

3）文字围绕浮动元素

原理：浮动元素不会遮住文字

给元素加上浮动即可

1. 行内块巧妙运用

给父盒子添加text-align：center那么父盒子里面所有的行内元素和行内块元素水平居中

5）css三角强化



原理



/\* 把上边框宽度调大 \*/

border-top: 100px solid transparent;

border-right: 50px solid pink;

/\* 左边和下边的边框宽度设置为0 \*/

border-bottom: 0px solid blue;

border-left: 0px solid green;

## 2、css初始化

不同浏览器对标签的默认值是不同的，为了消除不能浏览器对html文本呈现的差异，照顾浏览器的兼容，我们都需要对css初始化

简单理解：css初始化是指重设浏览器的样式（也称CSS rest）

**没个网页都必须进行CSS初始化**

**Unicode编码字体**

可有效避免浏览器解释css代码出现乱码问题

黑体:\9ED1\4F53

宋体:\5B8B\4F53

微软雅黑:\5FAE\8F6F\96C5\9ED1

# Html5和css提高

## html5新增特性

### 新增语义化标签

<header>：头部标签

<nav>：导航标签

<article>：内容标签

<section>：定义文档某个区域

<asider>：侧边栏标签

<footer>：尾部标签

### 新增的多媒体标签

2.1）视频：<video>

语法：

<video src="文件地址" controls:"controls"></video>



src=“url” 视频地址

autoplay="autoplay" 视频就绪自动播放（谷歌浏览器需要添加muted来解决自动播放问题）

controls="controls" 显示播放控件

muted=muted 静音播放

loop=“loop” 设置循环播放

preload="auto/none" 是否预加载

poster=“url” 加载等待图片 封面图片

width 设置宽度

height 设置高度

<video src=“文件地址” autoplay="autoplay" controls="controls" muted="muted" loop=“loop” ></video>

2.2）音频：<audio>

语法：

<audio src="文件地址" controls:"controls"></audio>

controls：显示播放控件

autoplay：（谷歌禁用）

loop=loop 设置循环播放

**谷歌阅览器把音频和视频自动播放禁止了**

### html5新增的input类型

3.1）

type="email"

type="url"

type="date"

type="time"

type="month"

type="week"

type="number" 限制数字类型

type="tel" 手机号码

type="color" 生成一个颜色选择表单

type="search" 搜索框

**我们验证时候必须添加form表单域**

### html5新增的表单属性

| **属性** | **值** | **说明** |
| --- | --- | --- |
| required | required | 表单拥有该属性表示其内容不能为空，必填 |
| **placeholder** | **提示文本** | **表单的提示信息** |
| autofocus | autofocus | 自动聚焦属性，页面加载完成自动聚焦到指定表单 |
| autocomplete | off/on | 当用户在字段开始键入时，浏览器基于之前键入过的值，应该显示出在字段中填写的选项（记住之前提交的）  默认已经打开,如autocomplete="on",关闭autocomplete ="off" 需要放在表单内，同时加上name属性，同时成功提交 |
| **multiple** | **multiple** | **可以多选文件** |

<form action="">

<input type="email" required=""required placeholder="placeholder">

</form>

**必须添加form表单域**

# css3的新特性

## 1、属性选择器

可以根据元素特定属性来选择元素。这样就可以不用借助于类，或者id选择器

1、E[att]|选择具有att属性的E元素

2、**E[att:val|选择具有att属性且属性值等于val的E元素（重点）**

input[type=text]

<input " type="text" value="123213">

3、E[att^=val]|匹配具有att属性且值以val开头的E元素

div[class^=icon]

选择是div并且具有class属性 并且属性值是icon**开头**的这些属性

4、E[att$=val]|匹配具有att属性且值以val结尾的E元素

div[class$=icon]

选择是div并且具有class属性 并且属性值是icon**结尾**的这些属性

5、E[att\*=val]|匹配具有att属性且值中含有val的E元素

div[class\*=icon]

选择是div并且具有class属性 属性值只要有icon就选择出来

**类选择器、伪类选择器、属性选择器，权重为10**

## 2、结构伪类选择器

权重为10

结构伪类选择器主要根据文档结构来选择器元素。常用于根据父级选择器里面的子元素

| 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 |

ul li:first-child {

background-color: pink;

}

ul li:last-child {

background-color: pink;

}

ul li:nth-child(5) {

background-color: skyblue;

}

**E: nth-child(n) 匹配父元素中的第n个子元素E**

**n可以是整数、关键字（even偶数/odd奇数）、公式（n/2n/2n+1）**

**(n+2)从第二个盒子开始、(-2+n)前两个盒子**



div: nth-child 会把所有的盒子都排列序号 执行的时候首先看 :nth-child(1) 之后回去看 前面 div

div: nth-of-type 会把指定元素的盒子排列序号 执行的时候首先看 div指定的元素 之后回去看 :nth-of-type(1) 第几个孩子

区别：

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

## 3、伪元素选择器（重点）

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。

|  |  |
| --- | --- |
| ::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 |

before 和 after 创建一个元素，但是**属于行内元素（不能设置宽高）**

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（div）**一样，权重为 1

## 4、css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：

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

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）

## 5、css3其他特性（了解）

### 1）Css3过渡

**Transition:**要过渡的属性 花费时间 运动曲线 何时开始

//有过想要更改多个属性可以利用“，”

* 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。
* 花费时间：单位是秒（必须写单位）比如 0.5s
* 运动曲线：默认是ease（逐渐慢下来）linear（匀速） ease-in（减速 （可以省略）
* 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Os （可以省略）

### 2）如何把图片变模糊

Css3滤镜filter

filter: 函数(); filter: blur(5px); blur模糊处理 数值越大模糊越大

### 3）css3的calc函数

Calc（）此函数让你在声明css属性值时执行一些计算

width: calc(100%-80px); 永远比父盒子宽度少80px

//可以不用考虑父盒子的宽度

括号里面+ - \* /

# 10、css3 2D转换

transform是CSS3中具有颠覆性特性之一 可以实现元素的位移、旋转、缩放

移动：translate

旋转：rotate

缩放：scale

## 1、移动translate

**语法**

transform: translate(x, y);

transform: translateX(x);

transform: translateY(y);

参数 x, y 可以是百分数，为盒子自身的宽度或高度。

**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 trainslate:(50%，50%)**，对行内标签没有效果**

2、

## 2、[让盒子实现水平和垂直居中](https://docs.mphy.top/" \l "/CSS/ch07?id=_33-%e8%ae%a9%e7%9b%92%e5%ad%90%e5%ae%9e%e7%8e%b0%e6%b0%b4%e5%b9%b3%e5%92%8c%e5%9e%82%e7%9b%b4%e5%b1%85%e4%b8%ad)

/\*子绝父相\*/

position: absolute;

top: 50%;

left: 50%;

width: 200px;

eight: 200px;

transform: translate(-50%,-50%);

background-color: black;

**//traslate对于行内元素是无效的**

## 3、旋转rotate

语法：

transform: rotate(45deg);

**值为正数则顺时针旋转，为负数则逆时针旋转。**

**以中心为旋转中心**

## 4、设置旋转中心

我们可以通过设置 transform-origin：x y 设置元素转换的中心点。

* 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）

## 5、2D转换之缩放

transform: scale(x, y);

x, y 不跟单位的话，是指缩放的倍数。

所有元素都放大(影子)

* 参数大于 1 则放大，小于 1 则缩小。
* 可以配合 transform-origin 使用，改变缩放中心。
* **scale 的优势：不占空间(位置)**

## 6、2D转换综合写法

* 同时多个属性： transform: translate(), rotate() scale()
* 其顺序会影转换的效果。（先旋转会改变坐标轴方向）
* **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.**

旋转木马先旋转在移动 要不就成\*

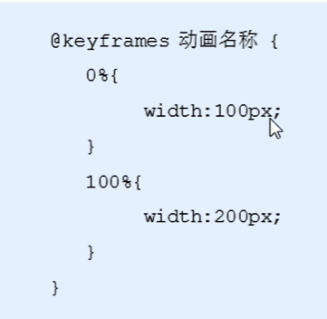
# 11、css3动画

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

## 动画的基本使用

1定义动画 (动画序列 %α)

2使用动画



    /\* 1定义动画 \*/

    @keyframes move {

        /\* 开始状态 \*/

        0% {

            transform: translateX(0px);

        }

        /\* 结束状态 \*/

        100% {

            transform: translateX(1000px);

        }

    }

    div {

        width: 200px;

        height: 200px;

        background-color: black;

        /\* 调用动画 \*/

        animation-name: move;

        /\* 持续时间 \*/

        animation-duration: 2s ;

Anmination:调用动画 持续时间

    }

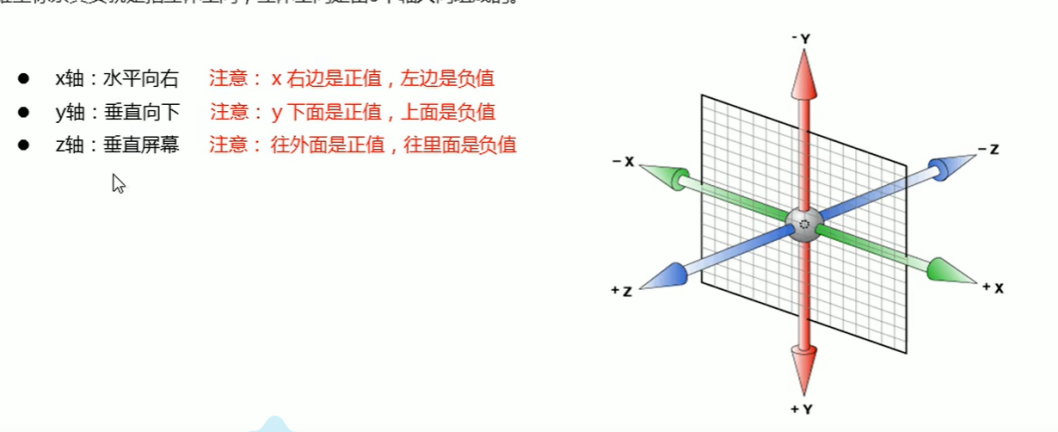
动画序列

* 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%。

## 2、动画常用属性

| 属性 | 描述 |
| --- | --- |
| keyframes | 规定动画。 |
| animation | 所有动画属性的简写属性,除了animation-play-state属性。 |
| animation-name | 规定@keyframes动画的名称。(必须的) |
| animation-duration | 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) |
| animation-timing-function | 规定动画的速度曲线，默认是“ease从慢到快” steps()分几步完成动画. |
| animation-delay | 规定动画何时开始，默认是0. |
| animation-iteration-count | 规定动画被播放的次数，默认是1，还有infinite（无限） |
| animation-direction | 规定动画是否在下一周期逆向播放，默认是 "normal",alternate逆播放 |
| animation-play-state | 规定动画是否正在运行或暂停。默认是"running",还有"paused". |
| animation-fill-mode | 规定动画结束后状态, 保持forwards 回到起始backwards |

# 12、3D转换



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); 仅仅在x轴上移动

  transform:translateY(100px) ; 仅仅在y轴上移动

  transform: translateZ(100px); 仅仅在z轴上移动（translateZ 一般用px单位）

  transform: translate3d(x,y,z); 其中x，y，z指在要移动的轴的方向距离(注意：xyz不能省略如果没有就写0)

## 透视perspective

**透视写在被观察元素的父盒子上面**

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

## 2、3D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轴或自定义轴旋转

语法

 transform: rotateX(45deg);

transform: rotateY(45deg);

transform: rotateZ(45deg);

transform: rotate3d(x,y,z,deg); 沿着自定义轴旋转(了解)

**X，y，z也不能省略**

用左手法则分辨旋转方向

## 3、3D呈现transform-style

* 控制子元素是否开启三维的立体环境
* 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**

# 移动端基础

## 视口

**视口(viewport)**就是浏览器显示页面内容的屏幕区域。视口可以分为布局视口、视觉视口、理想视口（常用）

### 1、布局视口**layout viewport（不适合）**

* 一般移动设备的默认浏览器都默认设置了一个布局视口，用于解决早期的PC端页面在手机上显示的问题
* IOS，Android基本都将这个视口分辨率设置为980px。所以PC上的网页大都能在手机上实现，只不过元素看上去很小，一般默认可以通过手动缩放网页

### 视觉视口**visual viewport（不适合）**

### 理想视口ideal viewport

* 理想视口，对设备来讲，是最理想的视口尺寸
* 需要手动添写meta视口标签通知阅览器操作
* Meta视口标签的主要目的：布局视口的宽度应该与理想视口的宽度一致。简单理解就是设备有多宽，我们布局视口就多宽

### meta视口标签

<meta name="viewport" content="width=device-width, initial-scale=1.0,maximum-scale=1.0,minimum-scale=1.0,user-scalable=no">



* 标准的viewport设置
* 视口宽度和设备保持一致
* 视口的默认缩放比例1.0
* 不允许用户自行缩放
* 最大允许的缩放比例1.0
* 最小允许的缩放比例1.0

## 二倍图

### 物理像素&物理像素比

* 物理像素点指的是**屏幕显示的最小颗粒**，是物理真实存在的
* 我们开发时候的1px**不是**一定等于1个物理像素的
* **PC端页面，一个px等于1一个物理像素的**，但是移动端就不尽相同
* 一个px的能显示的物理像素点的个数，**称为物理像素比或屏幕像素比**

### 多倍图

解决在移动端图片不清楚问题

我们采取使用的图片比我们实际需要的大小 2倍的方法

之后再手动缩小为原来的百分之五十

## 移动端开发选择

1. **单独制作移动端页面（主流）**

京东商城手机版

淘宝触屏版

苏宁易购

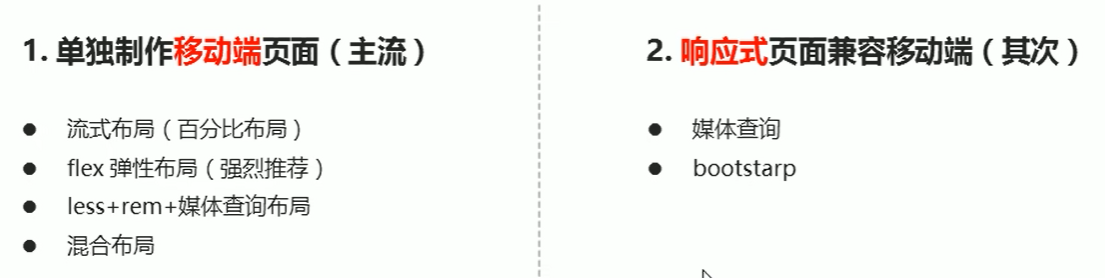
通常情况下，网址域名前面加m(mobile)可以打开移动端。通常判断设备，是否 是移动设备打开，则跳到移动端页面

1. **响应式兼容PC移动端**

通过判断屏幕宽度来改变样式，以适应不同终端

缺点：制作麻烦，需要花很大精力去调兼容性问题

## 4、移动端常用布局



# 流式布局(百分比布局)

* 流式布局，就是百分比布局，也称非固定像素布局
* 通过盒子的宽度设置百分比来根据屏幕的宽度来进行伸缩，不受固定像素的限制，内容向两侧填充
* 流式布局方式是移动web开发使用的比较常见的方式

新增加两个属性：

Max-width：最大宽度

Min-width：最小宽度

## 1、布局原理

Flex（flexbile Box）弹性布局

* 当我们为父盒子设置flex布局以后，子元素的float、clear和vertical-align就会失效
* 可以让我们的盒子并排显示，并且也可以让行内元素转换为行内块元素
* 采用Flex布局元素，称为Flex容器，简称容器。他的所有子元素自动成为容器成员，简称Flex项目，简称项目

**语法：display：flex**

## 二倍精灵图做法

* 在firework里面把精灵图等比例缩放为原来的一半
* 之后根据大小测量坐标
* 注意代码里面background—size也要写：精灵图原来宽度的一半

# 移动WEB开发的flex布局

## flex布局父项常见属性

* Flex-direction：设置主轴的方向
* Justify-content：设 置主轴上的子元素排列方式
* Flex-wrap：设置子元素是否换行
* Align-content：设置侧轴上的子元素的排列方式（多行）
* Align-items：设置侧轴上的子元素排列方式（单行）
* Flex-flow：复合属性，相当于同时设置了flex-direction和flex-wrap

### 1、Flex-direction：设置主轴的方向屏幕截图 2022-10-06 112409

Flex-direction属性决定主轴方向

### 2、Justify-content：设置主轴上子元素排列方式



### Flex-wrap设置子元素是否换行

默认情况下，子元素都是不换行的，如果装不开会缩小子元素宽度，以便放到父元素一行上



### Align-items：设置侧轴上的子元素排列方式（单行）



### Align-content：设置侧轴上的子元素的排列方式（多行）

在单行无效果



## 3、flex布局子项常见属性

### 1、flex属性

Flex属性定义子项目分配剩余空间，用flex来表示占多少份数

**记得加语法：display：flex**

.item {

    flex:<number/百分比(相对于父级)>; default 0

}

### 2、align-self控制 子项自己在侧轴上的排列方式

Align-self属性允许单个项目有与其他项目不一样的对齐方式，可覆盖align-item属性

### order属性定义项目的排序顺序

数值越小，排列越靠前，**默认为0**

注意与z-index不一样

# rem适配布局

苏宁

可以让我们屏幕大小发生变化时元素的高和宽等比缩放

## rem基础

Rem（root em）单位

Em相对于**父元素**的**字体大小**来设置

Rem是相对于**html**元素的**字体大小**来设置

Rem的优点就是可以通过修改html里面的文字大小来改变页面中元素的大小来实现整体控制

## 媒体查询

媒体查询（media query）是css3语法

* 使用媒体查询@media查询，可以针对不同的媒体类型定义不同的样式
* **@media可以针对不同的屏幕设置不同的样式**
* 当时重置浏览器大小过程中，页面也会根据浏览器的宽度和高度重新渲染页面

## 3、语法：

@media mediatype and|not|only (media feature) {

    Css-Code

}

用@mediatype媒体类型

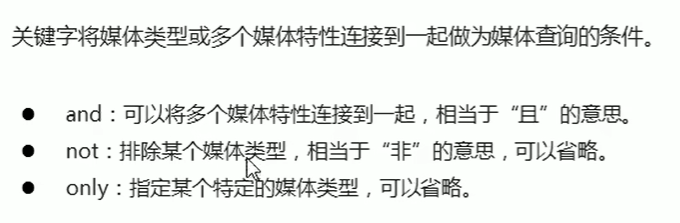
关键字and|not|only

Media feature媒体特性 必须有小括号包含

### mediatype查询类型



### 关键字



### 媒体特性



Max-width：小于等于

Min-width：大于等于

## 媒体查询+rem实现动态大小变化

<style>

    @media screen and (min-width:320px) {

        html {

            font-size: 50px;

        }

    }

    @media screen and (min-width:640px) {

        html {

            font-size: 100px;

        }

    }

    .top {

        height: 1rem;

        font-size: .5rem;

        background-color: green;

        color: #fff;

        text-align: center;

        line-height: 1rem;

    }

</style>

<body>

    <div class="top">购物车</div>

</body>

## 引入资源

当样式比较多的时候，**我们可以针对不同的媒体使用不同stylesheets（样式表）**

原理，就是直接在link中判断设备的尺寸，然后引用不同的css文件

### 语法

 <link rel="stylesheet" media="mediatype and|not|only (media feature)" href="mystylesheet.css" >

媒体查询从小到大

## less基础

Less是一门css预处理语言，它扩展了css的动态特性

### less变量

变量是指没有固定的值，可以改变的。因为我们css中的一些颜色和数值等经常使用

**语法：**

@变量名：值；

@color:pink;

div {

    color: @color;

}

方便后期维护

命名规范：

* 必须有@为前缀
* 不能包含特殊字符
* 不能以数字开头
* 大小写敏感

### less编译

利用插件编译生成css，之后link引入就可以使用样式

### less嵌套

.类名 {

.类名{

}

}

Less嵌套，子元素的样式直接写到父元素里面

如果有伪类选择器、交集选择器、伪元素选择器，我们内层选择器前面需要加**&**

### less运算

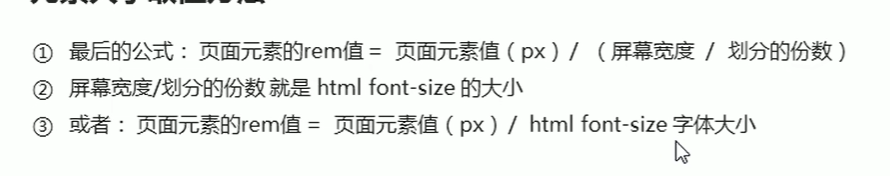
在less中提供加减乘除

**注意\***

1. 我们运算符的左右两侧必须敲一个空格隔开
2. 两个数参与运算 如果只有一个数有单位，则最后的结果以这个单位为准
3. 如果两个数参与运算，都有单位，以第一个单位为基准

### 5、rem适配方案

**方案一、动态设置html标签font-size大小**



**方案二、简介高效的rem适配方案flexible.js(推荐)**

Js文件提取的宽度是**整个窗口的宽度**，不是body的宽度

它的原理是把当前设备划分为10等份，但是不同设备下，比例还是一致的

我们需要做的就是，确定好我们当前设备的html文字大小

**cssrem插件可以快速计算rem值，但需要每次修改sizeroot（当前设备文字大小）**