用来方便查找

-----------------------------------------------------------------------------------------------------------------

HTML

表单的目的是为了收集用户信息

一个完整的表单通常由表单域，表单控件（也叫表单元素），提示信息三个部分构成

表单域是一个包含表单元素的区域

<form></form>标签用于定义表单域，以实现用户信息的收集和传递

form会把它范围内的表单元素信息提交给服务器

<form action="要提交到哪里" method="提交方式"name="表单域名字">

</form>

表单元素：在表单域中可以定义各种表单元素，这些表单元素就是允许用户在表单中输入或者选择的内容控件

input输入表单元素

select下拉表单元素

textarea文本域元素

input标签是一个单标签，是输入的意思，在表单元素中用来收集用户信息

<input type="属性值">

在input标签中，包含一个type属性，这个type是用来根据不同的type属性来选择各种形式的控件（可以是文本字段，复选框，按钮）

text文本

button按钮

checkbox复选框

file 输入字段和"浏览"按钮，供文件上传

hidden 隐藏输入字段

image图片形式的提交按钮

password密码字段

radio单选按钮

reset重置按钮会清除表单中的所有数据

submit提交按钮会把表单数据发送到服务器

input除了type属性外还有

name定义input元素名，单选框和复选框要有相同的name值；

value规定input元素的值，在text文本中可以看到直接效果，其他的都是传给后台 ,

checked规定input元素首次加载时应当被选中，主要针对单选按钮和复选按钮，默认选择 checked="checked"

maxlength规定输入字段中字符的最大长度 maxlength="长度"

<label></label>标签为input元素定义标注

label标签用于绑定一个表单元素，当点击label标签内的文本时，浏览器就会自动将焦点（光标）转到或者选择对应的表单元素上，用来增加用户体验

语法：

<label for="sex">要点击的内容</label>

<input type="radio"name=""id="sex">

label标签的for属性要和相关元素的id属性相同

<select>表单元素：

有多个选项选择是，想要节省网页空间，就可以使用select标签定义下拉列表

select语法：

<select>

<option>选项1</option>

<option>选项2</option>

<option>选项3</option>

</select>

在option中定义selected="selected"时，当前项默认选中项

<textarea>表单元素：

当用户输入内容较多的情况下，就不能使用文本框表单了，要用<textarea>

textarea标签是用于多行文本输入的控件

使用多行文本输入控件，可以输入更多的文字，该控件常见于留言板，评论

textarea语法：

<textarea name="#" id="" cols="30" rows="10"> //框的大小cols是框的列 rows是框的行

</textarea>

<!-- action 要提交的到哪里--> <!--method用什么方式传送-->

<form action="form.js" method="POST"name="名字">

<div>

<label>你的名字</label><!--标注-->

<!--type 是选择框的形式--> <!--name是用什么名字传输--> <!--placeholder是框里的字-->

<input type="text" name="name"placeholder="输入你的名字">

<label>你的性别</label>

<input type="radio" name="sex">男

<input type="radio" name="sex">女

<input type="email" name="email" placeholder="你的邮箱">

</div>

<input type="submit" name="submit" value="按钮">

</form>

列表有三种无序列表，有序列表，自定义列表

列表是用来布局的

ul标签标示无序列表，一般以项目符号呈现列表项，而列表项使用li标签定义

<ul>

<li></li>

</ul>

无序列表的个列表项之间没有顺序级别之分，是并列的

ul中只能嵌套li，直接在ul标签中输入其他标签或其他文字的做法是不被允许的

li之间相当于一个容器，可以容纳所有元素

ol标签表示有序列表，一般以项目符号呈现列表项，而列表项使用li标签定义

<ol>

<li></li>

</ol>

ol中只能嵌套li，直接在ol标签中输入其他标签或其他文字的做法是不被允许的

li之间相当于一个容器，可以容纳所有元素

自定义列表

dl标签用于定义描述列表，该标签会与dt标签（定义项目或名字）和dd标签（描述项目或名字）一起使用

<dl>

<dt></dt>

<dd><dd>

<dd><dd>

</dl>

dl中只能嵌套dt和dd，直接在ol标签中输入其他标签或其他文字的做法是不被允许的

dt和dd之间相当于一个容器，可以容纳所有元素

写在其他标签外面，不要写在里面

<a href="https://www.baidu.com">显示点击的名字</a>

在新的网页打开：

<a href="https://www.baidu.com"target="\_blank">显示点击的名字</a>

href是连接的路径

target是打开方式 默认为\_sel是在当前页面打开，\_blank是在新页面打开

内部连接 <a href="内部本地路径"></a>

空链接 如果当时没有确定链接的目标是什么时， <a href="#"></a>

下载链接 如果href里面地址是一个文件或者压缩包，点击就会下载这个文件

网页元素的链接 在网页中的各种网页元素，如文本，图像，表格，音频，视频等都可以添加插连接

以图片为例<a href=""><img src=""></img></a>

锚点链接 点击链接就可以快速定位到页面中的某个位置

<a href="#跳转名字"></a>

跳转目标位置写id="名字"好像只能在其他标签里用 例如<h1 id="跳转名字"></h1>

要写在别的标签外面，不要写在里面

<img src="" alt="" title=""> src是图片路径,可以是链接也可以是文件路径，alt是图片显示不出的时候的，提示文字,title是提示文本，鼠标放到这个位置，提示文字

width="" 设置宽度

height="" 设置高度

border="" 边框粗细

图像标签可以拥有多个属性，必须写在标签名后面

属性之间不分先后顺序， 标签名和属性，属性和属性之间均以空格分开

属性采取的格式 属性="属性值"

同一级路径 可以直接写文件名

下一级路径 / 进入下一级目录

上一级路径 ../ 返回上一级目录

绝对路径：

完整的文件夹所在位置

完整的网络地址

文字加粗 <string></string>或<b></b>

斜体<em></em>或<i></i>

删除线<del></del>或<s></s>

下划线<ins><ins>或<u></u>

div和span是没有语义的，就是一个盒子，用来装内容的

<div></div>

<span></span>

注释 <!-- -->

特殊字符

空格 &nbsp

< &lt

> &gt

其他的可以网上找

表格标签是实际开发中非常常用的标签

表格的主要作用 用来显示，展示数据的，因为可以让数据显示的非常规整

表格的基本语法

<table> //定义表格

<thead> //表格的头部

<tr>//行

<th>名字</th>//th是表头的单元格

<th>年龄</th>

<th>身高</th>

<th>体重</th>

</tr>

</thead>

<tbody>//表格的身体

<tr>//行

<td>Xman </td>//td是表格里的单元

<td>24</td>

<td>178</td>

<td>65kg</td>

</tr>

</tbody>

</table>

表格的属性，修改外观形态的，！！！这些属性要写到表格标签table里面去！！！

属性名 属性值 效果

align left，center，right 对齐方式，往哪边对齐

border 1或"" 规定表格单元是否拥有边框，默认是""表示没有

cellpadding 像素值 规定单元边沿与其内容之间的空白，默认像素为1

cellspacing 像素值 规定单元格之间的空白，默认像素为2

width 像素值或百分比 规定表格的宽度

跨行合并：rowspan="合并单元格的个数" 以要合并的最上侧单元格为目标单元格，在里面写合并代码

跨列合并：colspan="合并单元格的个数" 以要合并的最左侧单元格为目标单元格，在里面写合并代码

合并单元格三步曲

1 先确定要跨行还是跨列

2找到目标单元格，写上合并方式=“合并数量”

3删除被合并的单元格

HTML5 的新增特性主要是针对于以前的不足，增加了一些新的标签、新的表单和新的表单属性等。

这些新特性都有兼容性问题，基本是 IE9+ 以上版本的浏览器才支持，如果不考虑兼容性问题，可以大量使用这

些新特性。

新特性增加了很多，但是我们专注于开发常用的新特性

HTML5 新增的语义化标签

以前布局，我们基本用 div 来做。div 对于搜索引擎来说，是没有语义的。

HTML5 新增的语义化标签

<header>：头部标签

 <nav>：导航标签

 <article>：内容标签

 <section>：定义文档某个区域

 <aside>：侧边栏标签

 <footer>：尾部标签

这种语义化标准主要是针对搜索引擎的

 这些新标签页面中可以使用多次

 在 IE9 中，需要把这些元素转换为块级元素

 其实，我们移动端更喜欢使用这些标签

 HTML5 还增加了很多其他标签

1.2 HTML5 新增的多媒体标签

新增的多媒体标签主要包含两个：

1. 视频：<video>

2. 音频：<audio>

使用它们可以很方便的在页面中嵌入音频和视频，而不再去使用 flash 和其他浏览器插件。

HTML5 在不使用插件的情况下，也可以原生的支持视频格式文件的播放，当然，支持的格式是有限的。当前 <video> 元素支持三种视频格式： mp4，webm，ogg 尽量使用 mp4格式

HTML5 在不使用插件的情况下，也可以原生的支持音频格式文件的播放，当然，支持的格式是有限的。

音频：<audio> ：

语法：

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

video的常见属性：autoplay=autoplay //视频自动播放，谷歌浏览器需要添加muted来解决自动播放问题

controls=controls //向用户显示播放控件

width=多少px //设置播放器的宽度

height=多少px //设置播放器的高度

loop=loop //是否循环播放

preload=auto //预先加载视频，如果有了autoplay就忽略这个属性

preload=none //不预先加载视频

src=url（）//视频路径

poster=imgurl //加载等待的画面图片

muted=muted //静音播放

HTML5 在不使用插件的情况下，也可以原生的支持音频格式文件的播放，当然，支持的格式是有限的。

音频<audio>：当前 <audio> 元素支持三种音频格式：mp3，wav，ogg

语法：

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

常见属性： autoplay=autoplay //音频就绪后自动播放

controls=controls //向用户显示控件，比如播放按钮

loop=loop //循环播放

src=url //音频路径

谷歌浏览器把音频和视频自动播放禁止了

多媒体标签总结

音频标签和视频标签使用方式基本一致

 浏览器支持情况不同

 谷歌浏览器把音频和视频自动播放禁止了

 我们可以给视频标签添加 muted 属性来静音播放视频，音频不可以（可以通过JavaScript解决）

 视频标签是重点，我们经常设置自动播放，不使用 controls 控件，循环和设置大小属性

HTML5 新增的 input 类型

type="email" //限制用户输入必须为email类型

type="url" //限制用户输入必须为url类型

type="date" //限制用户输入必须为日期类型

type="time" //限制用户输入必须为时间类型

type="month" //限制用户输入必须为月份类型

type="week" //限制用户输入必须为周类型

type="number" //限制用户输入必须为数字类型

type="tel" //手机号码

type="search" //搜索框

type="color" //生成一个颜色选择表单

重点记住： number tel search 这三个

HTML5 新增的表单属性：

required=required //表单拥有该属性表示其内容不能为空，必填

placeholder="" //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，默认已经打开on 关闭off ，需要放在表单内，同时加上name属性，同时成功提交

multiple=multiple //配合文件文本，可以多选文件提交

可以通过以下设置方式修改placeholder里面的字体颜色：

input::placeholder {

color: ;

}

------------------------------------------------------------------------------------------------------------------------

CSS

<p class="名字">内容</p>

调用

.名字{

color:red;

}

<p#id="名字">内容</p> 用id方式时，需要唯一

CSS3 的现状

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 给我们新增了选择器，可以更加便捷，更加自由的选择目标元素。

1. 属性选择器

2. 结构伪类选择器

3. 伪元素选择器

属性选择器：

属性选择器可以根据元素特定属性的来选择元素。 这样就可以不用借助于类或者id选择器

选择符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

注意：类选择器、属性选择器、伪类选择器，权重为 10。

结构伪类选择器：

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

父元素 E：first-child //匹配父元素中的第一个子元素E 这个先匹配顺序，再对比元素

父元素 E：last-child //匹配父元素中最后一个E元素

父元素 E：nth-child(n) //匹配父元素中的第n个子元素E

父元素 E：first-of-type //指定类型E的第一个 这个先元素，再匹配顺序

父元素 E：last-of-type //指定类型E的最后一个

父元素 E：nth-of-type //指定类型E的第n个

区别：

1. 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

2. 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

nth-child（n） 选择某个父元素的一个或多个特定的子元素（重点）

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 )

2n 偶数

2n+1 奇数

5n 5的倍数

n+5 从第5个开始到最后

-n+5 前五个（到这数上去）

小结

 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。

::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

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。

2.5 CSS3 盒子模型

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

1. 图片变模糊

2. 计算盒子宽度 width: calc 函数

CSS3滤镜filter:filter CSS属性将模糊或颜色偏移等图形效果应用于元素。

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

CSS3 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。

CSS3 还增加了一些 动画 2D 3D 等新特性

CSS3 过渡（重点）：

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 渐渐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。

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）linear匀速 ease逐渐慢下来 ease-in加速 ease-out减速 ease-in-out先加速后减速

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

2D 转换

转换（transform）是CSS3中具有颠覆性的特征之一，可以实现元素的位移、旋转、缩放等效果，转换（transform）你可以简单理解为变形

移动：translate

 旋转：rotate

 缩放：scale

二维坐标系：

2D转换是改变标签在二维平面上的位置和形状的一种技术，先来学习二维坐标系、

2D 转换之移动 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。

语法：

transform: translate(x,y); 或者分开写

transform: translateX(n);

transform: translateY(n);

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

2D 转换之旋转 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

语法：

transform:rotate(度数)

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

2D 转换中心点 transform-origin

我们可以设置元素转换的中心点

语法：

transform-origin: x y;

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）

2D 转换之缩放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

语法：

transform:scale(x,y);

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

 sacle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 转换综合写法：

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 sacle(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

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

制作动画分为两步：

1.先定义动画

2.再使用（调用）动画

1. 用keyframes 定义动画（类似定义类选择器）

@keyframes 动画名称 {

0%{

width:100px;

}

100%{

width:200px;

}

}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. 元素使用动画

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;

动画常用属性：

@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”。

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

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

动画简写属性：

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;

简写属性里面不包含 animation-play-state

 暂停动画：animation-play-state: puased; 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

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）

}

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。

这里我们以 京东CSS初始化代码为例。

Unicode编码字体：

把中文字体的名称用相应的Unicode编码来代替，这样就可以有效的避免浏览器解释CSS代码时候出现乱码

的问题。

比如：

黑体 \9ED1\4F53

宋体 \5B8B\4F53

微软雅黑 \5FAE\8F6F\96C5\9ED1

/\* 把我们所有标签的内外边距清零 \*/

\* {

margin: 0;

padding: 0

}

/\* em 和 i 斜体的文字不倾斜 \*/

em,

i {

font-style: normal

}

/\* 去掉li 的小圆点 \*/

li {

list-style: none

}

img {

/\* border 0 照顾低版本浏览器 如果 图片外面包含了链接会有边框的问题 \*/

border: 0;

/\* 取消图片底侧有空白缝隙的问题 \*/

vertical-align: middle

}

button {

/\* 当我们鼠标经过button 按钮的时候，鼠标变成小手 \*/

cursor: pointer

}

a {

color: #666;

text-decoration: none

}

a:hover {

color: #c81623

}

button,

input {

/\* "\5B8B\4F53" 就是宋体的意思 这样浏览器兼容性比较好 \*/

font-family: Microsoft YaHei, Heiti SC, tahoma, arial, Hiragino Sans GB, "\5B8B\4F53", sans-serif

}

body {

/\* CSS3 抗锯齿形 让文字显示的更加清晰 \*/

-webkit-font-smoothing: antialiased;

background-color: #fff;

font: 12px/1.5 Microsoft YaHei, Heiti SC, tahoma, arial, Hiragino Sans GB, "\5B8B\4F53", sans-serif;

color: #666

}

.hide,

.none {

display: none

}

/\* 清除浮动 \*/

.clearfix:after {

visibility: hidden;

clear: both;

display: block;

content: ".";

height: 0

}

.clearfix {

\*zoom: 1

}

css有三个非常重要的特性：层叠性，继承性，优先级

层叠性：

相同选择器给设置相同的相识，此时一个样式就会覆盖（层叠）另一个冲突的样式层叠性主要解决样式冲突的问题

层叠性原则：

1. 样式冲突，遵循的原则就是就近原则，哪个样式离结构近，就执行哪个样式

2. 样式不冲突，就不会层叠

继承性：

现实中的继承：我们继承了父亲的姓

CSS中的继承：子标签会继承父标签的某些样式，如文本颜色和字号，简单的解释就是：子承父业

恰当的使用继承可以简化代码，降低css样式的复杂性

子元素可以继承父元素的样式（text-,font-,line-这些元素开头的可以继承，以及color属性）

行高的继承性：行高可以写单位，也可以不写单位，不写单位的时候是字体大小的几倍

如果子元素没有设置行高，则会继承父元素的行高

body行高1.5这样的写法最大的优势就是里面的子元素可以根据自己文字大小自动调整行高

优先级：

当同一个元素指定多个选择器时，就会有优先级的产生

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

继承或者\* 选择器的权重是0,0,0,0

元素选择器的权重是0,0,0,1

类选择器，伪类选择器的权重是0,0,1,0

ID选择器的权重是0,1,0,0

行内样式style="" 的权重是1,0,0,0

!important 重要的 的权重是无穷大，写在一句样式的后面;分号前面

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，子元素得到的权重都是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

，权重虽然会叠加，但是永远不会有进位的情况

按照CSS样式书写的位置（或者引入的方式），CSS样式表可以分为三大类

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）

行内样式表：是在元素标签内部的style属性中设定CSS样式，适合用于修改简单样式

例如：<div style="color: red;font-size: 12px">

文本内容

</div>

内部样式表：就是写在html页面内部，是将所有的CSS代码单独放到<style></style>标签里面

外部样式表：实际开发都是外部样式表，适合于样式比较多的情况，核心是：样式单独写到CSS文件中，之后把CSS文件引入到html页面中使用

引入外部样式表分为两步：

1.新建一个后缀名为css的样式文件，把所有css代码都放入此文件中

2.在html页面中，使用<link>标签引入这个文件

<link rel="stylesheet" href="css文件路径"> //rel引入的是什么 stylesheet是样式表 href是文件路径

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 …

选择器{样式}

css写在head里用<style></style>里写

可以理解为 给谁改样式{改成什么样}

例子：

p{color:red//选择p这个东西，给他改color成红色

font-size:12px} //选择p这个东西，给他改字体大小成12px像素

选择器是用于指定css样式的HTML标签，花括号里是对改对象设置的具体样式

属性和属性值以"键值对"的形式出现 什么：什么;

一组属性之后一定要加;

属性是对指定对象设置的样式属性，例如字体大小，文本颜色

属性和属性值之间用：分开

多个键值对之间用英文 ; 进行区分

选择器就是用来选择标签的

选择器有基础选择器和复合选择器

基础选择器有：标签选择器，类选择器，id选择器，通配符选择器

标签选择器：用来选择标签。但是不能只单一的修改这个标签的某一个，只能这个标签所以的修改

类选择器：可以选出一个或多个标签，用.去定义，class=""调用

格式：

.名字{

修改的内容

}

<标签 class="名字"></标签>//等于给标签取一个名字

id选择器：一次只能选择一个标签，用#定义，id=""调用

格式：

#名字{

修改的内容

}

<标签 class="名字"></标签>//等于给标签取一个身份证编号，唯一

通配符选择器：选择所有标签 用\*定义，直接作用所有标签

格式

\* {

修改的内容

}

<标签 ></标签>

复合选择器：是建立在基础选择器之上的，对基础选择器进行组合形成的

复合选择器是由两个或多个基础选择器通过不同的方式组合而成的

常用的复合选择器包括：后代选择器，子选择器，并集选择器，伪类选择器等等

后代选择器：

后代选择器又称为包含选择器，可以选择父元素里面的子元素，其写法就是把外层标签写在前面，内层标签写在后面，中间用空格分隔，当标签发生嵌套时，内层标签就成为外层标签的后代,可以有多个后代的后代的后代的后代

语法：

元素1 元素2 元素3{

样式声明

}

例如 ul li a{样式声明} //一层一层的去找 ，也可以用class 和id定义名字用来区分

元素和元素之间用空格隔开

最终选择的是最后一个后代

元素1和元素2和元素3可以是任意基础选择器

子选择器：

子选择器只能选择作为某元素的最近一级子元素，简单理解就是选亲儿子元素

语法：

元素1>元素2{样式声明}

上述语法表示选择元素1里面的所有直接后代元素2

例如：

div>p{样式声明} //选择div里面所有最近一级p标签元素

元素1和元素2之间用大于号>隔开或选择

元素1是父级，元素2是子级，最终选择的是元素2

元素2必须是亲儿子，直属后代

并集选择器：

并集选择器可以选择多组标签，同时为他们定义相同的样式，通常用于集体声明

并集选择器是各选择器通过英文逗号,连接而成，任何形式的选择器（基础选择器和复合选择器）都可以作为并集选择器的一部分

语法：约定的语法规范，并集选择器要竖着写

元素1，

元素2{样式声明}

上述语法表示选择元素1和元素2

例如：

ul,div{样式声明} //选择ul和div标签元素

元素1和元素2中间用逗号隔开

逗号可以理解为和的意思

并集选择器通常用于集体声明

伪选择器：

伪选择器用于向某些选择器添加特殊的效果，比如给链接添加特殊效果，或选择第一个，第n个元素

为类选择器书写最大的特点是用冒号:表示，比如 :选择器名

因为伪类选择器很多，比如有链接伪类，结构伪类等

链接伪类选择器

a:link //选择所有未被访问的链接

a:visited //选择所有已被访问的链接

a:hover //选择鼠标指针位于骑上的链接 （如果要经过一个，控制另一个，要包含关系，并且只能一个选择器）

a:active //选择活动链接（鼠标正在按下未弹起的链接）

链接伪类选择器注意事项：

1.为了确保生效，要按照link，visited，hover，active顺序进行声明

2.因为a链接在浏览器中具有默认样式，所以实际工作中都需要给链接单独指定样式

链接伪类选择器实际开发中的写法：一般不会全写

a {color: red}; //所有标签a

a:hover{color: gray} //鼠标经过的时候

focus伪类选择器

:focus 伪类选择器用于选取获得焦点的表单元素

焦点就是光标，焦点一般情况在<input>类表单元素才能获取，因此这个选择器也主要针对于表单元素来说

语法：

input:focus{background-color:yellow;}

font属于用于定义字体系列，大小，粗细，文字样式（如斜体）

字体系列

使用font-family属性定义文本的字体系列

body {font-family: Arial,"Microsoft Yahei","微软雅黑"}//先第一个，第一个兼容失败继续第二个，都失败就系统默认字体

各个字体之间必须使用英文状态下的,逗号隔开

一般情况下，如果有空格隔开的多个单词组成的字体，加引号

尽量使用系统默认的字体，保证所有用户都能使用

最常见的几个字体：'Microsoft YaHei',tahoma,arial,'Hiragion Sans GB'

字体大小

font-size:大小px；

最好设置一个大小给body

字体粗细

font-weight: 700; //700效果等同于加粗

font-weight: 400; //400效果等同于系统默认

normal; 是默认

文字样式

fon-style: italic; //改成斜体

normal；是默认

字体复合属性

字体复合可以让更节约代码

body{

font: style weight sizi/line-height family;

} //以这样的格式写出来，顺序不能错，并且各个属性间以空格隔开

不需要设置的属性可以省略（会取默认值），但是必须保留sizi和family属性，佛则font复合属性将不起作用

在head部分直接写link.css

会出来<link rel="stylesheet" href="./style.css">

<!--link是引用外部文档 rel是关联的意思 后面的值是关联方式 href是来自什么位置-->

list-style:none;

大部分网页美工效果图都是利用ps制作的，所以大部分切图工作也是在ps里面完成的

文件->打开：可以打开要测量的图片

ctrl+R：可以打开标尺，或者视图->标尺

右击标尺，把里面的单位改为像素

ctrl+加号（+）可以放大视图，ctrl+减号（-）可以缩小视图

按住空格键，鼠标可以变成小手，拖动ps视图

用选区拖动，可以测量大小

ctrl+D可以取消选区，或者在旁边空白处点击一下也可以取消选区

text文本属性可以定义文本的外观，如文本的颜色，对齐文本，装饰文本，文本缩进，行间距等

文本颜色：

color属性定义文本颜色

有三种写法

第一种 color：颜色；//预定义的颜色值

第二种 color: #FF0000 //十六进制

第三种 color: rgb（255,0,0）或rgb（100%，0%，0%）

对齐文本：

text-align属性用于设置元素内文本内容的水平对齐方式

属性值

text-align: left //左对齐

right //右对齐

center //居中对齐

装饰文本：

text-decoration属性规定添加到文本的修饰，可以给文本添加下划线，删除线，上划线

属性值

text-decoration: none //默认没有装饰线

underline //下划线，链接a自带下划线

overline //上划线

line-through //删除线

文本缩进：（收缩和前进）

text-indent属性用来指定文本的第一行缩进，通常是将段落首行缩进

text-indent: 20px;//负数也可以

2em;// 一个em=1个当前文字的大小

行间距：

line-height属性用于设置行间的距离（行高），可以控制文字行与行之间的距离

行间距由三部分组成 1上间距 2文本高度 3下间距 ，改变的是上间距和下间距，文本高度不变

line-height: 26px;// 26先给文本高度（字体）分，然后剩下的给上间距和下间距平分

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）和文字垂直对齐。

官方解释： 用于设置一个元素的垂直对齐方式，但是它只针对于行内元素或者行内块元素有效。

语法：

vertical-align : baseline | top | middle | bottom

baseline//默认，元素放置在父元素的基线上 | top//把元素的顶端与行中最高元素的顶端对齐 | middle//把元素放置在父元素的中部 | bottom//把元素的顶端与最低的元素的顶端对齐

图片、表单和文字对齐

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。

解决图片底部默认空白缝隙问题

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;

巧妙利用一个技术更快更好的布局：

1.margin负值的运用

2.文字围绕浮动元素

3.行内块的巧妙运用

4.CSS三角强化

1. margin负值运用：

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（如果没有有定位，则加相对定位（保留位置），如

果有定位，则加z-index）

2. 文字围绕浮动元素：

巧妙运用浮动元素不会压住文字的

特性

3. 行内块巧妙运用：

页码在页面中间显示:

1. 把这些链接盒子转换为行内块， 之后给父级指定 text-align:center;

2. 利用行内块元素中间有缝隙，并且给父级添加 text-align:center; 行内块元素会水平会居中

4. CSS 三角强化：

原理是三角形用定位覆盖，三角形用宽和高为0的盒子做的

width: 0;

height: 0;

border-color: transparent red transparent transparent;

border-style: solid;

border-width: 22px 8px 0 0;

为什么需要定位

某个元素可以自由的在一个盒子内移动位置，并且压住其他盒子.

当我们滚动窗口的时候，盒子是固定屏幕某个位置的。

以上效果，标准流或浮动都无法快速实现，此时需要定位来实现。

所以：

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。

以上效果，标准流或浮动都无法快速实现，此时需要定位来实现。

所以：

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：

static 静态定位

relative 相对定位

absolute 绝对定位

fixed 固定定位

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。

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 ：80px 底部偏移量，定义元素相对于其父元素下边线的距离

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：80px 右侧偏移量，定义元素相对于其父元素右边线的距离

1.静态定位 static：

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; } 然后在样式里写偏移

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

2. 相对定位 relative：

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。

因此，相对定位并没有脱标。它最典型的应用是给绝对定位当爹的。。。

3. 绝对定位 absolute：

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）

所以绝对定位是脱离标准流的。

子绝父相的由来：

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。

子绝父相的由来

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。

4.固定定位 fixed：

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

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。

5.粘性定位 sticky：（写多少参数就是自身碰到视窗为起点的这个距离的时候开始生效）

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。

1. 一定记住 相对定位、固定定位、绝对定位 两个大的特点： 1. 是否占有位置（脱标否） 2. 以谁为基准点移

动位置。

2. 学习定位重点学会子绝父相

定位叠放次序 z-index Z轴空间高度（优先级）

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

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。

定位特殊特性：

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

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

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文字（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

网页布局的本质——用css来摆放盒子

传统网页的三种布局方式

1.普通流（标准流）

2.浮动

3.定位

这三种布局方式都是用来摆放和字的，盒子摆放到合适位置，布局自然就完成了

实际开发中，一个页面基本都包含了这三种布局方式（后面移动端学习新的布局方式）

1. 标准流：所谓的标准流就是标签按照规定好默认方式排列：

1.块级元素会独占一行，从上向下顺序排列

常见的元素：div，hr，p，h1-h6,ul,ol,dl,form,table

2.行内元素会按照顺序，从左到右的顺序排列，碰到父元素边缘则自动换行

常见的元素：span,a,iu,em等

以上都是标准流布局，我们前面学习的就是标准流，标准流是最基本的布局方式

2. 浮动：

为什么需要浮动？

有很多的布局效果标准流没有办法完成，此时就可以利用浮动完成布局，因为浮动可以改变元素标签默认的排列

浮动最典型的应用：可以让多个块级元素一行内排列显示

网页布局第一准则：多个块级元素纵向排列找（是）标准流，多个块级元素横向排列找（是）浮动

什么是浮动？

float 属性用于创建浮动框，将其移动到一遍，直到左边缘或右边缘触及包含块或另一个浮动框的边缘

语法：

选择器{float: 属性值;}，属性值有none（元素不浮动，默认值），left（元素向左浮动），right（元素向右浮动）

浮动的特性

加了浮动之后的元素，会具有很多特性需要掌握：

1.浮动元素会脱离标准流（脱标）

2.浮动的元素会一行内显示并且元素顶部对齐

3.浮动的元素会具有行内块元素的特性

设置了浮动（float）的元素最重要的特点：

1.脱离标准普通流的控制（浮起来移动到）指定位置，俗称脱标

2.浮动的盒子不再保留原先的位置

如果多个盒子都设置了浮动，则他们会按照属性值一行内显示并且对齐顶端排列

注意：浮动的元素是相互贴靠在一起的（不会有缝隙），如果父级宽度装不下这些浮动的盒子，多出来的盒子会另起一行对齐

浮动元素会具有行内块元素特性：

任何元素都可以浮动，不管原先是什么模式的元素，添加了浮动之后具有行内块元素相似的特性

如果块级盒子没有设置宽度，默认宽度和父级一样宽，但是添加浮动之后，塔刀大小根据内容来决定

浮动的盒子中间是没有缝隙的，是紧挨着一起的

行内元素同理

为了约束浮动元素位置，网页布局一般采取的策略是：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列左右位置，符合网页布局第一准则

浮动布局的注意点：

1.浮动和标准流的父盒子搭配

先用标准流的父元素排列上下位置，然后内部的子元素采用浮动排列左右位置

2.一个元素浮动了，理论上其余的兄弟元素也要浮动

一个盒子里面有多个子盒子，如果其中一个盒子浮动了，那么其他兄弟也应该浮动，以防引起问题

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

为什么要清除浮动？

由于父级盒子很多情况下，不方便给高度，但是子盒子浮动又不占有位置，最后父级盒子高度为0时，影响下面的标准流盒子

由于浮动元素不再占用原来的位置，所以他会对后面的元素排版产生影响

清除浮动的本质：

清除浮动的本质是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

清除浮动之后，父级盒子就会根据浮动的子盒子自动检测高度，父级有了高度，就不会影响下面的标准流了

清除浮动的语法：

选择器 {clear:属性值；} //属性值有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），实际工作中，几乎只用cleear: both;

清除浮动的策略是：闭合浮动

清除浮动高度方法: 1.额外标签法（隔离法） 2.父级添加overflow属性 3.父级添加after伪元素4.父级添加双伪元素

1.额外标签法（隔离法）：

额外标签法会在最后一个浮动元素后面添加一个空的标签，例如<div style="clear:both"></div>或其他块级元素

优点：通俗易懂，书写方便

缺点：添加许多无意义的标签，结构化较差

注意：这个清除浮动的盒子必须是块级元素，不能是行内元素

2.清除浮动——父级添加overflow：

可以给父级添加overflow属性，将其属性设置为hidden，auto，scroll

注意：是给父级元素添加代码

优点：代码简介

缺点：无法显示溢出的部分

3.清除浮动——:伪元素法：

:after方式是额外标签法的升级版，也是给父元素添加

语法：

.clearfix:after{

content:"";

display:block;

height:0;

clear:both;

visibility:hidden;

}

.clearfix { //IE6,7专有

\*zoom: 1;

} //然后给父元素 弄一个class名clearfix

优点：没有增加标签，结构更简单

缺点：照顾低版本浏览器

用这个方法的网站：百度，淘宝网，网易等

3.清除浮动——双伪元素清除浮动：

这种方法也是给父级元素添加的

语法：

.clearfix:before, .clearfix:after{

content:"";

display:table;

}

.clearfix:after {

clear:both;

}

.clearfix {

\*zoom:1;

} //然后给父元素 一个类名：clearfix

优点：代码更简洁

缺点：照顾低版本浏览器

用这个方法的网站代表：小米，腾讯等；

网页布局要学习三大核心，盒子模型，浮动，定位，学号盒子模型能非常好的布局页面

网页布局的过程：

1. 先准备好相关的网页元素，网页元素基本都是盒子box

2. 利用css设置好盒子样式，然后摆放到相应的位置

3. 往盒子里面装内容

网页布局的核心本质就是利用CSS控制盒子

盒子模型box model的组成：

所谓 盒子模型 ：就是把html页面中的布局元素看作是一个矩形的盒子，也就是一个盛装内容的容器

css盒子模型本质上就是一个盒子，封装周围的html元素，这个盒子里面包括：有border边框，content内容，padding内边距，margin外边距

边框（border）：

border可以设置元素的边框。

边框由三部分组成：边框宽度（粗细），边框样式，边框颜色

语法：

border:border-width //定义边框的粗细，单位是px

border-style //边框的样式

border-color //边框的颜色

border-style的样式有：none（无边框），solid（视线边框），dashed（虚线边框），dotted（点线边框），hidden（隐藏边框），double（双线边框），groove（3D凹槽），ridge（菱形边框），inset（3D凹边），outset（3D凸边）

边框的复合写法（简写）：

border: width style color；没有顺序

边框也可以分开来写：

border-top: 1px solid red; //只设定上边框

-bottm //下边框

-left //左边框

-right //右边框

表格的细线边框：

border-collapse属性控制浏览器绘制表格边框的方式，它控制相邻单元格的边框

语法：

border-collapse:collapse;//collapse单词是合并的意思，border-collapse:collapse;表示相邻边框合并在一起

边框会影响盒子的实际大小，因此我们有两种方案解决：

1. 测量盒子大小的时候不量边框

2. 如果测量的时候包含了边框，则需要width和height减去边框宽度

内边距padding：

padding 属性用于设置内边框，即边框和内容之间的距离

语法：

padding-left: 1px; //左内边距

padding-right: 1px; //右内边距

padding-top: 1px; //上内边距

padding-bottm: 1px; //下内边距

padding的简写：可以有一到四个值

padding：1px; //1个值代表上下左右的内边框都可以设置

padding：1px 2px; //2个值代表上下和左右

padding：1px 2px 3px; //3个值代表上和左右和下

padding：1px 2px 3px 4px; //4个值代表上和右和下和左，顺时针方向走

实际发开，这四种情况都会遇到

padding也会影响盒子实际大小，也就是说，如果盒子已经有了宽度和高度，此时在指定内边框，会撑大盒子，解决方案：如果保证盒子跟效果图大小保持一致，则让width和height减去多出来的内边框大小即可

如果盒子本身没有指定width和height属性，则此时padding不会撑开盒子大小

外边距margin：

margin 属性用于设置外边框，即控制盒子和盒子之间的距离

语法：

margin-left: 1px; //左外边距

margin-right: 1px; //右外边距

margin-top: 1px; //上外边距

margin-bottom: 1px; //下外边距

margin的简写方式：代表意义和padding一样

margin：1px; //1个值代表上下左右的内边框都可以设置

margin：1px 2px; //2个值代表上下和左右

margin：1px 2px 3px; //3个值代表上和左右和下

margin：1px 2px 3px 4px; //4个值代表上和右和下和左，顺时针方向走

外边距典型应用：

外边距可以让块级盒子水平居中，但必须满足两个条件：

1. 盒子必须制定了宽度width

2. 盒子左右的外边距都是auto（自适应）

语法：

.header {width: 960px;margin: 0 auto;}

常见的写法，以下三种都可以：

第一种： margin-left:auto;margin-right: auto;

第二种: margin: auto;

第三种： margin: 0 auto; //这是最常用的

注意：以上方法是让块级元素水平居中,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

外边距合并：

使用margin定义块级元素的垂直外边距时，可能会出现外边距的合并

主要有两种情况：1.相邻块元素垂直外边距的合并 2.嵌套块元素垂直外边距的塌陷

1.相邻块元素垂直外边距的合并：两个盒子之间的距离由他们其中一个的最大值决定

当上下相邻的两个块元素（兄弟关系）相遇时，如果上面的元素有下外边框距margin-bottom,下面的元素有margin-top，那么他们之间的垂直间距不是margin-bottom跟margin-top的和，而是取两个值中的较大者当做他们之间的外边距，这种现象被称为相邻块元素垂直外边距的合并。

解决方案：尽量只给一个盒子添加margin值

2.嵌套块元素垂直外边距的塌陷：

对于两个嵌套关系（父子关系）的块元素，父元素有上边距的同时子元素也会有上外边框，此时父元素会塌陷较大的外边距值，父元素外边距和子元素外边距谁大就取谁的，如何解决？，同级可以相互作用边距，父子级需要有参照物边框，父级有参照物边框，子级才能和父级外边距边框操作

解决方案：

1. 可以为父元素定义上边框

2. 可以为父元素定义上内边框

3. 可以为父元素添加overflow:hidden;

清除内外边距：

网页元素很多都带有默认的内外边距，而且不同浏览器默认的也不一致，因此我们在不拒签，首先要清除网页元素的内外边距

语法： //这也是css的第一行代码，清除网页元素的内外边距

\* {

padding: 0; //清除内边距

margin :0 //清除外边距

}

需要注意的是：行内元素为了照顾兼容性，尽量只设置左右边距，不要设置上下左右边距，但是转换为块级元素和行内块元素就可以设置上下左右边距了

css3中新增了盒子阴影，我们可以使用box-shadow属性为盒子添加阴影

语法：

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

h-shadow//必需要有的，水平阴影的位置，允许负值 x轴正值是往右，负值是往左

v-shadow//必需要有的，垂直阴影的位置，允许负值 y轴正值是往下，负值是往上

blur //可选项，模糊距离 影子是模糊的还是实的 越大越模糊，越小越实

spread //可选项，阴影的尺寸 影子的大小

color //可选项，阴影的颜色

inset //可选项，将外部阴影（outset）改为内部阴影，默认的是外阴影，但是不可以写这个单词，否则导致阴影无效

盒子阴影不占用空间，不会影响其他盒子的排列

文字阴影：

在ccs3中，我们可以使用text-shadow 属性将阴影应用于文本

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

h-shadow //必需要有的，水平阴影的位置

v-shadow //必需要有的，垂直阴影的位置

blur //可选项，模糊距离

color //也遇到颜色

在css3中，新增了圆角边框样式，这样我们的盒子就可以变圆角了

border-radius 属性用于设置元素的外边框圆角

语法：

border-radius: length;//length是长度这个位置写数值

radius 半径（圆的半径）原理：（椭圆）圆与边框的交集形成圆角效果，生成一个多少半径的圆，用这个圆去碰边线，然后代替角

如果想要设置一个圆，需要是正方形的框，把数值修改为高度或宽度的一半即可，或者直接写为50%

如果想要一个矩形圆角，需要一个矩形，把数值设置高度的一半就可以

该属性是一个简写属性，可以跟四个值，分别代表左上角，右上角，右下角，左下角（顺时针）

如果跟2个值，分别代表左上角右下角，右上角左下角（对角线）

如果跟3个值，分别代表左上角，右上角左下角，右下角

也可以单独一个角写法：

border-top-left-radius:数值 //左上角

border-top-right-radius:数值 //右上角

border-bottom-right-radius:数值 //右下角

border-bottom-left-radius:数值 //左下角

div {

width: 0;

height: 0;

line-height: 0;

font-size: 0;

border: 50px solid transparent;

border-left-color: pink;

}

一条边有颜色，其他边透明色

为什么需要精灵图

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。

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（sprites）的使用

使用精灵图核心：

1. 精灵技术主要针对于背景图片使用。就是把多个小背景图片整合到一张大图片中。

2. 这个大图片也称为 sprites 精灵图 或者 雪碧图

3. 移动背景图片位置， 此时可以使用 background-position 。

4. 移动的距离就是这个目标图片的 x 和 y 坐标。注意网页中的坐标有所不同

5. 因为一般情况下都是往上往左移动，所以数值是负值。

6. 使用精灵图的时候需要精确测量，每个小背景图片的大小和位置。

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。）

网页常见布局就网上查好了

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。

favicon.ico 一般用于作为缩略的网站标志，它显示在浏览器的地址栏或者标签上。

目前主要的浏览器都支持 favicon.ico 图标。

一、制作favicon图标

二、favicon图标放到网站根目录下

三、 HTML页面引入favicon图标

制作favicon图标

1. 把品优购图标切成 png 图片。

2. 把 png 图片转换为 ico 图标，这需要借助于第三方转换网站，例如比特虫：http://www.bitbug.net/

HTML页面引入favicon图标

在html 页面里面的 <head> </head>元素之间引入代码。

<link rel="shortcut icon" href="favicon.ico" type="image/x-icon"/>

SEO（Search Engine Optimization）汉译为搜索引擎优化，是一种利用搜索引擎的规则提高网站在有关搜索

引擎内自然排名的方式。

SEO 的目的是对网站进行深度的优化，从而帮助网站获取免费的流量，进而在搜索引擎上提升网站的排名，提

高网站的知名度。

页面必须有三个标签用来符合 SEO 优化0

1. title 网站标题

title 具有不可替代性，是我们内页的第一个重要标签，是搜索引擎了解网页的入口和对网页主题归属的最佳判断点。

建议：网站名（产品名）- 网站的介绍 （尽量不要超过30个汉字）

例如：

 京东(JD.COM)-综合网购首选-正品低价、品质保障、配送及时、轻松购物！

 小米商城 - 小米5s、红米Note 4、小米MIX、小米笔记本官方网站。

2. description 网站说明

简要说明我们网站主要是做什么的。

我们提倡，description 作为网站的总体业务和主题概括，多采用“我们是…”、“我们提供…”、“×××网

作为…”、“电话：010…”之类语句。

例如：

<meta name="description" content="京东JD.COM-专业的综合网上购物商城,销售家电、数码通讯、电脑、

家居百货、服装服饰、母婴、图书、食品等数万个品牌优质商品.便捷、诚信的服务，为您提供愉悦的网上购物

体验!" />

3. keywords 关键字

keywords 是页面关键词，是搜索引擎的关注点之一。

keywords 最好限制为 6～8 个关键词，关键词之间用英文逗号隔开，采用 关键词1,关键词2 的形式。

例如：

<meta name= " keywords" content="网上购物,网上商城,手机,笔记本,电脑,MP3,CD,VCD,DV,相机,数码,配

件,手表,存储卡,京东" />

为了提高网页制作的效率，布局时通常有以下的整体思路：

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

实际开发中，我们不会直接用链接a 而是用 li 包含链接(li+a)的做法。

1. li+a 语义更清晰，一看这就是有条理的列表型内容。

2. 如果直接用a，搜索引擎容易辨别为有堆砌关键字嫌疑（故意堆砌关键字容易被搜索引擎有降权的风险），

从而影响网站排名

1.布局为什么用不同的盒子，什么地方用什么标签？

标签都是有语义的，合理的地方用合理的标签，比如产品标题就用h，大量文字段落就用p

2.为什么每个盒子都取一个类名？

为了更好的找到这个盒子，选取盒子更容易，后期维护也方便

3.到底用margin还是padding？

可以混用，两者各有优缺点，根据实际情况，用更简单的方法实现

4.布局思路

这有什么盒子，盒子里面有什么，先画张图，然后完善细节

单行文本溢出显示省略号--必须满足三个条件

/\*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;

多行文本溢出显示省略号

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;

所谓的界面样式，就是更改一些用户操作样式，以便提高更好的用户体验。

 更改用户的鼠标样式

 表单轮廓

 防止表单域拖拽

鼠标样式 cursor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。

语法：

li {cursor: pointer; }

default //默认

pointer //小手

move //移动

text //文本

not-allowed //禁止

轮廓线 outline

给表单添加 outline: 0; 或者 outline: none; 样式之后，就可以去掉选定这个框之后默认的蓝色边框。

语法：

input {outline: none; }

防止拖拽文本域 resize

实际开发中，我们文本域右下角是不可以拖拽的。

语法：

textarea{ resize: none;}

类似网站广告，当我们点击关闭就不见了，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

2. visibility 显示隐藏

3. overflow 溢出显示隐藏

display 属性

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 可见性

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 溢出

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。

visible(不剪切内容也不添加滚动条） hidden（超出的部分隐藏掉）scroll（不管有没有超出，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）

1. display 显示隐藏元素 但是不保留位置

2. visibility 显示隐藏元素 但是保留原来的位置

3. overflow 溢出显示隐藏 但是只是对于溢出的部分处理

元素显示模式。 作用：网页标签非常多，在不同的地方会用到不同的标签，了解标签的特点可以帮助更好的布局网页

元素显示模式就是（元素标签）以什么样的形式显示。 比如div自己占一行。 span一行可以有很多个

html元素一般分为块元素和行内元素两种类型

块元素：

常见的块元素有<h1>-<h6>,<p>,<div><ul>,<ol>,<li>等，div标签是典型的块元素

块元素的特点

1. 比较霸道，自己独占一行

2. 高度，宽度，外边距，内边距都可以控制

3. 宽度默认是容器（父级宽度）的100%

4. 是一个容器及盒子，里面可以放行内元素或者块级元素

需要注意的是：

1. 文字类的元素内不能使用块元素，p标签主要用于存放文字，所以p标签里面不能放块级元素，特别不能放div

2.同理h1到h6都是文字类块级元素，所以里面也不能放其他块级元素

行内元素：

常见的行内元素有<a>,<strong>,<b>,<em>,<i>,<del>,<s>,<ins>,<u>,<span>,其中<span>标签是最典型的行内元素，有的地方也将行内元素成为内联元素

行内元素的特点：

1. 相邻行内元素在一行上，一行可以显示多个

2. 高，宽直接设置是无效的

3. 默认宽度就是它本身内容的宽度

4. 行内元素智能容纳文本或其他行内元素

需要注意的是：

1. 链接里面不能再放链接了

2. 特殊情况链接a里面可以放块级元素，但是给a转换一下块级模式最安全

行内块元素：//简答来说，就是一个可以设置宽高，但是不独占一行的块元素

在行内元素中有几个特殊的标签——<img/><input><td>，它们同时具有块元素和行内元素的特点，有些资料称它们为行内块元素

行内块元素的特点：

1. 和相邻行内元素在一行上面，但是它们之间会有空白间隙，一行可以显示多个（行内元素的特点）

2. 默认宽度就是它本身内容的宽度（行内元素特点）

3. 高度，行高，外边距以及内边距都可以控制（块级元素特点）

元素显示模式的转换

特殊情况下，我们需要元素模式的转换，简单理解：一个模式的元素需要另外一种模式的特性

比如想要增加链接a的触发范围，就把a链接转换为块元素，触发范围就可以调整

转换块元素：display: block;

转换为行内元素：display: inline;

转换为行内块元素：display: inline-block

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。

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。

字体图标是一些网页常见的小图标，我们直接网上下载即可。 因此使用可以分为：

1. 字体图标的下载

2. 字体图标的引入 （引入到我们html页面中）

3. 字体图标的追加 （以后添加新的小图标）

下载：

icomoon 字库 https://icomoon.io

阿里 iconfont 字库 https://www.iconfont.cn/

给标签定义字体。

span {

font-family: "icomoon";

}

/\*CSS3盒子模型\*/

box-sizing: border-box;

-webkit-box-sizing: border-box;

/\*点击高亮我们需要清除清除 设置为transparent 完成透明\*/

-webkit-tap-highlight-color: transparent;

/\*在移动端浏览器默认的外观在iOS上加上这个属性才能给按钮和输入框自定义样式\*/

-webkit-appearance: none;

/\*禁用长按页面时的弹出菜单\*/

img,a { -webkit-touch-callout: none; }

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。这是厂商在出厂时就设置好了,比如苹果6\7\8 是 750\* 1334

 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，1个px 等于1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

PC端 和 早前的手机屏幕 / 普通手机屏幕: 1CSS像素 = 1 物理像素的

 Retina（视网膜屏幕）是一种显示技术，可以将把更多的物理像素点压缩至一块屏幕里，从

而达到更高的分辨率，并提高屏幕显示的细腻程度。

多倍图

对于一张 50px \* 50px 的图片,在手机 Retina 屏中打开，按照刚才的物理像素比会放大倍数，这样会造成图片模糊

 在标准的viewport设置中，使用倍图来提高图片质量，解决在高清设备中的模糊问题

 通常使用二倍图， 因为iPhone 6\7\8 的影响,但是现在还存在3倍图4倍图的情况，这个看实际开发公司需求

 背景图片 注意缩放问题

以二倍图为例

/\* 在 iphone8 下面 \*/

img{

/\*原始图片100\*100px\*/

width: 50px;

height: 50px;

}

.box{

/\*原始图片100\*100px\*/

background-size: 50px 50px;

}

背景缩放 background-size

background-size 属性规定背景图像的尺寸

background-size: 背景图片宽度 背景图片高度；

单位： 长度|百分比|cover|contain;

 cover把背景图像扩展至足够大，以使背景图像完全覆盖背景区域。

 contain把图像图像扩展至最大尺寸，以使其宽度和高度完全适应内容区域

多倍图切图 cutterman

@3X 3倍图

 @2X 2倍图

 @1X 1倍图原图

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、视觉视口和理想视口

布局视口 layout viewport

 一般移动设备的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上的网页大多都能在手机上呈现，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

字面意思，它是用户正在看到的网站的区域。注意：是网站的区域。

 我们可以通过缩放去操作视觉视口，但不会影响布局视口，布局视口仍保持原来的宽度

理想视口 ideal viewport

 为了使网站在移动端有最理想的浏览和阅读宽度而设定

 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视口标签通知浏览器操作

 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

局的视口就多宽

视口就是浏览器显示页面内容的屏幕区域

 视口分为布局视口、视觉视口和理想视口

 我们移动端布局想要的是理想视口就是手机屏幕有多宽，我们的布局视口就有多宽

 想要理想视口，我们需要给我们的移动端页面添加 meta视口标签

meta视口标签

<meta name="viewport" content="width=device-width, user-scalable=no,

initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

width 宽度设置的是viewport宽度，可以设置device-width特殊值

initial-scale 初始缩放比，大于0的数字

maximum-scale 最大缩放比，大于0的数字

minimum-scale 最小缩放比，大于0的数字

user-scalable 用户是否可以缩放，yes或no（1或0）

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

CSS初始化 normalize.css

移动端 CSS 初始化推荐使用 normalize.css/

 Normalize.css：保护了有价值的默认值

 Normalize.css：修复了浏览器的bug

 Normalize.css：是模块化的

 Normalize.css：拥有详细的文档

官网地址： <http://necolas.github.io/normalize.css/>

单独制作移动端页面（主流）

京东商城手机版

淘宝触屏版

苏宁易购手机版

响应式页面兼容移动端（其次）

三星手机官网

单独移动端页面（主流）

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

响应式兼容PC移动端

三星电子官网： www.samsung.com/cn/ ，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

总结

现在市场常见的移动端开发有 单独制作移动端页面 和 响应式页面 两种方案

现在市场主流的选择还是单独制作移动端页面

DPG图片压缩技术

京东自主研发推出DPG图片压缩技术，经测试

该技术，可直接节省用户近50%的浏览流量，

极大的提升了用户的网页打开速度。

能够兼容jpeg，实现全平台、全部浏览器的兼

容支持，经过内部和外部上万张图片的人眼浏

览测试后发现，压缩后的图片和webp的清晰度

对比没有差距

webp 图片格式

谷歌开发的一种旨在加快图片加载速度的图片

格式。图片压缩体积大约只有JPEG的2/3，并

能节省大量的服务器宽带资源和数据空间

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

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成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布局方式。

 max-width 最大宽度 （max-height 最大高度）  min-width 最小宽度 （min-height 最小高度）

设置视口标签以及引入初始化样式

<meta name="viewport" content="width=device-width, user-scalable=no,

initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<link rel="stylesheet" href="css/normalize.css">

<link rel="stylesheet" href="css/index.css">

常用初始化样式

body {

margin: 0 auto;

min-width: 320px;

max-width: 640px;

background: #fff;

font-size: 14px;

font-family: -apple-system, Helvetica, sans-serif;

line-height: 1.5;

color: #666;

}

移动端布局之流式布局

1. 标准viewport规范以及写法

2. 模拟移动端调试方法

3. 移动端常见的布局方案

4. 流式布局原理

5. 京东移动端首页布局技巧

背景线性渐变

语法1：

background: linear-gradient(起始方向, 颜色1, 颜色2, ...);

background: -webkit-linear-gradient(left, red , blue);

background: -webkit-linear-gradient(left top, red , blue);

背景渐变必须添加浏览器私有前缀

起始方向可以是： 方位名词 或者 度数 ， 如果省略默认就是 top

传统布局与flex布局

传统布局

 兼容性好

 布局繁琐

 局限性，不能再移动端很好的布局

flex 弹性布局

 操作方便，布局极为简单，移动端应用很广泛

 PC 端浏览器支持情况较差

 IE 11或更低版本，不支持或仅部分支持

建议：

1. 如果是PC端页面布局，我们还是传统布局。

2. 如果是移动端或者不考虑兼容性问题的PC端页面布局，我们还是使用flex弹性布局

CSS样式

① span 直接给宽度和高度，背景颜色，还有蓝色边框

② 给 div 只需要添加 “display：flex” 即可

布局原理

flex 是 flexible Box 的缩写，意为"弹性布局"，用来为盒状模型提供最大的灵活性，任何一个容器都可以

指定为 flex 布局。

 当我们为父盒子设为 flex 布局以后，子元素的 float、clear 和 vertical-align 属性将失效。

 伸缩布局 = 弹性布局 = 伸缩盒布局 = 弹性盒布局 =flex布局

采用 Flex 布局的元素，称为 Flex 容器（flex container），简称"容器"。它的所有子元素自动成为容器成

员，称为 Flex 项目（flex item），简称"项目"。

体验中 div 就是 flex父容器。

 体验中 span 就是 子容器 flex项目

 子容器可以横向排列也可以纵向排列

总结flex布局原理：

就是通过给父盒子添加flex属性，来控制子盒

子的位置和排列方式

常见父项属性

以下由6个属性是对父元素设置的

 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

flex-direction 设置主轴的方向

主轴与侧轴

在 flex 布局中，是分为主轴和侧轴两个方向，同样的叫法有 ： 行和列、x 轴和y 轴

 默认主轴方向就是 x 轴方向，水平向右

 默认侧轴方向就是 y 轴方向，水平向下

属性值

flex-direction 属性决定主轴的方向（即项目的排列方向）

注意： 主轴和侧轴是会变化的，就看 flex-direction 设置谁为主轴，剩下的就是侧轴。而我们的子元素是跟着主轴来排列的

row 默认值从左到右

row-reverse 从右到左

column 从上到下

column-reverse 从下到上

justify-content 设置主轴上的子元素排列方式

justify-content 属性定义了项目在主轴上的对齐方式

注意： 使用这个属性之前一定要确定好主轴是哪个

属性值 说明

flex-start 默认值 从头部开始 如果主轴是x轴，则从左到右

flex-end 从尾部开始排列

center 在主轴居中对齐（如果主轴是x轴则 水平居中）

space-around 平分剩余空间

space-between 先两边贴边 再平分剩余空间（重要）

flex-wrap 设置子元素是否换行

默认情况下，项目都排在一条线（又称”轴线”）上。flex-wrap属性定义，flex布局中默认是不换行的。 属性值 说明

nowrap 默认值，不换行

wrap 换行

align-items 设置侧轴上的子元素排列方式（单行 ）

该属性是控制子项在侧轴（默认是y轴）上的排列方式 在子项为单项（单行）的时候使用

属性值 说明

flex-start 从上到下

flex-end 从下到上

center 挤在一起居中（垂直居中）

stretch 拉伸 （默认值 ）

align-content 设置侧轴上的子元素的排列方式（多行）

设置子项在侧轴上的排列方式 并且只能用于子项出现 换行 的情况（多行），在单行下是没有效果的。 属性值 说明

flex-start 默认值在侧轴的头部开始排列

flex-end 在侧轴的尾部开始排列

center 在侧轴中间显示

space-around 子项在侧轴平分剩余空间

space-between 子项在侧轴先分布在两头，再平分剩余空间

stretch 设置子项元素高度平分父元素高度

align-content 和 align-items 区别

 align-items 适用于单行情况下， 只有上对齐、下对齐、居中和 拉伸

 align-content 适应于换行（多行）的情况下（单行情况下无效）， 可以设置 上对齐、 下对齐、居中、拉伸以及平均分

配剩余空间等属性值。

 总结就是单行找 align-items 多行找 align-content

flex-flow

flex-flow 属性是 flex-direction 和 flex-wrap 属性的复合属性

flex-flow:row wrap;

 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

flex布局子项常见属性

flex 子项目占的份数

 align-self 控制子项自己在侧轴的排列方式

 order属性定义子项的排列顺序（前后顺序）

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。

.item {

flex: <number>; /\* default 0 \*/

}

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-items 属性。

默认值为 auto，表示继承父元素的 align-items 属性，如果没有父元素，则等同于 stretch。

span:nth-child(2) {

/\* 设置自己在侧轴上的排列方式 \*/

align-self: flex-end;

}

order 属性定义项目的排列顺序

数值越小，排列越靠前，默认为0。

注意：和 z-index 不一样。

.item {

order: <number>;

}

VSCode px 转换rem 插件 cssrem

设置html字体大小基准值

1. 打开 设置 快捷键是 ctrl + 逗号

设置html字体大小基准值

1. 页面布局文字能否随着屏幕大小变化而变化？

2. 流式布局和flex布局主要针对于宽度布局，那高度如何设置？

3. 怎么样让屏幕发生变化的时候元素高度和宽度等比例缩放？

rem 基础

rem 单位

rem (root em)是一个相对单位，类似于em，em是父元素字体大小。

不同的是rem的基准是相对于html元素的字体大小。

比如，根元素（html）设置font-size=12px; 非根元素设置width:2rem; 则换成px表示就是24px。

rem的优势：父元素文字大小可能不一致， 但是整个页面只有一个html，可以很好来控制整个页面的元素大小

。

/\* 根html 为 12px \*/

html {

font-size: 12px;

}

/\* 此时 div 的字体大小就是 24px \*/

div {

font-size: 2rem;

}

媒体查询

媒体查询（Media Query）是CSS3新语法。

 使用 @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尺寸设置不同的样式

 当你重置浏览器大小的过程中，页面也会根据浏览器的宽度和高度重新渲染页面

 目前针对很多苹果手机、Android手机，平板等设备都用得到多媒体查询

语法规范

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

CSS-Code;

}

 用 @media 开头 注意@符号

 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

值 解释说明

all 用于所有设备

print 用于打印机和打印预览

screen 用于电脑屏幕，平板电脑，智能手机等

将不同的终端设备划分成不同的类型，称为媒体类型

关键字

关键字将媒体类型或多个媒体特性连接到一起做为媒体查询的条件。

and：可以将多个媒体特性连接到一起，相当于“且”的意思。

 not：排除某个媒体类型，相当于“非”的意思，可以省略。

 only：指定某个特定的媒体类型，可以省略

媒体特性

每种媒体类型都具体各自不同的特性，根据不同媒体类型的媒体特性设置不同的展示风格。我们暂且了解三个。

注意他们要加小括号包含

width 定义输出设备中页面可见区域的宽度

min-width 定义输出设备中页面最小可见区域宽度

max-width 定义输出设备中页面最大可见区域宽度

媒体查询+rem实现元素动态大小变化

rem单位是跟着html来走的，有了rem页面元素可以设置不同大小尺寸

媒体查询可以根据不同设备宽度来修改样式

媒体查询+rem 就可以实现不同设备宽度，实现页面元素大小的动态变化

引入资源（理解）

当样式比较繁多的时候，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。

语法规范

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

. 示例

<link rel="stylesheet" href="styleA.css" media="screen and (min-width: 400px)">

rem 适配方案

1. 我们适配的目标是什么？

2. 怎么去达到这个目标的？

3. 在实际的开发当中使用？

1. 让一些不能等比自适应的元素，达到当设备尺寸发生改变的时

候，等比例适配当前设备。

2. 使用媒体查询根据不同设备按比例设置html的字体大小，然后

页面元素使用rem做尺寸单位，当html字体大小变化元素尺寸

也会发生变化，从而达到等比缩放的适配。

rem 实际开发适配方案

① 按照设计稿与设备宽度的比例，动态计算并设置 html 根标签的 font-size 大小；（媒体查询）

② CSS 中，设计稿元素的宽、高、相对位置等取值，按照同等比例换算为 rem 为单位的值； rem 适配方案技术使用（市场主流）

4. rem 适配方案

技术方案1

 less

 媒体查询

 rem

技术方案2（推荐）

 flexible.js

 rem

总结：

1. 两种方案现在都存在。

2. 方案2 更简单，现阶段大家无需了解里面的js代码。

rem + 媒体查询 + less 技术

设计稿常见尺寸宽度

设备 常见宽度

iphone 4.5 640px

iphone 678 750px

Android 常见320px、360px、375px、384px、400px、414px、500px、720px

大部分4.7~5寸的安卓设备为720px

一般情况下，我们以一套或两套效果图适应大部分的屏幕，放弃极端屏或对其优雅降级，牺牲一些效果

现在基本以750为准。

2. 动态设置 html 标签 font-size 大小

① 假设设计稿是750px

② 假设我们把整个屏幕划分为15等份（划分标准不一可以是20份也可以是10等份）

③ 每一份作为html字体大小，这里就是50px

④ 那么在320px设备的时候，字体大小为320/15 就是 21.33px

⑤ 用我们页面元素的大小 除以不同的 html 字体大小会发现他们比例还是相同的

⑥ 比如我们以 750为标准设计稿

⑦ 一个100\*100像素的页面元素 在 750屏幕下， 就是 100 / 50 转换为rem 是 2rem \* 2 rem 比例是 1比1

⑧ 320屏幕下， html 字体大小为 21.33 则 2rem = 42.66px 此时宽和高都是 42.66 但是 宽和高的比例还是 1比1

⑨ 但是已经能实现不同屏幕下 页面元素盒子等比例缩放的效果

3. 元素大小取值方法

① 最后的公式： 页面元素的rem值 = 页面元素值（px） / （屏幕宽度 / 划分的份数）

② 屏幕宽度/划分的份数 就是 html font-size 的大小

③ 或者： 页面元素的rem值 = 页面元素值（px） / html font-size 字体大小

1. 技术选型

方案：我们采取单独制作移动页面方案

技术：布局采取rem适配布局（less + rem + 媒体查询）

设计图： 本设计图采用 750px 设计尺寸

2. 搭建相关文件夹结构

3. 设置视口标签以及引入初始化样式

<meta name="viewport" content="width=device-width, user-scalable=no,

initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<link rel="stylesheet" href="css/normalize.css">

4. 设置公共common.less文件

1. 新建common.less 设置好最常见的屏幕尺寸，利用媒体查询设置不同的html字体大小，因为除了首页其他页面也需要

2. 我们关心的尺寸有 320px、360px、375px、384px、400px、414px、424px、480px、540px、720px、750px

3. 划分的份数我们定为 15等份

4. 因为我们pc端也可以打开我们苏宁移动端首页，我们默认html字体大小为 50px，注意这句话写到最上面

5. 新建index.less文件

1. 新建index.less 这里面写首页的样式

2. 将刚才设置好的 common.less 引入到index.less里面 语法如下：

// 在 index.less 中导入 common.less 文件

@import “common”

3. 生成index.css 引入到 index.html 里面

6. body样式

body {

min-width: 320px;

width:15rem;

margin: 0 auto;

line-height: 1.5;

font-family: Arial,Helvetica;

background: #F2F2F2;

}

6.1 简洁高效的rem适配方案flexible.js

手机淘宝团队出的简洁高效 移动端适配库

我们再也不需要在写不同屏幕的媒体查询，因为里面js做了处理

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。

我们要做的，就是确定好我们当前设备的html 文字大小就可以了

比如当前设计稿是 750px， 那么我们只需要把 html 文字大小设置为 75px(750px / 10) 就可以

里面页面元素rem值： 页面元素的px 值 / 75

剩余的，让flexible.js来去算

github地址：https://github.com/amfe/lib-flexible

1. 技术选型

方案：我们采取单独制作移动页面方案

技术：布局采取rem适配布局2（flexible.js + rem）

设计图： 本设计图采用 750px 设计尺寸

2. 搭建相关文件夹结构

3. 设置视口标签以及引入初始化样式还有js文件

<meta name="viewport" content="width=device-width, user-scalable=no,

initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

<link rel="stylesheet" href="css/normalize.css">

<link rel="stylesheet" href="css/index.css">

我们页面需要引入 这个js文件

在 index.html 中 引入 flexible.js 这个文件

<script src=“js/flexible.js”> </script>

4. body样式

body {

min-width: 320px;

width:15rem;

margin: 0 auto;

line-height: 1.5;

font-family: Arial,Helvetica;

background: #F2F2F2;

}

维护 css 的弊端

CSS 是一门非程序式语言，没有变量、函数、SCOPE（作用域）等概念。

 CSS 需要书写大量看似没有逻辑的代码，CSS 冗余度是比较高的。

 不方便维护及扩展，不利于复用。

 CSS 没有很好的计算能力

 非前端开发工程师来讲，往往会因为缺少 CSS 编写经验而很难写出组织良好且易于维护的 CSS 代码项目。

Less 介绍

Less （Leaner Style Sheets 的缩写） 是一门 CSS 扩展语言，也成为CSS预处理器。

做为 CSS 的一种形式的扩展，它并没有减少 CSS 的功能，而是在现有的 CSS 语法上，为CSS加入程序式语言的

特性。

它在 CSS 的语法基础之上，引入了变量，Mixin（混入），运算以及函数等功能，大大简化了 CSS 的编写，并且

降低了 CSS 的维护成本，就像它的名称所说的那样，Less 可以让我们用更少的代码做更多的事情。

Less中文网址： http://lesscss.cn/

常见的CSS预处理器：Sass、Less、Stylus

一句话：Less 是一门 CSS 预处理语言，它扩展了CSS的动态特性

Less 安装（注意如果使用vscode无需安装less）

① 安装nodejs，可选择版本(8.0)，网址：http://nodejs.cn/download/

② 检查是否安装成功，使用cmd命令（win10 是 window +r 打开 运行输入cmd） --- 输入“ node –v ”查看版本即可

③ 基于nodejs在线安装Less，使用cmd命令“ npm install -g less ”即可

④ 检查是否安装成功，使用cmd命令“ lessc -v ”查看版本即可

vocode Less 插件

Easy LESS 插件用来把less文件编译为

css文件

安装完毕插件，重新加载下 vscode。

只要保存一下Less文件，会自动生成CSS

文件。

Less 使用

我们首先新建一个后缀名为less的文件， 在这个less文件里面书写less语句。

 Less 变量

 Less 编译

 Less 嵌套

 Less 运算

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。

@变量名:值;

变量命名规范

 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

变量使用规范

//直接使用

body{

color:@color;

}

a:hover{

color:@color;

}

Less 编译

本质上，Less 包含一套自定义的语法及一个解析器，用户根据这些语法定义自己的样式规则，这些规则

最终会通过解析器，编译生成对应的 CSS 文件。

所以，我们需要把我们的 less文件，编译生成为css文件，这样我们的html页面才能使用

Less 嵌套

我们经常用到选择器的嵌套

#header .logo {

width: 300px;

}

Less 嵌套写法

#header {

.logo {

width: 300px;

}

}

如果遇见 （交集|伪类|伪元素选择器）

内层选择器的前面没有 & 符号，则它被解析为父选择器的后代；

如果有 & 符号，它就被解析为父元素自身或父元素的伪类。

a:hover{

color:red;

}

Less 嵌套写法

a{

&:hover{

color:red;

}

}

Less 运算 任何数字、颜色或者变量都可以参与运算。就是Less提供了加（+）、减（-）、乘（\*）、除（/）算术运算。

/\*Less 里面写\*/

@witdh: 10px + 5;

div {

border: @witdh solid red;

}

/\*生成的css\*/

div {

border: 15px solid red;

}

/\*Less 甚至还可以这样 \*/

width: (@width + 5) \* 2;

注意：

乘号（\*）和除号（/）的写法  运算符中间左右有个空格隔开 1px + 5

 对于两个不同的单位的值之间的运算，运算结果的值取第一个值的单位

 如果两个值之间只有一个值有单位，则运算结果就取该单位

响应式开发原理

就是使用媒体查询针对不同宽度的设备进行布局和样式的设置，从而适配不同设备的目的。

超小屏幕（手机） < 768px

小屏设备（平板） >= 768px ~ < 992px

中等屏幕（桌面显示器） >= 992px ~ <1200px

宽屏设备（大桌面显示器） >= 1200px

响应式布局容器

响应式需要一个父级做为布局容器，来配合子级元素来实现变化效果。

原理就是在不同屏幕下，通过媒体查询来改变这个布局容器的大小，再改变里面子元素的排列方式和大小，从而实现不同

屏幕下，看到不同的页面布局和样式变化。

平时我们的响应式尺寸划分

超小屏幕（手机，小于 768px）：设置宽度为 100%

 小屏幕（平板，大于等于 768px）：设置宽度为 750px

 中等屏幕（桌面显示器，大于等于 992px）：宽度设置为 970px

 大屏幕（大桌面显示器，大于等于 1200px）：宽度设置为 1170px

原理就是在不同屏幕下，通过媒体查询来改变这个布局容器的大小，再改变里面子元素的排列方式和大小，从而实现不同

屏幕下，看到不同的页面布局和样式变化。

平时我们的响应式尺寸划分

但是我们也可以根据实际情况自己定义划分

Bootstrap 来自 Twitter（推特），是目前最受欢迎的前端框架。Bootstrap 是基于 HTML、CSS 和 JAVASCRIPT 的，它

简洁灵活，使得 Web 开发更加快捷。

 中文官网：http://www.bootcss.com/

 官网：http://getbootstrap.com/

框架：顾名思义就是一套架构，它有一套比较完整的网页功能解决方案，而且控制权在框架本身，有预制样式库、组件和

插件。使用者要按照框架所规定的某种规范进行开发。

优点

 标准化的html+css编码规范

 提供了一套简洁、直观、强悍的组件

 有自己的生态圈，不断的更新迭代

 让开发更简单，提高了开发的效率

Bootstrap 使用

在现阶段我们还没有接触JS相关课程，所以我们只考虑使用它的样式库。

控制权在框架本身，使用者要按照框架所规定的某种规范进行开发。

Bootstrap 使用四步曲： 1. 创建文件夹结构 2. 创建 html 骨架结构 3. 引入相关样式文件 4. 书写内容

我们创建bootstrap文件夹复制Bootstrap相关文件夹 粘贴

2. 创建html骨架结构

<!--要求当前网页使用IE浏览器最高版本的内核来渲染-->

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<!--视口的设置：视口的宽度和设备一致，默认的缩放比例和PC端一致，用户不能自行缩放-->

<meta name="viewport" content="width=device-width, initial-scale=1, user-scalable=0">

<!--[if lt IE 9]>

<!--解决ie9以下浏览器对html5新增标签的不识别，并导致CSS不起作用的问题-->

<script src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js"></script>

<!--解决ie9以下浏览器对 css3 Media Query 的不识别 -->

<script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>

<![endif]-->

引入相关样式文件

<!-- Bootstrap 核心样式-->

<link rel="stylesheet" href="bootstrap/css/bootstrap.min.css">

书写内容

 直接拿Bootstrap 预先定义好的样式来使用

 修改Bootstrap 原来的样式，注意权重问题

 学好Bootstrap 的关键在于知道它定义了哪些样式，以及这些样式能实现什么样的效果

Bootstrap 需要为页面内容和栅格系统包裹一个 .container 容器，它提供了两个作此用处的类

1. container 类

 流式布局容器 百分百宽度

 占据全部视口（viewport）的容器。

2. container-fluid 类

 响应式布局的容器 固定宽度

 大屏 ( >=1200px) 宽度定为 1170px

 中屏 ( >=992px) 宽度定为 970px

 小屏 ( >=768px) 宽度定为 750px

 超小屏 (100%)

栅格系统英文为“grid systems”,也有人翻译为“网格系统”，它是指将页面布局划分为等宽的列，然后通过列数

的定义来模块化页面布局。

Bootstrap 提供了一套响应式、移动设备优先的流式栅格系统，随着屏幕或视口（viewport）尺寸的增加，系统会

自动分为最多12列

栅格选项参数

栅格系统用于通过一系列的行（row）与列（column）的组合来创建页面布局，你的内容就可以放入这些创建好的布局中。

超小屏幕（手机）

< 768px

小屏设备（平板）

>=768px

中等屏幕（桌面显示器）

>=992px

宽屏设备（大桌面显示器）

>=1200px

.container 最大宽度 自动(100%) 750px 970px 1170px

类前缀 .col-xs- .col-sm- .col-md- .col-lg-

列（column）数 12

 按照不同屏幕划分为1~12 等份

 行（row） 可以去除父容器作用15px的边距

 xs-extra small：超小； sm-small：小； md-medium：中等； lg-large：大；

 列（column）大于 12，多余的“列（column）”所在的元素将被作为一个整体另起一行排列

 每一列默认有左右15像素的 padding

 可以同时为一列指定多个设备的类名，以便划分不同份数 例如 class="col-md-4 col-sm-6"

列嵌套

栅格系统内置的栅格系统将内容再次嵌套。简单理解就是一个列内再分成若干份小列。我们可以通过添加一个新的 .row 元素和一系

列 .col-sm-\* 元素到已经存在的 .col-sm-\* 元素内。

<!-- 列嵌套 -->

<div class="col-sm-4">

<div class="row">

<div class="col-sm-6">小列</div>

<div class="col-sm-6">小列</div>

</div>

</div>

列偏移

使用 .col-md-offset-\* 类可以将列向右侧偏移。这些类实际是通过使用 \* 选择器为当前元素增加了左侧的边距（margin）。

<!-- 列偏移 -->

<div class="row">

<div class="col-lg-4">1</div>

<div class="col-lg-4 col-lg-offset-4">2</div>

</div>

列排序

通过使用 .col-md-push-\* 和 .col-md-pull-\* 类就可以很容易的改变列（column）的顺序。

<!-- 列排序 -->

<div class="row">

<div class="col-lg-4 col-lg-push-8">左侧</div>

<div class="col-lg-8 col-lg-pull-4">右侧</div>

</div>

响应式工具

为了加快对移动设备友好的页面开发工作，利用媒体查询功能，并使用这些工具类可以方便的针对不同设备展示

或隐藏页面内容。

类名 超小屏 小屏 中屏 大屏

.hidden-xs 隐藏 可见 可见 可见

.hidden-sm 可见 隐藏 可见 可见

.hidden-md 可见 可见 隐藏 可见

.hidden-lg 可见 可见 可见 隐藏

Bootstrap 其他（按钮、表单、表格） 请参考Bootstrap 文档

隐身效果

opacity:1

disabled 让按钮改为不可点击状态

Object-fit:cover; 可以让图片以当前比例变大缩小，父盒子不够大就超出的不显示