

进行网页编写时，先进行布局 看能分为多少个大块 大div 然后在在里面划分其他小区域 并且按区域实现目的

HTML5

HTML超文本标记语言 纯文本编辑器编辑文本 纯文本（图片其他的都不属于）只能存数据
网页是由一个一个标签构成的<>

```
<html>
```

```
    <head>
```

```
        <title>网页名字</title>子标签 title标签的内容会作为搜索结果
```

的超链接上 的文字展示

```
    </head>
```

```
    <body>
```

```
    </body>
```

出现嵌套结构时 代码需要缩进可以用tab键

语义化标签 <h1>标题</h1>一共有6级标题（独占一行的元素称为（ block element）
（h,p）

<p>内容</p> （一个p标签表示页面中的一个段落） 块元素： 网页中页面的布局 换段

>可以折一行 or

<hgroup> 标签用来为标题分组，可以将一组相关的标题同时放入到hgroup

em标签可以表示一个语音语调的一个加重（会让字斜着） 斜体字

 语义加重 加深 粗体字

行内元素（inline element）：主要用来包裹文字

一般在块元素可以放行内元素 行内元素不能放块元素 p元素中不能放任何块元素 但是一般浏览器解析网页时，会自动对不规范的内容进行修正

比如：1 标签写在根元素外面。

2 p元素中嵌套了块元素

3 根元素中出现了出head和body以外的子元素

<blockquote></blockquote> 表示引用 会换行 独占一行

<q></q>表示一个短引用 不会换号 会加双引号

```
</body>
```

```
</html>根标签（元素） 有且只能有一个
```

`<input>` 或者 `<input />` 自结束标签 只有一个 一个框框

input标签取消选中黑框的css属性 `outline: none;`



`<!-- -->` 注释

`<html lang="zh">` (中文编辑 英文编辑 改为en)

在标签中 (开始标签或者自结束标签) 还可以设置属性 属性是一个名值对 (`x=y`) 属性名=属性值

`<font+空格隔开+属性名="属性值" > `

`
` 折一行 放前面

`<meta http-equiv="refresh" content='3 '; url=http://www.4399.com">` 将页面重新定向到另一个网站

! 加tab键快速生产模板 `ctrl+/` 注释快捷键

`ctrl+回车`可以让光标下移 `alt shift +方向键` 上下 可以快速复制上下的东西 *星号 相当于编写乘以几次



语义标签

`<header> </header>` 网页头部

`<main> </main>` 网页的主体部分 (一个页面只会有一个)

`<footer> </footer>` 网页底部 (网页可大致分为上中下三部分)

`<nav> </nav>` 表示网页的导航

`<aside> </aside>` 和主体相关的其他内容 (侧边栏) `figure` 只有一个 `figcaption`

`<article> </article>` 表示一篇文章 每个 `article` 里面要有一个 `h` 段、

`<section> </section>` 表示一个独立的区块, 上边的标签都不能表示时使用 `section` (比如在 `header` 里面再继续分小块) 相对于 `div` `section` 是有意义的分块 而且 `section` 内部必须有标题

`<div> </div>` 没有语义 用来表示一个区块 (目前 `div` 还是我们主要布局元素 可以替代上面的) 块

` ` 行内元素 无语义 一般用于网页中选中文字

`<center> </center>` 将文字居中

表单 `form`

表单 `<form action=" 提交到哪里去" method="提交方式"> <input type=" " value=" " name=" " placeholder=" " readonly/ > </form>` 给用户看 用户不用输入 可以给上 `value` 作为功能按钮名字

`placeholder` 作为给用户的提示信息, 输入则会消失 `readonly` 只能看不能修改

submit提交按钮 reset重置按钮 button属性 空按钮 <button> </button> 标签等同于 submit提交

fieldset>legend 表单字段集 只能有一个子集legend, <label for=""> </label> for值等于关联标签中的id值 label 标签中的 for 定义: for 属性规定 label 与哪个表单元素绑定。

type= "radio" 单选按钮 同组name必须保持一致, 保持互斥 checked= "checked" 给一个默认选项

type= "checkbox" 多选框 select>option 下拉列表 textarea多行文本输入框

resize:none; col= "宽度" rows= "高度" 禁止拖拽 type="file"文件上传格式

type="hidden"隐藏表单 (网页后台看不到) type= "image" 图片按钮 src= "地址"

label提示信息 for="关联标签中的id值"

hidden隐藏表单

```
value="0" id="Input1" hidden
```

- 新增type类型

- type="email" 限制用户必须输入email类型

- type="url" 限制用户必须输入url类型

- type="range" 产生一个滑动条表单

- type="number" 限制用户必须输入数字

- type="search" 产生一个搜索意义的表单

- type="color" 生成一个颜色选择的表单

- type="time" 限制用户必须输入时间类型

- type="month" 限制用户必须输入月类型

- type="week" 限制用户必须输入周类型

- type="datetime-local" 选取本地时间

- type="date" 日期

```
fieldset > legend 表单字段集
子集legend只有一个
legend的属性
align=""
left
center
right
justify
```

img标签可以用于向当前页面引入一个图片 是一个自结束标签 在div中插入img标签之后,默认状态下会把父元素高度撑大3~6像素

替换元素 介于块元素和行内元素之间具有俩种元素其他特点 src属性指的是外部图片的路径 (路径规则和超链接一样) alt属性必须要写, 空着也要写 alt属性是对图片的描述 (默认情况下不会显示, 有些浏览器会在图片无法加载时显示) 搜索引擎会根据alt属性描述的内容来识别图片

如果要引入其他网页的照片 可以直接复制该图片的网页路径

width宽度 height高度 单位像素 如果宽度高度只改了一个 高度宽度会等比例缩放 所以一般对于图片来说 只要宽度设置对了 高度会自动随比例变化

```

```

一般在pc端 不建议修改图片大小, 需要多大就裁多大 在移动端经常对图片进行 (大图缩小)

```

```

7.10

图片格式 jpg (颜色丰富, 不支持透明效果, 常用来照片) gif (颜色比较少, 支持简单透明, 支持动图) png (支持颜色丰富, 支持复杂透明, 不支持动态, 为网页而生) webp谷歌新推出的, 专门用来表示网页中图片的一种格式。兼具所有优点, 但是**兼容性不好** (一些老的浏览器运行出错)。效果如果一样用小的, 效果不同, 用好的 (也要看下大小)

base64 将图片使用base64进行编码, 可以将图片转化为字符, 通过字符形式引入图片。(需要图片加载速度快的时候可以用, 可以和网页同步加载) **使用得少** 百度上有在线转换器

```

```

内联框架 在当前网页嵌套另一个网页的页面 (作为一个窗口) <iframe src=""

frameborder="0/1"> </iframe> src属性指定的是内联的网站 frameborder 指定内联框架的边框 如果不写默认为1 有边框

<audio src=""> </audio> 或者可以用 <audio> <source src=""> </audio> 用source标签用来引入音频文件的 默认是不能自己控制的 加上controls属性 (无属性值) 可以控制 加上autoplay属性可以自动播放 但是目前来说大部分浏览器并不会自动对音乐进行播放 (怕影响用户体验) loop循环播放

```
<audio src="./北京东路的日子, 混音.mp3" controls autoplay></audio>
```

muted静音 poster第一帧图像 或者可以用 <audio> <source src=""> </audio> 用source来导路径 这样的好处是可以添加文字或者同时指定多个文件 (从上往下识别, 当第一种格式不能识别的时候自动识别第二种) 更换格式万一识别不了

```
<source src="video/3the811.mp4" />
<source src="video/mov_bbb11.mp4" type="video/mp4" />
<source src="video/movie12.ogg" type="video/ogg" />
<source src="video/movie1e.webm" type="video/webm" />
```

比如当 浏览器不支持的时候 (直接用 <embed src="" type=""> type内具体的格式比如 audio/mp3等 不兼容用embed)

```
<body>
  <!-- <audio controls autoplay>
    <source src="./北京东路的日子, 混音.mp3"> -->

  <embed src="北京东路的日子, 混音.mp3" type="" >
```

可以在版本低的浏览器引入成功 但是一般需要设置高度宽度)

source一般广泛一点

```
<audio controls autoplay>
  对不起, 您的浏览器版本过低, 请升级浏览器!
  <source src="./北京东路的日子, 混音.mp3">
</audio>
```

video使用方法跟上述一样 (除了Mp4格式之外 新出了个webm视频格式)

列表 html中一共有三种列表：有序 无序 定义

` `可以被用来表示列表项 无序列表 使用` `标签来创建 一个li表示一项 **ul是其中用的最多**

有序列表 用` ` 创建

- 我爱你
- 你爱我
- 1. 你爱我
- 2. 我爱你

用`<dl> </dl>`标签来定义列表 其中 dt 表示定义的内容（只能有一个） dd则对内容进行解释说明（有点像2级菜单那种，点击就会显示下面的东西

`<dl>`

`<dt1> </dt1>`

`<dd> </dd>`

`<dt2> </dt2>`

`<dd> </dd>`

`</dl>` 可以用来写那种悬浮出现的框 dt dd是并列的给个浮动 给定宽度 dl也要给宽度

什么是爱情

爱情是永恒的

超链接：`<a href> `标签可以用来添加超链接（超链接用来

跳转到其他页面或者跳转到当前页面的其他位置）并且属于行内元素 不独占一格 且a标签中可以嵌套除它本身之外的任何元素 属性：加href可以指定跳转的链接（值可以是一个外部网站的地址/也可以写一个内部页面的地址 必须在同一个大文件夹里面

```
<a href="02语义化标签.html">超链接</a>
```

若将href的属性值设置为# 则页面不会发生跳转 超链接会回到网页的顶部

```
<a href="#">回到顶部</a>
```

当我们需跳转一个服务器内部页面时，一般我们会使用相对路径 相对路径一般以./或者../开头 前者可以省略不写 如果 俩者都没写的话 就默认写了./ ./表示当前文件所在的目录 ../表示当前文件所在目录的上一级目录 一个../表示一级 可以加多个../ 就可以跳更多的级去找文件

```
<a href="../../../01语义化标签.html">超链接</a>
```

target属性 用来指定超链接打开的位置 1: _self 当前页面 默认的可以省略 2: _blank 新页面 打开

```
<a href="07.列表.html" target="_blank">超链接</a>
```

id属性 每一个标签可以添加一个id属性 是元素的唯一标识 且在同一页面中id属性不会重复

想跳转到啥位置时 就给那个位置加个id属性 然后 在一个地方加上a标签 给href属性值设定#id 然后超链接就会直接跳转到那个位置

```
<p id="sex">
```

javascript:; 可以作为href属性的属性值，用来表示占位，点该链接什么都不会发生

```
<a href="https://www.baidu.com">超链接1</a> <br>
<a href="./02语义化标签.html">超链接2</a><br>
<a href="http://www.4399.com" target="_blank">超链接3</a><br>
<a href="#sex">超链接4</a><br>
<a href="javascript:;">超链接5</a>
```

当我们需要在网页中书写某些特殊的字符，则需要使用html中的实体（转义字符）实体语法：
&实体的名字；

 空格 > 大于号 < 小于号 © 版权符号 w3school可以查询实体符号

<meta charset=" 字符集"> meta标签用来设置网页的元数据（底层的，不会改变的）
name 指定的数据名称

content 指定的数据的内容

加起来就是设置了一个元数据的名字 content就是指的里面的值 （可以用来搜索的名词）

再继续加其他属性> 一次一句 如果要同一句话不同颜色/大小 在要分开的地方再来一次..

在有属性值的条件下 双引号或者单引号

<!doctype html>HTML5文档声明

所有数据在计算机底层都会以二进制（满2进1）的形式保存 一般显示一个二进制数字时，都会转换为十六进制

内存是一个有很多小格子的容器 每个小格子可以存储一个0或者一个1
一个小格子 称为1bit

8bit = 1byte（字节）最小可操作单位

1024byte=1kb

编码和解码采用的规则称为字符集，如果采用的字符集不同就会出现乱码问题

UTF-8 万国码 通用字符集

CSS3 (表现)

css设置属性的时候，最后一个；不加也不影响效果

层叠样式表 网页是一个多层结构 每一层都可以设计样式 我们看到的只有最上边一层 设置网页元素样式

一：内联样式（行内样式）：**只能对一个标签生效** 如果需要影响多个还得全部复制 非常不方便 开发的时候绝对别使用 在**标签内部**通过style属性来设置元素的样式
color font-size (字体大小) background-color背景颜色

```
<p style="color: red;font-size: 50px ;" >锄禾日当午，汗滴禾下土</p>
```

二：内部样式表（将样式编写到head中的style标签里然后通过CSS选择器来选中元素并为其设置各种样式 **可以同时为多个标签设置样式**）只能对一个网页起作用 不能跨页面使用（采用css规则不同于html规则 比如注释就不一样）

```
<style>
p{color: red;
font-size: 30px;
}
</style>
```

三：外部样式表（将CSS样式编写到一个**外部CSS**文件中然后通过link标签来引入外部CSS文件 跨网页使用 在不同页面之间进行复用 还可以使用到浏览器的**缓存机制**加快网页的加载速度）最佳使用方式 <link rel="stylesheet" href=" " > href属性中加css文件路径

```
# style.css
```

```
1 p{
2   color: red;
3   font-size: 30px;
4 }
```

css中的注释/* */ CSS基本语法：1 选择器 选择页面中的指定元素 （选什么就写比如 p

h1

```
/*
p{
  color: red;
  font-size: 40px;
}
h1{
  color: green;
}
```

2 声明块 通过声明块来指定为元素设置的样式 由一个一个声明（是一个名值对结构 一个样式名对应一个样式值 名和值 之间以：连接，以；结尾 例如 color : red; ）组成

选择器

元素选择器 标签名{} p{} h1{}

id选择器 (id—**定不可以重复**) #id属性值{}

类选择器 (class选择器) **.class**属性值{} class是一个标签的属性 和id属性类似 不同的是class可以重复使用 可以通过class属性来为元素分组
可以给一个元素设置多个class 中间以空格隔开
class= "blue adc"

通配符选中页面中所有元素 ***{}** ***通配符** 匹配页面中的所有元素 *{
margin:0;padding:0;}样式初始化

复合选择器

1 交集选择器 选中同时符合多个条件的元素(后代)

语法：选择器1选择器2...{} div.red

```
div.red{  
  color: red;  
}
```

(如果交集选择器中有元素选择器。必须使用元素选择器开头)

2 并集选择器 同时选择多个选择器对应的元素(群组选择器)

语法：选择器1, 选择器2 比如 h1, span

```
h1,span{  
  color: green;  
}
```

父元素（直接包含子元素） 子元素（直接被父元素包含） 祖先元素（直接或间接包含） 后代元素（直接或间接被包含） 兄弟元素

子元素选择器 选中指定父元素的指定子元素 语法：父元素>子元素 1>子元素2（是子元素1的子元素） **可以一直更替下去**

后代元素选择器 选中指定元素内的**所有**指定后代元素 元素 空格元素格式

```
div span{  
  color: green ;
```

兄弟元素选择器 选择下一个兄弟 语法：前一个+ **下一个（后面最近的一个）**

前一个~后面所有的相同兄弟 p~

span(选择p元素的**后面所有**span兄弟元素)

属性选择器[]

属性不是杜撰的 [属性名] 选择含有指定属性的元素 [属性名~"属性值"] 包含该属性值(整体的) [属性名| "属性值"] 包含-破折号的能被选中 [属性名="属性值"] 选择含有指定属性和属性值的元素

[属性名^=属性值] 选择属性值以指定值开头的元素 [属性名\$=属性值] 选择属性值以指定值结尾的元素 [属性名*=属性值] 选择属性值含有某值的指定元素 例如设置个title (给附加个小标签 鼠标放上面就会显示) 属性

```
body>
<p title="我奥你">小小离家老大回</p>
<p title="我">乡音无改鬓毛衰</p>
<p title="12我2你">秋水共长天一色</p>
<p>落霞与孤鹜齐飞</p>
```

```
*[title*=我]{
```

```
p[title*=我]{
```

伪类 (不存在的类 用来描述一个元素的特殊状态 比如第一个子元素) 选择器 (一般情况下都是以一个冒号: 开头)

举例: 当你给同类元素设置属性的时候 当时又不想设置全部元素 可以使用伪类的子元素选择器

去除或者选中某个具体的元素来进行属性的设置

:first-child :last-child :nth-child()选中第n个**子元素** 特殊值n 表示0到正无穷 2n或者even为偶数 2n+1或者odd奇数 根据**所有子元素**进行排列:nth-last-child() 倒数 (如果是父元素为选择器, 去子元素中找则要加空格在前

```
ul :first-child{
```

, 如果直接是子元素中找则不用加空格

```
li:first-child
```

)

使用伪类选择器时, 一定要注意空格的使用, 加与不加空格结果会截然不同

使用如下伪类选择器 (即 :nth-child()、:first-child、:last-child、:only-child) 时, 冒号前必须加空格才是子元素选择器

X:first-of-type X:last-of-type X:nth-of-type()等等是**同类子元素中**排序 匹配父元素中的X元素

:not()将符合条件的元素从选择器中去除

:only-of-type

:only-child:only-child

匹配没有任何兄弟元素的元素

:empty 匹配里面没有任何子元素的 匹配空标签元素

:root根元素选择器

a元素（超链接）的伪类 :link没访问过（一般正常链接）

:visited访问过（由于隐私原因 visited这个伪类只能修改链接的颜色）

:hover（表示鼠标移入的状态）

:active表示鼠标点击

对于: hover 实现（当鼠标移动到一个元素A身上时，另外一个元素B显示。）css使用：`.father:hover .b { display: block }`的方式，来定义鼠标移动到父级身上时，B元素的样式。

伪元素（表示页面的一些特殊的并不真实存在的元素 **特殊的位置 比如第一个字母位置**）经常以::开头

::first-letter 第一个字母 ::first-line 第一行::selection 表示选中的内容（用鼠标选中 然后呈现出来的特征）

div元素 在第一个字前和最后一个字后都会有一个位置

而::before 元素的开始 可以用before来为li设置前面的序号形式 通过字体图标 link记得引入矢量图的地址

```
li:before{
  content: "\e62f";          /*用伪类使用*/
  font-family: "iconfont";
  font-size: 100px;
```

::after元素的最后 可以运用上 但是必须结合content 属性来使用.box1::after{

content: "" ; } 伪元素 这是通过css添加元素 默认是行内元素

: target伪元素

:target

:target **CSS 伪类** 代表一个唯一的页面元素(目标元素)，其 **id** 与当前URL片段匹配。

:target选择器用于选取页面中的某个target元素（说白了就是该元素的id被当做页面的超链接来使用）就是处于它的锚点被点击时候的一个状态

有一个超链接的**锚点** 然后通过:target 选中匹配这个id值的元素的id**被当作超链接来使用的状态** #id:target 关联后点击a相当于点击这个盒子 然后可以**控制子元素**或者**兄弟元素**的属性 #text1:target a{color:red;} or #text1:target div{display:block}等

```
<div id="text1">
  <a href="#text1">列表一</a>
  <div>我是你爸爸哈哈啊哈哈</div>
</div>
```

或者#box1:target~ul

```
<div id="box1"></div>
<div id="box2"></div>
<div id="box3"></div>
<a class="a1" href="#box1">1</a>
<a class="a2" href="#box2">2</a>
<a class="a3" href="#box3">3</a>
<ul>
  <li class="li1">one~点击切换</li>
  <li class="li2">two~点击切换</li>
  <li class="li3">three~点击切换</li>
</ul>
```

同理 通过input单选框/复选框 用label 关联上 然后通过伪类选择器: checked (表示任何处于选中状态的radio checkbox或select 中的option) () 因为需要实现点击之后才进行切换的效果, 此时将label指向相应的input标签的id (就是将) 来实现选择切换的效果。

cursor 规则是设定网页浏览时 用户 鼠标 指针的样式 (形状), pointer一只伸出食指的手 default箭头crosshair 十字, progress 箭头和沙漏 状态伪类选择器 enabled disabled :focus获取光标

```
UI状态伪类选择器:
:enabled{} 可用状态
:disabled{} 禁用状态
:checked{} 选中状态
:focus{} 获取焦点状态
```

(在输入法中打v1可以搜索特殊符号)

样式的继承: 我们为一个元素设置的样式同样也会应用到它的后代元素上 可将一些通用的样式统一设置到共同的祖先元素上, 这样只需要设置一次即可让所有元素都具有该样式 方便开发 重点: 并不是所有样式都会被继承 比如 背景相关的 (background-color) 和布局相关的

css属性继承

不可继承的: display、margin、border、padding、background、height、min-height、max-height、min-width、max-width、overflow、position、left、right、top、bottom、z-index、float、clear、table-layout、vertical-align

所有元素可继承: visibility和cursor。

内联元素可继承: letter-spacing、word-spacing、line-height、color、font、font-family、font-size、font-style、font-variant、font-weight、text-decoration、text-transform。

块状元素可继承: text-indent和text-align

列表元素可继承: list-style、list-style-type、list-style-position、list-style-image。

表格元素可继承: border-collapse。

txt

选择器的权重（针对同一个元素标签，当设置了不同的选择器时，要判断权值的大小，权值越大优先级越高）：内联样式（1, 0, 0, 0）> id选择器（0, 1, 0, 0）>类和伪类选择器（0, 0, 1, 0）>元素选择器（0, 0, 0, 1） 通配选择器（*）优先级为0 继承的样式没有优先级（比0还低） 而且比较优先级的时候需要把**所有优先级相加**（累加不会超过最大的数量级）最后优先级越高，则越优先显示（分组选择器是单独计算的） 优先级计算后相同的话，优先使用靠下的样式

```
css层叠性：
1: 指的是样式优先级,产生冲突时,优先级高的干掉优先级低的
2: 内联样式>内部样式>外部样式
3: !important 权重最高
4: id选择器 > class选择器 > 元素选择器
5: 权重相同时,最后写的干掉先写的（顺序要求）
```

```
#box1{
    background-color: green;
}
div#box1{
    background-color: red;
}
```

可以在样式后添加！important 改变优先级 甚至可以超过内联样式

```
#box1{
    background-color: green !important;
}
```

文本属性：

长度单位：像素

百分比 也可以将属性值设置为相对于其父元素属性的百分比（可以使子元素跟随父元素改变而改变）

em(相对于字体大小来表现的，字体大小改变它也改变) 1em=1font-size(被设置的标签元素的最近的父级元素的font-size) (em会根据字体大小改变而改变) font-size (字体大小 最小为12px，往下就不会缩小了) 默认为16px rem相对于根元素字体大小来计算

```
<style>
    .box1{
        height: 200px;
        width: 200px;
        background-color: gold;
    }
    .box2{
        height: 50%;
        width: 50%;
        background-color: pink;
    }
</style>
<div class="box1">
    <div class="box2"></div>
```

```
<style>
  html{
    font-size: 50px;
  }

```

颜色单位

1.直接用颜色名 (不太方便) 2.使用RGB (red, green, blue) 通过三种颜色不同浓度调配不同的范围 0~255 (0%~100%) 颜色 RGBA 在rgb基础上增加了一个透明度0 (完全透明) ~1 (不透明)

十六进制的颜色单位#红色蓝色绿色 颜色浓度: 00-ff 颜色俩位俩位重复可以简写#aabbcc-#abc

HSL, HSLA值 H色相 (0~360 一个圆 度数)S饱和度(0%~100% 颜色的浓度)L亮度 (0%~100% 颜色的亮度) hsla(色相, 饱和度, 亮度, 透明度)

font-family:字体1, 字体2, 字体3; 优先显示第一个字体, 若电脑上没有, 则会继续往下去编译

中文表示字体需要添加引号, 单个单词表示不需要加引号, 多个单词表示需要添加引号

文本属性

font-weight:文本加粗 bold(加粗的视觉效果) or 100~300(变细) 400(等价于normal)500(不加粗) 600~900(加粗) or normal(常规文字) or bolder(粗体字)

font-style:文本倾斜 italic斜体字 oblique(倾斜的视觉效果) normal(常规文本)

line-height:行高 文本垂直对齐方式 文字实际占有高度 可以直接写整数 不带单位 则时字体的指定倍数 行高距等于行高减去字体大小

字体框
- 字体框就是字体存在的格子, 设置font-size实际上就是在设置字体框的高度
行高会在字体框的上下平均分配

line-height:行高; 文本垂直对齐方式
行高值 < 高度值 偏上
行高值 = 高度值 垂直居中
行高值 > 高度值 偏下
【注】没有负数, 可以超出, 只针对单行文本有效

测量距离 文字顶部的距离

font综合写法 font: 倾斜 加粗 文本大小/行高 "字体" 倾斜加粗不能放后面写 最后必须是大小和字体 行高如果前面设置了

然后后面综合写法没写的话相当于使用默认值了, 会把之前写的值覆盖

例如: font:italic bold 50px/200px "宋体"

text-align水平对齐：left or right or center or **justify** (两段对齐)

text-decoration: underline(下划线) or none (没有线) or overline (上划线) or line-through (中划线 删除线)

text-indent:首行缩进，只针对第一行有效，可以为负 一般单位为em可以较清楚的展示

list-style:none (取消列表前面的排序)、

css层叠

背景相关属性

引用图片background-image:url("图片地址"),url("");可以引入多个

【拓展】多张背景图填充

```
background-image:url(),url(),...url() 颜色;
```

background-repeat:repeat or no-repeat(不重复) or (repeat-y 纵向重复)

{ 和img标签元素的区别 **不占位置**

背景图 与 img图的区别

- 1: 都是往页面中插入图片
- 2: 前者是css样式,后者是标签结构
- 3: 前者不占位,后者占位

}

background-position:背景定位 (水平 left **center** right 垂直 top **center** bottom) 也可以直接用具体的值 background-position:水平方向位置 垂直方向位置(水平,垂直);**如果同时引入多个图片的情况** 如果只写了一个方向 (水平或者垂直) 则垂直/水平 方向默认为center

background-attachment:fixed(固定在浏览器窗口上) /scroll(滚动查看 设置一个滚动按钮 双向都设置)

(当垂直方向一直设置为center的时候, 然后给一个fixed 固定 当你浏览器窗口缩小时 图片会一直处于浏览器窗口中间)

background-size: 宽度 高度; 或者 background-size: 值; 设置一个值的时候 宽高等比例缩放(这个值代表宽度) contain(哪条边在放大过程中先碰到 就停止扩大了)

```
background-size:背景图大小;  
px %  
cover 填满,肯超出  
contain 优先填满满某一边,另一边有可能空白
```

) cover (使图片始终填满 **可能会溢出**)

综合写法: background: 颜色 url("") 是否重复 定位 (position) /大小 (size) 是否固定

background综合写法: background:url(" 图片地址") 背景颜色

position(方位)/size(大小) 是否固定

顺序暂时没有规定

background-origin:背景图起始位置


```
background-origin:背景图起始位置;
padding-box      默认值
content-box      内容区域
border-box       边框区域
```

background-clip: 背景图裁剪区域

```
background-clip:背景图裁剪区域;
border-box      默认值
padding-box     内边距区域
content-box     内容区域
```

精灵图片：其实就是通过将多个图片融合到一张图里面，然后通过CSS background背景定位技术技巧布局网页背景。要哪张图像就可以从大图中只表现出这一种图像 调整position

文档流

浮动

特点：

浮动元素默认不会从父元素中溢出

浮动元素不会盖住文字 文字会自动环绕在浮动元素的周围 可以利用该特点来设置文字环绕图片的

脱离文档流后的特点

浮动元素大小由内容撑起来

```
元素设置浮动以后，将会从文档流中脱离，从文档流中脱离后，元素的一些特点也会发
脱离文档流的特点：
  块元素：
    1、块元素不在独占页面的一行
    2、脱离文档流以后，块元素的宽度和高度默认都被内容撑开
  行内元素：
    行内元素脱离文档流以后会变成块元素，特点和块元素一样
脱离文档流以后，不需要再区分块和行内了
```

效果

```
浮动：让竖着排列的元素横着排列
float:;
left
right |
none
影响：元素将脱离文档流（其实就是脱离这种排列顺序）
【注】文档流指的是 普通文档流， 从上至下， 从左至右的排列顺序就是普通文档流
```

```
我是div1
我是div2
我是span1 我是span2 我是span2 我是span2 我是span2 我是span2 我是span2
我是span2 我是span2 我是span2
```

盒子模型（盒模型，框模型）

css将页面中的所有元素都设置为了一个矩形的盒子，网页布局就是“将盒子摆到特定的位置”

盒子由以下部分组成：内容区（content 盒子内部装东西的空间） 内边距(padding 内容区和边框的距离) 边框(border) 外边距 (margin指定当前盒子和其他盒子的距离的，盒子和盒子之间的距离)

content：元素中所有的子元素和文件内容都在内容区当中排列 内容区的大小由width和height设置

```
.box1{
  width: 200px;
  height: 200px; /*设置内容区content的大小*/
}
```

border(border边框也会影响盒子的大小并不仅仅是内容区大小决定)



设置边框一定需要border-width(可以不写，是因为不写的话会有一个默认值大小 一般为3px)

border-color border-style三个样式 border-width还有一组border-xxx-width(top right bottom left 然后其他方位的如果没设置就会是默认的值)

```
/*
  边框 (border)，边框属于盒子边缘，边框里边属于盒子内部，出了边框都是盒子的外部
  边框的大小会影响到整个盒子的大小
  要设置边框，需要至少设置三个样式：
  边框的宽度 border-width
  边框的颜色 border-color
  边框的样式 border-style
*/
```

border-width如果写4个值 上 右 下 左 依次运用 3个值 上 左右 下 2个值 上下 左右。border-color也可以四个方向设置四个颜色（如果省略，则会自动使用color的颜色 默认是黑色）

border-style (solid实线,dotted点状虚线,dashed虚线,double双线，也可四个方向不同)

```
border-width: 10px;
border-color: red;
border-style: solid;
```

border简写属性，可同时设置所有边框相关样式，而且没有顺序比如border: solid 10px red; 也可四个方向单独设置，如果那边不想要可以单独给那边设置个none 比如border-right:none;则右边框变没有

border-image :url() 偏移量 填充方式

重复性 border-image-repeat:round会凑整填充（进行了适度的拉伸）。repeat 不进行拉伸，不凑整

```
边框模式：
border-image:url() 垂直偏移量 水平偏移量 填充方式;
border-image-outset:往外扩张倍数;
```

(利用盒子产生三角形 把盒子height width都设为0 给border的时候把颜色设置为透明 transparent or rgba设置为0 再给一个方向加上颜色就好了)

```
.box1{
  width:0;
  height:0;
  border:10px solid transparent;
  border-bottom-color:yellow;
}
```



盒模型占位大小计算：

```
盒模型大小计算:<br>
w = width + 左右内边距 + 左右边框 + 左右外边距<br>
h = height + 上下内边距 + 上下边框 + 上下外边距
```

padding(内边距)

举例padding-top:100px 红色部分则为添加padding扩大的区域 padding不能设置负值



外边距 (margin) 不影响可见框大小，会影响位置 (影响盒子**实际占用空间**)

```
外边距 (margin)
- 外边距不会影响盒子可见框的大小
- 但是外边距会影响盒子的位置
- 一共有四个方向的外边距：
```

margin-top和margin-left(设置一个正值，元素会向下向右移动)是移动自己 margin-right和margin-bottom(设置一个正值，**其下边的元素**会向下移动)是移动其他元素 margin也可以设置负值，会向**相反方向**移动

水平方向的布局：

```
元素的水平方向的布局：
元素在其父元素中水平方向的位置由以下几个属性共同决定“
margin-left
border-left
padding-left
width
padding-right
border-right
margin-right
```

```
一个元素在其父元素中，水平布局必须要满足以下的等式
margin-left+border-left+padding-left+width+padding-right+border-right+margin-right = 其父元素内容区的宽度 (必须满足)
```

以上等式**必须满足**，如果相加结果使等式不成立，则成为过度约束，则等式会自动调整

调整的情况：

以下三个可以设置为auto:width, margin-left, margin-right

如果这七个值没有为auto的情况，则浏览器会自动调整margin-right (默认调整这个 以因为是从左往右书写 默认调整最后一个) 等式成立

```
- 如果这七个值中没有为 auto 的情况，则浏览器会自动调整margin-right值以使等式满足
- 这七个值中有三个值和设置为auto
```

如果某个值为auto，则会自动调整为auto那个值以使等式成立

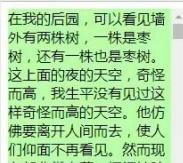
把不同属性设置为auto的情况如下：

- 如果将一个宽度和一个外边距设置为auto，则宽度会调整到最大，设置为auto的外边距会自动为0
 - 如果将三个值都设置为auto，则外边距都是0，宽度最大
 - 如果将两个外边距设置为auto，宽度固定值，则会将外边距设置为相同的值
- 所以我们经常利用这个特点来使一个元素在其父元素中水平居中

经常用第三个特点来使一个元素在其父元素中水平居中

垂直方向的布局：、

默认情况下，如果不给父元素设置height，父元素的高度是被内容撑开，如果子元素大小超过父元素，则子元素会溢出。可用overflow（-x/-y可分别处理水平或者垂直）属性来设置父元素如何处理溢出的子元素（文字的



可选值：

- visible, 默认值 子元素会从父元素中溢出，在父元素外部的的位置显示
- hidden 溢出内容将会被裁剪不会显示
- scroll 生成两个滚动条，通过滚动条来查看完整的内容
- auto 根据需要生成滚动条

)

white-space 文本换行方式（nowrap属性 强制不换行 但是遇到br还是要换行）

相邻的垂直方向外边距的重叠

text-overflow: ellipsis; 省略号（单独设置不会显示）

一般写省略号的方式是（主要针对单行文本）：width 设置一个宽度 overflow:hidden

white-space:nowrap; text-overflow:ellipsis;四部曲

兄弟元素（一般不需要处理）

垂直外边距的重叠（折叠）

- 相邻的垂直方向外边距会发生重叠现象
- 兄弟元素
 - 兄弟元素间的相邻垂直外边距会取两者之间的较大值（两者都是正值）
 - 特殊情况：
 - 如果相邻的外边距一正一负，则取两者的和
 - 如果相邻的外边距都是负值，则取两者中绝对值较大的
- 兄弟元素之间的外边距的重叠，对于开发是有利的，所以我们不需要进行处理

（父子元素间相邻外边距，子元素的会传递给父元素（上外边距）。如果给父元素加个边框就可以阻止 这是一个bug性问题 但是一般不会让它们相遇 可以给父元素加padding 不让它们相邻 再减少父元素的高度 或者让父元素触发BFC）

b:如果父元素和第一个子元素没有浮动的情况下，给第一个子元素添加margin-top,会错误放在父元素上面。

行内元素的盒模型：不支持设置高度及宽度 而且垂直方向的（border padding margin不会影响页面布局） 水平margin会相加 不会重叠

display属性可以设置元素显示的类型 inline 设为行内元素 block设为块元素

inline-block设置行内块元素 当成一个字（img标签和input是行内块元素）（既可以设置宽度和高度 又不会独占一行 还可以**正常设置内外边距**） table元素以表格形式显示

none元素不在页面中显示（**位置也不占据** 后面的元素会补充上去）（需要的时候可以利用特殊手段使它展示 比如鼠标悬停上去） 可以用伪类:hover悬停上使它内容展示

visibility: visible（默认值 正常显示） hidden（元素在页面中隐藏，但是依然占据位置）

浏览器的默认样式表 `: *{ margin: 0; padding: 0; }`重置样式表()

重置样式表：专门用来对浏览器的样式进行重置的
`reset.css` 直接去除了浏览器的默认样式
`normalize.css` 对默认样式进行了统一

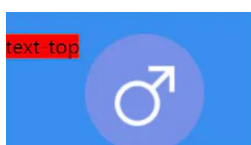
normalize 统一每个浏览器的默认样式

注释记得写

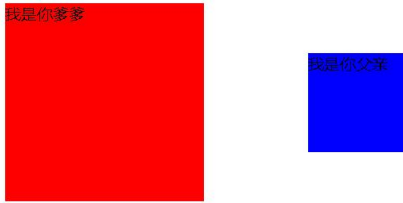
vertical-align：元素垂直对齐方式 top middle bottom（只有行内元素有效）

具体长度值 1 是正值基线就向上移动，如果是负值基线向下移动。2百分比值一样 但是是相对于行高值的百分比

- sub：元素的基线向下移
- super：元素的基线向上移
- top：（内联）元素的顶边和行内**最高元素的顶边**对齐 最高元素的顶边包括margin 不是和行的顶边对齐。因为设置了行的padding-top之后元素并没有顶在最上面
- bottom：元素的底边和行的底边对齐。
- middle：元素上下边的中心点和行基线向上1/2_x的高度位置对齐。
- text-top：元素顶边和父级的**内容区域顶边**对齐



- text-bottom: 元素底部和父级的内容区域底部对齐



标尺 用在图片上用的多 图片和文字 用来设置垂直对齐方式

```
标尺:
width:0;
height:100%;
display:inline-block;
vertical-align:middle;
```

置换元素(内容可变, 虽然改变了 但是浏览器可能不知道 比如 引用图片 地址不变 但是换了一张图): img input

非置换元素: 内容确定了

举例: 当要给列表ul中的li设置属性

 只需要给前两个设置, 不用设置最后一个的时候 一: 可以给前两个li设置类

二: 用伪类 ul>li:nth-child(3))

绝对定位 position:absolute 会改变元素的类型 将行内元素转变为行内块元素

第一种:

绝对定位 position:absolute;

- 1、相对于已经定位的父级元素去定位
- 2、如果父级没有定位, 就去找HTML根文档去定位
- 3、完全脱离文档流、不占位置
- 4、通过left、right、top、bottom去定位

相对定位 position:relative 不会改变元素的类型

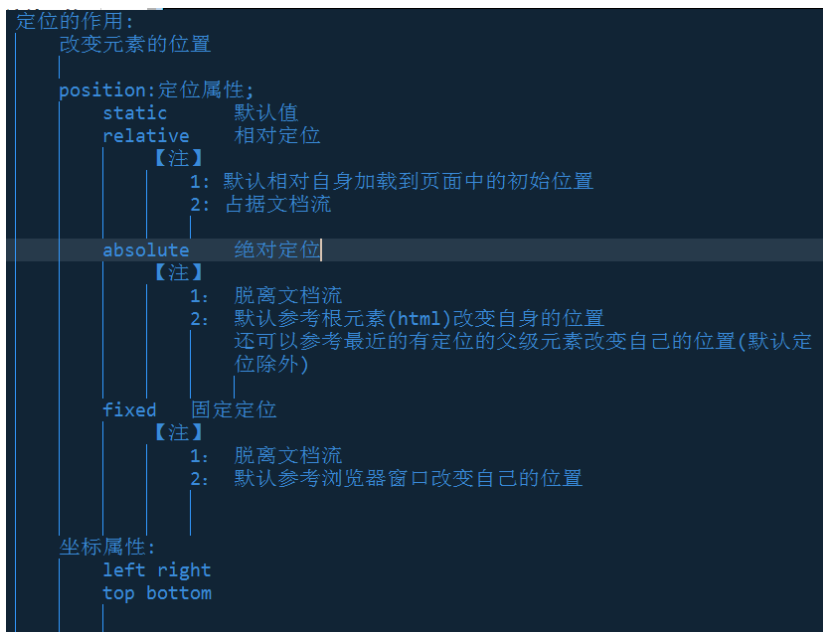
1 相对于自身的默认加载到页面的位置定位 2不脱离文档流 占有位置

与absolute的区别 1 参照物不同 2 相对定位不会对布局产生影响 3 主要作用是给绝对定位当爹使用 当参照物使用 (成为一个包含块 父元素使用relative 子元素使用absolute 默认值static 没用)

positive:static默认状态 设置与不设置一样 无区别)

fixed 固定定位 默认参考浏览器窗口改变自己位置 脱离文档流()

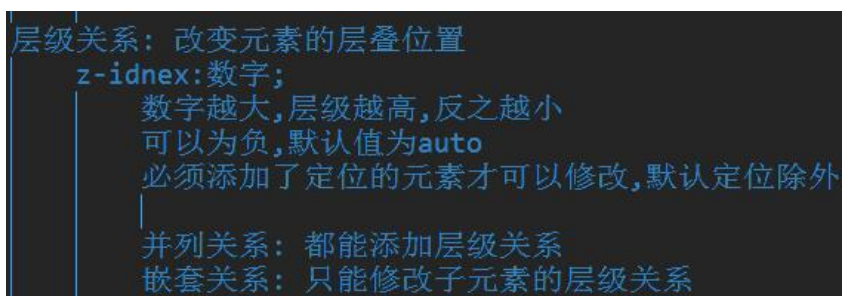
sticky粘性定位 fixed和relative的结合体



元素层级问题：z-index:value (static不算) 数字大层级高 可以为负数 默认为auto

必须添加定位才能给z-index 然后修改

分别情况：并列（都可以添加z-index 以此比较层级） 嵌套（只能修改子元素的层级关系 如果给父元素添加的话则效果不会展示出来）



盒子大小问题：默认情况下（可以更改）：盒子可见框大小由内容区 内边距 边框三者共同决定，需要加到一起计算

box-sizing 用来设置盒子尺寸的计算方式(设置width和height的作用)

可选值 content-box (标准盒模型)默认值 border-box(怪异盒模型) 宽度和高度用来设置整个盒子可见框的大小（设置完后 width和height指的就是内容区 内边距 边框的总大小）

outline设置元素的轮廓线 用法和border一样（不同点 轮廓不会影响到可见框的大小 不会影响页面的布局）

text-shadow文本阴影

box-shadow(水平, 垂直. 模糊度, 阴影大小, 颜色) 设置元素的阴影效果 阴影也不会影响页面布局（默认的话 就是只给它设置颜色 的话） inset属性值变成内部阴影

则阴影在元素的正下方 宽度和高度一样 不会有效果展示） 可以让双边都产生阴影（给两个）

```
box-shadow: 5px 0 5px #ebebeb, -5px 0 5px #ebebeb;
```

```
/* box-shadow 用来设置元素的阴影效果，阴影不会影响页面布局
   第一个值 水平偏移量 设置阴影的水平位置 正值向右移动 负值
   第二个值 垂直偏移量 设置阴影的水平位置 正值向下移动 负值
   第三个值 阴影的模糊半径
   第四个值 阴影的颜色
```

负值往相反方向移动，第三个值来设置阴影的模糊半径（越大效果越好）颜色一般用rgba来设置 来设置透明度

border-radius 四个方向 top-left bottom-left(该属性用来设置边框圆角 设置圆角的半径大小值)

border-bottom-right-radius:x y; 如果水平或者垂直方向上的半径不相等的话 即x不等于y 则会设置椭圆

一般可直接用border-radius同时设置四个方向

等比例缩放 img可以撑起来元素的高度 img自带3像素间距

元素在浏览器窗口居中

让一个元素始终咋子浏览器窗口水平，垂直位置居中：

position:fixed; left:0;right:0;top:0;bottom:0; margin:auto;

或者 position:fixed; left:50%;top:50%; margin-left:-宽度的一半; margin-top:-高度的一半;

问题：为啥分别用这两种方法居中的盒子不重叠呢？

制作锚点，给标签设置id属性

然后再给个超链接 超链接的href地址填那个id属性 然后点击超链接 就会直接跳转到那个标签上（在很多网页中结合 position:fixed;固定在浏览器窗口上 成为导航条）



BFC :是一个块级格式化上下文（块级格式化环境），是css中一个特殊的隐含属性，可以为一个元素开启BFC，开启后该元素会变成一个独立的布局区域 简单理解就是一个特殊的容器 不会影响外面，外面也不会影响到里面

只有块元素参与，一个 渲染区域

是要触发才能成为的一个状态 可以解决与浮动元素的重叠(因为BFC的区域不会和浮动元素的区域重合), 可以解决垂直方向子元素的margin带着父元素一起的问题 让父元素触发BFC

开启BFC状态后的特点: 开启后分内外

1 开启后的元素不会被浮动元素所覆盖 (**元素并列, 给没浮动的开启**) 2开启后的元素子元素和父元素外边距不会重叠3开启BFC的元素可以包含浮动的子元素 (**父元素高度由子元素撑起, 子元素浮动, 父元素高度塌陷, 给父元素开启BFC**)

可以通过一些特殊方式开启BFC (会有副作用, 找副作用小的) 某些方法如下:

1设置浮动 (除了默认值 none) 2设置为行内块元素 3 将元素的overflow设置为一个非默认值visible的值 (hidden,scroll) 常用

浮动导致的 脱离文档流 高度塌陷问题:

高度塌陷的问题:
在浮动布局中, 父元素的高度默认是被子元素撑开的,
当子元素浮动后, 其会完全脱离文档流, 子元素从文档流中脱离
将会无法撑起父元素的高度, 导致父元素的高度丢失

父元素高度丢失以后, 其下的元素会自动上移, 导致页面的布局混乱

所以高度塌陷是浮动布局中比较常见的一个问题, 这个问题我们必须要进行处理

版心{margin:0 auto;width: 内容范围宽度;} 写一次 多次用

要版心俩边是白色, 颜色不同就直接给footer这种给版心,

要颜色一样 给个新的子元素给版心

`text-transform:uppercase;` 定义只有大写字母 `lowercase`定义只有小写字母

`font-variant:small-caps; /*小型大写字母*/`

写一次 多次用,

父元素的高度一般不固定, 随着内容多少而变化 (自适应)

clear属性可以清除浮动元素对**当前元素**产生的影响 可选值 left清除左侧浮动影响 right 清除右侧浮动影响 both清除俩侧中**最大影响**的那侧 原理: 设置后, **浏览器会自动为当前元素添加一个margin-top** 以使其位置不受其他元素的影响 但是在开发者工具中看不到

利用::after伪元素解决高度塌陷问题 ::after 可以在元素的结束位置 添加一个子元素 必须结合 content 属性来使用.box1::after{

content: "" ; } 伪元素 这是通过css添加元素 默认是行内元素 原理：给他设置为块元素 然后用clear清除浮动对它的影响 让它回到它的位置撑起父元素

利用::before解决父子元素垂直外边距重叠：在元素的开始处添加一个元素 (要给字才行才能隔开 或则和设置为inline-block但是会多出来一块) display:

table最好

```
.box1::before{
  content: '';
  display: table;
}
```

```
/* clearfix 这个样式可以同时解决高度塌陷和外边距重叠的问题，
   .clearfix::before,
   .clearfix::after{
     content: '';
     display: table;
     clear: both;
   }
   I
```

clearfix类可以同时解决这2个问题 可以当作public.css里面的公共样式

写网页的时候先别着急写 看看布局 有些可以给的大盒子比如宽可以固定先写好 left布局还有 right布局

min-height: max-width: 最小高度 最大宽度自适应

min-width 一般可以给body 值为版心的宽度 防止缩小浏览器造成塌陷

网页导航

写网页导航栏的时候 用fixed定位 然后给定位确定要出来的框的位置 先把它位置放出去让看不到然后给个悬浮移动 加个过渡 transition就可以出效果

图标字体

```
fontawesome 使用步骤
1. 下载 https://fontawesome.com/
2. 解压
3. 将css和webfonts移动到项目中
4. 将all.css引入到网页中
5. 使用图标字体
   - 直接通过类名来使用图标字体
```

iconfont 阿里 字体库 在link引入地址 然后用类名或者实体都可以引入 但是记得给上iconfont类 伪元素也可以引入

针对的顺序

渐进增强

渐进增强 progressive enhancement：针对低版本浏览器进行构建页面，保证最基本的功能，然后再针对高级浏览器进行效果、交互等改进和追加功能达到更好的用户体验
从小到大

优雅降级

优雅降级 Graceful Degradation：一开始就构建站点的完整功能，然后针对浏览器测试和修复。比如一开始使用 CSS3 的特性构建了一个应用，然后逐步针对各大浏览器进行 hack 使其可以在低版本浏览器上正常浏览。

从大到小

区别：优雅降级是从复杂的现状开始，并试图减少用户体验的供给，而渐进增强则是从一个非常基础的，能够起作用的版本开始，并不断扩充，以适应未来环境的需要。降级（功能衰减）意味着往回看；而渐进增强则意味着朝前看，同时保证其根基处于安全地带。

移动端

媒体查询 为不同尺寸的屏幕设置不同的css样式 (MediaQuery) @media all(设备) and (min-width:320px) {

.box1{

} min-width从小到大 关键字only not 设备范围：默认样式写在最前面

```
/* 横屏 */
@media screen and (min-width: 320px) and (max-width: 480px) {}
/* 平板之类的宽度 1024 以下设备 */
@media only screen and (min-width: 321px) and (max-width: 1024px) {}
/* PC客户端或大屏幕设备: 1028px 至更大 */
@media only screen and (min-width: 1029px) {}
/* 竖屏 */
```



```
/* 竖屏 */
@media screen and (orientation:portrait) {对应样式}
/* 横屏 */
@media screen and (orientation:landscape){对应样式}
```

media_type	设备类型说明
all	所有设备
aural	听觉设备
braille	点字触觉设备
handheld	便携设备，如手机、平板电脑
print	打印输出设备
projection	投影设备
screen	显示器、笔记本、移动设备等
tey	盲打字机或盲码设备等
tv	电视机等设备类型
unbounded	自定义打印机

像素分为物理像素和css像素 默认情况下在pc端1个css像素等于1个物理像素

视口 默认1920px(css像素)屏幕中用来显示网页的区域 可以通过改变视口的大小（浏览器进行缩放），来改变css像素和物理像素的比值 **DPR设备像素比（要注意）=设备像素（物理像素）/设备独立像素（逻辑像素css像素）**

视口 (viewport)

- 视口就是屏幕中用来显示网页的区域
- 可以通过查看视口的大小, 来观察css像素和物理像素的比值
- 默认情况下:
 - 视口宽度 1920px (CSS像素)
 - 1920px (物理像素)
 - 此时, css像素和物理像素的比是 1:1

移动端默认的视口大小是980px(css像素),
默认情况下, 移动端的像素比就是 980/移动端宽度 (980/750)

编写移动端页面时确保有一个合理的像素比 完美视口 flex布局为多

自适应 user-scalable = no 属性能够解决 iPad 切换横屏之后触摸才能回到具体尺寸的问题。

`<meta name="viewport" content="width=device-width, initial-scale=1, minimum-scale=1, maximum-scale=1, user-scalable=no">` width=device-width等于当前设备宽度
initial-scale初始缩放比例minimum-scale允许用户缩放最小比例 maximum-scale允许用户缩放最大比例 默认值均为1

`<meta name="viewport" content="width=device-width, initial-scale=1.0">`

1css像素对应2或3个物理像素

Retina屏

DPR设备像素比=设备像素 (物理像素) /设备独立像素 (逻辑像素css像素)

DEVICE METRICS网址屏幕尺寸大全

每一款移动设备设计时, 都会有一个最佳的像素比。
一般我们只需要将像素比设置为该值即可得到一个最佳效果
将像素比设置为最佳像素比的视口大小我们称其为完美视口

一般设计图的长度都是375px的倍数比如(750px = 100vw) 需要考虑Retina屏幕 vw单位 ()

100vw=一个视口宽度(满屏) 把px转化为vw vh单位跟随屏幕高度变化 vmin谁小谁影响
vmax谁大谁影响

rem可用来做中介, 但是

网页中字体大小最小是12px, 不能设置一个比12像素还小的字体
如果我们设置了一个小于12px的字体, 则字体自动设置为12

需要转换一下

假如视口宽度为750px dpr=2 然后换算 $750/2=375$ css像素 (可以视口宽度或者css像素来换算) $375px=100vw$ $1vw=3.75px$

设置一个值1rem= X px 然后将1rem与vw的转换关系写出来 当作html的font-size

为了计算方便 让根标签大小为个整数

如果行内元素莫名其妙往下掉 解决方法: 1给这个元素添加display:block 2给他的父元素添加font-size: 0; 3 可能是换行问题 识别成了一个空格文字 (但是根元素font-size大 导致这个空格大)

写移动端两个方法:

第二种方法不需要转化了, 直接小数点往前移两位 缩小100倍


```

方式一：
html{font-size:32vw;}
html{font-size:37.5vw;}
所有的px数值需要利用工具转换成rem

方式二：
添加插件
<script type="text/javascript" src="js/flexible.js"></script>

```

css3渐变

线性渐变 background-image:linear-gradient(direction,color1 x,color2 y) x之前为color1 x-y之间为渐变 y之后为color2 direction (to top 从下至上 也可以使用deg角度 逆时针方向计算-) x<y 单位可为px可为%

可以为内容底部设置白色透明：怎么让文本框后一部分显得虚幻

给一个盒子加定位给上渐变background: linear-gradient(to bottom, rgba(255,255,255,0) 0,#fff 100%);

看看几位收藏“大咖”的故事，如果你也想要在明年此

次活动也欢迎所有创意达人加入到这场盛会中来。只要

repeating-linear-gradient 重复线性渐变

径向渐变 radial-gradient(圆心坐标, 半径, 形状, 颜色1起始位置, 颜色2结束位置)

background: radial-gradient(center, shape, size, start-color, ..., last-color); 当有前缀-

webkit-时候第一个值才为圆心坐标的值 不然就为半径的值

圆心坐标: px % 半径: px % center: 渐变起点的位置, 可以为百分比, 默认是图形的正中心。

size: 渐变的大小, 即渐变到哪里停止, 它有四个值。closest-side: 最近边; farthest-side: 最远边; closest-corner: 最近角; farthest-corner: 最远角。

...

shape: ellipse 椭圆 (元素本身是正方形 就不可能出现椭圆) circle 正圆

repeating-radial-gradient 重复径向渐变

过渡：过渡可以为一个元素在不同状态之间切换的时候定义不同的过渡效果。从一个有效数值向另一个有效数值进行过度: transition -property 规定参与的属性 -

duration 需要指定多少秒或毫秒才能完成 property和duration可以分别指定不同属性及不同属性的变化时间（逗号隔开） delay延迟 如果写延迟时间的话 则第二个时间值为delay时间值
-timing-function:过渡速度（linear ease ease-in ease-out ease-in-out steps（分步执行过渡效果

可以设置一个第二个值：
end . 在时间结束时执行过渡（默认值）
start . 在时间开始时执行过渡

) 贝塞尔曲线)

```
transition-timing-function:过渡速度;  
linear 匀速运动  
ease 加速运动  
ease-in 变慢  
ease-out 变快  
ease-in-out 先慢后快  
贝塞尔曲线
```

-timing-function 指定transition效果的转速曲线 -delay定义开始的时候（延迟）

```
transition-property:height;  
transition-duration:0.5s;  
transition-timing-function:linear;  
transition:all 1s;
```

2D功能函数

2D位移 transform变形（元素的形态变换） 属性 transform: 功能函数：

位移 translateX(),translateY(); 还有综合写法 translate(X,Y) 写一块就好 可以为负数

3d位移 translate3D(tx,ty,tz) tx ty 为x y轴上的位移量

缩放 scale(X,Y) scaleX() scaleY() 可以为负数，默认值为1 负数的时候 先缩小到0再变大

3d缩放 scale3D ()

旋转rotate()默认是z轴 一般在2d, 用x和y轴的 单位是deg x轴表现为高度减少
y轴表现为宽度减少rotateY()

skew 倾斜 skew (52deg) 默认为x轴 skew (x,y) skewX() skewY() 可以为负数

拓展

变形原点 transform-origin:水平left center right;垂直 top center bottom;

先旋转（改变位移方向）后位移 先位移后旋转 俩者的效果不一样

【拓展】
先旋转，后位移 和先位移，后旋转 效果是否一样
答：效果不一样，因为先旋转改变了位移的方向

不透明度opacity: 0~1; opacity属性指定了一个元素后面的背景的被覆盖程度。

是把这个元素（包括它的内容）当成一个整体看待，

透明 filter:alpha(opacity:0);0~100 100倍

其他的不能穿插到它们中间写

浏览器兼容前缀：
-webkit- 谷歌
-moz- 火狐
-ms- ie
-o- 欧朋
【注】不带兼容前缀的在最前或者最后写

3D

属性：perspective（元素距离 视线的距离）：景深 近大远小（**给父元素设置 并且它会影响到后代身上 所以给一次就好**）

transform-perspective（景深）；近大远小（给子元素设置）二选一同时会有冲突一般给父元素

step-start 直接从一帧跳到下一帧

transform-style:是否为3D环境；flat默认值 preserve-3d 提供3d环境 只能影响自己的子元素 不能隔代影响

观察3d元素的（位置）角度 perspective-origin（）观察角度可以给具体的值 或者left top 啥的

3D位移主要translateZ()和translate3d(tx,ty,tz) 两个功能函数 此值不能是一个百分比值，如果取值为百分比值，将会认为无效值

3D旋转 主要包括rotateX()、rotateY()、rotateZ()和rotate3d()四个功能函数；对于前三个**正值为顺时针**，

rotate3d(x,y,z,a)(建议取值0或1) a：是一个角度值，主要用来指定元素在3D空间旋转的角度，如果其值为正值，元素顺时针旋转，反之元素逆时针旋转。

3D缩放主要包括scaleZ()和scale3d()两个功能函数 默认值为1

css计算模式

calc(px - %)低版本不兼容 计算符号要前后要加空格

```
width: calc(200px*2);
```

相当于一个计算器自动计算

pointer-events是否阻止穿透；

none不阻止

_less简介 是一门css预处理语言 兼容性不好

less是一门css的预处理语言
- less是一个css的增强版，通过less可以编写更少的代码实现更强大的样式

css原生也支持变量的设置

```
html{
  /* css原生也支持变量的设置 */
  --color: #ff0;
  --length: 200px;
}
```

如果在css中设置变量则 通过var引入

```
height: var(--length);
background-color: var(--color);
```

在less中引入变量

变量的语法@变量名 一

```
// 变量的语法: @变量名
@a: 200px;
@b: #bfa;
```

```
.box5{
  //使用变量是，如果是直接使用则以 @变量名 的形式使用即可
  width: @a;
  color: @b;
}
```

二 作为类名

```
@c: box6;
```

```
//作为类名，或者一部分值使用时必须以 @{@变量名} 的形式使用
.@{c}{
  width: @a;
  background-image: url("@{c}/1.png");
}
```

变量发生重名时，会优先使用比较近的变量（就近原则） 并且可以在变量声明前就使用变量
先使用后设置

&符表示外层的父元素 extend

```
//:extend() 对当前选择器扩展指定选择器的样式（选择器分组）
.p2:extend(.p1){
  color: red;
}
```

```
.p3{
  //直接对指定的样式进行引用，这里就相当于将p1的样式在这里进行了复制
  //mixin 混合
  .p1();
}
```

flex布局 代替浮动完成页面布局

浮动，清除浮动，垂直对齐（vertical-align） 属性失效 行内元素为项目时可以正常设置宽高

- 1: 浮动 清除浮动 垂直对齐方式(vertical-align)无效
- 2: 默认主轴为水平方向
- 3: 项目沿着主轴方向排列
- 4: 分为容器属性和项目属性
- 5: 内联元素为项目时,可正常设置宽高

弹性盒子以及相关属性 给了display:flex (块弹性盒) or inline-flex (行内弹性盒) 属性

flex-direction 指定容器中弹性元素的排列方向 row默认值 自左向右 row-reverse自右向左 column自上向下column-reverse自下向上

flex-wrap设置弹性元素**是否在弹性盒子中自动换行**

```
flex-wrap:
设置弹性元素是否在弹性容器中自动换行
可选值:
    nowrap 默认值, 元素不会自动换行
    wrap 元素沿着辅轴方向自动换行
    wrap-reverse 元素沿着辅轴反方向换行
```

flex-flow (**direction和wrap的简写属性**)

```
/* flex-flow: wrap 和 direction 的简写属性
/* flex-flow: row wrap; */
```

justify-content-**如何分配主轴上的空白空间** (主轴上的元素如何排列) 水平方向

space-around两端平分 space-between两端对齐space-evenly平均分配 (每个元素之间的间隔相等)

```
justify-content
- 如何分配主轴上的空白空间 (主轴上的元素如何排列)
- 可选值:
    flex-start 元素沿着主轴起边排列
    flex-end 元素沿着主轴终边排列
    center 元素居中排列
    space-around 空白分布到元素两侧
    space-between 空白均匀分布到元素间
    space-evenly 空白分布到元素的单侧
```

align-items **元素在辅轴上如何对齐 垂直方向** (就相当于移动主轴在侧轴上的相应的位置)

stretch

主轴为水平拉伸的就是高度 (但是子盒子不要给高度) **主轴为垂直拉伸的就是宽度**

度

```
align-items:侧轴对齐方式;
    flex-start 起始位置
    center 居中
    flex-end 结束位置
    stretch 拉伸(默认值)
    baseline 文本底部对齐
```

多行找align-content 多根轴线对齐方式辅轴空白空间的分布 对于单行子元素，该属性不起作用。换行才起作用 但是它不对齐弹性项目，对齐的是换行后的几根主轴

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

)

弹性元素/弹性项目以及相关属性

弹性盒子的直接子元素叫做弹性元素 弹性元素的排列方向叫做主轴，与主轴垂直方向的叫做侧轴

align-self控制子项自己在侧轴上的排列方式

1. align-self: 侧轴对齐方式;	
flex-start	起始位置
center	居中
flex-end	结束位置
auto	继承父元素的align-items值 默认值
stretch	

flex-grow指定弹性元素伸展的系数 可给每个盒子单独设置，按比例分配

```
li:nth-child(2){
  background-color: pink;
  flex-grow: 2;
}

li:nth-child(3){
  background-color: orange;
  flex-grow: 3;
}
```

flex-shrink指定弹性元素收缩的系数

弹性元素的缩减系数
- 缩减系数的计算方式比较复杂
- 缩减多少是根据 缩减系数 和 元素大小来计算

弹性元素的属性:

- flex-grow 指定弹性元素的伸展的系数
 - 当父元素有多余空间的时，子元素如何伸展
 - 父元素的剩余空间，会按照比例进行分配
- flex-shrink 指定弹性元素的收缩系数
 - 当父元素中的空间不足以容纳所有的子元素时，如果对子元素进行收缩

flex-basis指定的是元素在主轴上的基础长度，设置的话会取代上面已经设置的长度

flex-basis 指定的是元素在主轴上的基础长度
如果主轴是 横向的 则 该值指定的就是元素的宽度
如果主轴是 纵向的 则 该值指定的是就是元素的高度
- 默认值是 **auto**，表示参考元素自身的高度或宽度
- 如果传递了一个具体的数值，则以该值为准

flex可以设置弹性元素三个样式 来表示占多少份数

flex 可以设置弹性元素所有的三个样式
flex 增长 缩减 基础;
initial "flex: 0 1 auto".
auto "flex: 1 1 auto"
none "flex: 0 0 auto" 弹性元素没有弹性

order可以决定弹性元素的排列顺序 数字越大越靠后 可以为负

```
li:nth-child(1){  
  /* order 决定弹性元素的排列顺序 */  
  order: 2;  
}  
  
li:nth-child(2){  
  background-color: pink;  
  /* flex-grow: 2; */  
  order: 3;  
}  
  
li:nth-child(3){  
  background-color: orange;  
  /* flex-grow: 3; */  
  order: 1;  
}  
/style>
```

布局方案 临界点

响应式布局

多列布局

column-count分列 column-gap列间距 column-rule:1px solid black;列边框 column-fill填充方式: auto 优先填满前面的 blance尽可能平均分 column-width列宽 columns:200px 综合写法

```
column-count:分列;  
column-gap:列间距;  
column-rule:列边框大小 形态 颜色;  
column-fill:填充方式;  
    balance 尽可能平均分配  
    auto 优先填满前面的列数  
column-width列宽;  
column-span:是否跨列;  
columns:列宽 列数;
```

break-inside:avoid;防止断层

GRID布局（网格布局） 网格 网线的摆放方向 有容器有项目 默认情况下，容器元素都是块级元素，但也可以设成行内元素。

容器属性

有几个数字就有几行 网格线可以超出区域

grid-template-rows定义行高 / grid-template-columns 定义列宽 有几个数字就有几列or几行

单位可用px或者% (如果设置太多了 网格就会超出范围) 给fr就会自动分配占几分 有固定了就从剩下的分配 repeat() 接受两个参数, 第一个参数是重复的次数 (上例是3), 第二个参数是所要重复的值。 auto-fill自动填充, 直到容器不能放置更多的列 (列宽固定)

minmax 表示长度就在这个范围之内 可以变化宽度 一个最小一个最大值

里面可以使用方括号, 指定每一根网格线的名字, [c1] 100px [c2] 100px [c3] auto [c4];

```
grid-template-columns:200px 250px 200px;
grid-template-columns:40% 20% 30% 10%;
grid-template-columns:1fr 1fr 1fr 2fr 3fr;
grid-template-columns:2fr 2fr 2fr 2fr;
grid-template-columns:repeat(6,1fr);
grid-template-columns:repeat(auto-fill,70px);
grid-template-columns:100px 100px 50px
minmax(50px,200px);
```

fr占几份 repeat重复 auto-fill填充 minmax功能函数

间距写法 简写gap

```
grid-row-gap:10px;
grid-column-gap:20px;
row-gap:20px;
column-gap:10px;
gap:10px 20px;
```

grid-template-areas 属性定义区域 如果某些区域不需要用到 则需要用 " . " 表示没有用到该单元格, 不属于任何区域

```
a . c'
d . f'
g . i'
```

' a b c ' 9个区域 也可以写相同的aaa 指这一行是一个的

' d e f '

' g h i ';

grid-auto-flow 设置先行后列(默认)或者先列后行row /row dense(后面这个先行后列并且尽可能紧密填满。不留空格) column/column dense (先列后行", 并且尽量填满空格。)

1	5	8
2	6	9
3	4	7

justify-items单元格内容的水平位置

align-items单元格内容的垂直位置 start center end stretch(默认值)

place-items 属性是 align-items 属性和 justify-items 属性的合并简写形式。如果省略第二个值，则浏览器认为与第一个值相等。

justify-content 整个内容区域在容器里面的水平位置(左中右)

align-content 整个内容区域的垂直位置（上中下）。 start center end stretch(默认值)

space-around space-between space-evenly

里面可以使用方括号，指定每一根网格线的名字， [c1] 100px [c2] 100px [c3] auto [c4];

项目属性 grid-area 为项目指定已经被定义的区域

grid-row grid-column

```
.Box1{background-color: red;grid-area: a;}
```

优化

1) 页面主题优化

实事求是的写下自己网站的名字，网站的名字要合理，最好包含网站的主要内容。

2) 页面头部优化

```
<meta name="keywords" content="" />向搜索引擎说明你的网页的关键词；  
<meta name="description" content="" />告诉搜索引擎你的站点的主要内容；
```

说明

- 1、“描述”部分应该用近乎描述的语言写下一段介绍你网站的文字，在其中，你应该适当的对你网站的特色内容加以重复以求突出；
- 2、“关键字”部分应该列出你认为合适的，能突出网站内容的关键字就可以了，关键字不要设置太多，可设置10---8个，搜索引擎只会浏览靠前的几个关键字。

4) 图片优化

图片优化并不是修改图片的大小、颜色，而是你应该为每个标签加上alt属性，alt属性的作用是当图片无法显示时以文字作为替代显示出来，而对于SEO来说，它可以令搜索引擎有机会索引你网站上的图片，对于一些确实没什么意义的图片，最好也不要省略alt，而应该留空，即 alt=""。

5) PageRank (pr值,友情链接)

PR值是Google提出的一个重要参数，它标明了某个网站的重要程度，那么pr值是如何确定的呢？目前普通的解释为：假如有ABC三个网站，彼此互作友情链接，那么当一个访客通过A上的友情链接来到B时，Google就认为A为B投了“一票”，同理，如果有人从C访问B，那么B又得一票，如果全世界的网站上都有B的友情链接，B就是世界上最重要的网站了！

css中模拟鼠标单击效果

1 通过:target伪类

2 通过input单选框/复选框 label 因为需要实现点击之后才进行切换的效果，此时将lable指向相应的input标签的id并使用伪类:checked来实现选择切换的效果。

```
<input type="radio" name="sildeInput" value="0" id="Input1" hidden>
<label class="label1" for="Input1">1</label>
<input type="radio" name="sildeInput" value="1" id="Input2" hidden>
```

3 如果为元素指定tabindex则可以使用:focus伪类来模拟单击。

```

```

```
#btnLeft:focus{
```