Html/Css 学习笔记

目录

1. [Html](#Html)

1.1[meta标签、实体](#meta标签、实体)

1.2[语义化标签](#语义化标签)

1.3[块元素和行内元素](#块元素和行内元素)

1.4[布局标签结构化语义标签](#布局标签结构化语义标签)

1.5[列表](#列表)

1.6[超链接](#超链接)

1.7[相对路径](#相对路径)

1.8[图片标签](#图片标签)

1.9[内联框架](#内联框架)

1.10[音视频](#音视频)

1. [Css](#Css)

2.1[css简介](#css简介)

2.2[css语法](#css语法)

2.3[常用选择器](#常用选择器)

2.4[复合选择器](#复合选择器)

2.5[关系选择器](#关系选择器)

2.6[属性选择器](#属性选择器)

2.7[伪类选择器](#伪类选择器)

2.8[超链接的伪类](#超链接的伪类)

2.9[伪元素选择器](#伪元素选择器)

练习： [CSS餐厅练习](#CSS餐厅练习)

2.10[继承](#继承)

2.11[选择器的权重](#选择器的权重)

2.12[像素、百分比](#像素、百分比)

2.13[颜色单位](#颜色单位)

3. [Layout](#Layout)

3.1[文档流](#文档流)

3.2[盒模型](#盒模型)

3.3[盒模型\_边框](#盒模型_边框)

3.4[盒模型\_内边距](#盒模型_内边距)

3.5[盒模型\_外边距](#盒模型_外边距)

3.6[盒模型\_水平布局](#盒模型_水平布局)

3.7[盒模型\_垂直布局](#盒模型_垂直布局)

3.8[盒模型\_外边距的折叠](#盒模型_外边距的折叠)

3.9[行内元素的盒模型](#行内元素的盒模型)

3.10[浏览器默认样式](#浏览器默认样式)

3.11[盒子大小](#盒子大小)

3.12[轮廓阴影和圆角](#轮廓阴影和圆角)

4.[Float](#Float)

4.1[浮动的简介](#浮动的简介)

4.2[浮动的特点](#浮动的特点)

4.3简单的布局

4.4[高度塌陷和BFC](#高度塌陷和BFC)

4.5 [Clear](#Clear)

4.6高度塌陷最终方案

4.7 [Clearfix](#Clearfix)

5.position

5.1[相对定位](#相对定位)

5.2[绝对定位](#绝对定位)

5.3[固定定位](#固定定位)

5.4[粘滞定位](#粘滞定位)

5.5[绝对定位元素的位置](#绝对定位元素的位置)

5.6[元素的层级](#元素的层级)

6.Font&Background

6.1[字体族](#字体族)

6.2[图标字体简介](#图标字体简介)

6.3[图标字体的其他使用方式](#图标字体的其他使用方式)

6.4阿里的字体库

6.5[行高](#行高)

6.6[字体的简写属性](#字体的简写属性)

6.7[文本的水平和垂直对齐](#文本的水平和垂直对齐)

6.8[其他文本样式](#其他文本样式)

6.9[背景](#背景)

6.10[背景2](#背景2)

练习：[练习：雪碧图](#练习：雪碧图)

6.11[线性渐变](#线性渐变)

6.12[径向渐变](#径向渐变)

7. Html补充

7.1[表格](#表格)

7.2[表格的样式](#表格的样式)

7.3[表单](#表单)

8.Animation

8.1[过渡](#过渡)

8.2[动画](#动画)

8.3[变形平移旋转](#变形平移旋转)缩放

9.Less

9.1[less简介](#less简介)

9.2[less中的变量](#less中的变量)

9.3[父元素和扩展](#父元素和扩展)

10.Flex

10.1[弹性盒简介](#弹性盒简介)

10.2[弹性容器上的样式](#弹性容器上的样式)

10.3[弹性元素上的样式](#弹性元素上的样式)

10.4[像素](#像素)

10.5[完美视口](#完美视口)

10.6[vw单位](#vw单位)

10.7[媒体查询简介](#媒体查询简介)

GitHub代码位置：

[mofeng007/Html-CssStudy: 这是我自学Html/Css的代码和笔记 (github.com)](https://github.com/mofeng007/Html-CssStudy)

1.html

1. meta标签、实体

<meta charset="utf-8">

*<!-- title的内容会作为搜索结果超链接上的文字展示 -->*

    <title>第一个网站</title>

*<!-- 几秒后跳转另一个页面 -->*

    <meta http-equiv="refresh" content="3;url=https://www.bilibili.com/">

*<!-- 搜索的关键字 -->*

    <meta name="keywords" content="购物，手表">

*<!-- 搜索结果的描述 -->*

    <meta name="description" content="这是我学习做的网站">

*<!--*

*多个空格默认为一个*

*不能直接书写一些特殊符号 eg：大于号、小于号*

*如果需要在网页中书写这些特殊符号，则需要使用html中的实体（转义字符）*

*&nbsp;空格*

*&gt;大于*

*&lt;小于*

*&copy;版权符号*

*更多的可以w3c中查找*

*-->*

1. 语义化标签

*<!-- 使用html标签时 关注的应该是他的语义而不是样式 -->*

*<!-- 在页面中独占一行的称为块元素 标题标签是块元素 -->*

*<!-- hgroup标题分组，可以将一组相关的标题放在一组*

*h1到h6 h1最重要 逐渐递减*

*-->*

1. 块元素和行内元素

*<!-- 网页中一般用块元素来对页面进行布局*

*行内元素一般用来包裹文字*

*一般会在块元素中放行内元素 而不会在行内元素中放块元素*

*p元素中不能放任何的块元素-->*

*<!-- 浏览器在解析网页时会自动对不符合规范的内容进行修正*

*右键检查或f12可以查看  （查看网页在内存中的结构）-->*

1. 布局标签 结构化语义标签

*<!-- header 表示网页的头部*

*main 表示网页的主体部分 一个网页只有一个main*

*footer 表示网页的底部*

*nav 表示网页中的导航*

*aside 表示和主体相关的其他内容（主要表现为侧边栏）*

*article 表示一篇独立的文章*

*section 表示一个独立的区块 以上标签都不能使用时用section*

*div没有语义 就用来表示一个区块 目前来讲div还是我们主要的布局元素*

*span 行内元素 没有语义 一般用来在网页中选中文字*

*-->*

1. 列表

*<!-- list 列表*

*1.铅笔*

*2.橡皮*

*3.尺子*

*在html里也能创建列表 总共三种*

*1.有序列表*

*2.无序列表*

*3.定义列表*

*有序列表：使用ol标签来创建无序列表*

*用li表示列表项*

*无序列表：使用ul标签来创建无序列表*

*用li表示列表项*

*定义列表：使用dl标签来创建定义列表*

*用dt来表示定义内容*

*用dd来对内容进行解释说明*

*（为了确保在不同的浏览器打开都是一样的 数字和小点一般都会去掉）*

*一般ul用的比较多*

*列表直接可以相互嵌套-->*

1. 超链接

*<!--*

*超链接可以让我们从一个页面跳转到另一个页面*

*或者是当前页面的其他位置*

*使用 a 标签来定义超链接*

*属性*

*href 指定跳转的目标路径*

*可以是外部网址，也可以是内部页面地址*

*超链接也是一个行内元素，在 a 标签中可以嵌套除它自身外的其他元素*

*-->*

*<!-- 超链接的其他用法*

*target属性，用来指定超链接打开的位置*

*可选值：*

*\_self 默认值 在当前页面中打开超链接*

*\_blank 在一个新的页面中打开超链接*

-*->*

*<!--*

*所谓的底部就是网页最后一个标签的位置*

*-->*

*<!--*

*可以直接将超链接的href属性设置为#，*

*这样点击以后，不会打开新页面，而是转到当前页面的顶部位置*

*可以跳转到页面的指定位置，将href属性的值设置为 #加上目标元素的id属性值*

*id属性（唯一不重复的）*

*每一个标签都可以添加一个id属性*

*id属性是元素的唯一标识，同一个页面不能出现重复的id属性*

*-->*

*<!--*

*可以使用#作为占位符，但是会回到顶部*

*-->*

*<!—*

*可以使用javascript:;作为占位符，点击后什么也不会发生*

*-->*

1. 相对路径

*<!--*

*当我们需要跳转到一个服务器内部的页面时，一般我们都会使用相对路径*

*相对路径都会使用.或..开头*

*./*

*../*

*./可以省略不写，如果不写./也不写../则相当于写了./*

*./表示当前文件所在的目录*

*在这里当前页面就是 07相对路径.html*

*./就等于 07相对路径.html 所在的目录 path*

*../表示当前文件所在目录的上一级目录*

*-->*

1. 图片标签

*<!--*

*用于向当前页面中引入一个外部图片*

*使用img标签来引入外部图片，img标签是一个自结束标签*

*img元素属于替换元素（介于块元素和行内元素之间，具有两种元素的特点）*

*属性：*

*src 属性指定的是外部图片的路径*

*alt 图片描述，这个描述默认情况下不会显示，有些浏览器会在图片无法加载是显示*

*搜索引擎会根据alt中的内容来识别图片，如果不写alt属性则图片不会被搜索引擎所识别*

*width 图片宽度（单位是像素）*

*height 图片高度*

*宽度高度如果只修改了一个，则另一个会等比例缩放*

*注意：*

*一般情况在pc端，不建议修改图片的大小，需要多大就裁多大 （大图缩小，浪费内存；小图放大，图片失帧）*

*但是在移动端，经常需要对图片进行缩放（大图缩小）*

*常见图片的格式：*

*jpeg（jpg）*

*-支持的颜色比较丰富，不支持透明效果，不支持动图*

*-一般用来显示照片*

*gif*

*-支持的颜色比较少，支持简单透明，支持动图*

*-颜色单一的图片，动图*

*png*

*-支持的颜色丰富，支持复杂透明，不支持动图*

*-颜色丰富，复杂透明图片（专为网页而生）*

*webp*

*-谷歌新推出的专门用来显示网页中的图片的一种格式*

*-它具备其它图片格式的所有优点，而且文件还特别的小*

*-缺点，兼容性不好，在一些比较老的浏览器要兼容就会出问题*

*base64*

*-将图片使用base64编码，这样可以将图片转换为字符，通过字符的形式来引入图片*

*-一般都是一些需要和网页一起加载的图片才会使用base64*

*一般使用原则：*

*效果一样的，用小的*

*效果不一样的，用效果好的*

*-->*

1. 内联框架

*<!--* iframe

*内联框架，用于向当前的页面引入一个其他的页面*

*src 指定要引入的网页的路径*

*frameborder 指定内联框架的边框*

*-->*

1. 音视频

*<!--*

*audio 标签用来向页面中引入一个外部的音频文件*

*音视频文件引入时，默认情况下不允许用户自己控制播放停止*

*属性：*

*controls 是否允许用户控制播放（有的属性不用给值 controls就是一个不用给值的属性）*

*autoplay 音频文件是否自动播放*

*-如果设置了autoplay 则音频文件在打开页面时会自动播放*

*但是目前来讲，大部分浏览器都不会自动对音频文件进行播放*

*loop 音频是否循环播放*

*-->*

*<!--*

*除了通过src来指定外部文件的路径以外，还可以通过source来指定文件*

*对于不支持的浏览器，会忽略audio和source标签，但是不会忽略文字，这样对用户比较友好，会有提示*

*对于不同的音频格式*

*可以有多个source标签，指定多个文件，多个文件不会一同使用，会逐渐往下识别，*

*第一个不行就播放第二个，第二个不行就显示文字或者继续往下播放，这样可以有效的解决浏览器之间的兼容性问题*

*-->*

*<!--*

*embed是一个比较老的标签，用于在老版本的浏览器中引入音视频文件，*

*也可以把embed放在audio里面，也可以解决兼容性问题（文字和embed就可以只选择一个）*

*属性：*

*type指定文件类型，前面是大类型，后面是具体类型*

*<embed src="source/Max Elto-Shadow Of The Sun.flac" type="audio/flac" width="200" height="100">*

*-->*

*<!--*

*使用video标签来向网页中引入一个视频*

*用法和audio一样的，也是建议用source,解决不同浏览器的兼容性问题*

*-->*

*<!--*

*一般音视频文件不会放在本地，一般会用云服务托管视频或者引入外部视频*

*-->*

2.Css

01.css简介

*<!--*

*网页分为三个部分*

*结构（HTML）*

*表现（CSS）*

*行为（JavaScript）*

*CSS*

*-层叠样式表*

*-网页实际上是一个多层的结构，通过css可以分别为网页的每一层来设置样式*

*而最终我们能看到的只是网页的最上边一层（类似于俯视一个蛋糕）*

*-总之，css用来设置网页中的元素的样式*

*-->*

*<!--*

*使用css来修改元素的样式*

*第一种方式（内联样式、行内样式）：*

*-在标签内部通过style属性来设置元素的样式，名字和值之间用：冒号后面加值最后用分号结尾；*

*-问题：*

*使用内联样式，样式只能对一个标签生效，如果希望影响到多个元素，必须在每一个元素中都复制一遍*

*并且当样式发生变化时，我们还得一个一个的修改，非常不方便*

*-注意：*

*开发时绝对不要使用内联样式*

*<p style="color: red; font-size: 20px;">生而为人我很抱歉</p>*

*第二种方式 （内部样式表）*

*-将样式编写到head标签中的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，并且修改时只需要修改一处即可全部应用*

*-内部样式表更加方便对样式进行复用*

*-问题：*

*我们的内部样式表只能对一个网页起作用，*

*他里边的样式不能跨网页进行复用*

*<style>*

*p{*

*color: yellow;*

*font-size: 50px;*

*}*

*</style>*

*第三种方式（外部样式表）*

*-可以将css样式编写到一个外部的css文件中*

*然后通过link标签来引入外部的css文件*

*-外部样式表需要通过link标签进行引入，意味着只要想使用这些样式的网页，*

*都可以进行引用，使样式在不同的页面进行复用*

*-将样式编写到外部的css文件中，可以使用到浏览器的缓存机制，*

*从而加快网页的加载速度，提高用户的体验。*

*-->*

02.css语法

*/\**

*CSS中的注释，注释中的内容会自动被浏览器忽略*

*CSS中的基本语法*

*选择器 声明块*

*选择器，通过选择器可以选中页面中的指定元素*

*比如 p 的作用就是选中页面中所有的 p 元素*

*声明块，通过声明块来指定要为元素设置的样式*

*声明块由一个一个的声明组成*

*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*

*\*/*

Eg：p{

            color: *aquamarine*;

            font-size: 50px;

        }

03.常用选择器

*<!--*

*class是一个标签的属性，它与id相似，不同的是class可以重复使用*

*可以通过class属性来为元素分组*

*可以同时为一个元素指定多个class属性，用空格隔开*

*-->*

*/\**

*一、元素选择器*

*-作用，根据标签名来选中指定的元素*

*-语法，标签名{}*

*-eg，p{}  h2{}  div{}*

*二、id选择器*

*-作用，根据元素的id属性值选中一个元素*

*-语法，#id属性值{}*

*-eg，#box{} #red{}*

*三、类选择器*

*-作用，根据元素的class属性值选中一组元素*

*-语法，.class属性值{}*

*-eg，.abc{} .blus{}*

*四、通配选择器*

*-作用，选中页面中的所有元素*

*-语法、\**

*\*/*

04.复合选择器

*/\**

*交集选择器*

*-作用，选中同时复合多个条件的元素*

*-语法，选择器1选择器2选择器3选择器n{}*

*-eg，div.class{}*

*-注意，交集选择器中如果有元素选择器，必须使用元素选择器开头*

*\*/*

*/\**

*选择器分组（并集选择器）*

*-作用，同时选择多个选择器对应的元素*

*-语法，选择器1，选择器2，选择器3，选择器n{} （逗号隔开）*

*-eg，h1，span{}*

*\*/*

05.关系选择器

*<!--*

*父元素*

*-直接包含子元素的元素叫做父元素*

*子元素*

*-直接被父元素包含的元素叫做子元素*

*祖先元素*

*-直接或间接包含后代元素的元素叫做祖先元素*

*-一个元素的父元素也是他的祖先元素*

*后代元素*

*-直接或间接被祖先元素包含的元素叫做后代元素*

*-子元素也是后代元素*

*兄弟元素*

*-拥有相同父元素的元素叫做兄弟元素*

*-->*

*/\**

*为div的子元素span设置一个字体颜色红色*

*（为div直接包含的span设置一个字体颜色）*

*子元素选择器*

*-作用，选中指定父元素的子元素*

*-语法，父元素 > 子元素*

*-eg，div > span{}*

*div>span{*

*color: red;*

*}*

*\*/*

*/\**

*后代元素选择器（范围会更大一些）*

*-作用，选中指定元素内的指定后代元素*

*-语法，祖先空格后代*

*-eg，div span{}*

*div span{*

*color: chartreuse;*

*}*

*\*/*

*/\**

*兄弟元素选择器（两个元素要紧挨着）*

*-作用，选择下一个兄弟*

*-语法，前一个加号后一个*

*-eg，p+span*

*p+span{*

*color: cyan;*

*}*

*-作用，选择下边所有的兄弟*

*-语法，前一个波浪号后一个*

*-eg，p~span*

*p~span{*

*color: darkorange;*

*}*

*\*/*

06.属性选择器

*/\**

*[属性名]选择含有指定属性的元素*

*[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的（^ shift加6）*

*[属性名$=属性值]选择属性值以指定值结尾的元素的*

*[属性名\*=属性值]选择属性值中含有某值的元素*

*p[title]{}*

*p[title=abc]{}*

*p[title^=ab]{}*

*p[title$=c]{}*

*p[title\*=c]{}*

*\*/*

07.伪类选择器

*/\**

*需求：想让第一个li变为红色，用ul>li.first可以实现*

*但是如果li结构发生变化，例如加了一个第零个，*

*需求就无法准确实现，不严谨，所以可以用伪类。*

*\*/*

*/\**

*伪类（不存在的类，特殊的类）*

*-伪类用来描述一个元素的特殊状态*

*-比如，第一个子元素，被点击的元素，鼠标移入的元素等*

*-伪类一般情况下用冒号开头，*

*-eg，:first-child 第一个子元素;*

*:last-child 最后一个子元素;*

*:nth-child() 选中第n个子元素（括号里写）;*

*特殊值：*

*n：第n个 ，n的范围是零到正无穷*

*2n或even：表示选中偶数位的元素*

*2n+1或odd：表示选中奇数位的元素*

*-注意：以上这些伪类都是根据所有的子元素进行排序*

*:first-of-type*

*:last-of-type*

*:nth-of-type()*

*-这几个伪类的功能和上述的类似，不同点是他们是在同类型元素中进行排序*

*-:not() 否定伪类*

*-将符合条件的元素从选择器中去除*

*\*/*

*/\* ul>li:first-child{*

*color: red;*

*} \*/*

*/\* ul>li:last-child{*

*color: blue;*

*} \*/*

*/\* ul>li:nth-child(){*

*color: blue;*

*} \*/*

*/\* ul>li:not(:nth-child(3)){*

*color: blue;*

*} \*/*

08.超链接的伪类

*<!--*

*特殊状态：*

*1.没有访问过的连接*

*2.访问过的链接*

*-->*

*顺序：*

*我定义的顺序是a:visited、a:hover、a:link，这时候就会发现，不管我有没有hover上去，样式都是link的样式的，这是因为鼠标经过的未访问链接同时拥有a:link、a:hover两种属性，在第一种情况下，a:link离它最近，所以它优先满足a:link，而放弃a:hover的重复定义。*

*所以，如果一个链接没有被访问过，那么它有可能同时拥有 link , hover 两个属性，不能让 link 写在后面，否则不管 hover 是否，都会显示link的样式；同样的道理，如果一个链接已经被访问过了，那么它有可能同时拥有visited , hover 两个属性，hover 要在 visited 的后面；*

*如果把hover放在active后面，那么实际你在激活（active）链接的时候就触发了hover伪类，hover在后面覆盖了active的颜色，所以始终无法看到active的颜色 。*

*其实 link visited 这两个伪类，并不需要顺序关系。（他们两的位置可以交换）*

*所以，我们总结出，要想让一个链接各个阶段都正常呈现样式，正确的顺序是* ***lvha*** *;*

*/\**

*link用来表示没访问过的链接（正常的链接）*

*\*/*

*/\**

*visited用来表示访问过的链接*

*由于隐私的原因，所以visited这个伪类只能用来修改链接的颜色*

*\*/*

*/\**

*:hover用来表示鼠标移入的状态*

*\*/*

*/\**

*:active用来表示鼠标点击*

*\*/*

09.伪元素选择器

*/\**

*伪元素：表示页面中一些特殊的并不真实存在的元素（特殊的位置）*

*-eg：首字母，选中的内容，第一行等。。。*

*::first-letter 表示第一个字母*

*::first-line 表示第一行*

*::selection 表示选中的内容*

*::before 元素的开始*

*::after 元素的最后*

*after和before这两个必须结合content属性来使用*

*content 属性与 :before 及 :after 伪元素配合使用，来插入内容。*

*\*/*

餐厅练习：

[CSS Diner - Where we feast on CSS Selectors! (flukeout.github.io)](https://flukeout.github.io/)

10.继承

*<!--*

*样式的继承，我们为一个元素设置的样式同时也会应用到它的后代元素上*

*继承是发生在祖先和后代元素之间*

*继承的设计是为了方便我们的开发，利用继承我们可以将一些通用的样式*

*统一设置到共同的祖先元素上，这样只需要设置一次样式即可*

*注意：并不是所有的样式都会被继承*

*-eg：背景相关的，布局相关的等这些样式都不会被继承，*

*-->*

1. 选择器的权重

*/\**

*样式的冲突*

*-当我们通过不同的选择器，选中同一个元素，*

*并且为相同的样式设置不同的值时，此时就发生了样式冲突*

*发生样式冲突时，应用哪个样式由选择器的权重（优先级决定）*

*选择器的权重（可以理解为：越具体优先级越高）*

*内联样式（行内样式） 1000*

*id选择器            100*

*类和伪类选择器       10*

*元素选择器           1*

*通配选择器           0（就是谁都可以覆盖）*

*继承的样式           没有优先级*

*比较优先级时，需要将所有的选择器的优先级进行相加计算，最后优先级越高，*

*越优先显示，（分组选择器单独计算）*

*注意：*

*选择器的累加不会超过其最大的数量级，类选择器再高也不会超过id选择器*

*如果优先级计算后相同，此时优先使用靠下的样式*

*可以在某一个样式后面添加!important，则此时该样式会获得最高的优先级，甚至超过内联样式*

*注意：在开发中这个玩意一定要慎用！*

*\*/*

1. 像素、百分比、em、rem

*/\**

*长度单位：*

*像素：*

*-屏幕（显示器）实际上是由一个一个的小点点构成的*

*-不同屏幕的像素大小是不同的，像素越小的屏幕显示的效果越清晰*

*-所以同样的200px在不同的设备下显示效果不一样*

*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*

*-设置百分比可以使子元素随父元素的改变而改变*

*em：*

*-em是相对于元素的字体大小来计算的*

*-1em = 1font-size   1em=16px*

*-em会根据字体大小的改变而改变*

*rem：*

*-rem是相对于根元素的字体大小来计算的 （html）*

*\*/*

1. 颜色单位

*/\**

*颜色单位：*

*-在CSS中可以直接使用颜色名来设置各种颜色*

*-eg：red，blue......*

*-但是在CSS中直接使用颜色名是非常的不方便（名字记不住等）*

*-RGB值：（可以用取色器来取，QQ截图也可以）*

*-RGB通过光的三原色的不同浓度来调配出不同的颜色*

*-R red，G green，B blue*

*-每一种颜色的范围在0~255（0%~100%）之间*

*-语法，RGB（红，绿，蓝）;*

*-RGBA：*

*-就是在RGB的基础上增加了一个A表示不透明度*

*-需要四个值，前三个和RGB一样，第四个表示不透明度*

*-rgba(red, green, blue, alpha);*

*-1表示完全不透明 ， 0表示完全透明，.5表示半透明*

*-十六进制的RGB值：*

*-语法，#红绿蓝*

*-颜色浓度通过00~ff调配*

*-如果颜色两位两位重复，可以进行简写*

*-#aabbcc --> #abc*

*-HSL值，HSLA值*

*-H 色相（一个圆环 范围：0~360）*

*-S 饱和度（颜色浓度 0%~100%）*

*-L 亮度（颜色的亮度 0%~100%）*

*-A 透明度*

*\*/*

background-color: rgb(255, 255, 255); */\* 白色 \*/*

  background-color: rgb(0, 0, 0); */\* 黑色 \*/*

3.Layout

01.文档流

*<!--*

*文档流（normal flow）*

*-网页是一个多层结构，一层摞着一层*

*-通过CSS可以分别为每一层设置样式*

*-作为用户来讲只能看到最顶上一层*

*-这些层中，最底下的一层称为文档流，文档流是网页的基础*

*我们所创建的元素默认都是在文档流中进行排列*

*-对于我们来说，元素主要有两个状态*

*-在文档流中*

*-不在文档流中（脱离文档流）*

*-元素在文档流中有什么特点*

*-块元素*

*-块元素会在网页中独占一行*

*-默认宽度是父元素的全部（会把父元素撑满）*

*-默认高度是被内容撑开（子元素）*

*-行内元素*

*-行内元素不会独占页面的一行，只占自身大小*

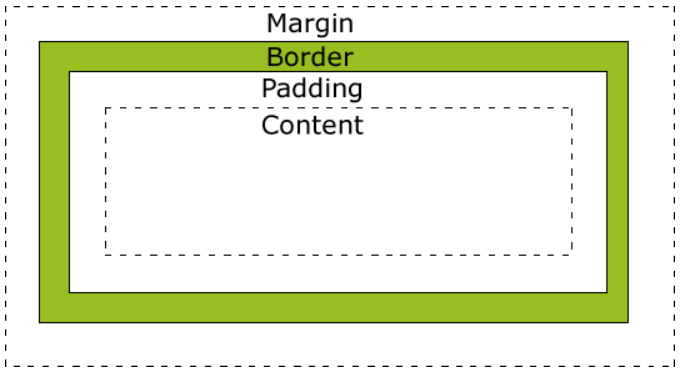
*-行内元素在页面中从左向右水平排列，如果一行中不能容纳下所有的行内元素*

*则元素会换到第二行继续自作向右排列（和我们的书写习惯一致）*

*-行内元素的默认宽高都是被内容撑开*

*-->*

02.盒模型



*<!--*

*盒模型，盒子模型，框模型，（box model）*

*-CSS将页面中的所有元素都设置为一个矩形的盒子*

*-将元素设置为矩形盒子后，对页面的布局就变成将不同的盒子摆放到不同的位置*

*-每一个盒子都由以下几个部分组成（内容区，内边距，边框，外边距）*

*-内容区（content），元素中的所有的子元素和文本内容都在内容区中排列*

*-内容区的大小由width和height两个属性来设置*

*-width 设置内容区的宽度*

*-height 设置内容区的高度*

*-内边距（padding）*

*-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*

*-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*

*-外边距（margin）*

*-->*

03.盒模型-边框

*/\**

*-边框的宽度 border-width*

*-默认值一般都是三个像素*

*-border-width可以用来指定四个方向的边框的宽度*

*-值的情况*

*四个值：上 右 下 左*

*三个值：上 左右 下*

*两个值：上下 左右*

*一个值：上下左右*

*-还有一组border-xxx-width*

*-xxx可以是 top right bottom left*

*-用来单独指定某一个边的宽度*

*-边框的颜色 border-color*

*-也可以省略不写，如果省略则自动使用color的颜色值*

*-border-color可以用来指定四个方向的边框的颜色，规则和border一样*

*-边框的样式 border-style （默认值是none）*

*-border-style 指定边框的样式 规则一样*

*-solid 表示实线*

*-dotted 表示点状虚线*

*-dashed 虚线*

*-double 双线*

*\*/*

            常用：

*/\**

*-border简写属性，通过该属性可以同时设置边框所有的相关样式，并且没有顺序要求*

*-除了border以外还有四个 border-xxx*

*-border-top*

*-border-right*

*-border-bottom*

*-border-left*

*\*/*

04.盒模型-内边距

*/\**

*内边距（padding）*

*-内容区和边框之间的距离是内边距*

*-一共具有四个方向的内边距*

*-padding-top*

*-padding-right*

*-padding-bottom*

*-padding-left*

*-内边距的设置会影响到盒子的大小*

*-背景颜色会延伸到内边距上*

*-一个盒子的可见框大小，由内容区 内边距 边框共同决定*

*所以在计算盒子大小时，需要将这三个区域加到一起计算*

*-padding 内边距的简写属性，可以同时指定四个方向的内边距*

*规则一样*

*\*/*

05.盒模型-外边距

*/\**

*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*

*-margin会影响到盒子实际占用空间*

*-一共有四个方向的外边距*

*-margin-top*

*-上外边距，设置一个正值，元素会向下移动*

*-margin-right*

*-默认情况下，设置margin-right不会产生任何变化*

*-margin-bottom*

*-下外边距，设置一个正值，其下边的元素会向下移动*

*-margin-left*

*-左外边距，设置一个正值，元素会向右移动*

*-margin也可以设置负值，如果是负值，则元素会向相反的方向移动*

*-元素在页面中是按自左向右的顺序排列的，*

*所以默认情况下如果我们设置的左和上外边距则会移动元素自身*

*而设置下和右外边距则会移动其他元素*

*-margin的简写属性*

*margin 可以同时设置四个方向的外边距，规则一样*

*\*/*

06.盒模型-水平布局

*/\**

*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*

*-左：*

*-margin-left*

*-border-left*

*-padding-left*

*-width（默认值就是auto）*

*-右：*

*-padding-right*

*-border-right*

*-margin-right*

*-一个元素在其父元素中，水平布局必须满足以下的等式：*

*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）*

*inner：（未设置的默认值为零）0+0+0+200+0+0+0=800*

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

*-调整的情况*

*-如果这七个值中没有 auto 的情况，则浏览器会自动调整margin-right的值以使等式满足*

*-这七个值中有三个值可以设置为 auto*

*-width*

*-margin-left*

*-margin-right*

*-如果某个值为auto，则会自动调整为 auto 的那个值以使等式成立*

*-eg：如果inner中width为 auto*

*所以：0+0+0+auto+0+0+0=800    所以：auto=800*

*如果inner中width为 auto ，margin-right为200*

*所以：0+0+0+auto+0+0+200=800  所以：auto=600*

*-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的外边距会自动为零*

*-如果将三个值都设置为 auto，则外边距都是零，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*

*-示例：*

*width：xxx px；*

*margin：0 auto；*

*\*/*

        .outer{

            width: 800px;

            height: 200px;

            border: 10px *red* *solid*;

        }

        .inner{

            width: 200px;

            height: 200px;

            background-color: *aqua*;

        }

07.盒模型-垂直布局

*/\**

*默认情况下父元素的高度被内容撑开*

*\*/*

*/\**

*子元素是在父元素的内容区中排列的，*

*-如果子元素的大小超过了父元素，则子元素会从父元素中溢出*

*-使用 overflow 属性来设置父元素如何处理溢出的子元素*

*-可选值：*

*-visible，默认值，子元素会从父元素中溢出，在父元素外部显示*

*-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（更灵活）*

*-overflow-x;*

*-overflow-y;*

*\*/*

08.盒模型-外边距的折叠

*/\**

*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*

*-如果都是负的，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*

*-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）*

*-父子外边距的折叠会影响到页面的布局，必须进行处理*

*\*/*

09.行内元素的盒模型

*/\**

*行内元素的盒模型*

*-行内元素不支持设置宽度和高度*

*-行内元素可以设置padding，但是垂直方向的padding不会影响页面的布局*

*-行内元素可以设置border，垂直方向的border不会影响布局*

*-行内元素可以设置margin，垂直方向的margin不会影响布局*

*\*/*

*/\**

*display 用来设置元素显示的类型*

*可选值：*

*-inline，将元素设置为行内元素*

*-block，将元素设置为块元素*

*-inline-block，将元素设置为行内块元素（既可以设置宽高又不会独占一行）*

*-table，将元素设置为表格*

*-none，元素不在页面中显示*

*visibility，用来设置元素的显示状态*

*可选值：*

*-visible，默认值，元素在页面中正常显示*

*-hidden，元素在页面中隐藏，不显示，但是依然占据页面的位置*

*\*/*

10.浏览器默认样式

*/\**

*浏览器的默认样式：*

*-通常情况下，浏览器都会为元素设置一些默认样式*

*-默认样式的存在会影响到页面的布局*

*通常情况下，编写网页时必须要去掉这些默认样式（pc端的页面）*

*\*/*

*<!--*

*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*

*通过link引入*

*-->*

11.盒子大小

*/\**

*默认情况下，盒子的可见大小由内容区、内边框、内边距和边框共同决定*

*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：默认值，宽度和高度用来设置内容区的大小*

*-border-box：宽度和高度用来设置整个盒子可见框的大小*

*width和height指的是内容区，内边距和边框的总大小*

*\*/*

12.轮廓阴影和圆角

*/\**

*-outline 用来设置元素的轮廓线，用法和border一致，*

*轮廓和边框不同的点：轮廓不会影响到可见框的大小*

*-box shadow 用来设置元素的阴影效果，阴影不会影响到页面布局*

*第一个值：水平偏移量，设置阴影的水平位置，正值向右移动，负值相反*

*第二个值：垂直偏移量，设置阴影的垂直位置，正值向下移动，负值相反*

*第三个值：阴影的模糊半径*

*第四个值：阴影的颜色*

*-border-radius （四个方向的圆角）*

*-border-top-left-radius*

*-border-top-right-radius*

*-border-bottom-left-radius*

*-border-bottom-right-radius*

*border-radius 可以用来指定四个方向的圆角*

*-四个值：左上，右上， 右下，左下*

*-三个值：左上，右上/左下，右下*

*-两个值：左上/右下， 右上/坐左下*

*\*/*

04.float

01.浮动的简介

*/\**

*通过浮动可以使一个元素向其父元素的左侧或右侧移动*

*使用 float 属性来设置元素的浮动*

*可选值：*

*-none 默认值，元素不移动*

*-left 元素向左移动*

*-right 元素向右移动*

*注意：元素设置浮动后，水平布局的等式不需要强制成立*

*元素设置浮动后，会完全从文档流中脱离，不再占用文档流的位置，*

*所以元素下边的还在文档流中的元素会自动向上移动*

*浮动的特点：*

*1、浮动元素会完全脱离文档流，不再占据文档流中的位置*

*2、设置浮动以后元素会向父元素的左侧或右侧移动，*

*3、浮动元素默认不会从父元素中移出*

*4、浮动元素向左或向右移动时，不会超过它前边的其他浮动元素*

*5、如果浮动元素上边是一个没有浮动的块元素，则浮动元素无法上移*

*6、浮动元素不会超过它上边的浮动的兄弟元素，最多最多就是和它一样高*

*简单总结：浮动目前来讲它的主要作用就是让页面中的元素可以水平排列*

*通过浮动可以实现一些水平方向的布局*

*\*/*

02.浮动的特点

*/\**

*-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，*

*所以我们可以利用浮动来设置文字环绕图片的效果*

*-元素设置浮动以后，将会从文档流中完全脱离，脱离以后，*

*元素的一些特点也会发生改变*

*-脱离文档流的特点：*

*-块元素：*

*1、块元素不再是独占一行*

*2、块元素的宽高都默认被内容撑开*

*-行内元素：*

*1、行内元素脱离后就变成了块元素，特点和块元素一样*

*-脱离文档流以后，不再区分行内元素和块元素*

*\*/*

03.简单的布局

04.高度塌陷和BFC

*/\**

*高度塌陷问题：*

*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*

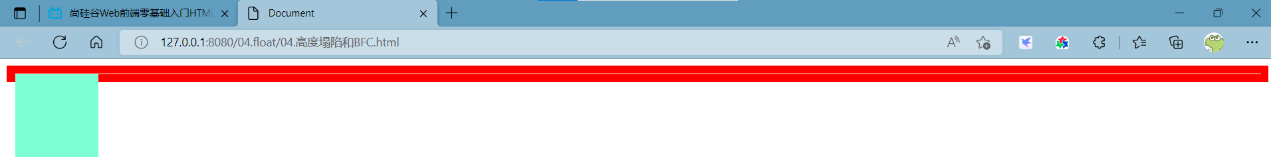
*父元素的高度，导致父元素的高度丢失。*

*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*

*\*/*

*绿色方块是红色框里的子元素*



*/\**

*BFC（Block Formatting Context）块级格式化环境*

*-BFC是一个CSS中的一个隐含的属性，可以为一个元素开启BFC*

*开启BFC后该元素会变成一个独立的布局环境*

*-元素开启BFC后的特点：*

*1、开启BFC的元素不会被浮动元素所覆盖*

*2、开启BFC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*

*-可以通过一些特殊方式来开启元素的BFC（都有副作用，使用副作用小的）*

*1、设置元素的浮动（不推荐）*

*2、将元素设置为行内块元素（不推荐）*

*3、将元素的overflow设置为一个非visible的值*

*-常用方式，为元素设置 overflow：hidden开启其BFC（使其可以包含浮动元素）*

*\*/*

05.clear

*\**

*如果不希望某个元素因为其他元素浮动的影响而改变位置，*

*可以通过clear属性来清除浮动元素对当前元素所产生的影响*

*clear：*

*-作用，清除浮动元素对当前元素所产生的影响*

*-可选值：*

*-left，清除左侧浮动元素对当前元素的影响*

*-right，清除右侧浮动元素对当前元素的影响*

*-both，清除两侧中最大影响的那侧*

*-原理：*

*设置清除浮动以后，浏览器会自动为元素添加一个上外边距，*

*以使其位置不再受其他元素的影响*

*\*/*

06.高度塌陷最终方案

::after{

            content: "";

            display: *block*;

            clear: *both*;

        }

07.clearfix

*/\* .clearfix 这个样式可以同时解决高度塌陷和外边距重叠的问题，当你遇到这些问题时，直接使用这个类即可 \*/*

        .clearfix::after,

        .clearfix::before{

            content: "";

            display: *table*;

            clear: *both*;

        }

5.position

01.相对定位

*/\**

*定位（position）：*

*-定位是一种更高级的布局手段*

*-通过定位可以将元素摆放到页面的任意位置*

*-使用position属性来设置定位*

*-可选值：*

*-static 默认值，元素是静止的没有开启定位*

*-relative 开启元素的相对定位*

*-absolute 开启元素的绝对定位*

*-fixed 开启元素的固定定位*

*-sticky 开启元素的粘滞定位*

*-相对定位：*

*-当元素的position属性值设置为relative时则开启了元素的相对定位*

*-相对定位的特点：*

*1、元素开启相对定位以后，如果不设置偏移量，元素不会发生任何变化*

*2、相对定位是参照于元素在文档流中的位置进行定位的*

*3、相对定位会提升元素的层级*

*4、相对定位不会使元素脱离文档流*

*5、相对定位不会改变元素的性质，块还是块，行内还是行内*

*-偏移量：（offset）*

*-当元素开启了定位以后，可以通过偏移量来设置元素的位置*

*-top：定位元素和定位位置上边的距离*

*-bottom:定位元素和定位位置下边的距离*

*-定位元素垂直方向的位置由top和bottom两个属性来控制*

*通常情况下我们只会使用其中一种*

*-top值越大，元素越往下移动*

*-bottom值越大，元素越往上移动*

*-left：定位元素和定位位置左边的距离*

*-right：定位元素和定位位置右边的距离*

*-定位元素水平方向的位置由left和right两个属性来控制*

*通常情况下我们只会使用其中一种*

*-left值越大，元素越往右移动*

*-right值越大，元素越往左移动*

*\*/*

02.绝对定位

*/\**

*绝对定位：*

*-当元素的position属性值设置为absolute时，则开启了元素的绝对定位*

*-绝对定位的特点：*

*1、开启绝对定位后，如果不设置偏移量，元素的位置不会发生变化*

*2、开启绝对定位后，元素会从文档流中脱离*

*3、绝对定位会改变元素的性质，行内变成块，块的宽高被内容撑开*

*4、绝对定位会使元素提升一个层级*

*5、绝对定位元素是相对于其包含块进行定位*

*-包含块（containing block）*

*-正常情况下：包含块是当前元素最近的祖先块元素*

*-绝对定位的包含块：此时包含块就是离他最近的开启了定位的祖先元素。*

*如果所有的祖先元素都没有开启定位，则根元素就是他的包含块*

*-html（根元素，初始包含块）*

*\*/*

03.固定定位

*/\**

*固定定位：*

*-将元素的position属性设置为fixed则开启了元素的固定定位*

*-固定定位也是一种绝对定位，固定定位的大部分特点也和绝对定位一样*

*-唯一不同的是固定定位永远参照于浏览器的视口进行定位*

*固定定位的元素不会随着网页的滚动而改变位置（eg：广告）*

*\*/*

04.粘滞定位

*/\**

*粘滞定位：*

*-当元素的position属性设置为sticky时则开启了元素的粘滞定位*

*-粘滞定位和相对定位的特点基本一致，不同的是粘滞定位可以在元素到达某个位置时将其固定*

*\*/*

05.绝对定位元素的位置

*/\**

*开启绝对定位后：*

*-水平布局必须满足以下的等式：（由原来的七个值变成九个值）*

*left + margin-left + border-left + padding-left*

*+ width + padding-right + border-right + margin-right + right*

*=其包含块内容区的宽度（必须满足）*

*-当我们开启绝对定位后 ：*

*-水平布局等式需要添加left和right两个值*

*此时规则和之前一样，只是多了两个值，*

*-当发生过度约束时：*

*-如果九个值中没有auto，则自动调整right值以使等式满足*

*-如果有auto，则自动调整auto的值以使等式满足*

*-可以设置auto的值：*

*margin，width，left，right*

*-因为left，right 的值默认是auto，所以如果不指定left和right*

*当等式不满足时，会自动调整这两个值*

*-垂直方向布局的等式也必须满足*

*top + margin-top/bottom + padding-top/bottom + border-top/bottom + height + bottom*

*\*/*

06.元素的层级

*/\**

*对于开启了定位的元素，可以通过z-index属性来指定元素的层级*

*z-index 需要一个整数作为参数，值越大元素的层级越高*

*-元素的层级越高，越优先显示*

*-如果元素的层级一样，则优先显示靠下的元素*

*-祖先元素的层级再高，也不会盖住后代元素*

*\*/*

6.font&background

01.字体族

*/\* @font-face可以将服务器中的字体直接提供给用户使用*

*问题：*

*1、加载速度*

*2、版权*

*\*/*

*/\* @font-face{*

*指定字体的名字，自己取*

*font-family: ;*

*服务器中字体的路径*

*src: url() format();   format():用来告诉浏览器格式，可以不写*

*} \*/*

*/\* 字体相关的样式  (字体不会失帧)*

*color：用来设置字体颜色*

*font-size：字体大小*

*font-size相关单位*

*-em：相对于当前元素的一个font-size（默认值16px）*

*-rem：相对于根元素的一个font-size*

*font-family：设置字体族（字体格式）*

*-可选值：（指定字体的类别，浏览器会自动使用该类别下的字体）*

*-serif：衬线字体*

*-sans-serif：非衬线字体*

*-monospace：等宽字体*

*-cursive：草书字体*

*-fantasy：虚幻字体*

*-font-family可以同时指定多个字体，多个字体间使用逗号隔开，*

*字体生效时优先使用第一个，第一个无法使用则使用第二个，以此类推*

*\*/*

02.图标字体简介

*<!--*

*图标字体（iconfont）*

*-在网页中经常需要使用一些图标，可以通过图片来引入图标*

*但是图片大小会比较大，并且非常的不灵活*

*-所以在使用图标时，我们还可以将图标直接设置为字体，*

*然后通过font-face的形式来对字体进行引入*

*-这样我们就可以通过使用字体的形式来使用图标*

*Font Awesome使用步骤：*

*1、下载*

*2、解压*

*3、将css和webfonts移动到项目中（这两个必须在同一级目录下）*

*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（fas或fab固定的，后面加想要的图标名字）*

*class="fas fa-skull-crossbones"*

*class="fab fa-accessible-icon"*

*-->*

03.图标字体的其他使用方式

*/\**

*通过伪元素来设置图标字体*

*1、找到要设置图标的元素通过before或after选中*

*2、在content中设置字体的编码*

*3、设置字体的样式*

*-font-family: 'Font Awesome 6 Free';*

*-font-weight: 900;*

*\*/*

        li::before{

            content: "\f1b0";

            font-family: 'Font Awesome 6 Free';

            font-weight: 900;

        }

*<!-- 通过实体来使用图标字体 -->*

    <span class="fas">&#xf0f3</span>

04.阿里的字体库

*使用方法大致相同*

05.行高

*/\**

*-行高（line height）*

*-行高指的是文字占有的实际高度*

*-可以通过 line height指定一个大小（px、em）*

*-也可以直接为行高设置一个整数*

*如果是一个整数的话，行高将会是字体的指定的倍数*

*-行高经常还用来设置文字的行间距*

*行间距 = 行高 - 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*

*-可以将行高设置为和元素高度一样的值，使单行文字在一个元素中垂直居中*

*\*/*

06.字体的简写属性

*/\**

*font 可以设置字体相关的所有属性*

*语法：（空格隔开）*

*font:字体大小/行高 字体族（字体大小 字体族必须写的）*

*行高可以省略不写，如果不写，自动使用默认值*

*font-weight 字重 字体加粗*

*可选值：*

*normal 默认值，不加粗*

*bold 加粗*

*100-900（九个级别，没什么用）*

*font-style 字体风格*

*可选值：*

*normal 正常的*

*italic 斜体*

*\*/*

07.文本的水平和垂直对齐

*/\**

*text-align 文本的水平对齐*

*可选值：*

*left 左侧对齐*

*right 右侧对齐*

*center 居中对齐*

*justify 两端对齐*

*vertical-align 设置元素垂直对齐的方式*

*可选值：*

*baseline 默认值，基线对齐*

*top 顶部对齐*

*bottom 底部对齐*

*middle 居中对齐*

*也可以直接指定值 eg：vertical-align: 10px;*

*图片默认也是基线对齐，会导致图片底下有一个缝，*

*影响布局，让他不以基线对齐即可改变*

08.其他的文本样式

*/\**

*text-decoration 设置文本修饰*

*可选值：*

*none 什么都没有（可以用来去除链接的下划线）*

*underline 下划线*

*line-through 删除线*

*overline 上划线*

*white-space 设置网页如何处理空白*

*可选值：*

*normal 正常*

*nowrap 不换行*

*pre 保留空白（网页中写的什么格式就什么格式，多个空格也可以显示）*

*配合 overflow: hidden;text-overflow: ellipsis；可以实现多余内容显示为省略号*

09.背景

*/\**

*设置背景图片*

*background-image: url('')*

*-可以同时设置背景图片和背景颜色，这样背景颜色会成为图片的背景色*

*-如果背景的图片小于元素，则背景图片会自动在元素中平铺将元素铺满*

*-如果背景的图片大于元素，将会有部分背景无法完全展示*

*-如果背景图片和元素大小一致，则会直接正常显示*

*background-repeat 用来设置背景的重复方式*

*可选值：*

*-repeat 默认值，背景会沿着X轴，Y轴方向重复*

*-repeat-x 沿着X轴方向重复*

*-repeat-y 沿着Y轴方向重复*

*-no-repeat 背景图片不重复*

*background-position 用来设置背景图片的位置*

*通过 top left right bottom center 几个表示方向的词来设置图片的位置*

*使用方位词时必须要同时指定两个值，如果只写一个则第二个默认就是center*

*通过偏移量来指定背景图片的位置，*

*水平方向的偏移量，垂直方向的偏移量*

*\*/*

10.背景2

*/\**

*background-clip 设置背景的范围*

*可选值：*

*-border-box 默认值，背景会出现在边框的下边*

*-padding-box 背景不会出现在边框，只出现在内容区和内边距*

*-content-box 背景只会出现在内容区*

*background-origin 设置背景图片的偏移量计算的原点*

*可选值：*

*-padding-box 默认值，背景图片的偏移量（background-position）从内边距处开始计算*

*-content-box 背景图片的偏移量（background-position）从内容区开始计算*

*-border-box 背景图片的偏移量（background-position）从边框处开始计算*

*background-size 设置背景图片的大小*

*可选值：*

*-第一个值表示宽度*

*-第二个值表示高度*

*-如果只写一个，则第二个值默认是auto*

*eg： background-size: auto 200px;*

*-cover 图片的比例不变，将元素铺满*

*-contain 图片的比例不变，将图片在元素中完整显示*

*background-attachment 设置背景图片是否随元素滚动*

*可选值：*

*-scroll 默认值，背景图片会随元素移动*

*-fixed 背景图片会固定在页面中，不会随元素移动*

*background 背景相关的简写属性，所有背景相关的属性都可以通过该样式来设置*

*-并且该样式没有顺序要求，也没有哪个属性必须写。*

*-注意： background-size必须写在background-position的后面，并且用 / 隔开*

*background-size/background-position*

*background-origin和background-clip两个样式，origin要在clip前面*

*\*/*

练习：雪碧图

*/\**

*图片属于网页中的外部资源，外部资源都需要浏览器单独发送请求加载*

*浏览器加载外部资源都是按需加载的，用则加载，不用则不加载*

*像这个练习（背景练习2（按钮）.html），link会首先加载，而hover和active会在指定状态触发时才会加载*

*\*/*

*/\**

*解决图片的闪烁问题：*

*可以将多个小图片保存到一个大图片中，这样图片会同时加载到网页中，*

*然后通过调整background-position来显示想要的图片，这样就可*

*以有效避免出现闪烁问题，这个技术在网页中应用十分广泛，被称*

*为CSS-Sprite，这种图我们称为雪碧图。*

*雪碧图使用步骤：*

*1、先确认要使用的图片*

*2、测量图标的大小*

*3、根据策略结构创建一个元素*

*4、将雪碧图设置为元素的背景图片*

*5、设置一个偏移量以显示正确的图标*

*雪碧图特点：*

*一次性将多个图片加载到网页，降低请求的次数，加快访问速度，提升用户体验*

*\*/*

11.线性渐变

*/\**

*通过渐变可以设置一些复杂的背景颜色，可以实现从一个颜色向其他颜色过渡的效果*

*注意：渐变是图片，需要通过background-image来设置*

*线性渐变，颜色沿着一条直线发生变化*

*-linear-gradient()*

*-线性渐变的开头，我们可以指定一个渐变的方向*

*-to left*

*-to right*

*-to bottom*

*-to top*

*-xxxdeg deg表示度数*

*-turn 表示圈*

*-渐变可以同时指定多个颜色，多个颜色默认情况下平均分布*

*也可以手动指定渐变的分布情况，在颜色后面加上像素*

*-repeating-linear-gradient();可以平铺的线性渐变*

12.径向渐变

*radial-gradient 径向渐变（放射性的效果）*

*默认情况下径向渐变圆心的形状根据元素的形状来计算的*

*正方形-->圆形*

*长方形-->椭圆形*

*我们也可以手动指定径向渐变的大小*

*circle 圆形*

*ellipse 椭圆形*

*也可以指定渐变的位置*

*语法：*

*radial-gradient（大小 at，位置，颜色 位置，颜色 位置，颜色 位置）*

*大小：*

*circle 圆形*

*ellipse 椭圆*

*closest-side 近边*

*closest-corner 近角*

*farthest-side 远边*

*farthest-corner 远角*

7.html补充

01.表格

*<!--*

*在现实生活中，我们经常需要使用表格来表示一些格式化的数据，*

*课程表，人名单，成绩单...*

*在网页中我们也需要使用表格，我们可以通过table标签来创建一个表格*

*-在table标签中使用 tr 表示表格中的一行，有几个 tr 就有几行*

*-在 tr 中使用 td 表示一个单元格，有几个 td 就有几个单元格*

*-colspan 橫向合并单元格*

*-rowspan 纵向合并单元格*

*-->*

*<!—*

*长表格：*

*可以将一个表格分成三个部分*

*-thead 头部*

*-tbody 主体*

*-tfoot 底部*

*-th表示头部的单元格*

*-->*

02.表格的样式

*/\**

*-border-spacing:指定边框之间的距离*

*-border-collapse：设置边框的合并（设置这个之后，border-spacing这个就没用了）*

*如果表格中没有使用tbody而是直接使用tr*

*那么浏览器会自动创建一个tbody，并且将tr全部放到tbody里面*

*所以，tr不是table的子元素，无法通过 table > tr: 选中tr 应该用 tbody > tr*

*默认情况下，元素在td中是垂直居中的 可以通过vertical-align来设置*

*\*/*

*/\* 隔行变色 \*/*

       tbody > tr:nth-child(2n) {

            background-color: *aqua*;

        }

03.表单

*<!--*

*表单：*

*-在现实生活中，表单用于提交数据*

*-在网页中也可以使用表单，网页中的表单用于将本地的数据提交给远程的服务器*

*-使用form标签来创建一个表单*

*form的属性：*

*-action：表单要提交的服务器地址*

*-->*

*<!--*

*-文本框：type="text"*

*-密码框：type="password"*

*-单选按钮：type="radio" 一组单选按钮需要相同的 name 属性，*

*而且必须要指定一个value属性，value属性最终会作为用户的填写的值传递给服务器*

*checked：可以将单选按钮设置为默认选中*

*-多选框：type="checkbox"*

*-下拉列表：select 下拉选项：option，selected：默认选中*

*-提交按钮：type="submit" value：设置按钮显示的文字*

*-->*

*<!-- button会用的更多，可以做一些更复杂的结构 -->*

*<!-- autocomplete="off" 关闭自动补全 -->*

*<!--*

*readonly 将表单项设置为只读 ，数据会提交*

*disabled  将表单项设置为禁用 ，数据不会提交*

*autofocus 设置表单项自动获取焦点，输入光标会自动在这里*

*-->*

8.animation

01.过渡

*/\**

*过渡(transition)*

*-通过过渡可以指定一个属性发生变化时的切换方式*

*-通过过渡可以创建一些非常好的效果，提升用户的体验*

*-transition-property：*

*指定要执行过渡的属性，多个属性间用逗号隔开，*

*如果所有属性都要过渡，就用all关键字。*

*大部分属性都支持过渡效果，注意，过渡时必须是*

*从一个有效值向另一个有效值进行过渡*

*-transition-duration：*

*指定过渡效果的执行时间*

*-transition-timing-function：*

*过渡的时序函数，指定过渡的执行方式*

*可选值：*

*ease，默认值，慢速开始，先加速，再减速*

*linear，匀速运动*

*ease-in，加速运动*

*ease-out，减速运动*

*ease-in-out，先加速，后减速*

*cubic-bezier() 贝塞尔曲线，*

*未指定时序函数，可以网上查找  https://cubic-bezier.com*

*eg:cubic-bezier(.24,.95,.82,-0.88)*

*steps，分布执行过渡效果*

*可以设置第一个第二个值：*

*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*

*-transition-delay:*

*过渡效果的延迟，等待一段时间后再执行过渡*

*-transition，简写属性，可以设置过渡相关的所有属性，只有一个要求，如果要写延迟，*

*则两个时间中第一个是持续时间 ，第二个是延迟时间*

*\*/*

02.动画

*/\**

*动画和过渡类似，都是可以实现一些动态的效果，*

*不同的是过渡需要再某个属性发生变化时才会被触发，*

*动画可以自动触发动态效果，更灵活*

*设置动画效果，必须先要设置一个关键帧，关键帧设置了动画执行的每一个步骤*

*@keyframes （设置关键帧，名字自己选）：*

*from动画的开始位置 也可以使用0%表示，*

*to表示动画的结束位置 也可以使用100%表示*

*animation-name，要对当前元素生效时的关键帧的名字*

*animation-duration，动画的执行时间*

*animation-delay，动画的延迟时间*

*animation-iteration-count，动画执行的次数*

*可选值：*

*次数，直接写数字*

*infinite，无限执行*

*animation-direction，指定动画运行的方向*

*可选值：*

*normal，默认值，从from向to运行，每次都是这样*

*reverse，从to向from运行，每次都是这样*

*alternate，从from向to运行，交替运行*

*alternate-reverse，从to向from运行，交替运行*

*animation-play-state，指定动画的执行状态*

*可选值：*

*running，默认值，动画执行*

*paused，动画暂停*

*animation-fill-mode，指定动画的填充模式*

*可选值：*

*none，默认值，动画执行完毕元素回到原来的位置*

*forwards，动画执行完毕元素会停止在动画结束的位置*

*backwards，动画延时等待时，元素就会处于开始位置*

*both，结合了上面两个特点*

*简写属性一样的。*

*\*/*

03.变形平移旋转

*/\**

*-变形就是指通过CSS来改变元素的形状或位置*

*变形不会影响到页面的布局*

*-transform，用来设置元素的变形效果,一个元素只有一个transform*

*-平移：(平移元素，百分比是相对于自身计算的)*

*-translateX()，沿着X轴方向平移*

*-translateY()，沿着Y轴方向平移*

*-translateZ()，沿着Z轴方向平移*

*-可以同时写多个，多个之间没有逗号*

*-eg：transform: translateX(-50%)translateY(-50%);*

*-Z轴平移：*

*调整元素在Z轴的位置，正常情况就是调整元素和人眼之间的距离，*

*距离越大，元素离人越近。*

*-Z轴平移属于立体效果（近大远小），一般情况下网页是不支持透视效果的，*

*如果需要看见效果，必须要设置网页的视距*

*\*/*

*/\* 设置当前网页的视距为800px，人眼和网页的距离 \*/*

             perspective: 800px;

注意：

*position: absolute;*

*这种居中方式只适用于元素的大小确定的*

*top: 0px;*

*bottom: 0px;*

*left: 0px;*

*right: 0px;*

*margin: auto;*

*/\**

*通过旋转可以使元素沿着X,Y,Z轴旋转指定的角度*

*-rotateX*

*-rotateY*

*-rotateZ*

*\*/*

*/\* 是否显示元素的背面 \*/*

*/\* backface-visibility: hidden; \*/*

*缩放：*

*/\**

*对元素进行缩放的函数：*

*scaleX：水平方向缩放*

*scaleY：竖直方向缩放*

*scale：双方向缩放*

*transform-origin:*

*变形的原点，默认值center*

*\*/*

9.less

01.less简介

 html {

*/\* css原生也支持变量的设置 \*/*

            --color:#aaa;

            --len:150px;

*/\**

*calc()计算函数*

*这两个功能都很强大，但是兼容性不好，例如IE*

*\*/*

        }

        .box1 {

            width: calc(300px/2);

            height: var(--len);

            background-color: var(--color);

        }

*<!--*

*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*

*-在less中添加了许多的新特性，像对变量的支持，对calc的支持...*

*-less的语法大体上和css语法一致，但是less中增强了许多对css的扩展*

*所以浏览器无法直接执行less代码，要执行必须先将less转换为css，然后再由浏览器执行*

*-->*

02.less中的变量

*//less中的单行注释，注释中的内容不会被解析到css中*

*/\**

*css中的注释，内容会被解析到css文件中*

*\*/*

*/\**

*变量，在变量中可以存储一个任意的值*

*-并且我们可以在需要时，任意的修改变量中的值*

*-变量名发生重名时，会优先使用比较近的变量*

*变量的语法：*

*@变量名*

*使用变量时：*

*-如果是直接使用则以 @变量名 的形式即可*

*-如果作为类名使用，或者一部分值使用时必须以 @{变量名} 的形式使用*

*\*/*

@a:100px;

@b:*red*;

@c:box6;

.box1{

    width: @a;

    color: @b;

    background-color: $*color*;

    height: $width;

}

.@{c}{

    width: @a;

    background-image: url('@{c}/1.png');

}

03.父元素和扩展

*/\**

*& 表示外层的父元素*

*\*/*

*// :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*;

}

*// 直接对指定的样式进行引用，这里就相当于将p1的样式复制到p3*

*// mixins 混合函数*

.p3{

    .p1;

}

*// 使用类选择器时可以在选择器后边添加一个括号，这时我们实际上就创建了一个mixins,在css中不会显示*

.p4(){

    width: 100px;

    height: *auto*;

}

*// 混合函数，在混合函数中可以直接设置变量，使用时给w传一个参数，更灵活（类似java）*

.test(@w,@co){

    height: @w;

    width: 200px;

    color: @co;

}

.div{

*// 调用混合函数，按顺序传递，也可以写上名字，就可以不按顺序；eg：@w：130px,@co:#bfa*

    .test(200px,#bfa);

*// less还自带了一些函数，average两个颜色取平均值*

    background-color: average(*red*,*yellow*);

}

补充

*// 在less中所有的数值都可以直接进行计算*

*// 加减乘除 + - \* /*

*// @import 引入其他less，方便维护，模块化*

@import "syntax.less";

10.Flex

01.弹性盒简介

*/\**

*flex（弹性盒，伸缩盒）*

*-是css中的又一种布局手段，它主要用来代替浮动来完成页面的布局*

*-flex可以使元素具有弹性，让元素可以跟随页面的大小的改变而改变*

*-弹性容器*

*-要使用弹性盒，必须先将一个元素设置为弹性容器*

*-我们通过 display 来设置弹性容器*

*-display：flex 设置为块级弹性容器*

*-display：inline-flex 设置为行内的弹性容器*

*-弹性元素*

*-弹性容器的子元素是弹性元素（弹性项）*

*-一个元素可以同时是弹性容器和弹性元素*

*-flex-direction 指定容器中弹性元素的排列方式*

*-可选值：*

*-row 默认值，弹性元素在容器中水平排列（左向右）*

*-主轴：自左向右*

*-row-reverse 弹性元素在容器中反向水平排列（右向左）*

*-主轴：自右向左*

*-column 弹性元素纵向排列（上向下）*

*-column-reverse 弹性元素反向纵向排列（下向上）*

*-主轴：*

*弹性元素的排列方向称为主轴*

*-侧轴：*

*与主轴垂直方向的称为侧轴*

*-弹性元素的属性：*

*-flex-grow：指定弹性元素的伸展系数 默认值：0*

*-当父元素有多余的空间时，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：指定弹性元素的伸缩系数 默认值：1*

*-当父元素的空间不足以容纳所有的子元素时，如何对子元素进行收缩*

*\*/*

02.弹性容器上的样式

*/\**

*-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 元素沿着侧着方向自动换行*

*-wrap-reverse 元素沿着侧轴反方向换行*

*-flex-flow：wrap和direction的简写属性*

*-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 元素沿着主轴的起边排列*

*-flex-end 元素沿着主轴的终边排列*

*-center 元素居中排列*

*-space-around 空白分布到元素两侧*

*-space-between 空白均匀分不到元素间*

*-space-evenly 空白分布到元素的单侧*

*\*/*

*/\**

*-align-items：元素在侧轴上如何对齐*

*-元素间的关系*

*-可选值：*

*-stretch 默认值 将元素的长度设置为相同的值，行与行之间*

*-flex-start 元素不会拉伸，沿着侧轴起边对齐*

*-flex-end 元素不会拉伸，沿着侧轴终边对齐*

*-center 居中对齐*

*-baseline 基线对齐*

*-align-content：侧轴空白空间的分配（和justify-content一样）*

*-align-self：在弹性元素设置的，用来覆盖当前弹性元素上的align-items*

*\*/*

03.弹性元素上的样式

*/\**

*-flex-grow: 弹性元素的增长系数*

*-flex-shrink: 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*

*- 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 可以设置弹性元素的所有的三个样式*

*flex：增长 缩减 基础；*

*-initial "flex:0 1 auto"*

*-auto "flex:1 1 auto"*

*-none "flex:0 0 auto" 弹性元素不缩减（没有弹性）*

*-order 设置弹性元素的排列方式（类似层级）*

*\*/*

04.像素

*<!--*

*-像素：*

*-屏幕是由一个个发光的小点构成的，这一个个的小点就是像素*

*-分辨率 1920\*1080 说得就是屏幕中小点的数量*

*-在前端开发中像素要分成两种情况讨论 CSS像素 和 物理像素*

*-物理像素：上述所说的小点就是物理像素*

*-CSS像素：编写网页时，我们所用像素都是CSS像素*

*-浏览器在显示网页时，需要将CSS像素转换为物理像素然后再呈现*

*-一个CSS像素最终由几个物理像素显示，由浏览器决定，*

*默认情况下在PC端，一个CSS像素 = 一个物理像素*

*-视口（viewport）*

*-视口就是屏幕中用来显示网页的区域*

*-可以通过查看视口大小来查看CSS像素和物理像素的比例*

*-默认情况下：*

*-视口宽度 1920px（CSS像素）*

*1920px（物理像素）*

*-此时，比例是 1：1*

*-放大两倍的情况：*

*-视口宽度 960px（CSS像素）*

*1920px（物理像素）*

*-此时，比例是 1：2*

*-我们可以通过改变视口的大小来改变比例*

*-->*

手机像素：

*<!--*

*在不同的屏幕，单位像素的大小是不同的，像素越小屏幕会越清晰*

*智能手机的像素点，远远小于计算机的像素点*

*问题，一个宽度为900px的网页如何在比这个网页小的手机中显示？*

*默认情况下，移动端的网页都会把视口设置为980px（CSS像素）*

*以确保pc端的网页可以在移动端正常访问，但是如果网页的宽度超过了980px*

*移动端的浏览器会自动对网页缩放以完整的显示网页*

*所以基本上大部分的pc端的网站都可以在移动端中正常浏览，但是往往都不会有一个好的体验*

*为了解决这个问题，大部分网站都会为移动端专门设置一个网站*

*-->*

05.完美视口

*<!--*

*默认情况下，移动端的网页都会把视口设置为980px（CSS像素）*

*像素比就是 980/移动端宽度*

*如果我们直接在网页中编写移动端代码，这样在980的视口下，像素比是非常不好的*

*导致网页中的内容往往非常小*

*编写移动端页面时，必须要确保有一个比较合理的像素比*

*可以通过meta标签来设置视口大小*

*每一款移动设备设计时，都会有一个最佳像素比，*

*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，我们称之为完美视口*

*设置完美视口：*

*<meta name="viewport" content="width=device-width">*

*-->*

06.vw单位

*<!--*

*不同设备的完美视口的大小是不一样的*

*iphone6 --375*

*iphone6plus --414*

*由于这个问题，所以同样的375个像素在不同的设备下意义是不一样的，*

*比如在iphone6中375是全屏，在plus就会缺一块*

*所以在移动端开发时，就不能再使用px来进行布局了*

*vw是一个新单位，表示的是视口的宽度（viewport width）*

*-100vw就等于一个视口的宽度*

*-1vw就等于百分之一视口宽度*

*vw这个单位永远相对于视口宽度进行计算*

*设计图的宽度*

*750px 1125px*

*创建一个48px\*35px大小的元素*

*6.4vw = 48px*

*4.667vw =35px*

*-->*

Vw适配：

*/\**

*网页中字体大小最小是12px，不能设置一个比12px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12px*

*\*/*

*<!--*

*rem*

*1 rem = 1 html的字体大小*

*-->*

07.媒体查询简介

*/\**

*使用媒体查询*

*语法：@media 查询规则{}*

*媒体类型：*

*all 所以设备*

*print 打印设备*

*screen 带屏幕的设备*

*speech 屏幕阅读器*

*-可以使用逗号连接多个媒体类型，这样它们之间就是一个或的关系*

*-可以在媒体类型前添加一个only，表示只有；only的使用主要是为了兼容一些老版本浏览器*

*\*/*

*<!--*

*响应式布局*

*-网页可以根据不同的设备或窗口大小呈现出不同的效果*

*-使用响应式布局，可以使一个网页适用于所有的设备*

*-响应式布局的关键就是媒体查询*

*-通过媒体查询，可以为不同的设备或设备不同的状态来分别设置样式*

*-->*

*<!--*

*响应式设计网页：*

*1.移动端优先*

*2.渐进增强*

*-->*

*/\**

*媒体特性：*

*width：视口的宽度*

*height：视口的高度*

*min-width:视口的最小宽度（视口大于指定宽度时生效）*

*max-width：视口的最大宽度（视口小于指定宽度时生效）*

*样式切换的分界点，我们称其为断点，也就是网页的样式会在这个点时发生变化*

*小于768 超小屏幕（max-width：768px）*

*大于768 小屏幕（min-width：768px）*

*大于992 中型屏幕（min-width：992px）*

*大于1200 大屏幕（min-width：1200px）*

*\*/*

        @media (min-width:500px),(max-width:300px) {

            body{

                background-color: *aqua*;

            }

        }

**陌风**

**7/31/2022**