CSS概述

1.什么是css

|  |
| --- |
| 层叠样式 |

2.作用

|  |
| --- |
| 修饰html元素，让元素变得好看 |

3.HTML和CSS的关系

|  |
| --- |
| Html:负责搭建网页结构，显示网页内容  Css负责网页的修饰 |

4.css与html属性之间的使用原则

|  |
| --- |
| W3c建议，尽量使用css的方式，来取代html属性  Css样式比属性更好维护，提高了代码的可维护性和可重用性 |

二、css的语法规范

①内联样式

|  |
| --- |
| 也叫行内样式  将css样式定义在元素的style属性中  <ANY style=”样式声明”> ANY：代表任意元素  样式声明： 样式属性:样式值  样式属性和值之间，用冒号  多个样式声明之间，用;隔开  <div style="color:red;font-size:24px">内联样式</div>  注意：特点①内联样式在项目中基本不用  原因：1.没有代码的重用性和可维护性  2.内联样式优先级最高 |

②内部样式

|  |
| --- |
| 在网页的头部<head>标签中添加<style></style>  在style标签中，定义该网页的所有样式  语法 <head>  <style>  选择器{样式声明;样式声明;….}  </style>  </head>  选择器：规范了，页面中哪些元素可以使用定义好的样式  div{}  span{}  a{}  内部样式，在项目中使用不多  可重用性只对当前页面有效，局限性很大  学习和测试使用较多 |

③外部样式

|  |
| --- |
| 单独创建一个.css文件，在此文件写样式  在.html文件的head标签中，使用link引入  <link rel="stylesheet" href="my.css">  href属性，引入的css资源路径  rel属性，引入的文件是什么，必须写  外部样式，是项目中使用最频繁的 |

④3个常用样式

|  |
| --- |
| 1、color 设置文本颜色  取值，合法的颜色值  2.font-size 设置字号大小  取值，px为单位的数字  3.background-color 设置背景颜色  取值，合法的颜色值 |

练习

|  |
| --- |
| 01\_ex.html  h1 lorem 内联，背景为pink,字体为yellow  h2 lorem 内部，背景为yellow字体为red  h3 lorem 外部，背景为red字体为blue |

2.css样式的特性

①继承性

|  |
| --- |
| 大部分的css效果是可以直接继承的  父元素的效果，作用在后代元素上 |

②层叠性

|  |
| --- |
| 可以为一个元素定义多个样式规则  每个样式规则中，样式属性不冲突的时候，可以同时作用在当前元素上 |

③优先级

|  |
| --- |
| 如果样式声明发生冲突，按照样式规则默认优先级去应用样式  默认优先级，由高到低  1.最高，内联样式  2.内部样式和外部样式，就近原则（谁离目标标签近，应用谁）  3.最低，浏览器默认样式  Ps:f12中的样式顺序，是按照默认优先级显示的 |

④调整优先级

|  |
| --- |
| !important 放在属性值之后，与值之间用空格分开  作用，调整样式的优先级  注意， !important的不能写在内联样式中  Ex: h4{background-color:pink !important;} |

练习

|  |
| --- |
| 02\_ex.html  P lorem  内部样式设置文字颜色blue,字号24px  外部样式设置文字颜色red,字号40px  将外部样式引入，f12查看页面效果  改变外部样式引入位置，f12查看效果  尝试使用!important调整样式优先级 |

三、基础选择器(重点)

1.选择器的作用

|  |
| --- |
| 规范了页面中哪些元素能够使用定义好的样式  选择器就是为了匹配元素（选择器就是一个条件，符合这个条件的元素就能使用定义的样式）  学习选择器，就是学习写条件的格式 |

2选择器详解

①通用选择器

|  |
| --- |
| \*{样式声明}项目中基本不用  唯一一种可能使用\*的方式  \*{margin:0;padding:0}所有元素内外边距清0 |

②元素选择器，标签选择器

|  |
| --- |
| div{}  p{}  页面中，所有对应元素/标签，都使用这个样式  设置页码中某种元素的公共样式 |

③id选择器，专属定制

|  |
| --- |
| 只对当前页中，一个标签生效  <any id=”id值”></any>  #id值{样式的声明}  项目中很少单独使用id选择器  通常会作为子代选择器和后代选择器的一部分 |

练习

|  |
| --- |
| 03\_ex.html  H2标签，lorem.id为text1  使用id选择器为text1设置背景为黄色，字体为purple  字体为斜体 font-style:italic;  再使用元素选择器，设置h2背景为红色，字体为white  坑 |

④类选择器

|  |
| --- |
| 作用：定义页面某个或者某类样式（公共样式，谁想用都可以用）  通过元素的class属性来引用  .d1{font-size:50px}  <div class=”d1”></div><span class=”d1”></span>  语法：  1.声明类选择器样式  .类名{样式声明}  注意：类名 1.点不能省略  2.类名不能以数字开头  3.类名只能包括- \_  2.引用  <any class=”类名”></any> |
| 多类选择器----一个元素引用多个类  <any class=”类1 类2 类3….”></any> |
| 分类选择器  1.将元素选择器和类选择器一起使用  更精准的确定元素的样式  语法：元素选择器.类选择器{样式声明}  引用此类名的这个元素，才能使用这个样式  2，将类选择器和另外要给类选择器一起使用  语法 .类选择器.类选择器{样式声明}  同时引用了这两个类选择器的元素，才能使用此样式  .dropdowm-menu.show{} |

练习

|  |
| --- |
| 04\_ex  div和p标签，内容假文  用类选择器为所有设置字体颜色为红色  再创建一个类选择器设置背景颜色为黄色  使用分类选择器设置p元素的字号为40px，同时背景色为黑色 |

⑤群组选择器

|  |
| --- |
| 将多个选择器放在一起定义公共样式  语法：选择器1,选择器2,选择器3,…{样式声明} |

⑥后代选择器

|  |
| --- |
| 后代：一级或多级的嵌套关系  通过元素的后代关系匹配元素  语法：选择器1 选择器2 选择器3 ….{样式声明}  div p span{color:red;} |

⑦子代选择器

|  |
| --- |
| 通过元素的子代关系匹配元素  子代：一级嵌套关系  语法：选择器1>选择器2>…{样式声明} |

⑧伪类选择器

|  |
| --- |
| 匹配元素不同状态的选择器  语法 选择器:伪类选择器{样式声明}  Love hate:  :link 匹配未访问过的状态  :visited 匹配已访问过的状态  :hover 匹配鼠标悬停在元素上的状态  :active 匹配元素被激活时的状态（鼠标店主元素不放开） |

练习

|  |
| --- |
| 06\_ex.html  一个a标签，内容随意，链接到http://www.sina,com  1.访问过后，文本orange  2.被激活时，文本颜色green  3.鼠标悬停时，文本颜色red  4.未被访问 文本颜色pink 没有下划线text-decoration:none;  坑：注意，4个伪类同时作用到1个元素上时，会发生冲突  解决方案，4个伪类需要按照一定顺序编写，就不会冲突  顺序：爱恨原则 love&hate  :link :visited :hover :active |
| :focus 匹配元素获取焦点的状态 |

⑨选择器权值

|  |
| --- |
| 权值：表示当前选择器的重要程度，权值越大优先级越高  !important >1000  内联样式 =1000  Id选择器 =100  类选择器 =10  伪类选择器 =10  元素选择器 =1  通用选择器\* =0  继承的样式 无  权值的特点  1.当一个选择器中含有多个选择器时，需要把所有的选择器的权值进行相加，然后比较，权值大的优先显示  2.权值相同，使用就近原则  3.群组选择器的权值单独计算，不能相加  4.样式后面添加!important，直接获取最高优先级  但是，内联样式不能添加!important  5.选择器的权值计算不会超过自己的最大数量级  （100个1加起来，也不会大于10  100个村民也打不过一个村长） |

四.尺寸和边框

1.尺寸属性

|  |
| --- |
| 设置元素的高度和宽度  width:  height:  取值：以px为单位的数字，%  max-width  max-height  min-width  min-height |

附加知识点，尺寸单位

|  |
| --- |
| 1.px像素  2.in 英寸 1in=2.54cm  3.pt 磅值，多用于设置字号大小 1pt=1/72in  4.cm 厘米  5.mm 毫米  6.em 相对父元素的单位乘以的倍数  7.rem 相对于html的单位乘以的倍数  8.% 相对于父元素的百分比 |

四.尺寸和边框

1.尺寸

②页面中允许设置尺寸的元素

|  |
| --- |
| 行内元素，设置宽高无效，尺寸是靠内容撑起来  自带宽高属性的元素，可以设置css尺寸，img table  块级元素 可以设置宽高  块级元素不设置宽，默认宽，宽度是父元素宽度的100%  不设置高，默认高，高度靠内容撑开，没内容高就是0  行内快，可以设置宽高  行内快不设置宽高，自带一个宽高尺寸，不同浏览器给的默认宽高不同 |

附加知识点

|  |
| --- |
| 由于不同浏览器，对元素默认提供的样式，有差异  在开之前，我们要把这些差异清除掉。  这个行为，叫做css reset,样式重置  Css reset 最经典的代码\*{margin:0;padding:0} |

③溢出处理

|  |
| --- |
| 当内容较大，元素内容区域小的时候，就会发生溢出效果，  默认，缺省情况，溢出部分显示，默认垂直溢出  Overflow:  取值：visible 默认值，溢出部分显示  hidden 溢出部分隐藏  scorll 不管是否溢出，水平和垂直都添加滚动条  auto 溢出的时候，在溢出方向添加滚动条 不溢出没有滚动条  overflow-x 设置水平滚动条  overflow-y 设置垂直滚动条  如果写水平溢出 |

练习

|  |
| --- |
| div#list>p>img\*6呆呆兵  横向溢出，有滚动条显示6张小兵 |

附加知识点，合法颜色值

|  |
| --- |
| 1.颜色的英文单词 red yellow…  2.#rrggbb 红绿蓝 6位16进制数字 #000000黑  #ffffff 白  3.#aabbcc--🡪#abc #000 #fff #f00 #0f0 #ccc #eee #666  4.rgb(0~255,0~255,0~255)  5.rgb(r%,g%,b%)  6.rgba(r,g,b,a)alpa:透明度0~1 0全透明，1不透明  7.hsl(270,100%,100%) |

3.边框

|  |
| --- |
| 简写方式  border:width style color;  wdth:边框的宽度  style:边框样式  最简方式border:style;默认1px黑色，必须写style  width:边框的宽度  style:边框样式  solid 实线  dotted 虚线，圆点  dashed 虚线，短线  double 双实线  color:边框颜色，合法的颜色值和transparent(等同于rgba(,,,0))  border:0/none;清除边框 |

练习

|  |
| --- |
| 03\_ex  创建一个div#d1,200px\*200px  设置4个方向的边框位1px实线orange  border:1px solid orange |

②单边设置

|  |
| --- |
| Border-top/right/bottom/left:width style color; |

③单属性设置

|  |
| --- |
| 同时设置4个方向的某一个属性  border-style:  border-width  border-color |

④单边单属性的设置

|  |
| --- |
| border-top-width  border-top-style  border-top-color  border-right-width  border- right - style  border- right - color  border-bottom-width  border- bottom - style  border- bottom - color  border-left-width  border- left - style  border- left - color |

⑤边框的倒角

|  |
| --- |
| 将直角倒成圆角  border-radius:  取值 1.px为单位的数字  2.% 50%就是圆 |

⑥圆角的单角定义

|  |
| --- |
| border-top-left-radius  border-top-right-radius  border-bottom-left-radius  border-bottom-right-radius |

⑦边框的阴影

|  |
| --- |
| box-show:h-shadow v-shadow blur spread color  最简方式 box-shadow:h-shadow v-shadow  h-shadow 水平方向阴影的偏移量  v-shadow 垂直方向阴影的偏移量  blur 阴影的模糊距离，值越大，越模糊  spread 阴影的尺寸  color 阴影的颜色  inset 将默认的外部阴影，变为内部阴影 |

练习

|  |
| --- |
| 03\_ex 写一个发光的球体 |

⑧轮廓

|  |
| --- |
| Outline:width style color;  Outline:0/none;清除轮廓  一般input需要清除轮廓 |

五.框模型---盒子模型

|  |
| --- |
| 盒子模型----元素在页面上实际的占地空间的计算方式  浏览器默认的元素实际占地宽度：  左外边框+左边框+左内边距+内容区域宽度+右内边距+右边框+右外边距  浏览器默认的元素实际占地高度：  上边外距+上边框+上内边距+内容区域宽度+下内边距+下边框+下外边距 |

1.外边框margin

|  |
| --- |
| 改变元素外边距，会产生位移效果  外边距margin:边框以外的距离（元素与元素之间的距离）  margin:value;同时设置4个方向的外边距  margin-top 上  margin-right 右  margin-bottom 下  margin-left 左  取值：1.以px为单位的数字  2.% 父元素宽度的%  3.正负值 margin-left:正值，元素向右移动  负值，元素向左移动  margin-top:正值，元素向下移动  负值，元素向上移动  4.auto 自动计算元素的左右外边距  让块级元素水平居中  auto对上下外边距无效  auto只对设置width的块级元素生效  margin取值的数量问题  margin:v1; 设置4个方向外边距  margin:v1 v2; v1:上下 v2:左右  margin:0 auto; margin:auto; 水平居中显示  margin:v1 v2 v3; 上 左右 下  margin:v1 v2 v3 v4 上右下左 |

练习

|  |
| --- |
| 04\_ex 坑  创建两个div，尺寸都是300px\*300px  背景颜色随意  设置两个div之间的间距为50px  1.单独设置一个div 50  2.每个div都设置25  3.一个div设置30，一个div设20 |

2.外边距的特殊效果

①外边距合并

|  |
| --- |
| 两个垂直外边距相遇时，他们将合并成一个  值以大的为准  解决方案：页面布局设计的时候，尽量的取规避 |

练习

|  |
| --- |
| 05\_ex  两个div,宽高200px,设置不同背景颜色  为两个div设置4个方向的外边距，f12观察效果  两个span,内容1111 2222,设置不同背景颜色  为两个span设置4个方向的外边距，f12观察效果  两个input  为两个input设置4个方向的外边距，f12观察效果 |

②关于块级元素，行内元素，行内块的总结

|  |  |  |
| --- | --- | --- |
| 块级 | 行内元素 | 行内块 |
| 设置宽高有效。如果不设置宽，宽度是父元素宽度的100%。如果不设置高，高度靠内容撑开 | 设置宽高无效。宽高靠内容撑开。 | 设置宽高有效。不设置，自带默认宽高。 |
| 独占一行 | 与其他行内元素/行内块共用一行 | 与其他行内元素/行内块共用一行 |
| 上下外边距有效 | 上下外边框无下限 | 上下外边框有效。但是，一行中一个行内块的上下外边框发生改变，整行元素，都会跟着一起移动 |

③默认自带外边距的元素

|  |
| --- |
| body h1~h6 ul ol dd p不同浏览器，给这些元素的默认外边距不同  所以一般开发之前，我们要做css reset  \*{margin:0;padding:0} |

练习

|  |
| --- |
| 06\_ex  两个div,id----🡪#d1,#d2  尺寸300px\*300px,设置不同的背景色 |

五.盒子模型

2.外边距的特殊效果

④外边距溢出

|  |
| --- |
| 在特殊情况下，为子元素添加上外边框，会作用到父元素  特殊情况：1.父元素没有上边框  2.设置上外边距的子元素的内容区域的上沿与父元素内容区域的上沿重合（为第一个子元素设置上外边距时，这种说法不严谨。）  解决方案：1.给父元素添加上边框，弊端，会影响父元素实际占地高度  2.给父元素添加上内边距  弊端，会影响父元素实际占地高度  3.给父元素设置overflow:auto/hidden  弊端，为了解决外边距溢出，而增加了新的限制  4.在父元素中，第一个子元素位置，添加一个空的table标签  <div id="d2">  <table></table>  <div id="d3"></div>  </div> |

⑤外边距

|  |
| --- |
| 1.元素位置发生改变，可以改变外边距  2.元素与元素之间有空隙，这个空隙用外边距 |

3.内边距padding

|  |
| --- |
| 边框与内容区域之间的距离  改变内边距，感觉上时改变了元素的大小  内边距有颜色，颜色同元素背景色  Padding:value;  Padding-top  Padding-right  Padding-bottom  Padding-left  取值数量  Padding:v1;  Padding:v1 v2; 上下 左右 但是padding没有auto  Padding:v1 v2 v3 上 左右 下  Padding:v1 v2 v3 v4 上右下左 |

4.box-sizing

|  |
| --- |
| 浏览器默认的盒子模型，元素实际占地宽度  左外边距+左边框+左内边距+内容区域宽度+有内边距+右边框+右外边距  Box-sizing:设置盒子模型的计算方式  1.默认公式 content-box  2.border-box  设置width是border+padding+内容区域的总宽度  实际占地宽度：左外边距+width+右外边距  如果元素的尺寸，使用%设置，你要留心，是不是需要设置border-box |

六.背景相关的样式

1.背景颜色

|  |
| --- |
| background-color：#ff0; |

2.背景图片

|  |
| --- |
| background-image:url(08.png); |

3.背景图片的平铺

|  |
| --- |
| Background-repeat:  取值 1.repeat 默认值 平铺  2.no-repeat 不平铺  3.repeat-x 水平平铺  4.repeat-y 垂直平铺 |

4.背景图片的定位

|  |
| --- |
| Background-position  取值 1.x y 以px为单位的数字，定义x轴和y轴的位置  2.x% y%  3.x或者x%，不写y，默认为50%  4.关键字x: left/center/right  y:top/center/bottom |

5.设置背景图片尺寸

|  |
| --- |
| background-size  取值：1.x y 以px为单位的数字  2.x% y%  3.只取一个值，同时设置宽高  4.cover 要求图片把容器全部覆盖，图片会显示不完整  5.contain 要求容器包含完整的图片。图片显示完整 但是容器会有空白区域 |

6.背景图片的固定

|  |
| --- |
| background-attachment  取值：1.scroll 默认值，背景图定位以容器为准  背景图会随着页面滚动条滚动而变化  2.fixed 固定，背景图定位以body为准  页面的滚动条，滚动，背景图不会随着发生位置变化  但是，只在本容器区域内显示 |

7.背景的简写方式

|  |
| --- |
| background:color image repeat attachment position  最简方式 background：color/image;  background:#f00;  background:url(..png);  注意：如果想在已有的背景设置中，添加或修改属性  不能使用简写方式修改，不然会覆盖之前的样式 |

七.渐变 gradient

1.什么是渐变

|  |
| --- |
| 渐变，就是指多种颜色，平缓变化的一种显示效果 |

2.渐变的主要因素

|  |
| --- |
| 色标：一种颜色，及其出现的位置  一个渐变，至少有两个色标 |

3.渐变的分类

①线性渐变 以直线的方式来填充渐变色

②径向渐变 以圆形的方式来填充渐变色

③重复渐变 将线性渐变，或者径向渐变，重复几次实现填充

4.线性渐变

|  |
| --- |
| background-image:  background-image:linear-gradient(方向,色标1,2,,……);  方向： 取值 1.to top  2.to right  3.to bottom  4.to left  2.角度 0deg== to top  90deg== to right  180deg== to bottom  270deg== to left  色标：颜色 位置  位置的写法 1. #f00 10px px为单位的数字  2. #f00 10% %  3.只有两种颜色，分别在0%和 100%，可以省略位置 |

5.径向渐变

|  |
| --- |
| Background-image:radial-gradient(半径 at 圆心,色标1……);  半径：以px为单位的数字  圆心：x y 以px为单位的数字  X% y%  关键字 x:left/center/right y:top/center/bottom  关于色标，注意，如果色标的位置，使用px为单位的具体数字。  那么这个径向渐变的半径，就失去作用了。 |

6.重复渐变

|  |
| --- |
| 重复的线性渐变  background-image:repeating-linear-gradient(方向,色标…);  重复的径向渐变  background-image:repeating-radial-gradient(半径,at圆心,色标…); |

7.渐变的浏览器兼容性问题

|  |
| --- |
| 如果想ie8以下，写兼容代码，写兼容的过程，css hack  第一步，添加浏览器内核  -webkit- chrome/safari  -moz- firefox  -ms- IE  -o- opera  注意：添加了浏览器内核的线性渐变，他的方向，只能写七点，不能加to  -webkit-linear-gradient(left,#f00 0%,#00f 100%);}  -ms-linear-gradient(left,#f00 0%,#00f 100%);}  -moz-linear-gradient(left,#f00 0%,#00f 100%);}  -o-linear-gradient(left,#f00 0%,#00f 100%);} |

八.文本格式化（重点）

1.字体属性

①字号大小

|  |
| --- |
| Font-size: 以px/pt/em/rem为单位的数字 |

②字体系列

|  |
| --- |
| font-family：系列1，系列2，系列3…….  取值：设备中字体库里有的字体，如果字体名称有空格，需要加””  用,隔开多个字体类型 |

③字体加粗

|  |
| --- |
| font-weight  取值1.关键字 lighter normal bold bolder  2.无单位的数字，必须是100的整倍数，最大值1000 |

④字体样式

|  |
| --- |
| font-style  取值：1.normal 默认值 正常 无样式  2.italic 斜体 |

⑤小型大写字母

|  |
| --- |
| font-variant:small-caps; |

⑥字体属性的简写方式

|  |
| --- |
| font:style variant weight size family;  最简方式:font:size family; |

2.文本属性text

①字体颜色

|  |
| --- |
| Color: |

②文本的水平对齐方式

|  |
| --- |
| text-align:left/center/right/justify两端对齐  text-align 设置的是当前元素的 子元素的水平对齐（子元素是块级无效）  margin:0 auto,是设置当前元素本身水平对齐 |

③行高

|  |
| --- |
| Line-height:  特点：如果行高大于字体大小，那么文字将在行高的中间显示，呈垂直居中  如果文字较多，一行放不下，不建议使用行高，文字会溢出  取值：1.以px为单位的数字，一般行高的取值与容器的高度相同，如果文字较多，一行放不下，不建议使用行高，文字会溢出  2.无单位的数字（整数和小数都可以）代表字号的倍数 |

④文本线条修饰

|  |
| --- |
| text-decoration:  取值：overline 上划线  underline 下划线  line-through 删除线  none 无线条 |

⑤首行缩进

|  |
| --- |
| Text-indent:以px为单位的数字 |

⑥文本阴影

|  |
| --- |
| text-shadow:h-shadow v-shadow blur color;  h-shadow 水平偏移  v-shadow 垂直偏移  blur 模糊度  color 阴影颜色 |

练习

|  |
| --- |
| 01\_ex  定义div,内容假文，有中文  1.设置字体系列  2.设置文字大小  3.加粗并斜体  4.所有小写字母转为小型大写字母  5.两端对齐  6.有下划线  7.首行缩进30px  8.文本垂直居中显示 |

九.表格相关的样式

①table的样式

|  |
| --- |
| 之前我们学习过的属性，基本都可以用  尺寸，边框，背景，字体，文本，内外边框 |

②td/th

|  |
| --- |
| 尺寸，边框，背景，字体，内边距都可以设置  但是，外边距无效  Vertical-align:top/middle/bottom |

2.表格的表现形式

|  |
| --- |
| table是一种特殊的显示方式  表格的实际尺寸，是根据内容数据多少决定的  单元格小，内容多，那么撑开  内容少，单元格设置的尺寸大，就按设置的尺寸展示  表格的特点  1.表格每一行，对应的列一样宽，其中一个变大，其他行的对应列跟着改变  2.一行中，某一列高度增加，当前行其它列的高度跟着增加 |

3.表格的专有属性

①边框合并

|  |
| --- |
| border-collspse:  取值：separate 默认值，边框分离  collspse 边框合并状态 |

②边框与边框的距离

|  |
| --- |
| Border-spacing:以px为单位的数字  取值数量 1个值，同时设置水平和垂直距离  2个值，第一个值设置水平，第二个值设置垂直  注意：如果边框是合并状态，此属性失效  所以必须保证border-collapse:separate; |

③标题的位置

|  |
| --- |
| Caption-side:top/bottom |

④设置table的显示规则

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| table-layout  取值：1.auto 默认值，自动布局的表格  2.fixed 固定布局的表格   |  |  | | --- | --- | | 自动布局 auto | 固定布局fixed | | 单元格的大小，匹配的内容和设置的尺寸，大的一方 | 单元格大小，取决于设置值 | | 先把所有读取到内存中一次性加载 | 浏览器默认加载方式 | | 表格复杂时，加载速度慢 | 在任何情况下，都是加速加载的 | | 自动布局比较灵活 | 固定表格布局，不够灵活 | | 适用于不确定每列大小，并且不复杂的表格 | 适用于确定每列尺寸的表格 | |

三.定位（重点\*\*\*\*\*\*\*\*\*\*）

改变元素在页面中的位置

1.分类

|  |
| --- |
| 1.普通流定位  2.浮动定位  3.相对定位  4.绝对定位  5.固定位 |

2.普通流定位,默认文档流

|  |
| --- |
| 每个元素，在页面都有自己的空间，  每个元素都是从父元素的左上角开始显示  块级元素按照从上到下的顺序逐个排列，独占一行  行内元素，是多个元素在一行中显示，从左往右逐个排列 |

3.浮动定位（重点\*\*\*\*\*\*\*\*\*\*\*\*）

|  |
| --- |
| Float::  取值：left 左浮动,让元素脱离文档流后，在当前行停靠在父元素的左侧，或者挨着之前已浮动元素  Right 右浮动，让元素脱离文档流后，在当前行停靠在父元素的右侧，或者挨着之前已浮动元素  None 默认值，不浮动  浮动的特点：  1.元素一旦浮动，就脱离文档流（不占页面空间，后面元素上前补位）  2.浮动元素会停靠在父元素的左边或者右边，或者在其他已浮动元素的边缘  3.父元素横向显示不下所有浮动元素时，最后显示不下元素，会自动换行。  4.浮动解决了多个块级元素横向显示的问题 |

4.浮动引发的特殊情况

①浮动元素的占位

|  |
| --- |
| 当父元素显示不下所有的浮动元素时，最后显示不下的元素会换行。  但是，已浮动元素会根据自己的浮动方向，占据位置，导致被挤下来的浮动元素，要在更下方显示，躲开被占据的位置。 |

②元素一旦浮动，如果此元素没有定义宽度，那么浮动后的元素宽度是靠内容撑开。

③元素一旦浮动，都会变成块级元素

|  |
| --- |
| 可以设置宽高，垂直外边距有效 |

④文本，行内元素，行内块，是不会被浮动元素压在下面的

而是，巧妙的避开，环绕着浮动元素显示

5.清除浮动效果 （清除浮动）

|  |
| --- |
| 清除之前浮动元素给我带来的影响  一个非浮动元素，之前有元素浮动，会这个浮动元素上前补位  清除浮动之后，这个元素就不去补位了  Clear:left/right/both/none |

四.显示

1.显示方式

|  |
| --- |
| 显示方式：决定了元素在网页中的表现形式（块级，行内，行内块，table）  取值：1.none 元素不显示，脱离文档流  2.block 让元素按照块级的方式显示  特点：独占一行，可设置宽高和上下外边距  3.inline 让元素按照行内的方式显示  特点:共用一行，宽高和上下外边距无效  4.inlin-block 让元素按照行内块的方式显示  特点:共用一行，但是可以设置宽高和上下外  5.table 让元素表现的和table一样  特点：尺寸以内容和设置宽度大的一个为准，独占一行，允许修改尺寸  项目中，修改显示方式，多用于  1.a标签设置按钮  2.显示/隐藏效果，都需要display |

2.显示效果

|  |
| --- |
| 显示/隐藏  visibility  取值：1.visible 可见的，默认值  2.hidden 隐藏  面试题：Visibility:hidden和display:none的区别  display:none;隐藏，脱离文档流，不占页面空间  visibility:hidden 隐藏，不脱离文档流，占据页面空间 |

3.透明度

|  |
| --- |
| Opacity  取值：0~1 1是不透明，0全透明  面试题：rgba和opacity的区别  rgba只会改变某一种颜色的透明度  opcity 作用于元素，此元素中（包括后代元素），只要是跟颜色相关的都会跟着变透明 |

4.垂直对齐方式

|  |
| --- |
| Vertical-align  只会在两个地方使用  1.table中  取值：top/middle/bottom  设置td中内容的垂直对齐方式  2.img中  设置img前后的文字，与img的对齐方式  Vertical-align 取值top/middle/bottom/baseline(默认)  通常会将所有图片的垂直对齐，设置为非基线的方式 |

5.光标的设置

|  |
| --- |
| Cursor:  取值：default箭头  Pointer 小手  Crosshair +  Text 文本  Wait 等待  Help 帮助 |

五.列表的样式

|  |
| --- |
| list-style-type:  取值：none/disc/circle/square  list-style-image:url(…..)  list-style-position:outside/inside  简写方式  List-style:type image position;  项目中使用最多的写法 list-style:none; |

六.定位---相对，绝对，固定定位

|  |
| --- |
| Position  取值：static 默认值，默认文档流，静态  relative 相对定位  absolue 绝对定位  fixed 固定定位  已定位元素：当一个元素，被position,并取值为relative,absolute,fixed其中一种时，这个元素就称为已定位元素  元素一旦变成已定位元素，会解锁4个偏移属性  top: +↓-↑  right: +← -→  bottom: +↑ -↓  left: +→ -← |

1.相对定位

|  |
| --- |
| position:relative;  配合4个偏移属性，实现位置的移动  特点1.相对原始位置偏移某个距离  2.相对定位不脱离文档流，后面元素不会上前补位  3.如果元素，只定义了position:relative,而没有写偏移属性,不影响任何布局效果，跟没写定位一样  使用相对定位的场合  1.元素位置微调，类似margin  2.为子元素做绝对定位时，提供参照物 |

2.绝对定位

|  |
| --- |
| Position:absolite;配合4个偏移属性，实现位置的移动  特点：  1.绝对定位时脱离文档流，后面元素会上前补位  2.绝对定位的偏移，  ①没有已定位祖先级元素，相对于body的左上角偏移！  ②绝对定位的元素，会相对于，离自己”最近的””已定位的””祖先级元素”去做偏移  脱离文档流，意味着4件事情  1.不占据页面空间  2.后续元素上前补位  3.变成块级  4.没写宽度，就靠内容撑开 |

3.固定定位

|  |
| --- |
| Position:fixed;配合偏移属性使用  将元素固定在页面上的某个位置，不会随着滚动条发生位置变化，一直固定在可视区域  特殊：脱离文档流 |

4.堆叠顺序

|  |
| --- |
| 默认：html后写的标签，会盖在之前的定位元素上  后写的堆叠顺序高  使用z-index:调整堆叠顺序  取值，整数，一般1000以下  数值越大，堆叠顺序越高  特点：1.默认，后定位的元素，堆叠顺序（设置的z-index:1,都比默认高）  2.定位的脱离文档流，和浮动的脱离文件流，不是一个体系  3.只有已定位元素，可以设置z-index;  4.父子级关系，z-index无效 |

CSS CORE

一.复杂选择器

1.兄弟选择器，只能往后找，不能往前找

兄弟：具有相同父元素的，平级元素之间，称为兄弟元素

①相邻兄弟选择器

|  |
| --- |
| 紧紧挨在某元素的兄弟元素  选择器1+选择器2{}  常用于，一组元素，刨除第一个以外，给其他元素设置样式 |

②通用兄弟选择器

|  |
| --- |
| 获取某元素后面所有符合要求的兄弟元素  选择器1~选择器2{} |

2.属性选择器 attr属性

|  |  |
| --- | --- |
| 允许通过元素所附带的属性及其值，来匹配元素，非常精准  1.[attr]{} 匹配所有带attr属性的元素  2.[attr1][attr2]….{}匹配所有带attr1属性，还带有attr2的元素  3.elem[attr]{} 匹配所有带attr属性的elem元素  4.[attr=value]匹配所有带attr属性，并且值为value的元素  Elem[attr1=value1][attr2=value2]{}   |  | | --- | | 5.模糊属性值  [attr^=value]{} 匹配属性值以value开头的元素  [attr$=value]{} 匹配属性值以value结尾的元素  [attr\*=value]{} 匹配属性值中有value的元素  [attr~=value]{} 匹配属性值中有value这个独立单词的元素 | |

3.伪类选择器

|  |
| --- |
| :link :visited :hover :active :focus |

①目标伪类

|  |
| --- |
| 让被激活的锚点，应用的样式  选择器：target{} |

练习，点击菜单，显示菜品图片

|  |
| --- |
| 02\_ex 菜单ul>li>a 菜单 img  海鲜面 四季春饼 八大碗 |

②结构伪类

|  |
| --- |
| 通过结构关系找到元素  1.elem:first-child{}  匹配的是elem元素，父元素的第一个孩子，第一个孩子，还必须是elem  2.elem：last-child{}  匹配的是elem元素，父元素的最后一个孩子，最后一个孩子，还必须是elem  3.elem:nth-child(n){} n:第几个孩子，从1开始  匹配的是elem元素，父元素的第n个孩子，这孩子，还必须是elem |

练习

|  |
| --- |
| 03\_ex 创建4\*4的表格，400px\*400px  通过结构伪类，设置样式  第一行背景色为#ff0  最后一行背景色为#ff0  第三行的第二列背景色为#0ff |

③:empty

|  |
| --- |
| 匹配内部没有任何元素的标签  连空格和回车都不能有，才叫空 |

④:only-child

|  |
| --- |
| 匹配属于其父元素的唯一子元素 |

⑤否定伪类

|  |
| --- |
| :not(selector){}  将满足selector条件的元素，都排除 |

⑥伪元素选择器

|  |
| --- |
| 1.:first-letter或者::first-letter  匹配该元素的首字符  2.:first-line或者::first-line  首字符和首行，样式冲突，遵循首字符的样式  3.::selection  匹配用户选取内容的样式，必须是两个双冒号  注意：只能修改文本颜色和背景颜色，其他样式无效 |

4.伪元素选择器---内容生成

|  |
| --- |
| 使用css命令，添加的html元素  1.::before或者:before  在当前元素的内容区域中，最前边，添加一个假的元素  我们可以给这个假的元素，定义显示方式display  还有其他样式，但是，这个假元素的内容，使用content添加  注意：content中，只能写字符串和url(img)  #d1::before{  display:inline;  color:#f00;  background:#ff0;  content:"蒙奇.D.路飞:";  width:100px;height:100px;  }  2.::after或者:after  在当前元素内容区域的最后，添加一个假元素 |

①伪元素内容生成，能够解决的问题-----外边距溢出问题

|  |
| --- |
| #parent:before{  content:"";  display:table;  }  这里的table不是table标签，而是把这个元素设置为table的显示方式 |

②伪元素内容生成，能够解决的问题----高度坍塌

|  |
| --- |
| .clearfix::after {  display: block;  clear: both;  content: "";  } |

二.弹性布局

1.什么是弹性布局

|  |
| --- |
| 弹性布局，是一种布局方式  主要解决，某个元素中，子元素的布局方式  为布局提供了很大的灵活性 |

2.弹性布局的概念

①容器

|  |
| --- |
| 要发生弹性布局的元素，他们的父元素称之为容器  就是设置了display:flex那个元素，就是容器 |

②项目

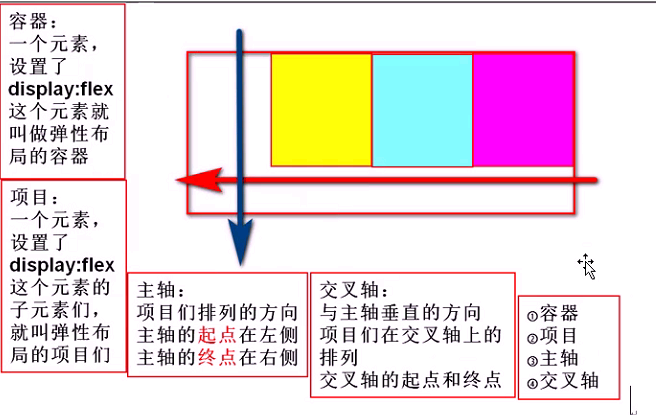
|  |
| --- |
| 要做弹性布局的子元素们，叫做项目  就是设置display:flex的元素，的子元素们 |

③主轴

|  |
| --- |
| 项目们排列的方向，叫做主轴  如果项目按照横向排列，主轴就是x轴  如果项目按照纵向排列，主轴就是y轴  项目在主轴上的排列顺序，主轴的起点和终点 |

④交叉轴

|  |
| --- |
| 与主轴垂直相交的一根轴  项目们在交叉轴上的排列方向，称之为交叉轴的起点和终点 |



3.语法

|  |
| --- |
| display  取值：1.flex 将块级元素设置为弹性容器  2.inline-flex 将行内元素设置为容器  将元素设置成弹性容器之后，这个元素所有的子元素，将变为弹性项目，这些项目，都允许按照弹性布局的方式排列  特点：  1.元素设置为容器之后，  容器的text-align,vertical-align失效  项目的float,clear失效  2.项目是可以修改尺寸的 |

4.容器的属性

①主轴的方向

|  |
| --- |
| flex-direction:  取值：1.row 默认值，主轴x轴，主轴起点左端  2.row-reverse 主轴是x轴，主轴起点右端  3.column 主轴是y轴，主轴起点是上端  4.column-reverse 主轴是y轴，主轴起点下端 |

②设置项目的换行

|  |
| --- |
| flex-wrap:  取值：1.nowrap 默认值，空间不够，项目进行压缩，不换行  2.wrap 空间不够项目就换行，往下挤  3.wrap-reverse空间不够项目就换行，往上挤 |

③上面两个属性合体

|  |
| --- |
| flex-flow:direction wrap; |

④定义项目在主轴上的对齐方式

|  |
| --- |
| jusitify-content:  取值：1.flex-start 主轴的起点对齐  2.flex-end 主轴的终点对齐  3.center 主轴中间对齐  4.space-between 两端对齐  5.space-around 两端有空白 |

⑤项目在交叉轴上的对齐方式

|  |
| --- |
| align-items  设置所有项目在交叉轴上的对齐方式  取值  1.flex-start 默认值，交叉轴起点对齐  2.flex-end 交叉轴终点  3.center 交叉轴中心  4.baseline 交叉轴的基线对齐，与起点一致  5.stretch 前提，项目不写高度。占满交叉轴上的所有空间 |

5.项目属性

①order

|  |
| --- |
| 定义项目在主轴上的排列顺序  值越小，越靠近起点，默认值为0  取值，无单位整数 |

②flex-grow

|  |
| --- |
| 定义项目的放大比例  如果容器有足够大的剩余空间，项目将变大  取值，无单位的整数，默认值0，不放大，取值越大，最后占的空间越大 |

③flex-shrink

|  |
| --- |
| 定义项目的缩小比例  如果容器剩余空间不够，项目将缩小  取值，无单位的整数，默认值1，取值越大，最后占的空间越小 |

④某一个项目在交叉轴的对齐方式

|  |
| --- |
| align-self:  取值  1.flex-start 交叉轴起点对齐  2.flex-end 交叉轴终点对齐  3.center 交叉轴中间对齐  4.baseline 基线对齐  5.stretch 项目不写高度，占满交叉轴  6.auto 使用容器定义的align-items的值 |

三.CSS hack

|  |
| --- |
| 由于不同浏览器对css的解析认知不同  会导致一份css在不同浏览器生成的页面效果不同  针对这种情况，开发人员要写不同代码取适应不同的浏览器  编写不同代码的过程叫做写css hack  -webkit-  -o-  -ms-  -moz-  tmooc有css hack的拓展视频 |

四.转换（重点\*\*\*\*）

1.什么是转换

|  |
| --- |
| 改变元素在页面中的位置，大小，角度，以及形状  2D, 只在x轴和y轴发生的转换效果  3D，增加了z轴的转换效果，页面上的3D效果都是假的 |

2.转换属性

|  |
| --- |
| transform:转换函数  取值：1.none 默认值，无任何转换  2.transform-function  转换函数，可以有1个或者多个转换函数 |

3.转换原点

|  |
| --- |
| transform-origin:  取值：1.以px为单位的数字  2.%  3.关键字 x(left/center/right) y(top/center/bottom)  取值数量：2个值，表示原点在x轴和y轴上  3个值，表示原点在x,y,z轴上  默认值，原点在元素中心位置(center center)/(50% 50%) |

4.2D转换

①位移

|  |
| --- |
| transform:translate();  取值：1.translate(x)元素在x轴上位移的距离  2.translate(x,y)  X: +→ -←  Y: +↓ -↑  3.translateX(x) 单独设置x轴  3.translateY(y) 单独设置y轴 |

②缩放

|  |
| --- |
| transform:scale();  取值：1.scale(value);一个值，同时缩放x轴和y轴的比例  value>1 x轴和y轴都放大  0<value<1 x轴和y轴都缩小  -1<value<0 x轴和y轴都缩小并反转  value<-1 x轴和y轴都放大并反转  2.scale(x,y);两个值，分别设置x轴和y轴的缩放比例  3.scaleX(x);单独设置x轴的缩放  4.scaleY(y);单独设置y轴的缩放 |

③旋转，改变元素的角度

|  |
| --- |
| transform:rotate(ndeg);  N:+ 顺时针 - 逆时针  坑：d1先旋转45度，然后位移500px  d2先位移500px,再旋转45度  注意：  1.转换原点，会影响旋转效果  2.旋转是连同坐标轴一起旋转的，会影响旋转之后的位移方向 |

④倾斜

|  |
| --- |
| transform:skew(ndeg);  取值：1.skew(ndeg);等同于skewX(ndeg);  让元素的y轴，向着x轴倾斜n度  n: + 逆时针   * 顺时针     2.skewY(ndeg);  让元素的x轴，向着y轴倾斜n度  n: + 顺时针   * 逆时针     3.skew(x,y); |

练习：

|  |
| --- |
| 01\_ex创建div 200px\*200px 背景随意，鼠标悬停时，该元素，向右偏移100px,向下偏移100px,旋转135deg,向x轴倾斜45deg,放大1.5倍 |

面试题

|  |
| --- |
| 使用css，让一个div在任何时候，都处于屏幕的正中心 |

5.3D转换（3D旋转）

①透视距离

|  |
| --- |
| 模拟人的眼睛，到3D转换元素之间的距离  距离不同，看到的效果也不同  perspective:距离  这个属性，要写在3D转换元素的父元素上 |

②3D旋转

|  |
| --- |
| transform  取值：1.rotateX(xdeg)  以x轴为中心轴，旋转元素的角度（烤羊腿，老式爆米花机）  2.rotateY(ydeg)  以y轴为中心轴，旋转元素的角度（旋转门，旋转木马）  3.roateZ(zdeg)  以z轴为中心轴，旋转元素的角度（电风扇，风车）  4.rotate3D(x,y,z,ndeg)  x,y,z取值为0，代表这个轴不参与旋转  >0,代表这个轴参与旋转 |

五.过渡

1.什么是过渡

|  |
| --- |
| 让css的值，在一段时间内平缓的变化  只有两个css状态的变化，使用过渡  超过两个css状态，使用动画 |

2.过渡的语法

|  |
| --- |
| 1.指定过渡属性  transition-property  取值:1.参与过渡的css属性名称，多个名称之间用空格分开  2.all 所有支持过渡的属性，都参与此次过渡  支持过渡的属性：  1.颜色属性  2.大多数取值为具体数字的属性  3.阴影  4.转换  5.visibility |
| 2.设置过渡的时长  transition-duration:  取值：以s/ms为单位 |
| 3.设置时间曲线函数  transition-timing-function  取值：1.ease 默认值 慢开，加速，慢关  2.linear 匀速  3.ease-in 一直加速  4.ease-out 一直减速  5.ease-in-out 先加速后减速 |
| 4.过渡的延迟  transition-delay:s/ms; |

练习

|  |
| --- |
| 翻滚吧，亮亮  爱的魔力转圈圈 |
| 5.过渡代码的编写位置  写在原来的选择器中，过渡效果有去有回  写在hover里，过渡效果有去无回 |
| 6.过渡的简写  transition:property duration timing-function delay;  最简方式transition:duration; |

六.动画

|  |
| --- |
| 1.超过两个样式之间的渐变，使用动画  2.不使用hover触发，刷新网页直接开始渐变，使用动画 |

1.关键帧

|  |
| --- |
| 控制动画的每一个状态  关键帧包括：1.动画的执行时间点  2.在这个时间点上的样式  0%{样式}  50%{样式} |

2.动画的使用步骤

①声明动画

|  |
| --- |
| @keyframes 动画名称{  0%{样式}  50%{样式}  .....  } |

②调用动画

|  |
| --- |
| 1.定义执行动画的动画名称  animation-name:change;  2.设置动画的持续时间  animation-duration:2s;  3.设置动画的时间曲线函数  animation-timing-function:ease/linear/ease-in/ease-out;  4.设置动画的延迟  animation-delay:2s; |

③动画的其他属性

|  |
| --- |
| 动画的播放次数  animation-iteration-count:  取值：具体的次数，数字  infinite 无限 |
| 动画的播放顺序  animation-direction:  取值1.mormal 默认值0%~100%  2.reverse 100%~0%  3.alternate 轮流播放  奇数次，正向，偶数次，逆向 |
| 动画的简写方式  Animation：name duration timing-function delay direction  最简方式 animation:name duration; |
| 设置动画的显示状态  在动画延迟时间内，显示的状态  在动画播放完成后，显示的状态  animation-fill-mode:  取值1.backwards 动画的延迟时间内，显示第一帧  2.forwards 动画结束后，显示最后一帧  3.both 上面两个都要  4.none 不填充，默认值 |
| 设置动画的播放和暂停  animation-play-state:  paused 暂停  running 播放 |

④浏览器对动画的兼容性问题

|  |
| --- |
| IE8.0以下需要些css hack  在定义动画的时候些css hack  @keyframes 动画名称{关键帧}  @-webkit-keyframes 动画名称{关键帧}  @-o-keyframes 动画名称{关键帧}  @-ms-keyframes 动画名称{关键帧}  @-moz-keyframes 动画名称{关键帧} |

3.项目中对动画的使用

|  |
| --- |
| 在html中link导入animation.css  使用animation调用动画名称 |

七.CSS优化

1.css优化的目的

|  |
| --- |
| 1.减少服务端的压力  2.提升用户体验 |

2.css优化的原则

|  |
| --- |
| 尽量减少http的请求个数（使用精灵图）  页面的顶部，引入css文件  将css和js放到单独的文件中 |

3.css代码优化

|  |
| --- |
| 合并样式（能写群组，就不单写，能用简写方式，就不单独定义属性）  缩小样式文件的大小（能重用代码，就重用）  避免出现空的href和src  代码压缩 |

八.CSS reset和normalize.css

1.什么是CSS reset

|  |
| --- |
| HTML标签，在浏览器中有默认样式  不同的浏览器内核，对默认样式的解析不同  在开始敲代码之前，把所有浏览器默认样式全部覆盖掉，这种行为就叫做css reset  使用css reset  1.自己写  2.使用别人写好的css reset文件  注意：css reset太霸道，把需要保留的默认样式，也覆盖了 |

2.normailize.css是什么

|  |
| --- |
| normailize.css在默认的html元素的样式上，提供了跨浏览器的高度一致性  相比于传统的css reset,normalize.css是一种现代的，为HTML准备，更优质的解决方案  normalize.css是css reset的替代方案  保护有用的浏览器样式，而不是完全的去覆盖  为大部分html元素提供了样式  修复了浏览器自身的一些bug并保证不同浏览器的一致性  优化css可用性：添加了一些小技巧  有详细的注释和文档 |

Bootstrap

一.响应是布局(css3 2010)

1.响应式布局是什么

|  |
| --- |
| Responsive web page 响应式/自适应网页(css3技术)  可以根据浏览设备不同（pc,pad,phone,print…）  而自动改变布局，图片，文字效果，不会影响用户体验 |

2.写响应式布局的要求

|  |
| --- |
| 1.布局，尽量对元素宽度，不写绝对值  必须是流式布局（默认文档流+浮动）  2.文字和图片大小随着容器的大小改变  3.使用css3提供的媒体查询技术  响应式布局存在问题：  1.代码量和复杂程度，几何性的增加  2.复杂的页面，不适合使用响应式 |

3.如何测试响应式网页

①使用真实设备测试

|  |
| --- |
| 好处，真实可靠  缺点，成本高，测试任务量巨大 |

②使用第三方的软件测试

|  |
| --- |
| 好处，不需要太多真实设备，测试方便  坏处，测试效果有效，有待进一步验证 |

③使用chrome的浏览器自带的模拟软件

|  |
| --- |
| 好处，简单方便  缺点，测试效果极其有限 |

4.编写响应式布局

①手机适配

|  |
| --- |
| 代码不需要背，你写的页面，如果不需要在移动端展示，下列代码不需要写  <meta name=”vie wport” content=”width=device-width,initial-scale=1.0,maximum-scale=1.0,user-scalable=0”>  width=device-width 设置视口宽度为设备宽度  initial-scale=1.0 设置视口宽度能否缩放1代表1倍，不缩放  maximum-scale=1.0 允许缩放的最大比例  user-scalable=0 是否允许用户手动缩放0代表不允许  最简方式  <meta name=”vie wport” content=”width=device-width,initial-scale=1”> |

②所有内容/文字/图片尽量使用相对尺寸，尽快不使用绝对值

③使用媒体查询，配合流式布局+弹性布局，完成响应式布局

④媒体查询技术

|  |
| --- |
| Css3 Media Query,做相应式的必备技术  媒体Media:浏览器页面的设备  设备：screen(pc/pad/phone)  TV  Print  媒体查询的原理：自动根据当前浏览设备的不同(尺寸，方向，解析度不同)，有选择性的，执行一部分css,忽略其他的css  语法  <style>  @media screen and (min-width:768px) and (max-width:991px){  选择器{}  选择器{}  }  </style> |
| 开发常用屏幕尺寸  超大屏 xl 大于等于1200px  大屏 lg 992px~1199px  中屏 md 768px~991px  小屏 sm 575px~767px  超小屏 xs 小于等于575px  响应式布局，基本上所有元素都要改变盒子模型计算方式  使用box-sizing:border-box; |

二.BookStrap

|  |
| --- |
| Boot给我们提供了好多封装的好样式，我们使用类名调用  但是项目中，需求比较繁琐，boot提供的样式不够，很多时候需要我们自己手写样式和手写媒体查询  https://www.boocss.com/  BootStrap分5个部分  1.如何使用boot  2.全局css  3.组件和js插件  4.定制css  5.boot项目 |

1.全局css样式

|  |
| --- |
| .container 定宽容器，不同屏幕尺寸的最大宽度写死了  .container-fluid 变宽容器，容器宽度永远是屏幕宽度的100%  Boot支持4个屏幕的响应式  lx-lg-md-sm |

①按钮相关的class

|  |
| --- |
| .btn 基本类  不同颜色按钮  .btn-success 绿色  .btn-warning 黄色  .btn-danger 红色  .btn-primary 主要的 蓝色  .btn-info 信息 青色  .btn-secondary 次要的 灰色  .btn-dark 深色  .btn-light 浅色  只有边框的按钮  .btn-outline-danger/warnind…  按钮的大小  .btn-sm 小按钮  .btn-lg 大按钮  .btn-block 块级按钮  .btn-link 链接 |

②图片相关的class

|  |
| --- |
| .rounded 0.25rem的圆角  .rounded-circle 50%圆角，圆形  Img-thumbnail 有内边距和边框，缩略图  .img-fluid 响应式图片，图片可以缩放，但是不会超过原始尺寸 |

③文字相关

|  |
| --- |
| 文本颜色  .text-danger….  .text-muted  文本大写，小写，首字母大写  text-uppercase/lowercase/capitalize  字体的加粗  font-weight-bold/light/normal  字体大小  .h1~.h6  文字水平对齐方式，带响应式  text-\*-left/center/right \*:xl/lg/md/sm  text-justify 没有响应式 |

④列表相关

|  |
| --- |
| .list-unstyled 去掉点，左内边距清0  ul.list-group  >li.list-group-item 创建列表组和列表项  列表项颜色  List-group-item-danger/warning…..  激活项 .active  禁用项 .disabled |

⑤表格相关class

|  |
| --- |
| .table 基本类，对table本身和table的后代写样式  .table-bordered 带边框的表格，为自己和后代添加边框  .table-dander/warning……表格的背景颜色  .table-hover 带鼠标悬停效果的表格  .table-striped 隔行变色  .table-reponsive-xl/lg/md/sm 响应式的表格，必须写在table的父元素上 |

二.BootStrap

1.全局css样式

2.辅助类

①边框

|  |
| --- |
| .border .border-top/right/bottom/left 基本类  .border-danger/warning… 边框颜色  .border-0 .border-top/right/bottom/left-0 清除边框 |

②浮动

|  |
| --- |
| float-\*-left/right/none; \*:xl/lg/md/sm  .clearfix  .clearfix 解决高度坍塌 |

③显示

|  |
| --- |
| .visible 显示  .invisible 隐藏 visibility:visible/hidden |

④背景颜色

|  |
| --- |
| bg-danger/warning/info…. |

⑤圆角

|  |
| --- |
| .rounded/.rounded-0/.rounded-circle  .rounded-top/right/left/bottom 单方向圆角，一次设置两个角 |

⑥内外边距

|  |
| --- |
| ml/ml/mt/mr/mb/mx/my-\*-0/1/2/3/4/5/auto  \*:xl/lg/md/sm  0 1:0.25rem 2:0.5rem 3:1rem 4:1.5rem 5:3rem  pl/pl/pt/pr/pb/px/py-\*-0/1/2/3/4/5 |

⑦尺寸

|  |
| --- |
| w/h/mw/mh-25/50/75/100 25% 50% 75% 100%  boot没有封装min-width |

3.栅格布局（重点\*\*\*\*\*\*）

①web页面的布局方式，一般三种

|  |  |  |
| --- | --- | --- |
| table布局 | div+css | boot中栅格 |
| 简单，容易控制 | 语义正确  渲染效率高 | 简单  容易控制  语义正确  渲染效率高  支持响应式 |
| 语义错误  渲染效率低 | 控制起来很麻烦 | 复杂页面不适用 |

②语法

|  |
| --- |
| Div.row>div.col-n n:1~12  .row 弹性，换行，主轴是x轴  带左右-15外边距  .col-n 带左右+15内边距  注意：  1.必须要放在.container或者.container-fluid  2.每一行都是一个row(弹性，主轴x,左右-15px外边距)  3.在row中声明col占几份（col:左右15px内边距） |

③响应式栅格

|  |
| --- |
| Col-\*-n \*:xl/lg/md/sm n:1~12 |

④不同屏幕有向上兼容的问题

|  |
| --- |
| Sm 有效的屏幕 sm/md/lg/xl  Md 有效的屏幕 md/lg/xl  Lg 有效的屏幕 lg/xl  Xl 只有xl有效 |

⑤.col类

|  |
| --- |
| 使用.col,不添加数组，自动的处理布局  每一个col平均分配，同一行的个数可以超过12个 |

⑥嵌套

|  |
| --- |
| <div class=”row”>  <div class=”col”>  <div class=”row”></div>  </div>  </div>  Row写直接子类只能是col ul的直接子类只能li |

⑦列偏移

|  |
| --- |
| Offset-n n:1~11 \*:xl/lg/md/sm |

4.弹性布局

|  |
| --- |
| d-\*-block/inline/inline-block/flex/table  响应式设置display  使用d-\*-flex/inline-flex 设置一个弹性容器 |

①设置主轴方向

|  |
| --- |
| flex-\*-row/column/row-reverse/column-reverse |

②项目在主轴上的对齐方式

|  |
| --- |
| justify-content-\*-start/end/center/between/around  boot对弹性布局，做了响应式的封装  让开发人员非常灵活的做响应式 |

5.表单

①表单元素的排列方向

|  |
| --- |
| .form-inline 内联表单，表单元素横向显示  .form-group 堆叠表单，表单元素纵向显示 |

②表单元素的样式

|  |
| --- |
| 关于文本框和密码框  .form-control 是文本框、密码框的基本类  块级，w100,字体，背景，边框，过渡  .col-fomr-label 设置内边距和字号大小  .col-fom-label-lg 大  .col-form-label-sm 小  关于checkbox,需要父子级，两个类配合使用  div.form-check 相对定位  >input.form-check-input 绝对定位，相对父级偏移  .form-text 块级，上外边距 |

注意

|  |
| --- |
| 使用栅格的原理，要求父容器  1.设置成弹性容器 display:flex;  2.主轴可以换换 flex-wrap：wrap;  满足这两个条件，容器就可以使用col所有的类 |

三.组件

|  |
| --- |
| boot封装好了一个一个的功能，不需要我们单独写js代码  可以直接实现交互 |

1.按钮组

|  |
| --- |
| 外层添加一个div,类名如下  .btn-group 横向按钮组 d-flex row  .btn-group-vertical 垂直按钮组 d-flex column  调整按钮组大小  .btn-group-lg/sm 调整内边距和字号  <**h3**>垂直按钮组(大)</**h3**> <**div class="btn-group-vertical bt-group-lg"**>  <**button class="btn btn-success"**>小鸡炖蘑菇</**button**>  <**button class="btn btn-danger"**>肉炖粉条</**button**>  <**button class="btn btn-warning"**>锅包肉</**button**>  <**button class="btn btn-primary"**>酸菜炖白肉</**button**> </**div**> <**br**/> <**hr**/> <**br**/> <**h3**>垂直按钮组（小）</**h3**> <**div class="btn-group-vertical btn-group-sm"**>  <**button class="btn btn-success"**>小鸡炖蘑菇</**button**>  <**button class="btn btn-danger"**>肉炖粉条</**button**>  <**button class="btn btn-warning"**>锅包肉</**button**>  <**button class="btn btn-primary"**>酸菜炖白肉</**button**> </**div**> <**br**/> <**hr**/> <**br**/> <**h3**>默认按钮组</**h3**> <**div class="btn-group"**>  <**button class="btn btn-success"**>小鸡炖蘑菇</**button**>  <**button class="btn btn-danger"**>肉炖粉条</**button**>  <**button class="btn btn-warning"**>锅包肉</**button**>  <**button class="btn btn-primary"**>酸菜炖白肉</**button**> </**div**> |

2.信息提示框

|  |
| --- |
| .alert 基本类  .alert-danger/warning/info…. 颜色  div.alert-dismissible 右边距4rem  >span.close 右浮动，配合使用控制小叉子的位置 |
| 事件 boot对事件的封装之后，我们调用的方式发生了改变  调用boot封装事件的步骤  1.使用自定义属性和值，调用事件（通过值决定调用的事件是什么操作）  给span添加自定义属性  <apan data-dismiss=”alert”></span>  2.确定事件发生后，改变的目标是谁  点击事件之后，改变的目标是直系的父系，不需要写确认目标代码 |

3.下拉菜单

|  |
| --- |
| 1.基本结构  div.dropdown> relative  >button.dropdown-toggle 添加向下的小箭头  +ul .dropdown-menu display-none  2.事件  button添加data-toggle=”dropdown”  事件目标 不需要写 |

4.导航

①水平导航

|  |
| --- |
| ul.nav x轴的弹性，所有弹性相关的样式，都可以设置  >li .nav-item 本身没有任何样式，需要父元素添加.nav-justified  .nav-justified与.nav-item配合  让每一个li等宽显示  >a.nav-link 块级，内边距和伪类的样式设置 |

②选项卡导航

|  |
| --- |
| 导航部分基本结构  ul.nav 添加nav-tabs 变成选项卡导航  li.nav-item  a.nav-link 添加a.active或者li.show让选项卡激活  内容部分的基本结构  div.tab-content>div.tab-pane  .tab-content与.tab-pane配合，让子div的display:none  子div添加active可以让当前内容显示 |
| 事件  1.事件的激活方式  给a标签添加自定义属性data-toggle=”tab”  2.事件的目标  div.tab-pane添加id ex:id=”tab1”  在a的href中绑定对应id  <a href=”#tab1”></a> |

练习

|  |
| --- |
| 选项卡导航 中国 美国 泰国 印度  内容自己写 |

③胶囊导航

|  |
| --- |
| 导航部分结构  ul.nav 添加.nav-pills变成胶囊导航  给a标签添加自定义属性data-toggle=”pill” |

5.导航栏

|  |
| --- |
| div.navbar 水平的弹性  添加.navbar-expand-\* 变成响应式导航栏  .navbar-expand-\*对父级div没有太多影响  >ul.navbar-nav 垂直弹性  配合父级.navbar-expand\_\*  在某个屏幕尺寸下把主轴变为水平  没有达到这个屏幕尺寸，主轴默认是垂直  形参响应是导航栏  >li.nav-item  >a.nav-link |

练习

1.栅格完成学子1L（不用定位）



2.重新复习，boot使用事件的方式

①按钮组

②信息提示框

④导航（水平导航，选项卡导航，胶囊导航）

⑤响应是导航栏

6.折叠

|  |
| --- |
| Button data-toggle=”collaspse” data-target=”#d1”  div.collapse#d1 内容部分，隐藏 |

7.卡片

|  |
| --- |
| 一个区域，内部内容是垂直排列，都可以使用卡片  div.card  >div.card-header  +div.card-body  +div.card-footer  a.card-link  注意事项  1.类collspse不能和.card-body在同一个元素  不然折叠的时候回卡顿，我们需要分层完成  2.及格折叠的部分，可以同时打开，而不能开启一个的时候，其他自动隐藏  解决方案，在3个card的外部，添加div#parent  在所有的div.collapse上添加时间  data-parent=”#parent”  这样就可以保证，在div#parent中只有一个打开状态 |

9.折叠导航栏

|  |
| --- |
| 结构  div.navbar①.navbar-expand-\*②.navbar-dark.bg-dark  >a.navbar-brand 不折叠的菜单首项  >button.navbar-toggler \*以上屏幕不显示，\*以下显示的按钮  >apsn.navbar-toggle-icon 按钮的三条线  +div.collspse③.navbar-collapse 折叠的内容  >ul.navbar-nav>li.nav-item>a.nav.link  ①.navbar-expand-\*  让按钮在\*以上的屏幕，不显示，在\*以下的屏幕显示  让ul中的li在\*以上的屏幕，横向显示  在\*以下的屏幕，垂直显示  ②.navbar-dark 对当前div没有任何样式  告诉子元素，这个导航栏是深色背景  .navbar-dark .navbar-band  .navbar-dark .navbar-toggler  .navbar-dark .nav-toggler-icon  .navbar-dark …..nav-link  都是浅色  ③.navbar-collapse 让折叠导航，打开的位置为导航栏下方 |

10.媒体对象

|  |
| --- |
| div.media>img+div.media-body |

11.轮播图

|  |
| --- |
| 轮播基本结构  1.轮播图片  div.carousel data-ride=”carousel”  >div.carousel-inner  >div.carousel-item.active 其中一个元素添加active  >img.w-100 |
| 2.左右箭头  div.carousel#demo  >a.carousel-control-prev  >span.carousel-control-prev-icon  +a.carousel-control-next  >span.carousel-control-next-icon  事件  a data-slide=”prev/next” href=”#demo” |
| 3.轮播指示器  div.carousel#demo  >ul.carousel-indicators 对后代li添加样式，宽高，背景，位置  >li.activedata-slide-to=”0/1/2/3”data-target=”#demo” |

12.模态框

|  |
| --- |
| Modal |

四.其他组件

1.巨幕

|  |
| --- |
| div.jumbotron 自带媒体查询 |

2.徽章（小按钮）

|  |
| --- |
| .badge 基本类  .badge-danger/warning…..颜色  .badge-pill 样式 |

3.分页

|  |
| --- |
| ul.pagination  >li.page-item .disabled/.active  >a.page-link |

4.面包屑导航

|  |
| --- |
| ul.breadcrumb>li.breadcrumb-item>a  需要重写样式，修改连接字符  .breadcrumb-item+.breadcrumb-item::before {content:">";} |

5.进度条

|  |
| --- |
| Div.progress 进度条的槽  >div.progress-bar 进度条 宽度设置进度，背景色设置进度条颜色  .progress-bar-striped 条纹进度条  Progress-bar-animated 带动画的进度条 |

**BootStrap重点：响应式+栅格布局+SCSS**

一、SCSS理论

|  |
| --- |
| SCSS在服务器运行  需要转换成.css文件，浏览器才能执行 |

1.css的缺点

|  |
| --- |
| 语法不够强大，没有变量和合理的代码重用机制  导致难以维护  我们需要使用动态的样式语言，赋予css新的特性  常见的动态css语言  1.scss/sass(scss兼容sass,scss更接近css基础语法  2.stylus  3.less |

2.scss是什么

|  |
| --- |
| scss是一款强化css的辅助工具  scss与css有高度相似  添加变量，嵌套，混合，导入，函数等高级功能  这些高级功能让css更加强大与优雅  浏览器无法解析运行scss文件，需要在服务器把scss文件转换成css |

二.scss的使用

1.转换工具安装

|  |
| --- |
| Scss在服务器使用  1.需要有nodejs解释器 v8.11以上  64位的scss需要V-10.0以上  2.在线安装 npm install –g node-sass  3.离线安装  ①scss包，解压后4个文件  ②找到nodejs安装路径，把上面4个文件复制进去  ③cmd输入node-sass-v,弹出版本号，说明安装成功 |

2.scss文件转换成css文件

①单文件转换

|  |
| --- |
| Node-sass scss文件路径 保存css文件路径  Ex:node-sass scss/1.scss css |

②多文件转换

|  |
| --- |
| node-sass scss文件夹 -o css 文件夹  ex: node-sass scss -o css |

③单文件监听

|  |
| --- |
| node-sass -w scss文件css文件  Ex:node-sass -w scss/1.scss css/1.css |

④多文件监听

|  |
| --- |
| node-sass -w scss文件夹 -o css文件夹  ex: node-sass -w scss -o css |

3.scss基础语法

①变量

|  |
| --- |
| 使用$表示变量，变量名可以保护字母，数字，- \_不能以数字开头，基本与css选择器相同，尽量做到见名知意  1.声明变量时，变量值可以引用其他变量  2.变量声明在{}外，整个scss文件都可以访问这个变量  声明在{}内，只有这个{}内部才能访问这个变量  3.！defalut规则  如果此变量在之前已经声明赋值了，那么使用之前的值  如果此变量之前没有声明赋值，使用现在的值 |

②嵌套

|  |
| --- |
| 1.根据scss |
| 2.伪类的嵌套  需要在:之前，添加占位符&。  如果不添加，会生成一个空格，变成后代选择器  **a**{  &:**hover**{**background**:**#00f**;} }  练习：使用a标签写学子商城的按钮  $myblue:#0aa1ed;  $font-color:#fff;  $myheight:2.5rem;  $mywidth:8rem;  Hover 样式随意 |
| 3.群组的嵌套  **nav**,**header**,**div**,**footer**{  **a**{**color**:**#f00**;} }  生成的css  **nav a**, **header a**, **div a**, **footer a** {  **color**: **#f00**; } |
| 4.属性嵌套  **div**{  **border**:{**width**:10**px**;**style**:**sold**;**color**:**#ff0**;} }  生成的css  **div** {  **border-width**: 10**px**;  **border-style**: **sold**;  **border-color**: **#ff0**; }  练习  一个200\*200px的div,红色背景  鼠标悬停时过渡到黄色背景，并且圆角  使用scss完成，过渡使用属性嵌套完成 |

③导入

|  |
| --- |
| Scss语法中，一个scss文件，如果用\_开头，说明这个scss是一个局部scss文件  局部scss的特点，不会转换成对应的css文件  一般在全局scss导入局部scss.  @import “局部文件名称” 不带下划线，不带后缀 |

④混合器

|  |
| --- |
| 把多个选择器都会使用的样式，封装进一个混合器  需要使用的时候，直接调用，实现代码的重用  混合器可以带参数  声明混合器  @mixin 混合器名称(参数1，参数2...){样式}  调用混合器  @include 混合器名称(参数1，参数2....)；  混合器使用最多的场合，编写css hack  **@mixin my\_bg**(**$fx**,**$color1**,**$color2**){  **background**:**-webkit-linear-gradient**(**$fx**,**$color1**,**$color2**);  **background**:**-o-linear-gradient**(**$fx**,**$color1**,**$color2**);  **background**:**-ms-linear-gradient**(**$fx**,**$color1**,**$color2**);  **background**:**-moz-linear-gradient**(**$fx**,**$color1**,**$color2**); } |

⑤继承

|  |
| --- |
| 一个选择器，可以继承另一个选择器的所有样式  使用@extend 选择器名称，继承这个选择器的样式  Css的代码解析为，群组选择器  练习  在选择器.my-border中定义1px solid transparent  设置宽度400px,左外边距20px  在选择器.my-bottom-border中继承.my-border  并设置底边框为红色，鼠标悬停底边框为蓝色 |

4.运算+ - /

|  |
| --- |
| 运算的时候，scss会自动的转换单位，进行运算  但是前提是几个单位可以转换in pt cm |

①加法 +

|  |
| --- |
| 字符串拼接的时候  如果用有引号的字符串去拼接无引号的字符串，结果是带引号  如果用无引号的字符串去拼接有引号的字符串，结果是无引号 |

②减法

|  |
| --- |
| 由于变量名称中可以使用  当减法前后有变量的时候，系统分不清 – 是属于变量名，还是单词的做减法，所有，需要减号前后空格，进行区分  Width:$my\_w - $my\_h; |

③除法

|  |
| --- |
| 在scss中， / 号作用，是除法和分隔符  在以下情况/会被解析成除法  1.如果计算式两边，有变量或者方法的返回值，视为除法  **width**:**$width**/2;  2.被()包裹的计算式，视为除法  **height**:(500**px**/2)  3.除法运算时，是其它算术运算式的一部分  5**px**+8**px**/2**px** |

④scss中的插值操作

|  |
| --- |
| **"liangliang ate** #{50+50} **baozis"** |

5.颜色的运算

|  |
| --- |
| #rrggbb和rgb(r,g,b)都是分段运算  红+红 绿+绿 蓝+蓝  Rgb(11,22,33)+rgb(22,33,44)=rgb(33,55,77)  Rgba的运算，要求参与运算的颜色的透明度必须相同 |

三.函数

1.可以直接在css语法中使用的函数

|  |
| --- |
| Rgba(red,green,blue,alpha)  hsl(hue,saturation,lightness)  hue:色调 0~360（0~120 121~240 241~360）  saturation:饱和度 0.0%~100.0%  lightness:亮度 0.0%~100.0% |

2.数学函数

|  |
| --- |
| round($v) 四舍五入  ceil($v) 向上取整  floor($v) 向下取整  min($v1,$v2….)  man($v1,$v2….)  random() 随机数 |

3.字符串函数

|  |
| --- |
| unquote($str) 去掉$str的双引号  quote 给$str添加双引号  to\_lower\_case($str) 变小写  to\_upper\_case($str) 变大写 |

4.自定义函数

|  |
| --- |
| **@function add**(**$a**,**$b**){  **@return $a**+**$b**; } **div**{  **width**:**add**(100,200); } |

四.指令

|  |
| --- |
| **@if （）**==**sun**{ } **@else if （）**==**moon**{ } **@else**{ }  注意，指令中条件部分，小括号可以去掉 |

使用boot完成学子商城项目，要求响应式

1.使用scss编写comm.css-----css reset

2.写heade.html

3.写footer.html

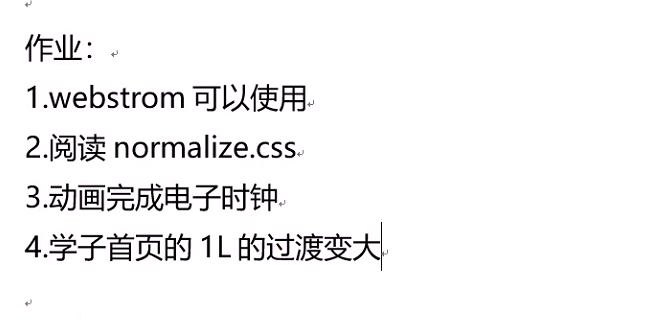
4.在主页使用iframe导入

作业：

1.手风琴

2.折叠导航栏

3.轮播图



样式编写步骤

|  |
| --- |
| 找目标元素，从上往下，从左往右 先写外后写里  1.先写尺寸和大体位置  2.边框和背景相关  3.文本相关  4.微调 |

|  |
| --- |
| div#parent 不写高，width:1100px,红色边框  这个div中有4张图片，每张图片的width为25%  看下效果 坑 |

作业：

1.ajax的注册模块

|  |
| --- |
| 01\_size  写两个span,内容111 222.有背景色，宽高200\*200  写两个div,内容111 222.有背景色，宽高200\*200  写两个input-text,宽高200\*200 |

