# HTML

day01

自我介绍:李然 Ran\_Lee0210

## 一.课程安排

1.HTML5 Basic (2天)

|  |
| --- |
| H5就是制作页面的标签,搭建网页的结构 |

2.ajax 异步数据交互(3天)

|  |
| --- |
| 异步完成前后端的数据交互,能发请求,能接响应  使用ajax就别用form. |

3.项目1(2天)

|  |
| --- |
| nodejs mysql ajax html  从前端页面实现对用户表数据的增删改查 |

4.CSS3 4+2天

|  |
| --- |
| 美化网页 |

5.Bootstrap 框架(4天)

|  |
| --- |
| 简化css开发  响应式网页 |

6.项目2(3天)

|  |
| --- |
| Bootstrap (简称boot) 完成学子商城官网 |

7.注意事项

|  |
| --- |
| 1.知识点细碎 ,没有条理  2.单词量巨大,关键有些单词很长  3.键速,提升  4.笔记,只抄有用的 |

## 二.web基础知识

|  |
| --- |
| Html:泛指前端网页技术  2014年9月,发布HTML5,代表大前端技术  H5:是html4.0和xhtml1.0的升级版  <img> <img/>  H5:升级之后, <img>或<img/> |

### 1.web与internet

|  |
| --- |
| internet:全球性计算机互联网  俗称:互联网,因特网,交互网,交际网  www服务:world wide web 万维网,访问网站就是www服务 |

### 2.internet上的应用程序

|  |
| --- |
| 1.C/S结构  c:client 客户端  s:server 服务器  ex:QQ 带客户端的网游  2.B/S结构  b:browser 浏览器  s:server  ex:所有的网站  3.C/S和B/S的区别  1.C/S需要升级  2.B/S不需要升级 |

### 3.web运行原理

|  |
| --- |
| internet:为web运行提供了网络环境  web:运行在internet上的一种B/S结构的应用程序,俗称网站  web运行原理:基于浏览器和服务器以及通信协议来实现的数据的传输和展示 |

#### ①通信协议

|  |
| --- |
| 规范了数据是如何打包和传递的 |

#### ②服务器

|  |
| --- |
| 1.功能:存储数据  接收请求,并给出响应  提供程序的运行环境  具备一定的安全功能  2.服务器产品  Tomcat  apache  IIS  3.服务器端的技术  1.java  2.php  3.net  4.nodejs  5.python |

#### ③浏览器

|  |
| --- |
| 1.功能:发送请求  把响应解析成图形界面给用户看  作为html.css.js的解析器  2.浏览器产品  chrome  safari  firefox  opera  ie---edga  3.浏览的技术  html5 css3 javascript |

## 三.HTML快速入门

### 1.什么是html

|  |
| --- |
| HTML: HyperText Markup Language  超文本 标记 语言 |

### 2.HTML的特点

|  |
| --- |
| 1.文件必须以.html或者.htm为后缀  2.由浏览器解析执行  3.有<>来标识  4.在页面中可以执行脚本语言 |

### 3.HTML基础语法

#### ①标记

|  |
| --- |
| 标记,标签,元素,对象,节点  <关键字></关键字>  关键字,是固定单词  标签是有功能的 |

#### ②标记分类

|  |
| --- |
| 1.双标记,有开始有结束  <标记>内容区域</标记> 成对出现  ex:<form></from>  2.单标记(空标记)  <标记> 或者 <标记/>  ex:<input> 或者 <input/> |

练习:

|  |
| --- |
| 01\_ex.html  创建一对html标记  一对head标记  一对body 标记 |

#### ③嵌套

|  |
| --- |
| 在一对标记中出现其他标记,形成功能叠加  <a href=""><b>111</b></a>  嵌套,必须有缩进,增加代码的可读性  嵌套的快捷键>  >代表嵌套 +代表兄弟 []属性 {}内容 .代表class #id |

#### ④属性和值对标记进行修饰

|  |
| --- |
| <div id="d1"></div>  1.属性必须声明在开始标签中  <标记 属性="值"></标记>  2.多个属性之间使用空格分开  <input type="text" value="123">  3.属性值可以添加"",' ',可以不带引号  习惯上加上双引号  4.属性,预定义属性,别人写好的属性名和值,我们背下来  自定义属性,自己定义属性名和属性值(忽略)  5.属性分类  通用属性,所有元素都生效的属性  (标准属性) id 元素在页面中的唯一标识  title 鼠标悬停在元素上显示的文本  style css中,定义内联样式  class css中,引用类选择器  专有属性,只有部分元素生效的属性 |

#### ⑤注释

|  |
| --- |
| 不被浏览器解析运行的内容  ctrl+/  <!-- --> |

总结

|  |
| --- |
| 学习固定的关键字,相关的功能  固定属性,固定值 |

## 四.HTML的文档结构

### 1.html文档类型的声明

|  |
| --- |
| <!doctype html>  告诉浏览器,本篇html文档,是使用h5的规则解析 |

### 2.网页结构

|  |
| --- |
| <html> 表示网页的开始和结束,有且只有一个  <head></head> 网页的头部,定义全局信息  <body></body> 网页的主体  </html> |

### 3. <head></head>

|  |
| --- |
| 网页的头部,定义全局信息,是其它头元素的容器  <title> </title> 网页的标题  <meta/> 元数据,定义网页的全局信息  <meta charset="utf-8"> 指定编码方式  <meta name="description" content=""> 描述  <meta name="Keywords" content=""> 关键字  <style></style> 定义网页的内部样式  <link> 引入外部样式  <script></script> 定义或者引入js |

### 4. <body></body>

|  |
| --- |
| 定义网页的主体  属性 bgcolor="black" 背景颜色  text="red" 字体颜色 |

## 五.文本标记(重点\*)

### 1.标题元素

|  |
| --- |
| 功能:醒目的显示文字  语法 <h1></h1> 一级标题  <h2></h2> 二级标题  <h3></h3> 三级标题  <h4></h4> 四级标题  <h5></h5> 五级标题  <h6></h6> 六级标题  特点:1.字号有变化,h1最大,h6最小  2.字体加粗  3.独占一行,上下有垂直间距  属性: align 设置文本的水平对齐方式  取值 left(默认缺省值)/center/right |

### 2.段落标记 paragraph

|  |
| --- |
| 功能:突出显示一段文本  <p></p>  特点:1.单独成行  2.上下有垂直间距  属性: align |

练习:02\_ex.html 模拟简历

### 3.水平分割线

|  |
| --- |
| 功能:页面添加一条横线  语法 <hr> 或 <hr/>  属性: size="50px" 粗细,以px为单位的数字  width="50%" 长短,以px为单位的数字,%  color="pink" 颜色,合法的颜色值  align="left" 水平对齐方式 left/center(缺省值)/right |

### 4.预格式化标签

|  |
| --- |
| 由于html有空格折叠现象  所有的空格和回车都会被浏览器解析成一个空格  使用<pre></pre>  可以把代码中的空格和回车全部保留,显示在页面上  单独成行 |

### 5.换行标签

|  |
| --- |
| <br/> 或者 <br> |

### 6.特殊字符(实体)

|  |
| --- |
| &nbsp; 空格  &lt; < less than  &gt; >  &times; 小叉叉  ¥ &yen;  ® &reg;  © &copy; |

### 7.文本样式标签

|  |
| --- |
| 加粗  <strong>111</strong>  <b>111</b>  斜体  <em>222</em>  <i>222</i>  删除线  <s>333</s>  <del>333</del>  下划线  <u></u>  尽量使用带语义的标签  <sub></sub> 下标  <sup></sup> 上标 |

杂谈:

1.笔记格式,模拟markdown笔记格式 markdown pad2

2.editplus快捷键

|  |
| --- |
| ctrl+d 删除当前行  ctrl+alt+↓ 复制当前行  alt+↓/↑ 移动当前行  lorem tab 生成假文 zc第2和第3项选中 |

作业:

1. nodejs6个接口,写明白

### 8.分区元素

#### ①块分区

|  |
| --- |
| 用来为页面做布局  <div></div>  单独成行 |

#### ②行分区

|  |
| --- |
| 同一行文字,有不同样式,用行分区处理  <span></span>  不独立成行 |

### 9.块级元素和行内元素

|  |
| --- |
| 所有元素,在页面显示,都有默认的显示规则  1.块级元素  单独成行,从上往下显示  ex:p h1~h6 pre hr div  2.行内元素  与其他行内元素共用一行  ex: span i em b strong u s del sup sub  3.行内块  input  4.table显示方式  <table></table> |

## 六.图像和链接(重点\*\*)

### 1.使用图像

|  |
| --- |
| <img> 或者 <img/>  <img src="图片路径">  属性src="资源路径URL" |

2.URL

|  |
| --- |
| Uniform Resource Locator 统一资源定位符,俗称:路径 |

### 2.URL表现形式

|  |
| --- |
| 1.绝对路径----完整的路径  协议+主机名称+目录结构+资源名称  使用网络资源的时候,使用绝对路径  使用网络资源的优点:节省存储空间  缺点:资源不稳定  本机资源项目中千千万万不能使用绝对路径 |
| 2.相对路径----参照物,是当前.html  ①同级兄弟元素,直接写资源名称  ②子集元素,使用/  ③父级元素,使用../ |

### 3.img的属性

|  |
| --- |
| src="" 图片的路径  alt="" 图片加载失败时,显示的文本  width="" 图片宽度,px为单位的数字  height=""图片高度,px为单位的数字  设置的宽高和图片原始比例不一致,会发生图片的失真  解决方法,图片只设置高或者宽,另外一个自动适应 |

### 4.链接

|  |
| --- |
| <a href=""></a>  属性: href 添加了href才有超链接的功能  指向要跳转的路径  target \_self 默认缺省,在当前页面打开新页面  \_blank 在新的页面打开链接 |
| a标签的其它表现方式  1.下载zip  <a href="1.zip">下载</a>  2.打开发送邮件的软件  <a href="mailto:chengliang@toufashao.com">发邮件</a>  3.返回页面顶部  <a href="#">返回页面顶部</a>  4.调用js代码  <a href="javascript:fn()">调用js</a> |

### 5.锚点

|  |
| --- |
| 锚点,就是页面中的一个记号  通过a标签,我们可以返回到这个记号的位置  1.定义锚点  html4: <a name="锚点名称"></a>  html5:用任意元素的id充当锚点名称  2.跳转到锚点  <a href="#锚点名称"></a> |

练习

|  |
| --- |
| 03\_ex.html  使用div定义两个锚点 d1 d2,div内容假文  通过a标签跳转到两个锚点  在通过一个a标签,跳转到03\_flag.html中一人之下 |

## 七.表格

|  |
| --- |
| 早期,表格用于布局,因为简单易操作  现在,表格布局已经淘汰,单纯用表格来显示数据 |

### 1.语法

|  |
| --- |
| <table>  <tr> ----table row  <td></td> ----table data  ......  </tr>  .....  </table>  做一个4\*4的表格,内容是1~16 14:52~15:07休息 |

### 2.属性

#### ①table

|  |
| --- |
| border="2px"  bordercolor="red"  bgcolor="gray"  width="400px"  height="400px"  align="center" 设置table自己的水平对齐方式  cellpadding="20px" 单元格内边距,边框到内容之间的距离  cellspacing="20px" 单元格外边距,边框与边框之间的距离 |

练习

|  |
| --- |
| 设置表格4\*4 400px\*400px  背景orange  边框1px blue  表格水平居中  单元格内边距5px  单元格外边距10px |

#### ②tr

|  |
| --- |
| bgcolor="orange"  align="right" 设置内部内容的水平对齐方式  valign="top"设置内部内容的垂直对齐方式 top/middle/bottom |

#### ③td/th

|  |
| --- |
| width="200px"  height="200px"  align="center"  valign="bottom"  bgcolor="yellow" |

#### ④浅谈table显示规则

|  |
| --- |
| 内容少,设置的宽高大,尺寸按照设置的宽高显示  内容多,设置的宽高小,尺寸按照内容的宽高显示  表格的渲染,先把表格中所有的数据读取到内存中  再一次性把表格画在页面上 |

### 3.不规则的表格使用

#### ①列合并

|  |
| --- |
| colspan="n"  从指定单元格的位置处开始,  横向向右合并n个单元格(n包含自己)  被合并的单元格要删除 |

#### ②行合并

|  |
| --- |
| rowspan="n"  从指定单元格的位置处开始,  纵向向下合并n个单元格(n包含自己)  被合并的单元格要删除 |

### 4.可选标记

#### ①标题

|  |
| --- |
| <caption> </caption>  如果设置标题,需要让caption标签紧跟着table标签 |

#### ②行/列标题

|  |
| --- |
| <th></th> |

### 5.表格复杂应用

#### ①行分组

|  |
| --- |
| 1.表头 <thead></thead> 表格最上面的一行或者几行  2.表主体<tbody></tbody>表格的实际内容(没写,浏览器自动添加)  3.表脚 <tfoot></tfoot>表格最下面的一行或者几行 |

## 八.列表

|  |
| --- |
| 有条理的显示数据  常用作网页布局 |

### 1.列表的组成

|  |
| --- |
| 列表类型+列表项  ①列表类型  有序列表 <ol></ol> order list  无序列表 <ul></ul> unorder list  ②列表项  <li></li> list item |

### 2.属性

|  |
| --- |
| ol type="I" 列表项的类型 1 a A i I  start="999"指定起始编号位置  ul type="none" disc 默认值,实心圆  circle 空心圆  square 实心方块  none 去点 |

### 3.列表的嵌套

|  |
| --- |
| 列表的嵌套,被嵌套的内容只能放在li中  <ul>  <li><img></li>  </ul> |

### 4.定义列表

|  |
| --- |
| 常用于给出一类事物或者对名词的解释说明  <dl>  <dt>名称 </dt>  <dd>名词解释 </dd>  </dl> |

## 九.结构标记

|  |
| --- |
| 用于描述整个网页结构,取代div做布局一套标签  使用div做布局,代码可读性非常差  使用结构标记,可以增强代码的可读性  百度搜索,结构标签比div排名高 |

### 1.常用的结构标记

|  |
| --- |
| <header></header>  定义网页头部,或者某个区域的头部  <footer></footer>  定义网页的脚部,或者某个区域的底部  <nav></nav>  定义网页导航部分  <section></section>  定义网页主体  <aside></aside>  定义侧边栏  <article></article>  定义与文本相关的内容 ex:回帖,评论 |

## 十.表单(重点\*\*\*\*\*)

### 1.作用

|  |
| --- |
| 1.提供可视化的输入控件  2.收集整理用户输入的数据,并提交给服务器  注意:使用ajax,就不使用form标签  form自带收集数据的功能  ajax需要手动收集数据(js--dom操作) |

### 2.表单的组成

|  |
| --- |
| 前端部分---提供表单控件,与用户交互的可视化控件  后端部分---后台接口对提交的数据进行处理 |

### 3.表单 <form></form>

#### ①action=""

|  |
| --- |
| 定义表单提交时发生的动作(后台接口)  缺省,默认提交给当前页 |

#### ②method=""

|  |
| --- |
| 设置请求的提交方法  缺省值:get  http原生提交方法:  get:明文提交  router.get(..,function(req,res){req.query.XXX},  提交的内容会在地址栏显示  向服务器要数据的时候,用get  提交的数据有大小限制 ,上限2kb  post:隐式提交,  router.post(..,function(req,res){req.body.XXX}  提交的内容不会再地址栏显示  给服务器数据的时候用post  提交的数据没有大小限制  delete 删除服务器上数据的方法  put 往服务器上传递数据  option 预请求 |

#### ③enctype=""

|  |
| --- |
| 指定表单数据的编码方式  允许将什么样数据传递给服务器  取值:1.缺省默认值  application/x-www-form-urlencoded  允许上传除了文件以外的所有字符  2.text/plain允许提交普通字符(不能有= & 等特殊符号)  3.multipart/form-data 允许提交文件给服务器 |

### 4.表单控件

|  |
| --- |
| form标签中,能够与用户做数据交互的可视化元素 |

#### ①分类

|  |
| --- |
| input元素 基础9中 h5新input 10种  textarea 多行文本域  select>option 下拉选择框  其它元素 |

#### ②input的公共属性

|  |
| --- |
| <input type="">  type 指定input的类型,不写type,或者单词拼错,默认是text  name 为控件定义名称,提供给服务器端使用.  如果没有name数据提交不了  ajax,input,除了单选和复选框,其它不需要使用name    value 用户输入的数据,传递给服务器的值  例外:所有按钮的value是不提交,  按钮的value是改变按钮的显示内容  disabled 禁用,只能看不能改,不能提交  无值属性,设置的时候,没有值,写上属性名就生效 |

### 5.input元素详解

#### ①文本框和密码框

|  |
| --- |
| 文本框 type="text"  密码框 type="password"  属性 maxlength 指定输入字符的最大长度  readonly 无值属性 只读,只能读不能改,可以提交  placeholder 占位提示符 |

#### ②按钮

|  |
| --- |
| type="submit" 提交按钮,将表单中的数据收集整理并提交  type="reset" 重置按钮,让表单恢复到最原始状态  type="button" 没有功能,配合事件,调用js方法  h5新出的元素<button onclick="fn()">登录</button>  代替了submit的功能,可以提交.  value属性,修改显示的文本 |

#### ③单选框/复选框

|  |
| --- |
| 单选按钮type="radio"  多选按钮 type="checkbox"  属性 name 除了给元素命名以外,还起到分组的作用  value 提交的值,如果不设置value,永远是on  checked 无值属性 默认选中 |

#### ④隐藏域

|  |
| --- |
| 数据不想暴露给用户,但是还需要提交,使用隐藏域  type="hidden" |

#### ⑤文件选择框

|  |
| --- |
| type="file"  前提: enctype="multipart/form-data" 可以传文件  method="post" 上传大小无限值  属性: multiple 无值属性,选择多个文件 |

### 6.textarea多行文本域

|  |
| --- |
| <textarea name="txt" cols="5" rows="3"></textarea>  cols="5"  rows="3" 根据计算硬件的不同,不准确 |

### 7.下拉选择框

|  |
| --- |
| select>option  提交的时候  如果option没有value属性,  选中的option的内容,当做select的value被提交  如果option有value属性,  选中的option的value,当做select的value被提交  属性:  select-----name value  size 如果>1 变成滚动选择框  multiple 多选  option---value  selected 无值属性,默认选中 |

### 8.其它元素

#### ①label

|  |
| --- |
| 功能:1.如果form中以后span,可用label代替span  2.关联文本与表单控件  <input id="auto" type="checkbox">  <label for="auto">自动登录</label> |

#### ②为控件分组

|  |
| --- |
| <fieldset> </fieldset> 为表单控件分组  <legend> </legend> 分组标题 |

#### ③浮动框架(重点\*\*)

|  |
| --- |
| 在一个html中,引入其它html页面  <iframe></iframe>  src="01\_form.html" 资源路径  frameborder="1px" 设置边框 0为无边框  scrolling="no" 设置滚动条 yes/no  width="100%" 宽度  height="600px" 高度 |

### 9.新表单元素,html5新提出的表单元素 10个

#### ①邮箱

|  |
| --- |
| <input type="email" name="email">  提交的时候,验证@前后有内容 |

#### ②搜索类型

|  |
| --- |
| <input type="search" name="search">  自带快速清除的按钮 |

#### ③url验证类型

|  |
| --- |
| <input type="url" name="url">  验证输入内容以http开头 |

#### ④电话号码

|  |
| --- |
| <input type="tel">  移动端使用,弹出虚拟键盘 |

#### ⑤数字类型

|  |
| --- |
| <input type="number" >  max="20" 最大值  min="10" 最小值  step="3" 步长 |

#### ⑥范围类型

|  |
| --- |
| <input type="range" max="20" min="10" step="3">  max="20" 最大值  min="10" 最小值  step="3" 步长  提供一个范围滑块 |

#### ⑦颜色类型

|  |
| --- |
| <input type="color">  弹出一个取色器 |

#### ⑧日期类型

|  |
| --- |
| <input type="date"> 提供一个日期选择控件 |

#### ⑨月份类型

|  |
| --- |
| <input type="month"> |

#### ⑩周类型

|  |
| --- |
| <input type="week"> |

# AJAX

## 一.HTTP协议

### 1.url

|  |
| --- |
| 结构:协议+主机名称+目录结构+文件名称  url完整的结构  <scheme>://<user>:<upwd>@<host>:<port>/<path>;  <params>?<query>#<flag> |

url完整结构详解

#### ①scheme

|  |
| --- |
| 方案,协议.以哪种方式获取服务器资源  不区分大小写. |

#### ②<user>:<upwd>

|  |
| --- |
| 用户名和密码 |

#### ③<host>

|  |
| --- |
| 主机名称  localhost 127.0.0.1 IP地址 域名 |

#### ④端口号<port>

|  |
| --- |
| 程序在计算机中,提供服务的窗口号 |

#### ⑤<path>

|  |
| --- |
| 资源在服务器上具体的路径结构 |

#### ⑥<params>

|  |
| --- |
| 参数,跟踪状态参数 session/cookie |

#### ⑦<query> get方法带参提交时的查询字符串

#### ⑧<flag>锚点

### 2.HTTP协议

|  |
| --- |
| HyperText Transfer protocol 超文本传输协议  规范了数据是如何打包传递的,专门用来传输html文件 |
| http协议的历史  C:\Users\web\Desktop\web1909\web第二阶段练习效果图示\HTTP协议历史.png |

### 3.web的请求原理

|  |
| --- |
| C:\Users\web\Desktop\web1909\web第二阶段练习效果图示\请求响应流程.png |

### 4.消息/报文 Message

|  |
| --- |
| C:\Users\web\Desktop\web1909\web第二阶段练习效果图示\message.png  1.request Message 请求消息(请求起始行,请求头,请求主体)  2.response Message响应消息(响应起始行,响应头,响应主体) |

### 5. request Message

|  |
| --- |
| 请求消息,客户端发送给服务器的数据块  由三部分组成: 请求起始行,请求头,请求主体 |

#### ①请求起始行

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1.请求方法   |  |  | | --- | --- | | 原生的http请求方法 | Restful Api | | get 客户端向服务器要数据的时候使用,靠地址栏传参,无请求主体 req.query.uname | get 查询,无请求主体,不用?传参 req.params.uname | | post 客户端传递数据给服务器,有请求主体,靠请求主体传递数据req.body.uname | post 增 有请求主体  req.body.uname | | delete 删除服务器上资源  无请求主体,跟get一样,有查询字符串 req.query.uname | delete 删除,  不用?传参  req.params.uname | | put 往服务器上放置资源  无请求主体,跟get一样,有查询字符串 req.query.uname | put 修改  有请求主体  req.body.uname |   2.协议版本号 HTTP/1.1  3.请求的url |

#### ②请求头

|  |
| --- |
| 1. Host: www.tmooc.cn  告诉服务器，我这个浏览器请求的是哪个地址  2. Connection: keep-alive  告诉服务器，开启持久连接  3. User-Agent:  告诉服务器,我现在这个浏览器的类型和版本号  4. Accept-Encoding: gzip  告诉服务器,我这个浏览器能够接受的压缩文件的类型  5. Accept-Language: zh-CN  告诉服务器,我这个浏览器能够接收的自然语言的类型  6. Referer: http://www.tmooc.cn/  告诉服务器,这个请求来自于哪个网页 |

#### ③请求主体

|  |
| --- |
| form data |

### 6.Response Message

|  |
| --- |
| 响应消息,服务器发送给客户端的数据块  由三部分组成,响应起始行 响应头 响应主体 |

#### ①响应起始行

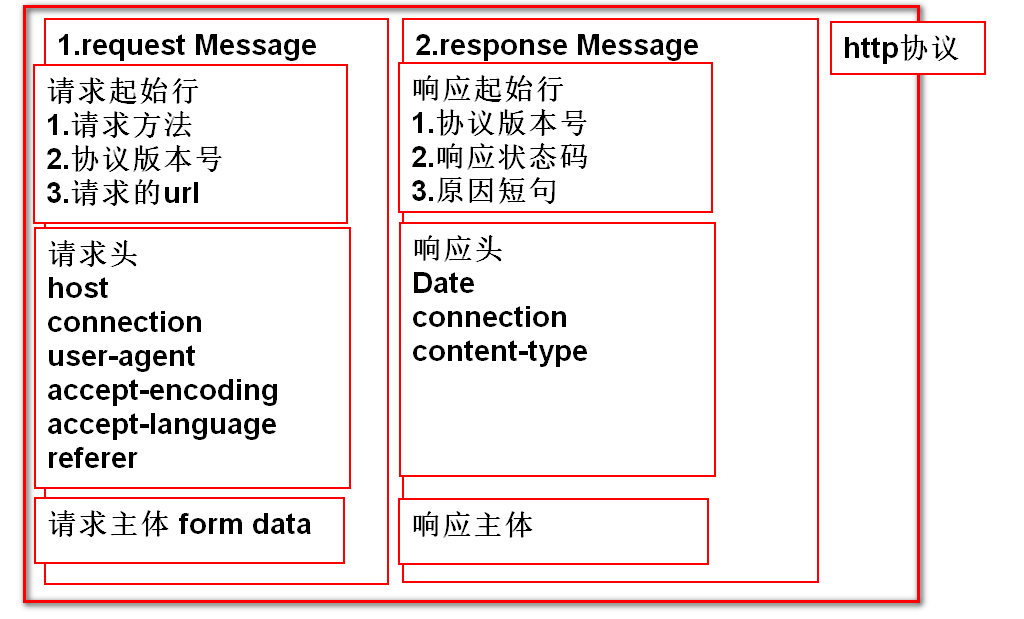
|  |
| --- |
| 1.协议版本号 http/1.1  2.响应状态码  1XX:正在请求,提示信息  2XX:200 响应成功  3XX: 301 永久重定向  302 临时重定向  304 命中缓存  4XX: 404 请求资源不存在(路径错误)  5XX:500 服务器代码错误  3.原因短句(对响应状态码的简短解释) |

#### ②响应头

|  |
| --- |
| 1. Date: Mon, 04 Nov 2019 07:34:59 GMT  服务器告诉浏览器,响应的时间  2. Connection: keep-alive  服务器告诉浏览器,已经开启了持久连接  3. Content-Type: text/html  告诉浏览器,响应主体的类型是什么  text/html html文本  text/css css文本  application/JavaScript js文本  image/jpeg png gif 图片  application/xml 响应主体是xml字符串  application/json 响应主体是json字符串 |

#### ③响应主体

总结:



### 7.缓存

|  |
| --- |
| 客户端将服务器响应回来的数据进行自动的保存  当再次发送同一个请求时,直接使用缓存数据 |
| 缓存的优点  1.减少冗余(rongyu)的数据出传输,节省客户端浏览器  2.节省服务器带宽  3.降低了对服务器资源的消耗和运行的要求  4.降低了由于远距离传输而造成的加载延迟 |
| 缓存能否命中的完整流程  C:\Users\web\Desktop\web1909\web第二阶段练习效果图示\缓存流程.png  1.请求--无缓存--连接服务器--存缓存--客户端得到  2.请求--有缓存--够新鲜--使用缓存--客户端得到  3.请求--有缓存--不新鲜--连服务器确认是否过期--没过期--更新缓存新鲜度--客户端得到  4.请求--有缓存--不新鲜--连服务器确认是否过期--过期--连服务器--存缓存--客户端得到 |
| 与缓存相关的消息头  Cache-Control: max-age=0  max-age=0 从服务器将文档传到客户端之时起  此文档处于新鲜的秒数  0 就是不缓存 3600就是1个小时的新鲜度 |
| 在网页中添加缓存  <meta http-equiv=" Cache-Control " content=" max-age=3600"> |

### 8.HTTP性能优化

|  |
| --- |
| http连接的过程  发起请求-->建立连接-->服务器处理请求-->访问资源-->构建响应-->发送响应-->记录日志 |

#### ①.http连接性能优化

|  |
| --- |
| 1.减少连接创建的次数(开启持久连接)  2.减少请求次数(合理编写css和js的文件)  3.尽可能减少响应数据的长度(代码压缩)  4.提高服务器端运行的速度 |

#### ②安全的http协议---https

|  |
| --- |
| SSL:为数据通信提供安全支持模块  浏览器发送请求---ssl层加密---服务器接收请求秘文---在ssl解密  服务器发送响应---ssl层加密---浏览器接收响应秘文---在ssl解密 |

## 二.简易DOM操作

|  |
| --- |
| ajax提交请求,不使用form  不使用form,就没有自动收集整理数据的能力  需要我们手写代码收集数据--------DOM操作 |

### 1.完整JavaScript的组成

|  |
| --- |
| 1.js核心 ECMA Script  2.DOM:document Object Model 文档对象模型  使用js动态的操作页面元素  3.BOM:Browser Object Model 浏览器对象模型  使用js动态的操作浏览器对象 |

### 2.使用dom

|  |
| --- |
| 1.绑定事件  <button onclick="fn()">提交</button>  2.在方法中,获取input对象  <input id="p1" type="text">  var obj1=document.getElementById("p1");  3.1获取/修改值 input才有值value  var uname= obj1.value; 获取  obj1.value="" 修改  3.2获取/修改内容 双标签才有内容 innerHTML |

|  |
| --- |
| 使用js操作页面的元素  1.获取元素对象  ①原生var obj=document.getElementById("目标id");  ②ES6规定,使用id可以直接当做对象  2.操作对象的值/内容  ①值,input对象才有值,obj.value  ②内容,双标签才有内容 obj.innerHTML |

### 3.innerHTML详解

|  |
| --- |
| 双标签的innerHTML中,可以赋值其它标签的字符串  动态的生成新的html结构 |

### 4.事件

|  |
| --- |
| 通过用户的行为,激发的操作,就是事件  单击事件onclick="fn()"  获取焦点 onfocus="fn()"  失去焦点 onblur="fn()"  onload="fn()" body的事件,页面加载完毕马上激活 |

## 三.ajax

### 1.同步Synchronous

|  |
| --- |
| 在一个任务进行的过程中,不能开启其他任务.  同步访问:浏览器在向服务器发送请求时,浏览器只能等待服务的响应,不能做其它任何事情  出现场合  1.地址栏输入url  2.a标签跳转  3.form提交请求 |

### 2.异步Asynchronous

|  |
| --- |
| 一个任务开启时,可以进行其它任务  异步的访问:浏览器在向服务器发送请求时,用户可以在页面上做其它操作  出现场合  1.注册时的用户名验证  2.股票软件  3.百度搜索建议  4.ajax |

### 3.什么是ajax

|  |
| --- |
| AJAX: Asynchronous javascript and xml  异步的 js 和 xml  本质:使用js提供的异步对象  异步的向服务器发送请求  并接收响应回来的数据  异步对象 XMLHttpRequest xhr |

### 4.使用ajax ,4步(小黄人4步曲)

|  |
| --- |
| 1.创建异步对象  var xhr=new XMLHttpRequest()  2.创建请求  xhr.open("method","url",isAsy);  method:字符串,请求的方法  url:字符串,请求的接口  isAsy:bool,是否使用异步的方式请求 true 是  3.发送请求  xhr.send(formdata) 如果是get/delete,没有请求主体,可以不填/null  4.创建监听,接收响应  监听的是xhr.readyState 表示xhr对象的请求状态  0:请求尚未初始化  1:已经连接服务器,请求正在发送  2:开始接收响应头信息  3:开始接收响应主体  4:接收响应数据成功,完毕  xhr.onreadystatechange=function(){  if(xhr.readyState==4 && xhr.status==200){  var result=xhr.responseText;  }  } |

#### 4.1.带参数的get

|  |
| --- |
| http原生get 跟服务器要数据的时候使用  xhr.open("get","url"+id.value,true) |
| restful的get 查询的时候使用 |

### 错误演示

#### 1.跨域

|  |
| --- |
| html文件使用本地模式打开的,  解决方案,把html文件放入服务器的静态资源文件夹  使用服务器访问页面的方式,打开这个html |

#### 2.打桩----不报错,没效果的情况

|  |
| --- |
| 在适当的位置,打印有意义的变量 |

#### 3.404

|  |
| --- |
| 1.查open中url  2.查接口中 router.get(url,function(){});  3.查app.js中app.use( '/ajax',ajaxRouter );  4.结构问题  5.系统硬件问题 |

#### 4.500

|  |
| --- |
| 查服务器接口代码 |

### 5.restful的get方法(同delete)

|  |
| --- |
| 后台  router.get("/login/:uname&:upwd",function(req,res){  //获取参数  var $uname=req.params.uname;  var $upwd=req.params.upwd;  });  前台  xhr.open("get","/ajax/login/"+$uname+"&"+$upwd,true); |

补全接口restful的get登录

创建04\_restful\_get.html 完成登录

|  |
| --- |
| 使用restful的get或者delete  不能在后台做非空验证  必须在前台做非空验证 |
| 提示:不管是http原生的请求方法,还是restful的请求  请求方法,是语义性,而不是语法  尽量符合语义 |

### 6.post/put

|  |
| --- |
| http原生和restful的post一模一样  //3.发送请求  //设置请求头信息,让请求可以传递所有字符  xhr.setRequestHeader("Content-Type","application/x-www-form-urlencoded");  //创建请求主体  var formdata="uname="+$uname+"&upwd="+$upwd;  //发送请求,带着请求主体发送  xhr.send(formdata);  注意:setRequestHeader这个方法,必须在open和send之间 |

## 四.数据格式

### 1.json

|  |
| --- |
| 本质就是一个string,所有又称json串  JavaScript object notation js对象表示法  以js对象的数据格式表现出来的字符串  后台数据在传递给前台的过程中,自动转换为json串 |
| json的格式  1.json中使用{}表示一个对象  2.json的属性,必须使用""括起来(写单引号不报错,但是推荐使用"")  3.属性值如果是字符串,也要加""  4.json本身是个字符串,所以最外层加一对单引号 |
| json解析  把json串变成js数组,以便可以使用for操作  var arr=JSON.parse(result); |

### 2.xml

|  |
| --- |
| eXtensible Markup language  可拓展 标记 语言  xml文件顶端定义声明  <?xml version="1.0" encoding="utf-8"?>  xml标签,只有双标签,区分大小写  xml标签可以嵌套  xml标签的属性值必须加""  每个xml文档,有且只有一个根元素 |

## 五.AJAX项目

|  |
| --- |
| restful API  get 查询  post 新增  put 修改  delete 删除  登录get url="/pro/v1/login/:uname&:upwd"  router.get("/v1/login/:uname&:upwd",funct(){});  查询所有用户get /pro/v1/getall  根据id删除用户delete /pro/v1/deluser/:uid  根据id查询用户get /pro/v1/searuser/:uid  根据id修改用户信息put /pro/v1/updateuser  根据uname查询用get /pro/v1/getUser/:uname  注册post /pro/v1/regUser  1.在routes文件夹中创建pro.js路由  2.在xz文件夹中创建pro文件夹  3.在app.js中,路由挂载,静态资源文件夹托管  4.登录接口,登录页面login.html  5.查询所有用户接口,页面getall.html |

# CSS

## 一.css3概述

### 1.什么是css

|  |
| --- |
| Cascading style sheets 层叠样式表,级联样式表,简称样式表 |

### 2.css的作用

|  |
| --- |
| 对html标签进行装饰,让页面变的好看 |

### 3.html与css 的关系

|  |
| --- |
| HTML:负责网页的结构,和内容的搭建  CSS:负责网页的修饰,样式的处理 |

### 4.css与html属性的使用规则

|  |
| --- |
| W3C建议我们尽量的使用css的方式来取代html属性  css可以是样式代码重用(有限)  方便后期维护,提高可维护性 |

## 二.css的语法规范

### 1.使用css样式的方式

#### ①行内样式(内联样式)

|  |
| --- |
| 将css样式定义在标签的style属性中  <any style="样式声明1;样式声明2;样式声明..."></any>  any:任意标签  样式声明: 样式属性:值;  多个样式声明之间使用;隔开  项目中基本不用行内样式,因为没有可重用性  只有在学习和测试中使用 |
| 常用样式  1.color:颜色单词 设置字体颜色  2.background-color:颜色单词 设置背景颜色  3.font-size: 以px为单位的数字,设置字号大小 |

#### ②内部样式

|  |
| --- |
| 在head标签中添加<style></style>标签  在style标签中定义该网页的样式  语法 选择器{样式声明1;样式声明2;.......}  项目中,内部样式使用较少,以为内部样式的重用,只在当前页面有效  学习和测试中,使用较多 |

#### ③外部样式

|  |
| --- |
| 单独创建一个css文件  在html的head使用link标签引入  <link rel="stylesheet" href="my.css">  外部样式,在项目中频繁使用 |

练习

|  |
| --- |
| 01\_ex.html  使用css3种方式完成样式  <h1>lorem</h1> 内联,背景为pink,字体yellow  <h2>lorem</h2> 内部,背景为yellow,字体red  <h3>lorem</h3> 外部,背景为red,字体blue |

### 2.CSS样式的特性

#### ①继承性

|  |
| --- |
| 大部分的样式可以直接被子元素继承 |

#### ②层叠性

|  |
| --- |
| 可以为一个元素定义多个样式规则  规则中属性不冲突,多个值都起作用 |

#### ③优先级

|  |
| --- |
| 如果样式声明发生属性冲突,按照样式的优先级规则来应用  优先级规则:高 内联样式  中 内部和外部样式,遵循就近原则  低 浏览器默认样式 |

#### ④调整优先级

|  |
| --- |
| !important规则  当样式值添加了!important之后  该样式优先级升为最高  注意,内联样式不能加!important |

练习

|  |
| --- |
| 02\_ex.html  p标签,内容假文  用内部样式设置文字颜色blue,字号24px  用外部样式设置文字颜色red,字号40px  将外部样式引入,f12查看效果  然后交换内部和外部的位置,f12查看效果  使用!important改变优先级,f12查看效果 |

## 三.基础选择器

### 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(符合选择器这个条件的元素,就能应用定义的样式) |

### 2.选择器详解

#### ①通用选择器

|  |
| --- |
| \*{样式声明}  由于\*号效率低下,有些项目经理不愿意使用  如果使用,只在一个地方用  \*{margin:0;padding:0}  所有元素内外边距清0 |

#### ②id选择器(专属定制)

|  |
| --- |
| <any id="id值"></any>  #id值{样式声明}  <div id="d1">Lorem ipsum dolor sit amet.</div>  #d1{color:red;} |

#### ③类选择器

|  |
| --- |
| 任何标签的class都可以调用的选择器  **.**类名{样式声明}  <any class="类名"></any> |
| 一个元素可以调用多个类名,多类选择器  <any class="类名1 类名2 类名3........"></any> |
| 分类选择器  ①元素选择器+类选择器{}  引用了这个类选择器的这个元素,应用这个样式  h3.text-red{background-color:green;}  更精确的匹配到元素  ②类选择器+类选择器{}  同时引用了这个两个类元素,应用此样式  .text-red.c1{background-color:orange;} |

#### ④元素选择器(标签选择器)

|  |
| --- |
| 元素名{}  定义页面中,此标签应用的公用样式  h3{}  p{}  div{} |

练习03\_ex.html

|  |
| --- |
| 页面中p和div,内容随意.用类选择器为所有元素设置字体红色  用分类选择器,为p设置背景黑色 |

#### ⑤群组选择器

|  |
| --- |
| 将多个选择器放在一起定义公共样式  选择器1,选择器2,选择器3,.....{样式声明}  #d1,.c1,div,p,.font-46{color:red;} |

#### ⑥后代选择器

|  |
| --- |
| 后代:一级或多于一级的嵌套关系  通过元素的后代关系来匹配元素  选择器1 选择器2 选择器3 ....{样式声明} |

#### ⑦子代选择器

|  |
| --- |
| 通过子代关系匹配元素  子代,一级嵌套关系  选择器1>选择器2>选择器3{}  子代和后代可以混写  选择器1>选择器2 选择器3{}  选择器1 选择器2>选择器3{} |

03\_ex

|  |
| --- |
| <div>  <span>11111</span>  <div>  <span>22222</span>  <div>  <span>33333</span>  </div>  </div>  </div>  所有span 红色  1111 黄色 2222 蓝色 3333绿色 |

#### ⑧伪类选择器

|  |
| --- |
| 匹配元素不同状态下的样式  :link{} 匹配链接未被访问的样式  : visited{}匹配链接访问过后的样式  :active{} 匹配元素被激活的状态  :hover{} 匹配鼠标悬停的状态  获取焦点的状态 :focus{} |

练习

|  |
| --- |
| 05\_ex 一个a标签内容随意  1.访问后,文本orange  2.被激活,文本green  3.鼠标悬停,文本红色  4.未被方访问,文本pink  坑:当4种状态,作用到同一个标签上时,要有编写顺序  爱恨原则 love hate |
| div lorem  给id=d1,class=c1  1.使用id选择器,设置div背景颜色yellow  2.使用类选择器,设置div 文本颜色红色  3.使用元素选择器,设置div 背景颜色 purple,字体颜色orange  4.给div添加hover,鼠标悬停时,背景颜色变为红色 |

#### ⑨选择器的权值

|  |
| --- |
| 权值,标识当前选择器的重要程度,权值高的优先级高  !important >1000  内联 1000  id 100  类/伪类 10  元素 1  \* 0  继承 无  总结:  1.当一个选择器中包含多个选择器时,需要将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,就近原则  3.群组选择器,权值不能相加,各算个的  4.样式后面添加了!important,直接拿到最高优先级  行内样式不能添加!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,加一起也不会大于10 |

## 四.尺寸和边框

### 1.尺寸属性

|  |
| --- |
| 作用,改变元素的宽度和高度  width: 以px为单位的数字,  %父元素的百分比  height: 以px为单位的数字  max-width:宽度可变的前提下,设置最大宽度  100%,图片可以缩放,但是最大不能超过本身默认尺寸  min-width: 宽度可变的前提下,设置最小宽度  max-height  min-height  如果,块级元素没设置宽度,默认宽度是父元素宽度100% |

附加知识点-------尺寸单位

|  |
| --- |
| 1.px像素  2.in 英寸 1in=2.54cm  3.pt 磅值 字号大小 1pt=1/72in  4.cm  5.mm  6.%  7.em 相对父元素的值乘以的倍数  8.rem 相对html的值乘以的倍数 |

### 2.页面中允许设置宽高的元素

|  |  |  |
| --- | --- | --- |
| 块级 | 行内 | 行内块 |
| 设置宽高有效 | 设置宽高无效 | 设置宽高有效 |
| 不设置宽,默认宽为父元素100%;  不设置高,默认高靠内容撑开,没有内容,就没有高 | 宽高靠内容撑起,没有内容,就没有宽高 | 默认自带一个宽高  但是,根据浏览器不同,这个宽高的值也不同 |
|  | 自带宽高属性的元素,可以设置宽高 |  |

附加知识点----颜色的取值

|  |
| --- |
| 合法的颜色值  1.颜色的单词 red green blue yellow black pink......  2.#rrggbb 每一个色段取值范围0~ff 相当于 0~255 |

作业:

1.css的单词

2.注册模块,登录模块,查询所有用户,删除,修改

### 1.1尺寸属性

附加知识点----颜色的取值

|  |
| --- |
| 合法的颜色值  1.颜色的单词 red green blue yellow black pink......  2.#rrggbb 每一个色段取值范围0~ff 相当于 0~255  3.#abc----->#aabbcc的简写  #f00 #0f0 #00f #ffa #faf #aff  4.rgb(0~255,0~255,0~255);  5.rgba(0~255,0~255,0~255,0~1) |

### 2.溢出

|  |
| --- |
| 当内容较大,元素区域较小,会发生溢出效果  默认都是纵向溢出  overflow: 默认缺省 visible 溢出部分可见  hidden 溢出部分隐藏  scroll 不管是否溢出,都有滚动条  auto 溢出时有滚动条,不溢出没有  overflow-x: 设置水平滚动条  overflow-y: 设置垂直滚动条 |

### 3.边框

#### ①边框的简写方式

|  |
| --- |
| border:width style color;  border:2px solid #f00;  最简方式:  border:style;  width:边框的宽度 以px为单位的数字,缺省值1.5px  color:边框的颜色 合法的颜色值, transparent,缺省值#000  style:边框的样式 solid 实线  dotted 点点虚线  dashed 断线虚线  double 双实线 |

练习

|  |
| --- |
| 一个div,尺寸300px\*300px  设置4个方向的边框为1px,点虚线,orange的颜色 |

#### ②单边的定义

|  |
| --- |
| border-top:30px dotted #00f;  border-right:30px dashed #00f;  border-bottom:30px solid #00f;  border-left:30px double #00f; |

#### ③单属性定义

|  |
| --- |
| 单独定义4条边的某一个属性  border-color:#f00;  border-width:100px;  border-style:dotted; |

#### ④单边单属性定义

|  |
| --- |
| border-top/right/bottom/left-width/style/color:  ex:  border-top-style:solid;  border-right-color:#0ff; |

### 4.边框的倒角(把直角倒成圆角)

|  |
| --- |
| 圆角不依赖边框  border-radius:50%;  取值:以px为单位的数字  % 50%就是圆  单角的设置,先写上下,再写左右  border-top-right-radius:20%;  border-bottom-left-radius:20%;  border-top-left-radius:20%;  border-bottom-right-radius:20%; |

### 5.边框阴影

|  |
| --- |
| 阴影不依赖边框  box-shadow:h-shadow v-shadow blur spread color;  h-shadow 阴影的水平偏移  v-shadow 阴影的垂直偏移  blur 阴影的模糊距离  spread 阴影的大小  color 阴影的颜色  inset 将默认的外部阴影变为内部阴影 |

### 6.轮廓

|  |
| --- |
| 边框外面的一圈线  outline:width style color;  outline:0/none;清除轮廓  border:0/none; |

## 五.框模型

|  |
| --- |
| 框模型-盒子模型,元素在页面上实际占地空间的计算方式  一切元素皆为框----所有元素都有占地空间  浏览器默认的元素实际占地宽度=  左外边距+左边框+左内边距+内容区域宽度+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=  上外边距+上边框+上内边距+内容区域高度+下右内边距+下边框+下外边距    margin:边框以外的距离(元素与元素之间的距离)  padding:边框与内容之间的距离 |

### 1.外边距,margin

|  |
| --- |
| 改变外边距,元素有位移效果  ①语法  margin:v1; 设置4个方向外边距  margin-top:  margin-right  margin-bottom  margin-left  当左右外边距,或者上下外边距冲突时,优先应用左和上  取值 以px为单位的数字  % 父元素宽度的百分比  auto 让元素水平居中,对上下外边距无效margin:auto;  只对设置了宽度的元素有效  ②简写方式  margin:v1; 设置4个方向  margin:v1 v2; v1:上下 v2:左右  margin:0 auto;  margin:v1 v2 v3; v1:上 v2:左右 v3:下  margin:v1 v2 v3 v4; 上右下左 |

练习03\_ex

|  |
| --- |
| 创建两个div,都是200px\*200px,背景颜色随意  设置两个div之间的间距为50px  三种方式设置  ①只设置上面div的外边距  ②只设置下面div的外边距  ③两个div都设置外边距 |

### 2.外边距的特殊效果

#### ①外边距的合并

|  |
| --- |
| 两个垂直外边距相遇时,他们会合并成一个  值以大的为准  解决方案,单独写某一个元素的上或者下外边距,  在设置的时候就要规避 |

练习04\_ex

|  |
| --- |
| 页面中两个div,宽高200px,设置不同背景颜色  页面中两个span,内容随意,设置不同背景颜色  页面中两个input,  为上述6个元素添加4个方向的外边距,看效果 |

### A.关于块级元素,行内元素,行内块的总结

|  |  |  |
| --- | --- | --- |
| 块级 | 行内 | 行内块 |
| 独占一行 | 与其他行内或者行内块共用一行 | 与其他行内块或者行内共用一行 |
| 可以设置宽高 | 设置宽高无效  靠内容撑开 | 可以设置宽高 |
| 默认宽,父元素100%  默认高,内容撑开 | 默认宽高,内容撑开 | 自带一个默认宽高,不同浏览器宽高不同 |
| 4个方向外边距都有效 | 上下外边距无效 | 4个方向外边距有效  改变垂直外边距,同行其它元素跟着一起动 |

### 3.自带外边距的元素

|  |
| --- |
| h1~h6 p body ol ul pre  不同浏览器,对这些元素的默认外边距的解析可能不同  所以开发之前,经常会把这些默认样式清空  \*{margin:0;padding:0} css reset  body margin:8px;  ul margin:16px 0;padding-left:40px; |

练习05\_ex

|  |
| --- |
| 两个div,200px\*200px,不同背景色,#d1,#d2  在#d2中添加div#d3,设置背景色,宽高100px  给#d3设置上外边距,观察效果 坑!!!! |

### 4.外边距溢出

|  |
| --- |
| 在特殊情况下,为子元素设置上外边距,会作用到父元素上  特殊情况:1.父元素不能有上边框  2.(为第一个子元素设置上外边距时,会溢出,不严谨)  子元素的内容区域上边沿与父元素内容区域上边沿重合  解决方案:1.给父元素设置上边框  弊端:父元素实际占地高度增加了  2.父元素添加上内边距  弊端:父元素实际占地高度增加了  3.父元素添加overflow:auto/hidden;  弊端:父元素不能在溢出显示了  4.在父元素的第一个子元素的位置,添加一个空的table  让父元素内容区域上边沿和子元素内容区域上边沿分离 |

### 5.内边距padding

|  |
| --- |
| 改变元素内边距的效果,感觉是改变元素大小  不会影响其他元素,但是会改变本身实际占地大小  边框与内容之间的距离  padding:v1;  padding-top:  padding-right  padding-bottom  padding-left  取值:px为单位的数字  % 父元素宽度的百分比  padding没有auto  简写方式  padding:v1; 4个方向  padding:v1 v2; 上下 左右  padding:v1 v2 v3; 上 左右 下  padding:v1 v2 v3 v4; 上右下左  padding常用于,撑开背景色 |

作业:

1.预习box-sizing属性

2.预习浮动

3.ajax项目,注册完成一遍

4.今天所有单词

### 6.改变框模型的计算方式(多用于,当width取值为%)

|  |
| --- |
| box-sizing:  取值 :默认值 content-box  border-box  border-box框模型计算公式  元素实际占地宽度:width+左右外边距  设置的width包含,左边框+左内+内容+右内+右边框 |

## 六.背景

### 1.背景颜色

|  |
| --- |
| background-color:合法的颜色值 |

### 2.背景图片

|  |
| --- |
| backgrond-image:url(图片资源路径)  使用背景图,可以让子元素堆叠显示在背景图上 |

### 3.背景图片的平铺

|  |
| --- |
| background-repeat:  默认值 repeat  no-repeat 不平铺  repeat-x x轴平铺  repeat-y y轴的平铺 |

### 4.背景图的定位

|  |
| --- |
| background-position:  取值 :1.x y 以px为单位的数字  2.x% y%  3.关键字 x:left/center/right y:top/center/bottom |

### 5.背景图片的尺寸

|  |
| --- |
| background-size:  取值 1.1个值,同时设置x轴和y轴的大小  2.2个值,分别设置x轴和y轴的大小  以px为单位的数字  %  3.cover覆盖,让图片覆盖整个容器,图片显示不全没关系  4.contain包含,让容器包含整个图片.容器有空白没关系 |

### 6.背景图的固定

|  |
| --- |
| background-attachment:  取值 scroll,背景图随着窗口滚动条,滚动  fixed背景图不随着窗口滚动条滚动,  但是只在原始的容器中才显示  当容器定义了background-attachment: fixed  背景图片的固定,相对于body偏移的 |

### 7.背景的简写方式

|  |
| --- |
| background:color image repeat attachment position;  最简方式 background:color/image; |

## 七.渐变gradient

### 1.什么是渐变

|  |
| --- |
| 渐变指多种颜色,平缓变化的一种效果 |

### 2.渐变的主要因素

|  |
| --- |
| 色标:一种颜色,以及出现的位置  一个渐变效果,至少两个色标 |

### 3.渐变的分类

|  |
| --- |
| 线性渐变,以直线的方式来填充渐变色  径向渐变,以圆形的方式填充  重复渐变,将线性和径向重复几次实现 |

### 4.线性渐变

|  |
| --- |
| background-image:linear-gradient(方向,色标1,色标2,......)  方向: 1. 方向的终点 to top/right/bottom/left  方向的角度 0deg to top  90deg to right  180deg to bottom  270deg to left  色标: 1.颜色 0%  2.颜色 px  3.只写颜色, |

### 5.径向渐变

|  |
| --- |
| background-image:radial-gradient(半径 at 圆心,色标1,色标2.)  半径:以px为单位的数字  圆心:1.关键字,x:left/center/right y:top/center/bottom  2.x y 以px为单位的数字  3.x% y%  色标:只有使用%的时候,半径才生效 |

### 6.重复渐变

|  |
| --- |
| background-image:repeating-linear-gradient(to right,#000 0px,#ff0 1px,#000 2px,#0ff 3px,#000 4px);  background-image:repeating-radial-gradient(100px at center center,#000 0px,#ff0 1px,#000 2px,#0ff 3px,#000 4px); |

### 7.浏览器兼容性问题

|  |
| --- |
| background:-moz-linear-gradient(top, #ff0 0%, #0ff 100%)  background:-o-linear-gradient(top, #ff0 0%, #0ff 100%)  background:-webkit-linear-gradient(top, #ff0 0%, #0ff 100%)  background:-ms-linear-gradient(top, #ff0 0%, #0ff 100%)  记忆4个内核  -webkit- chrome/safari  -moz- firefox  -o- opera  -ms- IE  添加了内核,方向不能带to |

## 八.文本格式化(重点\*\*\*\*\*\*\*)

### 1.字体属性

#### ①字号大小

|  |
| --- |
| font-size: px pt rem em |

#### ②字体系列

|  |
| --- |
| font-family:字体系列1,字体系列2,.....  取值,pc中字体库有的字体,如果字体名称带空格,添加"" |

#### ③字体加粗

|  |
| --- |
| font-weight:  取值,关键字 默认normal  bold lighter bolder  无单位,100的整倍数 100~1000 |

#### ④字体样式

|  |
| --- |
| font-style:italic; 斜体  normal 正常 |

#### ⑤小型大写字母

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

#### ⑥字体简写方式

|  |
| --- |
| font:style variant weight size family;  ④ ⑤ ③ ① ②  最简方式:font:size family; |

2.文本格式

样式编写思路

|  |
| --- |
| 1.找要写样式的元素  从外网里,从左往右,从上往下  2.写样式  ①宽高和大体位置  ②边框和背景  ③文字所有  ④微调 |

### 2.文本属性

#### ①字体颜色

|  |
| --- |
| color |

#### ②文本对齐方式

|  |
| --- |
| text-align: left/center/right/justify  注意：1.text-align让内部行内元素和行内块元素,水平对齐  2.margin:auto,让自己水平居中 |

#### ③行高

|  |
| --- |
| 如果设置的行高,大于文本本身字号大小  这行文本会在行高的范围内,垂直居中显示  所以,一般把行高设置成容器高度,让文本在容器垂直居中  line-height:取值 px为单位的数字  无单位的数字 ,行高是字号的倍数  文本有多行,不建议使用行高来设置文本垂直居中 |

#### ④文本的线条修饰

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

#### 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 模糊距离  color 阴影颜色 |

练习

|  |
| --- |
| 定义div,内容假文,有中文  1.设置字体系列 2.文字大小 3.加粗并斜体  4.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 |

## 九.表格

### 1.表格的常用属性

|  |
| --- |
| table的样式,尺寸,背景,边框,文本,内外边距都可以使用  td/th的样式尺寸,背景,边框,文本,内边距都可以使用,外边距无效  设置文本垂直居中 vertical-align:top/middle/bottom |

练习

|  |
| --- |
| 4\*4的表格  宽高400,内容随意  1.设置每个单元格100px\*100px  2.设置每个单元格边框1px solid #000  3.尝试给单元格设置上外边距 |

### 2.表格专有的样式

#### ①边框合并

|  |
| --- |
| border-collapse: separate 分离状态,默认值  collapse 合并状态 |

#### ②边框的边距

|  |
| --- |
| 前提,边框必须是分离状态  border-spacing:  取值 v1:同时设置水平和垂直的距离  v1 v2;分别设置水平和垂直的距离 |

#### ③标题位置

|  |
| --- |
| caption-side:top/bottom; |

#### ④设置表格显示规则

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| table-layout:默认值 auto.自动表格布局  内容大,尺寸小,以内容为准.内容小,尺寸大,以尺寸为准  渲染之前,把表格内容全部读到内存中,一次性渲染  fixed 固定表格布局,以尺寸为准  逐行渲染   |  |  | | --- | --- | | 自动布局 | 固定布局 | | 单元格大小会自动适应内容 | 单元格大小取决设置的尺寸 | | 表格复杂时,加载速度很慢 | 加速加载表格 | | 自动布局比较灵活 | 固定布局不灵活 | | 适用于不确定每列大小的,不复杂的表格 | 适用于确定每列大小的表格  11:01~11:16休息 | |

## 十.定位(重点\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

### 1.什么是定位

|  |
| --- |
| 改变元素在页面中的位置 |

### 2.分类

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

### 3.普通流定位

|  |
| --- |
| 默认文档流定位  1.每个元素在页面上都有自己的空间  2.每个元素都是父元素的左上角开始显示  3.块级元素按照从上往下的方式逐个排列,每个元素独立成行  4.行内,行内块元素是多个元素在一行中显示,从左往右排列,一行放不下,换行 |

### 4.浮动定位(重点\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

|  |
| --- |
| 块级元素横向显示  float:1.left 左浮动  让元素脱离文档流后,在当前行向父元素的左边对齐  2.right右浮动  让元素脱离文档流后,在当前行向父元素的右边对齐  3.none 不浮动,还在文档流里  特点:  1.元素一旦浮动,脱离文档(不占页面空间,后面元素上前补位)  2.浮动元素,最终会停靠在父元素的左边/右边,或者其他已浮动元素元素边缘  3.父元素横向显示不下所有浮动元素时,显示不下的元素,自动换行  4.浮动解决多个块级元素在同一行显示的问题 |

### 5.浮动引发的特殊情况

①浮动元素的占位问题

|  |
| --- |
| 当父元素显示不下所有浮动元素时,最后显示不下会换行.  但是,已浮动元素会根据自己的浮动方向占据位置,导致被挤下去的元素,需要躲开这个占位显示 |

②元素一旦浮动,如果元素未定义宽度,那么元素的宽度靠内容撑开

③元素一旦浮动,变为块级元素(设置宽高有效,上下外边距有效)

④文本,行内元素,行内块,是不会被浮动元素压在下面

而且巧妙的避开浮动元素显示

### 6.清除浮动

|  |
| --- |
| 清除之前浮动元素带来的影响(上前补位)  clear:left; 清除左浮动的影响  right; 清除右浮动的影响  both; 清除左右浮动的影响 |

### 7.高度坍塌

|  |
| --- |
| 父元素不写高度,内部子元素都浮动,父元素就找不到高了  解决方案:  1.父元素写高度,弊端,很多时候,高度确定不了  2.父元素也浮动,弊端,会影响父元素的兄弟元素  3.overflow:auto/hidden  4.在父元素中追加一个空的块级子元素,为其定义clear:both;(空的div放在最后面) |

### A.浮动

|  |
| --- |
| 什么叫 脱离文档流?  1.元素不占据页面空间,后面元素上前补位  2.元素一旦脱离文档流,元素就变为块级元素  3.元素脱离文档流,元素没有设置宽度,元素的宽度靠内容撑开 |

### 8.显示方式(我很重要)

|  |
| --- |
| display:none 隐藏  block 块级  inline 行内  inline-block 行内块  table 表格 |

### 9.显示效果

|  |
| --- |
| visibility: 默认值 visible 可见的  hidden 隐藏的  面试题:visibility:hidden和display:none的区别  display:none脱离文档流  visibility:hidden不脱离文档流 |

### 10.透明度

|  |
| --- |
| opacity:0~1 0 全透明 1 不透明  面试题:opacity和rgba的区别  opacity 元素内部所有颜色都会跟着变透明  rgba 只改变当前的透明度 |

### 11.垂直对齐方式

|  |
| --- |
| vertical-align  ①用在table,控制table中的内容的垂直对齐方式  vertical-align:top/middle/bottom  ②图片,给图片设置垂直对齐,图片本身不动,影响的是图片前后的文本位置.  vertical-align:top/middle/bottom/baseline(默认)  项目中,经常为图片设置vertical-align,让文本不以基线对齐 |

### 12.光标的设置

|  |
| --- |
| cursor:default 箭头  pointer 小手  crosshair +  help 帮助  text 文本  wait 等待 |

### 13.列表的样式

#### ①设置列表标识项

|  |
| --- |
| list-style-type:disc/circle/square/none; |

#### ②标识项设置为图片

|  |
| --- |
| list-style-image:url(true.png); |

#### ③列表项目标记的位置

|  |
| --- |
| list-style-position:inside/outside(默认值) |

#### ④简写方式

|  |
| --- |
| list-style:type image position;  项目中,使用最频繁的方式 list-style:none; |

#### ⑤无序列表的作用

|  |
| --- |
| 无序列表在项目中地位和使用方式  ul多用于布局 |

### 14.定位---相对定位,绝对定位,固定定位

|  |
| --- |
| position: static 静态,默认值,文档流  relative 相对定位  absolute 绝对定位  fixed 固定定位  注意:  1.一个元素被position修饰,并且取值为relative/ absolute/ fixed  此元素被称为已定位元素  2.一个元素变成已定位元素之后,  解锁4个偏移属性  top: 离顶部多远 + ↓ - ↑  right:  bottom:  left: 离左边多远 + → - ← |

#### ①相对定位

|  |
| --- |
| position:relative;  相对定位不脱离文档流,对周围元素的位置没有任何影响  如果相对定位不写偏移属性,效果与没写定位一样  相对自已原来位置做偏移  使用场合  ①类似于margin,做自身位置的微调  ②作为绝对定位的祖先级已定位属性 |

#### ②绝对定位

|  |
| --- |
| position:absolute;  绝对定位脱离文档流  1.没有已定位的祖先级元素,相对body左上角偏移  2.相对于,最近的,已定位的,祖先级元素做偏移  注意:最适合把祖先级变成已定位元素的样式,relative |

#### ③固定定位

|  |
| --- |
| position:fixed;  将元素固定在页面上某个位置,不会随着滚动条滚动,变化位置  一直固定在可视区域 |

#### ④堆叠顺序

|  |
| --- |
| z-index:无单位正整数  注意:  1.定位的堆叠和浮动的脱离文档流,不是一个体系  2.默认,后声明的定位元素,堆叠顺序高,但是不会超过1  3.z-index只对已定位元素生效  4.堆叠顺序对父子级元素无效,儿子永远在父亲的上面 |

# CSS3 CORE

## 一.复杂选择器

### 1.兄弟选择器

|  |
| --- |
| 兄弟元素:具备相同父元素的平级元素,称之为兄弟元素  兄弟选择器,只能通过哥哥找弟弟,不能往前找  ①相邻兄弟选择器  选择器1+选择器2{}获取紧紧挨着某元素后面的兄弟元素  ②通用兄弟选择器  选择器1~选择器2{}获取某元素后面所有符合要求的兄弟元素  使用场景,多用于淘汰第一个元素,其它兄弟都使用的样式 |

### 2.属性选择器

|  |
| --- |
| [attr]{} 匹配定义了attr这个属性的元素  [attr1] [attr2]{} 匹配同时定义了attr1和attr2属性的元素  [attr=value] 匹配定义了attr同时值为value的元素  [attr1=value1] [attr2=value2]{}  elem[attr]{} 匹配定义了attr的elem元素  elem[attr=value]{}  模糊属性值  [attr^=value]{} 匹配attr值的开头是value的元素  [attr$=value]{} 匹配attr值的结尾是value的元素  [attr\*=value]{} 匹配attr值里有value的元素  [attr~=value]{}匹配attr值有value这个单词(前后有空格)的元素 |

### 3.伪类选择器

#### ①目标伪类

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

#### ②结构伪类

|  |
| --- |
| selector:first-child 匹配属于selector父元素的第一个子元素,  并且这个元素还得符合selector  (找到是selector的大哥,这个大哥也得符合selector)  selector:last-child匹配属于selector父元素的最后一个子元素,  并且这个元素还得符合selector  (找到是selector的小弟弟,这个小弟弟也得符合selector)  :nth-child(n) 匹配属于父元素的第n个子元素 n是1开始 |

练习

|  |
| --- |
| 01\_ex.html 创建4\*4的表格 400px\*400px  通过结构选择器,设置以下样式  第一行背景色#0ff  最后一行背景#ff0  第三行的第二列,背景色为pink |

#### ③:empty

|  |
| --- |
| 匹配内部没有任何元素的标签  注意:所谓没有元素,包括不能有空格,回车,标签,文本 |

#### ④:only-child

|  |
| --- |
| 属于其父元素的唯一子元素  可以有其它的文本,空格,回车,没有其它标签就算独生子女 |

⑤否定伪类

|  |
| --- |
| :not(selector){}  符合selector都不要 |

### 4.伪元素选择器

|  |
| --- |
| ::first-letter 或者 :first-letter  匹配元素的首字符  ::first-line 或者 :first-line  匹配元素的首行  ::selection只有双冒号  匹配用户选中的内容  只能设置字体颜色和背景颜色 |

### 5.内容生成

|  |
| --- |
| ::before 或者 :before  在元素内容区域内部的最前端,添加一个假的元素.  这个假元素,有属性content,设置该元素中显示的内容  content中只能放文字和图片url()  在这个假元素中,我们可以使用display设置显示规则.  其它样式也会生效  : :after 或者 :after  在元素内容区域内部的最后端,添加一个假的元素. |
| 内容生成,可以解决的问题  1.解决外边距溢出  父元素:before{  content:"";  display:table;  }  2.解决高度坍塌  #parent:after{  content:"";  display:block;  clear:both;  } |

## 二.弹性布局 重要\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

### 1.什么是弹性布局

|  |
| --- |
| 是一种布局方式  主要解决某个元素中,子元素的布局方式,非常的灵活 |

### 2.弹性布局的相关概念

#### ①容器

|  |
| --- |
| 要发生弹性布局的子元素,他们的父元素称为容器  设置了display:flex的那个元素,就是容器 |

#### ②项目

|  |
| --- |
| 要发生弹性布局的子元素们,叫做项目 |

#### ③主轴

|  |
| --- |
| 项目们的排列方向,就是主轴  项目们横向排列,主轴就是x轴  项目们纵向排列,主轴就是y轴  项目们在主轴上的排列顺序,称为主轴的起点和终点 |

#### ④交叉轴

|  |
| --- |
| 与主轴方向,垂直的方向,称为交叉轴  项目们在交叉轴的对齐方式,称为交叉轴的起点和终点 |

### 3.语法

|  |
| --- |
| 将元素设置成弹性容器后,他的所有子元素将变成弹性项目  都允许按照弹性布局的方式排列  display:1.flex 将块级元素设置为容器  2.inline-flex 将行内元素设置为容器  特点  1.容器的text-align/vertical-align失效  2.项目的float/clear 失效  总结,项目们,水平对齐和垂直对齐,只能够靠主轴和交叉轴控制 |

### 4.容器的属性

#### ①主轴方向

|  |
| --- |
| flex-direction:  取值  row 默认值,主轴是x轴,主轴起点是左侧  row-reverse主轴是x轴,主轴起点是右侧  column 主轴是y轴,主轴起点是上  column-reverse 主轴是y轴,主轴起点是下 |

#### ②设置项目的换行

|  |
| --- |
| 容器空间不足是,项目是否换行  flex-wrap:nowrap 默认值,不换行  wrap 换行  wrap-reverse;换行并反转 |

#### ③flex-flow

|  |
| --- |
| flex-direction和flex-wrap的缩写  flex-flow: direction wrap; |

#### ④项目在主轴上的对齐方式

|  |
| --- |
| justify-content: flex-start 默认值,主轴起点  center; 主轴中心  flex-end 主轴终点  space-around 每个间距大小相同,两端有距离  space-between 两端贴边,每个间距大小相同 |

#### ⑤交叉轴的对齐方式

|  |
| --- |
| align-items:flex-start 默认,交叉轴起点对齐  center 交叉轴中心  flex-end 交叉轴终点  stretch 前提,项目不写高,沾满交叉轴 |

### 5.项目的属性,设置在某一个项目上,不影响其他项目

#### ①项目的排列顺序

|  |
| --- |
| order:无单位的整数  数字越小,项目离主轴起点越近 |

#### ②flex-grow

|  |
| --- |
| 主轴方向,有剩余空间,项目变大  flex-grow:默认值为0,不变大  取值数字越大,变大的越快 |

#### ③flex-shrink

|  |
| --- |
| 主轴的方向,空间不够,项目缩小  默认值是 1  取值数字越大,变小的越快 |

#### ④单个项目交叉轴对齐方式

|  |
| --- |
| align-self:  flex-start 默认,交叉轴起点对齐  center 交叉轴中心  flex-end 交叉轴终点  stretch 前提,项目不写高,沾满交叉轴  auto 继承容器的align-items的值 |

day14

## 三.css hack(过时)

|  |
| --- |
| 让代码能够被ie8以下浏览器正确识别  background-img:-webkit-linear-gradient(top,#aaa,#000);  background-img:-o-linear-gradient(top,#aaa,#000);  background-img:-moz-linear-gradient(top,#aaa,#000);  background-img:-ms-linear-gradient(top,#aaa,#000); |

## 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转换,x轴和y轴的转换  3D转换,添加了z轴 |

### 2.转换属性

|  |
| --- |
| transform:转换函数  转换函数 2d 4个  3d 1个  transform:转换函数1 转换函数2 转换函数3.....; |

### 3.2D转换

#### ①位移

|  |
| --- |
| transform:translate()  取值:translate(v1)和translateX(v1),都是x轴位移  translateY(v1) y轴的位移  translate(v1,v2) v1是x轴,v2是y轴  x:+ →, - ←  y:+ ↓ , - ↑ |
| 如何让一个div在任何屏幕尺寸下,都处于屏幕的正中间  position:absolute;  top:50%;left:50%;  transform:translate(-50%,-50%); |

#### ②缩放

|  |
| --- |
| transform:scale(v1) x轴和y轴同时缩放  scale(v1,v2) x轴和y轴分别设置缩放  scaleX(v1) x轴缩放  scaleY(v1) y轴缩放  v>1 放大  0<v<1 缩小  -1<v<0 反转并缩小  v<-1 反转并放大 |

#### ③旋转

|  |
| --- |
| transform:rotate(ndeg) n:+ 顺时针, - 逆时针  注意:1.转换原点会改变转换效果,默认元素中心  transform-origin:right center;  取值:1.%  2.关键字 x:left/center/right  y:top/center/bottom  3.取值数量:2个值,原点在x轴y轴上  3个值,添加了z轴  2.旋转是连同坐标轴一起旋转的,旋转之后,会影响位移方向 |

#### ④倾斜

|  |
| --- |
| transform:skew()  1.skew(45deg)和skewX(45deg)  y轴向着x轴倾斜45  + 逆时针 - 顺时针    2.skewY(45deg)  x轴向着y轴倾斜45  + 顺时针 - 逆时针    3.skew(45deg,45deg) |

### 4.3D转换

|  |
| --- |
| 3D转换都是模拟的 |

#### ①透视距离

|  |
| --- |
| 模拟人的眼睛到3D转换的元素之间的距离,距离不同,效果不同  perspective:距离 此属性,要定在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,旋转元素角度  烤羊腿,老式爆米花机  2.rotateY(ydeg) 以y轴为中心,旋转元素角度  旋转木马,旋转门,土耳其烤肉  3.rotateZ(zdeg) 以z轴为中心,旋转元素角度  车轮,风车,印度飞饼,摩天轮  4.rotate3D(x,y,z,ndeg)  x,y,x 取值为0,代表不参与旋转  >0 代表参与旋转 |

## 五.过渡

### 1.过渡:

让css的属性值,在一段时间内,平缓的变化(css是两个值)

### 2.语法

|  |
| --- |
| 1.transition-property 指定使用过渡的属性  取值,css的属性名称,多个属性名称之间,用空格分开  all,所有支持过渡的属性,都参与此次过渡  注意,transform必须执行过渡效果  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. transition-duration 过渡效果持续的时长  s/ms |
| 3.过渡时间曲线函数  transition-timing-function:  ease:默认值,慢速开始,中间加速,慢速结束  linear:匀速  ease-in:慢慢开始,一直加速  ease-out:快速开始,一直减速  ease-in-out:慢开,加速到中间,减速,慢关  steps(8) |
| 4.过渡的延迟  transition-delay:1s; |
| 5.过渡代码编写的位置  写在原来的样式中,过渡效果有去有回  写在hover,过渡效果有去无回 |
| 6.过渡的简写方式  transition:property duration timing-fucntion delay;  最简方式 transition:duration; |
| 7.总结:什么时候使用过渡  ①css的属性值,只有两个状态(起始状态,终止状态)  ②css的变化,只能用hover激活(终止状态必须写在hover里)  这个时候使用过渡 |

## 六.动画

### 1.什么是动画

|  |
| --- |
| 多个过渡效果放到一起,就是动画 |

### 2.使用关键帧,来控制动画的每一个状态

|  |
| --- |
| 关键帧 1.动画执行的时间点  2.在这个时间点上的样式 |

### 3.定义动画

|  |
| --- |
| @keyframes 动画名称{  0%{样式}  ...  100%{样式}  } |

### 4.调用动画

|  |
| --- |
| /\*1.使用动画名称\*/  animation-name:change1;  /\*2.动画执行的时间\*/  animation-duration:3s;  /\*3.动画执行的时间曲线函数\*/  animation-timing-function:linear;  /\*4.动画执行的延迟\*/  animation-delay:1s; |

### 5.动画的其它属性

#### ①动画播放次数

|  |
| --- |
| animation-iteration-count:具体数字/infinite无限; |

#### ②动画的播放顺序

|  |
| --- |
| animation-direction:  normal 0%--->100%  reverse 100%--->0%  alternate 播放的奇数次 0%--->100%  偶数次100%--->0% |

#### ③动画的简写方式

|  |
| --- |
| animation:name duration timing-function delay count direction;  最简方式  animation:name duration; |

#### ④动画的填充状态

|  |
| --- |
| animation-fill-mode:  backwards 动画的延迟状态中,显示动画第一帧  forwards 动画结束后,显示动画最后一帧  both 开头结尾都填充  none 不填充 |

#### ⑤动画的播放状态

|  |
| --- |
| animation-play-state:  paused 暂停  running 播放 |

### 6.动画的兼容性

|  |
| --- |
| 定义动画的时候,写css hack  @keyframes 动画名称{}  @-webkit-keyframes 动画名称{}  @-moz-keyframes 动画名称{}  @-o-keyframes 动画名称{}  @-ms-keyframes 动画名称{} |

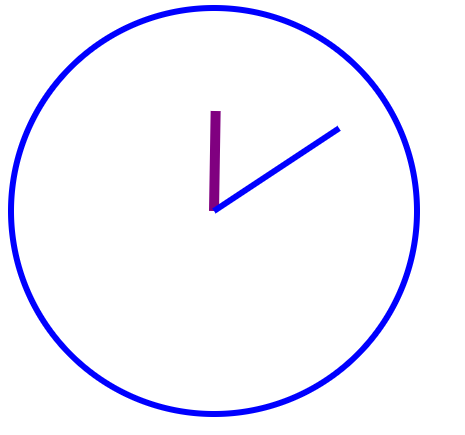
### 7.什么时候用动画,什么时候用过渡

①超过两个css的样式状态,必须用动画

②不使用hover触发,F5刷新,直接出效果,必须用动画

③如果只有两个状态,并且hover触发,动画和过渡都可以使用

推荐使用过渡



### 8.animate.css的使用

|  |
| --- |
| <https://daneden.github.io/animate.css/>  提供了一个动画css库 |

## 七.CSS优化

|  |
| --- |
| css优化的目的  1.减少服务器端的压力  2.提升用户体验 |
| css优化的原则  尽量的减少http的请求个数(缓存) 使用精灵图  页面的顶部,引入css文件  将css和js写在单独的文件中 |
| css代码优化  1.合并样式(能重用,就重用,能群组,就不单独定义,能用简写,就不单写)  2.代码压缩  3.避免出现空的src和href |

## 八.样式重置

|  |
| --- |
| 不同浏览器对元素默认的样式有很大差异 |

### 1.css reset

|  |
| --- |
| 比较霸道,不管你是不是需要html的原始样式,都清掉  我们可以自已编写css reset,也可以使用第三方别人写好的  先把所有样式都清除,让元素基本都没有  但是需要基本的样式的时候,还得手动添加,浪费资源 |

### 2.normalize.css

|  |
| --- |
| 是一种非常流行的css reset的解决方案  重写样式比较柔和  保存了很多默认样式,提升了css的兼容性,修复了很多浏览器的bug |

# BootStrap

简称boot <https://www.bootcss.com/>

## 一.响应式布局(css3 2010提出)

### 1.什么是响应式网页

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

### 2.响应式布局必须保证几件事

|  |
| --- |
| 1.布局,尽量不使用固定值宽度,  必须是流式布局(默认文档流+浮动)+弹性  2.文字和图片大小随着容器大小改变(% rem)  3.使用css3提供的媒体查询技术  注意:代码复杂的页面,不适合使用响应式布局 |

### 3.测试响应式网页的方式

#### ①使用真实设备

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

#### ②使用第三测试软件

|  |
| --- |
| 好处:不需要成本,测试方便  坏处:测试效果有限,需要进一步验证 |

#### ③学习过程,使用chrome自带的模拟软件

|  |
| --- |
| 好处:方便,简单  坏处:测试效果十分有限 |

### 4.编写响应式网页(重点\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

#### ①移动端适配(移动端运行,一定要写)

|  |
| --- |
| <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=0">  width=device-width 视口宽度为设备宽度  initial-scale=1.0, 设置视口初始缩放为1倍,不放大不缩小  maximum-scale=1.0, 允许视口最大放大到1倍  user-scalable=0 是否允许用户缩放视口 0,不允许  简洁的写法  <meta name="viewport" content="width=device-width, initial-scale=1"> |

#### ②所有的内容/文字/图片尽量使用相对尺寸,不使用绝对值

#### ③流式布局+弹性布局+媒体查询,完成响应式布局

#### ④媒体查询 CSS3 Media Query,做响应式必备的技术

|  |
| --- |
| Media 媒体,浏览网页的设备  设备的分类: screen (pc/pad/phone)  TV  print  Media Query:根据当前浏览页面的设备不同(尺寸,硬件,方向,解析度),有选择则性的,执行一部分css,忽略掉其它css样式  语法 @media sreen and 尺寸{ 样式 }  尺寸 w>=1200 xl 超大  992<=w<=1199 lg 大屏  768<=w<=991 md 中屏幕  576<=w<=767 sm 小屏  <=575 xs 超小 |

## 二.BootStrap

|  |
| --- |
| 起步,如何使用boot  全局css样式  组件+js插件  定制 sass  boot项目 |

### 1.如何使用boot

|  |
| --- |
| 按照顺序导入4个boot文件  <link rel="stylesheet" href="css/bootstrap.css">  <script src="js/jquery.min.js"></script>  <script src="js/popper.min.js"></script>  <script src="js/bootstrap.min.js"></script>  body中添加容器div.container  所有boot代码在容器中编写  boot只支持4个屏幕,没有xs  boot中1rem=16px |

### 2. 全局css样式

|  |
| --- |
| .container 定宽容器,每种不同分辨率下,给了写死的max-width  .container-fluid变宽容器,宽度永远是父元素的100% |

#### ①按钮

|  |
| --- |
| btn 基本类.  按钮颜色  btn-danger 红色  btn-success 绿色  btn-warning 黄色  btn-primary 主要的 蓝色  btn-secondary 次要的 灰色  btn-info 信息 藏青色  btn-dark 深色  btn-light 浅色  只有边框的按钮  btn-outline-info/danger/.......  不同按钮大小  btn-lg  btn-sm  btn-block  btn-link |

#### ②图片相关

|  |
| --- |
| rounded 圆角4px  rounded-circle 圆角50%  img-thumbnail 缩略图,有内边距有边框的图片  img-fluid 响应式图片,图片可以缩放,但是最大不能超过原始尺寸 |

#### ③文字相关

|  |
| --- |
| text-danger/info/warning..... 字体颜色  text-muted 灰色字体  text-capitalize 首字母大写  text-uppercase/lowercase 大小写  font-weight-light/normal/bold  字号 h1~h6 类名  文本对齐,封装媒体查询  text-\*-left/right/center \*:xl/lg/md/sm  text-justify没有封装媒体查询  任务,自己手写媒体查询,封装  text-lg-justify text-md-justify text-sm-justify  boot中封装的媒体查询有向上兼容的问题  sm 在sm/md/lg/xl下生效  md 在md/lg/xl下生效  lg 在lg/xl下生效  xl 在xl下生效 |

#### ④列表相关

|  |
| --- |
| list-unstyled 去点,清除左内边距  ul.list-group  >li.list-group-item 创建列表组和列表项  列表项颜色  li.list-group-item-danger/warning......  激活/禁用项  active/disabled |

#### ⑤表格

|  |
| --- |
| table 基本类,对表格和后代进行布局  table-bordered 为表格和后代添加边框  table-info/danger.... 为表格添加颜色  table-striped 隔行变色  table-hover 鼠标悬停变色 |

### 3.辅助类

#### ①边框

|  |
| --- |
| border 基本类,设置4个方向边框  border-top/right/bottom/left 基本类,设置一个方向的边框  border-0 清除边框  border-top/right/bottom/left-0 清除一个方向的边框  border-danger..... 边框颜色 |

#### ②浮动

|  |
| --- |
| float-\*-left/right/none \*:xl/lg/md/sm |

#### ③背景

|  |
| --- |
| bg-danger/info...... |

#### ④显示

|  |
| --- |
| visible 显示  invisible 隐藏 |

#### ⑤内外边距

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

#### ⑥尺寸

|  |
| --- |
| w-25/50/75/100  mw-100 max-width:100%  h-25/50/75/100  mh-100 max-height:100% |

#### ⑦圆角

|  |
| --- |
| rounded/rounded-0 圆角0.25rem/去除圆角  rounded-top/right/bottom/left |

### 3.栅格布局

|  |  |  |
| --- | --- | --- |
| table布局 | div+css布局 | boot中的栅格 |
| 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 | 控制起来比较麻烦  手写媒体查询 | 复杂页面不适用 |

#### ①使用栅格,容易控制

|  |
| --- |
| .row>col-1/2/3/4/5/6/7/8/9/10/11/12 |

#### ②响应式栅格

|  |
| --- |
| .row>col-\*-1/2/3/4/5/6/7/8/9/10/11/12  \*:xl/lg/md/sm |

#### ③使用栅格的注意事项

|  |
| --- |
| row:弹性,主轴横向,可换行 ,左右自带-15px外边距(清0)  col:左右自带15px内边距(清0)  no-gutters 写在row的div,把row的mx-0,把col的px-0 |

#### ④boot的响应式向上兼容的问题

#### ⑤col

|  |
| --- |
| .col 让内部元素平局分配空间,甚至可以大于12个 |

#### ⑥列偏移

|  |
| --- |
| offset-1/2/3/4/5/6/7/8/9/10/11 底层是margin-left |

#### ⑦栅格嵌套

|  |
| --- |
| .row直接子元素只能是col  需要在col中,再写.row>.col |

### 5.弹性布局

|  |
| --- |
| d-\*-flex/none/block/inline/inline-block/table  \*:xl/lg/md/sm  所有弹性,主轴为x轴(换行),都可以替代栅格的.row  1.主轴方向  flex-\*-row/row-reverse/column/column-reverse \*:xl/lg/md/sm  2.主轴对齐方向  justify-content-\*-around/between/start/center/end \*:xl/lg/md/sm |

### 6.表单

#### ①表单元素的排列方向

|  |
| --- |
| form-group 堆叠表单,垂直方向排列  form-inline 内联表单,水平方向排列(弹性布局,主轴x轴) |

#### ②表单元素的样式

|  |
| --- |
| 对文本框的设置  form-control 文本框的基本类 块级 w-100 边框 圆角 过渡 focus  col-form-label/-lg/-sm 设置文本到边框的距离  对checkbox的布局  父级.form-check 相对定位  <input checkbox> .form-check-input 绝对定位  .form-text 块级,上外边距4px |

## 三.组件

|  |
| --- |
| 页面中一些特殊的效果,被boot封装了  让我们更方便,快捷的写出页面效果  boot组件的事件,是通过自定义属性和值来调用的  1.在调用事件的元素上,写自定属性  2.事件发生之后,事件目标 |

### 1.下拉菜单

|  |
| --- |
| 基本结构  div.dropdown 相对定位  >button.dropdown-toggle 画三角  +ul.dropdown-menu 绝对定位,d-none  事件 button添加自定义属性 data-toggle="dropdown"  不需要写事件目标 |

### 2.按钮组

|  |
| --- |
| div.btn-group-vertical 垂直按钮组 .btn-group水平按钮组  >.btn  btn-group-sm/lg 小/大按钮组 |

### 3.信息提示框

|  |
| --- |
| 结构  div.alert alert-danger/warning..... alert-dismissible  >span.close 与父元素alert-dismissible配合,给小叉叉样式  事件  span data-dismiss="alert" |

### 4.导航

#### ①水平导航

|  |
| --- |
| ul.nav 弹性/x轴/wrap 可以设置nav-justified  >li.nav-item 与父元素nav-justified配合,让每一项等宽显示  >a.nav-link 块级,hover去下划线 |

#### ②选项卡导航

|  |
| --- |
| 结构  上部分,选项卡部分  ul.nav .nav-tabs 设置了边框  >li.nav-item  >a.nav-link 设置了hover的边框 .active 选中状态  下部分,内容  div.tab-content  >div.tab-pane .tab-content .tab-pane 隐藏  .active 显示  事件  a data-toggle="tab"  a href="#目标id" |

#### ③胶囊导航

|  |
| --- |
| ul.nav.nav-pills  a data-toggle="pill" |

### 5.响应式导航栏

|  |
| --- |
| div.navbar .navbar-expand-xl/lg/md/sm  >ul.navbar-nav 弹性,默认y轴  >li.nav-item  >a.nav-link  注意 div.navbar-expand-\* 与子元素 ul.navbar-nav 配合组成一个选择器.当父元素类名成立,子元素的主轴为row  当父元素类名不成立,子元素的主轴为column  形成了响应式的导航栏 |

### 6.折叠效果

|  |
| --- |
| 结构  button  +div.collapse 隐藏  事件  button data-toggle="collapse" data-target="#id" |

### 7.卡片

|  |
| --- |
| div.card  >div.card-header  +div.card-body  +div.card-footer |

### 8.手风琴效果

|  |
| --- |
| <**h3**>手风琴</**h3**> <**div id="parent"**>  <**div class="card"**>  <**div class="card-header"**>  <**a data-toggle="collapse" class="card-link" href="#c1"**>卡片1</**a**>  </**div**>  <**div data-parent="#parent" id="c1" class="collapse"**>  <**div class="card-body "**>  卡片1 lorem20  </**div**>  </**div**>  </**div**>  <**div class="card"**>  <**div class="card-header"**>  <**a data-toggle="collapse" class="card-link" href="#c2"**>卡片2</**a**>  </**div**>  <**div data-parent="#parent" id="c2" class="collapse"**>  <**div class="card-body "**>  卡片2 lorem20  </**div**>  </**div**>  </**div**>  <**div class="card"**>  <**div class="card-header"**>  <**a data-toggle="collapse" class="card-link" href="#c3"**>卡片3</**a**>  </**div**>  <**div data-parent="#parent" id="c3" class="collapse"**>  <**div class="card-body "**>  卡片3 lorem20  </**div**>  </**div**>  </**div**> </**div**>  注意事项  1. .card-body和.collapse 不能作用在同一个div上,不然会卡顿  解决方案 div.collapse>div.card-body  2.几个被折叠的部分,可以同时打开,而不是只能打开一个  解决方案:在所有card外面,添加父元素div#parent  在所有的div.collapse中添加自定义属性  data-parent="#parent" |

### 9.折叠导航栏

|  |
| --- |
| 结构  div.navbar 导航栏的标准类  .navbar-dark 告诉孩子们,导航栏背景是深色,你们用浅色的文字  控制了button的颜色,ul的文本颜色,a标签颜色  .bg-dark 背景为深色  .navbar-expand-\* 让按钮在\*以上的屏幕,隐藏  让导航在\*以上横向显示  >a.navbar-brand 不折叠的菜单项  +button.navbar-toggler 改变按钮背景色  >span.navbar-toggler-icon 按钮中间3条线  +div.collapse 隐藏  .navbar-collapse 调整隐藏菜单的显示位置  >ul.navbar-nav>li.nav-item>a.nav-link |

### 10.媒体对象

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

### 11.焦点轮播图

|  |
| --- |
| 1.轮播图片  div.carousel 相对定位  >div.carousel-inner w100 溢出隐藏  >div.carousel-item.active display:none  >img.w-100  事件  在最外层div中 data-ride="carousel" |
| 2.左右箭头  a.carousel-control-prev/next  >span.carousel-control-prev/next-icon  我们自己重写样式  .**carousel-control-prev**, .**carousel-control-next**{  **width**:4%;**height**:20%;  **background**:**#aaa**;  **top**:40%;  **border-radius**: 0.25**rem**; }  事件  a data-slide="prev/next" href="#最外层div 的id" |
| 3.轮播指示器  ul.carousel-indicators 弹性,x轴  >li 重写li的样式 .active  .**carousel-indicators li**{  **width**:0.8**rem**;**height**:0.8**rem**;  **background-color**:**#fff**;  **border-radius**: 50%;  **margin-left**: 0.4**rem**;  **margin-right**: 0.4**rem**; } .**carousel-indicators** .**active**{  **background-color**:**#0aa1ed**; }  事件  li data-target="#demo" data-slide-to="0"图片的下标,从0开始 |

### 12.模态框

|  |
| --- |
| <div class="modal">  <div class="modal-dialog">  <div class="modal-content">  <div class="modal-header"></div>  <div class="modal-body"></div>  <div class="modal-footer"></div>  </div>  </div>  </div> |

## 四.其它组件

### 1.徽章----小按钮

|  |
| --- |
| badge 基本类  badge-danger/warning.....  badge-pill 胶囊徽章 |

### 2.巨幕

|  |
| --- |
| jumbotron 巨大的内边距和背景色 |

### 3.分页条

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

### 4.面包屑导航

|  |
| --- |
| ul.breadcrumb  >li.breadcrumb-item  >a  重写样式,改变连接符号  .breadcrumb-item + .breadcrumb-item::before{  content:">";  } |

### 5.进度条

|  |
| --- |
| div. progress 进度条的槽  >div. progress-bar 进度条  使用bg-danger/... 控制颜色  使用w-75 控制进度  .progress-bar-animated 带动画的进度条  .progress-bar-striped 带斜线的进度条 |

## 五.boot定制

boot最重要的, 媒体查询+栅格布局+scss

### 1.css的缺点

|  |
| --- |
| 语法不够强大,没有变量和合理的样式复用机制,导致难以维护  使用动态样式语言,赋予CSS新的特性,提高样式的可重用性和可维护性  常用的动态样式语言  1.scss/sass(scss兼容sass,scss更贴近css的语法)  2.less  3.stylus |

### 2.scss是什么

|  |
| --- |
| scss是一款强大的css辅助工具  它和css的语法十分相似,在css的基础上  添加了变量,嵌套,混合,导入,函数等高级功能  这些拓展命令让css更加的强大,与优雅 |

### 3.scss的使用

|  |
| --- |
| 运行在服务器  把.scss文件转换成.css文件  传递给前端,让浏览器运行  浏览器本身不具备运行.scss的能力  需求,node 版本8.11(10.01)  1.使用无网络的安装方式  2.在线安装  cmd中 npm install -g node-sass  node-sass -v验证 |

### 4.scss文件转换成css文件

|  |
| --- |
| 1.单文件转换  node-sass scss文件路径 css文件路径  ex:node-sass scss/01.scss css/01.css  2.批量转换,一次转一个文件夹  node-sass scss文件夹 -o css文件夹  node-sass scss -o css  3.单文件监听  node-sass -w 要监听的scss文件 要转换成的css文件  node-sass -w scss/01.scss css/01.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 |

## 六.scss基础语法

### 1.变量

|  |
| --- |
| $jd\_red:#f10215;  使用$声明变量,变量名可以包含- \_,命名规则基本上与选择器规则相同  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.变量存在作用域,在{}外不能用  3.重复的变量声明,后声明会覆盖之前的声明的变量  !default规则,如果一个变量声明了!default.  这变量之前被声明过,就使用之前值  这个变量没有被声明过,就使用现在的值 |

### 2.嵌套

|  |
| --- |
| 结构的嵌套  伪类嵌套,需要占位符&. 不然会有空格  a{  color:#fff;  &:hover{color:#f00;}  }  群组的嵌套  属性嵌套(违反css编码原则)  border:{style:solid;width:10px;color:#ff0;}; |

练习

|  |
| --- |
| 1.学子商城的按钮  $myblue:#0aa1ed;  $font-color:#fff;  $my-width:130pxl;  $my-height:39px;  使用a标签,写出按钮 |
| 2.一个200\*200的div,红色背景  鼠标悬停时过渡到黄色,并且变为圆形  使用scss编写,伪类嵌套,过渡用属性嵌套 |

### 3.导入

|  |
| --- |
| 在scss语法中,如果一个scss文件以\_开头,那么这个文件就是一个局部scss文件.局部scss文件是不会转换成css文件的  需要在全局scss文件中导入局部文件,在把全局文件转换成css文件  \_abc.scss  导入语法 @import "abc";  局部文件被导入后,局部文件中的代码会在全局文件中生成  局部文件的变量可以在全局文件中使用 |

### 4.混合器

|  |
| --- |
| 把一段经常被使用的样式封装成混合器  哪里需要使用,就调用混合器,实现代码的重用  定义混合器  @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(实参1,实参2); |

### 5.继承

|  |
| --- |
| 一个选择器可以继承另外一个选择器所有的样式  @extend 选择器名称;  继承效果,两个选择器,形成群组选择器 |

练习

|  |
| --- |
| 定义一个带参数的混合器,编写线性渐变,  混合器参数,方向,颜色1,颜色2  .my\_bg1{调用混合器,参数随意}  .my\_bg2{继承my\_bg1}并且hover下改变背景色为绿色 (嵌套) |

### 6.运算 + - /

相对单位不能转换,绝对单位可以自动转换

#### ①加法

|  |
| --- |
| 字符串拼接的时候  如果使用带双引号的字符串+不带双引号的字符串,结果带双引号  如果使用不带双引号的字符串+带双引号的字符串,结果不带双引号 |

#### ②减法

|  |
| --- |
| 由于变量声明的时候,可以使用-  系统分不清楚,- 是变量名称还是减法  我们写的时候在-前后添加空格.告诉系统,我是减号  width:$my-width - $my-width1; |

#### ③除法

|  |
| --- |
| scss中 /,除了解析成除法外,还被看成分割符  视为除法的几种情况  1.如果运算式两边的数字,是变量或者方法的返回值  2.运算式被()包裹  3.除法运算式,是其它运算式的一部分 |

#### ④scss字符串中的插值操作

|  |
| --- |
| content:"liangCheng ate #{50+100} baozis"; |

#### ⑤颜色的计算

|  |
| --- |
| 分段计算 r+r g+g b+b 如果超过255,或者ff.就取ff或者255  rgb(11,22,33)+#162c42  rgba的计算,需要a的值,相同,才可以计算 |

### 7.函数

#### ①scss中定义了很多函数,有些函数直接在css中使用

|  |
| --- |
| rgba  hsl(hue,saturation,lightness)  hue:色调 0~360  saturation:饱和度 0.0%~100%  lightness:亮度 0.0%~100%  background:hsl(120,100%,50%); |

#### ②数学函数

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

#### ③字符串函数

|  |
| --- |
| unquote($str) $str去掉引号  quote($str) $str加上引号  to\_upper\_case($str) $str转成大写  to\_lower\_case($str) $str转成小写 |

#### ④自定义函数

|  |
| --- |
| @function 方法名称($v1,$v2....){  逻辑  @return 返回值;  } |

### 8.指令

|  |
| --- |
| @if(){  }@else if(){  }@else{} |