目录

[0.1mysql 21](#_Toc54875277)

[1.项目中的数据 21](#_Toc54875278)

[2.数据库 21](#_Toc54875279)

[(1)数据库发展历史 21](#_Toc54875280)

[(2)关系型数据库逻辑结构 21](#_Toc54875281)

[3.MySQL数据库 21](#_Toc54875282)

[4.mysql常用管理命令 22](#_Toc54875283)

[5.SQL命令 22](#_Toc54875284)

[6.常用的SQL命令 23](#_Toc54875285)

[7.关系型数据库逻辑结构 24](#_Toc54875286)

[8. 常用管理命令 24](#_Toc54875287)

[标准SQL命令分类 24](#_Toc54875288)

[1.计算机存储字符 24](#_Toc54875289)

[2.mysql中的列类型 25](#_Toc54875290)

[(1)数值类型——引号可加可不加 25](#_Toc54875291)

[(2)日期时间类型——必须加引号 25](#_Toc54875292)

[(3)字符串类型——必须加引号 26](#_Toc54875293)

[3.列约束 26](#_Toc54875294)

[(1)主键约束——PRIMARY KEY 26](#_Toc54875295)

[1.列约束 27](#_Toc54875296)

[(1)主键约束——PRIMARY KEY 27](#_Toc54875297)

[(2)非空约束——NOT NULL 27](#_Toc54875298)

[(3)唯一约束——UNIQUE 27](#_Toc54875299)

[(4)检查约束——CHECK 28](#_Toc54875300)

[(5)默认值约束——DEFAULT 28](#_Toc54875301)

[(6)外键约束——FOREIGN KEY 28](#_Toc54875302)

[2.mysql中的自增列 28](#_Toc54875303)

[3.简单查询 28](#_Toc54875304)

[(1)查询特定的列 28](#_Toc54875305)

[(2)查询所有的列 29](#_Toc54875306)

[(3)给列起别名 29](#_Toc54875307)

[(4)显示不同的记录/合并相同的记录 29](#_Toc54875308)

[(5)查询时执行计算 29](#_Toc54875309)

[(6)查询结果集排序 29](#_Toc54875310)

[(7)条件查询 30](#_Toc54875311)

[(8)分页查询 30](#_Toc54875312)

[02.js 31](#_Toc54875313)

[1.简单查询——模糊条件查询 32](#_Toc54875314)

[2.复杂查询 32](#_Toc54875315)

[(1)聚合查询/分组查询 32](#_Toc54875316)

[(2)子查询 33](#_Toc54875317)

[(3)多表查询 33](#_Toc54875318)

[1.JS概述 34](#_Toc54875319)

[(1)历史 34](#_Toc54875320)

[(2)现状 34](#_Toc54875321)

[(3)特点 34](#_Toc54875322)

[(4)应用场景 34](#_Toc54875323)

[2.JS的执行环境 35](#_Toc54875324)

[3.JS语法规范 35](#_Toc54875325)

[4.变量 35](#_Toc54875326)

[(1)声明变量 35](#_Toc54875327)

[(2)变量的命名规则 35](#_Toc54875328)

[(3)声明变量未赋值 36](#_Toc54875329)

[(4)一次性声明多个变量 36](#_Toc54875330)

[5.常量 36](#_Toc54875331)

[1.数据类型 36](#_Toc54875332)

[(1)数值型 36](#_Toc54875333)

[(2)字符串型 37](#_Toc54875334)

[(3)布尔型 37](#_Toc54875335)

[2.数据类型转换 37](#_Toc54875336)

[(1)隐式转换 37](#_Toc54875337)

[(2)强制转换 38](#_Toc54875338)

[3.运算符 38](#_Toc54875339)

[(1)算术运算符 38](#_Toc54875340)

[(2)比较运算符 38](#_Toc54875341)

[(3)逻辑运算符 39](#_Toc54875342)

[(4)位运算符(了解) 39](#_Toc54875343)

[1.赋值运算符 40](#_Toc54875344)

[2.三目运算符 40](#_Toc54875345)

[3.浏览器端函数 40](#_Toc54875346)

[4.流程控制 40](#_Toc54875347)

[(1)if语句 40](#_Toc54875348)

[(2)if-else语句 41](#_Toc54875349)

[(3)if-else嵌套 41](#_Toc54875350)

[(4)switch-case语句 41](#_Toc54875351)

[1.循环 42](#_Toc54875352)

[(1)while循环 42](#_Toc54875353)

[(2)do-while循环 42](#_Toc54875354)

[(3)for循环 43](#_Toc54875355)

[2.break和continue 43](#_Toc54875356)

[3.循环嵌套 43](#_Toc54875357)

[1.函数 44](#_Toc54875358)

[(1)创建普通的函数 44](#_Toc54875359)

[(2)创建带有参数的函数 44](#_Toc54875360)

[(3)带有返回值的函数 44](#_Toc54875361)

[2.变量作用域 45](#_Toc54875362)

[(1)作用域 45](#_Toc54875363)

[(2)变量声明提升 45](#_Toc54875364)

[3.函数的作用域 45](#_Toc54875365)

[4.递归调用 45](#_Toc54875366)

[1.匿名函数 45](#_Toc54875367)

[(1)创建函数 46](#_Toc54875368)

[(2)匿名函数自调用 46](#_Toc54875369)

[(3)回调函数 46](#_Toc54875370)

[2.全局函数 46](#_Toc54875371)

[3.对象 47](#_Toc54875372)

[(1)JS中的对象 47](#_Toc54875373)

[(2)自定义对象 47](#_Toc54875374)

[4.使用对象字面量创建对象 47](#_Toc54875375)

[5.访问对象的属性 47](#_Toc54875376)

[6.使用内置构造函数创建对象 47](#_Toc54875377)

[7.遍历对象的属性 48](#_Toc54875378)

[8.对象中的方法 48](#_Toc54875379)

[1.检测对象中是否含有某个属性 49](#_Toc54875380)

[2.创建数组 49](#_Toc54875381)

[(1)数组字面量 49](#_Toc54875382)

[(2)访问数组中的元素 49](#_Toc54875383)

[(3)内置构造函数 49](#_Toc54875384)

[(4)数组的长度 49](#_Toc54875385)

[2.数组的分类 49](#_Toc54875386)

[3.遍历数组 50](#_Toc54875387)

[(1)for-in 50](#_Toc54875388)

[(2)循环 50](#_Toc54875389)

[4.数组API(方法) 50](#_Toc54875390)

[1.数组API 51](#_Toc54875391)

[2.二维数组 51](#_Toc54875392)

[3.字符串 51](#_Toc54875393)

[(1)转义字符 —— \ 51](#_Toc54875394)

[(2)字符串API 52](#_Toc54875395)

[4.匹配模式(掌握) 52](#_Toc54875396)

[5.Math对象 53](#_Toc54875397)

[1.Date对象 53](#_Toc54875398)

[(1)创建Date对象 53](#_Toc54875399)

[(2)获取Date对象中的日期时间 53](#_Toc54875400)

[(3)转为本地字符串 54](#_Toc54875401)

[(4)修改Date对象中的日期时间 54](#_Toc54875402)

[2.Number对象 54](#_Toc54875403)

[3.Boolean对象 54](#_Toc54875404)

[4.错误处理 54](#_Toc54875405)

[5.ES6 55](#_Toc54875406)

[(1)块级作用域 55](#_Toc54875407)

[(2)箭头函数 55](#_Toc54875408)

[03.总监咨询日.李文华 55](#_Toc54875409)

[03.node.js 56](#_Toc54875410)

[3.全局对象 57](#_Toc54875411)

[(1)console对象——控制台 57](#_Toc54875412)

[(2)process对象——进程 57](#_Toc54875413)

[(3)Buffer对象——缓冲区 57](#_Toc54875414)

[4.模块 58](#_Toc54875415)

[(1)自定义模块 58](#_Toc54875416)

[day02 58](#_Toc54875417)

[1.模块分类 59](#_Toc54875418)

[2.npm和包 59](#_Toc54875419)

[(1)切换目录 59](#_Toc54875420)

[(2)使用npm下载安装第三方包 59](#_Toc54875421)

[(3)生成package.json 59](#_Toc54875422)

[3.全局函数 59](#_Toc54875423)

[(1)一次性定时器 60](#_Toc54875424)

[(2)周期性定时器 60](#_Toc54875425)

[(3)立即执行定时器 60](#_Toc54875426)

[4.querystring模块——查询字符串 60](#_Toc54875427)

[5.url模块 60](#_Toc54875428)

[6.fs模块——文件系统模块 61](#_Toc54875429)

[(1)查看文件的状态 61](#_Toc54875430)

[全局函数 61](#_Toc54875431)

[1.同步和异步 62](#_Toc54875432)

[2.fs模块——文件系统 62](#_Toc54875433)

[2.http协议 63](#_Toc54875434)

[(1)通用头信息(General) 63](#_Toc54875435)

[(2)响应头信息(ResponseHeaders) 63](#_Toc54875436)

[(3)请求头信息(RequestHeaders) 63](#_Toc54875437)

[3.http模块 63](#_Toc54875438)

[(1)模拟浏览器 63](#_Toc54875439)

[(2)创建web服务器 64](#_Toc54875440)

[1.express框架 65](#_Toc54875441)

[(1)路由 65](#_Toc54875442)

[2.post和get传递数据 66](#_Toc54875443)

[3.使用路由传递数据——路由传参 66](#_Toc54875444)

[4.路由器 67](#_Toc54875445)

[1.中间件 67](#_Toc54875446)

[(1)应用级(自定义)中间件 68](#_Toc54875447)

[(2)路由级中间件 68](#_Toc54875448)

[(3)内置中间件 68](#_Toc54875449)

[(4)第三方中间件 68](#_Toc54875450)

[2.mysql 68](#_Toc54875451)

[(1)连接mysql 68](#_Toc54875452)

[(2)连接池 69](#_Toc54875453)

[04.html 69](#_Toc54875454)

[一.课程安排 70](#_Toc54875455)

[二.web基础知识 70](#_Toc54875456)

[3.web运行原理 71](#_Toc54875457)

[1.什么是html 71](#_Toc54875458)

[2.html的特点 72](#_Toc54875459)

[3.html语法 72](#_Toc54875460)

[四.HTML的文档结构 73](#_Toc54875461)

[1.html文档的构成 73](#_Toc54875462)

[五.文本标记(重点\*\*\*) 74](#_Toc54875463)

[六.图形和链接 75](#_Toc54875464)

[1.图形图像 75](#_Toc54875465)

[六.图像和链接 77](#_Toc54875466)

[1.图像的使用 77](#_Toc54875467)

[2.URL 77](#_Toc54875468)

[3.属性 78](#_Toc54875469)

[4.链接 78](#_Toc54875470)

[七.表格 79](#_Toc54875471)

[1.语法 79](#_Toc54875472)

[2. 不规则的表格 80](#_Toc54875473)

[八.列表（重点\*\*\*\*\*\*\*） 81](#_Toc54875474)

[3.列表的嵌套 83](#_Toc54875475)

[4.定义列表 83](#_Toc54875476)

[九.结构标签 83](#_Toc54875477)

[十.表单，\*\*\*\*重点\*\*\*\*\* 83](#_Toc54875478)

[4.表单控件 84](#_Toc54875479)

[1.表单控件的分类 84](#_Toc54875480)

[2.input元素 84](#_Toc54875481)

[3.input详解9种 85](#_Toc54875482)

[4 textarea多行文本域 86](#_Toc54875483)

[5.下拉选 86](#_Toc54875484)

[6.其它元素 86](#_Toc54875485)

[7.为控件分组 86](#_Toc54875486)

[十一.浮动框架 87](#_Toc54875487)

[3.列表的嵌套 88](#_Toc54875488)

[九.结构标签 88](#_Toc54875489)

[十.表单，\*\*\*\*重点\*\*\*\*\* 88](#_Toc54875490)

[十一.浮动框架 91](#_Toc54875491)

[1.h5新表单元素 93](#_Toc54875492)

[ajax 93](#_Toc54875493)

[二.HTTP协议 93](#_Toc54875494)

[1.URL 93](#_Toc54875495)

[2.HTTP协议 94](#_Toc54875496)

[3.web请求原理详解 95](#_Toc54875497)

[4.消息/报文 Message 95](#_Toc54875498)

[5.Request Message 95](#_Toc54875499)

[6. Response Message 96](#_Toc54875500)

[7.缓存 97](#_Toc54875501)

[8.HTTP性能优化 98](#_Toc54875502)

[ajax常见错误 98](#_Toc54875503)

[一.Dom操作(简单dom操作) 99](#_Toc54875504)

[1.完整的javascript的组成 99](#_Toc54875505)

[2.使用js的Dom获取页面某个元素 99](#_Toc54875506)

[二.Ajax 101](#_Toc54875507)

[1.同步Synchronous 101](#_Toc54875508)

[2.异步Asynchronous 101](#_Toc54875509)

[3.什么是ajax 101](#_Toc54875510)

[4.使用ajax 101](#_Toc54875511)

[一.ajax的post请求 104](#_Toc54875512)

[1.任务，使用get请求再完成一遍登录模块 10:45我写 104](#_Toc54875513)

[2.使用post方法发送请求 104](#_Toc54875514)

[二.Json的解析 104](#_Toc54875515)

[1.js对象的数据格式 104](#_Toc54875516)

[2.JSON数据格式 104](#_Toc54875517)

[3.json字符串的格式 105](#_Toc54875518)

[4.json解析，把json字符串转换成js对象数组 105](#_Toc54875519)

[三.XML的解析 105](#_Toc54875520)

[1.什么是XML 105](#_Toc54875521)

[2.xml解析 105](#_Toc54875522)

[常见错误 106](#_Toc54875523)

[ajax项目 106](#_Toc54875524)

[一.搭建项目结构 106](#_Toc54875525)

[一.登录模块 107](#_Toc54875526)

[三.删除 109](#_Toc54875527)

[四.修改模块 112](#_Toc54875528)

[1.根据uid检索用户 112](#_Toc54875529)

[2.修改模块 112](#_Toc54875530)

[1.注册 115](#_Toc54875531)

[css 115](#_Toc54875532)

[一.CSS概述 115](#_Toc54875533)

[4.CSS与HTML的属性，使用原则 116](#_Toc54875534)

[二.css的语法规范 116](#_Toc54875535)

[1.使用css样式的方式 116](#_Toc54875536)

[2.CSS的特性 117](#_Toc54875537)

[三.基础选择器(重点\*\*\*\*\*\*\*\*\*) 118](#_Toc54875538)

[1.选择器的作用 118](#_Toc54875539)

[2.选择器详解 118](#_Toc54875540)

[四.尺寸和边框 121](#_Toc54875541)

[1.尺寸属性 121](#_Toc54875542)

[2.页面中允许设置尺寸的元素 121](#_Toc54875543)

[3.溢出处理 122](#_Toc54875544)

[4.边框 123](#_Toc54875545)

[5.边框的倒角(圆角) 124](#_Toc54875546)

[6.边框阴影 124](#_Toc54875547)

[7.轮廓 125](#_Toc54875548)

[五.框模型---盒子模型 125](#_Toc54875549)

[1.外边距margin 125](#_Toc54875550)

[2.外边距的特殊效果 126](#_Toc54875551)

[五.框模型 127](#_Toc54875552)

[2.外边距的特殊效果 127](#_Toc54875553)

[3.内边距 129](#_Toc54875554)

[4.box-sizing属性(设置元素实际占地尺寸的公式) 129](#_Toc54875555)

[六.背景 129](#_Toc54875556)

[1.背景颜色 129](#_Toc54875557)

[2.背景图片 129](#_Toc54875558)

[3.背景图片的平铺 129](#_Toc54875559)

[4.背景图片的定位 130](#_Toc54875560)

[5.背景图片的尺寸 130](#_Toc54875561)

[6.背景图片的固定 130](#_Toc54875562)

[7.背景的简写方式 130](#_Toc54875563)

[七.渐变 gradient 130](#_Toc54875564)

[3.渐变的分类 130](#_Toc54875565)

[4.线性渐变 130](#_Toc54875566)

[5.径向渐变 131](#_Toc54875567)

[6.重复渐变 131](#_Toc54875568)

[7.浏览器兼容问题 131](#_Toc54875569)

[八.文本格式化(重点\*\*\*\*\*\*\*\*) 132](#_Toc54875570)

[1.字体属性 132](#_Toc54875571)

[三.文本格式化(重点\*\*\*\*\*\*\*\*) 132](#_Toc54875572)

[1.字体属性 132](#_Toc54875573)

[2.文本属性 133](#_Toc54875574)

[四.表格的样式 134](#_Toc54875575)

[1.表格的常用样式属性table>tr>td 134](#_Toc54875576)

[2.表格特有的样式属性 135](#_Toc54875577)

[五.定位(重点\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*) 136](#_Toc54875578)

[2.定位的分类 136](#_Toc54875579)

[一.元素的显示方式 138](#_Toc54875580)

[二.显示效果 138](#_Toc54875581)

[三.透明度 138](#_Toc54875582)

[四.垂直对齐方式 139](#_Toc54875583)

[五.光标的设置 139](#_Toc54875584)

[六.列表的样式 139](#_Toc54875585)

[七.定位---相对，绝对，固定定位（\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*） 140](#_Toc54875586)

[CSS3 CORE 141](#_Toc54875587)

[一.复杂选择器 141](#_Toc54875588)

[1.兄弟选择器 141](#_Toc54875589)

[2.属性选择器 142](#_Toc54875590)

[3.伪类选择器 142](#_Toc54875591)

[一.复杂选择器 143](#_Toc54875592)

[3.伪类选择器 143](#_Toc54875593)

[4.伪元素选择器 144](#_Toc54875594)

[二.弹性布局（重点\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*） 144](#_Toc54875595)

[1.什么是弹性布局 144](#_Toc54875596)

[2.弹性布局相关的概念和名词解释 145](#_Toc54875597)

[3.语法 145](#_Toc54875598)

[4.容器的属性 145](#_Toc54875599)

[5.项目的属性 146](#_Toc54875600)

[三.CSS hack 147](#_Toc54875601)

[四.转换（重点\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*） 147](#_Toc54875602)

[四.转换 148](#_Toc54875603)

[4.2D转换 148](#_Toc54875604)

[5.3D转换---3d都是模拟 150](#_Toc54875605)

[五.过渡 150](#_Toc54875606)

[六.动画 151](#_Toc54875607)

[3.使用动画 152](#_Toc54875608)

[5.动画的兼容性 153](#_Toc54875609)

[七.CSS优化 153](#_Toc54875610)

[BootStrap 154](#_Toc54875611)

[一.响应式布局(css3) 154](#_Toc54875612)

[4.编写响应式布局 155](#_Toc54875613)

[三.如何使用bootstrap 156](#_Toc54875614)

[四.全局样式 157](#_Toc54875615)

[1.按钮相关的class 157](#_Toc54875616)

[2.图片相关 157](#_Toc54875617)

[3.文本相关的class 157](#_Toc54875618)

[4.列表相关class 158](#_Toc54875619)

[5.表格相关的样式 158](#_Toc54875620)

[6.辅助类 158](#_Toc54875621)

[①边框 158](#_Toc54875622)

[②浮动 158](#_Toc54875623)

[③显示 158](#_Toc54875624)

[⑤圆角 158](#_Toc54875625)

[⑥内外边距 159](#_Toc54875626)

[7.栅格布局 159](#_Toc54875627)

[五.弹性布局 160](#_Toc54875628)

[1.主轴方向 160](#_Toc54875629)

[2.主轴对齐方式 160](#_Toc54875630)

[六.表单相关的class 160](#_Toc54875631)

[1.表单元素的排列放向 160](#_Toc54875632)

[2.表单控件元素样式 161](#_Toc54875633)

[七.组件 161](#_Toc54875634)

[1.下拉菜单 161](#_Toc54875635)

[2.按钮组 162](#_Toc54875636)

[3.信息提示框 162](#_Toc54875637)

[4.导航 162](#_Toc54875638)

[②选项卡导航 163](#_Toc54875639)

[6.折叠效果 164](#_Toc54875640)

[7.卡片 164](#_Toc54875641)

[8.卡片+折叠=手风琴 164](#_Toc54875642)

[9.折叠导航栏 165](#_Toc54875643)

[10.媒体对象 165](#_Toc54875644)

[11.焦点轮播图 165](#_Toc54875645)

[12.模态框 166](#_Toc54875646)

[八.其它组件 167](#_Toc54875647)

[1.徽章 167](#_Toc54875648)

[2.巨幕 167](#_Toc54875649)

[3.分页条 167](#_Toc54875650)

[八.其他组件 167](#_Toc54875651)

[4.面包屑导航 167](#_Toc54875652)

[BOOT定制:scss 168](#_Toc54875653)

[1.CSS有很多缺点 168](#_Toc54875654)

[十.SCSS详解 168](#_Toc54875655)

[十一.SCSS基本语法 169](#_Toc54875656)

[十二.函数 172](#_Toc54875657)

[一.配色网站 174](#_Toc54875658)

[二.好的软件 176](#_Toc54875659)

# 0.1mysql

## 1.项目中的数据

内存/特定的文件(exel、word)/第三方服务器/数据库服务器

## 2.数据库

数据库是按照一定的形式来组织、存储数据，目的是为了对数据操作——增删改查

### (1)数据库发展历史

网状数据库->层次型数据库->关系型数据库->非关系型数据库(NoSql)

### (2)关系型数据库逻辑结构

|  |
| --- |
| Server->Database->Table->Row->Column |

(3)常见的关系型数据库

SQLite 微型数据库

SQL Server 适用于Windows系统，收费，中型数据库

Oracle 大型数据库，收费

MySQL 中小型数据库，免费的，适用于各种操作系统

## 3.MySQL数据库

MySQL AB -> SUN -> Oracle

oracle分支：mysql

martin分支：MariaDB

XAMPP软件

|  |
| --- |
| 服务器套装，包含多款服务器软件mysql、Apache...  <https://www.apachefriends.org/download.html> |

(1)mysql部署结构

服务器端：服务存储/维护数据 ——银行服务器机房

c:/xampp/mysql/bin/mysqld.exe 启动服务

确保3306端口不被占用

客户端：负责向服务器端发起增删改查 ——ATM机

c:/xampp/mysql/bin/mysql.exe 客户端工具

(2)使用客户端连接服务器端

mysql.exe -h127.0.0.1 -P3306 -uroot -p

-h host 主机(IP地址/域名)

-P port 端口

-u user 用户

-p password 密码

mysql -uroot 简写形式

**注意事项：**连接的时候，不能在结尾加分号。

练习：尝试使用错误信息登录。

## 4.mysql常用管理命令

quit; 退出服务器连接

show databases; 显示服务器上当前所有的数据库

use 数据库名; 进入指定的数据库

show tables; 显示当前数据库中所有的数据表

desc 表名; 描述表中都有哪些列(表头)

练习：使用root用户登录查看服务器上有哪些数据库。

练习：查看phpmyadmin和test中分别都有哪些数据表。

练习:查看phpmyadmin中以下表都有哪些列

pma\_\_history 6

pma\_\_recent 2

pma\_\_favorite 2

## 5.SQL命令

SQL: Structured Query Language，结构化查询语言，用于操作关系型数据库服务器，对数据执行增删改查等操作。

SQL命令的两种执行方式

(1)交互模式: 客户端输入一行，点击回车，服务器执行一行。适用于临时性的查看数据

(2)脚本模式: 客户端把要执行的多行命令写在一个文本文件中，一次性的提交给服务器。适用于批量的操作数据

不能进入mysql数据库

mysql -uroot < c:/xampp/.../02.sql 回车

练习：编写脚本文件03.sql，向服务器提交多行SQL命令，包括显示所有的数据库，进入到phpmyadmin，显示所有的数据表，显示pma\_\_recent有哪些列。如果运行没问题，尝试将其中一行写错，看结果。

**SQL语法规范**

(1)每条SQL命令可以跨越多行，遇到英文分号作为结束

(2)假如某一条命令出现语法错误，则此条语句以及后边所有的语句不会再执行

(3)SQL命令不区分大小写，习惯上关键字用大写，非关键字用小写

(4)SQL命令中可以使用单行注释(#...)和多行注释(/\*...\*/)，注释的内容不会被服务器所执行

## 6.常用的SQL命令

(1)丢弃指定的数据库，如果存在的话

DROP DATABASE IF EXISTS xuezi;

(2)创建新的数据库

CREATE DATABASE xuezi;

(3)进入创建的数据库

USE xuezi;

(4)创建保存数据的表

CREATE TABLE student(

sid INT,

name VARCHAR(8),

sex VARCHAR(1),

score INT

);

(5)向数据表中插入数据

INSERT INTO student VALUES('1','tom','m','85');

(6)查询表中所有的数据

SELECT \* FROM student;

(7)修改数据

UPDATE student SET name='lucy',score='100' WHERE sid='2';

(8)删除数据

DELETE FROM student WHERE sid='3';

练习：编写脚本文件04\_td.sql，先丢弃数据库td，如果存在；创建数据库td；进入到该数据库。

练习：在td数据库中创建保存员工数据的表emp，包含员工的编号eid、姓名ename、地址addr、电话phone

练习：编写脚本文件05\_dang.sql，先丢弃再创建数据库dang;进入该数据库，创建保存图书数据的表book，包含图书编号bid，书名title，价格price；插入两条数据，修改1条，删除1条，查询结果。

课后任务

(1)复习今天内容

(2)练习

编写脚本文件xz.sql，先丢弃在创建数据库xz，进入该数据库，创建保存用户数据的表user，包含(uid, uname, upwd, email, phone, sex, userName真实姓名, regTime 注册时间,isOnline 是否在线)，插入5条记录，删除1条，修改1条，查询结果。

(3)预习mysql中的列类型 手册第11章

## 7.关系型数据库逻辑结构

Server -> Database -> Table ->Row ->Column

mysql

服务器端：开启服务 mysqld.exe

客户端：连接服务器 mysql.exe

连接

mysql.exe -h127.0.0.1 -P3306 -uroot -p

mysql -uroot

## 8. 常用管理命令

quit/show databases/use 数据库名/show tables/desc 表名

SQL命令

交互模式

脚本模式 mysql -uroot < C:/....

DROP DATABASE IF EXISTS ...;

CREATE DATABASE ...;

USE ...;

CREATE TABLE ...(

列名 列类型,

列名 列类型

);

INSERT INTO 表名 VALUES(...);

SELECT \* FROM 表名;

UPDATE 表名 SET 列名='值',列名='值' WHERE ...;

DELETE FROM 表名 WHERE ...;

## 标准SQL命令分类

|  |
| --- |
| DDL: Data Define Language 定义数据结构  CREATE/DROP/ALTER  DML: Data Manipulate Languge 操作数据  INSERT/UPDATE/DELETE  DQL: Data Query Language 查询数据  SELECT  DCL: Data Control Language 控制用户权限  GRANT(授权)/REVOKE(收权) |

## 1.计算机存储字符

(1)如何存储英文字符

ASCII: 总共有128个,对所有的英文字母和符号进行编码

abc 979898

Latin-1: 总共有256，兼容ASCII码，同时对欧洲符号进行了编码，mysql默认使用这种编码。

(2)如何存储中文字符

GB2312: 对常用的6千多汉字进行编码，兼容ASCII码

GBK: 对2万多汉字进行了编码，兼容GB2312

BIG5: 台湾繁体字编码，兼容ASCII码

Unicode: 对世界上主流的国家常用的语言进行了编码，兼容ASCII，不兼容GB2312,GBK,BIG5，具体分为UTF-8，UTF-16，UTF-32存储方案

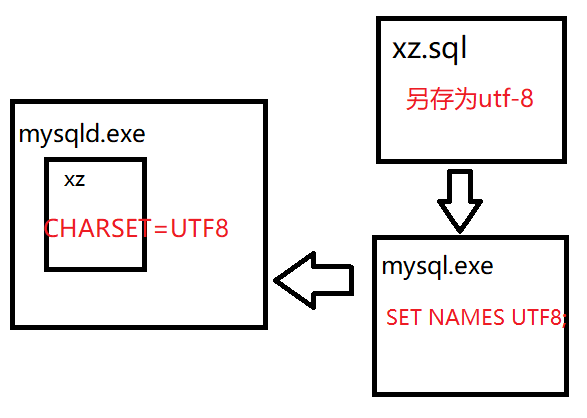
(3)解决mysql存储中文乱码

使用UTF-8编码形式

sql脚本文件另存为的编码

客户端连接服务器的编码(SET NAMES UTF8)

服务器端创建数据库使用的编码(CHARSET=UTF8)



练习：编写脚本文件01\_dang.sql,创建数据库dang，设置编码为utf8；创建保存图书的数据表book，包含bid，title标题，author作者，price价格，publish出版社，pubTime出版时间

插入4条记录，查询结果。

## 2.mysql中的列类型

创建数据表的时候，指定的列可以存储的数据类型

CREAT TABLE book( bid 列类型 );

### (1)数值类型——引号可加可不加

TINYINT 微整型，占1个字节 范围-128~127

SMALLINT 小整型，占2个字节 范围-32768~32767

INT 整型，占4个字节

范围 -2147483648~2147483647

BIGINT 大整型，占8个字节

FLOAT(M,D) 单精度浮点型，占4个字节，范围3.4E38，范围比INT大的多，可能产生计算误差。

DOUBLE(M,D) 双精度浮点型，占8个字节，范围比BIGINT大的多

DECIMAL(M,D) 定点小数，不会产生计算误差，M代表总的有效位数，D代表小数点后的有效位数

BOOL 布尔型，只有两个结果TRUE、FALSE(不能加引号)，真正存储数据的时候，会自动变成1和0；也可以直接使用1和0；数据库的列类型自动变成TINYINT

### (2)日期时间类型——必须加引号

DATE 日期型 '2018-12-31'

TIME 时间型 '14:37:30'

DATETIME 日期时间型 '2018-12-31 14:37:30'

### (3)字符串类型——必须加引号

VARCHAR(M) 变长字符串，不会产生空间浪费，操作速度相对慢，M的最大值是65535

CHAR(M) 定长字符串，可能产生空间浪费，操作速度快，用于存储手机号码、身份证号等固定长度字符，M最大值是255

TEXT(M) 大型变长字符串，M最多存2G

|  |  |  |
| --- | --- | --- |
|  | CHAR(5) | VARCHAR(5) |
| a | a\0\0\0\0 | a\0 |
| ab | ab\0\0\0 | ab\0 |
| abc | abc\0\0 | abc\0 |
| abcde | abcde | abcde |
| 一二三 | 一二三\0\0 | 一二三\0 |

CREATE TABLE t1(

id INT,

age TINYINT,

commentCount INT,

price DECIMAL(6,2), # 9999.99

phone CHAR(11),

article VARCHAR(3000),

sex BOOL, #1 男 0女

pubTime DATE

);

使用合理的列类型；

练习：编写脚本文件02\_xuezi.sql，先丢弃再创建数据库xuezi，使用utf8编码，进入该数据库，创建保存笔记本数据的表laptop，包含lid，title，price，stockCount库存量，shelfTime上架时间，isIndex是否为首页推荐

插入4条记录

练习：编写脚本文件03\_tedu.sql，创建数据库tedu，设置编码为utf8，进入该数据库。

创建保存部门数据的表dept，包含did、dname部门名称、empCount部门员工数量，插入以下数据

10 研发部 3， 20 市场部 2， 30 运营部 2

创建保存员工数据的表emp，包含eid、ename、sex、birthday生日、salary工资、deptId所属部门编号

插入若干条记录

## 3.列约束

mysql可以对插入的数据进行特定的验证，只有满足条件才允许插入到数据表中，否则认为是非法的插入。

例如:一个人的性别只能是男或者女，一个人的年龄0~100

### (1)主键约束——PRIMARY KEY

声明了主键约束的列上不能出现重复的值，表中查询的记录会按照主键从小到大排序——加快查找速度；通常主键添加到编号列中。

注意：一个表中只能出现一个主键，主键列上不允许插入NULL。

|  |
| --- |
| NULL表示空，在插入数据时，无法确定要保存的值；  例如：无法确定员工的工资、姓名、声明... |

课后任务

(1)复习今天内容

(2)练习

编写脚本文件xz.sql，创建数据库xz，设置编码utf；

创建保存笔记本家族分类的表laptop\_family( fid, fname 分类名称, laptopCount笔记本数量 )

10 联想 2 20 戴尔 2 30 小米 3

创建笔记本数据表laptop(lid，title，price，spec规格，detail详情，shelfTime上架时间，isOnsale是否在售，familyId 所属分类编号)

插入若干条数据

day03

复习

解决中文乱码

ASCII/Latin-1/GB2312/GBK/BIG5/unicode

utf-8 脚本文件，客户端连接服务器，创建数据库

列类型

数值型

TINYINT/SMALLINT/INT/BIGINT

FLOAT/DOUBLE/DECIMAL

BOOL TRUE/FALSE 1/0

日期时间型

DATE/TIME/DATATIME

字符串型

VARCHAR/CHAR/TEXT

## 1.列约束

### (1)主键约束——PRIMARY KEY

### (2)非空约束——NOT NULL

声明了非空约束的列上不允许插入NULL值

### (3)唯一约束——UNIQUE

声明了唯一约束的列上不能插入重复的值，允许插入NULL，甚至多个NULL；一个表中可以出现多个唯一约束

|  |
| --- |
| 说明：NULL这个值比较特殊，它和任何值都不等，甚至和自身都不等。 |

练习：给笔记本家族表中，分类名称添加唯一约束，插入数据测试。

### (4)检查约束——CHECK

检查约束可以对插入的数据进行自定义验证

CREATE TABLE student(

score TINYINT CHECK(score>=0 AND score<=100)

);

mysql不支持检查约束，会降低数据的插入速度。

### (5)默认值约束——DEFAULT

可以使用DEFAULT关键字声明默认值，有两种方式可以使用默认值

INSERT INTO laptop\_family VALUES(50,'华硕',DEFAULT);

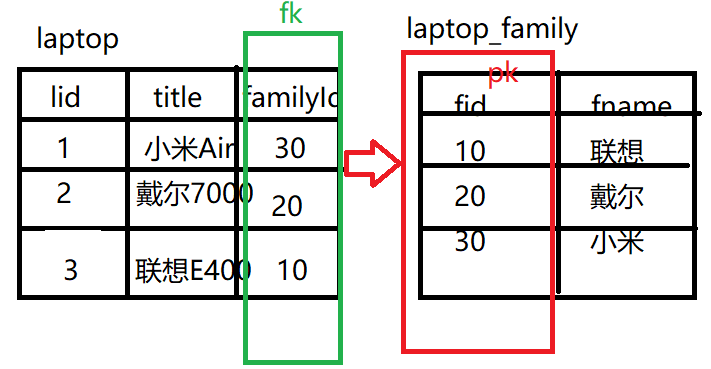
INSERT INTO laptop\_family(fid,fname) VALUES(60,'神州');

练习：设置笔记本表中价格默认值为6999,分别使用两种方式调用默认值。

### (6)外键约束——FOREIGN KEY

声明了外键约束的列上，取值必须在另一个表中的主键列上出现过，两者的列类型要保持一致，允许使用NULL

FOREIGN KEY(外键列) REFERENCES 另一数据表(主键列)



## 2.mysql中的自增列

AUTO\_INCREMENT: 自动增长，假如一个列声明了自增列，无需手动赋值，直接赋值为NULL，会获取当前的最大值，然后加1.

注意：

只适用于整型的列上

自增列允许手动赋值

练习：编写脚本文件01\_tedu.sql，创建数据库tedu，设置编码为UTF8；

创建保存部门数据的表dept，包含(did, dname)

10 研发部 20市场部 30运营部 40 测试部

创建保存员工数据的表emp，包含(eid, ename, sex, birthday, salary, deptId)

使用恰当的列约束和自增；

插入15条记录。

## 3.简单查询

### (1)查询特定的列

示例：查询所有员工的姓名、生日

SELECT ename,birthday FROM emp;

练习：查询所有员工的编号、姓名、性别、工资

SELECT eid,ename,sex,salary FROM emp;

### (2)查询所有的列

SELECT \* FROM emp;

SELECT eid,ename,sex,birthday,salary,deptId FROM emp;

### (3)给列起别名

示例：查询所有员工的姓名和工资，使用中文别名

SELECT ename AS 姓名,salary AS 工资 FROM emp;

练习：查询所有员工的编号，姓名，性别，生日；使用中文别名

SELECT eid AS 编号,ename AS 姓名,sex AS 性别,birthday AS 生日 FROM emp;

练习：查询所有员工的编号和姓名，使用一个字母作为别名

SELECT eid a,ename b FROM emp;

|  |
| --- |
| 在起别名的时候，AS关键字可以省略 |

### (4)显示不同的记录/合并相同的记录

练习：查询出都有哪些性别的员工

SELECT DISTINCT sex FROM emp;

练习：查询出员工都在哪些部门

SELECT DISTINCT deptId FROM emp;

### (5)查询时执行计算

示例：计算2+3-4+5\*6/3

SELECT 2+3-4+5\*6/3;

练习：查询所有员工的姓名及其年薪

SELECT ename,salary\*12 FROM emp;

练习：假如每个员工工资加500，年终奖加5000，查询所有员工的姓名及其年薪，给列起中文别名。

SELECT ename 姓名,(salary+500)\*12+5000 年薪 FROM emp;

### (6)查询结果集排序

示例:查询所有部门表数据,结果集按照编号从小到大排序

SELECT\* FROM dept ORDER BY did ASC; #ascendant

示例:查询所有部门表数据,结果集按照编号从大到小排序

SELECT\*FROM dept ORDER BY did DESC;#descendant;

练习：查询员工所有的列，结果集按照工资降序排列

SELECT \* FROM emp ORDER BY salary DESC;

练习：查询员工所有的列，结果集按照年龄从小到大排列

SELECT \* FROM emp ORDER BY birthday DESC;

练习：查询员工所有的列，结果集按照姓名的升序排序

SELECT \* FROM emp ORDER BY ename;

练习：查询员工所有的列，结果按照工资降序排序，如果工资相同，按照姓名升序排列

SELECT \* FROM emp ORDER BY salary DESC,ename;

练习：查询员工所有的列，要求女员工显示在前，如果性别相同按照生日降序排列。

SELECT \* FROM emp ORDER BY sex,birthday DESC;

|  |
| --- |
| ORDER BY 可以按照数值、字符串、日期时间排序  默认是按照升序排列(ASC) |

### (7)条件查询

示例：查询出编号为5的员工所有列

SELECT \* FROM emp WHERE eid=5;

练习：查询出姓名为king的员工的编号，工资，姓名，生日

SELECT eid,salary,ename,birthday FROM emp WHERE ename='king';

练习：查询出20号部门下员工所有列。

SELECT \* FROM emp WHERE deptId=20;

练习：查询出工资在6000以上员工所有列

SELECT \* FROM emp WHERE salary>=6000;

|  |
| --- |
| 比较运算符：> < = !=(不等于) >= <= |

练习：查询出1993-1-1后出生的员工所有列

SELECT \* FROM emp WHERE birthday>'1993-1-1';

练习：查询出不在10号部门的员工所有列

SELECT \* FROM emp WHERE deptId!=10;

练习：查询出没有明确部门的员工所有列

SELECT \* FROM emp WHERE deptId IS NULL;

练习：查询出有明确部门的员工所有列

SELECT \* FROM emp WHERE deptId IS NOT NULL;

练习：查询出工资在6000以上的男员工所有列

SELECT \* FROM emp WHERE salary>=6000 AND sex=1;

练习：查询出工资在7000~10000之间员工所有列

SELECT \* FROM emp WHERE salary>=7000 AND salary<=10000;

SELECT \* FROM emp WHERE salary BETWEEN 7000 AND 10000;

练习：查询出工资不在7000~10000之间员工所有列

SELECT \* FROM emp WHERE salary NOT BETWEEN 7000 AND 10000;

SELECT \* FROM emp WHERE salary<7000 OR salary>10000;

练习：查询出1990年之前和1995年之后出生的员工所有列

SELECT \*FROM emp WHERE birthday<'1990-1-1' OR birthday>'1995-12-31';

练习：查询出20号和30号部门员工所有的列

SELECT \* FROM emp WHERE deptId=20 OR deptId=30;

SELECT \* FROM emp WHERE deptId IN(20,30);

练习：查询出不在20号和30号部门员工所有的列

SELECT \* FROM emp WHERE deptId NOT IN(20,30);

|  |
| --- |
| IS NULL / IS NOT NULL  AND / OR  BETWEEN..AND.. / NOT BETWEEN..AND..  IN( ) / NOT IN( ) |

### (8)分页查询

假如查询的结果集有太多的数据，一次显示不完，可以使用分页显示。

需要两个条件：当前的页码，每页的数据量

每页的开始=(当前的页码-1)\*每页数据量

|  |
| --- |
| 分页查询语法：  SELECT \* FROM emp LIMIT start,count;  start:每页的开始  count:每页数据量 |

假设每页显示5条数据

第1页：SELECT \* FROM emp LIMIT 0,5;

第2页：SELECT \* FROM emp LIMIT 5,5;

第3页：SELECT \* FROM emp LIMIT 10,5;

练习：假设每页显示6条记录，查询前3页。

第1页：SELECT \* FROM emp LIMIT 0,6;

第2页：SELECT \* FROM emp LIMIT 6,6;

第3页：SELECT \* FROM emp LIMIT 12,6;

课后任务

(1)复习今天内容

(2)练习：

删除代码，保留注释，重新编写代码。

(3)查看学子商城数据库资料，尝试创建

# 02.js

01.day

复习

性别使用整型(布尔型)

性别显示 男/女 男孩/女孩 男人/女人

存储1/0

日期时间使用大整型(BIGINT)

存储的是距离计算机元年的毫秒数

1000毫秒=1秒

计算机元年 1970年1月1日 0时0分0秒

后期结合JS可以得到日期时间中任意的时间信息

49\*365\*24\*60\*60\*1000

## 1.简单查询——模糊条件查询

示例：查询出姓名中含有字母e的员工所有列

SELECT \* FROM emp WHERE ename LIKE '%e%';

练习：查询出姓名中倒数第2个字符为e的员工所有列

SELECT \* FROM emp WHERE ename LIKE '%e\_';

练习：查询出姓名中以e结尾的员工所有列

SELECT \* FROM emp WHERE ename LIKE '%e';

|  |
| --- |
| SQL中提供了两个模糊查询的字符  % 可以匹配任意个字符 >=0  \_ 可以匹配任意1个字符 =1  注意：在模糊条件查询中必须使用关键字LIKE，不能使用= |

## 2.复杂查询

### (1)聚合查询/分组查询

示例：查询出所有员工的数量

SELECT COUNT(eid) FROM emp;

SELECT COUNT(\*) FROM emp; #推荐写法

练习：使用员工的姓名计算员工数量

SELECT COUNT(ename) FROM emp;

练习：使用员工部门编号计算员工数量

SELECT COUNT(deptId) FROM emp;

练习：查询所有男员工的数量

SELECT COUNT(\*) FROM emp WHERE sex=1;

|  |
| --- |
| 聚合函数  函数就是一个功能体，需要提供若干个数据，产出某个结果。——饺子机  COUNT() 总数量  SUM() 总和  AVG() 平均  MAX() 最大  MIN() 最小 |

练习：查询出所有员工的工资总和

SELECT SUM(salary) FROM emp;

练习：查询出所有员工的平均工资

SELECT AVG(salary) FROM emp;

SELECT SUM(salary)/COUNT(\*) FROM emp;

练习：查询出男员工工资最高的

SELECT MAX(salary) FROM emp WHERE sex=1;

练习：查询出年龄最大的员工

SELECT MIN(birthday) FROM emp;

|  |
| --- |
| 分组查询：只能查询分组条件和聚合函数 |

练习：查询出男女员工的平均工资，最高工资

SELECT sex,AVG(salary),MAX(salary) FROM emp GROUP BY sex;

练习：查询出每个部门的员工数量，最高工资，最低工资

SELECT deptId,COUNT(\*),MAX(salary),MIN(salary) FROM emp GROUP BY deptId;

|  |
| --- |
| 函数补充  YEAR() 获取日期中的年份  MONTH() 获取日期中的月份 |

练习：查询出1993年出生的员工所有列

SELECT \* FROM emp WHERE YEAR(birthday)=1993;

练习：查询出5月份出生的员工所有列

SELECT \* FROM emp WHERE MONTH(birthday)=5;

### (2)子查询

|  |
| --- |
| 把一个SQL语句的查询结果作为另一个SQL语句的查询条件 |

示例：查询出研发部员工所有的列

步骤1：查询出研发部的部门编号——10

SELECT did FROM dept WHERE dname='研发部';

步骤2：根据研发部部门编号查询员工

SELECT \* FROM emp WHERE deptId=10;

综合：

SELECT \*FROM emp WHERE deptId=(SELECT did FROM dept WHERE dname='研发部');

练习：查询出比tom工资高的员工有哪些

步骤1：查询出tom的工资——6000

SELECT salary FROM emp WHERE ename='tom';

步骤2：查询出比6000工资高的

SELECT \* FROM emp WHERE salary>6000;

综合：

SELECT \* FROM emp WHERE salary>( SELECT salary FROM emp WHERE ename='tom');

练习：查询出和tom同一年出生的员工

步骤1：查询出tom出生的年份

SELECT YEAR(birthday) FROM emp WHERE ename='tom';

步骤2：查询出1990年出生的员工

SELECT \* FROM emp WHERE YEAR(birthday)=1990;

综合：

SELECT \* FROM emp WHERE YEAR(birthday)=(

SELECT YEAR(birthday) FROM emp WHERE ename='tom'

);

### (3)多表查询

示例：查询所有的员工姓名及其部门名称。

SELECT ename,dname FROM emp,dept;

错误：笛卡尔积

多表查询如何避免笛卡尔积：添加查询条件

SELECT ename,dname FROM emp,dept WHERE deptId=did;

|  |
| --- |
| 上述多表查询语法是SQL-92中的，无法查询出没有部门的员工，也无法查询出没有员工的部门。  SQL-99中提出了新的多表查询语法。 |

(1)内连接 INNER JOIN..ON 和SQL92结果一致

SELECT ename,dname FROM emp INNER JOIN dept ON deptId=did;

(2)左外连接 LEFT OUTER JOIN.. ON

SELECT ename,dname FROM emp LEFT OUTER JOIN dept ON deptId=did;

查询结果是左侧所有的记录都显示；OUTER可以省略

(3)右外连接 RIGHT OUTER JOIN..ON

SELECT ename,dname FROM emp RIGHT OUTER JOIN dept ON deptId=did;

查询结果是右侧表中所有记录都显示，OUTER可以省略

(4)全连接 FULL JOIN ON

显示左侧和右侧所有的记录——mysql不支持

UNION 合并相同的项

UNION ALL 不合并相同的项

(SELECT ename,dname FROM emp LEFT OUTER JOIN dept ON deptId=did)

UNION

(SELECT ename,dname FROM emp RIGHT OUTER JOIN dept ON deptId=did);

有基础的学员

程序员必做50题

<https://wenku.baidu.com/view/af66e2f14afe04a1b071de42.html>

## 1.JS概述

### (1)历史

1995年，JS出现在Netscape的浏览器中

1997年，JS提交给了ECMA, 出现了ECMAScript标准规范

2009年，JS遵循CommonJS规范，开始向服务器端发展

### (2)现状

既可以运行在客户端浏览器，也可以运行在服务器端NodeJS

### (3)特点

解释型语言，编译一行执行一行。

弱类型语言

基于对象

跨平台

### (4)应用场景

制作浏览器端的交互效果

创建移动App

创建web服务器、操作数据库等服务器端应用

2D绘图

## 2.JS的执行环境

(1)浏览器自带的JS解释器

(2)NodeJS下的JS解释器

[https://nodejs.org](https://nodejs.org/) NodeJS下载地址

在命令行下 node -v 查看当前的NodeJS版本号

(3)执行JS代码

浏览器下：

创建01\_hello.js和01\_hello.html

把js文件嵌入到html文件中

|  |
| --- |
| <script src="01\_hello.js"></script> |

NodeJS下：

node C:/xampp/.../01\_hello.js 回车

## 3.JS语法规范

(1)区分大小写

(2)每行代码结束的分号可加可不加，习惯都加

(3)分为单行注释(//...)和多行注释(/\*...\*/)

## 4.变量

变量是用于存储数据的容器

x=1

y=2

## (1)声明变量

var x=1; 在内存在创建空间，名称叫x；把数据1存放到空间x中。

练习：声明多个变量，分别保存一个员工的编号、姓名、性别、生日、工资、部门编号，并打印出来。

## (2)变量的命名规则

变量的名称由字母、数字、美元符号、下划线组成的；不能以数字开头。

多个连词之间命名(驼峰命名、下划线命名)

userName userAge userPassword

user\_name user\_age

不能使用关键字和保留字作为变量名



### (3)声明变量未赋值

var num;

变量声明后未赋值，结果为undefined

### (4)一次性声明多个变量

var a=1,b=2,c;

多个变量之间用逗号隔开。

练习：声明多个变量，分别保存语文、数学、总成绩；最后打印总成绩。

## 5.常量

一旦声明不能重新赋值

例如：圆周率、春节的日期...

const PI=3.14;

课后任务

(1)复习今天内容，整理思维导图

(2)练习

声明变量保存圆的半径，声明常量保存圆周率；声明变量保存圆的面积和周长，并打印出来。

(3)预习js的数据类型和运算符

02.day

复习

复杂查询

聚合查询/分组查询/子查询/跨表查询

查询所有男员工的所有列，获取工资最高的前三个人。

SELECT \* FROM emp

WHERE sex=1

ORDER BY salary DESC

LIMIT 0,3;

开发环境

浏览器/NodeJS

变量 var a=1; 字母、数字、下划线、美元符号，不能以数字开头。

常量 const PI=3.14;

## 1.数据类型

数据分为原始类型和引用类型

原始类型分为数值型、字符串型、布尔型、未定义型、空

### (1)数值型

分为整型和浮点型

整型在内存中占4个字节，浮点型占8个字节

八进制 以0开头， 例如 013 -> 11

十六进制 以0X开头， 例如 0XA -> 10

A~F 代表10~15 不区分大小写

0XFF -> 255

浮点型分为普通小数和指数型小数

3.14E6 -> 31400000

3.14E-6 -> 0.00000314

typeof 检测数据类型

### (2)字符串型

数据被引号包含就是字符串类型，不区分单双引号

查看任意一个字符的Unicode码

'a'.charCodeAt() //97

### (3)布尔型

true/false

在程序中表示真或者假

一般用于是否的结果，例如是否登录，是否注册，是否在售，是否为会员...

(4)未定义型

声明了变量未赋值，结果就是undefined

(5)空——null

常和引用类型数据一起使用

## 2.数据类型转换

### (1)隐式转换

①数字+字符串 数字转换成字符串

1+'2' // '12'

②数字+布尔型 布尔型转成数字 true->1 false->0

3+true //4

3+false //3

③字符串+布尔型 布尔型转成字符串

'5'+true //'5true'

JS中加号(+)的作用

执行加法运算

执行字符串拼接

减法(-)、乘法(\*)、除法(/) 隐式换行

将运算符两端的数据转为数值型(自动调用Number)，如果转换失败，返回NaN(Not a Number)，不是一个数字，任何值和NaN执行减乘除运算符都会返回NaN

练习：查看以下程序的运行结果

var num1=3,num2=true,num3='tedu';

console.log(num1+num2+num3);//'4tedu'

console.log(num2+num3+num1);//'truetedu3'

console.log(num3+num1+num2);//'tedu3true'

### (2)强制转换

①将任意的数据强制转为数值型

Number()

|  |
| --- |
| Number('a'); //NaN  Number('1'); //1  Number('1a'); //NaN  Number(undefined); //NaN  Number(null); //0  Number(true); //1 |

②将数据转为整型

parseInt()

常用于将字符串或数字转为整型，其它的数据返回NaN；如果要转换的字符串以非数字开头，也返回NaN

③将数据转为浮点型

parseFloat()

和parseInt的用法基本一致，只是转的数据是浮点型。

④数值型和布尔型转为字符串型

toString()

var num=10;

num.toString(); // '10'

## 3.运算符

表达式: 由运算符连接的操作数据，所组成的形式。

运算符分为算术运算符、比较运算符、逻辑运算符、位运算符、赋值运算符、三目运算符

### (1)算术运算符

+ - \* / % ++ --

% 取余

++ 自增，在原来的基础之上加1

-- 自减，在原来的基础之上减1

console.log(num++); 先打印num的值，再执行自增

console.log(++num); 先执行自增，再打印num的值

练习：

var num=3;

console.log(num-- + --num);

### (2)比较运算符

> < >= <= != == ===(全等于) !== 全不等于

返回一个布尔型的值

== 只是比较两个值是否相等

=== 不仅比较值，还会比较类型是否相等

|  |
| --- |
| 如果数据类型不同，会发生数据类型转换 |

3>'10' 字符串转成数字10

'3'>'10' 两个字符串比较的是Unicode码

3>'10a' //false

3<'10a' //false

3=='10a' //false

'10a' -> NaN

NaN和任何值比较(> < >= <= == ===)都返回false

NaN==NaN 返回false

### (3)逻辑运算符

或者( || ) 关联的两个条件只需要满足其一, 结果是true，否则false

并且( && ) 关联的两个条件都是true，结果是true，否则false

非(反向) ! !false -> true !true -> false

**逻辑短路**

&& 当第一个条件为false的时候，就不需要再执行第二个条件

|| 当第一个条件为true的时候，就不需要再执行第二个条件

注意：逻辑短路无需关注最终结果是true还是false，重点是看是否会执行第二个表达式。

练习：以下程序是否会报错

|  |
| --- |
| var num=3;  num>5 && console.log(a);  num<1 || console.log(a); |

练习：声明变量保存一个人的年龄，如果满18岁，打印'成年人'

### (4)位运算符(了解)

在执行运算的时候，计算机会把数据转成二进制进行运算。

1 10 11 100 101 110 111 1000 1001 1010

2 4 8

100110=100000+100+10

32 + 4 + 2

25 = 16+8+1

10000+1000+1=11001

按位与( & ) 上下两位都是1，结果是1，否则是0

按位或( | ) 上下两位含有1，结果是1，否则是0

按位异或( ^ ) 上下两位不同为1，相同为0

按位右移( >> ) 删除二进制的最后一位或者多位，每次缩小到原来的一半或者更多。

按位左移( << ) 在二进制的最后添加0，成倍增加

课后任务

(1)复习今天内容，整理思维导图

(2)练习：

声明变量保存一个年份，判断这个年份是否为闰年，结合逻辑短路，如果是闰年，打印"闰年".

闰年：4年一个闰年(能被4整除)，并且不能被100整除，或者能被400整除。

(3)预习逻辑结构 if if-else switch

复习

数据类型

原始类型和引用类型

数值型、字符串型、布尔型、未定义型、空

检测数据类型 typeof

数据类型转换

隐式转换

+ - \* / % ++ -- 3>'10'

强制转换

Number()/parseInt('3a')/parseFloat()/num.toString()

运算符

算术运算符 + - \* / % ++ --

比较运算符 > < >= <= == != === !==

逻辑运算符 || && ! 逻辑短路

位运算符 & | ^ >> <<

赋值运算符

三目运算符

## 1.赋值运算符

= += -= \*= /= %=

练习：声明变量保存商品的价格，价格在原来基础之上打八折，然后打印价格。

## 2.三目运算符

由三个表达式或者数据组成的形式。

条件表达式 ? 表达式1 : 表达式2

如果条件表达式为true，执行表达式1

如果条件表达式为false，执行表达式2

## 3.浏览器端函数

alert() 弹出警示框(消息框)

prompt() 弹出提示框(输入框)，需要使用变量来接收输入的值；值的类型是字符串型。

## 4.流程控制

程序=数据 + 算法

程序分为顺序执行、选择执行、循环执行

### (1)if语句

|  |
| --- |
| if(条件表达式){  语句;  } |

如果条件表达式为true，会执行后边的语句

在条件表达式中，有一些值默认转成false

0、NaN、''、undefined、null

### (2)if-else语句

|  |
| --- |
| if(条件表达式){  语句1;  }else{  语句2;  } |

如果条件表达式为true，执行语句1，否则条件表达式为false，执行语句2

|  |
| --- |
| 如果if或者else后只有一行语句，则后边的大括号可以省略的。 |

练习：使用弹出提示框分别输入商品的单价和数量，计算出商品的总价，如果总价满500打八折；使用变量保存账户余额，如果余额足够支付，打印pay success，否则打印pay error。

07\_exercise.html 07\_exercise.js

### (3)if-else嵌套

用于判断多种情况

|  |
| --- |
| if(条件表达式1){  语句1;  }else if(条件表达式2){  语句2;  }else ... if(条件表达式n){  语句n;  }else{  语句n+1; //以上所有的条件表达式都是false  } |

### (4)switch-case语句

是一种特殊的分支语句，可以根据一个表达式的值，来选择执行不同的程序

|  |
| --- |
| switch(表达式){  case 值1: //如果表达式的值是值1  语句1;  break;  ....  case 值n:  语句n;  break;  default:  语句n+1; //以上所有的表达式和值比较都是false  } |

注意：在case中，表达式和值比较实用的是全等于比较，要求值和类型都满足才是true

**对比if-else嵌套和switch-case语句**

相同点：两者都可以用于多项分支语句

不同点：if-else既可以判断相等，也可以判断不等的情况，使用范围更广；switch-case只能用于全等于的比较，结构上更为清晰，执行效率相对高。

## 1.循环

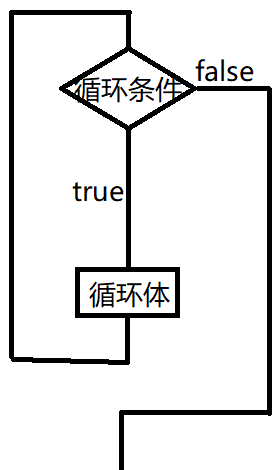
循环：就是一遍又一遍执行相同或者相似的代码

循环的要素

循环条件：重复的次数

循环体：执行的代码

### (1)while循环



|  |
| --- |
| while(循环条件){  循环体;  } |

break 可以结束任何形式的循环，break执行后，后边所有的循环体和循环条件都不再执行。

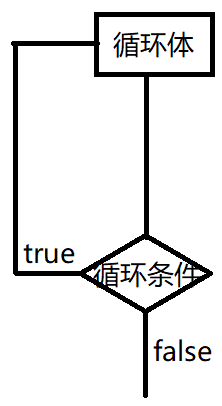
《JavaScript高级程序设计》第三版

练习：声明变量保存1个数字，无限循环弹出提示框，输入数字；如果输入的比保存的大，警示框提示'big'，如果输入的比保存的小，警示框提示'small'，否则警示框提示'right'，结束循环。

03\_break.html

思路：循环前声明变量保存数字，无限循环的循环体中弹出提示框，获取输入的值，再和之前的变量比较(if-else嵌套)。

### (2)do-while循环



|  |
| --- |
| do{  循环体;  }while(循环条件); |

不管循环条件是否为true，都会先执行一次循环体

练习：打印100~1之间所有整数

练习：打印70 75 80 85 90 95 100

练习：计算1~100之间所有能被3整除的数字的和。

练习：声明变量保存密码'123456'；无限循环弹出提示框，输入密码，如果输入正确，警示框弹出'login success'，并结束循环 break

05\_dowhile.js 05\_dowhile.html

### (3)for循环

|  |
| --- |
| for(初始值;循环条件;增量){  循环体  } |
| ①执行初始值  ②判断循环条件  ③如果循环条件是true执行循环体，是false结束循环  ④如果执行完循环体，执行增量  ⑤重新执行第二步 |

## 2.break和continue

break: 结束循环，后续不会再执行其它的循环

continue: 跳过本次循环体，继续执行下一次循环

## 3.循环嵌套

while、do-while、for循环三者之间可以相互嵌套

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

day05

复习

循环

两要素：循环条件，循环体

while(循环条件){ 循环体 }

do{ 循环体 }while(循环条件)

for(初始值;循环条件;增量){ 循环体 }

break和continue

循环嵌套

for while do-while

|  |
| --- |
| var i=0,sum=0;  do{  i++;  if(i%2==0){  continue;  }  if(i%5==0){  break;  }  sum+=i;  }while(i<10); |

## 1.函数

parseInt()/parseFloat()...

分为系统函数和自定义函数

自定义函数

function: 功能体,函数，用于封装反复执行的代码，可以接受若干个数据，返回处理的结果。——饺子机

### (1)创建普通的函数

|  |
| --- |
| function 函数名称(){  函数体; //封装的反复执行的代码  } |

调用

函数名称()

练习：使用函数封装1+2的计算结果，调用三次。

练习：使用函数封装计算1~100之间所有整数的和，并调用三次。

### (2)创建带有参数的函数

|  |
| --- |
| function 函数名称(参数列表){//形参 ->形式上的参数  函数体;  } |

调用

函数名称(参数列表); //实参 -> 传递的数据

参数列表：可以是0个或者多个，之间用逗号隔开；创建时的参数称为形参，调用时的参数称为实参，实参会赋值给形参。

形参可以理解为是一个变量，只是未赋值，默认是undefined.

练习：创建函数，传递1个参数，计算1~任意数字之间的和。

练习：创建函数，传递2个参数，计算任意两个年份之间的闰年个数。 n1~n2

### (3)带有返回值的函数

|  |
| --- |
| function 函数名称(参数列表){  函数体;  return 值;  } |

调用

函数名称(参数列表)

return表示函数执行后，返回的结果；可以保存下来。

注意事项：

①如果没有return和return后没有值，结果都是undefined

②return后的所有代码都不会执行

练习：创建函数getMax，返回任意两个数字中的最大值。

练习: 创建函数getMax2，返回任意三个数字的最大值。

练习：创建函数getStatus，传递状态码，返回对应的中文

1-等待付款 2-等待发货 3-运输中 4-已签收 5-已取消 其它-无法追踪

练习：创建函数getDays，传递任意一个年份，返回对应的天数。

## 2.变量作用域

### (1)作用域

变量和函数的可访问范围，分为两种

函数作用域：在函数中使用var关键字声明的变量，只能在函数的内部访问。

全局作用域：在函数外部使用var声明的变量，可以在任意的位置访问

注意：在函数内使用var声明的变量是局部变量，不使用var声明的变量是全局变量。

### (2)变量声明提升

JS程序执行前，会将使用var声明的变量提升到所在作用域的最前边，赋值还是在原来的位置。

## 3.函数的作用域

函数的可访问范围，也分为全局作用域和函数作用域。

(1)函数声明提升

和变量声明提升一样，JS程序执行前，会将function创建的函数提升到所在作用域的最前边，调用可以在任意合法的位置。

## 4.递归调用

在函数的内部调用自身

递归要有结束的条件，结合return来使用。

练习：使用递归来计算1~任意数字之间的和。

斐波那契数列

1 1 2 3 5 8 13 21 34....

day06

## 1.匿名函数

没有名称的函数 function(){ }

### (1)创建函数

函数声明方式

|  |
| --- |
| function fn(){ } |

函数表达式方式

|  |
| --- |
| var fun=function(参数列表){ 函数体 }  此时的变量fun就是函数名称 |

对比fun和fun()的区别

fun() 是函数的调用，返回函数的调用结果

fun 是函数的名称，对应的是函数的整体结构

**函数声明和函数表达式创建函数的区别**

函数声明存在函数提升，调用可以在任意合法位置

函数表达式不存在函数提升，必须先创建，再调用

练习：使用函数表达式创建函数，计算任意两个数字之间所有整数的和，并返回结果。

### (2)匿名函数自调用

目的：创建局部作用域，防止污染全局。

|  |
| --- |
| (function(形参列表){  函数体中的变量和创建的函数只能在内部使用  })(实参列表); |

### (3)回调函数

将匿名函数以实参形式传递，这个匿名函数就称为“回调函数”；此时的形参就是函数名称，如果要调用匿名函数，只需要 形参名称()

|  |
| --- |
| function fn(a){  //a 就是传递的匿名函数的名称  a(); //执行传递的匿名函数函数体中的代码  }  fn( function(){ ... } ); |

练习：创建函数，传递两个参数，实参都是以匿名函数的形式传递，在匿名函数中分别返回数字；计算两个数字相加的和并打印出来。

## 2.全局函数

parseInt() 将数据转为整型

parseFloat() 将数据转为浮点型

encodeURI() 对一个URL中的中文进行编码

decodeURI() 对已经编码的URL进行解码

isNaN() 检测一个值是否为NaN 是->true不是->false

isFinite() 检测一个值是否为有限值 是->true 不是->false 1/0 Infinity 无穷 无限值

eval() 执行字符串表达式

练习：使用弹出提示框输入一组运算，使用eval来执行该运算，并打印出来

06\_eval.js 06\_eval.html

## 3.对象

是引用类型数据

对象：是一组属性和方法(功能)的集合

哪些是对象？

一个手机: 属性颜色，品牌，尺寸；方法有打电话，发短信，看视频，玩游戏;

一个汽车：属性颜色，品牌，类型；方法有代步，拉货，载客，撞人;

一个人: 属性性别，姓名，年龄；方法有说话，工作，写代码....

万物皆对象

### (1)JS中的对象

内置对象: JS提供的

宿主对象: 根据JS不同的执行环境来划分

自定义对象: 自己创建的对象

### (2)自定义对象

对象字面量

内置构造函数

自定义构造函数

## 4.使用对象字面量创建对象

使用大括号{ }创建空对象

属性名和属性值之间用冒号隔开

多组属性之间用逗号隔开

属性名中引号可加可不加，如果含有特殊字符必须加引号

|  |
| --- |
| { 属性名:属性值, ... } |

练习: 创建一个人对象，添加姓名、性别、年龄

练习：创建一个员工对象，包含编号、姓名、性别、生日、工资、所属部门编号

## 5.访问对象的属性

对象.属性名

对象['属性名']

可以添加之前不存在的属性；

如果属性不存在，属性值为undefined

练习：创建一个电脑对象，包含的属性有屏幕尺寸，品牌，颜色，内存大小；获取品牌的属性值，修改颜色的属性值，添加产地属性。

## 6.使用内置构造函数创建对象

new Object(); 创建一个空的对象

需要单独添加每一个属性

对象['属性名'] = '属性值'

对象.属性名='属性值'

练习：使用内置构造函数创建用户对象，包含编号，用户名，密码，邮箱，手机。

## 7.遍历对象的属性

访问所有的属性

|  |
| --- |
| for(var key in 对象){  key 对象中所有属性名  对象[key] 通过属性名找对应的属性值  } |

练习：创建对象，包含若干个成绩属性，遍历对象中属性获取总成绩。

提前声明sum用于保存总和。

## 8.对象中的方法

方法也称为成员方法，对应的是一个匿名函数

|  |
| --- |
| var person={  name:'tom',  say:function(){  this.name //this指代当前所在的对象  }  };  person.say(); //调用方法 |

day07

复习

匿名函数 function(){ }

创建函数

var fn=function(){ }

function fn(){ } fn其实也是一个变量

函数自调用 防止污染全局

(function(){ 函数作用域; })();

回调函数 将匿名函数以参数传递

function fn(a){ a() }

fn( function(){ 函数体; } )

全局函数

parseInt/parseFloat/encodeURI/decodeURI

isNaN/isFinite/eval

对象

内置对象/宿主对象/自定义对象

对象字面量/内置构造函数/自定义构造函数

{属性名:属性值,...}

new Object();

访问对象中的属性

对象.属性名 / 对象['属性名']

遍历对象属性

for(var key in 对象){ 对象[key] }

对象中的方法

{ name:'tom', say:function(){ this.name } }

对象.say()

## 1.检测对象中是否含有某个属性

对象.属性名===undefined true->不存在 false->存在

对象.hasOwnProperty('属性名'); true->存在 false->不存在

'属性名' in 对象 true->存在 false->不存在

数组

数组是由多个元素组成的集合，每个元素就是一个数据

## 2.创建数组

### (1)数组字面量

[元素1,元素2...]

练习：创建数组，包含多个商品的名称；创建数组，包含多个城市名称

### (2)访问数组中的元素

数组[下标] 下标是从0开始，第一个元素的下标就是0

### (3)内置构造函数

new Array(元素1,元素2...);

new Array(3) 初始化元素个数为3，也可以添加更多个元素

练习：创建数组，保存若干个课程名称；

练习：创建数组，初始化长度为5，添加篮球场上的五个位置

### (4)数组的长度

数组.length 获取数组中元素的个数

作用：在数组的末尾添加新的元素

数组[数组.length] = 值;

练习：创建一个空数组，使用数组长度添加若干个国家名称。

## 2.数组的分类

数组分为索引数组和关联数组

索引数组以0以上的整数作为下标

关联数组以字符串作为下标，只能单独的添加元素

## 3.遍历数组

### (1)for-in

|  |
| --- |
| for(var key in 数组){  key 要遍历的下标  数组[key] 下标对应的元素  }  既可以遍历关联数组，也可以遍历索引数组。 |

练习：创建数组，包含有多个学生的成绩，获取总成绩。

### (2)循环

|  |
| --- |
| for(var i=0;i<数组.length;i++){  i 代表下标 0 ~ 数组长度-1  数组[i] 代表下标对应的元素  }  只能遍历索引数组 |

练习：创建数组，包含有多个姓名，把tom全部改成"汤姆"

练习：创建数组，包含多个姓名，查询tom出现的次数

练习：创建数组，包含多个数字，获取这组数字的最大值

练习：创建函数getAvg，传递一个参数(一组工资)，返回工资的平均值

## 4.数组API(方法)

(Application Programming Interface)

API 应用程序编程接口，预定义好的一些方法或者函数

toString() 将数组中的元素按照逗号组合成字符串

join('-') 将数组中的元素按照指定字符组合成字符串，默认是逗号

concat(arr2,arr3) 拼接多个数组

slice(start, end) 截取数组中的元素，start开始的下标，end结束的下标，不包含end本身，如果是负数表示倒数；返回一个数组。

练习：创建数组a~g，每个字母是一个元素，分别截取cd，f，b，把截取的结果拼接成一个新的数组。

splice(start, count, value1,value2...) 删除数组中的元素，start开始的下标，count删除的数量，value删除后插入的元素； 返回删除的元素，原数组会发生变化。

练习：创建数组a~h，每个字母是一个元素，删除d、e，替换f为m，在下标为1的位置插入字母z

reverse() 翻转数组中的元素

sort() 对数组中的元素进行排序，默认按照Unicode码从小到大排序

|  |
| --- |
| 对数字排序  sort(function(a,b){  return a-b; //从小到大排序  // return b-a; //从大到小排序  }) |

day08

复习

检测对象中是否含有某个属性

对象.属性名 === undefined

对象.hasOwnProperty('属性名')

'属性名' in 对象

数组

数据的集合

创建数组

[ ] new Array();

访问数组元素

数组[下标]

数组的长度

数组.length

数组分类

关联数组/索引数组

数组的遍历

for(var key in 数组){ }

for(var i=0;i<数组.length;i++){ }

数组API

toString/join/concat/slice/splice/reverse/sort

## 1.数组API

push() 往数组的末尾添加元素，返回数组的长度

pop() 删除数组末尾的元素，返回删除的元素

unshift() 往数组的开头添加元素，返回数组的长度

shift() 删除数组开头的元素，返回删除的元素

## 2.二维数组

数组中的每个元素也是数组

|  |
| --- |
| var arr=[ [], [], [] .. ] |

访问二维数组中的元素 arr[下标][下标]

## 3.字符串

包装对象：目的是让原始类型数据像引用类型一样，具有属性和方法。

JS提供了3种包装对象：String Number Boolean

将任意数据转为字符串

new String() 返回对象，使用和字符没有区别

String() 返回字符串

### (1)转义字符 —— \

转换字符的意义

用法：在要转义的字符前加 \

\' 将引号转义成普通的字符

\n 将字符n转义成换行符

\t 将字符t转义成制表符(tab键效果)

...

练习：打印出现 welcome to chi\na

### (2)字符串API

length 获取字符串的长度

charAt() 获取下标对应的字符，也可以使用数组形式 字符串[下标]

练习：遍历字符串，打印出每一个字符（循环下标）。

charCodeAt() 获取某个字符的Unicode码

indexOf(value, start) 查找某个字符串的下标，value要查找的字符串，start开始的下标，默认是0，如果找不到返回-1

lastIndexOf(value) 查找某个字符串最后一次出现的下标，找不到返回-1

练习：声明变量保存字符串，检测该字符串中是否含有@，如果有打印“合法的邮箱“，否则打印“不合法邮箱”。

toUpperCase() 将英文字母转为大写

toLowerCase() 将英文字母转为小写

练习：声明变量保存4个英文字母，无限循环弹出提示框，输入验证码(不区分大小写)，如果输入正确结束循环。

06\_code.js 06\_code.html

slice(start, end) 截取字符串，start开始的下标，end结束的下标，不包含end本身；如果为负数表示倒数；end为空截取到最后。

练习：声明变量保存邮箱，分别截取出邮箱的用户名和域名 tom123456@163.com

substr(start, count) 截取字符串，start开始的下标，count截取的长度，如果count为空截取到最后，start为负数表示倒数

练习：声明变量保存身份证号，截取出年月日和性别；打印“1998年06月20日 性别女”

110236199806202589

substring(start,end) 截取字符串，start开始的下标，end结束的下标，如果end为空截取到最后；如果下标为负数，自动转成0。

对比slice和substring

slice中下标允许使用负数，substring会自动将负数转为0；slice中下标的顺序start小于end，substring中下标大小不分顺序

练习：将一个英文单词的首字母转大写，其余的字母转小写 heLLo -> Hello

split(sep) 将字符串按照指定字符分隔成数组，sep是指定的字符

练习：使用split分隔邮箱，获取用户名和域名

## 4.匹配模式(掌握)

作用：用于查找、替换字符串

match(value) 用于查找匹配的字符串，返回所有满足条件的元素，组成数组 /china/ig

i -> ignore 忽略大小写

g -> global 全局查找

search(value) 用于查找满足条件的第一个字符串的下标，如果找不到返回-1

replace(value1,value2) 查找并替换，value1要查找的字符串，value2要替换的字符串

## 5.Math对象

不需要使用new创建，可以直接使用

PI 取圆周率

abs() 取绝对值

floor() 向下取整

ceil() 向上取整

round() 四舍五入取整

max() 取一组数字的最大值

min() 取一组数字的最小值

pow() 求x的y次幂

random() 取随机数 >=0 <1

练习：随机产生0~9之间的一个整数。

课后任务

(1)复习今天内容，整理思维导图

(2)练习：

创建数组，包含有若干个元素，每次随机取出1个元素； 数组有10个元素 随机取0~9,作为下标

创建数组，包含所有的英文字母(大写，小写)和数字(0~9)，随机取4个元素(循环4次)，放入一个新数组 push。如何不重复？

(3)预习Date对象

day09

'hOw aRE yOu' -> 'How Are You '

获取每个单词，按空格将字符串分隔为数组

遍历数组，将每个单词首字母取出转大写，其它的字母取出转小写，拼接起来替换之前的字符串

## 1.Date对象

用于对日期时间进行存储和计算

### (1)创建Date对象

new Date('2019/4/18 10:30:40')

new Date(2019,3,18,10,30,40); //第二个参数月份范围0~11

new Date() 存储当前的系统时间

new Date(1500000000000) 存储的是距离计算机元年的毫秒数对应的日期时间 2017-07-14T02:40:00.000Z

### (2)获取Date对象中的日期时间

getFullYear/

getMonth范围0~11/

getDate/

getHours/getMinutes/getSeconds/getMilliseconds

getDay 范围0~6/

getTime 距离计算机元年毫秒数

练习：创建Date对象，保存当前系统的日期时间；根据对象打印 2019年04月19日 星期四

### (3)转为本地字符串

toLocaleDateString() // 年/月/日

toLocaleTimeString() // 时:分:秒

toLocaleString() // 年/月/日 时:分:秒

### (4)修改Date对象中的日期时间

setFullYear/setMonth/setDate/setHours/

setMinutes/setMilliseconds/setTime

setTime修改后，可能会影响其它的日期时间

练习：创建Date对象，保存员工的入职时间'2019/4/18'；3年后合同到期，计算到期时间；合同到期前一个月续签合同，如果是周末，提前到周五，计算续签时间。

入职时间：2019-4-18

到期时间：2022-4-18

续签时间：2022-3-18

## 2.Number对象

new Number() 将数据转为数值型，返回对象

Number() 将数据转为数值型，返回数值

Number.MAX\_VALUE 获取Number的最大值

Number.MIN\_VALUE 获取Number的最小值

toFixed(n) 保留小数点后n位

toString(n) 将数值转为字符串，n转换的进制

## 3.Boolean对象

new Boolean() 将数据转为布尔型，返回对象

Boolean() 将数据转为布尔型，返回布尔型

!!值 隐式将数据转为布尔型

toString() 将布尔型数据转为字符串

## 4.错误处理

Syntax Error: 语法错误，错误的使用了中文，缺少括号等; 出现后所有的代码都不执行

References Error: 引用错误，使用了未声明的变量，属于运行时的错误，影响后续代码的执行

TypeError: 类型错误，错误的使用了数据，例如把变量、数组当做函数使用，属于运行时的错误，影响后续代码的执行

RangeError: 范围错误，参数的使用超出了范围；属于运行时的错误，影响后续代码的执行

|  |
| --- |
| try{  尝试执行的代码，可能产生错误  }catch(err){  err 捕获到的错误  具体处理错误的内容  } |

练习：声明一个变量add未赋值；尝试调用add函数；如果执行错误，给add赋值一个匿名函数，然后再调用add

## 5.ES6

ECMAScript6

ECMAScript 2015 2016 2017

[http://es6.ruanyifeng.com](http://es6.ruanyifeng.com/) 《ES6入门》

### (1)块级作用域

使用let关键字声明的变量，只能在块级作用域下使用，不能被外部访问，不存在变量提升。

块级作用域：{ } for、while、do、if....

防止污染全局

### (2)箭头函数

是回调函数的另一种写法，和之前的匿名函数不完全一样

|  |
| --- |
| sort( (a,b)=>{  return a-b;  } ) |

如果箭头函数的函数体中只有一行代码，并且是return形式的，可以简化为 sort( (a,b)=>a-b )

练习：创建函数add，传递两个参数，每个参数都是回调函数，在回调函数中返回一个数字；在函数add中计算两个数字相加的和。

1.ES6

(1)函数中的参数

ES6允许为参数设置默认值，如果没有传递实参，自动调用形参的默认值

(2)模板字符串

|  |
| --- |
| ` 一组字符串，可以直接写JS ${js表达式} ` |

练习：创建一个对象，保存一个员工的对象(姓名，性别0/1，生日，工资)；使用模板字符串打印员工的信息，性别0显示女，1显示男

# 03.总监咨询日.李文华

李文华

电话/微信：15510318281

Application Programming Interface

一天知识点难度配比： 6:3:1 —— 思维导图

第一阶段：数据API

项目流程(1)

数据库和SQL语言(3)

编程语言入门——十步(3/5)

Node.js编写Web服务器(6)

第二阶段：静态页面显示数据

第三阶段：用户交互

第四阶段：移动App

第五阶段：高阶框架

面试题：如何自学一门编程语言？

1. 了解背景：产生、演变、现状、特点、应用领域....
2. 搭建开发/运行环境，输出Hello World
3. 变量和常量(内存)
4. 数据类型
5. 运算符
6. 逻辑结构
7. 通用小程序
8. 函数和对象
9. 官方提供的函数和对象
10. 第三方提供的函数和对象/小项目

项目实现的步骤：

UI美工 > 前端 > 后台 > 测试 > 部署

错误即宝贝！调试能力！

Error：You have an error in your sql syntax, refer the manual..... near ...

推荐的学习网站：

[www.csdn.net](http://www.csdn.net) 了解行业动态

[www.w3school.com.cn](http://www.w3school.com.cn) 适合入门

developer.mozilla.org/zh-CN 深入学习，搜MDN

# 03.node.js

nodejs基于谷歌V8引擎，运行在服务器端的语言，基于JS

对比JS和nodejs

(1)JS运行在客户端浏览器，存在多个浏览器，容易产生兼容性的问题；nodejs在服务器端只有一个环境，不存在兼容性。

(2)两者都有内置(ES)对象、自定义对象、宿主对象(根据执行环境的不同划分)

(3)JS用于网页中的交互效果，nodejs用于服务器的操作，例如web服务器创建，数据库操作，文件操作等

nodejs.org 英文官网

nodejs.cn 中文网

nodejs的特点

单线程逻辑处理

非阻塞的异步I/O处理

支持数万个并发连接

nodejs的应用场景

基于社交网络的大规模web应用

nodejs不适合CPU密集型的应用

递归、数据加密解密、数据挖掘和数据分析

nodejs的执行方式

脚本模式 node c:/xampp/.../1.js

交互模式

node 回车 进入交互模式

两次ctrl+c 或者 .exit 退出交互模式

## 3.全局对象

全局作用域下的变量就是全局对象下的属性，全局对象下的函数就是全局对象下的方法，可以使用全局对象来访问

nodejs: global

在交互模式下，声明的变量和创建的函数都属于全局下的，可以使用global来访问；

在脚本模式下，声明的变量和创建的函数都不属于全局下的，一个文件默认会创建一个独立的作用域，叫做文件（模块）作用域，防止污染全局。

js: window

在浏览器下，文件中声明的变量或者创建的函数都属于是全局作用域下的，会污染全局。

### (1)console对象——控制台

global.console.log() 打印日志

global.console.info() 打印消息

global.console.warn() 打印自定义的警告

global.console.error() 打印自定义的错误

global.console.time('自定义字符串') 开始计时

global.console.timeEnd('自定义字符串') 结束计时

自定义字符串前后保持一致

练习：计时查看for、while、dowhile循环10000的耗时

### (2)process对象——进程

process.arch 查看当前CPU的架构

process.platform 查看当前的操作系统

process.env 查看当前系统的环境变量

process.version 查看当前nodejs的版本号

process.pid 查看当前的进程编号

process.kill() 结束某个编号的进程

### (3)Buffer对象——缓冲区

一块用于临时存储数据的内存区域，可以存储文件数据、网络上传输的资源(视频、在线直播...)

创建Buffer

var buf=Buffer.alloc(5, 'abcde')

buffer数据 <Buffer 61 62 63 64 65>

将buffer数据转为字符串

buf.toString()

## 4.模块

模块就是一个封装好的功能体

在nodejs下模块分为自定义模块、核心模块(官方提供)、第三方模块

### (1)自定义模块

在nodejs下，任意一个js文件都是一块模块，文件中的代码默认被一个构造函数所包含

|  |
| --- |
| (function(exports,require,module,\_\_filename,\_\_dirname){  //程序员编写的代码  }) |
| \_\_filename 当前模块的完整路径和文件名称  \_\_dirname 当前模块的完整路径  require() 引入一个模块  module 指代当前的模块对象  module.exports 当前模块导出的对象，包含供其它模块使用的属性和方法  exports 等价于module.exports |

练习：在07\_2.js声明变量b，创建函数fn，导出变量和函数。

课后任务

(1)复习今天内容，整理思维导图

(2)练习

创建两个模块main.js（主模块），circle.js(功能模块)，在功能模块中创建两个函数，传递1个参数半径，计算圆的周长和面积，导出这两个函数；在主模块中引入功能模块，并调用两个方法

(3)预习核心模块下的querystring，url，fs

# day02

复习

全局对象：global/window

交互模式和脚本模式

console.log/info/warn/error/time/timeEnd

process.arch/platform/env/version/pid/kill

Buffer.alloc(5,'abcde') toString()

模块

(function(exports,require,module,\_\_filename,\_\_dirname){

程序员代码

})

module.exports === exports

## 1.模块分类

|  |  |  |
| --- | --- | --- |
|  | 以路径开头 | 不以路径开头 |
| 文件模块 | require('./circle.js')  常用于用户自定义的模块，如果后缀名是js，则可以省略 | require('url')  用于引入官方提供了核心模块 |
| 目录模块 | require('./02\_2')  在02\_2目录下寻找package.json中寻找main属性对应的文件，找不到会去引入index.js | require('04\_2')  自动到当前目录下的node\_modules中寻找目录模块04\_2，如果找不到会到上一级目录下寻找，直到顶级目录。常用于第三方模块 |

练习：创建模块03\_1.js，引入当前目录下的03\_2目录模块；在03\_2下创建test.js，导出一个函数fn(计算两个数字相加)，在03\_1.js调用

练习：在05目录下创建模块05\_1.js，引入不带路径模块05\_2,在05\_2下含有hello.js文件(打印一句话)

## 2.npm和包

包(package): 是一个目录模块，里边包含多个文件，其中有一个文件命名为package.json，是包说明文件，含有包的信息

npm: 包管理工具，安装nodejs的时候会附带安装，用于管理包，包括下载，上传，升级...

npm官网： www.npmjs.com

### (1)切换目录

cd 完整路径

如果要进入其它的盘符

d: 回车

进入指定目录下，按住shift键，在空白区域单击鼠标右键->在此处打开powershell

### (2)使用npm下载安装第三方包

npm install 包名称

### (3)生成package.json

npm init -y 自动生成一个package.json文件，后期使用npm 安装的包都会记录到这个文件中。  
 npm install 自动安装package.json文件中记录的包

## 3.全局函数

parseInt/parseFloat/encodeURI/decodeURI/isNaN

isFinite/eval

### (1)一次性定时器

|  |
| --- |
| 开启  var timer=setTimeout(回调函数, 间隔的时间);  当间隔的时间到了，执行回调函数，单位是毫秒  清除  clearTimeout(timer); |

### (2)周期性定时器

|  |
| --- |
| 开启  var timer=setInterval(回调函数,间隔的时间);  每隔一段时间，执行一次回调函数  清除  clearInterval(timer) |

练习：使用周期性定时器每隔3秒钟打印hello，打印三次以后，清除定时器

变量记录打印次数，如果次数为3，清除定时器。

### (3)立即执行定时器

|  |
| --- |
| 开启  var timer=setImmediate(回调函数)  回调函数会放入到队列中  当主线程程序执行完，才会执行队列中的内容  清除  clearImmediate(timer) |

|  |
| --- |
| process.nextTick(回调函数)  在主程序的末尾执行 |

## 4.querystring模块——查询字符串

属于核心模块，nodejs官方提供的，可以直接引入，不需要创建

www.codeboy.com/detail.html?lid=5&name=dell

浏览器向服务器发请求，传递数据的一种方式

parse() 将查询字符串格式化为对象

stringify() 将对象转换成查询字符串

练习：获取百度查询中查询字符串中的关键字

e=utf-8&tn=baidu&wd=电脑

https://www.codeboy.com:80/product/detail.html?lid=5&name=dell

## 5.url模块

parse() 将url格式化为对象

protocol 协议

hostname 服务器域名/IP地址

port 端口号

pathname 文件在服务器上的路径

query 查询字符串

format() 将对象转成url

query属性对应的是对象

练习：获取浏览器中传递数据

https://www.tmooc.cn:3000/web/1903.html?sid=10&course=javascript

获取url中的查询字符串

将查询字符串格式化为对象

09\_exercise.js

## 6.fs模块——文件系统模块

用于文件的操作，目录的创建、删除、读取；文件的创建、读取、写入、删除..

### (1)查看文件的状态

fs.stat(path, callback)

path 要查看的文件的路径

callback 回调函数，用于获取文件的状态

err 如果查看失败的错误信息

stats 文件的状态信息

isDirectory() 是否为目录

isFile() 是否为文件

课后任务

(1)复习今天内容，整理思维导图

(2)练习

创建模块01\_1.js，引入02\_2不带路径的目录，在02\_2中创建文件file.js，在file.js中，导出一个查询字符串，在01\_1.js中获取该字符串，并获取其中的数据。

(3)预习文件系统模块中的mkdir，rmdir，readdir，writeFile，appendFile，readFile

预习http协议

复习

|  |  |  |
| --- | --- | --- |
|  | 以路径开头 | 不以路径开头 |
| 文件模块 | require('./circle.js') | require('url') |
| 目录模块 | require('./02') | require('mysql') |

npm和包

npm install 模块名称

## 全局函数

setTimeout/clearTimeout

setInterval/clearInterval

setImmediate/clearImmediate

process.nextTick

querystring

parse/stringify

url

parse/format

fs

stat('05', function(err,stats){

isFile/isDirectory

})

## 1.同步和异步

同步：方法调用后，必须等待方法调用结束，才能继续后边的任务；通过返回值获取结果

异步：方法调用后，无需等待结束，直接去执行后续的代码，整个过程，不会阻碍后边的任务；通过回调函数获取结果

## 2.fs模块——文件系统

(1)查看文件状态

fs.stat(path,callback) / statSync(path)

(2)创建目录

fs.mkdir(path,callback) / mkdirSync(path)

(3)移除目录

fs.rmdir(path,callback) / rmdirSync(path)

(4)读取目录

fs.readdir(path,callback) / readdirSync(path)

callback

err 可能产生的错误信息

files 读取的结果，格式为数组

(5)清空写入

fs.writeFile(path, data, callback) / writeFileSync()

path 要写入的文件的路径

data 要写入的数据

callback 回调函数

如果文件不存在会创建文件，如果文件已经存在，会清空文件中的内容，然后写入

(6)追加写入

fs.appendFile(path,data,callback)/appendFileSync()

如果文件存在，在文件的末尾写入数据

练习：创建数组，包含多个姓名，遍历数组获取每个元素，把每个元素使用同步方法写入到文件data.txt

(7)读取文件

fs.readFile(path, callback)/readFileSync()

callback

err 可能产生的错误信息

data 读取的文件中的数据，格式为buffer

(8)删除文件

fs.unlink(path, callback) / unlinkSync()

(9)判断文件(目录)是否存在

fs.existsSync(path)

存在->true 不存在->false

练习：判断文件2.txt是否存在，如果存在删除。

判断目录mydir2是否存在，如果不存在创建

## 2.http协议

是浏览器和web服务器之间的通信协议

### (1)通用头信息(General)

Request URL: 请求的URL，对应浏览器地址栏的内容，要向服务器获取哪些内容

Request Method: 请求的方法，GET/POST...，获取内容的方式

Status Code: 响应的状态码,可以查看到响应的结果

1\*\*: 正在请求，没有结束

2\*\*: 成功的响应

3\*\*: 响应的重定向，跳转到另一个URL;通常结合响应头信息中Location一起使用

4\*\*: 客户端错误

5\*\*: 服务器错误

Remote Address: 请求的服务器的IP地址和端口号

### (2)响应头信息(ResponseHeaders)

Connection: 连接的方式，keep-alive 持续连接

Content-Type:响应的文件类型

Content-Length: 响应的文件长度

Transfer-Encoding: 传输方式 chuncked 分段传输

Content-Encoding: 压缩模式 gzip

Location: 当响应重定向的时候，跳转的URL

### (3)请求头信息(RequestHeaders)

Accept: 客户端接受的文件类型有哪些

Accept-Encoding: 客户端接受的压缩形式

User-Agent: 客户端发请求使用的浏览器

(4)请求主体

可有可无，客户端向服务器端传递的数据

## 3.http模块

既可以模拟浏览器向web服务器发请求，也可以创建web服务器

### (1)模拟浏览器

http.get( url, callback )

get 请求的方法

callback 回调函数，用于获取服务器端的响应

res 响应的对象

statusCode 响应的状态码

res.on('data', function(buf){ })

通过事件获取服务器端的响应，当有数据传输的时候自动触发；通过回调函数来接收数据；buf就是传输的数据，格式为buffer

练习：使用http模块下的get方法向

http://www.tmooc.cn发送请求，并获取响应的数据。

### (2)创建web服务器

var server=http.createServer() 创建web服务器

server.listen(8080) 设置web服务器端口号，监听端口变化

server.on('request', function(req,res){ })

接收浏览器的请求，是一个事件，当有请求发生自动触发

req 请求的对象

url 请求的URL，显示端口后的部分

method 请求的方法

res 响应的对象

writeHead(code, obj) 设置响应的状态码和头信息

code状态 obj是一个头信息的对象

write() 设置响应的内容

end() 结束响应，发送响应的内容到浏览器

练习：当请求的url为/login，响应一句话' login ',当请求的url为/study，跳转到http://www.tmooc.cn

使用if判断请求的url的值

'/login' write()

'/study' writeHead()

课后任务

(1)复习今天内容，整理思维导图

(2)练习

使用http模块创建web服务器，设置端口8080；接收浏览器的请求

/login 响应 this is login page

/member 响应 welcome to home

/ 重定向 /login

以上都没有 响应内容 404 not found

(3)预习express

复习

fs模块

fs.stat/statSync/mkdir/rmdir/readdir/writeFile

appendFile/readFile/unlink/existsSync

http协议

通用头信息 请求方法 请求URL 响应状态码

响应头信息

请求头信息

请求主体

http模块

模拟浏览器向服务器发请求

http.get(url,function(res){

res.statusCode

res.on('data',function(buf){ })

} )

创建web服务器

var server=http.createServer();

server.listen(8080);

server.on('request', function(req,res){

req.method/url

res.writeHead(200,{ })

res.write()

res.end()

})

## 1.express框架

基于nodejs的，快速、开发、极简的web开发框架

www.expressjs.com.cn

安装 npm install express

|  |
| --- |
| const express=require('express');//引入express  var server=express(); //创建web服务器  server.listen(8080); //设置端口 |

### (1)路由

浏览器向web服务器发送请求，web服务器根据请求的方法和请求的URL来做出响应。

三要素：请求方法、请求的URL、响应（回调函数）

**响应的对象(res)**

send() 响应内容，只能响应一次；

sendFile() 响应文件，必须使用绝对路径(\_\_dirname)

redirect() 响应的重定向

**请求的对象(req)**

req.method 请求的方法

req.url 请求的URL

req.headers 请求的头信息

req.query 获取请求时查询字符串传递的数据，并格式化为对象

练习：创建文件03\_express.js，使用express创建web服务器，添加以下路由

get /index '这是首页'

get /login 响应文件login.html

post /reg '欢迎注册'

get / 跳转到 /index

练习：创建文件04\_post.js，使用express创建web服务器，添加路由

get /reg 响应文件 reg.html

### 2.post和get传递数据

post请求是通过表单提交(暂时)传递数据，服务器端是通过事件的形式获取数据(后期会有简单的方法)

|  |
| --- |
| req.on('data', function(buf){  buf就是获取的数据，格式为buffer，转为字符串后格式为查询字符串，需要借助查询字符串模块格式化为对象  }) |

|  |
| --- |
| **对比get和post两种请求方法**：  get请求以查询字符串形式传递数据，可能被浏览器所缓冲，速度相对快，传递的数据不超过4K；  post传递的数据以请求为主体 ，不被浏览器所缓冲，速度相对慢，可以传递文件 |

get请求以查询字符串形式传递数据，服务器端使用req.query获取数据，结果是对象。

查询字符串传递数据容易被浏览器所缓存，而post传递数据不会出现在地址栏。

|  |  |  |  |
| --- | --- | --- | --- |
| 传递方式 | 格式 | 获取 | 地址栏查询 |
| get传递 | 查询字符串 | req.query 格式为对象 | http://127.0.0.1:8080/user/select\_id**?uid** =1 |
| post传递 | 流(不可见) | 通过事件获取  req.on('data', (chunk)=>{  //chunk是分段的数据，格式为buffer，需要转字符串，转完后为查询字符串，需要解析为对象  }) |  |
| 路由传参 | 通过URL | req.params 格式为对象 |  |

练习：添加路由，请求方法get，请求的url为/login，响应login.html，在html文件中创建登录的表单

### 3.使用路由传递数据——路由传参

设置路由中接收的名称

|  |
| --- |
| server.get('/detail/:lid', function(req,res){  req.params //获取路由传递的数据，格式为对象  }); |

浏览器传递

http://127.0.0.1:8080/detail/5

说明：5就是传递的数据，使用lid来接收

练习：创建购物车的路由，请求的方法get，请求的URL:/shopping，传递商品的名称(pname)和价格(price)

商品模块

商品列表/product/list 删除/product/delete 修改/update

用户模块

用户列表/user/list 删除/user/delete 修改/update

### 4.路由器

路由的使用过程中，不同的模块可能出现相同的URL，把同一个模块下的所有路由放到一个容器，这个容器就是路由器。路由器最终要引入到web服务器下才能使用。

创建路由器模块(自定义模块)

|  |
| --- |
| const express=require('express');  var router=express.Router(); //创建路由器对象  router.get('/list', function (req,res){ });//添加路由  module.exports=router; //导出路由器对象 |

在web服务器下使用路由器

|  |
| --- |
| const productRouter=require('./product.js');  server.use('/product', productRouter);  //把路由器挂载到/product下，  访问形式 /product/list |

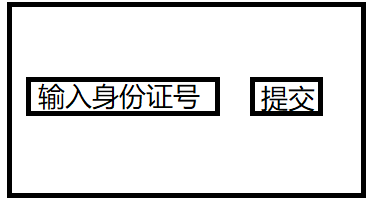
练习：创建用户模块路由器(user.js)，添加路由用户列表(/list, get)，在web服务器下引入并使用，挂载到/user下

课后任务

(1)复习今天内容，整理思维导图

(2)课后练习

创建web服务器，添加路由(get /page)，响应一个html文件；输入身份证号，点击提交(post, /mypage)，响应出生的年月日以及性别。



(3)复习mysql中sql语句

day05

复习

## 1.中间件

中间件理解是一个过滤器，作用是为主要的业务逻辑服务

分为应用级中间件、路由级中间件、内置中间件、第三方中间件、错误处理中间件

### (1)应用级(自定义)中间件

每个中间件都是一个函数

server.use( path, function(req,res,next){ } ) 过滤路由中url为path的路由。

server.use(function(req,res,next){ }) 过滤所有的路由

练习：给url为/reg的路由添加前置的中间件，验证用户名是否可用。

练习：创建路由(get , /view)响应当前的浏览次数；每次浏览次数加1。

在中间件的外部声明变量初始化为0，在中间件中完成加1,在路由中把该变量响应到浏览器。

### (2)路由级中间件

路由器的使用

server.use(path, 路由器);

### (3)内置中间件

在express4下只保留了一个内置的中间件

express.static('目录')

把静态资源托管到指定的目录，如果浏览器请求静态资源，自动到该目录下查找。

静态资源：html、css、浏览器js、图像....

### (4)第三方中间件

npm install 中间件名称

使用中间件

|  |
| --- |
| server.use( bodyParser.urlencoded({  extended:false  }) )  urlencoded: 将post请求的数据格式化为对象  extended: 不是用第三方qs模块，而是使用核心模块querystrinig将查询字符串格式化为对象 |

在路由中获取post请求数据

req.body 格式为对象

练习：创建04\_post.js，创建web服务器，托管静态资源到public下，在public下，创建login.html，使用浏览器请求该文件，点击按钮，向服务器发请求。

## 2.mysql

连接 mysql.exe -h127.0.0.1 -P3306 -uroot -p

mysql -uroot

增 INSERT INTO emp VALUES(NULL, 'tom'...)

删 DELETE FROM emp WHERE uid=5;

改 UPDATE emp SET uname='jerry',salary=8000 WHERE uid=5;

查 SELECT \* FROM emp;

### (1)连接mysql

var connection=mysql.createConnection(); 创建连接对象，传递连接数据库需要的服务器、端口、用户名、密码、要使用的数据库

connection.connect() 建立连接

connection.query(sql, callback) sql是要执行的SQL语句，callback回调函数，用于获取SQL语句的结果

connection.end() 执行完所有的SQL语句，关闭连接

### (2)连接池

var pool=mysql.createPool(); 创建连接池对象，传递连接需要的服务器地址、端口、用户名、密码、要使用数据库、设置连接池的大小 connectionLimit，默认是15

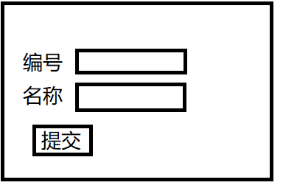
pool.query(sql, callback) 执行SQL语句，sql要执行的SQL语句，callback回调函数，用于获取结果

课后任务

(1)复习今天内容，整理思维导图

(2)练习

创建web服务器，托管静态资源到public下，在该目录下创建add.html，点击提交按钮，向服务器发请求(get /add)，服务器获取数据，并将数据插入到tedu下的dept，如果插入成功，响应“部门添加成功”



day06

复习

分页查询

页码

每页数据量 2

start=(页码-1)\*数据量

select \* from xz\_user limit 0,2;

select \* from xz\_user limit 2,2;

课后任务

完成商品模块 xz\_laptop

添加、修改、删除、检索、列表

创建商品路由器 product.js

# 04.html

day01

李然 QQ1535212067

## 一.课程安排

1.Html5 Basic 2天

|  |
| --- |
| 搭建网页结构 |

2.Ajax异步数据交换(3天)

|  |
| --- |
| 异步完成前后台数据的交互 |

3.项目1（2天）

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

4.css3 （4+2天）

|  |
| --- |
| 美化页面 |

5.Bootstrap框架 （4天）

|  |
| --- |
| 响应式的框架，css的框架  简化开发 |

6.项目2 （3天）

|  |
| --- |
| boot完成商城的商品部分所有页面 |

7.这20天课程遇到的问题

|  |
| --- |
| 1.知识点细碎，没条理  2.单词量大，关键词比较长，初期没有提示  3.键速 200 |

## 二.web基础知识

|  |  |
| --- | --- |
| HTML:泛指前端网页技术  2014年9，html4.0升级html5.，简称h5  Html5 大前端技术   |  | | --- | | html4.01在1999.12月发布  xhtml1.0 在2000.1月发布，语法更严谨  <img/> <img> | |

1.web与internet的关系

|  |
| --- |
| internet 全球性计算机互联网络  俗称 互联网，因特网，交换网，交际网  www服务world wide web 万维网  ftp:文件的上传下载的服务  Email:电子邮件的服务  telnet：远程登陆的服务  BBS：电子公告板，俗称论坛  以上都是运行在internet应用程序 |

2. internet上的应用程序都会分成两类

|  |
| --- |
| 1. C/S client 客户端 server服务器  王者荣耀 QQ.exe |
| 2. B/S browser浏览器 server服务器  所有的网站 网页版微信 |
| 3.C/S结构和B/S结构的区别  C/S需要升级 B/S不需要升级 |

## 3.web运行原理

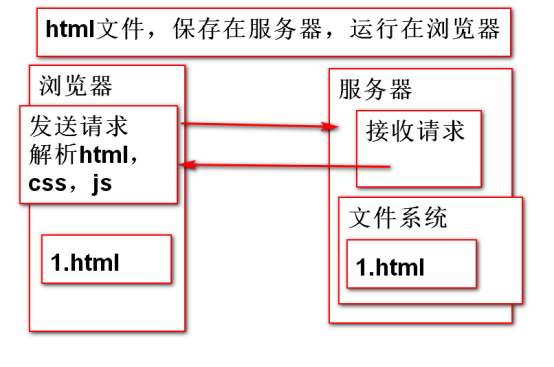
|  |
| --- |
| web：运行在internet上的一种B/S结构的应用程序  俗称网站  internet:为web运行提供了网络环境  web运行的原理：基于浏览器和服务器，以及通信协议来实现数据的传输和展示  通信协议：规范了数据是如何打包和传递的  网页的通信协议 http:// https:// |
| 服务器  1.功能  存储数据  接收用户的请求，并且给出响应  提供程序的运行环境  具备一定的安全功能  2.服务器产品  Apache--php  TOMCAT--java  IIS .net  3.服务器端技术  nodejs  java  php  .net  python |
| 3.浏览器  1.功能  代表用户发送请求  作为html，css和js的解析器  以图形化的界面展示给用户看  2.浏览器产品  safari  chrome  firefox  opera  ie  3.浏览器端的技术  html5 css3 javascript 11:04~~11:19 |

三.html快速入门

## 1.什么是html

|  |
| --- |
| HyperTexts Markup Language  超文本标记语言  标记：超文本组成形式，<关键字>，具体独特的功能 |

## 2.html的特点



|  |
| --- |
| 1.以.html或者.htm为后缀  2.由浏览器解析执行  3.用<>来定义标记  4.可以嵌套js脚本 |

## 3.html语法

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1.标记   |  |  |  | | --- | --- | --- | | 标记，元素，标签，节点  <关键字>  标签具备一些功能  学习html第一步，背关键字，熟悉对应的功能  标记分为两类   |  | | --- | | 1.双标记（封闭类型标记）  <关键字></关键字>  双标记成对出现，有开有关  ex:<a></a> | | 2.单标记(非封闭类型标记，空标记)  <关键字> 或者 <关键字/>  ex：<input/> <br/> <img/> |   练习：  新建一个text文档，改名为first.html  写一对html标记  一对head标记  一对title标记  一对body标记 |   2.标签的嵌套   |  | | --- | | 在一对标签内部，出现另一对或者另一个标签，组合形成功能的叠加  学习html第二步，背嵌套顺序  语法：缩进，嵌套顺序  <标签1>  <标签2>  <标签3></标签3>  </标签2>  </标签1> |   3.属性和值   |  | | --- | | 通过属性和值，对标签进行修饰  <标签 属性1="值1" 属性2="值2" ...></标签> | | 1.标准属性，通用属性。所有标签都有的属性  id:定义元素在页面中唯一的标识  title：鼠标悬停在元素生显示的文本  style：css中，定义内联样式  class：css中，引用类选择器时使用 | | 2.专有属性，只针对一个标签起作用的属性 |   学习html第三步，背专有属性，及效果  4.注释   |  | | --- | | 不会被浏览器解析运行的部分  一般编写代码说明  <!-- -->  注释不能嵌套注释  注释不能写在标签内部 | |
| 学习html就三步  1.学习固定标签关键字，及功能  2.学习固定的嵌套关系。  3.学习固定的属性及其值 |

## 四.HTML的文档结构

### 1.html文档的构成

|  |
| --- |
| 1.文档类型声明  <!doctype html>  告诉浏览器，解析运行本篇代码，使用h5的规则解析 |
| 2.网页的结构  <html></html>  表示网页的开始和结束。一个html文件中只能有一对html标签  <!doctype html>  <html>  <head></head> 网页头部，定义网页的信息  <body></body> 网页主体，展示内容  </html> |
| 3.head标签，定义网页的信息   |  | | --- | | 1.<meta> 元数据标签  <meta charset="utf-8">  <meta name="description" content="描述内容">  <meta name="Keywords" content="关键字"> | | 2.head中其它标签  <title></title> 网页标题  <script></script> 编写js或者引用js文件  <style></style> css中定义内部样式  <link>引入外部css样式 |   4.body元素   |  | | --- | | 指定网页的主体  <body></body>  属性：bgcolor body的背景颜色，取值颜色合法值  text body的文本颜色，取值颜色的合法值 |   练习：01\_ex.html  编写完整的html页面结构 |

## 五.文本标记(重点\*\*\*)

1.标题元素

|  |
| --- |
| 在页面中以醒目的方式显示文本  <h1></h1>  ....  <h6></h6>  特点：1.字体大小有变化 h1最大，h6最小  2.字体加粗  3.单独成行，上下有垂直间距 属性：align 设置标记内容水平对齐方式  取值：left/center/right |

2.段落标记

|  |
| --- |
| <p></p> 以突出的形式表示一段文字  特点：1单独成行  2.上下有垂直间距  属性：align:left/cener/right |

3.换行标记

|  |
| --- |
| 空格折叠现象：在html中，文本中不管有多少个空格和回车，都会被浏览器解析成一个空格显示  <br> 或者 <br/> |

4.特殊字符(实体)

|  |
| --- |
| &nbsp; 空格  &lt; <  &gt; >  &times； X  &reg; 商标注册 ®  &copy；版权 ©  &yen; |

5.分割线，水平线

|  |
| --- |
| <hr> 或者 <hr/>  属性  size="5px" 取值px为单位的数字，水平线的粗细  width="50%" 取值px或者%,水平线的宽度  align="left" left/center/right  color="blue" 取值为合法颜色值 |

6.预格式化

|  |
| --- |
| <pre></pre>  保留html代码中的回车和空格效果 |

7.文本样式标签

|  |
| --- |
| <em></em> <i></i> 斜体  <strong></strong> <b></b> 加粗  <del></del> <s></s> 删除线  <u></u> 下划线  <sup></sup> 上标  <sub></sub> 下标 |

8.分区元素--不写样式的分区元素，是看不见的。

|  |
| --- |
| 1.块分区<div></div>  用于页面布局 |
| 2.行分区<span></span>  为文本添加样式的时候用span |

9.元素的显示方式

|  |
| --- |
| 1.块级元素  元素独立成行，在页面中从上往下排列  ex:h1~h6 p div pre |
| 2.行内元素(行级元素)  多个行内元素在一行中显示，从左往右排列  ex:span i em strong b u sub sup del s |
| 3.行内块  表现方式是行内元素(多个行内块可以共用一行)  但是具有块级元素的其它特征(宽高，边距)还没学  ex:input |
| 4.table |

## 六.图形和链接

### 1.图形图像

|  |
| --- |
| <img/> 或者 <img>  属性：src 引用图片资源 url  \*.html和图片是兄弟关系，直接在src中写图片名称 |

练习



EditPlus快捷键设置总结

|  |
| --- |
| 加载1.key。1.key eclipse的开发快捷键  1.设置代码补全和lorem的补全  在zc下，把第二个和第三个勾勾上  2.设置tab空两个格    把默认的4变成2  3.导入1.key    选择1.key就可以使用了  4.常用快捷键  ctrl+d 删除当前行  alt+↓/↑ 移动当前行下移/上移  ctrl+alt+↓ 复制当前行  5.打开浏览器运行html的快捷方式  ctrl+b 打开浏览器    6.添加注释快捷方式  ctrl+/ |

作业:

1.完成nodejs6个接口的编写

2. web第二阶段练习效果图示\day01\_html\_01作业



day02

## 六.图像和链接

### 1.图像的使用

|  |
| --- |
| <img>  属性  src="url/图片资源路径"  ex：<img src="图片资源路径"> |

### 2.URL

|  |
| --- |
| 1.URL是什么  Uniform Resource Locator 统一资源定位器  就是路径 |
| 2.绝对路径  完整的路径  协议+主机名称+文件目录结构+文件具体的名称  使用场合：使用网络资源的时候，使用绝对路径  使用网络资源图片  优点：节省服务器本地存储空间  缺点：资源不稳定  绝对路径使用本地的资源，是从最高盘符开始查找  c:\img\08.png  但是本服务器资源，在项目中，不能用绝对路径 |
| 3.相对路径    使用本服务器资源，用相对路径  1.同级目录资源  直接使用资源名称 src="08.png"  2.兄弟文件夹的子元素  直接使用兄弟文件夹的名称，在用/调用资源名称  src="img/08.png"  3.父级目录中资源  先使用../返回上一级，在引用资源名称  src="../08.png" |

### 3.属性

|  |
| --- |
| src：source 源，资源路径  alt： 图片错误时，显示的文本  title：鼠标悬停时显示的文本  width：设置图片的宽度  height：设置图片的高度  如果设置宽高比，不符合图片原始宽高比。  图片会产生失真效果  解决图片失真效果：  一般情况下，width和height只设置一个，让另外一个自动适应 |

练习

|  |
| --- |
| 01\_ex.html  页面中引入6张图片  1.图片与页面是兄弟关系  2.图片是页面的兄弟的儿子  3.图片是页面的兄弟的儿子的儿子  4.图片是页面父亲的兄弟  5.图片是页面的父亲的兄弟的儿子的儿子  6.图片是页面父亲的父亲的兄弟 |

### 4.链接

|  |
| --- |
| 1.语法  <a href=""></a>  属性：href 链接路径  target 指定打开链接的方式  取值 \_self 默认值，在当前页面打开新网页  \_blank 在新的页面打开网页 |
| 2.a标签的其它表现形式  1.新建邮件，配合windows的邮件软件使用  <a href="mailto:cheng@t.com">发邮件</a>  2.执行js  <a href="javascript:show()">执行js</a>  3.下载  <a href="1.zip">下载</a>  4.返回页面顶部  <a href="#">返回页面顶部</a> |
| 3.锚点  什么是锚点  锚点是网页中的一个记号  可以通过超链接的方式，链接到锚点，让页面跳转到锚点位置显示  使用锚点   |  | | --- | | 1.定义锚点  第一种方式：任何标签的id中写锚点名称  <Any id="锚点名称"> </ Any >  第二种方式：a标签的name属性中写锚点名称  <a name="锚点名称"></a> | | 2.链接到锚点  <a href="#锚点名称"></a> | | 3.链接到其它页面的锚点  <a href="其它页面的url#锚点名称"></a>  <a href="02\_a.html#hl">葫芦娃</a> | |

## 七.表格

### 1.语法

|  |
| --- |
| 早期table用于布局  由于效率低下，后来被淘汰，使用div+css的布局  表格现在就一个单纯的作用---显示数据  <table> 表格  <tr> 行 table row  <td></td> 列 table data  ....  </tr>  ....  </table> |
| 属性   |  | | --- | | table的属性  width="300px" 表格的宽度  height="300px" 表格的高度  border="1px" 表格的边框  align="left/center/right" 表格的水平对齐方式  bordercolor="red" 边框颜色  bgcolor="yellow" 背景颜色  cellpadding 单元格内边距，内容到边框的距离  cellspacing 单元格的外边距，边框与边框的距离 |   练习：   |  | | --- | | 03\_ex.html  3行3列的表格，宽高400px\*400px  背景颜色pink，边框为2px 表格居中显示  表格内边距5px 外边距10px | | bgcolor="yellow" 背景颜色  align="right" 表格这一行的内容水平对齐方式  valign表格这一行的内容垂直对齐方式  取值 top/middle/bottom | | td的属性  width="200px"  height="200px"  align="right" 内容的水平对齐方式  valign="bottom" 内容的垂直对齐方式 只能在图片和td中使用  bgcolor="red"  colspan 跨列  rowspan 跨行 | |

### 2. 不规则的表格

|  |
| --- |
| 1.跨列：colspan  从指定单元格位置处开始，横向向右合并n个单元格(n包括自己)，被合并的单元格要删除 |
| 2.跨行 rowspan  从指定单元格位置处开始，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 |

3.可选标记

|  |
| --- |
| 1.表格的标题  <caption></caption>  如果要设置标题，caption必须放在<table>之后  2.行/列的标题  <th></th>,td的属性，th都可以用  th的特点：文字加粗，水平居中 |

4.表格的复杂应用

|  |
| --- |
| 我们制作的表格，浏览器解析的时候会自动添加上<tbody></tbody>标签  行分组：可以将连续的几行，划分到同一个组中，进行统一的管理。  1.表头<thead></thead>  2.表主体<tbody></tbody>  3.表脚<tfoot></tfoot>  <table>  <thead>  <tr><th></th></tr>  <tr><th></th></tr>  </thead>  <tbody>  <tr>...</tr>  <tr>...</tr>  <tr>...</tr>  </tbody>  <tfoot>  <tr></tr>  </tfoot>  </table> |
| 2.表格的嵌套  表格中所有的嵌套，都只能放在td中完成 |

## 八.列表（重点\*\*\*\*\*\*\*）

1.列表的作用

|  |
| --- |
| 按照从上到下，或者从左到右来显示所有的数据  并且可以在数据前增加标识 |

2.列表的组成

|  |
| --- |
| 列表都是由列表类型和列表项来组成的  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 |

3.有序列表

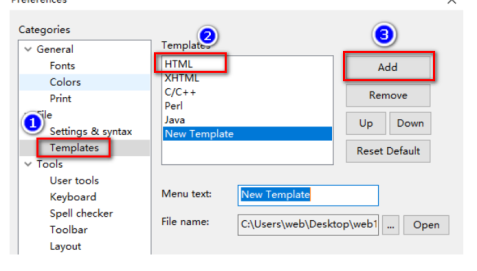
|  |
| --- |
| <ol>  <li></li>  <li></li>  ...  </ol>  属性：type 指定标识项的类型  取值 1 默认值，数字  a/A 字母  i/I 罗马数字  start 指定起始编号 取值：无单位数字 |

4.无序列表---经常用于页面布局

多个li的样式大体一致的时候，使用ul>li布局

|  |
| --- |
| <ul>  <li></li>  <li></li>  </ul>  属性：type 设置列表标识类型  取值 1.disc 默认值 实心圆  2.circle 空心圆  3.square 实心方块  4.none 不显示 |

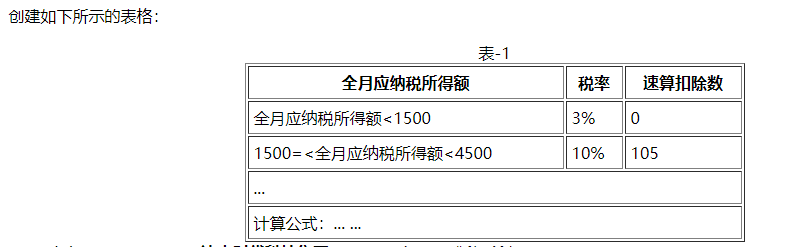
设置html模板



作业

1.nodejs 6个接口

2.表格练习





day03

八.列表

1.有序

2.无序

## 3.列表的嵌套

|  |
| --- |
| 1.列表中嵌套其他元素 |
| 2.列表中嵌套列表 |
| 原则，嵌套只能写在li中。不管是ol还是ul，直接子元素只能是li. |

## 4.定义列表

|  |
| --- |
| h5新出的，一个名词，以及对这个名词的解释和定义  <dl> 定义列表  <dt> </dt> 要解释说明的名词  <dd></dd> 对名词解释的内容  </dl> |

## 九.结构标签

|  |
| --- |
| h5新出的标签，专门用于网页布局  为了让代码有可读性，h5推出一堆与div功能一模一样的标签，专门用作网页布局  1.<header></header>  定义网页的头部，或者某个区域的顶部  2. <section></section>  定义网页的主体内容  3. <footer></footer>  定义网页的脚部，或者某个区域的底部  4. <nav></nav>  定义网页的导航栏  5.<aside></aside>  定义网页的侧边栏  6.<article></article>  定义与文字相关的内容  比如，论坛，回帖，用户评论.... |

## 十.表单，\*\*\*\*重点\*\*\*\*\*

1.作用

|  |
| --- |
| 提供可视化的输入控件  收集用户输入的信息，并提交请求给服务器  总结：form自带提交请求的功能  ajax提交不需要form的支持  在同一个功能中，使用form就不用ajax,使用ajax就不用form |

2.form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 nodejs+mysql |

3.form表单详解

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| <form></form>  属性：   |  | | --- | | 1.action="" 定义表单提交时，发送请求的动作(url)  如果不写值，默认提交给本页 | | 2.method="" 定义表单的提交方法  取值   |  | | --- | | 1.get 默认值  特点：明文提交(参数在地址栏中显示)，不安全  提交的数据有大小限制，最大为2kb  使用场合：向服务器要数据的时候，使用get | | 2.post  特点：隐式提交，参数不在地址栏中显示  提交数据没有大小限制  使用场合：要传递数据给服务器的时候，使用post | | 3.其它的提交方法  delete put option....... | | | 3.enctype="" 指定表单数据的编码方式  设置允许将什么类型的属性传递给服务器  取值：  1.application/x-www-form-urlencoded  默认值，允许将任意的字符提交给服务器  (不能提交文件)  2.text/plain 允许提交普通字符给服务器，不能包含特殊符号  3.multipart/form-data 允许提交文件给服务器 |   <form action="url" method="get"  enctype="application/x-www-form-urlencoded">  </form>  收集的数据，发送给url这个接口，使用get提交  要求提交的数据是任意字符，不是文件 |

## 4.表单控件

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 在form标签中，能够与用户交互的可视化控件 1.表单控件的分类  |  | | --- | | input元素 基础9种  textarea 多行文本域  select+option下拉选择器  其它元素  h5新出的input元素 10种 |  2.input元素  |  | | --- | | <input type="">  在页面中，input提供了各式各样的输入控件  文本框，密码框，单选框，多选等....  共同的属性  1.type="" 指定input元素的类型  2.name="" 为控件定义名称，此名称是给服务器用的，如果想往服务器传递参数，此名称必须写  3.value 是控件的值，真正传给服务器的值  例外,在按钮控件中,value是设置按钮上显示的文字  4.disabled 禁用，不能修改值，也不能提交  取值：disabled="disabled"  disabled="true" --js使用  disabled  这种属性，称之为无值属性 |  3.input详解9种  |  | | --- | | 1.文本框和密码框  文本框 type="text" input默认是text  密码框 type="password"  属性：  maxlength 设置输入的最大值  readonly 无值属性，不能修改，允许提交  placeholder 占位提示符 | | 2.按钮  提交按钮 type="submit"  将表单中的数据收集整理，发送给服务器  普通按钮 type="button"  没有功能，配合事件，调用js代码  重置按钮 type="reset"  将当前表单内容恢复到初始化状态  按钮中value，设置按钮上的文本，此value不提交 | | 关于按钮的题外话  h5给我们推出了新的按钮标签  <button></button> 自带submit功能 | | 3.单选和复选按钮  单选按钮 type="radio"  多选按钮 type="checkbox"  属性name，除了定义控件名称之外，还起到分组的作用，分了组才会有单选效果  必须有value属性，不然传递的参数永远是on  checked 无值属性 设置默认被选中的项  单选框样式  content: "\2714";  content表示在方框中显示的内容，"\2714"、"\2713"都表示对号，只是显示的瘦弱程度不同 | | 4.隐藏域  type="hidden"  想把数据提交给服务器，但是又不想用户看到  这种数据放到隐藏域中  用户看不见，但是可以提交 | | 5.文件选择框，上传文件  type="file"  前提：提交方法，使用post  enctype属性，设置为multipart/form-data  属性multiple，可以选择多个文件上传 |  4 textarea多行文本域  |  | | --- | | <textarea></textarea>  允许录入多行文字  cols 设置输入的列数  rows 设置输入的行数  根据计算硬件不同，不准确 |  5.下拉选  |  | | --- | | <select>  <option></option>  <option></option>  ....  </select>  注意：  1.如果option没有value，那么select的value是option的内容  2.如果option有value,那么select的value是选中option的value值  属性 name value  selected 默认选中  size 默认值为1，  值为1的时候，为下拉选择框  值>1的时候，为滚动选择框  multiple 无值属性，多选 |  6.其它元素  |  | | --- | | 1.label 关联文本与表单的控件  <label for=""></label>  属性for绑定要关联的表单控件的ID |  7.为控件分组  |  | | --- | | <fieldset> 分组  <legend>标题</legend>  form表单控件......  </fieldset> | |

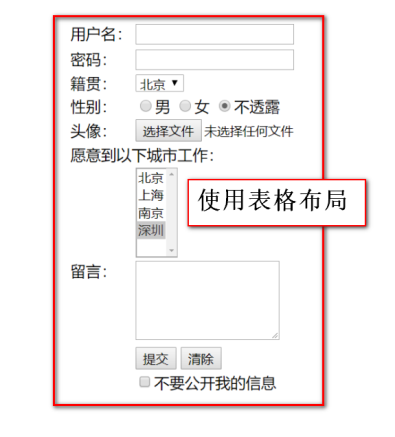
## 十一.浮动框架

|  |
| --- |
| <iframe style="width:100%" src="01\_ex.html"  frameborder="0" scrolling="no"></iframe>  语法 <iframe></iframe>  属性 src 引用的资源路径  frameborder="0" 清除边框  scrolling="no" 清除滚动条  width/height |

|  |
| --- |
| colspan 跨列，删除当前tr中td  rowspan 跨行，删除其他tr中td |

作业：

1.表单练习



2.完成下图布局



3.nodejs 6个接口，配合今天学习form。

day03

八.列表

1.有序

2.无序

## 3.列表的嵌套

|  |
| --- |
| 1.列表中嵌套其他元素 |
| 2.列表中嵌套列表 |
| 原则，嵌套只能写在li中。不管是ol还是ul，直接子元素只能是li.  li元素中直接嵌套li，浏览器按没有嵌套的方式解析，所以li中嵌套列表只能写ol或ul |

4.定义列表

|  |
| --- |
| h5新出的，一个名词，以及对这个名词的解释和定义  <dl> 定义列表  <dt> </dt> 要解释说明的名词  <dd></dd> 对名词解释的内容  </dl> |

## 九.结构标签

|  |
| --- |
| h5新出的标签，专门用于网页布局  为了让代码有可读性，h5推出一堆与div功能一模一样的标签，专门用作网页布局  1.<header></header>  定义网页的头部，或者某个区域的顶部  2.<footer></footer>  定义网页的脚部，或者某个区域的底部  3.<nav></nav>  定义网页的导航栏  4.<section></section>  定义网页的主体内容  5.<aside></aside>  定义网页的侧边栏  6.<article></article>  定义与文字相关的内容  比如，论坛，回帖，用户评论.... |

## 十.表单，\*\*\*\*重点\*\*\*\*\*

1.作用

|  |
| --- |
| 提供可视化的输入控件  收集用户输入的信息，并提交请求给服务器  总结：form自带提交请求的功能  ajax提交不需要form的支持  在同一个功能中，使用form就不用ajax,使用ajax就不用form |

2.form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 nodejs+mysql |

3.form表单详解

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| <form></form>  属性：   |  | | --- | | 1.action="" 定义表单提交时，发送请求的动作(url)  如果不写值，默认提交给本页 | | 2.method="" 定义表单的提交方法  取值   |  | | --- | | 1.get 默认值  特点：明文提交(参数在地址栏中显示)，不安全  提交的数据有大小限制，最大为2kb  使用场合：向服务器要数据的时候，使用get  后台：req.query | | 2.post  特点：隐式提交，参数不在地址栏中显示  提交数据没有大小限制  使用场合：要传递数据给服务器的时候，使用post  后台：req.body | | 3.其它的提交方法  delete put option....... | | | 3.enctype="" 指定表单数据的编码方式  设置允许将什么类型的属性传递给服务器  取值：  1.application/x-www-form-urlencoded  默认值，允许将任意的字符提交给服务器  (不能提交文件)  2.text/plain 允许提交普通字符给服务器，不能包含特殊符号 & \*  3.multipart/form-data 允许提交文件给服务器 |   <form action="url" method="get"  enctype="application/x-www-form-urlencoded">  </form>  收集的数据，发送给url这个接口，使用get提交  要求提交的数据是任意字符，不是文件 |

4.表单控件

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 在form标签中，能够与用户交互的可视化控件  1.表单控件的分类   |  | | --- | | input元素 基础9种  textarea 多行文本域  select+option下拉选择器  其它元素  h5新出的input元素 10种 |   2.input元素   |  | | --- | | <input type="">  在页面中，input提供了各式各样的输入控件  文本框，密码框，单选框，多选等....  共同的属性  1.type="" 指定input元素的类型  2.name="" 为控件定义名称，此名称是给服务器用的，如果想往服务器传递参数，此名称必须写  3.value 是控件的值，真正传给服务器的值  例外,在按钮控件中,value是设置按钮上显示的文字  4.disabled 禁用，不能修改值，也不能提交  取值：disabled="disabled"  disabled="true" --js使用  disabled  这种属性，称之为无值属性 |   3.input详解9种   |  | | --- | | 1.文本框和密码框  文本框 type="text" input默认是text  密码框 type="password"  属性：  maxlength 设置输入的最大值  readonly 无值属性，只读，不能修改，允许提交  placeholder 占位提示符 | | 2.按钮  提交按钮 type="submit"  将表单中的数据收集整理，发送给服务器  普通按钮 type="button"  没有功能，配合事件，调用js代码  重置按钮 type="reset"  将当前表单内容恢复到初始化状态  按钮中value，设置按钮上的文本，此value不提交 | | 关于按钮的题外话  h5给我们推出了新的按钮标签  <button></button> 自带submit功能 | | 3.单选和复选按钮  单选按钮 type="radio"  多选按钮 type="checkbox"  属性name，除了定义控件名称之外，还起到分组的作用，分了组才会有单选效果  必须有value属性，不然传递的参数永远是on  checked 无值属性 设置默认被选中的项 | | 4.隐藏域  type="hidden"  想把数据提交给服务器，但是又不想用户看到  这种数据放到隐藏域中  用户看不见，但是可以提交 | | 5.文件选择框，上传文件  type="file"  前提：提交方法，使用post  enctype属性，设置为multipart/form-data  属性multiple，可以选择多个文件上传 |   4 textarea多行文本域   |  | | --- | | <textarea></textarea>  允许录入多行文字  cols 设置输入的列数  rows 设置输入的行数  根据计算硬件不同，不准确 |   5.下拉选   |  | | --- | | <select>  <option></option>  <option></option>  ....  </select>  注意：  1.如果option没有value，那么select的value是option的内容  2.如果option有value,那么select的value是选中option的value值  属性 name value  selected 默认选中  size 默认值为1，  值为1的时候，为下拉选择框  值>1的时候，为滚动选择框  multiple 无值属性，多选 |   6.其它元素   |  | | --- | | 1.label 关联文本与表单的控件  <label for=""></label>  属性for绑定要关联的表单控件的ID |   7.为控件分组   |  | | --- | | <fieldset> 分组  <legend>标题</legend>  form表单控件......  </fieldset> | |

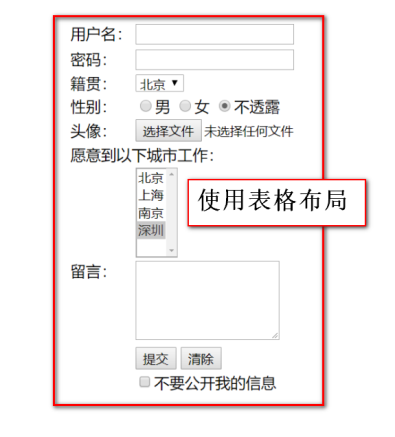
## 十一.浮动框架

|  |
| --- |
| <iframe style="width:100%" src="01\_ex.html"  frameborder="0" scrolling="no"></iframe>  语法 <iframe></iframe>  属性 src 引用的资源路径  frameborder="0" 清除边框  scrolling="no" 清除滚动条  width/height |

|  |
| --- |
| colspan 跨列，删除当前tr中td  rowspan 跨行，删除其他tr中td  之前在网页中实现共享广告的嵌套，现在一般在服务器编程中用“SSI”技术实现 |

作业：

1.表单练习



2.完成下图布局



3.nodejs 6个接口，配合今天学习form。

day04

一.form表单

## 1.h5新表单元素

### ①邮箱，提交的时候，验证@以及@前后至少有一个字符

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

### ②搜索类型，自带快速清除功能

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

### ③url类型，提交时，验证以http://开头

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

### ④电话号码，移动端使用时，会弹出虚拟键盘

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

### ⑤数字类型

|  |
| --- |
| <input type="number"  max="10" min="2" step="3">  max：能接收的最大值  min：能接收的最小值  step：步长，按一次按钮，变化几个数字 |

### ⑥范围类型

|  |
| --- |
| <input type="range" max="22"  min="10" step="4">  max：能接收的最大值  min：能接收的最小值  step：步长，按一次按钮，变化几个数字 |

### ⑦颜色类型

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

### ⑧日期类型，提供了一个日期选择的控件

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

### ⑨月份类型

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

### ⑩周类型

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

超纲

|  |
| --- |
| <button></button>  替代<input type="submit"> 增加可读性 |

# ajax

## 二.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>:<pwd>@<host>:<port>/  <path>;<params>?<query>#<frag> |

①scheme：方案，协议。设置以哪种方式获取服务器资源

协议是不区分大小写的，常用的协议http:/ftp：...



②<user>:<pwd> 用户名：密码。此种写法已过时。

③<host> 主机名 (域名、ip地址)

④<port> 端口号

⑤<path>资源，在服务器上具体存放的位置

⑥<params> 参数，保存跟踪状态的参数(session/cookie)

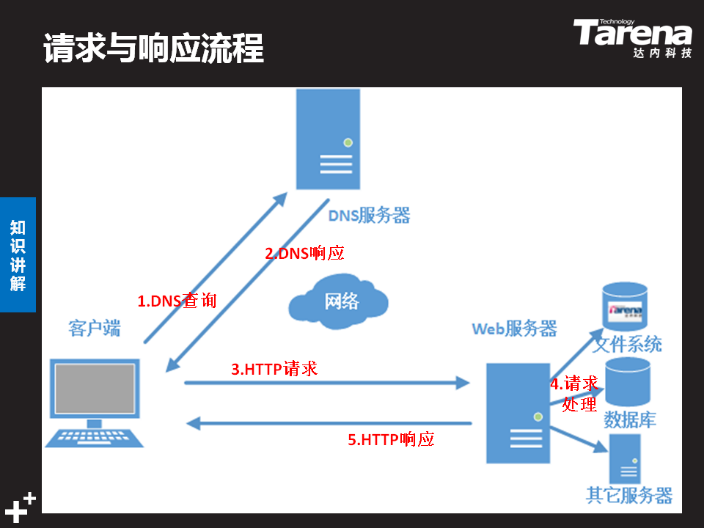
⑦<query>get方法提交请求时的查询字符串

⑧#<frag>锚点

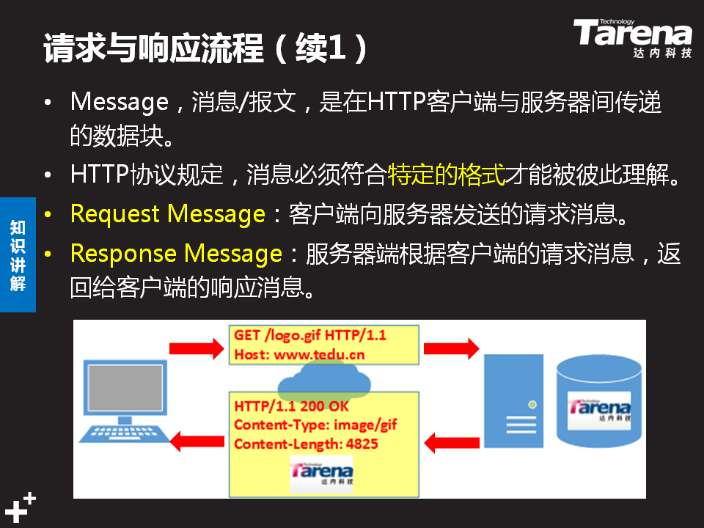
## 2.HTTP协议

|  |
| --- |
| HTTP:HyperText Transfer Protocol 超文本传输协议  规范了数据是如何打包以及传递的  (专门用于传输html文件的协议，专门浏览网页的协议)  HTTP协议的历史  C:\Users\web\Desktop\web1903\web第二阶段练习效果图示\HTTP协议历史.png |

### 3.web请求原理详解



## 4.消息/报文 Message



①请求消息 Request Message(请求起始行，请求头，请求主体)

②响应消息 Response Message(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  get:明文提交，不安全，有大小限制 2kb  客户端向服务器要数据的时候使用get  靠地址栏传递查询字符串的方式传数据，  无请求主体  post:隐式提交，无大小限制  客户端要传递数据给服务器时使用。  有请求主体，靠请求主体传递数据  delete:客户端想要删除服务器上的内容(一般禁用)  put：客户端想往服务器上放文件内容(一般禁用)  connect:测试连接  trace：追踪请求路径  option 预请求  head：表示客户端只获取响应消息头  2.协议版本号 HTTP/1.1  3.请求的url |

②请求头

|  |
| --- |
| 1.host：www.tmooc.cn  告诉服务器请求的是哪一个主机  2.Connection:keep-alive  告诉服务器开启持久连接  3.User-Agent：告诉服务器，浏览器的类型和浏览器版本号  4.Accept-Encoding:gzip  告诉服务器，我这个浏览器可以接收的压缩文件的格式  5.Accept-Language:zh-CN  告诉服务器，我这个浏览器可以接收的自然语言的类型  6. Referer: http://www.tmooc.cn/free/  告诉服务器，当前请求来自于哪个网页 |

③请求主体

|  |
| --- |
| form data  get请求没有请求主体  post有请求主体 |

## 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 http/1.1  2.响应状态码  告诉浏览器，服务器的响应状态是什么  1XX 正在请求，提示信息  2XX 200 响应成功  3XX 重定向 301 永久重定向  302 临时重定向  304 同一个请求，没有任何变  命中缓存  4XX 404 请求资源不存在  403 权限不够  405 请求方法不被允许  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:Sun, 05 May 2019 08:19:02 GMT  告诉浏览器，服务器响应的时间，格林威治时间  2.Connection:keep-alive  告诉浏览器，服务器已经开启了持久连接  3.Content-Type:  告诉浏览器，响应主体的类型是什么  取值 text/html 响应回来的数据是html文本  text/css 响应回来的数据是css文本  application/JavaScript 是js文本  image/jpg gif png 响应回来的是图片  text/plain 响应回来的是普通文本  application/json响应回来的是json字符串  application/xml 响应回来的是xml字符串 |

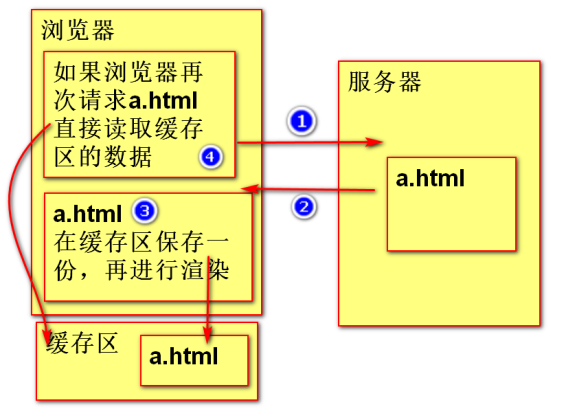
③响应主体

|  |
| --- |
| 服务器传递给浏览器的数据 |

## 7.缓存

客户端经服务器响应回来的数据进行自动的保存

当再次访问的时候(请求不改变)，直接使用保存的数据



①缓存的优点：

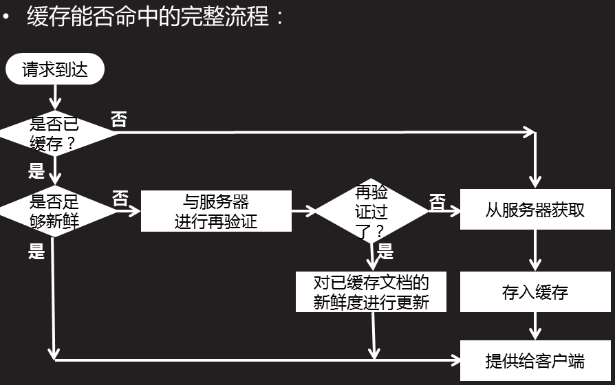
1.减少冗rong余的数据传输，节省客户端流量

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加载延迟

②缓存的新鲜度和过期



|  |
| --- |
| 1.请求--无缓存--连接服务器--存缓存--客户端得到数据  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--连接服务器确认是否过期--没过期--更新缓存的新鲜度--客户端得到数据  4.请求--有缓存--不新鲜--连接服务器确认是否过期--已过期--连接服务器拿数据--存缓存--客户端得到数据 |

③与缓存相关的消息头

|  |
| --- |
| Cache-Control:max-age=0  从服务器将响应传到客户端之时起，此数据处于新鲜的秒数，这是一个相对时间  语法：Cache-Control:max-age=3600 |

④在网页中添加缓存，需要修改消息头

|  |
| --- |
| <meta http-equiv="消息头属性" content="值">  <meta http-equiv="Cache-Control"  content="max-age=3600"> |

## 8.HTTP性能优化

①HTTP的连接过程

|  |
| --- |
| 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 |

②HTTP连接性能优化

|  |
| --- |
| 1.减少连接的创建次数(开启持久连接)  2.减少请求的次数（整合代码，不要让文件过多） 3.提高服务器端运行速度  4.尽可能的减少响应数据的长度 |

9.安全HTTP协议。https

|  |
| --- |
| HTTPS：安全版本的http协议  SSL：为数据通信特供安全支持  ①客户端发送请求--->ssl层加密-->服务器接收到加密文件-->ssl层解密，得到明文  ②服务器发送响应--->ssl层加密-->客户端得到加密文件-->ssl层解密，得到响应内容 |

## ajax常见错误

|  |
| --- |
| 404 请求资源不存在  url是否写错  接口的路径是否写错  路由是否写错  文件位置是否放错 |
| 500服务器代码错误  一定是服务器中代码出错 |

作业

1.熟练完成nodejs接口，路由，静态资源

2.使用table完成下面布局



day05 Ajax

## 一.Dom操作(简单dom操作)

|  |
| --- |
| ajax提交请求，不需要使用form表单  但是，form表单自带收集数据的功能  不使用form标签，就没有自动收集数据的功能了  我们需要自己手敲代码，收集数据  使用dom技术 |

## 1.完整的javascript的组成

①js核心，ECMA Script 6

②Dom:Document Object Model 文档对象模型

③Bom:Browser Object Model 浏览器对象模型

## 2.使用js的Dom获取页面某个元素

①获取元素对象

|  |
| --- |
| 1.使用元素的id获取对象  var elem=document.getElementById("id"); |

②通过对象，获取元素的内容/值

|  |
| --- |
| 双标签中间的东西，叫内容  单标签的value，叫值 |
| 获取内容，使用dom对象的innerHTML属性  获取：var res=elem.innerHTML;  设置：elem.innerHTML="abc"; |
| 获取值，使用dom对象的value属性 |

③使用dom操作的进阶

|  |
| --- |
| 获取对象的方案进阶  1.不使用id就能获取对象(后期xml中要使用)  通过标签名称获取当前页面中所有的该标签的元素，通过下标，获取某一个元素  var elem=document.getElementsByTagName("div")[0];  2.使用id获取对象的简写方式，ES6支持的写法  直接使用id代表对象  <div id="d1">测试文本1</div>  var res=d1.innerHTML; |

练习：01\_ex.html 一个div，一个input-text,button

需求，点击按钮，把input-text中填入的文本

显示到div上

|  |
| --- |
| 对于innerHTML的进阶  innerHTML是可以改变html的结构的  <div id="d1">  <a href="">123</a>  </div>  <script>  console.log(d1.innerHTML);  //innerHTML是可以改变html的结构的  d1.innerHTML+="<h1>abc</h1>"  </script> |

④事件

|  |
| --- |
| 通过用户的行为来激发的操作，就是事件  用户发生一个行为，这个行为绑定的方法就会被调用  button的事件  onclick 单击事件  文本框和密码框  onblur 焦点移除事件  onfocus 获取焦点事件  body标签的事件  onload 页面加载完成后，马上调用js方法 |

练习

|  |
| --- |
| 03\_ex.html  1.页面中创建如下元素  文本框 uname  密码框 upwd  重复验证密码框 cpwd  在上面三个元素后面各创建一个span（3个span）  2.需求  1.文本框获取焦点时，span提示用户名6~10位  失去焦点时，非空验证，显示span中  2.密码框同上  3.cpwd失去焦点时，判断与upwd的值是否一致，提示写在span中 |

## 二.Ajax

### 1.同步Synchronous

|  |
| --- |
| 在一个任务进行的过程中，不能开启其它任务  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器响应，不能做其他事  同步访问的出现场合：  1.地址栏输入url，是同步访问  2.a标签跳转  3.form提交 |

### 2.异步Asynchronous

|  |
| --- |
| 在一个任务开启时，可以进行其它的任务  异步访问：浏览器在向服务器发送请求时，用户可以在页面上做其它的操作  使用异步的场合：  1.用户名重复的验证  2.聊天室  3.百度搜索建议  4.股票走势图 |

### 3.什么是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 |

### 4.使用ajax

|  |
| --- |
| 1.创建异步对象  2.创建请求  3.发送请求  4.接收响应数据 |

①创建异步对象

|  |
| --- |
| var xhr=new XMLHttpRequest();  这种创建方式，不兼容IE8以下的版本  下面是兼容ie8以下版本的创建方式  if(window.XMLHttpRequest){  //如果有这个属性，说明是IE8以上的浏览器  var xhr=new XMLHttpRequest();  }else{  //如果没有这个属性,说明是IE8以下的浏览器  var xhr=new  ActiveXObject("Microsoft.XMLHttp");  } |

②创建请求，打开连接

|  |
| --- |
| xhr.open(method,url,isAsyn);  method：string类型，请求的方法  url:string类型，请求的url  isAsyn:boolean，是不是要采用异步的方式提交请求 |

③发送请求

|  |
| --- |
| xhr.send(formdata);  注意：只有post请求，才有请求主体formdata  get方法不需要请求主体。  所以使用get请求的时候，有两种发送请求的写法  xhr.send(),xhr.send(null) |

④接收响应

|  |
| --- |
| 1.xhr.readyState属性  用于表示xhr对象的 请求状态  一共有5个状态  0：请求尚未初始化  1：已经打开连接，请求正在发送中  2：接收响应头  3：接收响应主体  4：接收响应数据成功 |
| 2.xhr.onreadystatechange监听  监听xhr.readyState值的改变，每改变一次，方法会调用一次，一共会调用4次。  注意：只有当xhr.readyState=4的时候，是我们要接收响应正确时刻 |
| 3.xhr.status响应状态码  只有响应状态码为200的，我们才接收响应  4.响应数据放在了xhr.responseText属性中 |
| xhr.onreadystatechange=function(){ if(xhr.readyState==4&&xhr.status==200){ var result=xhr.responseText;  console.log(result);  }  } |

5.带参数的get请求

注意：

|  |
| --- |
| 1.服务器中的get接口，可以使用浏览器地址栏验证 |
| 2.跨域错误    把ajax的html文件放入node服务器的静态资源中托管，使用访问服务器的方式打开页面 |
| 3.大括号缺失 |
| 4.没有错误信息的错误  使用console.log在不同位置打桩观察 |
| 5.错误的符号 |

作业1：

|  |
| --- |
| 使用get请求，完成完整的登录模块  input输入---ajax异步发送请求---服务器接收数据--查询数据库--发送响应--页面接收响应 |

作业2：

|  |
| --- |
| 使用get请求，获取商品列表，把响应得到的数据在div中显示 |

提高作业：

|  |
| --- |
| 注册验证，验证新用户名是否可用  模仿学子商城注册页面，使用onblur事件 |

day06

作业1：

|  |
| --- |
| 使用get请求，完成完整的登录模块  input输入---ajax异步发送请求---服务器接收数据--查询数据库--发送响应--页面接收响应 |
| 1.完成接口  2.完成ajax  C:\Users\web\Desktop\web1903\day06\day06_getlogin_html.pngC:\Users\web\Desktop\web1903\day06\day06_getlogin_node.png |

## 一.ajax的post请求

### 1.任务，使用get请求再完成一遍登录模块 10:45我写

接口名称 login\_post

### 2.使用post方法发送请求

①xhr.open，需要改成post方法

②xhr.open中的url，不需要？和参数了

③添加创建请求主体的代码，把数据放到请求主体中

④修改请求消息头，可以发送特殊符号

⑤xhr.send(请求主体)

|  |
| --- |
| //url不要?和后面参数  var url="http://127.0.0.1:8080/ajax/login\_post";  xhr.open("post",url,true);  //创建请求主体  1.ajax更改  var formdata="uname="+u\_name  +"&upwd="+u\_pwd;  //由于ajax默认传输是text/plain  //无法传递特殊符号，我们需要更改消息头  //让ajax请求可以传递特殊符号  xhr.setRequestHeader("Content-Type","application/x-www-form-urlencoded");  xhr.send(formdata);  1.接口更改  方法为post  改为req.body.uname获取接收数据 |

## 二.Json的解析

作业2：

|  |
| --- |
| 使用get请求，获取商品列表，把响应得到的数据在div中显示  1.创建一个接口userlist  2.在html使用ajax |
| 前台，ajax内部，xhr.responseText得到数据，是string类型。string中放着js对象组成的数组 1.js对象的数据格式 var obj={name:"abc",age:18}  语法：var 对象名称={属性1：值1，属性2：值2.....}  注意：js对象中，属性名没有双/单引号 2.JSON数据格式 json是一个string字符串，亲切的称之为json串  JavaScript object Notation  js 对象 表示法  以js对象的格式表现出来的字符串 3.json字符串的格式 ①用一对{}表示一个对象  ②json中对象的属性名，必须使用""引起来(使用单引号也是正确的，但是不推荐)  ③属性的值如果是字符串，也要带双引号  ④json的表现是一个字符串，所有最外层加引号(推荐使用单引号)  ⑤请求访问服务器，调用数据库代码，得到的结果，默认是一个json串 4.json解析，把json字符串转换成js对象数组 var arr=JSON.parse(result); |

练习

|  |
| --- |
| 04\_list.html 重写一遍get方法获取userlist01接口  要求，数据最后变成下图样式 15:05~15:20休息  C:\Users\web\Desktop\web1903\web第二阶段练习效果图示\ajax用户列表效果.png |

## 三.XML的解析

### 1.什么是XML

|  |
| --- |
| eXtensible Markup Language  可拓展 标记 语言  是html的一个变种，专门负责承载数据用的  xml中标签，关键字/属性都需要自己定义  语法  1.第一行写版本声明  <?xml version="1.0" encoding="utf-8"?>  2.xml的标记只有双标记，必须成对出现  3.xml的标签严格区分大小写，开始标签和结束标记必须一致  4.xml标记，允许嵌套，但是要注意嵌套顺序  5.每个xml文档，有且只有一个根元素 |

### 2.xml解析

|  |
| --- |
| 使用dom解析  ??.getElementsByTagName("student")  ①不能使用xhr.responseText获取xml。这样获取的xml不能解析。要使用var result=xhr.responseXML;  ② |

四.项目阶段

## 常见错误

|  |
| --- |
| 数据库出问题，有可能是sql语句错误 |
| 设置请求消息头，必须在xhr.opne和xhr.send之间写 |
| xml文件中，有结构错误 |

## ajax项目

### 一.搭建项目结构

|  |
| --- |
| router文件夹中创建路由pro.js---------接口  挂载路由  创建静态文件夹mypro，与public平级-------html  静态文件夹资源的托管 |

1.登陆模块

|  |
| --- |
| 1.在pro.js中定义接口login  使用post方法，接收前端传递过来的用户名和密码  查询数据库，给出响应  2.在mypro文件夹下，新建网页01\_login.html  添加文本框和密码，点击按钮完成ajax异步请求  登陆响应结果使用弹窗显示17:13完成 |

2.显示用户列表的模块

|  |
| --- |
| 1.在pro.js中创建接口list get  查询用户所有数据，并响应给前端  2.在mypro文件夹中，创建网页02\_list.html  页面加载时，发起ajax异步请求。把数据格式显示成下图模式  C:\Users\web\Desktop\web1903\web第二阶段练习效果图示\ajax用户列表效果.png |

作业：

1.使用get方法获取userlist接口数据，显示成表格



2.删除

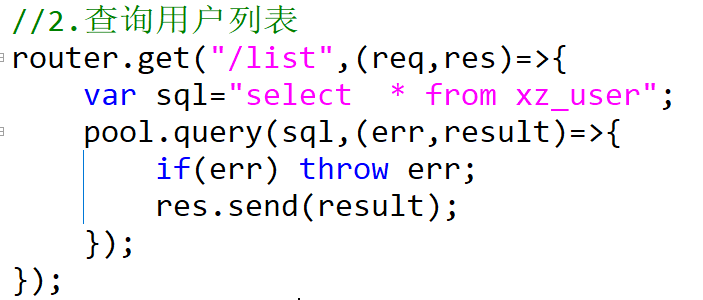
|  |
| --- |
| input-text输入uid,点击按钮，删除数据库中数据 |

day07

## 一.登录模块

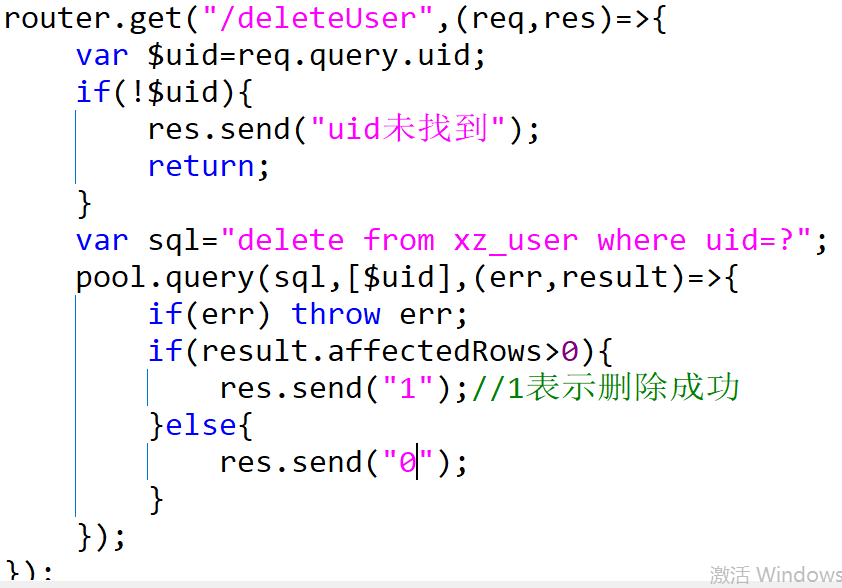
二.查询所有的用户信息9:35我写

|  |
| --- |
| 1.在pro.js中创建接口list get  查询用户所有数据，并响应给前端  2.在mypro文件夹中，创建网页02\_list.html  页面加载时，发起ajax异步请求。把数据格式显示成下图模式  C:\Users\web\Desktop\web1903\web第二阶段练习效果图示\ajax用户列表效果.png |



## 三.删除

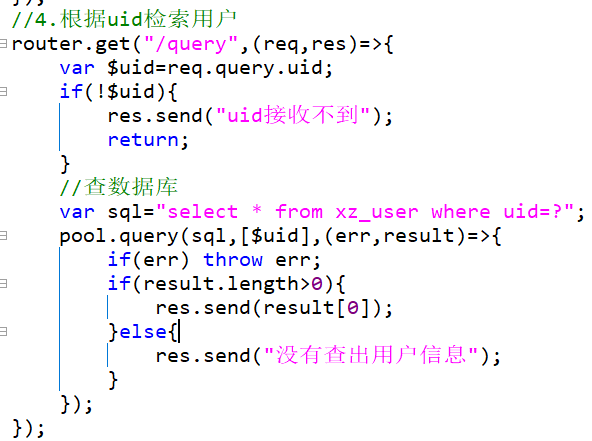
|  |
| --- |
| input-text输入uid,点击按钮，删除数据库中数据  get方法 |





## 四.修改模块

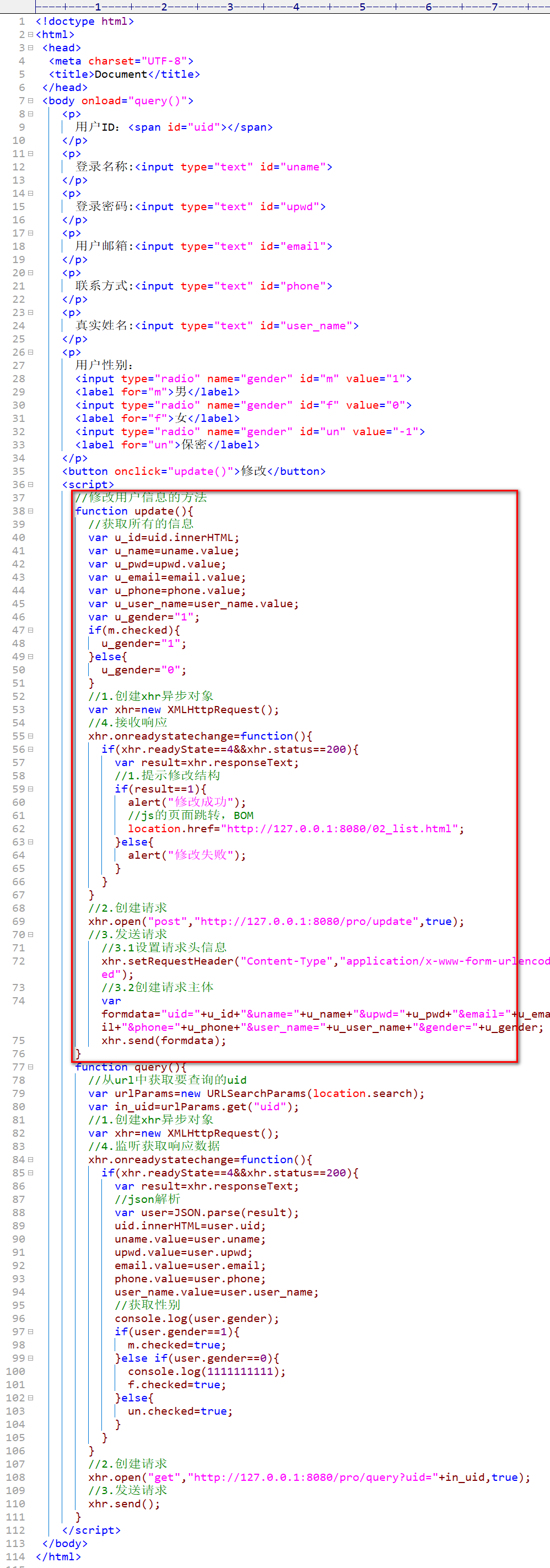
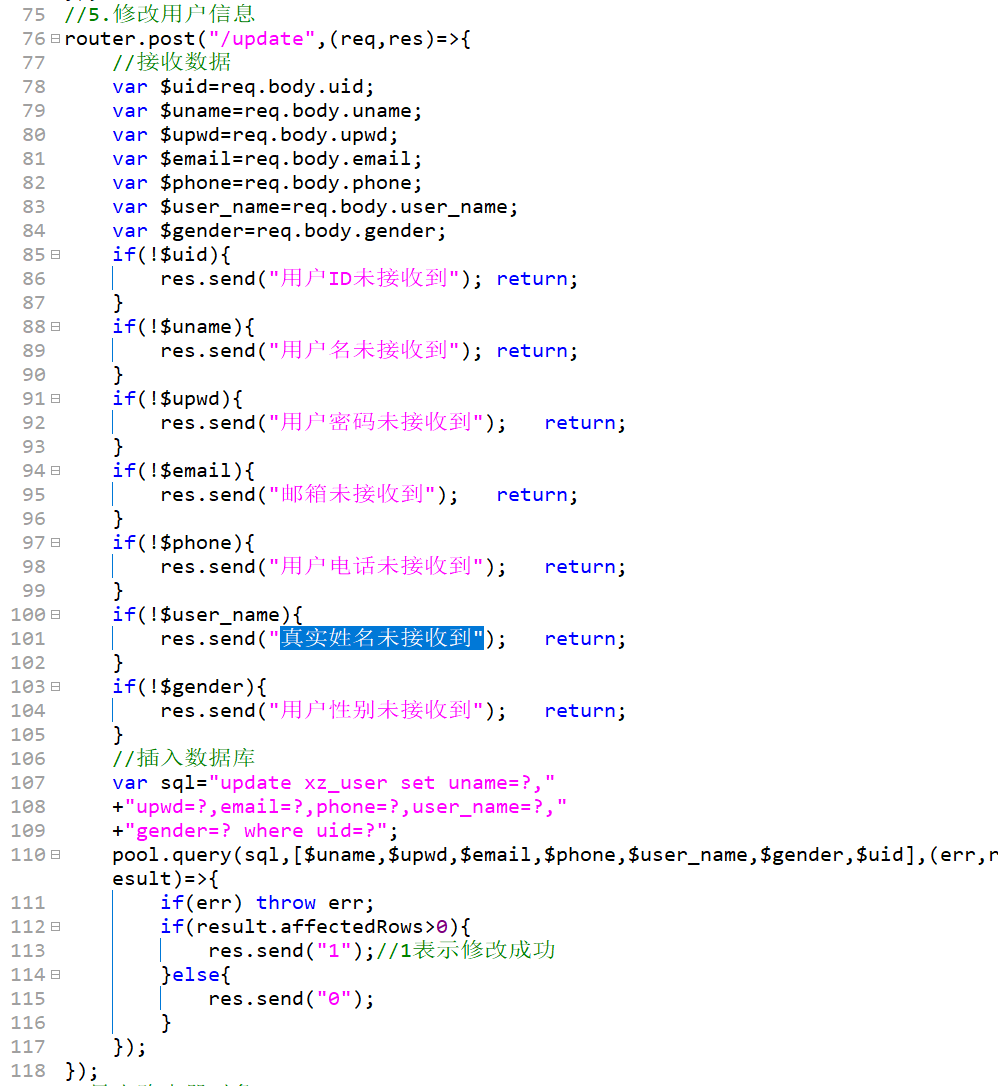
### 1.根据uid检索用户





### 2.修改模块

接口post方法 /update



作业：

## 1.注册

|  |
| --- |
| 1.注册名重复的验证  2.账户注册 |

常见错误

|  |
| --- |
| 服务器没有接受到参数，把提示当成响应传递给前台  前台把提示按照json格式解析了 |

# css

day08

## 一.CSS概述

1.什么是css

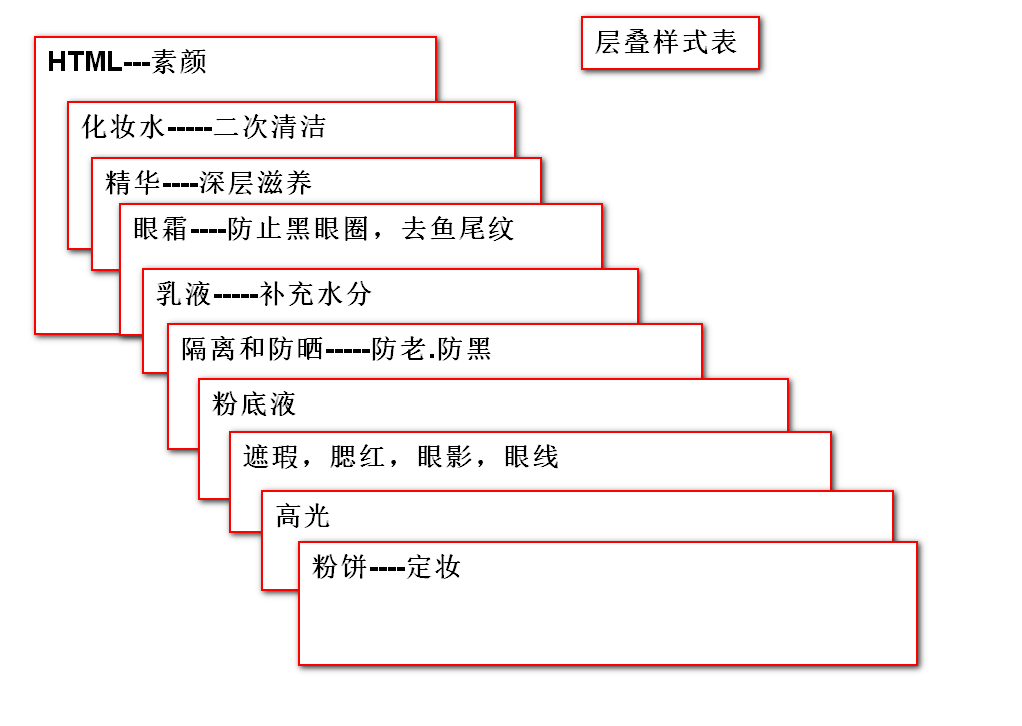
|  |
| --- |
| Cascading style sheets 层叠样式表，级联样式表  简称样式表 |

2.css的作用

|  |
| --- |
| 设置html网页中元素的样式，美化页面 |

3.HTML与CSS的关系

|  |
| --- |
| HTML：负责网页的搭建，内容的展示  CSS：负责网页的修饰，样式的构建 |



### 4.CSS与HTML的属性，使用原则

|  |
| --- |
| W3C规定我们尽量的使用css的方式来取代HTML属性  css样式  1.样式代码可以高度重用，html属性不能重用  2.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)

|  |
| --- |
| 在html标签的style属性中，编写样式  语法：<any style="样式声明"></any> any:任意标签  样式声明：样式属性 : 值  有多个样式声明：样式属性1：值1 ；样式属性2：值2  ex:  <div style="color:red;background-color:yellow">  内联样式在项目中，很少使用。  学习和测试的时候用  原因：1.内联样式不能重用  2.内联样式优先级最高 |

目前常用的样式

|  |
| --- |
| color:orange; 字体颜色  background-color:yellow; 背景颜色  font-size:36px； 字号大小 |

②内部样式

|  |
| --- |
| 在网页头部中，添加一对<style></style>标签  在style标签定义此网页所有的样式  <style>  选择器{样式属性：值；样式属性：值；........}  </style>  选择器：就是一个条件，符合这个条件元素，可以应用这个样式  项目中使用内部样式较少，学习和测试使用较多  内部样式的重用不能在其他html页面中生效 |

③外部样式

|  |
| --- |
| 单独创建一个css文件，编写样式  在html页面的head中使用link，引入这个外部样式  <link rel="stylesheet" href="css文件的url">  注意：rel="stylesheet"必须写，不然无效  此种方式，是开发中使用最多方式。 |

练习

|  |
| --- |
| 01\_ex.html  h1 内容随意，内联，背景为粉色pink，字体为yellow  h2 内容随意，内部，背景为粉色yellow，字体为red  h3 内容随意，外部，背景为粉色red，字体为blue |

### 2.CSS的特性

①继承性

|  |
| --- |
| 大部分的css效果，是可以被子元素继承的  必须是父子结构，子继承父 |

②层叠性

|  |
| --- |
| 可以为一个元素定义多个样式规则  多个规则中样式属性不冲突时，可以同时应用到当前元素上 |

③优先级

|  |
| --- |
| 如果对一个元素的多个样式声明，发生冲突时，按照样式规则的优先级去应用  默认优先级：由高到低  1.内联样式(行内样式)  2.内部样式和外部样式，就近原则  3.浏览器默认样式 |

④调整优先级

|  |
| --- |
| !important规则  放在属性值后面，与值之间有一个空格  作用，调整样式显示的优先级  内联样式不可以写!important  !important优先级，比内联样式高 |

练习

|  |
| --- |
| 02\_ex.html中  一个p标签，内容假文lorem  用内部样式设置文本颜色为蓝色，字号24px  用外部样式设置文本颜色为 红色，字号40px  让外部样式引入，F12查看页面效果  然后对调内部样式和link的位置，F12观察结果  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用  (选择器是一个条件，符合这个条件的元素就可以使用这个样式) |

### 2.选择器详解

①通用选择器

|  |
| --- |
| \*{样式声明}  由于\*的效率低下，项目中很少使用通用选择器  唯一使用的场合\*{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)

|  |
| --- |
| 页面中所有对应的元素，都应用这个样式  设置页面中某种元素的默认样式  元素名称{样式声明}  ex:p{} div{} |

③ID选择器，专属定制

|  |
| --- |
| <any id="id值"></any>  #id值{样式声明} 这种写法仅仅对页面上一个标签生效  一般id选择器在项目中很少单独使用。  通常会作为子代选择器或者后代选择器的一部分 |

练习

|  |
| --- |
| 03\_ex.html h2标签，内容随意(假文)，这个h2标签id为text1  使用id选择器设置文本为purple，背景yellow，  字体为斜体font-style:italic；查看页面观察效果  再使用元素选择器，设置文本为红色，背景pink。  这里有坑，会对就近原则产生怀疑，选择器权值问题 |

④类选择器

|  |
| --- |
| <any class="类名"></any>  .类名{样式声明}  作用：定义页面上某个或者某类元素的样式  (公共样式，谁想用，谁就可以用)  class属性来引用类名  类名的规则：1.类名之前的点不能省略，引用的时候没有点  2.类名不能以数字开头  3.类名只能包含 - \_ 符号  4.类名尽量的见名知意  类选择器的特殊用法  1.多类选择器  让元素引用多个类名，这些类的样式都会作用到当前元素上  <div class="text-danger bg-warning font-24">d</div>  2.分类选择器  元素选择器+类选择器{}  div.text-danger{ }div元素有text-danger类,才能应用这个样式  类选择器+类选择器{}  .font-24.text-danger{ }  元素，必须有font-24类和text-danger，才能应用此样式  作用：1.更精确的确定使用样式的元素  2.增加选择器的权值 |

练习

|  |
| --- |
| 04\_ex.html 页面中有div和p元素，内容假文随意  用类选择器设置字体颜色为红色，注意类名的命名规范  用分类选择器为p元素设置背景色为green |

⑤群组选择器

|  |
| --- |
| 将多个选择器放在一起定义公共的样式  语法：选择器1,选择器2,......{公共样式声明}  ex:  #content,div,.mycolor,p.text{color:red}  相当于  #content{color:red}  div {color:red}  p.text {color:red}  .mycolor {color:red} |

⑥后代选择器

|  |
| --- |
| 通过元素的后代关系匹配元素  后代：一级嵌套或者多级嵌套  语法：选择器1 选择器2 选择器3........{}  #content p span{color:red;}  id我content的元素，内部不管隔着多少代，有一个p元素  p元素内部，不管隔着多少代，有一个span.这个span就符合要求 |

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，直接的儿子  选择器1>选择器2>....{}  子代选择器和后代选择器可以混写  #content p>span{background-color:yellow;} |

练习

|  |
| --- |
| 05\_ex.html  ul#content>li\*4>span>a  a标签字体36px，字体颜色red，背景颜色yellow  使用后代或者子代选择器，至少3种选择器写法 |

⑧伪类选择器

|  |
| --- |
| 匹配同一个元素，不同的状态下的样式  所有的伪类选择器都是这样开头的 选择器:  1.链接伪类  a标签，没有访问的状态  a:link{color:red;}  a标签，被访问过之后的状态  a:visited{color:black;}  2.动态伪类  :hover 鼠标悬停在元素上时，元素的样式  :active 匹配元素被激活时的状态  :focus 匹配元素获取焦点时的状态 |

练习

|  |
| --- |
| 1.06\_ex.html  一个a标签，内容随意，href随意-------------坑  1.访问后，文本颜色orange  2.被激活时，文本颜色绿色  3.鼠标悬停时，文本颜色红色  4.未被访问时，文本pink  2.页面在红添加一个input--text  默认字体为gray，font-style:italic；  1.被激活时，字体不斜：font-style:normal;  2.获取焦点时：文本为红色 |

⑨选择器的权值

|  |
| --- |
| 权值：标识当前选择器的重要程度，权值越大，优先级越高  !important >1000  内联样式 1000  id选择器 100  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，需要将所有的选择器权值进行相加，然后比较，权值大的优先显示  2.权值相同，就近原则  3.群组选择器的权值，单独各算各的，不能相加  4.样式后面追加!important直接获取最高优先级  内联样式不能加!important  5.选择器的权值计算，结果不会超过自己的最大数量级  (1000个1相加，也不会大于10) |

作业1：css使用方法3个

css特性3个

选择器8个

权值

上述知识点的demo重做一遍

作业2：重新完成学子注册模块

day09

## 四.尺寸和边框

### 1.尺寸属性

|  |
| --- |
| 作用，设置元素的宽高  属性：  width: 设置宽  height:设置高  ----------------------  max-width 最大宽度  min-width 最小宽度  max-height 最大高度  min-height 最小高度  使用场合，响应式布局  取值：以px为单位的数字  父元素尺寸的%百分比 |

附加知识点：单位

|  |
| --- |
| 尺寸的单位  1.px像素  2.in英寸 1in=2.54cm  3.pt 磅值，多用于修饰字体大小粗细 1pt=1/72in  4.cm  5.mm  ---------相对单位-------------  6. % 父元素的百分之多少  7.em 是相对于父元素数值的单位  8.rem 是相对html标签数值的单位 |

### 2.页面中允许设置尺寸的元素

①块级元素

|  |
| --- |
| 所有的块级元素都可以设置尺寸  块级元素不设置宽，宽度占父元素100%。  块级元素不设置高，高度靠内容撑开，没有内容，就没有高 |

②行内元素

|  |
| --- |
| 行内元素设置宽高无效，  行内元素宽高，是靠内容撑开  但是，自带宽高属性的行内元素，可以设置尺寸 |

③table

|  |
| --- |
| table自带宽高属性，可以设置宽高 |

④行内块

|  |
| --- |
| input  行内块可以设置宽高 |

|  |  |  |
| --- | --- | --- |
| 块级元素 | 行内元素 | 行内块 |
| 独占一行 | 与其他行内和行内块共用一行，一行放不下自动换行 | 与其他行内和行内块共用一行，一行放不下自动换行 |
| 设置宽高有效 | 设置宽高无效 | 设置宽高有效 |
| 不设置宽，宽默认为父元素宽度100%，不设置高，高度默认为内容撑开 | 宽高都靠内容撑开 | 不设置宽高，浏览器会给一个默认的宽高。不同浏览器给的默认宽高不同 |
| 4个方向外边距有效 | 左右外边距有效  上下外边距无效 | 4个方向外边距有效  当一个行内块通过外边距改变垂直的位置时，会带着同一行其它行内块和行内元素一起移动 |

## 3.溢出处理

|  |
| --- |
| 当内容较大，元素区域较小，就会发生溢出效果  默认是纵向溢出  属性 overflow  取值 1.visible 默认值，溢出部分可见  2.hidden 溢出部分隐藏  3.scroll，不管是否溢出，都添加滚动条。  不溢出的时候，滚动条不能拖动  4.auto 自动，溢出的时候，溢出的方向有滚动条。  不溢出的时候，没有  控制滚动条的方向  overflow-x/overflow-y  如何让内容横向溢出  需要在宽度比较小的容器内部，添加一个宽度较大的元素盛放内容  在父容器上写overflow:auto。就可以做到横向溢出 |

附加知识点---颜色

|  |
| --- |
| 合法颜色值  1.颜色的英文单词(red,blue,yellow,pink,purple......)  2.#rrggbb  #000000---黑 #ffffff---白 #ff0000---红  #00ff00---绿 #0000ff---蓝  3.#aabbcc---->#abc  #000 #fff #f00 #0f0 #00f #666  4.rgb(0~255,0~255,0~255)  rgb(255,0,0)  5.rgba(r,g,b,alpha) alpha透明度 0~1 1不透明，0透明  #f00 #0f0 #00f #ff0 #0ff #f0f  #faa #afa #aaf #ffa #aff #faf |

## 4.边框

①边框属性---简写方式---4个方向边框一起设置

|  |
| --- |
| 边框简写 border:width style color;  width:边框的宽度，取值以px为单位的数字  style:边框的样式 取值：solid实线  dotted 点点虚线  dashed 线状虚线  double 双实线  color：边框的颜色 取值 合法的颜色值/transparent(透明)  最简方式 border:style;  取消边框 border:0; 或者 border:none; |

练习

|  |
| --- |
| 03\_ex.html  3个div,给每个div设置边框，每个div 10px\*10px  第一个 1px 实线 红色  第二个 5px 点虚线 蓝色  第三个 10px 线虚线 黄色 |

②单边定义边框

|  |
| --- |
| 只设置某一条边的3个属性  border-top：width style color;  border-right  border-bottom  border-left |

练习

|  |
| --- |
| 03\_ex中，div#d4 200px\*200px  上边框，3px实线 红色  右边框 5px 点 蓝色  下边框 10px 线虚线 黄色  左边框 15px 双实线 #f0f |

③单属性定义

|  |
| --- |
| border-width   1. border-width:thin medium thick 10px;（可自定义数字宽度）  * 上边框是细边框 * 右边框是中等边框 * 下边框是粗边框 * 左边框是 10px 宽的边框   ②border-width:thin medium thick;   * 上边框是 10px * 右边框和左边框是中等边框 * 下边框是粗边框   ③border-width:thin medium;   * 上边框和下边框是细边框 * 右边框和左边框是中等边框   ④border-width:thin;   * 所有 4 个边框都是细边框   border-style:可以一到四个值dotted solid double dashed  border-color:  ex:  border-width:3px;  border-style:solid;  border-color:#f00;  border:3px solid #f00; |

④单边单属性

|  |
| --- |
| 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(圆角)

|  |
| --- |
| 将直角设置成倒角，圆角  border-radius：  取值：以px为单位数字  %  50%就是个圆  单角设置  border-top-left-radius  border-top-right-radius  border-bottom-left-radius  border-bottom-right-radius |

练习

|  |
| --- |
| 画柠檬/芒果 |

## 6.边框阴影

|  |
| --- |
| box-shadow  取值：h-shadow v-shadow blur spread color inset;  h-shadow:水平方向的阴影偏移量  +：往右移动， -：往左移动  v-shadow:垂直方向的阴影偏移量  +：往下 -：往上  blur：阴影模糊距离，越大越模糊  无负值  spread：阴影尺寸，在基础阴影上扩出来的大小  负值，尺寸变小  color:阴影颜色  inset:向内扩撒阴影 |

练习

|  |
| --- |
| 圆形发光体 太阳，吸顶灯，日食，月食 |

## 7.轮廓

|  |
| --- |
| 轮廓指的是边框的边框，绘制于边框外的线条  outline:width style color;  outline:none;或者outline:0 去除 轮廓 |

## 五.框模型---盒子模型

|  |
| --- |
| 框模型--元素在页面上实际占地空间的计算方式  默认情况，一个元素在页面的实际占地宽度  左外边距+左边框宽度+左内边距+内容区域宽度+右内边距+右边框+右外边距  实际占地高度  上外边距+上边框宽度+上内边距+内容区域高度+下内边距+下边框+下外边距 |

### 1.外边距margin

|  |
| --- |
| 元素边框以外的距离，改变margin，元素有位移的效果 |

①语法

|  |
| --- |
| margin:v1; 设置4个方向的外边距  margin:v1 v2; v1上下 v2左右  margin:0 auto; 块级元素水平居中  auto对垂直外边距无效  margin:auto;  auto只对设置了宽的块级元素生效  margin:v1 v2 v3; v1上 v2左右 v3下  maring:v1 v2 v3 v4; 上右下左  单方向外边距设置  margin-top:  margin-right:  margin-bottom:  margin-left:  取值：1.以px为单位数字  2.%  3.+ margin-top ↓ margin-left →  - margin-top ↑ margin-left ←  4.auto 自动计算块级元素外边距，让块级元素水平居中  auto只对设置了宽度的块级生效  auto对下上外边距无效 |

|  |  |  |
| --- | --- | --- |
| 块级元素 | 行内元素 | 行内块 |
| 独占一行 | 与其他行内和行内块共用一行，一行放不下自动换行 | 与其他行内和行内块共用一行，一行放不下自动换行 |
| 设置宽高有效 | 设置宽高无效 | 设置宽高有效 |
| 不设置宽，宽默认为父元素宽度100%，不设置高，高度默认为内容撑开 | 宽高都靠内容撑开 | 不设置宽高，浏览器会给一个默认的宽高。不同浏览器给的默认宽高不同 |
| 4个方向外边距有效 | 左右外边距有效  上下外边距无效 | 4个方向外边距有效  当一个行内块通过外边距改变垂直的位置时，会带着同一行其它行内块和行内元素一起移动 |

**自带外边距的元素**

|  |
| --- |
| **body 自带8px,4个方向**  **ul 上下16px外边距，左内边距40px**  **h1~h6 p hr dl dd dt fieldset form legned** |

练习

|  |
| --- |
| 05\_ex  两个div，尺寸300px\*300px  背景色随意。  设置两个div之间的间距是50px，  两个div都水平居中 |

### 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  只能在布局设计的时候，尽量避免发生 |

作业

1：坑

|  |
| --- |
| 页面中两个div，宽高200px,分别设置背景颜色  为两个div设置4个方向外边距，观察效果  与div平级，添加两个span，内容随意，分别设置背景颜色  为两个span添加4个方向外边距，观察效果  与div平级，添加两个input-text，  为两个input添加4个方向外边距，观察效果 |

2.坑

|  |
| --- |
| div#d1>div#d2  #d1宽高300px  #d2宽高100px  分别设置背景颜色  给#d2设置上外边距，观察效果 |

3.今天所有知识点的demo

4.ajax 登录模块/注册模块/查询用户列表/删除/修改

day10

## 五.框模型

### 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。值以最大的为准。 |

练习

|  |
| --- |
| 01\_ex  1.两个div 200px\*200px 背景颜色不同  为两个div单独设置4个方向外边距 margin-方向  2.两个span，内容随意，背景颜色不同  为两个span单独设置4个方向外边距  3.两个文本框，  为两个input单独设置4个方向外边距 |

②关于块级元素，行内元素，行内块的外边距总结

|  |
| --- |
| 行内元素的特点：  设置宽高无效，宽高根据内容自动撑开  上下垂直外边距无效。可以与其他的行内元素和行内块元素共用一行 |
| 块级元素特点：  设置宽高有效。如果不设置宽高，高按内容撑开，宽占父元素宽度的100%。上下外边距有效，独占一行. |
| 行内块的特点：input  设置宽高有效，不设置宽高，自带默认宽高。  上下外边距有效，但是同一行修改一个行内块的垂直外边距，整行都会跟着一起改变位置。可以与其他行内元素和行内块共用一行 |

③自带外边距的元素

|  |
| --- |
| body h1~h6 p ol ul dl pre  **p自带margin-bottom 16px;**  **dl自带margin-bottom 16px;**  **h1~h6自带margin-bottom 8px;**  **ul 自带padding-left 40px;**  **margin-bottom 16px;**  **li 不带内外边距**  **pre标签会保留格式，源代码中的空格，换行等都会在页面上还原出来**  由于不同浏览器对默认的外边距解析可能有差别  所以一般情况下，开发之前，需要把内外边距清空  \*{margin:0;padding:0}  或者通过群组选择器来写  blockquote,body,button,dd,dl,dt,fieldset,form,  h1,h2,h3,h4,h5,h6,hr,input,legend,li,ol,  p,pre,td,textarea,th,ul{margin:0;padding:0} |

练习

|  |
| --- |
| 01\_ex  div#d3 尺寸300px\*300px  #d3中有div#d4 100px\*100px  两个div不同背景颜色。  给#d4设置上外边距，在f12中改变上外边距的数值  观察效果 |

④外边距溢出

|  |
| --- |
| 在特殊情况下，为子元素设置外边距，会作用到父元素上  特殊情况：  1.父元素没有上边框  2.子元素内容区域上边与父元素内容区域上边重合的时候  (为父元素中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，影响父元素实际占地高度  2.给父元素添加上内边距，弊端，影响父元素实际占地高度  3.给父元素的一个子元素的位置添加空table元素 |

### 3.内边距

|  |
| --- |
| 内边距的改变效果，感觉是改变了元素的大小  不会影响其它元素，但是真正改变的是本元素的占地尺寸  内边距颜色和元素背景相同  语法：  padding:v1; 设置4个方向的内边距  padding:v1 v2; 上下 左右  padding无auto  padding:v1 v2 v3; 上 左右 下  padding:v1 v2 v3 v4;上右下左  单方向设置  padding-top  padding-right  padding-bottom  padding-left  取值，以px为单位的数字 %  padding无auto值 |

### 4.box-sizing属性(设置元素实际占地尺寸的公式)

|  |
| --- |
| box-sizing:  取值：1.content-box.默认值，按照之前的盒子模型计算元素占地大小(左外+左边+左内+内容区域宽度+右内+右边+右外)  2.border-box。设置的width和height是包含border+padding+内容区域的 整体宽高  元素实际占地大小  左外+设置的width+右外  上外+设置的height+下外  给复杂元素关系做布局时，经常使用border-box |

## 六.背景

### 1.背景颜色

|  |
| --- |
| background-color:合法颜色值 |

### 2.背景图片

|  |
| --- |
| background-image:url(08.png);  使用背景图片，可以让该元素的子元素，堆叠显示在背景图片上。  而使用img标签，默认不会有堆叠效果 |

### 3.背景图片的平铺

|  |
| --- |
| background-repeat:  取值：repeat 默认值，平铺  no-repeat 不平铺  repeat-x 水平方向平铺  repeat-y 垂直方向平铺 |

### 4.背景图片的定位

|  |
| --- |
| background-position:  取值：1. x y 以px为单位的数字  2.x% y% 百分比  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. x y 以px为单位设置具体宽高  2.x% y% 按父元素宽高比设置  以元素的角度考虑这两个单词  3.cover 充满，只要求元素被背景图充满，背景图显示不全，也没关系.  4.contain 包含，只要求元素要把背景图整个包含住，元素有空白，也没关系 |

### 6.背景图片的固定

|  |
| --- |
| background-attachment  取值： scroll 默认值，背景图会随着窗口滚动条滚动  fixed 固定，背景图相对于窗口固定，窗口滚动时，背景图位置不变，但是只会在原容器内显示 |

### 7.背景的简写方式

|  |
| --- |
| background:  取值 color image repeat attachment position  最精简方式 background:color/image; |

## 七.渐变 gradient

1.什么是渐变

|  |
| --- |
| 渐变是指多种颜色，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，以及这种颜色出现的位置  一个渐变，最少两个色标 |

### 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，径向渐变重复几次显示 |

### 4.线性渐变

|  |
| --- |
| background-image: linear-gradient(angle,color-point1, color-point2......);  angle:表示渐变的方向  取值 1.关键字 to top 从下往上  to right 从左往右  to bottom 从上往下  to left 从右往左  2.角度 0deg to top  90deg to right  180deg to bottom  270deg to left  可以取负值  color-point:色标 颜色 %/px |

### 5.径向渐变

|  |
| --- |
| background-image:  radial-gradient(半径 at 圆心x 圆心y,color-point1,  color-point2.................)  半径 以px为单位的数字  圆心 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  color-point: 位置取值 1. % 半径的百分比%  2.px为单位的数字，半径失效 |

### 6.重复渐变

|  |
| --- |
| 重复的线性渐变  background-image:  repeating-linear-gradient(to left,#000 0px,  #ff0 25px,#000 50px);  重复的径向渐变  background-image:  repeating-radial-gradient(50px at center center,  #000 0%,#0ff 25%,#000 50%); 17:07~17:23休息 |

## 7.浏览器兼容问题

|  |
| --- |
| 如果有低版本(ie8.0以下)，想要使用渐变  chrome/safari -webkit-  firefox -moz-  opera -o-  IE -ms-  要低版本兼容渐变，需要在linear-gradient之前添加浏览器内核  background-image:  -webkit-linear-gradient()  background-image:  -moz-linear-gradient()  ......  兼容低版本浏览器，线性渐变的方向，要改变写法  不写to top/to left/to bottom/to right  改成初始点 top/left/bottom/right |

## 八.文本格式化(重点\*\*\*\*\*\*\*\*)

### 1.字体属性

①指定字号大小

|  |
| --- |
| font-size:  取值 px/pt/rem/em为单位的数字 |

②字体的类型

|  |
| --- |
| font-family |

一个页面中，编写css的思路

|  |
| --- |
| 从上往下，从左往右，从外往里  1.尺寸，大体位置  2.边框，背景相关  3.文本相关  4.微调 |

作业1：

margin:0 auto和text-align:center区别

作业2：完成登陆页的样式(table布局)



作业3.渐变的所有demo

day11

|  |
| --- |
| 1.尺寸，大体位置  2.边框，背景  3.文本  4.微调 |

## 三.文本格式化(重点\*\*\*\*\*\*\*\*)

### 1.字体属性

①字号大小

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

②设置字体类型

|  |
| --- |
| font-family:"mv boli",华文彩云,黑体;  在当前设备的字体库，查找字体，如果有就使用，如果没有，就查找下一个。  如果字体名称中间有空格，必须加双引号 |

③粗体

|  |
| --- |
| font-weight:  取值 1.关键字 lighter normal bold bolder  2.无单位的100的整倍数 一般400~1000 |

④字体样式

|  |
| --- |
| font-style:normal/italic; 正常和斜体 |

⑤小型大写字母

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

⑥简写方式

|  |
| --- |
| font:style variant weight size family;  font:italic small-caps bolder 40px chiller;  最精简的方式 font:size family; |

### 2.文本属性

①字体颜色

|  |
| --- |
| color |

②文本水平对齐方式

|  |
| --- |
| text-align:  取值 left/center/right/justify两端对齐 |
| margin:0 auto和text-align:center的区别  margin:0 auto，控制的是当前元素本身在页面中居中(让自己居中)  text-align:center 控制的是当前元素内部的内容，在元素内部居中(让我的儿子居中) |

③行高

|  |
| --- |
| 定义一行数据的高度  特性：如果行高的高度大于字体本身的大小  那么该行文本将在指定的行高内，呈垂直居中方式显示  line-height  取值：1.以px为单位的数字，一般，行高的值与容器高度相同。  这样就可以让文字在容器垂直中间显示了  2.无单位的数字(整数小数都可以)，行高为字号大小的倍数  注意:文字如果有多行，不建议使用行高，文字会溢出，每一行的行高都是你设置的大小 |

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 去掉所有的修饰线条  a{text-decoration:none}去除a标签下划线 |

⑤首行缩进

|  |
| --- |
| text-indent:  取值：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：阴影模糊度  color：阴影颜色 |

⑦文本换行

|  |
| --- |
| word-break  normal 使用浏览器默认的换行规则。  break-all 允许在单词内换行。  keep-all 只能在半角空格或连字符处换行  ex: 在合适的点换行：  p {word-break:break-all;} |

⑧文本溢出显示省略标记

|  |
| --- |
| text-overflow:ellipsis;  当对象内文本溢出时显示省略标记(...) ；需与overflow:hidden;一起使用。 |

练习

|  |
| --- |
| 01\_ex  定义div，内容随意，假文，有中文  1.设置字体系列  2.设置字号 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.文本有下划线  7.首行缩进30px  8.文本垂直居中 |

## 四.表格的样式

### 1.表格的常用样式属性table>tr>td

|  |
| --- |
| 1.table:之前学习过的样式，基本都可以使用  尺寸，边框，背景，字体，文本，内外边距  给table设置border,只设置最外面的大边框 |
| 2.td/th的样式  尺寸，边框，背景，字体，文本，内边距  外边距无效  vertical-align指定单元格数据的垂直对齐方式  取值：top/middle/bottom |

练习

|  |
| --- |
| 02\_ex  创建4\*4表格，宽高400px。内容随意  1.设置每个单元格的尺寸为100px\*100px  2.设置每个单元格边框1px solid #000  3.尝试给每个单元格设置上外边距20px.上内边距20px； |

表格是一种特殊的表现方式

|  |
| --- |
| 表格实际尺寸是根据内容数据的多少而决定的  单元格小，内容多，自动撑开  内容少，单元格就按照设置的尺寸来展示 |

### 2.表格特有的样式属性

①边框合并

|  |
| --- |
| border-collapse:  取值 ：1.separate 默认值，边框分离模式  2.collapse;边框合并模式 |

②边框边距

|  |
| --- |
| 向设置边框边距，必须保证边框是分离状态  border-collapse:separate ;  属性border-spacing:  取值：1.只有一个值 设置水平和垂直边框的 外边距  2.两个值 第一个值设置水平 第二个值设置垂直 |

③标题位置

|  |
| --- |
| caption-side:  取值：top 默认值  bottom |

④设置表格的显示规则

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 用来确定如何布局一张表格  table-layout  取值 1.auto 默认值 自动布局表格  列的尺寸实际使用内容决定的，内容比尺寸大，按照  内容大小显示。内容比尺寸小，按照尺寸显示  2.fixed 固定表格布局，列的尺寸以设置的为准  自动布局与固定布局   |  |  | | --- | --- | | 自动布局 | 固定布局 | | 单元格的大小自动适应内容 | 单元格大小取决于设置的值 | | 表格复杂时，加载速度较慢 | 任何情况下，都会加速加载表格(优点) | | 比较灵活(优点) | 固定布局不够灵活 | | 适用于不确定每列大小的，并且不复杂的表格 | 适用于能够确定每列尺寸大小的表格 | |

练习



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

1.什么是定位

|  |
| --- |
| 改变元素在页面中的位置 |

### 2.定位的分类

|  |
| --- |
| 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 |

①普通流定位

|  |
| --- |
| 又称为默认文档流定位  1.每个元素在页面上都有自己的空间  2.每个元素都是从父元素的左上角开始渲染（显示）  3.块级元素按照从上到下逐个排列，每个元素单独成行  4.行内元素是多个元素在一行中显示，从左往右顺序排列，显示不下换行 |

②浮动定位

|  |
| --- |
| 让块级元素横向显示  float：  取值：left 让元素浮动后，在当前行停靠在父元素的左侧，或者挨着左侧已浮动元素  right让元素浮动后，在当前行停靠在父元素的右侧，或者挨着右侧已浮动元素  none 默认值，不浮动  浮动的特点  1.元素一旦浮动，脱离文档流(不占页面空间，后面未浮动元素会上前补位)  2.浮动元素会在当前行，停靠在父元素的左/右边，或者停靠在其他已浮动元素的边缘  3.当父元素横向显示不下所有浮动元素时，最后显示不下的，会自动换行  4.浮动用于解决，多个块级元素在同一行显示的问题 |

③浮动定位引发的特殊情况

|  |
| --- |
| 1.浮动元素占位的问题  当元素显示不下所有浮动元素时，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，  导致被挤下去的浮动元素，需要让开位置，在更下面的地方显示 |
| 2.元素一旦浮动，如果元素未定义宽度，那么元素浮动之后的宽度将以内容为准 |
| 3.元素一旦浮动，都会变成块级元素  允许设置尺寸，可以设置垂直外边距 |
| 4.文本，行内元素，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 |

④清除浮动

|  |
| --- |
| 清除之前的浮动元素给自己带来的影响  由于元素浮动之后，会脱离文档流，会让后续不浮动的元素上前补位.  如果后续元素不想上前补位，需要对此元素设置清除浮动  clear:  取值：1.left 清除左浮动对我的影响  2.right 清除右浮动对我的影响  3.both 清除左右浮动对我的影响  4.none 不清除影响 |

⑤高度坍塌

|  |
| --- |
| 什么叫高度坍塌  1.父元素不写高，靠子元素撑起高度  2.所有子元素都浮动  那么所有子元素都脱离文档流，父元素认为自己内部没有元素了  所以父元素就没有高度了  解决方案：  1.父元素也浮动，弊端，影响父元素后面的非浮动元素  2.直接给父元素写高度 弊端，不是每次都能知道具体高是多少  3.overflow：hidden 弊端，会让真正要溢出不能显示  4.在父元素中追加一个块级元素，这个块级元素，没有内容，不写高。只写clear:both。就可以让父元素在文档流中找到内容的高度  解决，高度坍塌问题 |

作业

1.掌握css编写思路



|  |
| --- |
| 1.宽高，大体位置  2.边框，背景  3.文字所有  4.微调 |

2.今天关于table和浮动的所有demo

3.回家安装WebStorm10.zip，并且保证注册成功

day12

## 一.元素的显示方式

|  |
| --- |
| display  取值： 1.block 让元素的表现和块级一致  2.inline 与行内元素一致  3.inline-block 与行内块元素一致  4.table 与table一致  5.none 不显示元素，隐藏  块级：独占一行，可以设置尺寸，上下外边距有效  行内：共用一行，尺寸无效，上下外边距无效  行内块：共用一行，可以设置尺寸，上下外边距有效  table：独占一行，可以设置尺寸，尺寸以内容为主 |

## 二.显示效果

|  |
| --- |
| visibility  取值：1.visible 默认值，可见  2.hidden 隐藏不可见  问题：visibility:hidden和display:none区别  visibility:hidden；隐藏，元素不脱离文档流，在当前页面不可见，但是占据位置 看不见但是占位置  display:none；隐藏，元素脱离文档流，隐藏后不占位置，后面元素上前补位 看不见也不占位置 |

## 三.透明度

|  |
| --- |
| transparent----rgba(0,0,0,0)  opacity  取值：0~1 值越小越透明  问题：opacity和rgba的区别  rgba只会改变当前颜色的透明度  opacity，元素内部只要元素相关的颜色都会跟着透明(包括后代元素) |

## 四.垂直对齐方式

|  |
| --- |
| vertical-align**（\*\*\*\*\*\*\*\*考试必考\*\*\*\*\*\*\*）**  使用场合  1.表格中 td/th  取值 top/middle/bottom  2.img与文字的排版  改变的是img与前后文本的对齐方式  取值 top/middle/bottom/baseline基线  通常，会将所有的图片与文字的对齐方式，改为非基线对齐的方式  关于vertical-align 只在3种情况下可以使用 ①在table中使用 vertical-align：top/middle/bottom 让内部的文本垂直对齐 ②给行内块设置 vertical-align：top/middle(默认值)/bottom 控制当前行内元素，之前/后，文本，行内，行内块与本元素的垂直对齐方式 ③给img设置 vertical-align：top/middle/bottom/baseline(基线，默认值) 控制当前img，之前/后，文本，行内，行内块与本img的垂直对齐方式 通常项目中，会把img的vertical-align设置为非基线 |

## 五.光标的设置

|  |
| --- |
| cursor  取值：1.default 箭头  2.pointer 小手  3.crosshair 十字  4.text 文本输入的I  5.wait 等待  6.help 帮助  7.crosshair截图光标 十字  光标颜色：caret-color：颜色； |

## 六.列表的样式

|  |
| --- |
| ul的样式  1.列表项标识  list-style-type:  取值 none/disc/circle/square  2.列表项标识，设置为图片(图片要小)  list-style-image:url(路径);  3.列表项标识的位置  list-style-position:inside(li内部)/outside(默认值 li外部)  设置列表项是在li的内部还是外部  4.简写方式  list-style:type image position;  项目中最常用的写法list-style:none;  去掉列表标识 |

## 七.定位---相对，绝对，固定定位（\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*）

|  |
| --- |
| position  取值：static 默认，静态(默认文档流定位)  relative 相对定位  absolute 绝对定位  fixed 固定定位  当一个元素，被position修饰，并且取值为relative/absolute/fixed其中一种时，这个元素被称为已定位元素  已定位元素，解锁了4个偏移属性  top + 往下 - 往上  right + 往左 -往右  bottom + 往上 -往下  left +往右 -往左  偏移属性，定义了元素距离某一个方向移动了多少距离 |

1.相对位移

|  |
| --- |
| 相对定位，相对自己原来位置偏移某个距离  position:relative;配合4个偏移属性使用  特点：1.不脱离文档流。后面元素不会上前补位  2.相对定位，如果不写偏移量，元素效果与没写定位是一样的.  只设置相对定位，不设置偏移属性值，这个元素就出现在原始位置，同时不影响周围元素  使用场合  1.自身位置的微调 但是会出现堆叠现象  2.作为绝对定位的祖先级元素 |

2.绝对定位

|  |
| --- |
| position:absolute；配合偏移量使用  特点：  1.绝对定位，脱离文档流，元素不占页面空间，后面元素上前补位。  2.绝对定位的元素，会相对于  "离自己最近的""已定位的""祖先元素"的左上角实现位置的偏移  如果没有 "已定位的""祖先元素"，则相对于body的左上角实现位置偏移  最适合做“最近的”，“祖先级”，“已定位元素”是相对定位relative  3.绝对定位元素会变成块级元素  4.绝对定位元素，如果不写宽，定义之后，宽靠内容撑开 |
| 脱离文档流(脱离文档流会完成4件事)  ①不占页面空间  ②后续元素上前补位  ③没有设置宽度的元素，脱离文档流之后，宽度靠内容撑开  ④任何元素脱离文档流之后，都可以设置宽高，都可以设置上下外边距 |

3.固定定位

|  |
| --- |
| position:fixed;配合偏移量使用  将元素固定在页面的某个位置，不会随着滚动条发生位移变化  特点：1.脱离文档流  2.元素变为块级元素  3.不写宽的会被内容撑开  4.相对body做位置的初始化 |

4.堆叠顺序

|  |
| --- |
| 特点：  1.默认堆叠顺序:后定位的元素，堆叠顺序高 默认堆叠在0~1，取不到1  2.定位的脱离文档流，和浮动的脱离文档流，不是一个体系  浮动和定位的堆叠对比  浮动的堆叠在-1~0之间，取不到0，还不能用z-index设置  3.使用z-index设置堆叠顺序  z-index:无单位数字 一般情况 1~1000  4.堆叠顺序，只对已定位元素有效  5.堆叠顺序，对父子级无效。子元素永远在父元素上面显示 |

# CSS3 CORE

## 一.复杂选择器

### 1.兄弟选择器

|  |
| --- |
| 兄弟元素：具有相同父元素的平级元素  兄弟选择器，只能找弟弟，不能找哥哥 |

①相邻兄弟选择器

|  |
| --- |
| 获取紧紧挨在某元素后面的兄弟元素  选择器1+选择器2{} |

②通用兄弟选择器

|  |
| --- |
| 获取某元素后，所有符合要求的兄弟元素  选择器1~选择器2{}（效率低）  项目中的应用场景  一组相同的元素，除了老大以外，其他兄弟都选中  使用相邻兄弟选择器 选择器1+选择器2{} |

练习

|  |
| --- |
| <h3>小白兔</h3>  <div>小了白了兔</div>  <div>白了又了白</div>  <div>爱了吃了萝了卜了和了青了菜</div>  <div>蹦了蹦了跳了跳了真了可了爱</div>  标题与正文之间间距30px  正文每行之间间距10px |

### 2.属性选择器

|  |
| --- |
| id class style title name value width.......  允许通过元素所附带的属性，及其值来匹配页面元素，很精准 |

①[attr] attr代表任意属性

|  |
| --- |
| 匹配页面中所有带attr这个属性的元素  [id]{} [id][title]{} |

②elem[attr]

|  |
| --- |
| 匹配页面中所有带attr属性的elem元素  p[title]{} p[id][title][class].....{} |

③elem[attr=value]

|  |
| --- |
| 匹配页面中带attr属性，并且值为value的elem元素  div[title="woyouyeye"][class="c"]{} |

④模糊属性值匹配

|  |
| --- |
| [attr^=value]{ } 匹配attr的值以value开头的元素  [attr$=value]{ } 匹配attr的值以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 已经学过的伪类 |

①目标伪类(锚点)

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

练习

|  |
| --- |
| 09\_ex  点击菜名，显示图片  菜名放在ul>li>a  海鲜面 四季春饼 八大碗 |

作业：1.完成学子首页1楼



2.完成学子商城轮播图的静态页面



day13

## 一.复杂选择器

### 3.伪类选择器

①目标伪类target

②结构伪类

|  |
| --- |
| elem:first-child{} 代表两个条件  匹配elem的父元素的第一个儿子（elem的大哥）  这个大哥必须是elem元素  elem:last-child{}  匹配elem的父元素的最后一个儿子(elem的小弟)  这个最小的弟弟必须是elem元素  elem:nth-child(n)  匹配elem的父元素的第n个儿子(n从1开始)  这个儿子也必须是elem元素 |

练习

|  |
| --- |
| 01\_ex 4\*4的表格 200px\*200px. 有边框，内容随意  需求：表格第一行背景为红色  表格最后一行最后一列的背景为蓝色  表格第三行第二列背景为黄色 |

③匹配空元素

|  |
| --- |
| :empty{}  空元素  空元素：没有文本，没有空格，没有其他子元素的元素 |

④:only-child

|  |
| --- |
| 匹配，当前元素是其父元素的唯一的子元素 |

⑤否定伪类

|  |
| --- |
| :not(selector)  /\*除了第一个a标签，其它a的字体都变成黄色 \*/  a:not(:first-child){color:#ff0;}  /\*除了第三个a标签，其它a的字体都变成蓝\*/  a:not(:nth-child(3)){color:#f0f;}  /\*除了类名为checked的其他内容\*/ 括号内不加冒号，加了报错  :not(.checked){color:#000000} |

## 4.伪元素选择器

①内容伪元素

|  |
| --- |
| 1::first-letter{ } 或 :first-letter{}  匹配首字符的样式  2::first-line{ } 或 :first-line{}  匹配首行的样式  当首行和首字符冲突的时候，以首字符为准  3.::selection{} 必须是两个 ::  匹配选中部分的文字样式  注意：只能改变字体颜色和背景颜色 |

②伪元素选择器，内容生成

|  |
| --- |
| 使用css，添加httml元素，称之为伪元素内容生成  :before 或者 ::before  匹配到某个元素的内容区域的最前面，  添加一个内容，content：添加的文本或者图片  display：是设置这个添加的内容的显示规则  ::after 或者 :after  匹配到某个元素的内容区域的最后面，添加一个内容  注意：content中，只能添加文本或者图片  我们可以理解为在内容区域中，最前或者最后面，添加了一个元素。这个元素的显示方式，由display来设定  默认情况下：添加了一个行内元素，记双冒号 |

③伪元素内容添加，可以解决的问题

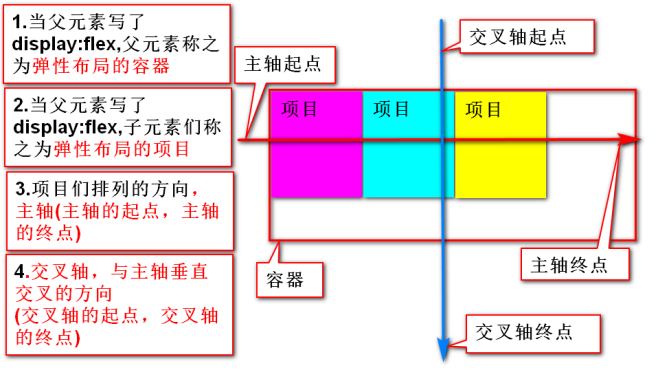
|  |
| --- |
| 1.外边距溢出  #parent::before{  content:"";  display:table;  }  2.思考，伪元素内容添加，解决高度坍塌？  #d1:after{  content:"";  display:block;  clear:both;  } |

## 二.弹性布局（重点\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*）

### 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  为布局提供很大灵活性 |

### 2.弹性布局相关的概念和名词解释



|  |
| --- |
| 1.容器  要发生弹性布局的子元素，他们的父元素称之为容器  容器要设置属性display:flex;  这个元素本身不是弹性布局，儿子们按照弹性布局的方式来布局  2.项目  要发生弹性布局的子元素们，称之为项目  就是设置了display:flex那个元素的，子元素们  3.主轴  项目们在容器中排列的方向，就是主轴 4条主轴  如果项目横向排列，x轴就是主轴  如果项目纵向排列，y轴就是主轴  项目们的排列顺序，靠主轴的起点和主轴的终点来定义  4.交叉轴  与主轴垂直相交的一条轴，叫做交叉轴  项目们在交叉轴上的对齐方式，是交叉轴的起点和终点 |

### 3.语法

|  |
| --- |
| display，写在父元素中  取值：1.flex 将块级元素（父元素）设置为容器  2.inline-flex 将行内元素设置为容器  特点：  1.弹性项目，默认x是主轴，主轴起点在左侧  (块级元素横向排列的第二个解决方案)  2.项目的float/clear/text-align/vertical-align属性失效  3.每个项目可以自由的设置尺寸 |

### 4.容器的属性

①主轴的方向

|  |
| --- |
| flex-direction:  取值： row默认值，主轴是x轴，主轴起点是左端 （起点是箭头的左边）  row-reverse,主轴是x轴，主轴起点是右端 （起点是箭头的右边）  column 主轴是y轴，主轴起点是顶端 （ …… 上边）  column-reverse 主轴是y轴，主轴起点是底部 （…… 下边） |

②设置项目换行

|  |
| --- |
| flex-wrap:  取值 nowrap 默认值，容器空间不够，也不换行，项目自动缩小  wrap 空间不够就换行  wrap-reverse 换行并反转 |

③主轴方向，项目换行的缩写

|  |
| --- |
| flex-flow:  取值 direction wrap |

④定义项目在主轴上的对齐方式

|  |
| --- |
| justify-content:  取值 1.flex-start 默认值，以主轴起点对齐  2.flex-end 以主轴终点对齐  3.center 在主轴上居中对齐  4.space-between 主轴两端对齐，两端无空白  5.space-around 每个间距大小相同 |

⑤项目在交叉轴上的对齐

|  |
| --- |
| align-items:  取值1.flex-start 交叉轴起点对齐  2.flex-end 交叉轴终点对齐  3.center 交叉轴居中对齐  4.baseline 交叉轴基线对齐 以项目内部文本的基线对齐  5.stretch 前提，项目不写高，占满交叉轴上所有的空间 |

练习

|  |
| --- |
| 06\_ex,ul>li 配合弹性布局完成 |

## 5.项目的属性

①order

|  |
| --- |
| 定义项目的排列顺序  取值，无单位整数，值越小，越靠近主轴起点 |

②flex-grow

|  |
| --- |
| 定义项目的放大比例  如果容器有足够大的剩余空间，项目将按比例方大  设置的值，是把剩余空间分成n份，分别增加n份  取值：无单位的数字 默认值0，不放大 |

③flex-shrink

|  |
| --- |
| 定义项目缩小的比例，容器空间不够时，项目该如何缩小  取值，无单位数字 默认值1. 取值越大，缩小比例越大。0，不缩 |

④align-self

|  |
| --- |
| 设置此项目在交叉轴上的对齐方式，不影响其它项目  取值：flex-start  flex-end  center  baseline  stretch  auto 使用容器定义的align-items的值 |
| ⑤flex-basis  设置项目在主轴方向上的基本尺寸  优先级大于width/height  问题，一般img不使用这个属性，有可能会出bug  flex-basis:px为单位的数字  % |
| ⑥简写方式  flex:grow shrink basis;  flex:0 0 25%;  **图片在弹性里面基本都包一个div，防止图片本身因弹性而变形** |

## 三.CSS hack

|  |
| --- |
| div{  background:-webkit-linear-gradient(....);  background:-o-linear-gradient(....);  background:-ms-linear-gradient(....);  background:-moz-linear-gradient(....);  }  由于不同的浏览器对css的解析认知不同，会导致同一份css在不同浏览器中生成页面效果不同  面对这种情况，开发人员需要针对不同浏览器写不同的css  这个行为，就叫做css hack  tmooc有css hack相关拓展视频 |

### 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转换。只在x轴和y轴发生的转换效果  3D转换。增加了z轴的转换效果。3D是模拟出来的 |

2.转换的属性

|  |
| --- |
| transform: 取值 1.none 默认值，无任何转换效果  2.transform-function 转换函数  表示1个或者多个转换函数  如果是多个转换函数，每个函数之间用空格分开  transform:translate(400px) rotate(90deg);  学习转换，就是学习转换函数 |

作业1.使用弹性布局完成学子商场首页1F



作业2：复杂选择器，弹性布局的所有属性，单词和效果

作业3：确保webstrom10 确保可以正确使用(注册码的问题)

day14

## 四.转换

1.什么是转换

2.转化属性

|  |
| --- |
| transform：转换函数1 转换函数2 ........  学习转换，就是学习在转换函数 |

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)  取值个数： 2个值，原点在x轴和y轴上的位置  3个值，原点在x轴，y轴和z轴上的位置  默认值 center center |

### 4.2D转换

①位移（改变元素的位置）

|  |
| --- |
| transform:translate(参数)  参数：1.translate(x) 等同于translateX(x)  指定元素在x轴上的位移距离  + 往右 - 往左  2.translate(x,y)指定元素在x轴和y轴上的位移距离  x: + 往右 - 往左  y:+ 往下 - 往上  3.translateX(x)  4.translateY(y)  取值：1.px为单位的数字  2.% |

面试题

|  |
| --- |
| 一个div200px\*200px。使用css操作，如何在浏览器窗口不断变化的情况下，让该div永远在窗口中心显示 |

②缩放(改变元素的尺寸)

|  |
| --- |
| transform:scale(value)  取值 一个值 value>=1 x轴和y轴都放大的倍数  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180度  value<=-1 x轴和y轴放大，并反转180度  两个值 scale(x，y) 分别设置x轴和y轴的放大比例  scaleX(x) 单独设置x轴  scaleY(y) 单独设置y轴 |

③旋转(改变元素的角度)

|  |
| --- |
| transform:rotate(190deg)  取值： + 顺时针旋转 - 逆时针  练习：01\_ex  #d1 先旋转45deg,再位移300px  #d2 先位移300px，再旋转45deg  注意：旋转原点会影响效果  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 |

练习

|  |
| --- |
| 翻滚吧，亮亮  div200px\*200px,倒角50%，背景为亮亮照片  鼠标悬停时，该元素向右位移1000px，旋转1080deg |

④倾斜

|  |
| --- |
| 1.skew(x)等同于skewX(x)  让元素向着x轴发生倾斜，实际上改变的是y轴的角度  x:+ 逆时针倾斜  - 顺时针倾斜    2.skewY(y)  让元素向着y轴发生倾斜，实际上改变的是x轴的角度  y:+ x轴顺时针倾斜  - x轴逆时针倾斜 |

练习

|  |
| --- |
| 02\_ex 创建div 200\*200 背景，鼠标悬停时  该元素向右偏移200px,向下偏移200px,旋转135deg，用x轴倾斜45deng，放大1.5倍 |

### 5.3D转换---3d都是模拟

①透视距离

|  |
| --- |
| 模拟人的眼睛到3D转换元素之间的距离，透视距离不同，看到的效果不同  设置透视距离  perspective:距离，此属性要加载到3d转换元素的父元素上 |

②3D旋转

|  |
| --- |
| 属性：transform  取值：1.rotateX(xdeg)  以x轴为中心轴旋转，烤羊腿 老式爆米花机  2.rotateY(ydeg)  以y轴为中心轴旋转，旋转门，旋转木马，钢管舞  3.rotateZ(zdeg)  以z轴为中心轴旋转,风扇，风车，摩天轮  4.rotate3D(x,y,z,ndeg)  x,y,z取值0，代表这条轴不参与旋转  取值>0, 表示该轴参与旋转 |

## 五.过渡

1.什么是过渡

|  |
| --- |
| 让css属性的值，在一段时间内平缓变化的效果  哪些css属性，可以过渡  怎么设置一段时间是多少  变化速度 |

2.语法

①指定过渡的属性

|  |
| --- |
| transition-property  取值：直接写css的属性，多个属性值之间用空格分开  all 所有支持过渡的属性，都参与此次过渡效果  哪些属性支持过渡  1.颜色属性  2.大多数取值为具体的数字的属性  3.阴影  4.转换transform  5.visibility |

②指定过渡时长

|  |
| --- |
| transition-duration  指定用多长时间完成此次过渡操作  取值 ：s/ms为单位的数字 1s=1000ms |

③过渡时间曲线函数

|  |
| --- |
| transition-timing-function:  取值  1.ease 默认值，慢速开始，中间变快，慢速结束  2.linear 匀速  3.ease-in 慢慢开始，快速结束  4.ease-out 快速开始，慢慢结束  5.ease-in-out 慢速开始，慢速结束，中间先加速后减速 |

④过渡的延迟时间

|  |
| --- |
| transition-delay:  让过渡效果，延迟多少时间执行  取值 s/ms为单位的数字 |

⑤过渡代码的编写位置

|  |
| --- |
| 原始选择器中，过渡效果有去有回  :hover中，过渡效果有去无回 |

⑥简写方式

|  |
| --- |
| transition：property duration timing-function delay;  最简方式  transition:duration; |

练习：

|  |
| --- |
| 1.翻滚吧，亮亮  2.亮亮转圈圈  03\_ex 一个div,向右移动800px.并且旋转1080deg  14:58~15:13休息 |

## 六.动画

1.什么是动画

|  |
| --- |
| 元素从一种样式逐渐变为另一种样式  其实就是多个过渡效果放到一起 |

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.使用动画

①使用关键帧定义动画

|  |
| --- |
| @keyframes 动画名称{  /\*关键帧\*/  0%{样式}  ...  50%{样式}  ..  100%{样式}  }  ex:  @keyframes liangjump{  0%{transform:translate(0px,0px);}  20%{transform:translate(0px,200px);}  40%{transform:translate(0px,0px);}  60%{transform:translate(0px,150px);}  80%{transform:translate(0px,0px);}  100%{transform:translate(0px,110px);}  } |

②调用动画

|  |
| --- |
| 1.调用动画名称  animation-name:动画名称  2.设置动画执行时间  animation-duration: s/ms  3.设置动画的时间曲线函数  animation-timing-function  4.设置动画延迟播放  animation-delay:2s; |

练习

|  |
| --- |
| 05\_ex  div#d1  创建一个动画change  0% 背景色为蓝色，矩形  25% 背景色为黄色，圆形  50% 背景色为红色，矩形  75% 背景色为orange,圆形  100% pink 矩形 |

③动画的其它属性

|  |
| --- |
| 设置动画的播放次数  animation-iteration-count:  取值 1.具体次数，无单位数字  2. infinite 无限 |
| 设置动画的播放顺序  animation-direction:  取值：1.normal 默认 0%~100%  2.reverse 100%~0%  3.alternate 轮流播放，第一遍正向，第二遍逆向，... |
| 动画的简写方式  animation：name duration timing-function delay count direction  最简方式 animation:name duration; |
| 设置动画，播放前后的状态  animation-fill-mode  取值：1.backwards，动画播放之前的延迟时间内，显示第一帧  2.forwards，动画播放完成，保存在最后一帧  3.both,同时使用backwards和forwards  4.none 默认值，不填充 |

设置动画的播放状态

|  |
| --- |
| animation-play-state  取值：1.paused 暂停  2.running 播放 |

### 5.动画的兼容性

|  |
| --- |
| 如果要兼容低版本浏览器，需要在动画声明的时候加前缀  @keyframes 动画名称{}  @-webkit-keyframes 动画名称{}  @-o-keyframes 动画名称{}  @-moz-keyframes 动画名称{}  @-ms-keyframes 动画名称{} |

6.开发中使用的动画样式

|  |
| --- |
| 使用animate.css中预定义好动画 |

## 七.CSS优化

1.CSS优化的目的

|  |
| --- |
| 1.减少服务器压力  2.提升用户体验 |

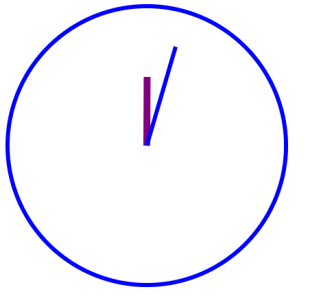
2.CSS优化原则

|  |
| --- |
| 尽量减少http的请求个数 .css .js .jpg  在页面顶部，引入css文件  将css和js文件放到外部独立的文件中 |

3.CSS代码优化

|  |
| --- |
| 1.合并样式（能简写，就不分开写.能写群组，就不单写）  2.缩小样式文件的大小（能重用的样式，尽量重用）  3.减少样式重写  4.避免出现空的href和空的src  5.选择更优的样式属性  6.代码压缩 |

**作业：1使用动画完成时钟**



**作业2：弹性布局完成**



**作业3：webstorm10 保证可以使用**

# BootStrap

day15

准备webstrom10,见视频

## 一.响应式布局(css3)

1.什么是响应式网页

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

2.响应式网页必须保证几件事情

|  |
| --- |
| 1.布局，不能固定元素宽度.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而改变  3.媒体查询技术(css3的技术) |

3.如何测试响应式网页

|  |
| --- |
| 1.使用真实设备测试  好处：真是可靠  缺点：成本高，测试任务巨大  2.使用第三方测试软件  好处：不需要太多真实设备，测试方便  坏处：测试效果有限，有待进一步验证  3.使用chrome等浏览器自带的模拟软件  好处：简单方便  坏处：测试效果十分有限，需要进一步验证 |

4.编写响应式布局(重点\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*)

①手机适配,设置视口

|  |
| --- |
| 只有移动端设备，需要设置视口  <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=0">  viewport 视口  width=device-width 视口宽度等于设备宽度  initial-scale=1.0 设置视口宽度能否缩放 1.0不能缩放  maximum-scale=1.0 允许视口缩放的最大倍率 1.0不缩放  user-scalable=0 是否允许用户手动缩放 0不能  最简洁的写法  <meta name="viewport" content="width=device-width, initial-scale=1"> |

②所有内容/文字/图片都尽量使用相对尺寸，不要使用绝对数字

③流式布局+弹性布局+媒体查询，组成响应式布局

④媒体查询 CSS3 Media Query

|  |
| --- |
| 媒体查询是写响应式必须具备的技术  Media：媒体，不同的用于浏览网页的设备  设备：screen（pc/pad/phone）  TV  print  Media Query:可以自动的根据当前浏览设备不同(尺寸，方向，解析度。。。)，有选择性的，执行一部分代码，忽略其他代码 |

使用媒体查询

|  |
| --- |
| @media 条件1 and 条件2 and 条件3{  #content{width:50%}  #d1{color:red}  }  @media 条件4 and 条件5 and 条件6{  #content{width:100%}  #d1{color:green}  } |
| 媒体查询的条件  1.设备 screen （pc/pad/phone）  TV  print  2.屏幕尺寸，约定俗成把所有屏幕尺寸分为5类  超大屏 xl v>=1200px min-width:1200px  大屏 lg 992px<=v<1200px min-width:992px and max-width:1199.99px  中屏 md 768px<=v<992px  小屏 sm 576px<=v<768px  超小屏 xs v<576px |
| 响应式标准写法 |
| 不带超小屏，并向上兼容的写法，一定要注意编写顺序 |

二.BootStrap

|  |
| --- |
| boot封装了很多常用样式，以类名的形式让我们调用  但是，真正项目，很多样式还是需要我们手动开发和媒体查询  学会查boot文档的能力  boot内容分为5部分  1.如何使用bootstrap  2.全局样式  3.组件  4.js插件  5.sass  6.boot项目 |

### 三.如何使用bootstrap

|  |
| --- |
| 导入boot文件  *<!--设置视口-->* <**meta name="viewport" content="width=device-width,initial-scale=1"**/> *<!-- 导入css-->* <**link rel="stylesheet" href="../css/bootstrap.css"**/> *<!-- 导入3个js,有顺序要求-->* <**script src="../js/jquery.min.js"**></**script**> <**script src="../js/popper.min.js"**></**script**> <**script src="../js/bootstrap.min.js"**></**script**>  制作boot模板  file--->new--->edit Template file |

### 四.全局样式

|  |
| --- |
| .container 定宽容器，在每种尺寸下，都定义了最大宽度  .container-fluid 变宽容器，每种尺寸都是100%  boot支持4种屏幕尺寸 xl--lg--md--sm  boot中rem，默认一个rem是16px |

### 1.按钮相关的class

|  |
| --- |
| 基本类 .btn 所有的按钮，都需要添加这个类  按钮颜色  btn-success 绿色  btn-danger 红色  btn-warning 黄色  btn-info 青色  btn-primary 蓝色  btn-secondary 灰色  btn-dark 深色  btn-light 浅色  btn-link 链接  不同边框的按钮  btn-outline-danger/success/warning......  不同大小的按钮  btn-lg  btn-sm  btn-block |

### 2.图片相关

|  |
| --- |
| rounded 圆角 0.25rem  rounded-circle 圆形  img-thumbnail 缩略图，有一圈白色的边(padding)  img-fluid 响应式图片，图片可以缩放，但是不会超过原始大小 |

### 3.文本相关的class

|  |
| --- |
| .h1~.h6 字号大小，同h1标签到h6标签  text-danger/primary.......文本颜色  text-uppercase/lowercase/capitalize首字母大写  font-weight-bold/light/normal 字体 粗细正常  text-left/right/center/justify 文本对齐  text-\*-left/right/center 响应式 \*:sm/md/lg/xl |

### 4.列表相关class

|  |
| --- |
| list-unstyled 取点，左内边距清0  ul.list-group>li.list-group-item 创建列表组和列表项  激活项 .list-group-item.active 两个类都引用  禁用项 .list-group-item.disabled 两个类都引用  列表项颜色 list-group-item-danger/warning...... |

### 5.表格相关的样式

|  |
| --- |
| .table 基本量，对table本身和table的后代布局  .table-bordered 一个有边框的边框，后代td、th也添加边框  .table-danger/warning....表格颜色  .table-striped 隔行变色，奇数行变色  .table-hover 带鼠标悬停效果  .table-responsive-xl/lg/md/sm 响应式表格，写在table的父元素上 |

### 6.辅助类

### ①边框

|  |
| --- |
| 基本类  border/border-0/ 定义边框/取消边框  border-left/top/bottom/right 单边定义  border-left-0 单边的取消  border-danger/warning/info.... 边框颜色 |

### ②浮动

|  |
| --- |
| float-left/right/none  float-sm/md/lg/xl-left响应式的浮动  clearfix 解决高度坍塌，写在父元素 |

### ③显示

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

④背景颜色

|  |
| --- |
| bg-danger/warning/success/primary.... |

### ⑤圆角

|  |
| --- |
| rounded/rounded-0 设置圆角/圆角清0  一个方向的圆角设置  rounded-top/right/bottom/left  rounded-circle 50% |

### ⑥内外边距

|  |
| --- |
| m-n mt/mr/mb/ml-n mx-n左右外边距 my-n上下  p-n pt/pr/pb/pl-n px-n 左右内边距 py-n上下  n:0---取消,  1--0.25rem,  2---0.5rem  3--1rem，  4---1.5rem,  5---3rem  上面所有的都有响应式  m-\*-n  m:m/mt/mb/ml/mr/mx/my \*:xl/lg/md/sm n:0~5  p-\*-n  p:p/pt/pb/pl/pr/px/py \*:xl/lg/md/sm n:0~5 |

day16

四.全局css

7.栅格布局 （重点\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*）

①web页面中的布局方式，一般有三种

|  |  |  |
| --- | --- | --- |
| table布局 | div+css | Boot的栅格布局 |
| 简单，易控制 | 语义正确 渲染效率高 | 简单，易控制。语义正确，渲染效率高，支持响应式 |
| 语言错误 渲染效率低 | 控制起来很麻烦 | 不适用于复杂页面 |

②栅格的使用

|  |
| --- |
| 1.栅格必须在container或者container-fluid中  2.每一行都是一个row(弹性，主轴是x轴，可以换行，  左右-15px外边距)  3.每一个col自带左右15内边距 |

③响应式栅格布局

|  |
| --- |
| col-\*-n  \*:lg/md/sm/xl  n:1~12  在不同的屏幕下，执行不同的列数  <**div class="row"**>  <**div class="col-lg-2 col-md-3 col-sm-12"**>111</**div**>  <**div class="col-lg-8 col-md-9 col-sm-12"**>222</**div**>  <**div class="col-lg-2 d-lg-block d-md-none col-sm-12"**>333</**div**> </**div**> |

④不同屏幕尺寸，向上兼容

|  |
| --- |
| 只写小屏，在小屏，中屏，大屏，超大屏，都会应用  只写中屏，在中屏，大屏，超大屏，都会应用  只写大屏，大屏，超大屏，都会应用  只写超大屏，只在超大屏应用 |

⑤使用.col这个类

|  |
| --- |
| 使用.col,不添加数字，自动平局分配占地宽度  每一个col平局分配空间，col的个数可以超过12个  使用场合：  当一行内，元素需要横向现实，元素个数不能被12整除  每个元素占地空间一样 |

⑥列偏移

|  |
| --- |
| offset-\*-n 源码就是在操作左外边距 margin-left  \*:xl/lg/md/sm  n:0~11 |

⑦栅格嵌套

|  |
| --- |
| 在一个.col中再写一个row |

### 五.弹性布局

|  |
| --- |
| d-block/inline/inline-block/none display的取值  d-\*-block.....响应式 \*：xl/lg/md/sm  想使用弹性布局，需要在父元素上添加d-flex/d-inline-flex  创建弹性容器 |

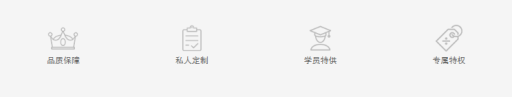
### 1.主轴方向

|  |
| --- |
| flex-row/column/row-reverse/column-reverse  响应式设置  flex-\*-row/column/row-reverse/column-reverse  \*:xl/lg/md/sm |

### 2.主轴对齐方式

|  |
| --- |
| justify-content-start/center/end/between/around  响应式设置  justify-content-\*-start/center/end/between/around  \*：xl/lg/md/sm |

练习 ul>li boot完成



## 六.表单相关的class

### 1.表单元素的排列放向

|  |
| --- |
| form-group 堆叠表单，内部控件垂直显示  form-inline 内联表单，内部控件横向显示(弹性，x轴) |

### 2.表单控件元素样式

|  |
| --- |
| form-control 对于text,password 基本类  块级，w100 字体 背景颜色 边框 圆角 过渡  col-form-label 设置文本框上下内边距  col-form-label-lg 上下内边距比较大  col-form-label-sm 上下内边距比较小  checkbox的基本类  form-check-input,绝对定位  要配合父元素的 form-check使用  form-text 块级，有上外边距 15:06~15:21休息 |

练习：



|  |
| --- |
| form-inline可以代替row作为栅格行  form-inline没有左右-15px的外边距 |

## 七.组件

|  |
| --- |
| 网页中有一些复杂的特殊效果，之前必须使用js,dom,jq  boot为我们封装好了很多常用的组件，我们可以直接使用。  方便，快捷。但是如果需要个性化的处理，比较繁琐。  boot中，大量使用元素的自定义属性，来调用方法 |

### 1.下拉菜单

|  |
| --- |
| 基本结构  <**div class="dropdown"**><**button class=" dropdown-toggle"**  **data-toggle="dropdown"**>下拉菜单</**button**>  <**ul class="dropdown-menu"**> </**ul**> </**div**>  div.dropdown 父级，相对定位  button.dropdown-toggle 画向下小三角  ul.dropdown-menu display:none  **data-toggle="dropdown"** 自定义属性的作用  1.当button被点击的时候激活事件  2.自定义属性的值，决定激活哪一个事件 |

### 2.按钮组

|  |
| --- |
| <div class="btn-group">  <button></button>  .....  </div>  外层div的class有以下几个值  btn-group 横向按钮组  btn-group-vertical 纵向按钮组  可以添加 btn-group-lg/sm 调整按钮组的大小 |

### 3.信息提示框

|  |
| --- |
| <div class="alert alert-danger alert-dismissible">  <span class="close" data-dismiss="alert">  &times;</span>  提示的信息  </div>  alert 基本类  alert-danger/warning/.....提示框颜色  alert-dismissible 为了修饰内部span.close  .close 右浮动  data-dismiss="alert" 自定义属性，事件使用alert的方式执行 |

### 4.导航

①水平导航

|  |
| --- |
| 基本结构 ul.nav>li.nav-item>a.nav-link  <ul class="nav nav-justified">  <li class="nav-item">  <a class="nav-link"></a>  </li>  </ul>  ul.nav 定义了弹性布局，主轴为x  那么之前学习的所有的弹性布局的样式都可以用  .nav-justified>.nav-item 要同时设置，让元素等宽显示  a.nav-link 块级，设置了hover，focus等样式 |

### ②选项卡导航

|  |
| --- |
| 基本结构  上层是选项卡  ul.nav.nav-tab>li.nav-item>a.nav-link  下层内容  div.tab-content>div.pane  ul.nav 变成选项卡添加类 nav-tab  nav-tab设置自己的孩子 nav-item nav-link  a标签添加自定义属性 data-toggle="tab"  以tab的方式激活事件  绑定目标元素 a的href属性指向下面内容的id值  div.tab-content>div.tab-pane配合使用  让div.tab-pane都是是display:none  然后div.tab-pane要写id，让a标签绑定 |

|  |
| --- |
| 一个事件激活，关注几件事情  1.激活这个事件的元素是啥 <a>  2.这个事件以什么方式激活 tab  3.事件激活后，改变的目标元素是谁 |

#### 5.导航栏

|  |
| --- |
| <**h1**>导航栏</**h1**> <**div class="navbar navbar-expand-lg"**>  <**ul class="navbar-nav"**>  <**li class="nav-item"**><**a class="nav-link" href=""**>长的丑</**a**></**li**>  </**ul**> </**div**>  最外层div.navbar.navbar-expand-xl/lg/md/sm  .navbar-expand-xl/lg/md/sm作用，作为子元素ul.navbar-nav的选择器条件，让ul.navbar-nav的主轴方向从y轴变成x轴(ul.navbar-nav原本主轴方向是y轴)    响应式的导航栏，以.navbar-expand-lg为例  超大屏，大屏，li横向显示  中屏，小屏，li纵向显示  div.navbar-expand-\* 配合 ul.navbar-nav可以让导航栏在不同屏幕下，横向或者纵向显示 |

### 6.折叠效果

|  |
| --- |
| <**button data-target="#demo" data-toggle="collapse" class="btn btn-info"**>折叠</**button**> <**div id="demo" class="collapse"**>Lorem  折叠，需要折叠内容添加.collapse--->display:none  折叠事件 data-toggle="collapse"  折叠目标  如果使用a,就可以使用href=""  如果使用button,则使用data-target="" |

### 7.卡片

|  |
| --- |
| <**div class="card"**>  <**div class="card-header"**></**div**>  <**div class="card-body"**></**div**>  <**div class="card-footer"**></**div**> </**div**> |

### 8.卡片+折叠=手风琴

|  |
| --- |
| <div id="parent">  <div class="card">  <div class="card-header">  <a data-toggle="collapse"  href="#one" class="card-link">折叠</a>  </div>  <div id="one" data-parent="#parent"  class="collapse">  <div class="card-body">内容</div>  </div>  </div>  </div>  总结：  1. .collapse不能和.card-body在同一个标签上  不然会发生折叠/显示的卡顿  解决方案 div.collapse>div.card-body  2.多个折叠部分，可以同时打开，而不是开启一个，其他的关闭  解决方案，在所有的card外部，添加一个父元素div#parent  在所有的div.collapse上添加自定义属性  data-parent="#parent"  这样就可以保证，在外层div中，只存在一个折叠区域被打开 |

练习1：



练习2：响应式导航栏，中屏，小屏 垂直排列



### 9.折叠导航栏

|  |
| --- |
| 1.div.navbar-expand-sm和.anvabar-toggler的关系和作用    2.在外层div中.navbar-dark/light的作用  这个类对本div没有任何修饰  修饰的是后代元素的字体颜色  3..navbar-collapse的作用  .navbar-expand-sm .navbar-collapse配合  让ul的父级div在sm/md/lg/xl下显示  在xs下，ul的父级div执行.collapse的样式，隐藏  4.小总结 .navbar-expand-sm决定了几件事？  1.按钮在sm以上隐藏，在sm以下显示  2.ul在sm以上横向显示，在sm以下垂直显示  3.折叠的div在sm以上显示，在sm以下隐藏 |

### 10.媒体对象

|  |
| --- |
| boot提供了很多类，来处理媒体相关的对象  图片 影音  <div class="media">  <img>  <div class="media-body"><div>  </div> |

### 11.焦点轮播图

|  |
| --- |
| 1.轮播图结构  div.carousel data-ride="carousel" 相对定位  >div.carousel-inner w100 溢出隐藏  >div.carousel-item display:none  >img  2.左右箭头  a.carousel-control-prev/next 宽度，背景，位置  >span. carousel-control-prev/next-icon 左右箭头  需要重写.carousel-control-prev/next中的宽高，位置  .**carousel-control-prev**, .**carousel-control-next**{  **width**:4%;**height**: 20%;  **background**:**#aaa**;  **border-radius**: 0.25**rem**;  **top**:40%; }[查看详情](http://www.codeboy.com/product_details.html?lid=1)  a标签要添加自定义属性 data-slide="prev/next"  事件的目标是最外层div#demo href="#demo"  3.轮播指示器  ul.carousel-indicators 绝对定位，横向弹性  >li 宽高，背景色，外边距  我们需要重写li的样式  .**carousel-indicators li**{  **width**: 0.75**rem**;**height**:0.75**rem**;  **background-color**:**#fff**;  **border-radius**: 50%;  **margin**:0 4**px**; }  如果让li有被选中的状态，需要第一个li添加.active  我们需要重写.active .**carousel-indicators** .**active**{  **background-color**:**#0aa1ed**; }  点击li激活事件  data-slide-to="图片下标" 下标从0开始  事件改变的目标  data-target="#demo" 整个轮播最外层div的id |

### 12.模态框

|  |
| --- |
| 覆盖在父窗体上的一个子窗体，用于与用户做一些互动  div.modal>div.modal-dialog>div.modal-content  >div.modal-header+div.modal-body+div.modal-footer  关闭的事件  button data-dismiss="modal" |

## 八.其它组件

### 1.徽章

|  |
| --- |
| 跟小按钮十分类似  基本类 badge  颜色 badge-danger/warning/info...........  徽章样式 badge-pill |

### 2.巨幕

|  |
| --- |
| 巨大的内边距，有背景颜色，有圆角  div.jumbotron |

### 3.分页条

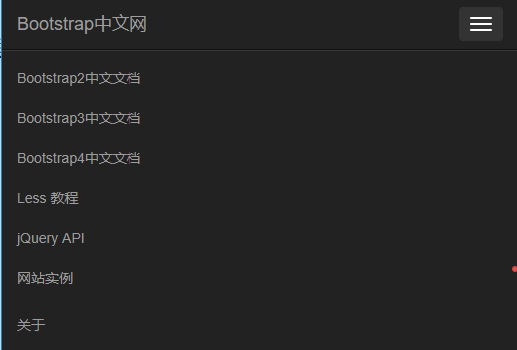
|  |
| --- |
| ul.pagination>li.page-item>a.page-link  li的修饰类 .active激活  .disabled禁用 |

作业

1.重新完成手风琴

2.重新完成焦点轮播图

3.折叠导航栏



day18

## 八.其他组件

### 4.面包屑导航

|  |
| --- |
| 常用于有路径关系的导航  ul.breadcrumb>li.breadcrumb-item  每个li中间连接的符号，需要重写  .**breadcrumb-item** + .**breadcrumb-item**::**before**{  **content**: **'>'**; } |

5.进度条

①普通进度条

|  |
| --- |
| div.progress>div.progress-bar.w-50.bg-danger  父div是进度条的槽  子div是进度条，子div的宽度，是进度条的进度  子div背景，是进度条的颜色 |

②带条纹的进度条

|  |
| --- |
| 在子div添加.**progress-bar-striped** |

③带动画的条纹进度条

|  |
| --- |
| <**div class="progress-bar w-100 bg-secondary progress-bar-striped progress-bar-animated"**></**div**> |

## BOOT定制:scss

|  |
| --- |
| **bootStrap最重要：响应式+栅格布局+scss** |

### 1.CSS有很多缺点

|  |
| --- |
| 语法不够强大，没有变量，没有合理的样式复用机制  导致难以维护  我要使用动态样式语言，赋予css新的特性  提高样式语言的可维护性  常用的动态样式语言  1.SCSS/SASS （scss兼容sass,scss更接近css语法）  2.stylus  3.less |

## 十.SCSS详解

1.什么是scss

|  |
| --- |
| scss是一款强化css的辅助工具  它和css语法很像。  它在css的语法基础上，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scss更加强大和优雅  浏览器不能直接解析scss文件，需要在想项目中把scss转义成css文件，让浏览器解析  scss可以让css开发更高效 |

2.scss的安装使用

|  |
| --- |
| scss在服务器端使用  nodejs v8.11以上，才可以使用scss  1.在线安装scss  在cmd中，输入命令npm install -g node-sass  2.无网络安装  找到sass的4个文件，  找到nodejs的安装路径,把sass4个文件放入node文件夹  在cmd中使用node-sass -v 验证版本  黑窗口打开项目路径：C: 切换进入c盘  D: 切换进入d盘 大小写不限  进入指定路径： cd空格 文件夹路径 回车  退回上一级目录：cd 空格 ../ |

3.SCSS文件转换成css文件

①单文件的转换

|  |
| --- |
| 1.创建scss/01.scss文件  2.在项目路径下，打开黑窗口  3.输入下面命令 node-sass scss/文件名称.scss css/文件名称.css  node-sass scss/1.scss css/1.css |

②多文件转换 文件夹--->文件夹

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

③单文件监听转换

|  |
| --- |
| node-sass -w scss/01.scss css/01.css |

④多文件监听，监听一个文件夹中所有文件

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

## 十一.SCSS基本语法

1.变量

|  |
| --- |
| 使用$声明变量，变量名可以包含-，\_  命名规则基本与css选择器相同，尽量做到见名知意  变量声明在{}外，整个scss文件都可以使用。  变量声明在{}内，只有当前{}内可以使用  ！default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。如果之前没有声明赋值，使用现在值 |

2.嵌套

|  |
| --- |
| 1.选择器的嵌套  **#content**{  **width**:**$my\_width**;  **div**.**top**{  **margin**:**$my\_width**;  **h1**{**font-size**:46**px**;}  **p**{**padding**:12**px**;}  }  **div**.**bottom**{  **border**: **$my\_border**;} }  自动转换成css的**后代**选择器  **#content** { **width**: 521**px**; } **#content div**.**top** { **margin**: 521**px**; } **#content div**.**top h1** {**font-size**: 46**px**; } **#content div**.**top p** {**padding**: 12**px**; } **#content div**.**bottom** { **border**: 1**px solid #00f**; } |
| 2.伪类的嵌套  需要在伪类选择器之前添加&（&作为占位符），如果不添加，会生成一个空格导致伪类失效  a{width:21px;  &:hover{width:20px}  } |

练习 使用scss完成

|  |
| --- |
| $myblue:#0aa1ed;  $fontcolor:#fff;  $myheight:2.5rem;  $fontsize:0.8rem; |



|  |
| --- |
| 3.属性的嵌套  **div**{  **border**:{**style**:**solid**;**width**:1**px**;**color**:**#fff**;}; } |
| 4.群组选择器的嵌套  **nav**,**div**,**header**,**footer**{  **a**{**width**:100**px**;} } |

练习

|  |
| --- |
| 一个div#main.200px\*200px;红色背景  鼠标悬停，过渡到黄色背景，并且变为圆形  使用scss编写样式 |

3.导入

|  |
| --- |
| 在scss的语法中，如果一个scss文件以下划线开头，  那么这个scss文件就是一个局部scss文件  scss文件转换成css文件的时候，不会把局部scss文件进行转换  只转换，不以下划线开头的scss文件(全局scss文件)  导入的语法 @import "name";  真正导入的文件名称 \_name.scss 掐头去尾  导入时，不写\_,不写.scss后缀  局部文件被导入后，局部文件中的样式，会在全局文件转换的css中生成。  同时，局部文件中声明的变量，可以在全局文件中使用 |

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  需要使用的这些样式选择器，可以调用这个混合器  实现代码的重用。  关键字 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)  混合器使用场合---css hack |

5.继承

|  |
| --- |
| 一个选择器，可以使用另外一个选择器的所有样式  .my1{  width:100px;height:100px;  }  .my2{  background:#f00;  @extend .my1;  }  转换之后，继承的表现方式，是群组选择器  .my1, .my2 {  width: 100px;  height: 100px; }  .my2 {  background: #f00; } |

练习

|  |
| --- |
| 在选择器.my-border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，当鼠标悬停时，底边框为蓝色 |

6.运算

|  |
| --- |
| 加减乘除余  如果必要，会在不同单位间转换(前提是scss能转) |

①加法

|  |
| --- |
| + 除了做加法，还做字符串拼接  字符串拼接的时候  如果用有引号的字符串拼接无引号的，结果带引号的  如果用无引号的字符串拼接有引号的，结果不带引号 |

②减法

|  |
| --- |
| 由于变量声明的时候可以使用-  系统分不清楚 - 是不是属于变量名称  所以我们写减法的时候，要在 - 前后添加空格  width:$my-w(空格)-(空格)$my-h; |

③除法

|  |
| --- |
| 在scss中， / 的作用是分隔符  只有下面几种情况，我们判定为除法  1.运算式的两边，有变量，或者函数返回值的时候，是除法  width: $w/2;  2.运算式被()包裹的时候，是除法  height:(500px/2);  3.运算式是其他算术运算式的一部分的时候，是除法  margin-left:5px+8px/2px; |

④字符串的插值操作

|  |
| --- |
| 使用#{}在字符串中做插值  content: "liangliang ate #{50+32} baozis"; |

⑤颜色的运算

|  |
| --- |
| #112233+#445566=#557799  #rrggb rgb(r,g,b) 都是分段计算，红+红 绿+绿 蓝+蓝  rgba的元素，需要透明度相同，才允许计算  background:rgba(11,22,33,0.6)+rgba(22,33,44,0.6); |

## 十二.函数

1.scss预定义很多函数，有些函数直接可以在css中使用

|  |
| --- |
| rgba(r,g,b,alpha)  hsl(h,s,l)  hue:色调 取值0~360 3个色段 0~120 120~240 240~360  saturation：饱和度 0.0%~100%  lightness:亮度 0.0%~100% |

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,$v2....)  max($v1,$v2....)  random 随机数 |

3.字符串

|  |
| --- |
| unquote($v) 去掉双引号  quote($v) 加双引号  to\_upper\_case($str) 把$str转成大写  to\_lower\_case($str) 把$str转成小写 |

4.自定义函数

|  |
| --- |
| @function get\_msg($a,$b){  @return $a\*$b+($a/$b);  }  div{  width:get\_msg(3,2)+px;  }  **关键字前面加@，变量前面加$** |

练习

|  |
| --- |
| 定义一个带参数的函数get-width($n)  函数内定义两个变量，分别是60px和70px  函数的返回值为，$n乘以两个数的最大值，加上$n乘以两个数的最小值。把返回值赋给div的width属性  min() max() |

十三.指令

|  |
| --- |
| @if(){}  @else if(){}  @else{}  bool的小括号可以去掉 |

项目阶段

|  |
| --- |
| 学子首页，产品列表页，产品详情页  注意：整个学子不是使用响应式布局设计，也没有使用boot  有一些细节，不去实现  在项目中创建一个文件夹，pro  在pro中，把css文件夹创建好，把图片资源导入 |

作业

1.使用面包屑导航完成



2.作业2，boot项目 头部



1.初始化配置用户名和邮箱，只有第一次使用git时配置

git config --global user.name "liran" #配置用户名

git config --global user.email "liran@123.com" #配置邮箱

git config --list #查看配置列表

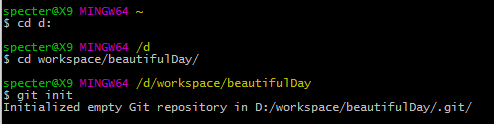
# 代表注释

q 结束

clear 清屏

2.操作本地仓库

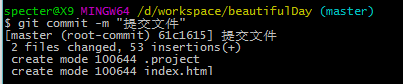
**git init #把文件夹初始化为本地仓库**



**git status #观察那些文件没有放入暂存区**

git add 文件名称 #一次往暂存区添加一个文件

**git add . #把所有文件都添加到暂存区**

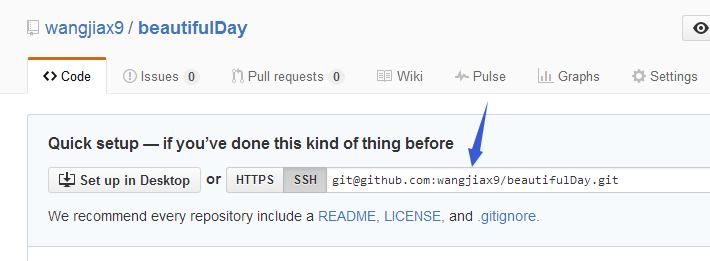


**git commit -m 解释说明 #把暂存区的文件添加到本地仓库**

不管是上传还是删除文件都要commit后push

**Github**

**到github beautifulDay仓库复制仓库地址**



**#关联远程仓库**

**git remote add origin** [**https://github.com/liran0210/2007-1.git**](https://github.com/liran0210/2007-1.git)

**GitHub远程仓库名称**

**#把本地库的所有内容推送到远程库上**

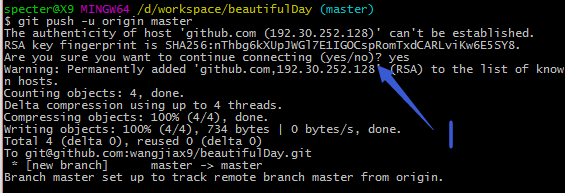
**git push -u origin master #把本地仓库推送给网络仓库**

**username:**

irene-1

**password:**

3631018huang



**敲一个：yes， 然后回车**

**#删除 .git**

**$ rm -rf .git**

git log #查看当前版本之前的所有版本

git checkout 文件名称 #把暂存区中的文件覆盖操作空间中的文件,不能断电

git reset --hard 提交的id #把仓库中的代码恢复到操作空间

git reflog #查看仓库所有版本以及恢复记录

创建.gitignore文件，把不想提交的文件名写在这个文件中

.gitignore文件必须使用开发软件创建

git clone https://github.com/liran0210/2007-1.git

#在本地没仓库的情况下，下载网络仓库使用clone

git pull https://github.com/liran0210/2007-1.git

#在本地存在仓库中，从网络仓储拿代码

git 分支

git branch #查看所有分支

git branch 分支名称 #新建分支

git checkout 分支名称 #切换分支

git merge 分支名称 #当前分支合并另一个分支

删除本地项目：

　　git rm -rf project  或者 rm -rf project   【删除工作区项目】

　　git add project 【将删除的项目添加到版本库，即删除版本库中的项目】

　　git commit project -m '删除项目'   【将需要删除的项目提交到分支，即删除分支中的项目】

删除远程项目（本地项目删除后）：

　　git push origin dev 【将需要删除的项目推送到远程，即删除远程的项目】

# 一.配色网站

[1.NIPPON COLORS。一款很好用的日本配色网站](http://jingyan.baidu.com/album/e75057f231e73debc91a89fc.html?picindex=6" \t "_self)

[](http://jingyan.baidu.com/album/e75057f231e73debc91a89fc.html?picindex=1)

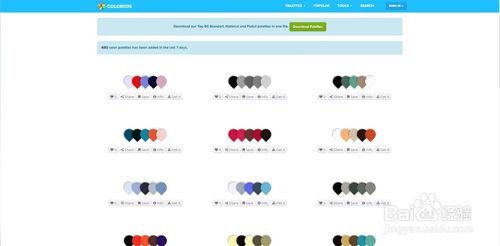
2.uiGradients。丰富的渐变色库可供选择

[](http://jingyan.baidu.com/album/e75057f231e73debc91a89fc.html?picindex=2)

3.Material design palette。即时配色演示网站，效果直观。

[](http://jingyan.baidu.com/album/e75057f231e73debc91a89fc.html?picindex=3)

4colorion。拥有很多已经设计好的色库，直接选择使用

[](http://jingyan.baidu.com/album/e75057f231e73debc91a89fc.html?picindex=4)

1.Color Scheme Designer。可即时选色，支持单色搭配，互补色搭配，三角形搭配等多种方案，并且能够预览配色网站效果

[](http://jingyan.baidu.com/album/e75057f231e73debc91a89fc.html?picindex=5)

1. 6

配色表。此网站为国内网站，有强大的搭配库，还能根据色调或者场景选择色系。

[](http://jingyan.baidu.com/album/e75057f231e73debc91a89fc.html?picindex=6)

# 二.好的软件

**sublime 编辑器 +插件**

**markdown 笔记软件**

**bootstranp的**