笔记

[文档副标题]

目录

赵旭
zhaoxu@tedu.cn
1、问题描述
2、错误的效果图
3、将源码打包发给我

HTML5

DAY1

1、WEB 前端
WEB 前端 : Web Front-End
WEB : 网页
需求:
1、HTML/CSS/Javascript
2、各种框架
jQuery
AngularJS
Bootstrap
zepto
Vue
React
... ...
2、学习方法
1、英文
背
2、练
敲代码
3、打字
3、课程体系
1、阶段一 ： 基础
1、HTML5Basic
2、CSS3Basic
3、PROJECT1 - JD Index
4、CSS3Core
5、PROJECT2 - JD Details
6、Bootstrap 框架
2、阶段二 ： Javascript(JS)
3、阶段三 ： 高级
4、阶段四 ： 框架

1、HTML5Basic (3 天)
****

1、Web 基础知识

1、Internet
1、简介

Internet 实际上就是由计算机所组成的网络结构

服务：

1、Telnet
远程登录

2、Email
电子邮件

3、WWW
万维网服务，World Wide Web

4、BBS
电子公告板 (论坛)
百度贴吧，天涯论坛，... ...
CSDN ... ...

5、FTP
文件传输协议

基本实现技术：

1、分组交换原理
将传递的数据 拆分成若干数据包

2、TCP/IP 协议

2、Web

Web ： 运行在 Internet 上的最流行的应用

WWW : World Wide Web

W3C : World Wide Web Consortium(万维网联盟)

将 各类信息 以及 服务 进行无缝连接：

信息：文字，图像，音频，视频，文件

服务：BBS，Telnet，Email

3、Web 的工作原理

WEB 是 基于 浏览器 / 服务器 模式的程序(B/S)

B : Browser 浏览器

S : Server 服务器

基于 客户端 / 服务器 模式的程序(C/S)

C : Client 客户端

S : Server 服务器

必须通过指定的客户端才能访问服务器数据的一种模式

由 Web 服务器，浏览器 以及 通信协议 来组成

服务器：提供服务的机器

浏览器：工具

通信协议：web 中使用的时 http 通信协议

http: Hyper Text Transfer Protocol
超级 文本 传输 协议

规范了

数据是如何打包的

数据是如何传递的

1、Web 服务器

功能：

1、存储 Web 上内容信息

2、接收客户端请求，并给出响应
3、具备一定的安全功能

产品：

1、Tomcat
2、Apache
3、IIS
... ...

2、WEB 浏览器

功能:

1、代替用户提交请求(User Agent)
2、作为 HTML/CSS/Javascript 的解析器
3、以图像化的方式显示网页文档

产品:

1、Microsoft IE
2、Mozilla FireFox
3、Google Chrome
4、Apple Safari
5、Opera

4、Web 相关技术

1、服务器端技术

运行于服务器端，具备访问数据库的能力

1、PHP
2、JSP
3、ASP
4、ASP.NET
5、Python
6、NodeJS

2、浏览器端技术(客户端)

运行在客户端，由浏览器负责解释

1、HTML
2、CSS
3、JavaScript(JS)

2、HTML 入门

1、HTML 概述

Web : 一种应用

HTML 是开发 Web 网页程序的一种语言

1、什么是 HTML

HTML：Hyper Text Markup Language
超级 文本 标签 语言

超级文本：具备超能力的文本
字符 a ：首字符
超文本 a ：链接

标签/标记：超文本的组成形式

语言：具备不同的语法规范

由 HTML 编写的文本最终是以 .html 或 .htm 作为结尾的文件 , 并且由浏览器解释运行

2、HTML 语法规范(重点)

1、标记

在 HTML 中，用于描述功能的符号称之为 "标记"

语法：

标记在书写时，必须用 尖括号 括起来(<>)

标记分成 封闭类型的标记 和 非封闭类型的标记

1、封闭类型标记

必须成对出现

<标记> 内容 </标记>

注意：

1、封闭类型标记必须成对出现

2、标记必须要完整,否则会有意想不到的效果

2、非封闭类型标记

又称为 单标记 或 空标记

<标记> 或 <标记/>

ex :

 : 换行

<hr/>: 水平线

2、元素

元素 即 标记

ex:

<a>百度

1、元素的嵌套

元素之间可以相互嵌套，形成更为复杂的页面结构

语法：

<标记><标记 1></标记 1></标记>

注意：

1、注意嵌套顺序

2、必须完整嵌套

3、格式问题

被嵌套的内容要通过缩进(Tab)表示层级关系

ex:

```
<a><b><span><i><u>Hello World</u></i></span></b></a>
```

推荐的格式：

```
<a>
    <b>
        <span>
            <i>
                <u>
                    Hello World
                </u>
            </i>
        </span>
    </b>
```


2、属性 和 值

属性 是用来修饰 元素的

语法：

1、属性的声明必须位于开始标记中

2、属性名称与标记名称之间用空格隔开

<标记 属性></标记>

<标记 属性/>

3、属性值 与 属性之间 用 "=" 来连接

属性值要用 "" 或 ' 引起来

<标记 属性="值">

4、一个元素允许有多属性，多属性之间排名不分先后，中间用 空格 隔开

<标记 属性1="值" 属性2="值">

ex:

通用属性：大部分元素都会具备的属性

1、id

定义元素在页面中独一无二的名称

2、title

鼠标移入到元素上时所提示的信息

3、class

指定元素所引用的类选择器(CSS 中使用)

4、style

定义元素的内联样式(css 中使用)

3、注释

要编写在源文档中，但不想被浏览器解释运行的内容

<!-- 注释 -->

注意：

1、注释不能嵌套

<!--

这是一段注释

<!--

这是另一段注释

-->

-->

以上结构是错的

2、注释不能出现在标记(<>)里

<a <!-- 这是一个 a -->>

以上的写法是错误的

3、文档结构

1、两部分组成

1、文档类型声明

指定当前 html 文档用的时哪个版本

语法：

文档中的第一句话位置处

<!doctype html>

2、html 页面

网页要表示的信息的开始与结束

语法：

<html></html>

属性：

1、lang

取值：zh-cn

子级内容：

1、网页头部信息

作用：用于定义网页的全局信息

语法：

<head></head>

子级：

1、网页标题

<title>标题内容</title>

2、网页元数据

指定网页编码格式

<meta charset="utf-8">

注意：

必须保证网页文档的编码格式也是 utf-8 的

2、网页主体信息

包含要显示给用户去看的所有内容

<body></body>

属性：

1、text

作用：控制当前文档的文本颜色

取值：颜色的英文表示方式

2、bgcolor

作用：控制当前文档的背景颜色

取值：同上

3、文本

1、作用

以不同的形式展现文字

2、特殊字符

默认下，任意多个 回车 和 空格 最后都会被折叠成一个空格

通过转义字符表示特殊字符：

1、

一个空格

2、>

>

3、<

<

4、©

©

5、¥

¥

3、文本标记

1、文本样式

内容 斜体显示文本

<u>内容</u> 下划线的文本

<s>内容</s> 删除线的文本

内容 加粗显示文本

 下标

 上标

特点：

所有的内容会在一行内显示

2、标题元素

作用：以标题的方式显示文本(突出显示)

语法：

<hn></hn>

n : 1~6

<h1></h1>

...

<h6></h6>

属性：

1、align ：文本的水平排列方式

取值：left / center / right

特点：

1、独自成行

2、加粗显示文本

3、上下会有垂直的空白

3、段落元素

语法：

属性：

align

特点：

1、垂直空白

2、独占一行

4、换行元素

语法：
 或

5、分割线元素

语法：<hr> 或 <hr/>

属性：

1、size

尺寸，以 px 或%为单位(省略单位的话是 px)

2、width

宽度，以 px 或%为单位

3、align

水平对齐方式

4、color

颜色

6、行分区元素

语法：

作用：包裹文本并且设置不同的样式

7、块分区元素

语法：

作用：布局

8、预格式化

作用：保留标记内的格式(回车 和 空格)

语法：<pre></pre>

9、块级元素和行内元素

1、块级元素

每一个块级元素独占一行

块级元素的主要作用：布局

2、行内元素

多个元素会在一行内显示，显示不下自动换行

span,i,b,s,u,sub,sup

作用：设置文本样式

```
<a>
<b>
<span>
<i>
<u>
<b>
Hello World
</b>
</u>
</i>
</span>
</b>
</a>
```

Ctrl + S : 保存

DAY2

1、复习

1、HTML 语法结构

1、标记

描述功能的符号称为标记

使用时要使用 <> 括起来

标记分为

1、封闭类型

双标记

语法:<标记>内容</标记>

ex:

```
<i></i>
<b></b>
<div></div>
```

2、非封闭类型

单标记 或 空标记

<标记> 或 <标记/>

ex:

<hr>

<input>

2、元素

即标记，包含 标记，属性，内容

1、元素的嵌套

在一个元素中，嵌入另一个元素

ex:

 --父元素(祖先元素)

 --子元素

注意：

1、p 标记不能嵌套块级元素

2、嵌套顺序

3、嵌套格式(缩进)

2、属性

属性是用来修饰元素的

<标记 属性="值" 属性="值">

通用属性：

1、id

2、title : 提示

3、class

4、style

3、注释

<!-- -->

注意：

1、注释不能嵌套

2、注释不能出现在标记中

2、HTML 文档结构

1、页面组成

1、文档类型声明

<!doctype html>

2、页面

<html lang="zh-cn">

</html>

2、页面

1、页面头部

标记：<head></head>

子元素：

1、<title>标题 </title>

2、<meta charset="utf-8">

注意：保证文件也是 utf-8

3、设置网页关键字

面向搜索引擎(SE),不面向用户

<meta name="keywords" content="关键字 1,关键字 2,关键字 3">

4、设置网页描述

面向 SE，不面向用户

<meta name="description" content="描述信息">

5、样式声明

<style></style>

6、样式的引入

<linK>

7、JS 的声明和引用

<script></script>

2、页面主体

<body></body>

属性:

1、text ： 文本颜色

2、bgcolor ： 背景颜色

3、文本

1、特殊字符

  空格

¥ ￥

© © copyright

< < less than

> > greater than

2、文本标记

1、文本样式

 加粗

 斜体

<u></u> 下划线

<s></s> 删除线

 下标

 上标

2、标题

<h1></h1>

...

<h6></h6>

特点:

1、加粗

2、上下独立空白

3、字体大小

4、独立成行

3、段落

4、换行

5、水平线

<hr>

属性：

1、size
2、width
3、align
4、color

6、行分区元素

7、块分区元素

8、预格式化

<pre></pre>
保留源文档中的 空格 和 回车

9、行内元素 与 块级元素

1、行内元素

多个元素会在一行内显示
通常是处理文字的样式

2、块级元素

每个元素独立成行
做布局
所有的块级元素 都具备 align 属性

****

1、图像和链接

1、URL

1、目录 & 目录结构

目录：web 站点中保存文件的文件夹

2、URL

URL:Uniform Resource Locator 即统一资源定位符/定位器，俗称 路径 是 描述资源文件位置的 信息

ex：a.html 中想使用 b.jpg
b.jpg 就是资源文件
a.html 就是当前文件

URL 的三种表示方式：

1、绝对路径

从资源文件所在的最高级目录下开始的完整路径表示

1、获取网络资源文件(只能是绝对路径)

由

通信协议 http / https
主机名(域名/IP 地址) www.baidu.com
目录路径 ： 目录结构
文件名

组成

ex：获取 百度 LOGO 图像
通信协议：https ://
域名：www.baidu.com
目录路径：img
文件名：bd_logo1.png

https://www.baidu.com/img/bd_logo1.png

2、获取本机资源文件

从盘符位置处开始一直到资源文件名字位置

D:/My/Images/page.JPG

2、相对路径

从当前文件位置处开始，去查找资源文件所经过的路径，就是相对路径

ex:

../../Images/a.jpg

3、根相对路径

从 web 站点所在的服务器根目录下开始查找的

以 / 作为开始的

ex:

/images/a.jpg

2、图像

1、语法

标记：<img> 或 <img/>

属性：

1、src (全称：source，源)

<img src="url">

<img src="a.jpg">

<img src="Images/a.jpg">

...

注意：URL 严格区分大小写

2、width

宽度

3、height

高度

注意：

1、如果 width 和 height 只设置其中一个属性的话，那么另外一个将等比缩放

2、尽可能的设置图像的宽和高

3、链接 (超链接)

1、语法

标记：<a>内容

注意：默认情况下，a 是不能被点击的

属性：

1、href

链接的 URL

只有设置 href 属性后，才允许被点击

2、target

目标，打开新网页的方式

取值：

1、_self

默认值，在自身标签页中，打开新网页

2、_blank

在新标签页中，打开新网页

2、链接的表现形式

1、资源下载

链接地址为 **.zip / **.rar

2、电子邮件链接

发送邮件

3、返回页面顶部的空链接

返回顶部

4、链接到 Javascript

3、锚点

1、锚点(Anchor)

在 html 文档的某行位置处做一个记号

允许通过 超链接 跳转到该记号位置处

2、锚点的使用方式

1、定义锚点(做记号)

1、通过 a 标记的 name 属性

内容

2、通过 任意标记的 id 属性

<标记 id="名称"></标记>

2、链接到锚点(跳转到锚点处)

2、表格

1、表格的作用

表格，是由一些称之为 单元格 的东西按照从左到右，从上到下的顺序排列而成的

2、语法

定义表格:<table></table>

定义表行:<tr></tr>

定义单元格:<td></td>

尽量保证默认情况下，每行中的单元格数量是相同的

3、表格属性

1、表格的属性

1、width ： 宽度以 px 或%为单位

2、height： 高度以 px 或%为单位

3、align ： 控制表格在其父元素中水平排列方式 取值：left/center/right

4、border： 边框宽度，默认为 0

5、cellpadding ： 设置单元格内边距

单元格边框与内容之间的距离

6、cellspacing ： 设置单元格外边距

单元格与单元格之间的距离

7、bgcolor ： 背景颜色

2、表行(tr)的属性

1、align

当前行内容的水平对齐方式

2、valign
当前行内容的垂直对齐方式
取值：
top / middle / bottom
3、bgcolor
当前行的背景颜色
3、单元格(td)的属性
1、align
2、valign
3、width
4、height
5、colspan
设置单元格的跨列
6、rowspan
设置单元格的跨行
7、bgcolor
4、单元格的特点
1、某一行单元格的高度，以最高的单元格高度为准
2、某一列的单元格宽度，以最宽的单元格宽度为主
5、table 的子元素
1、表格标题
<caption>标题文本</caption>
注意：
1、一个表格最多只能有一个标题
2、caption 必须位于 <table>下的第一句话
2、td 允许被 th 替换
6、表格的复杂应用
1、行分组
1、表头行分组
<thead></thead>
允许包含 一行或多行 tr
2、表主体行分组
<tbody></tbody>
允许包含任意多的连续 tr
3、表尾行分组
<tfoot></tfoot>
允许包含 一行或多行 tr

注意：
如果不对 table 中的数据进行显示分组的话，默认都在 tbody 中
2、不规则表格
通过 td 的 colspan 和 rowspan 属性创建不规则的表格
1 、colspan
跨列
在一行中，从指定单元格位置处开始，横向向右合并几个单元格(包含自己)

注意：被合并掉的单元格，要删除
2、rowspan
跨行

在同一列中, 从指定单元格位置处开始, 纵向向下合并几个单元格(包含自己)

注意：被合并掉的单元格，要删除

3、表格的嵌套

允许在 单元格 中在放入另一个表格

```
<table>
    <tr>
        <td>
            <table>
                <tr>
                    <td></td>
                    <td></td>
                </tr>
            </table>
        </td>
    </tr>
</table>
```

3、列表

1、语法

1、组成

1、列表的类型

有序列表： --> Order List

无序列表： --> Unorder List

2、列表项

显示在列表中的内容

 --> List Item

2、属性

1、有序列表(ol)的属性

1、type

取值：

1、1 ，数字(默认)

2、a ，小写字母

3、A ，大写字母

4、i ，小写罗马字符

5、I ，大写罗马字符

2、start

起始编号是多少

取值：具体数字

2、无序列表(ul)的属性

1、type

取值：

1、disc

默认值，实现圆点

2、circle

空心圆

3、square

实现方块

4、none
不显示标识

2、列表的嵌套
允许在一个列表中出现另一个列表
被嵌套的列表必须放在 中

3、定义列表
1、作用
往往用于给出一类事物的定义情形，如：名词解释
2、语法
<dl></dl> : 表示定义列表
<dt></dt> : 定义列表中的标题(事物，名词)
<dd></dd> : 对标题(事物，名词)解释说明的内容
3、使用场合
图文混排时使用

DAY3

1.结构标记

1.什么是结构标记?
在 HTML5 中，专门提出的一组用来制作网页布局的
目的：取代 div 布局，提升布局代码的语义性和可读性。
ex:
 --> <header></header>
2.常用结构标记
1.header 元素
语法：<header>内容</header>
作用：定义网页或其他部分内容的页眉信息（网页靠顶部的内容）
2.nva 元素
语法：<nav>内容</nav>
作用：定义页面的导航内容
3.section 元素
语法：<section>内容</section>
作用：定义网页的主体内容
4.aside 元素
语法：<aside>内容</aside>
作用：定义网页任何一个位置的边栏
5.footer 元素
语法：<footer></footer>
作用：定义网页的底部信息，用户不太关注的内容
6.article 元素
语法：<article></article>
作用：定义与文章相关的内容部分，比如：论坛帖子，微博条目，简短新

闻...

2.表单（重点、难点）

1.表单的作用
用于显示，收集数据，并提交信息给服务器
完整的表单处理包含以下两部分：
1.实现数据交互的可见界面元素（前端）
2.提交后的表单数据处理(服务器端)
2.表单元素
标记：<form></form>
属性：
1.action
定义表单提交时发生的动作
指定服务器端处理程序的地址(url)
具体取值要与服务器端人员协同完成
如果省略不写，默认提交给本页
ex:http://www.jd.com/index.jsp
2.method
定义表单数据的提交方式
取值：
1.get(默认值)
1.明文提交，提交的数据会显示在地址栏上
2.安全性不高
3.提交数据有大小限制-2kb
4.场合：向服务器要数据时使用：搜索关键字提交
2.post
1.隐式提交，不会将提交信息显示在浏览的任何位置
2.安全性较高
3.提交数据无大小限制
4.场合：要传递数据给服务器进行处理时，使用 post 提交：注册，登录，上传文件等
3.delete,put...
3.enctype
作用：指定数据进行编码的方式
取值：
1.application/x-www-form-urlencoded
默认值，可以将表单中的普通文本，特殊字符，标点符号都进行二进制编码然后进行提交
2.multipart/form-data
可以将表单中的文件进行二进制编码再提交
字符，标点符号无法进行编辑提交
3.text/plain
可以将表单中的普通字符进行二进制编码再提交，其余都无法提交
4.name 定义表单名称，JS 用到比较多
5.id 定义唯一标识
3.表单控件

1. 作用：用于接收用户数据并依托于表单提交服务器
2. 表单控件的分类
   1. input 元素：文本输入框（文本框，密码框），按钮，单选按钮，复选框...
      标记：<input>
   2. textarea 元素
      多行文本域
      标记：<textarea></textarea>
   3. select 和 option 元素
      选项框
      标记：<select>
            <option></option>
            </select>
   4. 其它元素

3. <input>元素(重点)
   1. 作用：用于收集用户信息
   2. 语法
      标记：<input>
      主要属性：通有属性，所有的 input 元素都具有的属性
      1. type
         根据 type 的值，来创建各种类型的输入控件，比如：文本框，密码，按钮...
      2. name
         控件名称，提供给服务器用
         注意：如果不设置控件名称，则数据无法提交
         匈牙利命名法：控件缩写+功能名称
      3. value
         控件的值，最终提交给服务器的值
      4. disabled
         禁用控件
         表示控件不可用（不能获取值，不能被提交）
         注意：该属性无值
   3. input 控件详解
      1. 文本框和密码
         文本框：<input type="text">
         密码框：<input type='password'>
      2. 专有属性
         1. maxlength
            限制输入字符的数量
            取值：数字
         2. readonly
            只读
            该属性无值
            disabled: 禁用，不能被提交
            readonly: 只读，允许被提交
         3. name
            文本框和密码框的缩写：txt
            ex：<input type="text" name="txtName">
            <input type="password" name="txtPwd">

4.placeholder
占位符

2.单选按钮和复选框

单选按钮：<input type="radio">

复选框：<input type="checkbox">

专有属性：

1.name

名称，同时具有分组作用

单选按钮：一组中只能有一个按钮被提交

复选框：一组数据后台获取时名称相同

名称缩写：

radio-->rdo

checkbox-->chk

2.value

值，被选中中后提交的值

3.checked

设置默认被选中

该属性无值

3.按钮

1.提交按钮

<input type="submit">

作用：提交表单的数据给服务器

2.重置按钮

<input type="reset">

作用：将表单恢复到初始化的状态

3.普通按钮

<input type="button">

作用：由用户自己定义功能(JS)

他们共有属性：

1.name

定义按钮名称，缩写：btn

2.value

按钮上的文字

其它按钮：

1.图片按钮（提交功能）

<input type="image">

属性：src

2.按钮（提交按钮）

<button>内容</button>

4.隐藏域和文件选择框

1.隐藏域

<input type="hidden">

作用：要提交给服务器，但是用户不想看的数据，放在隐藏域中。

2.文件选择框

<input type="file">

作用：允许打开文件选择框，选择文件进行上传

共有属性：

name: 名称
缩写：txt
value: 值
3.注意
使用文件选择框上传文件是，有以下两点要求：
1.form 的 method 属性必须是 post 值
2.enctype 属性值必须是 multipart/form-data

4.textarea 元素
1.作用：完成多行文本的录入功能
2.语法
标记：<textarea>默认值</textarea>
属性：
1.name
名称，缩写为：txt
2.cols
指定文本区域的列数
变相设置控件的宽度
3.rows
指定文本区域的行数
变相设置控件的高度
4.readonly 只读，不能写，但能提交值给服务器
5.选项框
分为：
下拉选项框
滚动列表
语法：
1.标记
<select></select>
创建选项框(滚动列表)
属性：
1.name
名称，缩写：sel
2.size
指定选项框默认能显示几项内容
默认值为 1
如果取值大于 1，则为滚动条列表
3.multiple
设置允许多选
该属性无值
2.<option></option>
作用：定义选项框的子选项
属性：
1.value
选项的值
2.selected
预选中，设置默认被选中的选项
该属性无值
6.其它元素

1.label 元素

1.作用

关联文字与表单控件，关联后，点击文字如同点击表单控件一样

2.语法

标记：<label></label>

属性：

for：表与该元素相关联的表单控件的 ID 属性值

2.为控件分组

标记：<fieldset></fieldset>

子元素：<legend></legend> 指定分组的标题

3.其它常用标记

1.浮动框架

1.作用

可以在一个浏览器窗口中同时显示多个 html 文档

2.语法

1.标记

<iframe>文字内容</iframe>

注意：当浏览器不支持 iframe 元素时，文字内容就会显示

2.属性

1.src

浮动框架中的网页的 url

即要引入的网页的地址

2.height

高度

3.width

宽度

4.frameborder

浮动框架的边框，如果不想要边框的话，设置为 0

4.新表单元素（HTML5 新标记）

缩写：txt

属性：required 非空限制

1.电子邮件类型

<input type="email">

2.搜索类型

<input type="search">

3.url 类型

数据必须符合 url 规范

<input type="url">

4.电话号码

<input type="tel">

移动端：弹出数字键盘

5.数字类型

<input type="number">

属性：

min 定义控件接受的最小值

max 定义控件接受的最大值

step 控制控件递增的步长，默认为 1

6.范围类型
作用：允许选择指定范围内的一个值
<input type="range">
属性：
min:指定范围的最小值（下限值）
max：指定范围的最大值（上限值）
step：值变化的步长
value：设置初始值
7.颜色类型
作用：颜色拾取控件
<input type="color">
8.日期类型
<input type="date">
作用：允许用户选择日期
9.周类型
作用：与 date 相似，但是只能选择周
<input type="week">
10.月份类型
作用：只能选择月份
<input type="month">

CSS3

DAY1

赵旭
zhaoxu@tedu.cn
CSS3Basic : C3 基础

1、CSS 概述

1、问题
1、在 HTML 中，所有的文本颜色变为红色
2、在 HTML 中，让所有的 div 变为蓝色
3、在 HTML 中，让所有的 div 变为黄色

HTML 属性带来的弊端
1、相同的事情要用不同的属性完成
<body text="">
<font color="">
2、相同的事情重复的做(可重用性较低)
2、什么是 CSS
Cascading Style Sheets
层叠样式表，级联样式表，通常称为：样式表
用户在 HTML 中定义元素样式
1、实现内容与表现相分离

2、提供代码的可重用性

3、CSS 与 HTML 之间的关系

HTML ： 定义网页结构

CSS ： 构建页面的样式

HTML 属性 与 CSS 内容 如果能完成相同的事情，优先使用 CSS

2、CSS 语法

1、使用 CSS 样式表的方式

1、内联方式(行内样式)

将 CSS 样式 定义在单个 HTML 元素内

<ANY style="样式声明 1;样式声明 2;...样式声明 n;"></ANY>

样式声明：(重点)

样式属性 1:值;样式属性 2:值;

ex:

样式属性	属性值
color	任意合法颜色值
background-color	任意合法颜色值
font-size	以 px 或 pt 为单位

2、内部样式表

在 <head>元素中的 <style></style> 里 编写若干"样式规则"

```
<html>
    <head>
        <style>
            样式规则 1
            样式规则 2
            ...
            样式规则 n
        </style>
    </head>
</html>
```

样式规则：

```
选择器{
    样式声明 1;
    样式声明 2;
    ...
    样式声明 n;
}
ex
h1{
    样式声明 ...
}
```

特点：

提升了样式的可重用性 和 可维护性

3、外部样式表

将样式规则以及声明，存放在 独立的 样式文件中(**.css)

使用步骤：

1、创建样式表文件并编写样式

2、在使用的网页中，使用 <link>标记链接到外部样式表

```
<head>
    <link rel="stylesheet" href="样式表 URL">
</head>
```

2、样式表的特征

1、继承性

直接使用父元素声明好的样式

大部分样式属性是可以被继承的

2、层叠性

可以为一个元素定义多个选择器

如果样式不冲突时，他们可以合并为一个，都应用在当前元素上

3、优先级

样式定义冲突时，会按照 不同 使用方式的优先级来应用样式

浏览器的缺省设置(UA Stylesheet)　　最低

外部样式表或内部样式表　　中

就近原则 - 后定义者优先

内联样式　　高

4、!important 规则

!important 可以显示调整优先级，永远都以 !important 的为准

样式属性:值 !important;

注意：尽量少用

3、排错

1、Unknown property name

不认识的属性名称 - 属性名写错了

2、Invalid property value

未验证的属性值 - 属性值写错了

3、缺失分号

一部分样式 不识别

4、中文标点

一部分样式 不识别

4、选择器(重点 & 难点)

1、作用

p{} , div{} ,h1{}

规范了页面中哪些元素能够使用定义好的样式

为了匹配页面的元素

2、详解

1、通用选择器

作用：用于匹配页面中所有的元素

语法：* { ... ... }

注意：

尽量少用 通用选择器

大部分属性是可以通过继承来取代 *

2、元素选择器

又称为：标签选择器，标签样式，元素样式

作用：用于定义或重写页面中某元素的默认样式

语法：页面元素作为选择器 {}

h1{ ... }

div{ ... }

...

input{ ... }

3、类选择器

作用：用于定义一些元素的通用样式。类选择器定义好之后，允许被任意元素的 class 属性进行引用

语法：

.类名{}

类名规范：

1、英文，数字，下滑下(_),连字符(-)

2、不能以数字开头

页面元素 通过 class 属性值 ，可以对类选择器进行引用

<ANY class="类名"></ANY>

类选择器-多类选择器的引用方式

即 让一个元素同时引用多个类选择器

语法：

<ANY class="class1 class2 class3">

类选择器-分类选择器的定义方式

即 将元素选择器 和 类选择器结合到一起来定义

目的：为了实现对某种元素中不同样式的的细分控制

语法：元素选择器.类选择器{}

div.important{} ： 匹配页面中 class 为 important 的 div 元素

span.redBack{} :匹配页面中 class 为 redBack 的 span 元素

4、ID 选择器

作用：用于 匹配 页面中 指定 ID 值的 元素

（专属定制/私人订制）

语法：#ID 值{}

样式表中：

#container{

}

5、群组选择器

作用：定义多个选择器中的通用样式

语法：以 , 隔开的选择器列表

ex:

.heavy,#container,div.important{

font-size:25px;

color:orange;

}

等价于

```
.heavt{font-size:25px;color:orage;}
#container{font-size:25px;color:orage;}
div.important{font-size:25px;color:orage;}
```

6、后代选择器

后代元素：出现在某元素的任意层级的子元素都可以称之为 "后代元素"

语法：

选择器 1 选择器 2{ .. ..}

ex:

```
#d1 span{
    /*定义 id 为 d1 中 出现的所有 span 元素的样式*/
}
```

7、子代选择器

子代元素：出现在某元素中，只具备一级层级关系元素称之为 "子代元素"

语法：

选择器 1>选择器 2{}

8、伪类选择器

作用：匹配页面元素不同状态的选择器

语法：

:伪类选择器{}

选择器:伪类选择器{}

分类：

1、链接伪类

1、:link , 定义未被访问的超链接状态

2、:visited, 定义访问过的超链接状态

2、动态伪类

1、:hover, 定义鼠标悬停在元素时的状态

2、:active, 定义元素被激活时的状态

3、:focus, 定义元素获取焦点时的状态

3、目标伪类

4、结构伪类

5、否定伪类

3、尺寸 与 边框

1、单位

1、尺寸单位

1、%

相对单位

2、in

1in=2.54cm

3、cm

厘米

4、mm

毫米

5、pt

磅(点) -> Point

1pt = 1/72in

多数用在 文字 大小

6、px

Pixel ，像素，计算机屏幕上的一个点

屏幕分辨率：1024*768

7、em

倍数，父元素文字大小倍数

1em ： 父元素字体原始大小

2em ： 父元素字体大小的 2 倍

8、rem

倍数，相对于根元素(html)设置字体的尺寸倍数

2、颜色单位

在 css 中合法的颜色取值

1、rgb(r,g,b)

r:red (0-255)

g:green (0-255)

b:blue (0-255)

ex:

```
#d1{
    color:rgb(0,0,0);/*黑色*/
}
```

2、rgb(r%,g%,b%)

3、rgba(r,g,b,alpha)

alpha : 透明度

0 - 1 之间的数字

4、#rrggbb

由 6 位 十六进制数字 表示一个颜色

十六进制：0-9 A-F 组成

A:10,B:11,C:12,D:13,E:14,F:15

ex:

#000000 : 黑色

#111111 : 深深深灰

#eeeeee : 浅浅浅灰

#ffffff : 白色

#1a2b3c : ?

5、#rgb

#rrggbb 的缩写，每两位数字相同时，可以采用缩写

#000000 -> #000

#aabbcc -> #abc

#1abbcc -> 无法缩写

6、颜色的英文表示法

red,blue,darkred,lightright,

2、尺寸属性

1、作用

用于设置元素的宽度和高度

单位通常为 px 或 %

2、语法

1、宽度

width : 指定元素的宽度

max-width : 最大宽度

min-width : 最小宽度

注意：

块级元素默认宽度：父元素宽度的 100%

行内元素默认宽度：以内容为准

2、高度

height : 指定元素的高度

max-height: 最大高度

min-height: 最小高度

注意：

所有元素默认高度：以内容为准

3、哪些元素允许修改尺寸

1、块级元素 都允许修改尺寸

div,p,h1~h6,section,header,footer,...

2、大部分行内块元素允许修改

input 元素允许改，但 checkbox,radio 除外

3、存在 width，height 属性的 HTML 元素允许修改

<img width="" height="">

<table width="" height="">

4、溢出

1、什么是溢出

使用尺寸属性控制元素大小时，如果内容所需空间大于元素本身，会导致内容 "溢出"

2、溢出处理属性

属性：

overflow

overflow-x : 横向溢出处理

overflow-y : 纵向溢出处理

取值：

1、visible

默认值，溢出可见

2、hidden

隐藏，溢出隐藏

3、scroll

滚动，让元素显示滚动条，溢出时可用

4、auto

自动，溢出时显示滚动条并可用

3、边框属性

```
<div id="d1">
    <span>#d1 中的 span</span>
    <p>
        <span>
            <b>#d1 中的 p 中的 span 中的 b</b>
        </span>
```

```
    </p>
</div>
```

DAY2

1、CSS 概述

1、CSS 与 HTML 之间的关系

```
<p align="center"></p>
css:
p{
    align:center;/*错误*/
}
```

2、CSS 语法

1、CSS 的使用方式

1、内联方式

```
<ANY style="样式声明;">
样式声明的组成 ： 样式属性:值;
```

2、内部样式表

```
<style>
    样式规则

    选择器{
        样式声明;
    }
</style>
```

3、外部样式表

1、创建独立的 css 文件，并编写样式

编写 样式规则

2、在使用的页面中进行引入(链接)

```
<link rel="stylesheet" href=" url " />
```

2、选择器

1、通用选择器

```
*{}
```

2、元素选择器

```
html{}
```

3、类选择器

```
.类名{}
```

多类选择器的引用方式

```
<ANY class="class1 class2 class3">
```

分类选择器的定义方式

```
元素选择器.类选择器{}
```

4、ID 选择器

专属定制

```
#ID{}
```

5、群组选择器

```
选择器 1,选择器 2,选择器 3,...{}
```

6、后代选择器
后代：无限子级
选择器 1 选择器 2{}
7、子代选择器
子代：一级层级 (下一级)
选择器 1>选择器 2{}
8、伪类选择器
1、链接伪类
:link
:visited
2、动态伪类
:hover (重点)
:active
:focus

3、尺寸属性
1、尺寸
1、宽度
width
max-width
min-width
2、高度
height
max-height
min-height
3、允许设置尺寸属性的元素
1、块级元素
2、行内块
表单元素(radio,checkbox 除外)
3、具备 width 和 height html 属性的元素
<img>
<table>
4、溢出
空间已经包含不下所有的内容
属性:
overflow
overflow-x
overflow-y
取值
1、visible : 默认值，可见
2、hidden : 隐藏
3、scroll
4、auto

==========

img : display:inline;(行内)
p: display:block;(块级)
div: display:block;(块级)
table : ?
ul : ?
li : ?

==========

1、边框属性

1、边框

1、简写方式

border:width style color;

width:边框的宽度，以 px 为单位的数值

style:边框的样式

solid : 实线

dotted : 虚线(点状)

dashed : 虚线(线状)

color:边框的颜色，合法颜色值

特殊： transparent , 透明

ex:

border:1px solid #000;

border:0; 或 border:none;

2、单边定义

只定义某一方向的边框的 粗细，样式，颜色

border-方向:width style color;

方向： top/right/bottom/left

ex:

border-top:1px solid red;

3、单属性定义

定义四个边框的某一个属性值

border-属性:值;

属性： width/style/color

ex:

border-color:#e4393c;

4、单边单属性定义

定义某一方向的具体某一属性值

border-方向-属性:值;

方向： top/right/bottom/left

属性： width/style/color

ex:

1、下边框的颜色为 #ddd

border-bottom-color:#ddd;

2、右边框的样式为 dotted

border-right-style:dotted;

5、边框的组成

是由四个等边三角形组成的

2、边框倒角

将 元素的 四个直角 倒换成 圆角

语法：

属性： border-radius

取值：

1、具体数值

2、百分比

采用当前元素 宽度 和 高度的占比 来作为倒角圆的半径

单角定义:

border-top-left-radius:左上角倒角半径值;
border-top-right-raius:右上角...
border-bottom-left-radius:
border-bottom-right-radius:

3、边框阴影

语法:

属性: box-shadow

取值:

取值为多个值的列表

h-shadow v-shadow blur spread color inset;

h-shadow:必须的，阴影的水平偏移距离

取值为正，阴影右偏移

取值为负，阴影左偏移

v-shadow:必须的，阴影的垂直偏移距离

取值为正，阴影下偏移

取值为负，阴影上偏移

blur:模糊距离

spread:阴影大小

color:颜色

inset:将默认的外阴影改为内阴影

4、边框轮廓

1、什么是轮廓

是绘制与元素周围的一条线，位于边框之外，起到突出显示的作用

2、语法

outline:width style color;

outline-width:轮廓的宽度

outline-style:

outline-color:

outline:none; 或 outline:0;

2、框模型

1、框模型

框: Box，泛指页面中所有的元素

页面元素皆为框

框模型: Box Model，定义了元素框处理元素内容(尺寸)，内边距，外边距，以及边框的方式

外边距: 元素(框) 与 元素(框)之间的距离

内边距: 元素(框) 与 内容之间的距离

元素的占地尺寸

元素总宽度=左右外边距 + 左右边框 + 左右内边距 + width;

元素总高度=上下外边距 + 上下边框 + 上下内边距 + height;

元素的可视化尺寸(边框内)

边框内宽度=左右边框+左右内边距+width;

边框内高度=上下边框+上下内边距+height;

2、外边距

1、什么是外边距

元素边框(边缘)之外的距离

2、语法

属性：

margin:值;

margin-top/right/bottom/left:值;

取值：

1、具体数值 px

2、百分比 %

3、auto

默认情况下，可以自动计算左右外边距

实现 块级元素 水平居中对齐

注意：必须为元素设置宽度才能居中

4、负数

向相反的方向移动元素

margin-top

取值为正，元素下移动

取值为负，元素上移动

margin-left

取值为正，元素右移动

取值为负，元素左移动

简写方式：

margin:value; 四个方向外边距值

ex

margin:10px;

margin:v1 v2; v1:上下 , v2:左右

ex

margin:0 auto;

margin:v1 v2 v3;v1:上 v2:左右 v3:下

ex

margin:0 auto 15px;

margin:v1 v2 v3 v4;上 右 下 左

3、页面中具备默认外边距的元素

在页面中，通过 CSS Reset(CSS 重写)的方式将默认的外边距全部都清除

```
body,h1,h2,h3,h4,h5,h6,p,ul,ol,dl,dd,pre{
    margin:0;
}
```

4、特殊问题

1、外边距的合并

当两个垂直外边距相遇时，他们将形成一个外边距，合并后的外边距值等于两个元素外边距中较大的数值

2、外边距的溢出

在某些特殊的情况下，为子元素设置上外边距时，有可能会作用到父元素上

条件：

1、父元素没有上边框

2、必须为第一个子元素设置上外边距

解决方案：

1、给父元素加上边框

弊端：父元素高度会发生变化

2、通过给父元素增加上内边距取代子元素上外边距

弊端：改变父元素高度

3、在父元素中增加一个空 table

弊端：多了一个空元素

4、... ...

后续完成...

3、为行内元素 和 行内块元素 设置 上下外边距

行内元素：上下外边距无效(img 除外)

行内块元素：设置上下外边距时，整行元素都跟着动

3、内边距

1、什么是内边距

元素的边框(边缘) 到 内容之间的距离

注意：内边距会扩大元素边框所占用的区域

2、语法

属性：

padding

padding-top/right/bottom/left

取值：

1、px

2、%

3、auto

简写

padding:v1; 四个方向内边距

padding:v1 v2;上下 左右

padding:v1 v2 v3;上 左右 下

padding:v1 v2 v3 v4; 上 右 下 左

3、特殊效果

为行内元素 和 行内块元素 增加上下内边距时有特殊效果

特殊效果：只会影响自己，并不会影响其他元素

4、box-sizing

作用：指定 框模型的 计算方式

```
div{
    width:471px;
    border-left:2px solid #ddd;
    border-right:6px solid #fff;
    padding-left:18px;
    padding-right:3px;
}
```

Element Width = 471 + 2 + 6 +18 + 3;

属性：box-sizing

取值：

1、content-box

元素的 width 属性值，不包含 padding 和 border

Element Width = width + padding-left/right + border-left/right;

2、border-box

元素的 width 属性值，会包含 padding 以及 border
Element Width = width;

3、背景属性

1、背景颜色
属性：background-color
取值：合法颜色值 或 transparent
注意：背景颜色会填充到边框，内边距以及内容区域

2、背景图片
属性：background-image
取值：url("图片 url")

背景图 与 <img> 有什么区别?
<img>是独立的元素
背景图：不是元素，是某元素背景的衬托

3、背景重复(平铺)
当元素区域大于背景图的话，默认是以平铺的方式出现
属性：background-repeat
取值：

1、repeat
默认值，水平和垂直方向都平铺
2、repeat-x
只在水平方向平铺
3、repeat-y
只在垂直方向平铺
4、no-repeat
不平铺

4、背景图片尺寸
属性：background-size
取值：

1、width height
2、width% height%
3、cover
覆盖，将背景图等比扩大，直到背景图覆盖到元素所有区域为止
4、contain
包含，将背景图等比扩大，直到背景图的一个边缘碰到元素右边或下边为止

5、背景图片固定
作用：让背景图一直保持在可视化区域中，不随着滚动条而发生改变
属性：background-attachment
取值：

1、scroll
默认值，滚动
2、fixed
固定

注意：尽量为 body 设置背景

6、背景定位/位置
属性：background-position

取值：

1、x y

x ： 背景图横向偏移距离

x 取值为正，背景图，向右偏移

x 取值为负，背景图，向左偏移

y ： 背景图纵向偏移距离

y 取值为正，背景图，向下偏移

y 取值为负，背景图，向上偏移

2、x% y%

0% 0% : 左上角

100% 100% : 右下角

50% 50% : 正中间

3、关键字

x : left / center / right 取代

y : top / center / bottom 取代

CSS Sprites

雪碧图，精灵图

将若干幅小图像封装到一幅大图像中，以便减少 http 请求次数

实现步骤：

1、在页面中，根据要看的图像的尺寸，创建一个一模一样大小的元素

2、为该元素设置背景图，并且通过 背景图像定位 将要显示的图移动到 元素区域内

7、背景属性(简写)

background:color url() repeat attachment position;

ex:

background:#f00;

background:url(iconlist_1.png) -243px -112px;

DAY3

**********复习**********

1、边框

1、边框

border:width style color;

border-方向:width style color;

border-属性:值;

border-方向-属性:值;

2、边框倒角

border-radius:值;

3、边框阴影

box-shadow:h-shadow v-shadow blur spread color inset;

4、轮廓

outline:width style color;

outline:none;

2、框模型

1、外边距

属性：

margin
margin-top/right/bottom/left

注意：

1、外边距合并
2、外边距溢出
3、行内元素 行内块元素 设置垂直外边距

2、内边距

属性：

padding
padding-top/right/bottom/left

注意

1、行内元素 和 行内块元素 设置垂直外边距时，只影响自己

3、box-sizing

1、content-box
元素的 width 和 height 是不包含 border 以及 padding
2、border-box
元素的 width 和 height 会包含 border 以及 padding

3、背景

1、background-color
2、background-image
3、background-repeat
no-repeat,repeat,repeat-x,repeat-y
4、background-size
1、x y
2、x% y%
3、cover
4、contain
5、background-attachment
scroll,fixed
6、background-position
1、x y
2、x% y%
3、关键字
x : left / center / right
y : top / center / bottom

CSS Sprites 减少 HTTP 的请求次数

7、background
background:color url repeat attachment position;

background:red;
background:url(a.jpg);

==========

1、渐变

1、什么是渐变

两种或多种颜色平缓过渡的效果

2、重要元素
色标：指定每个颜色的值以及出现的位置
3、分类
1、线性渐变
以直线的方式填充渐变颜色
2、径向渐变
以圆形的方式填充渐变色
3、重复线性渐变
4、重复径向渐变
4、语法
1、属性
background-image
2、取值
1、线性渐变
linear-gradient(angle,color-point1,color-point2,color-point3);
1、angle
1、关键字
1、to top :
从下向上 填充渐变色
第一个色标，在最底下出现
最后一个色标，在最上面出现
2、to right
从左向右 填充渐变色
第一个色标，在最左边出现
最后一个色标，在最右边出现
3、to bottom
从上向下 填充渐变颜色
4、to left
从右向左 填充渐变颜色
2、具体角度值
0deg ~ 360deg
2、color-point
色标：包含颜色 及其 出现的位置
ex:
1、red 0%
从开始的位置 显示为红色
2、blue 50%
中间位置处 显示为蓝色
3、yellow 200px
4、red
省略位置，由浏览器自动计算
2、径向渐变
属性：background-image
取值：radial-gradient([size at position],color-point1,color-point2);
1、size at position
size : 半径大小，以 px 或%为单位
at : 关键字，不能省略
position : 圆心所在位置
3、重复线性渐变

background-image:repeating-linear-gradient(angle,color-point);

background-image:repeating-linear-gradient(to top,red 0%,blue 10%);

4、重复径向渐变

background-image:repeating-radial-gradient([size at position],color-point1);

5、浏览器兼容性

渐变：CSS3 提出的新属性

C3 新属性，放在 老版浏览器中 支持性不好!

目前为止，主流浏览器的主流版本，对 C3 支持性特别好!但对于不支持 C3 新属性的版本浏览器，需要通过增加 "浏览器前缀"实现兼容性

常用浏览器前缀：

1、Firefox : -moz-

2、Chrome & Safari : -webkit-

3、Opera : -o-

4、IE : -ms-

浏览器前缀加在哪？？？

1、如果浏览器不支持属性的话，则将前缀加在属性名称前

animation : C3 的新属性

兼容：

-moz-animation:值;

-webkit-animation:值;

-o-animation:值;

-ms-animation:值;

animation:值;

2、如果浏览器不支持属性值的话，则将前缀添加在属性值前

background-image:-moz-linear-gradient();

background-image:-webkit-linear-gradient();

background-image:-o-linear-gradient();

background-image:-ms-linear-gradient();

background-image:linear-gradient();

2、文本格式化

1、字体属性

1、指定字体系列

属性：font-family

取值：字体 1,字体 2,字体 3;

注意：

如果字体中包含 中文 或 空格的话，最好用 "" 或 ' 括起来

ex:

font-family:"微软雅黑";

font-family:"microsoft yahei",arial;

2、字体大小

属性：font-size

取值：px,pt,em,rem

3、字体加粗

属性：font-weight

取值：

1、normal

正常体，非加粗

2、bold

加粗

3、value

400 ： 相当于 normal

900 ： 相当于 bold

4、字体样式

属性：font-style

取值：

1、normal

正常，非斜体

2、italic

斜体

5、小型大写字母

作用：将所有的小写字符改成大小，但是大小和小写字符一样大

属性：font-variant

取值：

1、nomal

正常

2、small-caps

小型大写字符

6、简写模式

font:style variant weight size family;

注意：

使用简写属性时，必须设置 family 的值，否则无效!!!

font:12px;//无效果

font:12px "微软雅黑";//OK

2、文本属性

1、文本颜色

属性：color

取值：合法颜色值

2、文本排列

属性：text-align

取值：

1、left

2、center

3、right

4、justify

两端对齐

3、文本修饰(线条)

属性：text-decoration

取值：

1、none

没有线条显示

2、underline

下划线

3、overline
上划线

4、line-through
删除线

4、行高
作用：一行文本所占高度,如果行高大于文字高度大小，那么文字将在行高范围内垂直居中
属性：line-height
取值：px 为单位
ex:
line-height:24px;
line-height:1.5;

5、首行文本缩进
属性：text-indent
取值：以 px 为单位的数值

6、文本阴影
属性：text-shadow
取值：h-shadow v-shadow blur color;

3、表格属性

1、表格常用属性
1、边距属性
padding
2、尺寸属性
with height
3、文本格式化
font,text
4、背景属性
5、边框属性
border
6、垂直方向对齐
属性：vertical-align
取值：top / middle / bottom

2、表格特有属性
1、边框合并
属性：border-collapse
取值：
1、separate
默认值，即分离边框模式
2、collapse
边框合并模式

2、边框边距
属性：border-spacing
取值：
1、指定一个值
水平 和 垂直间距相等
2、指定两个值

第一个值：水平间距
第二个值：垂直间距

注意：保证 border-collapse 的值为 默认值(separate),否则该属性无效

3、标题位置

属性：caption-side

取值：

1、top
默认值，标题在表格之上

2、bottom
标题在表格之下

4、显示规则

指定表格布局的算法

属性：table-layout

取值：

1、auto
自动，默认值
列宽度由单元格内容来决定

2、fixed
固定
固定表格布局，列宽度由设定的值为准，不以数据来决定

自动表格布局 VS 固定表格布局

1、自动表格布局

1、单元格的大小会适应内容的大小
2、表格复杂时，加载速度慢(缺点)
3、灵活(优点)
4、不确定每列大小时，优先使用自动表格布局

2、固定表格布局

1、单元格大小取决于 css 中设定的值，与内容无关
2、加速显示表格(优点)
3、不够灵活(缺点)
4、已知每列大小时，可以使用固定表格布局

DAY4

1、渐变

属性：background-image

取值：

线性渐变：linear-gradient(angle,color-point);
径向渐变：radial-gradient([size at position],color-point);
重复线性渐变：repeating-linear-gradient()
重复径向渐变：repeating-radial-gradient()

兼容性：

浏览器前缀：

Firefox : -moz-
Chrome & Safari : -webkit-
Opera : -o-
IE : -ms-

浏览器不支持属性，前缀加在属性前

浏览器不支持值，前缀加在属性值前

2、文本格式化

1、字体

font-family,font-size,font-weight,font-style,font-variant,font

2、文本

color,text-decoration,text-align,line-height,text-indent,text-shadow

3、表格

1、常用属性

margin 不能用在 td 中

vertical-align : top / middle / bottom

2、特有属性

1、border-collapse : separate / collapse

2、border-spacing : 值 ;

3、caption-side : top / bottom;

4、table-layout : auto / fixed;

==========

1、定位 - 浮动

1、什么是定位

定位，就是改变元素在页面中的默认位置

2、定位的分类

1、普通流定位 - 页面默认定位方式

2、浮动定位 - 多个块级元素在一行内显示

3、相对定位

4、绝对定位

5、固定定位

3、定位-普通流定位

普通流定位，又称为 "文档流定位", 是页面中默认的定位方式

特点：

1、所有的元素都是从父元素的左上角开始出现

2、每个元素都有空间，默认不能重叠

3、块级元素 - 从上到下显示

4、行内 & 行内块 - 从左到右，显示不下换行

4、浮动定位（重点）

1、什么是浮动

如果将元素的定位方式设置为浮动定位，将具备以下特征：

1、浮动，会将元素排除在文档流之外(脱离文档流),不占据页面空间，其它元素要上前占位

2、浮动元素会停靠在父元素的左边或右边,或其它平级的已浮动元素的边缘上

3、浮动元素只能在当前行浮动

4、浮动元素依然在父元素中

5、浮动定位处理的问题-让多个块级元素在一行内显示

2、语法

属性：float

取值：

1、none

默认值，即元素无任何浮动效果

2、left

左浮动，让元素停靠在父元素的左边，或者挨着左侧已有浮动元素的右边上

3、right

右浮动，让元素停靠在父元素的右边，或者挨着右侧已有浮动元素的左边上

3、浮动引发的特殊效果

1、父元素中，如果显示不下所有的已浮动子元素，最后一个会换行，有可能会被卡住

2、元素一旦浮动起来之后，元素宽度将由内容来决定(非指定情况下)

3、元素一旦浮动起来之后，那么都将变成块级元素，尤其对行内元素影响较大

块级元素：允许修改尺寸

行内元素：不能修改尺寸

4、文本，行内元素，行内块元素 是采用环绕的方式来排列的，不会被浮动元素压在低下，而会巧妙的避开浮动元素

4、清除浮动所带来的影响

元素一旦浮动起来，会对后面的元素带来位置的影响，如果后面的元素不想被影响的话，可以通过 "清除" 的方式 来解除影响

属性：clear

取值：

1、none

默认值，即不做任何清除浮动

2、left

清除 前面 元素左浮动所带来的影响，即前面元素左浮动的话，当前元素不会上前占位

3、right

清除 前面 元素右浮动所带来的影响，即前面元素右浮动的话，当前元素不会上前占位

4、both

清除 前面 元素任何一种浮动所带来的影响

5、浮动元素对父元素的影响

由于浮动元素会脱离文档流，所以会导致元素不占据空间.如果一个元素中所有的子元素全部都是浮动的，那么该元素的高度就是 0 ，父元素的高度，是以未浮动元素的高度为准

解决方案：

1、直接设置父元素的高度

弊端：必须要知道父元素的高度

2、设置父元素也浮动

弊端：对后续元素时有位置影响的

3、为父元素设置 overflow 取值为 hidden 或 auto

弊端：如果有要溢出显示的内容，也一同被隐藏

4、在父元素的最后一个子元素位置处，增加 一个空块级元素，并且设置其 clear:both

5、... ...

2、显示

1、显示方式

作用：指定了元素在页面中默认的布局方式
属性：display
取值：

1、none
让生成的元素不显示并且脱离文档流

2、block
让元素变的与块级元素一样

3、inline
让元素变的与行内元素一样

4、inline-block
让元素变的与行内块元素一样
特点：
1、多个行内块以及行内元素能在一行内显示
2、行内块元素允许修改尺寸(除 radio,checkbox)

5、table
让元素变的与 表格 一样
特点：
1、独占一行
2、宽度以内容为准

6、table-cell
让元素变的与 单元格 一样

2、显示效果

1、可见性
属性：visibility
取值：

1、visible
默认值，元素可见

2、hidden
隐藏，没脱离文档流

display:none 与 visibility:hidden 的区别
display:none 会脱离文档流，所以不占空间

3、collapse
用在表格元素上，删除一行或一列时，表格整体布局不受影响

2、透明度
属性：opacity
取值：0.0(完全透明) ~ 1.0(完全不透明)

rgba(r,g,b,alpha);

```
div{
    background-color:rgba(255,0,0,0.5);
    opacity:0.5;
}
```

3、垂直对齐
属性：vertical-align
作用：

1、作用在表格(table,tr,td,th,...)元素上
指定表格中的内容的垂直对齐方式
取值：top / middle / bottom

2、作用在 图像 以及 行内块元素上时
指定 图像 或 行内块 元素 两端的文本的垂直对齐方式
取值：
1、top
2、middle
3、bottom
4、baseline
基线对齐
对于行内块元素：基线在最后一行文本的底部
对于图片元素：基线是在图片底部向下 3px 位置处(默认情况)

vertical-align:bottom;解决图像多占 3px 高的问题

3、光标
属性：cursor
取值：
1、default
默认值
2、pointer
小手
3、crosshair
+
4、text
I
5、wait
等待
6、help
帮助

3、列表

1、列表项标识
属性：list-style-type
取值：
1、none
... ...
2、列表项图像
属性：list-style-image
取值：url(图像的 url)
3、列表项位置
属性：list-style-position
取值：
1、outside
默认值，标识位于列表项之外
2、inside
将标识位置改为列表项之内
4、列表属性
属性：list-style
取值：type url() position;

常用形式：list-style:none;

4、定位

1、定位属性

1、定位属性

属性：position

作用：指定元素的定位方式

取值：

1、static
默认值

2、relative
相对定位

3、absolute
绝对定位

4、fixed
固定定位

注意：如果元素设置为 相对定位/绝对定位/固定定位的话，则会被称之为"已定位元素"

2、偏移属性

作用：对已定位属性实现位置的移动

属性：

top:以上边为基准边改变元素位置
right:以右边为基准边改变元素位置
bottom:以下边为基准边改变元素位置
left:以左边为基准边改变元素位置

3、堆叠顺序

属性：z-index

取值：无单位的数字

2、定位-相对定位

1、什么是相对定位

元素会相对于它原来的位置偏移某个距离

2、语法

属性：position

取值：relative

配合偏移属性 完成位置的移动

3、注意

相对定位元素 原来的位置 会被保留，不能被其它元素所占据（相对定位与 margin 的区别）

3、定位-绝对定位

1、什么是绝对定位

1、绝对定位元素会脱离文档流-不占据页面空间

2、绝对定位的元素会相对于 离他最近的 已定位的 祖先元素 去实现位置初始化

3、配合 偏移属性 实现位置的改变

4、如果 绝对定位元素 没有 已定位的祖先元素，那么它的位置就相对于最初的包含框(body 或 html)实现初始化

2、语法

属性：position

取值：absolute
配合 偏移属性 实现元素的移动

3、特点

1、会脱离文档流
2、绝对定位元素会变成块级元素
3、绝对定位元素在正常情况下，margin:auto 会失效，其它正常
4、非正常情况，auto 会生效
绝对定位元素的 top,right,bottom,left 同时为 0 时，auto 会生效

CSS3 core

Day1

1、复杂选择器
2、内容生成
3、弹性布局
4、CSS Hack(IE 浏览器兼容性)
****

1、复杂选择器

选择器作用：匹配 页面的元素

1、兄弟选择器

兄弟：具备 同一 父元素的 平级元素称之兄弟元素

1、相邻兄弟选择器

相邻：挨着的

```
<div id="d1"></div>
<div id="d2"></div>
<div id="d3"></div>
```

语法：选择器 1+选择器 2

```
1、div+div{color:red;}
    d2，d3 的颜色为 red 色
```

2、通用兄弟选择器

通用：后面所有
语法：selector1~selector2

特点：只能向后找兄弟，不能向前找兄弟

2、属性选择器

1、什么是属性选择器

允许通过 元素所附带的属性 及其 值来匹配页面元素

```
<div id="parent"></div>
```

想匹配页面中所有的文本框？？？

```
<input type="text">
```

2、语法

基础语法规范：[]

1、[attr]
匹配页面中附带 attr 属性的所有元素

ex:

1、[id]

匹配页面中所有 有 id 属性的元素

2、[type]

匹配页面中所有 有 type 属性的元素

3、匹配页面中所有被禁用的元素

[disabled]

2、elem[attr]

elem : 表示 任意 元素

attr : 表示 任意 属性

匹配 具备 attr 属性的 elem 元素

ex:

1、div[id]

匹配页面中所有 附带 id 属性的 div 元素

3、[attr1][attr2]

匹配 同时具备 attr1 以及 attr2 属性的元素

ex:

1、匹配页面中所有 有 class 的文本框元素

input[class][type]

4、[attr=value]

匹配页面中所有 attr 属性值为 value 的元素

ex:

1、[id=parent]

等同于 #parent

2、[class="container"]

等同于 .container

3、想匹配页面中所有的文本框

[type='text']

4、想匹配页面中所有的提交按钮

[type='submit']

5、[class~=value]

匹配的元素的 class 属性值是由多个类选择器组成的列表，value 是该列表中的一个独立选择器

即：匹配选择器列表中包含 value 选择器的元素

6、[attr ^= value]

^= : 以 ... 作为开始

匹配 attr 属性值 是以 value 字符作为开始的元素

匹配 class 属性值中 是以 col 作为开始的 div 元素

div[class^=col]

7、[attr *= value]

*= : 包含...字符

匹配 attr 属性值中 包含 value 字符的元素

...

匹配 class 属性值中 包含 md 的所有 div 元素
div[class*=md]{}

8、[attr $= value]
$= : 以 ... 作为结束
匹配 attr 属性值 是以 value 字符作为结尾的元素

3、伪类选择器
1、目标伪类
1、作用
突出显示 被激活的 HTML 锚点 元素
2、语法
:target / elem:target
2、结构伪类
通过元素间的结构关系来匹配页面元素
语法：
1、:first-child
匹配 属于其 父元素中的 首个子元素
2、:last-child
匹配 属于其 父元素中的 最后一个子元素
3、:nth-child(n)
匹配 属于其 父元素中的 第 n 个子元素

想获取 第二行 里的 第三列 ， 背景颜色改为 红色
第二行：tr:nth-child(2)
第三列：td:nth-child(3)
第二行 第三列：
#tbl tr:nth-child(2) td:nth-child(3)
4、:empty
匹配每个没有子元素(包含文本)的元素

ex:

 非 empty 的 div

Hello World
 非 empty 的 div

 非 empty 的 div

 纯 empty 元素
5、:only-child
匹配 属于其父元素中的 唯一子元素
3、否定伪类
作用：将 指定选择器匹配的元素 排除出去
语法：

:not(selector)

ex：

获取 除第一行以外的 所有行

#tbl tr:not(:first-child){}

4、伪元素选择器

1、伪类 和 伪元素的区别

伪类：匹配的是元素(不同状态或结构的)

伪元素：匹配的是元素中的一部分内容(首字符，首行文本)

2、语法

1、:first-letter 或 ::first-letter

匹配 某元素的 首字符

2、:first-line 或 ::first-line

匹配 某元素的 首行文本

3、::selection

匹配 被用户选取的 部分

3、: VS ::

在 CSS3 之前，伪元素 使用的是 :

:first-letter

:hover

在 CSS3 之后，伪元素 一律使用 ::

::first-letter

:focus

2、内容生成

1、什么是内容生成

通过 css 的属性和选择器，向某元素内增加一部分内容

2、伪元素选择器

1、:before 或 ::before

匹配 某元素的内容区域之前

(内容区域之前)这是一个 div

2、:after 或 ::after

匹配 某元素的内容区域之后

这是一个 div(内容区域之后)

3、属性

属性：content

取值：

1、字符串

一系列的文本字符，用"" 或 ' 引起来

2、图像

提供图像地址

url(路径)

3、计数器

4、解决问题

1、浮动元素父元素高度

2、上外边距溢出

为子元素设置上外边距时，在某种条件下，会作用到父元素上

<table></table>

3、弹性布局

Flexible Box ，处理某元素内子元素的排列方式

页面中任何一个元素都可以指定为 弹性布局(Flex)

属性：display

取值：

1、flex

将块级元素变为弹性布局容器

2、inline-flex

将行内元素变为弹性布局容器

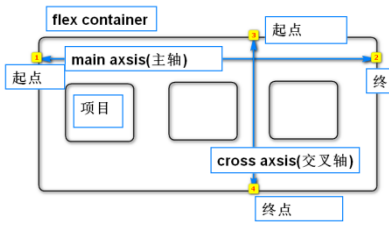
兼容性：

display:-webkit-flex;

... ...

注意：将元素设置为 flex 后，子元素的 float，clear 以及

1、基



(Flex Container)

)

2、容器属性

该组属性，要设置在容器(container)元素上，用于统一的控制子元素的排列方式

1、flex-directic

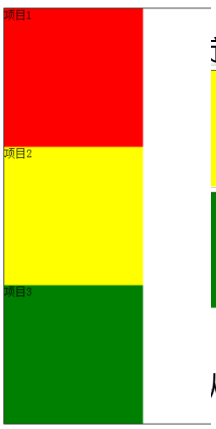
作用：指 定项目的排列方式)

取值：

1、rc

2、rc

3、cc



轴），起点在顶端

4、column-reverse

主轴为交叉轴（纵轴），起点在底

2、flex-wrap

作用：一行内显示不下所有项目时，如何换行

取值：

1、nowrap

默认值，不换行，会将项目等比缩放

2、wrap

换行

3、wrap-reverse

换行，第一行在下方

3、flex-flow

作用：该属性是 flex-direction 和 flex-wrap 的缩写

取值：

1、默认值

row nowrap

2、direction wrap

4、justify-content

作用：该属性定义了项目在主轴上的对齐方式

注意：根据主轴的方向来决定效果

取值：

1、flex-start

主轴的起点对齐

2、flex-end

主轴的终点对齐

3、center

居中对齐

4、space-between

5、sp

练习：主轴为 交叉轴时，justify-content 是如何排列的？？

5、align-items

作用：定义项目在交叉轴上是如何排列对齐的（主要针对一行项目）

取值：

1、flex-start

交叉轴起点对齐

2、flex-end

3、center

4、baseline

基线对齐，第一行文本的基线(下方)

5、stretch

默认值，如果项目未设置高度或高度为 auto 时，那么将占满整个容器的高度

6、align-content

作用：容器具备多跟轴线时(自动换行时)，项目的对齐方式。

取值：

1、flex-start

项目在交叉轴起点对齐

2、flex-end

项目在交叉轴终点对齐

3、center

项目在交叉轴中间对齐

4、space-between

与交叉轴两端对齐,轴线之间的间隔相等

5、space-around

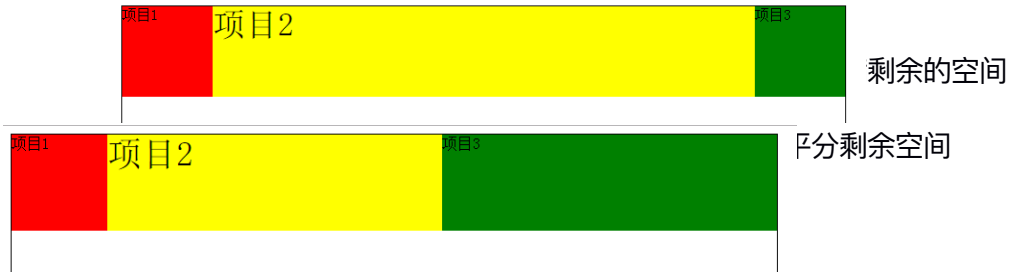
每个轴线两端的间隔相等,轴线间间隔比轴线与父元素边框间隔大一倍

6、stretch
默认值，在不指定项目高度时，占满整个交叉轴

3、项目属性

1、order
作用：排序，定义项目的排列顺序。值越小，越靠前
取值：
默认为 0，
自定义数值

2、flex-grow
作用：定义放大比例，如果当前排列轴有空余空间，项目将如何放大
取值：正数数字



3、flex-shrink
作用：该属性定义了项目的缩小比例，默认为 1，即空间不足时，等比缩小
取值：数字
如果取值为 0 ，空间不足时不缩小
如果取值为 1 ，空间不够时等比缩小

4、flex-basis
作用：指定项目所占据的空间大小
取值：
1、auto
默认值，项目本身大小
2、length，指定 宽 或 高(取决于主轴的方向)

5、flex
该属性是，flex-grow，flex-shrink，flex-basis 的简写模式
取值：
1、auto
相当于 ：1 1 auto
2、none
相当于 ：0 0 auto
3、flex-grow,flex-shrink,flex-basis

6、align-self
作用：单独定义当前元素与其他元素不一样的交叉轴对齐方式，可以覆盖 align-items 的值。默认为 auto，即继承父元素的 align-items
取值：
1、auto
自动，继承父元素 align-items
2、flex-start
交叉轴起点
3、flex-end
4、center
5、baseline

6、stretch

Day2

1、CSSHack

IE 浏览器兼容性
CSSHack : 针对不同的浏览器编写不同的属性

1、浏览器浏览模式
1、标准模式
完全按照 W3C 的标准显示网页
2、准标准模式
支持标准，但同时向前兼容非标准代码
3、混杂模式
不按标准解析内容，以 IE 自己的标准为主

不同的模式，浏览器对 CSS(框模型) 以及 Javascrpt 解析效果会有不同

在 HTML 中，可以通过 <!doctype> 进行模式选择
1、触发混杂模式
不声明 <!doctype>即可
采用 IE5.5 内核进行页面渲染
2、触发准标准模式

CSSHack 的原理:
使用样式属性的 “优先级” 来解决兼容性的问题
CSSHack 的实现方式
1、CSS 类内部 Hack
在 样式属性名称前 或 样式属性值后 增加一些前后缀, 来适配不同的浏览器
2、CSS 选择器 Hack
在选择器前 加前缀 识别不同的浏览器
* : IE6 能识别
*div{ ... }
*+:IE 能识别
..
3、HTML 头部引用 Hack(掌握)
通过 IE 条件注释完成 Hack，通过条件判断浏览器版本再执行相应内容
条件注释语法:
<!--[if 条件 IE 版本号]>
满足浏览器要执行的内容
<![endif]-->
版本号:
1、6~10
2、省略版本号，判断是否为 IE 浏览器
条件:
1、gt
判断当前浏览器是否大于指定版本的浏览器

```
<!--[if gt IE 8]>
    只有在 版本大于 IE 8 的时候才执行
<![endif]-->
```

2、gte
判断当前浏览器是否大于等于指定版本的浏览器

3、lt
判断当前浏览器是否小于指定版本浏览器

4、lte
判断当前浏览器是否小于等于指定版本浏览器

5、!
判断当前浏览器是否是非指定版本浏览器

6、省略条件
判断当前浏览器是否是指定版本浏览器

```
<!--[if IE 6]>
    只在 IE6 中执行
<![endif]-->
```

2、转换

1、转换介绍
改变元素在页面中的位置，大小，角度以及形状的一种效果

在 css 中允许向元素应用 2D 转换 或 3D 转换 效果
2D 转换：让元素在 x 轴 和 y 轴 做转换效果
3D 转换：在 2D 基础上，增加了对 z 轴 的转换效果

转换属性：
属性：transform
取值：
1、none ：默认值，无任何转换效果
2、一组转换函数
表示一个 或 多个转换函数，中间用 空格 分开

转换原点：
属性：transform-origin
取值：
1、取 2 个值 表示原点在 x 轴 和 y 轴上的位置
元素的左上角：
0px 0px 或
0% 0% 或
left top
2、取 3 个值 表示原点在 x，y，z 轴上的位置

默认原点位置：在元素的中心位置处(50% 50% 或 center center)

2、2D 转换
1、2D 位移
1、what
改变元素在页面中的位置
2、语法

属性：transform
函数：

1、translate(x);
在 x 轴的位移距离
x 值，取值为正，元素右移
x 值，取值为负，元素左移

2、translate(x,y)
增加了在 y 轴的位移距离
y 值，取值为正，元素下移
y 值，取值为负，元素上移

3、translateX(x)
4、translateY(y)

2、2D 缩放

1、what
改变元素的尺寸

2、语法
属性：transform
函数：

1、scale(value)
value：表示 x 轴 和 y 轴的缩放比例
取值：
1 ：原始大小
>1：放大比例
1>value>0：缩小的比例
负数：物极必反

2、scale(x,y)
分别指定 x 轴 和 y 轴的缩放比例

3、scaleX(x)
4、scaleY(y)

3、2D 旋转

1、what
改变元素在页面上的角度

2、语法
属性：transform
函数：rotate(ndeg)
n：具体旋转角度
取值为正，顺时针旋转
取值为负，逆时针旋转

3、注意
1、转换原点的位置会影响转换效果
2、旋转后，连同坐标轴也一起旋转
会影响旋转后的位移效果

4、2D 倾斜

1、what
改变元素的形状

2、语法
属性：transform
函数：

1、skew(xdeg)

x 轴倾斜，实际上时改变了"y 轴"的倾斜角度，以至于让元素产生向横向倾斜的效果

取值为正，y 轴按逆时针产生倾斜角度

取值为负，y 轴按顺时针产生倾斜角度

2、skewX(xdeg)

同上

ex:skew(30deg)

3、skewY(ydeg)

y 轴倾斜，实际上是改变"x 轴"的倾斜角度

4、skew(xdeg,ydeg)

同时让 x 轴，y 轴产生倾斜效果

3、3D 转换

3D 转换是在 2D 转换的基础上多了一根 z 轴 转换效果

1、属性

属性：perspective

作用：模拟人眼睛到 3D 投射平面的距离。取值越大，相当于离转换物体越远。取值越小，相当于离转换物体越近。

注意：该属性要加在 3D 转换元素 的父元素上

浏览器兼容性：

Chrome & Safari :

-webkit-perspective:...px;

2、3D 旋转

属性：transform

函数：

1、rotateX(xdeg)

以 x 轴为中心轴，旋转元素

2、rotateY(ydeg)

以 y 轴为中心轴，旋转元素

3、rotateZ(zdeg)

以 z 轴为中心轴，旋转元素

4、rotate3d(x,y,z,ndeg)

x,y,z 取值为 1，说明该轴要参与旋转

x,y,z 取值为 0，说明该轴不参与旋转

ex:

```
rotate3d(1,0,0,45deg);
rotate3d(1,1,0,45deg);
rotate3d(1,1,1,45deg);

transform:rotatex(45deg) rotatey(45deg) rotatez(45deg);
```

取值为正，都为顺时针旋转

取值为负，都为逆时针旋转

3、3D 位移

2D 位移基础上，增加在 z 轴上的位移距离

语法：

属性：transform

函数：

translateZ(z)

translate3D(x,y,z)

属性：

transform-style

取值：

1、flat

默认值，不保留其子元素的 3D 位置

2、preserve-3d

保留其子元素的 3D 位置

3、过渡

1、什么是过渡

过渡(transition),使得 CSS 属性值在一段时间内平滑过渡的效果

2、过渡的要素 & 属性

1、过渡属性

1、作用

指定 哪个属性值 在变化的时候采用过渡的效果体现出来

2、语法

属性：transition-property

取值：

1、none ：默认值，任何属性值变化时都不采用过渡效果

2、all ：所有，任何属性值在变化时，都采用过渡效果

3、propertyName ：要采用过渡效果的属性名称

ex:

背景颜色变化时，将采用过渡效果实现

transition-property:background-color;

3、允许使用过渡的属性

1、颜色属性

2、取值为数值的属性

width,height,margin,left,top,bottom ..

3、转换属性

transform

4、渐变

5、visibility

6、阴影

2、过渡时间

1、作用

指定过渡效果在多长时间内完成

2、语法

属性：transition-duration

取值：以 s 或 ms 为单位数值

l000ms = ls

注意：默认值为 0 ， 即无任何过渡效果产生

ex: 让 过渡效果在 2s 内完成

transition-property:background-color;

transition-duration:2s;

3、过渡的速度时间曲线函数(速率)

1、语法

属性：transition-timing-function

取值：

1、ease
默认，先慢 后快 再慢

2、linear
匀速

3、ease-in
慢速开始，加速结束

4、ease-out
快速开始，减速结束

5、ease-in-out
先慢 后快 再慢

ex：指定变化速率为匀速

```
transition-property:background-color;
transition-duration:2s;
transition-timing-function:linear;
```

4、过渡延迟

1、作用
激发操作后，等待多长时间后，再执行变化效果

2、语法
属性：transition-delay
取值：以 s 或 ms 为单位的数值

3、简写属性

```
transition:property duration timing-function delay;

transition:property1 duration1 timing-function1 delay1,property2 duration2;
```

Day3

1、动画(animation)

1、什么是动画
使得元素从一种样式逐渐变化为另一种样式的过程
动画是复杂的过渡
动画是通过 “关键帧” 来控制每个步骤
关键帧：由时间 和 动作来组成

2、动画的使用步骤
1、声明动画
1、为动画定义名称
2、定义动画的所有关键帧
2、为元素调用动画

3、声明动画的语法
通过 @keyframes 的规则来声明动画

```
@keyframes 名称{
    /*声明 若干 关键帧 （时间 ： 动作(样式)）*/
    0% | from{
        /*动画开始时的动作(样式)*/
    }
    25%{
        /*动画运行在 1/4 时的动作*/
    }
```

100% | to{
/*动画结束时的动作(样式)*/
}
}
兼容性
@-webkit-keyframes 名称{}
@-moz-keyframes 名称{}
@-o-keyframes 名称{}
@-ms-keyframes 名称{}

4、调用动画的语法

1、animation-name
指定当前元素 调用的动画名称

2、animation-duration
指定当前动画执行的总时长(一次动画过程)
取值：以 s 或 ms 作为单位的数值

3、animation-timing-function
指定动画执行时的速度时间曲线函数
ease,linear,ease-in,ease-out,ease-in-out

4、animation-delay
延迟

5、animation-iteration-count
指定动画的播放次数
取值：
1、具体数值
2、infinite
无限次播放

6、animation-direction
指定动画的播放方向
取值：
1、normal
默认值，从 0% ~ 100%
2、reverse
从 100% ~ 0%
3、alternate
轮流播放
奇数播放次数时，从 0%~100%
偶数播放次数时，从 100%~0%

7、动画简写
animation:name duration timing-function delay iteration-count direction;

8、animation-fill-mode
指定动画播放前后的填充模式
取值：
1、none
默认行为
2、forwards
动画完成后，保持在最后一个帧的状态上
3、backwards
动画播放前(延迟时间内)，保持在第一个帧的状态上
4、both

动画播放前后分别保持在第一帧和最后一帧的状态上

9、animation-play-state
指定动画的播放状态(暂停 | 播放)
取值：

1、paused
动画暂停

2、running
动画播放

Bootstrap

DAY1

1、开发工具
1、记事本，Editplus,... ...
2、Sublime，Dreamweaver
3、Webstorm

==========

1、什么是响应式网页

Responsive Web Page,响应式网页/自适应网页,即一个页面既可以在 PC 浏览器中浏览，也可以在手机/平板中浏览。并且配合不同设备有不同的响应结果

响应式网页的特点：

1、页面上的图片和文字要随着屏幕尺寸发生改变
屏幕分辨率：1211px ，图像 ： 1200px

2、页面的布局随着屏幕尺寸而发生改变

2、如何测试响应式网页

1、使用真实的物理设备
优势：测试结果真实可靠
不足：设备太多，成本太大，测试任务量大

方式：

1、搭建本地服务器，部署项目
2、移动终端设备与服务接入相同网络

2、使用第三方模拟设备
优势：无须添加更多设备
不足：效率偏低

3、使用浏览器自带的设备模拟器(Emulator)
优势：功能丰富
不足：... ...

3、视口 - Viewport

IOS 中的 Safari 最早引入的概念

视口：移动设备中，浏览器里显示网页的一块区域(PC 端会忽略)

对于响应式网页，设置视口的信息：

1、视口的宽度 ： 要与设备宽度一致

2、视口的缩放倍率 ： 设置为 1，即不缩放

3、视口的手动缩放 ： 不允许缩放网页

在 HTML 中指定视口信息：

<meta name="viewport" content="">

1、视口的宽度 ： width

取值：

1、device-width

2、具体数值

2、视口的初始缩放倍率 : initial-scale

取值：缩放倍数

1 : 原始大小

3、是否允许视口手动缩放 : user-scalable

取值：

1/0/yes/no

<meta name="viewport" content="width=device-width,initial-scale=1,user-scalable=no">

以上代码，移动端必备!!!

4、如何编写响应式网页 (重点)

1、必须声明视口(已解决)

2、文字使用相对尺寸(em,rem)，尽量不用绝对尺寸(px)

CSS 中的 1px 并不代表真实物理设备的 1px

比如：iPhone4 以后，屏幕为 Retina 屏幕，屏幕大小没有变化，但分辨率提升一倍，

iphone4 ： 屏幕 320 * 480

分辨率：640 * 960

em : 父元素字体大小倍数

rem : 根元素(html)字体大小倍数

3、容器元素使用相对尺寸(%,auto)，尽量不用绝对尺寸(px)

4、图片使用相对尺寸(%,auto)，尽量不用绝对尺寸(px)

```
img{
    width:50px;/*不推荐*/
    width:50%;推荐使用，指定父容器的占比，可以无限缩放
    max-width:50%;推荐使用，指定父容器的占比，最大不能超过图像原始大小

}
```

5、页面元素使用流式布局

流式布局特点：

1、元素默认靠向容器的左上方

2、横向排列，排列不下则换行

方法 1：float

方法 2：display:inline-block;

6、响应式网页都要使用 CSS3 Media Query 技术 - 最重要

ex:

1、浏览器宽度 w < 768，背景色 红色

2、浏览器宽度 768<= w <=991 背景色 绿色

3、浏览器宽度 w > 991 背景色 粉色

5、CSS3 Media Query

作用：可以根据不同的媒体类型以及特性执行不同的 CSS

Media ：媒体，指浏览网页设备的类型 如：screen(PC/Pad/Phone),tv,tty

语法：

通过 @media 规则进行声明

@media MEDIA-TYPE and|not|only (MEDIA-FEATURE)

MEDIA-TYPE:媒体类型

取值：

1、all，默认值，所有设备

2、screen：电脑屏幕，智能手机，平板电脑

3、tv ：电视设备

MEDIA-FEATUER:媒体特性

取值：

1、width ：指定浏览器窗口宽度大小

2、min-width ：指定浏览器窗口宽度最小值

3、max-width ：指定浏览器窗口宽度最大值

ex：

1、在屏幕下，宽度在 767 以内，执行操作

@media screen and (max-width:767px){}

2、在屏幕下，宽度在 768~991 之间，执行操作

@media screen and (min-width:768px) and (max-width:991px){}

@media 的用法

1、有选择性的执行某个外部 CSS 文件

<link rel="" href="" media="screen">

<link rel="" href="" media="screen and (max-width:767px)">

不足：

即使不满足当前设备条件的 CSS 文件也会被请求，但不会生效

2、有选择性执行 CSS 片段中的内容

在样式表中

@media screen and (max-width:767px){

选择器{属性:值;}

}

常见屏幕尺寸：

1、超小屏幕(Extra Small : xs)

w <= 767

例如：手机屏幕

2、小型屏幕(Small : sm)

768 <= w <= 991

例如：pad

3、中型屏幕(Medium : md)

992 <= w <= 1199

例如：电脑屏幕

4、大型屏幕(Large : lg)

w >= 1200

例如：分辨率较大的电脑屏幕

DAY2

1、Twitter Bootstrap

官网：http://getbootstrap.com
中文官网：http://www.bootcss.com

重点 bootstrap.css - 提供了上千个 class
依赖于 JS 库 - jQuery

Bootstrap 分为 5 部分
1、起步
2、全局 CSS 样式
3、组件
4、JS 插件
5、定制

2、Bootstrap 第一步 - 起步

基本模板：

1、<html lang="zh-cn">
指定当前文档的基础语言,zh-cn,zh-tw,jp,en
作用：
1、为浏览器的自动翻译功能指定语言基础
2、为读屏软件指定基础发音
2、<meta name="viewport"> 必须
3、<meta http-equiv="x-ua-compatible" content="IE=edge">
x-ua-compatible
Cross UserAgent Compatible
跨(IE)浏览器兼容性
作用:指定用 哪个 IE 的内核进行页面渲染
IE=6 : 指定用 IE6 内核渲染页面
IE=7 : 指定用 IE7 内核渲染页面
IE=8 :
IE=9
...
IE=edge : 采用 IE 最新版内核渲染页面
4、两个 JS
1、html5shiv.min.js
第三方的 JS，自调函数，用于让老 IE(IE8 及以下)支持 HTML5 新标记,header,footer,aside
2、respond.min.js
第三方的 JS，自调函数，用于让老 IE(IE8 及以下)支持 CSS3 媒体查询技术 - 响应式必备

通过头部引用 Hack 判断是否为 IE8 以及以下的浏览器
<!--[if lt IE 9]>

```
        <script src="***.js"></script>
        <script src="***.js"></script>
    <![endif]-->
```

5、两个 JS

1、jQuery.js 引入到页面中(先)

2、bootstrap.js 引入到页面中(后)

建议：尽量将以上两个文件放在页面最底端引入

3、Bootstrap 第二步 - 全局 CSS 样式

bootstrap.css 内容分为两部分

1、CSS Reset

```
*{box-sizing:border-box;}
html{font-size:10px;}
body{margin:0;font:... ...;}
h1{}
...
h6{}
p{}
```

2、上千个 class

```
.btn    .btn-default    .btn-danger
.container
.container-fluid
... ...
```

4、Bootstrap 默认将屏幕分成四大类

1、大型 PC 屏幕(lg) : w >= 1200px

2、中型 PC 屏幕(md) : 992px <= w <= 1199px

3、小型 PAD 屏幕(sm): 768px <= w <= 991px

4、超小型 PHONE 屏幕(xs):w <= 767px

5、Bootstrap 提供的两种容器

1、定宽容器

在不同大小的设备上提供不同的 width 固定值

类：.container

```
lg : width:1170px
md : width:970px
sm : width:750px
xs : width:100%
```

2、变宽容器

在任何设备中，宽度都是 100%

类：.container-fluid

```
width:100%;
```

6、Bootstrap 第二步 - 全局 css 样式-按钮

.btn

.btn-default 白底深色字
.btn-danger/success/warning/info/primary
.btn-lg/sm/xs
.btn-block 块级按钮

7、Bootstrap 第二步 - 全局 css 样式-列表

.list-unstyled 不带标识的列表
.list-inline 行内列表
.dl-horizontal 定义列表

8、Bootstrap 第二步 - 全局 css 样式-图片

.img-rounded ?
.img-circle ?
.img-thumbnail ?
.img-responsive ?

9、Bootstrap 第二步 - 全局 css 样式-表格

.table
.table-bordered 带边框表格
.table-striped 隔行变色表格
.table-hover 待悬停效果的表格
.table-responsive 响应式表格（为表格父元素添加）

10、Bootstrap 第二步 - 全局 css 样式-文本 & 排版

1、文本颜色相关
.text-danger/success/warning/info/primary
.bg-danger/success/warning/info/primary
2、文本大小写
.text-uppercase/lowercase/capitalize
3、文本对齐方式
.text-left/center/right/justify

11、Bootstra 第二步 - 全局 css 样式 - 栅格布局

页面中可以实现布局的技术
1、table 布局
好处：简单，容易控制
不足：效率低
2、div + css
好处：效率高
不足：灵活，不易控制
3、栅格布局
好处：效率高，容易控制，实现响应式
不足：没有
栅格布局实际上就是由 div 组成的 table 样式的响应式结构

使用方法：

1、栅格布局系统的最外层，必须是 bootstrap 提供的容器

.container 或 .container-fluid

2、允许在容器中放置若干行 div.row

每行中最多等分为 12 列

3、行中放置 div.col 即列，每列都需要指定宽度 1/12,2/12

语法：

```
<div class="container">
    <div class="row">
        <div class="col-*-*"></div>
    </div>
</div>
```

4、列 根据适用屏幕分成四中类型

.col-xs-*

.col-xs-1 : 在超小屏幕中，占一列的宽(8.33%)

.col-xs-2 : 在超小屏幕中，占两列的宽(16.66%)

... ...

.col-xs-12 : 在超小屏幕中，占 12 列的宽(100%)

.col-sm-*:

在 小型 屏幕中 所占列宽数

.col-md-*

在 中型 屏幕中 所占列宽数

.col-lg-*

在 大型 屏幕中 所占列宽数

5、列偏移数量

每个 列 都可以指定向右偏移几列位置

.col-xs-offset-n : 在 xs 下，当前列向右偏移 n 列的距离

.col-sm-offset-n : sm 下，向右偏移 n 列

.col-md-offset-n :

.col-lg-offset-n :

6、栅格布局系统可以嵌套

.container>.row>.col-*-*>.row>.col-*-*

```
<div class="container">
    <div class="row">
        <div class="col-md-3">
            <div class="row">
                <div class="col-md-1">MD-1</div>
            </div>
        </div>
    </div>
</div>
```

7、适用于不同屏幕的列的 class(xs/sm/md/lg) 可以兼容更大的屏幕

```
<div class="container">
    <div class="row">
        <div class="col-xs-3"></div>
        <div class="col-md-3"></div>
    </div>
</div>
```

.col-xs-* : 适用于 xs/sm/md/lg
.col-sm-* : 适用于 sm/md/lg
.col-md-* : 适用于 md/lg
.col-lg-* : 适用于 lg

大屏幕 class 在小屏幕中，永远是垂直显示

8、可以在一个 div 中指定在不同屏幕下的宽度占比

在 xs 中 占 9 列宽
在 sm 中 占 6 列宽
在 md 中 占 3 列宽

9、指定列在特定屏幕下不显示

.hidden-lg : 在 lg 下隐藏
.hidden-md : 在 md 下隐藏
.hidden-sm : 在 sm 下隐藏
.hidden-xs : 在 xs 下隐藏

DAY3

1、全局样式 - 表单

遵循 HTML5 的规范,共三种

1、默认表单(垂直排列)

.form-group : 定义表单控件组
.form-control ： 定义表单控件
.control-label ： 定义控件对应的 label
.help-block ： 定义提示文本

2、行内表单

为 <form> 添加 class="form-inline"
其他同上

3、水平表单

水平表单 = 表单 + 栅格布局系统
栅格 ：
最外层：.container / .container-fluid
行：div.row
列：div.col-*-*
水平表单栅格系统
最外层：form.form-horizontal / .container
行：div.form-group / div.row
列：div.col-*-*

```
<label class="col-md-3"></label>
<div class="col-md-6">
    <input class="form-control">
</div>
<span class="col-md-3"></span>
```

或

```
<div class="col-md-3">
    <label>
```

```
</div>

<div class="col-md-6">
    <input class="form-control">
</div>

<div class="col-md-3">
    <span class="help-block">
</div>
```

2、Bootstrap 第三部分 - 组件

1、下拉菜单

```
<select>
    <option></option>
</select>
```

1、外层必须是 .dropdown/.dropup 或 position:relative;

2、内层：为 <button> 或 <a>

class="dropdown-toggle"

data-toggle="dropdown" ：切换内容的显示和隐藏

3、内层：

内容 <ul> 或

class="dropdown-menu"

li.divider : 分割线效果

li.disabled : 禁用菜单项

li.dropdown-header:标题

2、导航

1、标签页式导航

```
<ul class="nav nav-tabs">
    <li class="active">
        <a href="#" data-toggle="tab">....</a>
    </li>
</ul>
```

li.active : 默认被激活

li.data-toggle="tab" ：允许切换并且指定切换方式

3、图标字体

在页面中，显示为图标，本质上是文字，可以设置字体，颜色，大小，阴影。。。。

Web 程序中常用的图标字体：

1、Glyphicons 收费

2、FontAwesome 675 个 免费

由于客户端不具备 bootstrap 中的图标字体，所以使用自定义的 图标字体，必须声明

在服务器端做以下操作：

***.css

```
/*声明字体*/
@font-face{
```

font-family:"名称";//自定义名称
src:url('地址 ');//图标字体文件所在路径
}
/*对使用字体图标的选择器进行声明*/
.glyphicon{
font-family:"名称";
}
使用方法:
必须保证是空元素

练习:
创建多个按钮，在每个按钮上显示一个小图标
首页(房子),配置(齿轮),刷新,定位，购物车，放大镜，发邮件，照相机，播放相关(播放，暂停，上一曲，下一曲)

4、按钮组
将多个按钮放在一个组中(btn-group)
1、基本按钮组

<button class="btn btn-default">xx</button>
<button class="btn btn-default">xx</button>

2、将一组 .btn-group 组合进一个 .btn-toolbar(按钮工具栏)

<button></button>

3、按钮组尺寸
为 .btn-group 增加 .btn-group-* 类 设置尺寸

4、按钮组的嵌套
在一个 btn-group 中嵌套另一个 btn-group
5、两端对齐按钮组
.btn-group .btn-group-justified
6、垂直放置的按钮组
.btn-group-vertical

5、警告框
允许将任意字符 与 可选的关闭按钮 组合在一起的结构
所有警告框 依赖于 .alert
.alert
.alert-success
.alert-danger
...
1、允许关闭的警告框
.alert-dismissible

lorem

```
        <button class="close" data-dismiss="alert">&times;</button>
    </div>
```

2、警告框中的链接

```
    <a class="alert-link"></a>
```

6、面包屑导航/路径导航

.breadcrumb

```
<ul class="breadcrumb">
    <li>
        <a href="#">首页</a>
    </li>
    <li>
        <a href="#">产品大全</a>
    </li>
</ul>
```

7、分页条

.pagination

.active：被激活的页码

```
<ul class="pagination">
    <li>
        <a href="#">上一页</a>
    </li>
    <li>
        <a href="#">1</a>
    </li>
    <li class="active">
        <a href="#">2</a>
    </li>
    <li>
        <a href="#">...</a>
    </li>
    <li>
        <a href="#">下一页</a>
    </li>
</ul>
```

8、分页器

.pager

```
<ul class="pager">
</ul>
```

9、标签

所有的标签都依赖于 .label

.label

.label-default

.label-danger

.label-success

...

```
<span class="label label-danger">标签内容</span>
```

10、徽章

.badge

```
<span class="badge">35</span>
```

11、巨幕

.jumbotron

12、页头

允许为 标题元素 增加适当的空间,与其他元素有一定的间隔

.page-header

13、Well (水井)

.well

14、进度条

外层：.progress

内层：

.progress-bar

.progress-bar-danger

.progress-bar-success

...

.progress-bar-striped

.active： 被激活的

通过 给内层元素 增加 style="width:50%" 增加宽度

15、缩略图

.thumbnail

.caption

```
<div class="thumbnail">
    <img>
    <div class="caption">
        <p>文本 1</p>
        <p>文本 2</p>
        <p>按钮</p>
    </div>
</div>
```

16、媒体对象

```
<div class="media">
    <div class="media-left">
        <img>
    </div>
    <div class="media-body">
        <h2 class="media-heading">标题</h2>
        形容的文本
    </div>
    <div class="media-right">
        <img>
    </div>
</div>
```

17、列表组

ul : class="list-group"

li : class="list-group-item"

18、面版

呈现头部 主体 尾部 结构的组件

```
<div class="panel panel-default panel-primary">
    <div class="panel-heading">
```

```
            <h2 class="panel-title">标题文本</h2>
        </div>
        <div class="panel-body">
            主体内容
        </div>
        <div class="panel-footer">
            脚注内容
        </div>
    </div>
```

DAY4

1、组件 - 导航条

1、基本导航条

1、向 nav 元素添加 class.navbar .navbar-default

有需要的话，允许添加 div.container

2、向 nav 元素中添加 div.navbar-header ，内部允许包含 class 带有 .navbar-brand <a>元素

3、允许向导航条中添加链接列表 ，只需要添加 class.nav .navbar-nav 的列表即可

2、导航条中的表单

导航中的表单不适用 bootstrap 中默认 class

使用的时 .navbar-form (具备垂直对齐效果),配合 .navbar-left / .navbar-right

3、导航条中的按钮

class.navbar-btn 允许向不在 form 中的 button(a)增加样式(垂直居中)

4、导航条中的文本

普通文本的话，需要增加 class .navbar-text 属性来保证格式

5、组件的对齐方式

允许通过 .navbar-left 实现左浮动

允许通过 .navbar-right 实现右浮动

6、导航栏的固定

不会随着滚动条发生滚动，一直在可视化区域中

固定在页面顶端：.navbar-fixed-top

固定在页面底端：.navbar-fixed-bottom

注意：最好为 body 设置内边距至少 50 px

2、JS Plugin

Bootstrap 基于 jQuery ，在 jQuery 基础上提供了 十几个 插件函数，每个都是一个独立的 JS 文件,可以一次性引入全部的 JS 操作 - bootstrap.js

每个插件函数都有两种调用方式：

1、data-* 方式调用

<a data-toggle="dropdown">

2、JS 编程方式

手动编写 JS 代码完成行为的调用

<script>

$("选择器") : 在 Javascript(jQuery) 中获取页面指定选择器的元素

$("#id")

$(".class")

$("div p")

下拉列表：

$("选择器").dropdown();

</script>

3、警告框

父元素 class ="alert alert-danger alert-dismissible"

关闭：<button class="close" data-dismiss="alert"></button>

JS :

$("关闭选择器").click(function(){

$(this).alert("close");

});

4、按钮

1、设置按钮的操作文本

为 按钮元素 添加 data-loading-text="显示的文本"

<button class="btn btn-default" data-loading-text="请稍后....">文本</button>

//点击按钮时，改变按钮的文字为 data-loading-text

$("按钮选择器").click(function(){

$(this).button("loading");

});

2、设置 单选按钮 / 复选框

1、将 若干 单选按钮 / 复选框 放到 btn-group 中 ， 为 btn-group 增加属性 data-toggle="buttons"

5、工具提示

为元素增加

data-toggle="tooltip"

data-palcement="top/right/bottom/left"

title="提示的文本"

配合 JS 代码

$("选择器").tooltip();

6、弹出框

为元素增加

data-toggle="popover" //指定为弹出框方式

data-placement="top/right/bottom/left"//方向

data-content="弹出框内容区域的文本"

title="弹出框的标题";

配合 JS 代码如下:

$("选择器").popover();

7、标签页

1、为导航组件里面 a 增加：

1、data-toggle = "tab"

2、href="#对应元素内容的 ID"

2、创建内容组

1、class 为 tab-content

2、在 内容组中 增加对应显示的内容模块

1、增加 id 属性

2、增加 class="tab-pane active"

ex:

```
<div class="container">
    <ul class="nav nav-tabs">
        <li class="active">
            <a href="#tab1" data-toggle="tab"></a>
        </li>
    </ul>
    <div class="tab-content">
        <!-- 内容 1 -->
        <div class="tab-pane" id="tab1">
            ...
        </div>
    </div>
</div>
```

8、模态对话框

模态对话框：父窗口中弹出一个子窗口，只要子窗口不关闭，父窗口就无法获得输入的焦点

模态对话框由两部分组成：

1、触发元素，通常 a / button 组成

```
<a href="#模态框 ID" data-toggle="modal"></a>

<button data-toggle="model" data-target="#模态框 ID"></button>
```

2、模态框元素

```
// 提供了半透明的遮罩层
<div class="modal" id="" data-backdrop="static">
    // 提供了 宽度，高度，定位
    <div class="modal-dialog">
        //背景色，边框，倒角，阴影
        <div class="model-content">
            <div class="modal-header">
                <h4></h4>
                <button> ... </button>
            </div>
            <div class="modal-body">
                显示主题内容
            </div>
            <div class="modal-footer">
                脚注信息
            </div>
        </div>
```

```
            </div>
        </div>
```

9、折叠效果

1、触发元素

```
<a data-toggle="collapse" href="#id"></a>

<button data-toggle="collapse" data-target="#id"></button>
```

2、被折叠元素

```
<div class="collapse" id="id">
    ... ...
</div>
```

特殊效果：

1、手风琴(Accordion)

Accordion = 面板组(panel-group) + 折叠插件

DAY5

1、组件 - 导航栏

1、基本导航栏

```
<nav class="navbar navbar-default">
    <div class="navbar-header">
        <a href="#" class="navbar-brand">
            Text Or Img
        </a>
    </div>
</nav>
```

2、导航栏中的表单

```
<form class="navbar-form"></form>
```

3、导航栏中的按钮

```
<button class="navbar-btn"></button>
```

4、导航栏中的独立文本

```
<p class="navbar-text">Text</p>
```

5、内容对齐方式

navbar-left / navbar-right

6、导航栏固定

顶部：.navbar-fixed-top(body 给至少 50px 上内边距)

底部：.navbar-fixed-bottom

2、JS-插件(Plugin)

1、行为调用

1、data-*

2、$("选择器").dropdown();

2、警告框

```
<div  class="alert-dismissible"  >
    <button data-dismiss="alert">&amp;times;</button>
```

```
</div>
```

3、按钮

1、点击按钮 改变文本及状态

```
<button data-loading="Text"></button>

$("button").click(function(){
    $(this).button("loading");
    //业务逻辑处理
    $(this).button("reset");
});
```

2、单选按钮和复选框

```
<div class="btn-group" data-toggle="buttons">
    <button class="btn btn-success">
        <input type="radio">
    </button>
</div>
```

4、工具提示

为元素增加

```
data-toggle = "tooltip"
data-placement="top/right/bottom/left"
title="提示的文本"
```

配合 JS 代码:

```
$("[data-toggle='tooltip']").tooltip();
```

5、弹出框

伪元素增加

```
data-toggle="popover"
data-placement="top/..."
title="弹出框的标题"
data-content="内容"
```

配合 JS 代码

```
$("[data-toggle='popover']").popover();
```

6、标签页

为 导航 组件 a 标记 增加

```
data-toggle="tab"
href="#对应的内容元素 ID"
```

增加 内容组元素

为内容增加

```
ID :
class ： .tab-pane  .active
```

7、模态对话框

弹出一个子窗口，如果不处理的话无法进行其他操作

1、触发元素

```
<a> / <button>
属性:
    data-toggle="modal"
    href="#对应打开的元素 ID"
```

2、模态框元素

```
<div class="modal">
```

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

8、折叠(Collapse)

1、触发元素

```
<a href="#折叠元素的 ID" data-toggle="collapse">
<button data-target="#折叠元素的 ID" data-toggle="collaps"></button>
```

2、折叠元素

```
<div id="" class="collapse">
    内容
</div>
```

折叠组件的扩展应用：

1、Accordion

面版组(panel-group) + 折叠插件

```
<div class="panel-group" id="">
    <div class="panel">
        <div>
            <a href="# 展开元素的 ID" data-parent="panel-group 的 ID"></a>
        </div>
    </div>
</div>
```

****

1、折叠

1、响应式导航条

当屏幕尺寸大于 768px 时候，可以正常显示出所有的内容，当屏幕尺寸小于 768px 的时候，一部分内容就会隐藏，通过点击弹出

响应式导航条由 两部分 组成

1、class .navbar-header

用于显示 navbar-brand 和 折叠点击按钮

.navbar-brand：定义 brand 内容

折叠按钮：

屏幕大于 768px 时不显示

屏幕小于 768px 时显示

class .navbar-toggle 完成以上操作

2、class .navbar-collapse

被折叠的内容，当屏幕大于 768px 正常显示

屏幕尺寸小于 768px，隐藏通过 按钮 点击完成展开显示

2、广告轮播（Carousel）

1、基本

```
<div class="carousel" data-ride="carousel">
```

```
<div class="carousel-inner">
    <div class="item">
        <img>
    </div>
</div>
</div>
```

2、指定轮播时间

```
<div class="carousel" data-ride="carousel" data-interval="2000">
    <div class="carousel-inner">
        <div class="item">
            <img>
        </div>
    </div>
</div>
```

3、带说明文本的轮播

```
<div class="carousel" data-ride="carousel" data-interval="2000">
    <div class="carousel-inner">
        <div class="item">
            <img>
            <div class="carousel-caption">
                <h4>标题</h4>
                <p>文本</p>
            </div>
        </div>
    </div>
</div>
```

4、带方向按钮的轮播

```
<a href="#Carousel 的 ID" class="carousel-control left" data-slide="prev">
    <span class="glyphicon glyphicon-chevron-left"></span>
</a>
<a href="#Carousel 的 ID" class="carousel-control right" data-slide="next">
    <span class="glyphicon glyphicon-chevron-right"></span>
</a>
```

5、带圆点导航的轮播

```
<ul class="carousel-indicators">
    <li class="active"
        data-target="#Carousel 的 ID"
        data-slide-to="0"></li>
</ul>
```

==========

Less 和 Bootstrap 定制

1、样式语言的分类

1、静态样式语言 ： CSS

可以被浏览器直接解析处理；但 CSS 并不是合格的语言，缺少了基本编程的要素，如：变量，运算符，函数。。。。 由于不是动态的，所以导致了 CSS 的可维护性差

2、动态样式语言 ：

如： Less , Sass/SCSS , Stylus ...

不可以被浏览器直接解析处理；必须经过编译(Compile)得到 CSS 文件后，才能使用。会具备语言的基本要素 ： 变量 。。。极大的提高的 CSS 代码的可维护性

2、Less 语言

官网：http://lesscss.org

中文网：http://less.bootcss.com

Less 是一本 CSS 预处理语言，它扩充了 CSS，在纯静态的 CSS 基础上增加了一部分内容 如：变量，混合(Mixin) ... 对静态 CSS 完全支持 100%兼容

在 Web 项目中使用 Less 的两种方式：

1、在客户端浏览器中编译 Less - 不推荐使用

1、编写 xx.less 文件

2、编写 xx.html，引入 xx.less;再引入 less.js

3、浏览器访问 xx.html 会自行下载 xx.less,less.js 文件，并且在客户端进行编译转换成 xx.css

2、在服务器端编译 less - 推荐使用

1、编写 xx.less

2、在服务器端搭建 Less 编译器，把 xx.less 转换为 xx.css

3、再编写 xx.html 直接引入 xx.css 文件即可

3、搭建 Less 服务器端编译环境 - 重点&乱点

Less 编译器 实际上是由 Javascript 编写的

1、安装独立的 JS 解释器 - node.exe

命令行中：执行 node -v 显示 ： 0.12.4 / 4.4.7

2、安装 Less 编译器程序 - less.js

确保： lessc.cmd 文件存在即可

3、在控制台中 测试 less 转换为 css

在 lessc.cmd 文件位置处 打开 控制台

输入 ： lessc.cmd D:\1.less > D:\1.css

4、在 WebStrorm 中 配置 FileWatchers(文件监视器)

由 WS 自动检测 less 文件的编写与更改，自动进行编译 得到 css 文件

配置 FileWatchers

WS --> File --> Settings --> Toos --> FileWatchers --> 添加 选择 Less --> 指定 lessc.cmd 文件地址即可

4、Less 语法

1、Less 完全支持 CSS 语法

2、Less 支持多行注释和单行注释，只有多行注释能被编译到 css 中

多行:/* */

单行：// 注释内容

3、Less 支持 变量(Variable)

变量：在 less 中可以变化的数据

语法：@变量名:值;

ex:

1、@jd_color:#e4393c; //颜色值

2、@border_width:5px; //数值

3、@base_font:"微软雅黑";//字符串

4、@border:1px solid #ddd;

使用变量：

变量是作为值，出现在 css 属性名称后的

@变量名;

ex:

```
#top{
    color:@jd_color;
```

```
        border-color:@jd_color;
    }
```

4、Less 变量可以 使用运算符

+,-,*,/,%

% : 取余数

5%2 结果：1

2%5 结果：2

5、Less 中支持 在一组样式中 混入 另一种样式(Mixin)

选择器 1{ ... }

选择器 2{ ... ;选择器 1;...;}

最终：选择器 2 中会包含该 选择器 1 定义好的样式

带参数的混合：

声明选择器的时候，允许使用 参数 来表示暂时不确定的数据，最终在调用时，要将具体的数值传递进来

语法：

```
选择器 1(@参数名 1,@参数名 2){
    widht:@参数名 1;
    height:@参数名 2;
}
```

使用带参的混合写法:

```
选择器 2{
    background:#fff;
    选择器 1(值 1,值 2);
}
```

6、嵌套规则

```
#top{}
#top p{}
#top p span{}
```

在 less 中，允许在一个选择器内再声明另一个选择器，以便完成 父子结构 或后代结构

语法：

```
选择器 1{
    ...;
    ...;
    选择器 2{
        ...;
        ...;
    }
}
```

最终编译成 CSS 的结果为：

```
选择器 1{
    ...
    ...
}
选择器 1 选择器 2{
```

```
        ...
        ...
    }
```

声明子代嵌套:

```
选择器 1{
    ...
    >选择器 2{
        ...
    }
}
```

7、Less 中提供的 功能函数

1、lighten(@color,20%) 返回一个变亮的颜色值(颜色减淡)

ex:

```
color:lighten(#e4393c,20%);
```

2、darken(@color,30%) 返回一个变暗的颜色(颜色加深)

3、image-width("xx.jpg") 返回指定图片的宽度

4、image-height("xx.jpg")返回指定图片的高度

5、ceil(@num) 对数字向上取整

ceil(55.787865235) --> 结果为 ： 56

6、floor(@num)对数字向下取整

floor(55.787865235) --> 结果为 ： 55

7、percentage(@num) 把小数转换为%数字

8、@import 功能

在 Less 中的@import ，在服务器端将多个 less 文件的内容整合到一个 less 文件中

语法:

xx.less

```
@import "xxx.less";
```

5、Bootstrap 定制

DAY6

复习

1、响应式导航条

```
<nav class="navbar navbar-default">
    <div class="navbar-header">
        <a class="navbar-brand"></a>
        <button class="navbar-toggle"></button>
    </div>
    <div class="navbar-collapse">
        /*...*/
    </div>
</nav>
```

2、广告轮播(Carousel)

```
<div id="ccc" class="carousel slide" data-interval="3000" data-ride="carousel">
    <div class="carousel-inner">
        <div class="item">
            <img>
        </div>
    </div>
    <!-- 控制按钮 -->
    <a href="#ccc" class="carousel-control left" data-slide="prev">
        <span class="glyphicon-chevron-left"></span>
    </a>
    <!-- 轮播导航 -->
    <ul class="carousel-indicators">
        <li class="active" data-slide-to="索引" data-target="#ccc"></li>
    </ul>
</div>
```

3、Less

1、什么是 Less

动态的样式语言，允许出现：变量，函数，运算符... ...

2、语法

1、Less 完全支持 CSS

2、变量

@变量名:值;

ex:

```
@jd_color:#e4393c;
@border:1px solid #ddd;
@font:"microsoft yahei";
```

使用变量

xx.less

```
.box{
    background-color:@jd_color;
}
```

3、运算符

+,-,*,/,%

ex:

```
@border-width:1px;
.box{
    border-width:@border-width+1;
}
```

4、混合（Mixin）

在一组样式中混入另一组样式

选择器 1{}

选择器 2{ 选择器 1; }

ex:

```
.width{
    width:200px;
}
.height{
    height:200px;
```

```
}

.box{
    .width;
    .height;
    border:@border;
}
```

编译结果为：

```
.box{
    width:200px;
    height:200px;
    border:1px solid #ddd;
}
```

带参数的混和

```
.box-shadow(@h:0px,@v:0px){
    box-shadow:@h @v;
}

.box1{
    .box-shadow();
}
```

编译后的结果：

```
.box1{
    box-shadow:0px 0px;
}

.box2{
    .box-shadow(2px,2px);
}
```

编译后的结果：

```
.box2{
    box-shadow:2px 2px;
}
```

5、嵌套

允许在一个选择器 嵌入 另一个选择器，以便完成父子结构 和 后代结构

```
选择器1{
    >选择器2{

    }
}
```

ex:

```
#top{
    width:200px;
    height:200px;
    .header{
        background:#ddd;
    }
}
```

编译后的结果：

```
#top{
    width:200px;
    height:200px;
}
#top .header{
    width:200px;
    height:200px;
    background:#ddd;
}
```

6、函数

1、lighten()
颜色 减淡

2、darken()
颜色加深

3、img-width()

4、ceil()
向上取整

5、floor()
向下取整

张东
zhangdong@tedu.cn

一本书：犀牛书
微信公众号：前端大全
上届的笔记：打印

正课：

1. 什么是 JavaScript
2. 变量/常量
3. 数据类型

JavaScript 基础

DAY1

1. 什么是 JavaScript

前端三大语言:
HTML 专门编写网页内容的语言
CSS 专门编写网页样式的语言
JavaScript 专门添加网页交互行为的语言

什么是交互: 3 步:
1. 用户的输入或操作
2. 接收用户输入的数据，并处理数据
   响应用户的操作
3. 返回处理结果

今后，所有静态页面必需添加交互行为后，才能让用户使用

何时使用 JavaScript:
1. 客户端数据计算:
2. 客户端表单验证:
3. 动画效果:

JavaScript 发展史:

鄙视: ECMAScript JavaScript JScript
ECMAScript: 由 ECMA 组织制定的 JavaScript 语言国际标准
JavaScript: NetScape 按照 ECMAScript 标准，实现的 JavaScript 语言版本
JScript: 微软按照 ECMAScript 标准，实现的 JavaScript 语言版本

JavaScript: 3 部分:
1. 核心语法: ECMAScript 7+5
   ES 版本: 3.1 + 5 + 6/2015
2. DOM 标准: 规定了专门操作网页内容的程序 2
3. BOM: 专门操作浏览器的程序 2

如何使用 JavaScript:
运行环境: 专门解释并执行程序语言的小软件
——脚本解释引擎
2 种:
1. 浏览器自带:
   包括 2 部分:
   内容排版引擎: 专门解释 HTML 和 CSS 等静态语言
   脚本解释引擎: 专门解释并执行 js 程序
   鄙视: 做动画选择 css vs js
   如果不需要交互行为的动画，首选 CSS，效率高
   如果必须交互行为的动画，只能用 js
2. 独立安装: nodeJS
   从 chrome 浏览器中剥离的独立的脚本解释引擎

以前：前端 js　后端 PHP/JAVA　数据库 SQL
现在：前端 js　后端 nodeJS　数据库 MongoDB

编写：

1. 在网页内编写：初学者
   所有 js 程序必需放在<script>内
   问题：不便于维护和重用
2. 在专门的外部 js 文件中编写，在网页中引入：网站项目
   在.js 文件中编写 js 程序
   在 HTML 文件中用<script src="url"></script>
   强调：一个 script 标签要么用于包含 js 代码，要么用于引入外部 js 文件。不能同时使用。否则，带 src 的 script 内部的 js 代码自动失效
   优：内容与行为分离，便于维护
3. 在浏览器控制台/nodeJS 中编写 js 指令：测试小程序
   客户端：浏览器控制台：专门编写并运行 js 小程序的窗口
   操作：按回车　执行
   shift+回车 仅换行，暂不执行
   上下键　切换写过的程序，反复修改使用
   清屏　左上角叉号/Ctrl+L

nodeJS: win+R->node
向控制台中输出一句话: console.log("...")
好处：调试的内容不影响页面布局
问题：编辑极其不方便
解决：在 WS 中编写独立的 js 文件
然后，选择在 node 中运行

基本语法规范：

1. 区分大小写：
2. 字符串可用''或""包裹都行
3. 每条语句必须以分号结束
4. 注释：
   //单行注释
   /*多行注释*/

程序：3 步：1.输入，2.处理，3.输出

1. 输入：向网页中输入一个值
   prompt("提示")
2. 输出：
   向页面中输出: document.write("...");
   向控制台输出: console.log("...");
   弹出提示框: alert("...")

对话框缺点：1. 样式不可修改
2. 会阻断程序继续执行以及用户操作

2. 变量：

什么是变量：内存中存储一个数据的存储空间，再起一个名字
何时：只要反复使用的数据，都要先保存在变量中，再反复使用变量。
如何：

声明：在内存中创建一个存储空间再起一个名字
何时：只要使用变量前，必须先声明变量
如何：var 变量名;
默认值：undefined
简写：var 变量 1,变量 2,...;
变量名：
1. 使用字母，数字，下划线组成，
不能以数字开头
2. 不允许使用 js 语言的保留字/关键词
3. 见名知义
4. 驼峰命名：
首字母小写，中间单词首字母大写
比如：backgroundColor:
listStyleType
赋值：将等号右边的值保存到等号左边的变量中
如何：变量名=值;
简写：
1. 声明同时，初始化变量的值：
var 变量名=值;
2. 同时声明并初始化多个变量：
var 变量 1=值 1, 变量 2=值 2,...;
特殊：
普通模式：强行给未声明变量赋值，不会报错！会自动创建该变量。
严格模式：比普通 js 运行模式要求跟严格的机制
1. 禁止给未声明的变量赋值
如何启用：在代码开始位置："use strict";
取值：任何情况下，使用变量名，等效于直接使用变量中的值。
特殊：强行从未声明的变量中取值，报错！
ReferenceError：要用的东西，没找到

常量：值不允许改变的量
何时：只要一个数据一旦定义，就不允许改变
如何：
声明：const 常量名=值;
强调：声明常量时，必须初始化
习惯上，常量名要全大写！
赋值：常量不允许在声明后再赋值
取值：常量的用法和普通变量完全一样
特殊：强行修改常量的值：
普通模式：不报错，但也不能修改——静默失败
严格模式：报错！
2. 将所有静默失败都升级为错误！

2. 数据类型：
什么是：数据在内存中的存储结构
为什么：为了让不同类型的数据，更便于执行专门的操作
包括：2 大类：
1. 原始类型：值直接保存在变量本地的数据类型
5 种：

number string boolean null undefined
2. 引用类型: 值无法保存在变量本地的数据类型

number: 存储一切数字
何时: 只要不带引号的数字直接量，默认就是 number
只要经常用作算数计算或比较的数值都不加引号
存储空间: 8 字节二进制(整数 4 字节)
1 字节(Byte)=8 位二进制数(Bit)
1KB=1024 字节 B
1MB=1024KB
将十进制数转二进制: n.toString(2);
强调: number 即包含整数，又包含浮点数(小数)

string: 保存一串字符组成的字符串
何时: 用于显示或记录一段文字
凡是用"或'包裹的一串文字，自动就是字符串类型
存储空间: 每个字符，数字，或英文标点占 1 字节
每个汉字占 2 字节
js 中采用 unicode 编码存储每个字符:
unicode 是对全球主要语言的每个文字编一个号
为什么: 计算机只认数字，不认字符
查看一个字的 unicode 号: str.charCodeAt();
数字 0~9: 48~57
大写字母 A~Z: 65~90
小写字母 a~z: 97~122
汉字: \u4e00~\u9fa5
转义字符: \
何时: 2 种:
1. 当字符串中包含和 js 语言标点符号冲突的内容，用\将冲突的字符转为原文，不再按程序解析
2. 要表示特殊意义:
\n 换行 \t 制表符 \u unicode
特殊情况: 路径中的\
2 种: 1. 前再加\ ——不建议，只有 windows 才认\
2. 将\换成/ ——强烈建议，任何操作系统都认

boolean: 布尔类型: 只有两个值的类型: true/false
何时: 专门用于表示一个判断的结论

null: 空 让程序员手动清空一个变量

undefined: 空 由程序自动使用，为变量赋初值

DAY2

正课:
1. 数据类型转换:
2. 运算符和表达式:

1. 数据类型转换：

什么是：将不需要的数据类型，转化为需要的数据类型

js 四大特点之一：弱类型：3 大特点：

1. 声明变量时，不需要提前指定变量的数据类型
2. 同一个变量先后可保存不同类型的数据
3. 不同数据类型间可相互转化

何时：只要给定的数据类型不是想要的

如何：2 种：

1. 隐式转换：不需要程序员干预，由 js 自动完成的数据类型转换

何时：只要给定的数据类型和 js 程序需要的类型不相符，js 就会根据自己的需要，自动转化数据类型。

比如：算数计算中：

默认：一切都转为 number 类型，再做计算

为什么：因为 number 类型最适合计算

bool：true->1　false->0

特殊：+运算中，只要碰到一个字符串，那么一切都转为字符串，+运算变为字符串拼接！

2. 强制转换：程序员主动调用函数完成的数据类型转换

何时：只要给定的数据类型不是想要的，且自动转换的结果也不是想要的，就要强制转换。

如何：

其他类型转 number：2 种：

1. Number(x)：将 x 转为 number 类型

何时：都是隐式转换，其实相当于自动调用 Number(x)，很少主动使用。

问题：只能转换纯数字组成的字符串和 bool 类型

如果转不了：就返回 NaN

NaN：不是一个数字的任何值 Not a Number

NaN 参与任何运算结果只能是 NaN

2. parseFloat/parseInt(str)：

将 str 转为 number 类型，自动去除末尾非数字字符

parseFloat 可保留小数部分

parseInt 去掉小数部分

何时：只要将字符串转 number，首选 parseFloat

如果确实需要去掉小数，才选 parseInt

如果转不了，也返回 NaN

强调：参数应该是 string 类型

如果给定的值不是 string 类型，则先执行隐式转换，转为 string 类型，再转 number

比如：parseFloat(true)

parseFloat("true")->NaN

其他类型转 string：2 种：

1. x.toString()　将 x 转为字符串类型

x 不能是 null 或 undefined ——不是万能

2. String(x)　将 x 转为字符串类型——万能

其实，隐式转为 string 时，都是自动调用 String

其他类型转 bool 类型：1 种

Boolean(x)

规则：只有 5 个值会被转为 false：

0,"",NaN,null,undefined
其余都转为 true
其实，隐式转为 bool 时，都是自动调用 Boolean(x)

练习中：
1. 凡是从页面上获得的一切都是字符串类型
2. typeof 变量 可输出变量中值的数据类型名

2. 运算符和表达式：
程序：人的想法在计算机中的执行
运算符：程序中模拟人的想法的特殊符号
表达式：由数据，变量和运算符组成的完成一项单一任务的语句。

算数运算：+,-,*,/,%
%：模运算/取余数：被除数/除数，不要商，要除不尽的余数部分
何时：1. 取余数
2. 判断能否整除
判断偶数：能被 2 整除 n%2==0
判断奇数：不能被 2 整除 n%2!=0
隐式转换：
默认：都转 number
特殊：+运算中，碰到字符串，都转字符串，+运算变为字符串拼接。
舍入误差：计算机中，也有计算不尽的数值
解决：
1. 按指定位数四舍五入：n.toFixed(2)
今后几乎所有显示钱数的地方都要 toFixed(2)
2. 存储：保存很长位数的小数：0.39999999

关系运算：用两个值做比较，做判断
包括：> < >= <= == !=
返回值：bool
隐式转换：
默认：都转 number
特殊：
1. 两个值都是字符串，则不再转数字，而是依次比较每个字符的 unicode 号
2. 判断 NaN：
NaN 和任何值做> < >= <= ==5 种比较时，永远 false
NaN 和任何值做!=比较是，永远是 true
问题：NaN==NaN =>false 用普通的==无法判断 NaN
解决：isNaN(num) 专门代替==，用来判断 num 是不是 NaN
何时：只要判断是不是 NaN，都用 isNaN(num)
反用：判断一个值是不是数字：!isNaN(num)
3. 区分 null 和 undefined
问题：==比较时，会自动将 undefined 隐式转为 null
null==undefined => null=null
解决：=== 全等：类型必须先相同，值再相等
其实===就是不带隐式转换的==
4. ?
逻辑运算：将多个关系运算，综合起来得到最终结论

何时：只要根据多个条件，综合得出最终结论时
包括：&&而且　||或(要么)　!不(没有)
条件 1&&条件 2...：
要求所有条件都为 true，结果为 true
只要一个条件为 false，结果为 false
条件 1||条件 2...：
只要一个条件为 true，结果为 true
除非所有条件都为 false，结果才为 false
!条件：颠倒条件的判断结果
隐式转换：默认将每个条件都转为 bool 类型
短路逻辑：如果前一个条件已经可以得出最终结论，则后续条件不再执行。
&&：只有前一个条件为 false 时，后续条件才不执行
||：只要前一个条件为 true 时，后续条件不再执行
利用短路：
1. 简单分支结构：1 个条件，1 件事，只有满足才做
如何：条件&&(操作)
练习中：运算符优先级：优先级高的运算，先计算
改变优先级：用()
2. 实现默认值/备用值：
如何：值 1||值 2
如果值 1 可以被转为 true，就选择值 1 使用
如果值 1 不能转为 true，就选择值 2 使用
位运算：了解
1. 左移/右移：
1<<3=8　1 左移 3 位 相当于 1*　2 的 3 次方
2<<3=2* 2 的 3 次方=16
8>>3=1　8 右移 3 位 相当于 8/　2 的 3 次方
2. 取整：
n^0　n|0　n>>>0　代替/简化 Math.floor(n)
3. 交换两变量的值：
var a=3,b=5;
方法一：var t=b; b=a; a=t;
方法二：a+=b; b=a-b; a=a-b;
方法三：a^=b; b^=a; a^=b;
赋值运算：
扩展赋值运算：对特殊赋值运算的简写：
a=a+b 将 a 的值+b 的值，再存回 a 中
——将 b 的值累加到 a 上
可简写为 a+=b;
a=a-b -> a-=b;
a=a*b -> a*=b;
a=a/b -> a/=b;
a=a%b -> a%=b;
更简化：递增递减运算：如果每次累加 1/累减 1
a+=1　->　a++
a-=1　->　a--

前++和后++：
1. 如果单独使用，前后都一样

2. 如果参与到其他表达式中时:
相同: 变量中的值一定都会+1
不同: *前*++，返回+1 后的*新*值
后++，返回未+1 的*旧*值

鄙视:

```
var n=3;
console.log(n++ + ++n + n++);
            //  ?   ?     ?
//console.log(++n + n++ + ++n);
              // ?     ?     ?
 console.log(n);//6
```

提示: 所有表达式都是从左向右依次执行
如果前一个表达式修改了变量的值，则会影响后续表达式的执行。

DAY3

正课:
1. ***函数
2. 全局函数
3. 分支结构

1. ***函数:

什么是: 内存中封装一项任务步骤清单的代码段，再起一个名字。
为什么: 代码重用
何时: 只要一段代码，可能被反复使用，都要先封装在函数中，再反复调用函数。
如何: 2 步:

1. 声明: function 函数名(参数列表){
函数体;
return 返回值;
}
参数: 函数运行时，接收传入函数的数据的变量
只不过不用 var 创建
参数列表: 多个参数间用逗号分隔
何时: 当一项任务必须某些数据才能正常执行时
为什么: 参数可让函数变得更灵活
返回值: 函数执行的结果
何时: 如果函数的调用者需要获得函数执行结果
如何返回: return 返回值;
其实，return 可单独使用: 退出函数
如果没有返回值的函数，其实也返回东西:
默认返回 undefined
2. 调用: 让引擎按照函数的步骤清单，执行任务。
var 返回值=函数名(参数值列表)
参数值列表: 传入函数的执行时必须的数据列表
只要函数定义时规定了参数变量，调用时都要传入参数值。且顺序和个数保持一致。
每个参数值之间用逗号分隔

强调：函数不调用不执行，只有调用才执行。

***作用域 scope：一个变量的可用范围

包括：2 种：

1. 全局作用域：window

全局变量：直接定义在全局中，不属于任何函数的变量

特点：可反复使用，随处可用

何时：如果一个变量需要反复使用，或跨函数随处可用时

2. 函数作用域：？

局部变量：2 种：

1. 函数的参数变量
2. 在函数内 var 出的变量

特点：仅在函数内可用，出了函数不能使用

不可重用

何时：如果一个变量，仅希望在函数内有效时

总结：优先定义并使用局部变量

尽量少使用全局变量——避免被污染

变量的使用顺序：优先使用函数内的局部变量

只要局部声明了，就不用全局的

除非局部没有声明，才去全局找

***声明提前(hoist)：

什么是：在开始执行程序前，引擎会将 var 声明的变量和 function 声明的函数，提前到*当前作用域*顶部集中优先创建。再开始执行程序。____赋值留在原地。

鄙视：如果先使用，后声明，一定在考声明提前

就要将所有 var 和 function 提前到当前作用域的顶部，先创建，再判断程序的输出

解决：

变量：

传统：强烈建议将所有的声明都集中在当前作用域顶部。

ES6：let 代替 var 声明变量

let 要求：在 let 之前，不允许提前使用该变量

函数：

var 函数名=function(…){…}

本质：函数名其实仅是一个普通的变量

函数其实是保存代码段的引用类型的对象

函数名通过函数对象的地址引用着函数对象

***按值传递：

什么是：两变量间赋值，或向函数中传入参数时，其实只是将原变量中的值复制一个副本给对方。

结果：

原始类型的值：修改新变量，不影响原变量的值

2. 全局函数：了解

什么是：ES 标准中规定的，不需要任何对象就可直接调用的函数。

比如：String()/Number()/Boolean()

parseInt/parseFloat()

isNaN()

反例：alert()/prompt()…

编码解码:
什么是: 将 url 中多字节字符或保留字符编码为单字节
为什么: url 中不允许出现多字节字符以及和保留字符冲突的字符
何时: 只要 url 中包含多字节字符以及和保留字冲突的字符时，都要先编码为单字节。
如何: 编码: var code=encodeURI(url)
解码: var text=decodeURI(url)
问题: 不能对保留字符编码解码,比如: ":" , "/"
解决: 编码: var code=encodeURIComponent(url);
解码: var text=decodeURIComponent(url);

eval 函数: 执行一段字符串格式的 js 表达式

程序三大结构: 顺序，分支和循环

3. 分支结构:

什么是: 让程序根据不同的条件执行不同的操作
何时: 只要让程序根据不同的条件执行不同的操作
如何:

1. 一个条件，一件事，只有满足才做，不满足就不做
   如果操作简单: 短路: 条件&&(操作)
   如果操作复杂: if(条件){
   操作
   }
2. 一个条件，二件事，二选一执行:
   如果仅两个值二选一:
   (条件?值 1:值 2)
   如果操作简单: 三目/三元/条件
   条件?操作 1:操作 2
   如果操作复杂:
   if(条件){
   操作 1;
   }else{
   操作 2;
   }
3. 多个条件，多件事，多选一执行
   如果多个值，多选一: 三目
   条件 1?值 1:
   条件 2?值 2:
   ...?... :
   默认值 ——强调: 默认值不能省略!
   如果操作简单: 三目
   条件 1?操作 1:
   条件 2?操作 2:
   ...?... :
   默认操作 ——强调: 默认操作不能省略!
   如果操作复杂: if..else if 结构
   if(条件 1){
   操作 1 //如果当前条件满足，则不再向后执行

}else if(条件 2){
//如果进入条件 2，暗示条件 1 不满足
操作 2
}else if(...){
...
}else{ —— 如果没有默认操作，则 else 省略
默认操作
}
简写：如果 if/else if/else 下只有一句话，可省略{}
——强烈不建议
特殊：如果所有条件都是等于比较，可简写为
switch(表达式){ //先计算表达式的值
//用表达式的值和每个 case 后的值做*全等*比较
case 值 1: //如果表达式的值全等于 case 后的值
操作 1; //就进入 case 下执行操作
case 值 2:
操作 2;
...
default: //如果所有 case 的值都不等于表达式的值
默认操作; //则执行默认操作
} ——强调：最后一个 default 可省略
问题：只要前一个 case 满足条件，则后续所有 case 都被执行。
原因：switch 是入口
解决：break; 可终止当前结构继续向后执行
如何：每个 case 之间，都要加 break

DAY4

正课：
1. ***循环
2. ***数组

1. ***循环：

什么是：让程序反复执行同一段代码
何时：只要让程序反复执行同一段代码
如何：三要素：
1. 循环条件：让循环可以继续执行的条件
2. 循环变量：在循环条件中用作比较和判断的变量
从几开始，每次递增/递减几，到几结束
3. 循环体：循环反复执行的代码段
几乎都要修改循环变量，向着不满足循环条件的趋势。

包括：3 种：
while: var 循环变量=初始值;
while(循环条件){
循环体;
修改循环变量
}

何时：必须先验证循环条件，才能执行循环体时

do while:

var 循环变量=初始值;

do{

循环体;

修改循环变量

}while(循环条件);

何时：希望即使不满足，也至少可以执行一次时

while vs do while

如果第一次循环都满足，两者的输出是完全一样的

while：先验证后执行

do while：先执行一次，再验证

如果第一次条件不满足，则 while 是一次都不执行，do while 至少执行一次。

for:

for(var 循环变量=初始值;循环条件;修改循环变量){

循环体;

}

for 循环和 while 循环的原理是完全一样的。

for vs while：如果循环变量的变化有规律：首先 for

如果循环变量的变化没有规律：首先 while

强调：js 中没有块级作用域：

放在 if/else if/else/for/while 中的变量，出了块，依然可用

vs java 所有{}都认为是块级作用域，块中的变量出了块，不能使用。

解决：让循环中的变量不要泄露到循环外

用 let 代替 var：let 声明的变量仅能在{}内使用

简写：1. 第一部分：可同时声明并初始化多个变量

2. 循环体：

如果 for 循环下只有一句话，可省略{}——不建议

如果 for 循环下只有一句话，且很简单：

for 循环的第三部分可先后执行多个短小的操作，每个操作之间用逗号分隔。——不能改变原来的执行顺序

break vs continue:

break：中断并退出当前循环

何时：当判断条件非常复杂时，就可用死循环+break 的方式灵活控制循环退出。——降低循环编写的难度

continue：跳过本轮循环，继续执行下一轮

只要颠倒判断条件，就可避免使用 continue

强调：break 不能参与到任何简写中，必须独立一句。

100 题：1,9,11,12,14,17

嵌套循环：循环内，又执行了另一个循环

如何：1. 截取片段找规律

2. 用外层循环反复调用规律的公式

作业：打印反三角和等腰三角

2. ***数组：

什么是：内存中连续存储多个数据的一块存储空间，再起一个统一的名字。
为什么：便于维护和查找
程序=数据结构+算法
数据结构：数据在内存中的存储结构
算法：解决问题的步骤
好的数据结构可极大提高程序的执行效率
何时：今后只要保存多个同一类型的数据时，都必须用数组
如何：3 件事：
创建：3 种：
1. 创建空数组：2 种：var arr=[];
new Array();
何时：创建数组时，暂时不知道数组内容
2. 创建数组同时初始化数组内容：
var arr=[值 1，值 2，...];
new Array(值 1,值 2,...)
何时：创建数组时已知数组的内容
3. 创建 n 个空元素的数组：
var arr=new Array(n);
何时：创建数组时仅知道数组元素个数，暂时不知道内容。
下标：数组中唯一标识一个元素存储位置的序号
从 0 开始，连续不重复

赋值：将数据保存到数组的某个元素中
arr[i]=新值;
取值：数组中每个元素的用法和普通的变量完全一样
数组也称为一组变量的集合，起一个统一的变量名
js 的数组三个不限制：
1. 不限制元素的数据类型
2. 不限制下标越界：
赋值：不会报错！自动在指定的新位置添加新元素。
如果添加后，数组的元素下标不连续——稀疏数组
取值：不会报错！仅返回 undefined
3. 不限制元素个数：
在任何时候，在任何位置添加新元素
.length 属性：记录理论上数组中的元素个数
.length 属性永远等于最后一个数字下标+1
.length 属性永远指向最后一个元素的下一个新位置
.length 可修改数组大小——缩容
固定套路：1. 获取数组最后一个元素：
arr[arr.length-1]
2. 获得倒数第 n 个元素
arr[arr.length-n]
3. 末尾追加一个新元素：
arr[arr.length]=新元素

4. 删除数组最后一个元素
arr.length-=1;
5. 删除数组末尾倒数 n 个元素
arr.length-=n;

***数组是引用类型的对象:
按值传递:
原始类型的值: 修改新变量，不影响原变量
原因: 复制的是值本身
引用类型的对象: 通过新变量修改对象，等效于直接修改原对象。新旧变量都受影响。
原因: 仅复制的是地址值，原对象始终只有一个
新旧变量使用相同的地址值，引用同一个对象
任何一方修改对象，另一方都同时受影响。

遍历:
什么是: 对数组中每个元素执行相同的操作
何时: 只要对数组中每个元素执行相同的操作
如何: for(var i=0;i<arr.length;i++){
arr[i] //当前元素
}

DAY5

正课:
补: ***关联数组
1. 数组 API:
2. 栈和队列:
3. 二维数组:

什么是索引数组: 下标都为数字的数组
***关联数组:
什么是: 可自定义下标名称的数组
为什么: 为了让每个元素都有一个专门的名称:
好处: 1. 便于维护
2. 便于查找
何时: 1. 希望每个元素都有专门的意义时
2. 希望快速查找元素时
如何: 2 件事:
创建: 2 步:
1. 先创建空数组: var ym=[];
2. 向空数组中添加新元素，要使用自定义的下标名称
ym["name"]="杨幂"
ym["math"]=81;
ym["chs"]=59;
ym["eng"]=89;
访问: 访问关联数组和访问索引数组完全一样
只不过要使用自定义的下标名称

关联数组中 length 属性失效，永远等于 0
因为关联数组中没有数字下标,length 无法+1
遍历关联数组:
for(var key in ym){//in 依次取出 ym 中每个可以
key //当前房间号
ym[key]//当前元素值
强调: key 不能加引号, 因为每次循环都在变化
}

关联数组原理:
问题: 索引数组，无法提前预知元素的具体位置
只能遍历查找: 受元素个数和元素存储位置影响
解决: hash 算法: 根据一个字符串，计算出尽量不重复的一个序号。相同的字符串计算出的序号一定相同
向关联数组中保存元素:
将自定义下标名交给 hash 算法，计算一个散列的位置序号。将元素保存到序号指定位置
从关联数组中取值时:
将自定义下标名交给 hash 算法，计算出和存入时完全相同的序号，引擎直接去序号位置获得元素。
优: 不受存储位置和元素个数的影响

1. 数组 API:

什么是数组: 连续保存一组数据，并提供了操作数据的简化 API
API: 别人已经实现的，咱们用现成的程序
包括:
转字符串: 2 种:
String(arr): 将 arr 中每个元素都转为字符串，用逗号连接
何时: 给数组拍照, 用于比较操作前后数组是否发生了变化。
arr.join("连接符"):将 arr 中每个元素都转为字符串，用自定义的连接符连接元素
何时: 只要希望使用逗号以外的其它连接符时
固定套路: 1. 把单词拼接为句子:
2. 无缝拼接:
错误: arr.join() => String(arr) =>逗号连接
正确: arr.join("")
重要用途: 判断数组是否为空!
3. 动态生成页面元素
console.log(String(arr)),console.log(arr),console.dir(arr)
console.log(arr); 先输出 dir 的结构，刷新后变为 String(arr)
仅输出内容，不关心结构: console.log(String(arr))
查看数组存储结构: console.dir(arr)
dir 查看结果时，不遵循先后执行的顺序

拼接和选取: 都无权修改原数组，只能返回新数组对象
拼接: 将其他数据或另一个数组，和当前数组拼接为新数组。
如何: var newArr=arr1.concat(值 1,值 2,arr2,..);
强调: 1. 不修改原数组，只返回新数组

2. 可打散数组类型参数为单个数据,再拼接
——concat 独有!

选取: 复制原数组中指定开始位置到结束位置之间的元素，组成新数组——原数组保持不变

如何: var subArr=arr.slice(starti,endi+1);

强调: 1. 如果一个函数两个参数都是下标时，就会含头不含尾。

简写: 1. 如果位置离结尾近，可用负数下标:

arr.length-n 可简写为: -n

2. 可省略第二个参数，表示一直选取到结尾

3. 两个参数都可省略: 完整复制一个数组

修改数组: splice

强调: 直接修改原数组

删除元素: arr.splice(starti,n)

删除 starti 位置开始的 n 个元素

强调: 不必考虑含头不含尾

其实: 有返回值: 返回被删除的元素组成的临时数组

var deletes=arr.splice(starti,n);

简写: 1. 支持负数下标，表示倒数第 n 个

2. 省略 n

插入新元素: arr.splice(starti,0,值 1,值 2,...)

在 starti 位置插入值 1,值 2,...

原 starti 位置的值向后顺移

强调: 不支持打散数组参数

替换: 先删除旧的，再在同一位置插入新的

arr.splice(starti,n,值 1,值 2,...)

先删除 starti 开始的 n 个元素

再在 starti 位置插入值 1,值 2,...

强调: 删除的个数和插入的个数不必相同

翻转: arr.reverse();

***排序:

什么是: 将数组中的元素按从小到大或从大到小的顺序重新排列

何时: 所有要给用户展示的数据必须都要先排序

如何: arr.sort();

默认排序规则: 将所有元素都转为字符串，再按字符串升序排列。

何时: 只有按字符串升序排列时，才用默认的 sort()

问题: 只能按字符串升序排列

解决: 自定义比较器函数:

什么是: 专门比较任意两值大小的函数

要求: 2 个参数: a,b

返回值: 如果 a>b，就返回正数

如果 a<b，就返回负数

否则返回 0

最简单的数字升序比较器:

function cmp(a,b){return a-b;}

如何使用: 将比较器函数对象作为参数传入 sort()

arr.sort(cmp);

强调: 不加()

回调：自己定义的函数，自己不调用
而是传入另一个函数中，被另一个函数反复调用。
何时：只要对数字元素排序，都要自定义比较器函数
问题 2：如何降序：
解决：只要颠倒比较器结果的正负号，就可改升序为降序
最简单的数字降序比较器：

```
function cmp(a,b){return b-a}
```

2. 栈和队列：

说明：js 中没有专门的栈和队列结构，都是用普通数组模拟的
栈 stack：
什么是栈：一端封闭，只能从另一端进出的数组
何时：只要希望始终使用最后进入数组的新元素时
如何：

1. 结尾出入栈：
   结尾入栈：arr.push(值) => arr[arr.length]=值
   强调：1. 其实 push 可压入多个值
   2. 不支持打散数组参数
   结尾出栈：var last=arr.pop()
2. 开头出入栈：
   开头入栈：arr.unshift(值)
   强调：开头入栈后的元素顺序和结尾入栈后的元素顺序是相反的。
   开头出栈：var first=arr.shift();

总结：向数组中添加元素 4 种：

1. concat(): 1. 不修改原数组，返回新数组
   2. 在结尾拼接元素
   3. 支持打散数组类型参数
2. splice(): 1. 直接修改原数组
   2. 在任意位置插入新元素
   3. 不支持打散数组类型参数
3. push(): 1. 直接修改原数组
   2. 只能在结尾拼接元素
   3. 不支持打散数组类型参数
4. shift(): 1. 直接修改原数组
   2. 只能在开头拼接元素
   3. 不支持打散数组类型参数

取出数组元素：4 种：

1. slice(): 1. 可获取任意位置的任意个元素
   2. 不修改原数组，返回选中的元素组成的新数组
2. splice(): 1. 删除任意位置的任意个元素
   2. 直接修改原数组
   3. 返回被删除的元素组成的新数组
3. pop(): 1. 只能从结尾删除一个元素，并返回
4. shift(): 1. 只能从开头删除一个元素，并返回

队列 queue:
什么是: 只能从结尾进入，从开头出的数组
何时: 只要希望按照先来后到的顺序使用数组元素时
如何:
1. 从结尾入队列: arr.push(值)
2. 从开头出队列: var first=arr.shift();

3. 二维数组:

什么是: 数组中的元素，又引用了另一个子数组
何时: 2 种:
1. 保存横行竖列的二维数据
2. 一个大的数组中，还需要对元素进行更细致分类

如何:
创建: 2 种:
1. 先创建空数组，再添加子数组:

```
var arr=[];
arr[0]=[0,0,0,0];
arr[1]=[0,0,0,0];
```

2. 创建数组同时，初始化子数组

```
var arr=[
  [0,0,0,0],
      ... ,
  [0,0,0,0]
];
```

访问: arr[r][c] 每个元素的用法和普通数组的元素完全一样
越界: 二维数组的行下标 r，不能越界

遍历二维数组: 外层循环控制行，内层循环控制列

```
for(var r=0;r<data.length;r++){
  for(var c=0;c<data[r].length;c++){
    data[r][c] //当前正在变量的元素
}
}
```

DAY6

正课:

1. String
   内置对象
   包装类型
   字符串 API
2. **** 正则表达式

1. String

什么是：由多个字符组成的*只读*字符数组
  vs 数组：相同：1. 下标；2. .length；3. slice
    不同：类型不同，API 不通用！
内置对象：ES 标准中规定的，浏览器厂商已经实现的对象
  包括：11 个：
    String Number Boolean ——包装类型
    Array Date Math RegExp
    Error
    Function Object
    Global (在浏览器中被 window 代替)
  包装类型：
    什么是：专门包装一个原始类型的值，并提供操作原始类型值的 API
    为什么：原始类型的值本身不具有任何功能
    何时：一般不会主动使用
      只要试图用原始类型的值调用函数时，会自动创建包装类型的对象，调用对象的函数执行操作。
    比如：n.toFixed(2) => typeof n=>number
      =>new Number(n).toFixed(2)
      str.charCodeAt()=>typeof str=>string
      =>new String(str).charCodeAt()
  String API：强调：所有 String API 都无权修改原字符串，只能返回新字符串
 大小写转换：
  将字符串中所有字母，统一转为大写或小写
   str.toLowerCase() 转小写
   str.toUpperCase() 转大写
    何时：不区分大小时：
      验证码，用户名，电子邮件
    获得指定位置的字符：
      str.charAt(i) <=> str[i]
      获得指定位置字符的 unicode 号
      str.charCodeAt(i) 获得 i 位置的字符的 unicode 号
        将 unicode 号，反向转回原文
        String.fromCharCode(unicode)
          强调：一次只能转一个字
    获取子字符串：
      str.slice(starti,endi+1)
      str.substring(starti,endi+1) 用法同 slice，但不支持负数
      str.substr(starti,n) => str.slice(starti,starti+n);
    查找关键词：4 种：
      1. 查找一个固定关键词的位置：

var i=str.indexOf("关键词",fromi)
从 fromi 位置开始，找 str 中下一个"关键词"的位置
返回值：找到的关键词的下标位置 i
如果没找到，返回-1
简写：省略 fromi，默认从 0 开始

查找最后一个关键词的位置：
var lasti=str.lastIndexOf("关键词");
问题：只能查找一个固定关键词的位置
解决：正则表达式

2. 判断字符串中*是否包含*符合规则的关键词
var i=str.search(/reg/i)
返回值：返回关键词的位置
如果没找到返回-1
问题：正则表达式默认都是大小写敏感的
解决：在第二个/后加 i，表示忽略(ignore)
何时：只要仅判断有没有关键词时，首选 search
问题：1. 只能查找一个关键词
2. 只能返回位置，无法返回关键词内容

3. 查找所有关键词的内容：
var kwords=str.match(/reg/ig);
返回值：返回所有关键词组成的数组
如果没找到，返回 null
问题：正则表达式默认都只匹配第一个关键词
解决：在第二个/后加 g，表示全部(global)
何时：仅希望获得关键词内容时
问题：仅能获得内容，无法获得每个关键词的位置
解决：

4. 即查找关键词内容，又查找位置：？

替换：将字符串中找到的关键词，替换为指定新内容
1. 简单替换：将所有关键词都替换为统一的内容
str=str.replace(/reg/ig,"新值");
2. 高级替换：根据每个关键词的不同，动态选择替换不同的新值
```
str=str.replace(/reg/ig,function(kw){
  kw//可自动获得本次找到的关键词
  return 根据 kw 的不同，动态返回不同的值
})
```
衍生：删除：将关键词替换为""
str=str.replace(/reg/ig,"")

切割：按指定字符，将一个字符串切割为多段子字符串
var subs=str.split(/reg/i);
说明：在切割后的结果数组中，不包含切割符了
固定套路：将字符串打散为字符数组：

2. ****正则表达式：Regular Expression

什么是：规定一个字符串中字符出现规律的规则
何时：2 种：

1. 模糊匹配多种关键词
2. 密码格式

如何:

1. 最简单的正则表达式，是关键词原文本身
2. 字符集:

   什么是: 规定一位字符可用的备选字符列表的集合

   何时: 只要一位字符，有多种可能备选字时

   如何: [备选字符列表]

   强调: 一个字符集只能规定一位字符的备选字符列表

   简写: 如果字符列表中部分字符是连续的，可用-省略中间字符

   比如: [0-9] 一位数字

   [a-z] 一位小写字母

   [A-Z] 一位大写字母

   [A-Za-z] 一位字母

   [0-9A-Za-z] 一位字母或数字

   [\u4e00-\u9fa5] 一位汉字

   特殊: 除了...

   [^字符列表]
3. 预定义字符集:

   什么是: 对常用字符集的简写: 4 个:

   \d 一位数字 =>[0-9]

   \w 一位字母,数字或 =>[0-9A-Za-z_]

   \s 一位空字符: 空格, Tab...

   . 通配符
4. 量词:

   什么是: 专门规定字符集出现次数的规则

   何时: 只要规定字符集出现的位数/次数

   如何: 量词必须紧跟在字符集之后，修饰相邻的前一个字符集。

包括:2 大类:

1. 有明确数量边界的:

   字符集{n,m} 至少 n 个，最多 m 个

   字符集{n,} 至少 n 个，多了不限

   字符集{n} 必须 n 个
2. 没有明确数量边界的:

   字符集? 可有可无，最多 1 个

   字符集* 可有可无，多了不限

   字符集+ 至少一个，多了不限
5. 选择和分组:

   选择: 或 多个规则，只要匹配其一即可!

   规则 1|规则 2|...

   |在正则中，优先级最低

   分组: 用()将多个规则分为一组

   何时: 如果希望量词同时修饰多个字符集时，就要先将多个字符集分为一组，再用量词修饰分组

   为什么: 默认字符集仅修饰相邻的前一个字符集

身份证号: 15 位数字 2 位数字 1 位数字或 Xx

\d{15} (\d\d [0-9Xx]) ?

后两部分，可有可无，最多 1 次

手机号: +86 或 0086　可有可无，最多一次
空字符　可有可无，多了不限
1
3,4,5,7,8　选一个
9 位数字
(\+86|0086)?\s* 1[34578]\d{9}

微 信 wei xin w x
(微|w(ei)?)\s*(信|x(in)?)
6. 指定匹配位置: 3 种:
^ 字符串开头 比如: 字符串开头的空字符: ^\s+
$ 字符串结尾 比如: 字符串结尾的空字符: \s+$
开头或结尾的空字符:
\b 单词边界: ^ $ 空格 标点符号
比如: 查找单词 no，不包含单词内的 no
\bno\b

DAY7

正课:
1. RegExp
2. Math
3. Date

1. RegExp

什么是: 专门封装一条正则表达式，并使用正则表达式执行查找和验证的 API
何时: 1. 使用正则表达式验证用户输入的格式
2. 即查找关键词内容，又查找关键词位置
如何:
创建: 2 种:
1. 直接量: var reg=/正则/ig;
何时: 如果正则表达式是固定不变的
强调: // 之中如果再出现/，就必须改为\/
2. 用 new: var reg=new RegExp("正则","ig");
何时: 如果正则表达式需要动态拼接生成
强调: ""中再出现\或"，都要改为\\和\"
API: 2 个:
1. 即查找内容，又查找位置:
var arr=reg.exec(str) 在 str 中查找下一个符合 reg 规则的关键词。
返回值: 返回一个数组 arr:
arr[0] 本次找到的关键词的内容
arr["index"] 本次找到的关键词的位置
如果没找到，就返回 null
原理: 每次查找 3 件事:
1. 将本次找到的关键词保存在 arr 的 0 位置
2. 将本次找到的关键词位置保存在 arr 的 index 位置

3. 每个 reg 对象都有一个 lastIndex 属性，标识下次开始的查找位置。开始时为 0。
每执行完一次 exec，exec 会自动修改 lastIndex 为当前位置+关键词长度

2. 验证格式:
var bool=reg.test(str) 验证 str 是否符合 reg 规则要求
验证通过返回 true，不通过返回 false
问题: 默认正则只要找到部分匹配就返回 true
解决: 凡是验证，必须前加^，后加$

2. Math:

什么是: 封装算数计算的常量和 API
何时: 只要执行数学计算
如何:
创建: Math 不用创建，不能 new，直接使用!
API:
取整: 3 种:
1. 上取整: 只要超过，就取下一个整数
Math.ceil(num)
2. 下取整: 只要超过，就省略小数部分
Math.floor(num)
vs parseInt(str)
3. 四舍五入取整:
Math.round(num)
缺: 只能取整,不能指定小数位数
优: 返回 number, 可直接计算
vs n.toFixed(d)
优: 可按任意小数位数四舍五入
缺: 返回字符串，要先转化，再计算
可自定义 round 函数:

乘方和开平方:
Math.pow(底数,幂)
Math.sqrt(n)
最大值和最小值:
Math.max(值 1,值 2,值 3,...)
Math.min(值 1,值 2,值 3,...)
问题: max/min 不支持数组参数
解决: Math.max.apply(null,arr)
随机数:
Math.random() 0<=r<1
公式: 在任意 min~max 之间取一个随机整数:
parseInt(Math.random()*(max-min+1)+min)
简写: 如果 0~max 之间
parseInt(Math.random()*(max+1))

3. Date

什么是: 封装一个时间值，并提供操作时间的 API
何时: 今后只要保存，操作时间，都要使用 Date
如何:
创建: 4 种:
1. 自动获得当前系统时间: var now=new Date();
强调: 只能获得客户端本地时间
2. 创建自定义时间点:
var date=new Date("yyyy/MM/dd hh:mm:ss");
如果只关心日期，不关心时间，后半部分时分秒可省略。省略后，默认是 00:00:00
var date=new Date(yyyy, MM-1,dd,hh,mm,ss);
3. 使用毫秒数创建日期对象:
原理: 日期对象中保存的是 1970 年 1 月 1 日 0 点至今的毫秒数——整数
为什么: 时间段，不受时区影响
相同的毫秒数: 在不同时区可显示不同的时间点
总结: 今后在数据库中存储时间或在程序中存储时间都用毫秒数，不要用文字。
问题: 人看不懂毫秒数
解决: 将毫秒数转化为程序中的日期对象，再输出
var date=new Date(ms);
4. 复制一个日期对象:
为什么: 日期的计算只能直接修改原日期
后果: 计算后，原日期无法保存
解决: 今后如果希望同时保存开始时间和结束时间，
都要先将开始时间复制一份，再用副本计算结束时间。
如何: var date2=new Date(date1);
API: 3 句话:
1. 单位: FullYear Month Date Day
Hours Minutes Seconds Milliseconds
2. 每个单位都有一对儿 getXXX()/setXXX()方法
getXXX() 获取指定单位的值
setXXX() 设置指定单位的值
特例: Day 没有 setXXX()方法
3. 取值范围:
FullYear: 和现实中年份一致
Month: 0~11 现实中: 1~12
计算机中的月份值，比现实中少 1
需要修正
Date: 1~31 和现实一样
Day: 0~6 现实: 日~六 和现实中一致
Hours: 0~23 和现实一样
Minutes/Seconds: 0~59
日期计算:
1. 两日期对象可直接相减计算时间差(ms 差)
2. 对任意单位做加减:3 步:
1. 取值: var date=now.getDate();
2. 做加减: date+=30;
16+30=46
3. 放回去: now.setDate(date);

setDate 可自动调整进制
简写: now.setDate(now.getDate()+30)
问题: setDate 直接修改原日期对象，无法保留开始时间。
解决: 在计算前，都要先将开始时间复制一个副本，再用副本计算结束时间。
转字符串:
String(date) 将 date 转为当地时间的完整格式
date.toLocaleString() 将 date 转为当地时间简化版格式
date.toLocaleDateString()将 date 转为当地时间简化版格式，仅保留了日期部分.
date.toLocaleTimeString()将 date 转为当地时间简化版格式，仅保留了时分秒部分.
date.toGMTString() 将 date 转为 0 时区国际标准时间

DAY8

项目课 2048

DOM

DAY1

正课:
1. 什么是 DOM
2. ***查找
3. 修改

1. 什么是 DOM

DOM: Document Object Model
什么是: 专门操作网页内容的 API
W3C 指定的标准，所有浏览器厂商遵照实现
何时: 今后只要操作网页内容，就必须用 DOM
为什么: 为了统一操作网页内容的 API
用 DOM 标准操作网页内容几乎 100%兼容
分为: 核心 DOM 和 HTML DOM
核心 DOM: 万能，繁琐
HTML DOM: 对核心 DOM 部分常用 API 的简化
不是万能，但简化
***DOM 树:
网页中的一切内容在内存中都是以树形结构存储
所有内容(元素，属性，文本)在树上，都是一个节点对象
DOM 有唯一的一个根节点 document
所有内容节点都是 document 的后代节点
***节点对象:
网页中的所有内容，都是 DOM 树上的节点对象

类型: Node
三大属性:
nodeType: 表示节点的类型
值是整数, 4 个:
document 9
element 1
attribute 2
text 3
何时: 只要区分节点类型时
为什么: 不同类型的节点可执行的操作不同
问题: 无法进一步区分不同元素
解决:
nodeName: 节点名(元素的标签名)
何时: 只要进一步细致区分元素种类时
包括: document #document
element 元素的标签名(全大写)
attribute 属性名
text #text
其实, nodeName 可代替 nodeType 来鉴别节点类型
nodeValue: 节点值——了解
包括: document null
element null
attribute 属性值
text 文本内容

DOM 操作:
查找触发事件的元素->绑定事件
->查找要操作的元素->修改/添加/删除

2. ***查找: 4 种:

1. 不需要查找就可直接得到:
document
document.documentElement —— html
document.head
document.body
2. 按节点间关系查找:
何时: 只要已经获得一个节点，要找周围节点时
包括: 2 大类关系:
节点树: 包含一切内容节点的树结构
父子: elem.parentNode
elem.childNodes 返回*直接*子节点的集合
elem.firstChild
elem.lastChild
兄弟: elem.previousSibling
elem.nextSibling
问题: 包括看不见的空字符文本节点的干扰
元素树: 只包含元素节点的树结构
父子: elem.parentElement
elem.children 返回*直接*子元素的集合

elem.firstElementChild
elem.lastElementChild
兄弟: elem.previousElementSibling
elem.nextElementSibling
优: 不受空文本的干扰
缺: IE8 不兼容
说明: 元素树不是一棵新树，仅是节点树的一个子集

强调: childNodes 和 children 返回的不是数组，而是类数组对象。
类数组对象: 长的像数组的对象
vs 数组: 相同: 1.下标, 2. .length
不同: 类型不同，API 不通用
简称: 集合
其实,childNodes 和 children 都返回动态集合
动态集合: 不实际存储数据，每次访问集合，都重新查找 DOM 树。
问题: 反复访问集合，会导致反复查找 DOM 树
遍历动态集合:
错误: for(var i=0;i<childNodes.length;i++){...}
正确: for(var i=0,len=childNodes.length;i<len;i++)

鄙视: 用节点间关系，遍历指定父节点下所有后代节点
1. 递归遍历: 2 步:
Step1: 先仅查找直接子节点
Step2: 对每个直接子节点，调用和父节点完全相同的函数
算法: 深度优先遍历: 每个节点都优先遍历子节点，子节点遍历完，才遍历兄弟节点。
问题: 递归时，函数内的函数名不能写死
解决: 用 arguments.callee 指代当前函数自己。

2. 用循环代替递归:
节点迭代器对象: 依次遍历，并返回每个节点对象的小工具——内置深度优先算法
如何: 2 步:
1. 创建节点迭代器对象:
```
var iterator=document.createNodeIterator(
    parent, NodeFilter.SHOW_ALL, null, false
                        SHOW_ELEMENT
);
```
2. 循环调用迭代器的 nextNode()，调到下一个节点。直到返回 null 结束。
```
do{
  var node=iterator.nextNode();
  if(node!=null){
    //输出 node
}else break;
}while(true);
```
3. 按 HTML 查找: 4 种:
1. 按 id 查找: var elem=document.getElementById("id")
强调: 1. 必须用 document 调用!

2. 按标签名查找:

var elems=
parent.getElementsByTagName("标签名");

强调: 1. 可在任意父元素上调用，仅找当前父元素下的指定元素。
2. 返回动态集合
3. 不但找直接子元素，而且找所有后代元素

3. 按 name 属性查找: ——了解

何时: 在表单中查找有 name 属性的表单元素时

var elems=document.getElementsByName("name");

强调: 1. 只能用 document 调用
2. 返回动态集合

4. 按 class 属性查找:

var elems=parent.getElementsByClassName("class");

强调: 1. 在任意父元素上调用
2. 返回动态集合
3. 不但找直接子元素，而且在所有后代中查找
4. 不要求完整匹配 class，只要包含就找到

问题: 每次只能按一个条件查找，如果条件复杂，代码会很繁琐

解决: 当查找条件复杂时，要用选择器查找

4. 按选择器查找: Selector API: 2 个:

1. 只找一个元素:

var elem=parent.querySelector("selector");

2. 找多个元素:

var elems=parent.querySelectorAll("selector");

强调: 1. 可在任意父元素上调用
2. 不仅查找直接子元素，且查找所有后代元素
3. 返回非动态集合
非动态集合: 数据直接保存在集合本地，无序反复查找。
4. 选择器兼容性，受制于当前浏览器

鄙视: 按 HTML 查找 vs 按选择器查找

1. 返回值: 按 HTML 查找: 返回动态集合:
优: 效率高！只需返回需要的数据即可，不需要返回完整数据
缺: 造成反复查找
按选择器查找: 返回非动态集合
优: 不会反复查找
缺: 首次执行效率低
2. 易用性: 当查找条件复杂时
按 HTML 查找: 繁琐
按选择器查找: 简洁

总结:

如果根据一个条件就可获得想要的元素:
首选按 HTML 查找
如果查找条件复杂时
首选按选择器查找

总结返回值:

1. 凡是返回一个元素的 API，如果没找到，都返回 null

2. 凡是返回多个元素的 API，如果没找到，都返回空元素的集合

练习：事件绑定：

```
//当事件发生时，自动执行保存的事件处理函数
elem.on 事件名=function(){
    this //自动获得触发事件的.前的当前元素
}
```

3. 修改：

内容：

innerHTML：获取或设置元素开始标签到结束标签之间的 HTML 代码片段。

textContent：获取或设置元素开始标签到结束标签之间的纯文字内容。

2 件事：1. 去掉文本中内嵌的标签

2. 将所有转义字符翻译为正文

IE8：innerText

.value：获取或设置表单元素的内容

属性：

样式：

DAY2

正课：

1. 修改：
   属性，样式
2. 添加/删除

1. 修改：

属性：3 类：

标准属性：2 种方式访问：

1. 核心 DOM：4 个 API：——繁琐
   所有属性节点都保存在元素的 attributes 集合中
   获取属性值：elem.getAttribute("属性名")
   修改属性值：elem.setAttribute("属性名","属性值")
   是否包含：elem.hasAttribute("属性名")
   移除属性：elem.removeAttribute("属性名")
2. HTML DOM：——简洁——首选
   HTML DOM 将 HTML 标准属性都封装为了元素对象的属性，可直接用.访问。

获取属性值: elem.属性名
设置属性值: elem.属性名=值
是否包含: elem.属性名==""
移除属性: elem.属性名=""
特殊: class 属性: class 是 js 语言的保留字，所以 DOM 中的 class 属性被迫改名为 className
状态属性: selected checked disabled
不能用核心 dom 访问，只能用 HTML DOM 访问
自定义扩展属性: data-toggle="dropdown"
1. 核心 DOM: ——繁琐
不能用 HTML DOM 访问
2. HTML5 语法: ——简洁
定义自定义属性: data-属性名="值"
获取/修改属性值: elem.dataset.属性名=值

样式: 2 种:
单独修改一个 css 属性
1. 内联: 写在元素的 style 属性中的样式
获取或设置内联样式:
elem.style.css 属性
强调: 1. css 属性名都要去横线变驼峰:
比如: background-color->backgroundColor
list-style-type->listStyleType
2. 所有 css 属性值都是字符串(可能带单位)
计算前都要先去单位，转数字再计算。
设置内联: 优: 优先级最高，当前元素独有，不影响其它元素
获取内联:
问题: 实际开发中可能几乎不包含内联样式。用 elem.style 可能无法获得任何样式。
解决: 用 getComputedStyle(elem)代替 elem.style
ComputedStyle: 计算后的样式: 最终应用到元素上的所有样式的综合。
如何: var style=getComputedStyle(elem)
style.css 属性
强调: 计算后的 css 属性都是只读
因为计算后的 css 属性可能原来是共用的，一旦修改，牵一发而动全身。

2. 内部/外部样式表: ——危险
3 步:
1. 找样式表对象: var sheet=document.stylesheets[i]
2. 找 cssRule 对象: var cssRule=sheet.cssRules[i]
如果是 keyframes,就需要继续找下级 cssRule
3. 修改 cssRule 的 style 下的 css 属性值:
cssRule.style.css 属性=值

问题: 每次只能修改一个 css 属性
解决:
通过修改元素的 class 属性，来批量应用多个 css 属性

2. 添加/删除：

添加：3 步：

1. 创建空元素：

var elem=document.createElement("标签名")

比如：var a=document.createElement("a")

相当于<a>

2. 设置关键属性：

a.href="http://tmooc.cn";

a.innerHTML="go to tmooc";

 go to tmooc

3. 将新元素添加到 DOM 树指定父元素下：

新建的元素，必须添加到 DOM 树才能显示出来

3 种：

末尾追加：parent.appendChild(elem)

中间插入：parent.insertBefore(elem, child)

替换：parent.replaceChild(elem,child)

优化：尽量减少操作 DOM 树

原理：页面加载过程：

html -> DOM Tree

↓

Render Tree->***layout***->paint

↑ 计算每个元素的精确布局

css -> CSSRules 耗时

频繁修改 DOM 树，会导致频繁 layout，降低页面响应速度。

解决：2 种：

1. 如果需要同时添加父元素和子元素时，就要在内存中先将子元素添加到父元素，再将父元素整体一次性添加到 DOM 树

2. 如果父元素已经在页面上，要添加多个平级子元素，就要借助文档片段。

文档片段：内存中临时保存多个子元素的虚拟父元素

何时：要添加多个平级子元素

如何：3 步：

1. 创建文档片段：

var frag=document.createDocumentFragment();

2. 将平级子元素添加到 frag 中

frag.appendChild(elem);

3. 将 frag 整体添加到页面父元素下

parent.appendChild(frag);

说明：将子元素添加到 DOM 树后，frag 自动释放，不占用页面空间。

删除：parent.removeChild(child);

DAY3

正课：

1. HTML DOM 常用对象：

Image Select/Option Table/…. Form/..
2. 什么是 BOM
3. 打开和关闭窗口
4. ****定时器
5. BOM 常用对象:
history location navigator screen

1. HTML DOM 常用对象:

Image: 指代页面上一个 img 元素
创建: var img=new Image();

Select: 指代页面上一个<select>元素
属性: sel.selectedIndex 获得当前选中项的下标位置
sel.value 获得当前选中项的值(value)
如果当前 option 没有 value，则用 innerHTML 代替
sel.options 获得当前 sel 下所有 option 元素的集合
sel.options.length 选项的个数
sel.options.length=0 清除所有选项
sel.length => sel.options.length 选项的个数
sel.length=0 清除所有选项
方法: sel.add(option) 向 sel 末尾追加一个 option
强调: 不支持文档片段
sel.remove(i) 移除 i 位置的选项
事件: sel.onchange 当选中项发生改变时触发
Option: 指代页面上一个 option 元素
创建: var opt=new Option(text,value)
属性: opt.text 代替 opt.innerHTML

Tabel: 指代一个 table 元素:
管着行分组: 创建，删除，获得
创建: var thead=table.createTHead();
var tbody=table.createTBody();
var tfoot=table.createTFoot();
删除: table.deleteTHead();
table.deleteTFoot();
获取: table.tHead
table.tBodies[i]
table.tFoot
行分组: tHead tBody tFoot
管着 tr:
创建: var tr=tXXX.insertRow(i);
说明: 如果 i 位置有行，则原位置的行向后顺移
常用操作: 1. 末尾追加一个新行: tXXX.insertRow();
2. 开头插入: tXXX.insertRow(0)
删除: tXXX.deleteRow(i) 删除当前行分组中下标为 i 的行
获取: tXXX.rows
行:

管着格:
创建: tr.insertCell(i)
强调: insertCell 只能创建 td
删除: tr.deleteCell(i)
获取: tr.cells

删除行: 2 种:
1. 行分组.deleteRow(i)
i 是行在行分组内的相对位置, 无法直接获得
2. table.deleteRow(i)
i 是行在整个表中的绝对位置，可直接获得:
tr.rowIndex 表示 tr 在整个表中的位置
总结: 今后凡是删除行: table.deleteRow(tr.rowIndex)

Form: 代表页面上一个 form 元素
获取: var form=document.forms[i/id]
属性: form.elements 包含所有表单元素: input select button textarea
form.elements.length 获得表单中表单元素的个数
form.length => form.elements.length
方法: form.submit(); //手动提交表单
只有验证通过才提交: 2 种:
1. input button 代替 input submit
在 input button 的单击事件中自己调用 form.submit();
2. input submit 配合 onsubmit 事件 ?

表单元素:
获取: form.elements[i/id/name]
更简写: 如果表单元素有 name 属性，可直接:
form.name
方法: elem.focus(); 让 elem 获得焦点
elem.blur(); 让 elem 失去焦点

2. 什么是 BOM: Browser Object Model

BOM: 专门操作浏览器窗口/软件的 API
window: 2 个角色:
1. 代替 ES 中的 global 充当全局作用域对象
2. 封装所有 dom 和 BOM 的 API
window 的功能: 打开和关闭窗口，弹出对话框...
包括:
history: 保存当前窗口打开后，成功访问多的 url 的历史记录栈。
location: 保存当前窗口正在打开的 url 的对象
document: 封装页面内容和 DOM API 的根对象
navigator: 保存浏览器的配置信息
screen: 保存客户端显示设备的信息
event: 定义事件对象

3. 打开和关闭窗口：

打开一个新窗口: /*window.*/open("url","name")
其中: name: 新窗口在内存中的名称
浏览器规定: 相同 name 的窗口只能开 1 个
后开的会替换先开的。
预定义值: _self: 用当前窗口的 name 打开新窗口
结果: 新窗口替换当前窗口
_blank: 不指定 name 属性，让浏览器随机分配
结果: 每个窗口的 name 都不一样
就可打开多个
打开一个超链接: 4 种:
1. 在当前窗口打开，可后退:
HTML:
js: open("url","_self")
2. 在当前窗口打开，不可后退:
当前窗口每打开一个新 url，都会将新 url 保存在 history 中。
如果新 url 是追加进 history 中，则可后退
如果新 url 将当前 url 替换掉，则无法后退
js: location.replace(url)
3. 在新窗口打开，可打开多个:
HTML:
js: open("url","_blank")
4. 在新窗口打开，只能打开一个:
HTML:
js: open("url","自定义 name")

4. ****定时器: 2 种:

周期性定时器:
什么是: 让程序每隔一段时间间隔反复执行一次任务
何时: 让程序每隔一段时间间隔反复执行一次任务
如何: 3 件事:
1. 任务函数: function task(){...}
定义了定时器每次要执行的任务
2. 启动定时器: var timer=setInterval(task,间隔 ms)
强调: task 是回调，不用加()
其中: timer 是定时器序号，在内存中唯一标识一个定时器的整数。
3. 停止定时器: clearInterval(timer)
其中: timer 是要停止的定时器序号
问题: 停止定时器不会自动清除 timer 变量中的序号，有可能影响下次启动。
解决: 凡是停止定时器，都要手动清除 timer
2 种方式:
1. 手动停止: 用户通过操作来停止定时器
2. 自动停止:
在任务函数中判断临界值
只要达到临界值，就自动停止定时器

一次性定时器:
什么是: 程序先等待一段时间后，再自动执行一次任务

何时：先等待一段时间再自动执行一次
自动执行一次后，自动停止
如何：3 件事：
1. 任务函数：
2. 启动定时器：var timer=setTimeout(task,等待 ms)
3. 停止定时器：在任务执行前，就停止等待
clearTimeout(timer);

原理：定时器中的回调函数只能在主程序所有语句执行完才能开始执行。
鄙视：var a=10;
function fun(){ a=100; }
setTimeout(fun,0); //fun()
console.log(a);//10 //100

5. BOM 常用对象：

history：保存当前窗口打开后成功访问过的 url 的历史记录栈
history.go(n) : history.go(0) 刷新
history.go(-1) 后退
history.go(1) 前进

location：保存当前窗口正在打开的 url 对象
属性：.href 获取或设置完整 url
.protocol 协议
.host 主机名 包含：主机名+端口号
.hostname 仅主机名
.port 端口号
.pathname 相对路径
.hash 获得锚点地址
.search 获得?后的查询字符串
方法：location.assign("url") 在当前页面打开新 url
其实：=>location.href="url"
=>location="url"
location.replace("url") 在当前窗口打开，替换 history 中当前 url，实现禁止后退
location.reload(fales/true) 刷新
鄙视：2 种刷新：
1. 默认刷新：优先使用本地缓存中的文件，除非文件比服务器上的旧，才被迫下载新文件。
F5
history.go(0)
location.reload()
2. 强制刷新：每次强制跳过浏览器本地缓存，总是从服务器下载新文件。
location.reload(true)

screen：保存客户端显示设备的信息：
鄙视：判断屏幕宽
css：媒体查询
js：screen.width

navigator：封装了浏览器的配置信息：

cookeEnabled 判断是否启用 cookie
plugins 保存所有插件信息
plugins["插件名"]!==undefined 说明装了!

DAY4

正课:

1. ****event:

1. ****event
什么是: 用户手动触发的或浏览器自己触发的页面状态的改变。
当事件发生时，都可以执行事件处理函数来响应事件的操作。

绑定事件处理函数: 3 种:
1. 在 HTML 中绑定: <ANY on 事件名="js 语句"
比如: <button onclick="fun()">
问题: 不符合内容与行为分离的原则，不便于维护
2. 在 js 中绑定:
1. on 事件名: elem.on 事件名=function(){
this->elem
}
局限: 一个事件，是能绑定一个处理函数
2. addEventListener: 可一个事件同时绑定多个处理函数
elem.addEventListener("事件名",fn);
还可移除:
elem.removeEventListener("事件名",函数名)
问题: 绑定时如果使用匿名函数绑定，移除时无法找到原函数
解决: 如果一个处理函数可能被移除，就必须用有名的函数

事件模型/周期:
3 个阶段:
1. 捕获: 由外向内，记录各级父元素上绑定的事件处理函数——只记录，不触发
2. 目标触发: 优先触发目标元素上的事件处理函数
3. 冒泡: 由内向外，按捕获顺序的反向，依次执行父元素上的事件处理函数

事件对象: e
什么是: 事件发生时，自动创建的，封装事件信息的对象
如何获取: 事件对象，默认作为处理函数的第一个参数传入: 事件处理函数(e){ e//自动获得事件对象 }
何时: 只要希望获取事件的数据，或修改事件的默认行为时
包括:
取消冒泡: e.stopPropagation();
利用冒泡:
问题: 浏览器通过遍历方式查找事件处理函数执行。如果添加的事件监听越多，遍历越慢，网页响应速度越慢
优化: 尽量少的添加事件监听
如何: 如果多个平级子元素绑定相同事件时，可在父元素仅添加一个事件监听，所

有子元素共用!

问题: 1. 如何获得目标元素:

错误: this -> 父元素

正确: e.target

2. 可能目标元素不是想要的,就要先鉴别目标元素，再决定是否执行操作

另一个作用: 如果一个父元素下的子元素，需要动态生成，并绑定事件。则必须将事件绑定在父元素。动态生成的子元素，才能自动使用父元素的事件，而不需要反复单独绑定。

阻止默认行为: e.preventDefault();

2 个典型应用:

1. 阻止 a 元素作为按钮时，自动添加#锚点地址
2. 表单验证未通过时，阻止默认的提交

事件坐标: 3 种:

1. 相对于屏幕左上角: e.screenX/screenY
2. 相对于文档显示区左上角: e.clientX/clientY
3. 相对于当前元素左上角: e.offsetX/offsetY

页面滚动:

事件: window.onscroll

获得页面滚动的位置:

document.body.scrollTop

JAVASCRIPTCORE

JS 高级

DAY1

正课:

1. ****Function
1. 创建
2. 重载
3. 匿名函数
4. ***作用域和作用域链
5. *****闭包

2. 错误处理

1. ****Function

1. 创建: 3 种:

1. 声明:

function 函数名(参数列表){函数体; return 返回值}

特点: 会被声明提前

2. 直接量:

var 函数名=function(参数列表){函数体; return 返回值}

特点：不会被声明提前
何时：只要希望函数的定义不被声明提前时

3. 用 new：

var 函数名=
new Function("参数 1","参数 2","函数体;return 返回值")

强调：所有参数变量和函数体，必须用""包裹
鄙视：以下函数创建正确的是：

```
function fun(a,b){return a-b}
var fun=function(a,b){return a-b}
var fun=new Function(a,b,"return a-b")    XXX
var fun=new Function("a","b","return a-b")
```

2. 重(chong)载(overload)：

什么是：相同函数名，不同参数列表的多个函数，在调用时可根据参数的不同，自动选择对应的函数执行。

何时：如果一件事，根据不同的参数，执行不同的逻辑时
为什么：减少函数的数量，减轻调用者的负担
问题：js 语法，默认不支持多个同名函数同时存在！
如何变通：

1. 仅定义一个函数，不要写参数列表
2. arguments：函数内自定创建的
   用于接收传入函数的所有参数值的
   类数组对象

   类数组对象：长的像数组的对象
   vs 数组：相同：1. 下标，2. length，3.for 遍历
   不同：类型不同——API 不通用
3. 函数内，根据 arguments 中参数的个数或类型，动态决定执行不同操作。

问题：参数是否还需要
答：需要：

1. 参数用于提示调用者如何正确使用函数
2. 参数名一般都比 arguments 简洁且见名知义

3. 匿名函数：

什么是：创建函数时，不指定函数名的函数
何时：2 种：

1. 当一个函数仅使用一次时
2. 划分临时作用域，避免全局污染时

为什么：1. 节约内存，2. 避免全局污染
如何：2 种：

1. 回调：
   比如：

```
arr.sort(function (a,b){return a-b})
str.replace(reg,function(kw){return 替换值})
btn.addEventListener("click",function(){...})
```

2. 自调：定义匿名函数后，立刻调用自己，调用后立刻释放
   如何：2 种：

+function(...){...}(...)
(function(...){...})(...)
何时：只要不希望造成全局污染时，就用匿名函数包裹所有自定义的代码。

4. ***作用域(scope)和作用域链(scopechain)

什么是作用域：一个变量可用范围，也是一个变量的实际存储位置。
包括：2 种：
1. 全局作用域：window
保存全局变量：特点：随处可用，可重复使用
2. 函数作用域：AO
保存局部变量：特点：仅函数内可用，不可重用

*****函数的生命周期：
程序开始执行前：
创建执行环境栈(数组),用于保存将要调用的素有函数
在栈中添加第一个函数调用：浏览器的主程序 main()
主程序创建全局对象 window
定义函数：
用函数名创建全局变量
创建函数对象封装函数体
函数名变量引用函数对象
函数对象的 scope 属性(籍贯)指回 window
调用函数：
创建函数作用域对象 AO(活动对象)
在 AO 中添加函数中定义的所有局部变量(var 声明的和参数变量)
变量使用顺序：优先使用 AO 中的局部变量
AO 中有，就不去全局找
AO 中没有，才去全局找
调用后：
ECS 中的函数调用出栈
导致 AO 被释放
导致 AO 中的局部变量一同释放

作用域链：由各级作用域对象连续引用形成的链式结构
保存着所有变量，且控制着变量的使用顺序

5. *****闭包(closure):

什么是：即重用变量，又保护变量不被污染的一种机制
为什么：全局变量和局部变量都有优缺点：
全局：优：可反复使用
缺：随处可用——易被污染
局部：优：仅在函数内可用，不会被污染
缺：不可重用
何时：即重用变量，又保护变量不被污染
如何：3 步：
1. 用外层函数，包裹住受保护的变量和内层函数

2. 外层函数将内层函数返回到外部
3. 使用者调用外层函数，获得返回的内层函数对象

鄙视：闭包如何形成：外层函数的作用域对象(AO)无法释放

闭包的问题：比普通函数占用更多内存空间——多的是外层函数的 AO

解决：将引用闭包函数的变量赋值为 null
导致内层函数对象先释放
导致外层函数的 AO 一并释放

鄙视：画简图:2 步：

1. 找受保护的变量：
外层函数的局部变量(var 的或参数变量)
确定在外层函数调用后，受保护变量的最终值
2. 找抛出的内层函数:2 个途径：
1. return
2. 直接给全局变量赋值
3. 隐藏：将内层函数放入一个数组/对象中隐藏返回
一次外层函数调用，返回的多个内层函数，共用同一个受保护的变量。

2. 错误处理：

什么是错误：程序运行过程中，导致程序无法继续执行的异常情况。
发生错误后，程序会立刻中断退出

什么是错误处理：即使程序发生错误，也保证不强行退出的一种机制

为什么：避免程序强行中断，导致极差的用户体验

何时：只要希望即使程序发生错误，也保证不强行退出

如何：

```
try{
    可能发生错误的语句
}catch(err){// 只有发生错误时，才执行
    错误处理代码：1.保存进度；2. 提示用户；3. 记录日志
}
```

其中：err 是错误对象：

什么是：错误发生时，自动创建的，保存错误信息的对象

性能问题：放入 try catch 中的代码，可能执行效率会降低，且一旦出错，还要额外创建 error 对象，占用更多内存空间。

解决：如果可提前预知错误的原因，可用 if...else...来代替 try catch——对于开发人员的经验要求非常高

主动抛出错误：

什么是：当前程序运行出错时，主动新建一个错误，抛出

何时：在协作开发中，函数的作者，可用主动抛出错误的方式提醒调用者如何正确使用函数。

如何：throw new Error("错误提示");

鄙视：js 中错误的类型：6 种

SyntaxError：语法错误

ReferenceError: 引用错误, 要用的变量未找到
TypeError: 类型错误, 使用错误的类型调用函数或访问元素。
RangeError: 范围错误, 参数超范围

URIError,EvalError;

bug: 只要导致程序出现问题的原因

DAY2

正课:

1. *****面向对象 OOP

三大特点: 封装，继承，多态

什么是面向对象: 程序中都是用对象结构来描述现实中一个具体事物。
什么是对象: 程序中封装现实中一个事物属性和功能的存储空间。
为什么: 现实中，任何数据都有明确的归属，都不是孤立的。
何时: 只要用程序描述现实中一个事物，都要将事物的属性和功能封装在一个对象中
如何: ****三大特点: 封装，继承，多态
封装:
什么是封装: 将现实中一个事物的属性和功能集中定义在一个对象结构中:
事物的属性会成为对象的属性
事物的功能会成为对象的方法
为什么封装: 让每个数据都有其专门的归属，便于维护和操作
何时封装: 只要使用面向对象，都要先将数据封装在对象中，再使用。
如何封装: 3 种:
1. 对象直接量:
```
var obj={//创建一个新对象
  属性名:值,
    ... : ... ,
  //方法名:function(参数列表){
  方法名(参数列表){//ES6
    this.属性名...
},
方法名(...){...}
}
```
何时使用直接量: 如果创建对象时就已经知道对象的成员。

*****问题: 对象自己的方法，要访问自己的属性:
错误: 对象名.属性名, 一旦对象名修改，方法内的对象名要同时修改。不便于维护
正确: 在方法内使用关键词 this 自动指代当前对象本身。
this.属性名
优: 即使对象名修改，this 也可自动获得当前对象本身，和对象名无关
this 可翻译为: 当前对象的/自己的
总结: 今后只要对象自己的方法中，要使用自己的属性，就必须用 this.属性名

2. 用 new: 2 步:

1. 创建空对象: var obj=new Object();
2. 向空对象中添加新属性:
   obj.属性名=值;
   obj.方法名=function(){ this.属性名 }

何时使用: 如果创建对象时暂时不知道对象的成员
对象创建后，随时可添加新属性。

js 中对象本质: 其实就是一个关联数组
对象其实是关联数组的简化版用法:

	关联数组	对象
访问元素:	ym["属性名"]	ym.属性名
创建:	2 步: 先创建空[]	可用直接量{}一次性创建
	再添加新元素	

问题: 一次只能创建一个对象
反复创建多个相同结构的对象，代码冗余太多，不便于维护
解决:
3. 用构造函数反复创建多个相同结构的对象:
   构造函数: 专门描述一类对象统一结构的函数
   还用于将一个新的空对象装修成想要的结构并存入数据。
   何时: 反复创建多个相同结构的对象
   都要先用构造函数描述统一的结构
   如何: 2 步:
   1. 定义构造函数:

```
function 类型名(属性参数列表){
  this.属性名=属性参数;
  ... ...
  this.方法名=function(){
    this.属性名
  }
}
```

   2. 用 new 调用构造函数创建新对象:
   var obj=new 类型名(属性值列表);
   new 4 件事:
   1. 创建一个新的空对象
   2. 自动设置新对象继承构造函数的原型对象
   3. 调用构造函数，向新对象中添加新属性
   4. 返回新对象的地址保存在变量中

如何访问对象的成员:
成员=属性+方法
访问属性: 对象.属性名 用法和普通的变量完全一样
属性其实就是保存在对象内的变量
调用方法: 对象.方法名() 用法和普通的函数完全一样
方法其实就是保存在对象内的函数
.读作"的"
其实，也可用对象["属性名"]方式访问

问题: 方法定义在构造函数内。每创建一个新对象，都会重复创建相同的函数副本——浪费内存!
总结: 构造函数: 优: 代码重用! 缺: 无法节约内存

解决:

*****继承:

什么是: 父对象的成员，子对象无需重复创建就可直接使用

为什么: 不但可代码重用，且还可节约内存

何时: 只要多个子对象，拥有相同的成员时，都要将相同的成员，仅保存在父对象中一份即可。所有子对象共用。

如何:

原型对象(prototype): 专门集中保存同一类型的多个子对象，共有成员的父对象。

如何获得原型对象:

1. 买一赠一: 创建构造函数同时，已经自动创建了该类型的原型对象。构造函数的prototype 属性引用着原型对象。原型对象的 constructor 也引用着构造函数对象
2. 自动继承: 创建子对象时，会自动设置新对象的 __proto__ 属性继承构造函数的原型对象

如何向原型对象中添加共有属性:

构造函数.prototype.属性/方法名=值/function(...){...}

强调: 原型对象中的方法，要想访问对象自己的属性，也必须加 this.

总结:

每个子对象，值不同的属性，都要定义在构造函数中

所有子对象共有的相同方法和属性值，都要集中定义在原型对象中。

共有属性和自有属性:

自有属性: 直接保存在当前对象本地的属性

共有属性: 保存在原型对象中，所有子对象共用的属性

相同: 取值时，对象.属性

不同: 修改时:

自有属性可直接通过自对象修改:

子对象.自有属性=值

共有属性只能通过构造函数的原型对象

构造函数.prototype.共有属性=值

原型链: 由各级父对象逐级继承形成的链式结构

任何对象都有 __proto__ 属性指向其父对象

保存了所有对象中的成员

控制着对象成员的访问顺序: 优先使用自有属性

自己没有才延原型链向父级找

只要找到就不再向上找

内置对象的原型对象:

其实每种内置对象也都有一对儿构造函数和原型对象:

其中:

构造函数负责创建新对象:

比如: var arr=new Array();

var now=new Date();

var reg=new RegExp();

特例: Math 和 window 不是构造函数，不能 new!

原型对象负责集中存储该类型可用的所有 API

比如: arr.sort() arr.push() arr.slice()

因为 Array.prototype:{

sort(){...},

push(){...},

```
        slice(){...},
        ...
    }
```

解决浏览器兼容性问题：旧浏览器无法使用新 API

2 步：

1. 判断：如果当前浏览器的指定类型的原型中不包含想要的 API

```
if(!"indexOf" in Array.prototype)
```

其中 in 用于检查左边的成员名是否在右边的对象中或对象的原型链上

```
if(typeof Array.prototype.indexOf!=="function")
```

2. 如果没有就向原型中添加一个新函数

```
Array.prototype.indexOf=function(){
    this //代表当前数组对象
}
```

问题：从父对象继承来的成员不一定都是想要的

解决：

多态：同一个函数在不同情况下，表现出不同的状态

重写(override)：如果子对象觉得父对象的成员不好用，就可在子对象本地定义同名成员，覆盖父对象的成员。

为什么：从父对象继承来的成员不一定都是想要的

何时：如果子对象觉得父对象的成员不好用

总结：面向对象三特点：

封装：将事物的属性和功能集中定义在一个对象中

为什么：便于维护

继承：父对象的成员，子对象无需重复创建，可直接使用

为什么：代码重用，节约内存——偷懒

多态：如果父对象的成员不好用，可在子对象中重写同名成员。

为什么：为了体现父子对象间的差异

自定义继承：

何时：只要希望获取其它对象现成的成员时

如何：3 种：

1. 直接修改一个对象的__proto__属性指向新父对象

```
child.__proto__=father
```

问题：__proto__是内部属性，不推荐使用

解决：Object.setPrototypeOf(child,father)

2. 通过修改构造函数的原型对象来批量修改所有子对象的父对象：

```
构造函数.prototype=father
```

强调：时机：在开始创建子对象之前，就要换。

3. 两种类型间的继承：更像 Java 的继承

问题：如果两种类型间拥有部分相同的属性结构和方法定义。

解决：抽象出一个父类型：

1. 父类型的构造函数中包含子类型相同的部分属性
2. 父类型的原型对象中包含子类型相同的部分方法
3. 在子类型构造函数中借用父类型构造

错误：直接调用父类型构造：

任何一个函数不用.不用 new 调用，其中的 this 默认指向 window
解决: 用 call 强行调用，并替换 this 为指定对象:
父类型构造.call(this,参数...)
call: 可强行调用一个函数，并替换函数中的 this
何时: 如果函数中默认的 this 不是想要的
如何: fun.call(obj,参数值...)
调用 fun，替换 fun 中的 this 为 obj
4. 让子类型原型继承父类型原型
Object.setPrototypeOf(
子类型.prototype, 父类型的 prototype
)

DAY3

正课:

1. ****ES5
****保护对象: 保护属性，防篡改
Object.create:
bind:
数组 API:
严格模式:
2. ES6
模板字符串:
let
箭头函数
for of
****class

1. ****ES5:

ECMAScript: ECMA 组织制定的 JavaScript 语言的国际标准，所有浏览器厂商遵照实现。规定了 JS 语言的核心语法。

保护对象:
为什么: JS 中的对象可随意修改属性值，可随意添加删除属性。
如何保护:
保护属性: 保护对属性值的修改
对象属性分为:
命名属性: 可直接用.访问到的属性
数据属性: 直接存储属性值的属性
如何保护: 四大特性:
value: 实际存储属性值
writable: 控制属性是否可修改
enumerable: 控制属性是否可被遍历
仅控制遍历，无法控制用.访问
configurable: 控制是否可删除属性
控制是否可修改其他两个特性
强调: configurable 经常作为前两个属性的双保险，且一旦设为 false，不可

逆!

如果查看四大特性:

Object.getOwnPropertyDescriptor(obj,"属性名")

如何修改四大特性:

```
Object.defineProperty(obj,"属性名",{
    特性:值,
    特性:值,
        ... ...
})
```

问题: defineProperty 一次只能修改一个属性

解决: 同时修改多个属性:

```
Object.defineProperties(obj,{
  属性名:{ 要修改的特性 },
  属性名:{ 要修改的特性 },
      ... : ...
})
```

问题: 无法使用自定义逻辑保护属性

解决:

访问器属性: 不直接存储属性值

仅提供对其他数据属性的保护

何时: 只要用自定义逻辑保护属性时

如何: 2 步:

1. 定义一个隐藏的数据属性实际存储属性值
2. 添加访问器属性保护隐藏的数据属性:

```
Object.defineProperty(obj,"属性名",{
  get(){//在试图获取属性值时自动调用
  //返回受保护的数据属性值
 },
 set(val){//在试图修改属性值时自动调用
  //参数 val 会自动获得要修改的新值
  //如果验证 val 符合规则
        //才将 val 赋值给受保护的属性
  //否则
     //报错!
 },
 enumerable:true,
 configurable:false
})
```

如何使用访问器属性: 同普通属性用法完全一致

其中赋值时，自动调用 set，取值时自动调 get

内部属性: 不能用.直接访问的隐藏属性

__proto__

防篡改: 保护对对象结构的修改

3 个级别:

1. 防扩展: 禁止添加新属性

Object.preventExtensions(obj)

原理: 每个 obj 内部都有一个隐藏属性:

Extensible，默认为 true

preventExtensions 将 obj 的 Extensible 改为 false

2. 密封: 在防扩展基础上，进一步禁止删除现有属性

Object.seal(obj)

原理: 修改 obj 的 Extensible 为 false

将所有属性的 configurable 都改为 false

3. 冻结: 在密封基础上禁止修改任何值

Object.freeze(obj)

原理: 修改 obj 的 Extensible 为 false

将所有属性的 configurable 都改为 false

还将所有属性的 writable 都改为 false

Object.create(): 可直接用一个父对象创建一个子对象。

如何: var child=Object.create(father,{

自有属性:{

value:值,

writable:true,

enumerable:true,

configurable:true,

},

... : {

...

}

});

强调: 只要添加到对象中的属性，四大特性默认为 false，必须显式写为 true。

原理: 1. 创建一个空对象

2. 让新对象自动继承 father

3. 为新对象扩展新的自有属性

何时: 如果没有构造函数，只有父对象，也想创建子对象时。

bind: 基于一个现有函数，创建一个新函数，并永久绑定 this 为指定对象。

何时: 只要回调函数中的 this 不是想要的，就可用 bind 替换。

如何: var newFun=fun.bind(obj)

基于现有函数 fun，创建一个功能完全相同的新函数 newFun，但永久绑定 newFun 中的 this 为 obj

鄙视: call,apply,bind

call 和 apply: 强行*调用*一个函数并*临时*替换函数中的 this 为指定对象。

差别: 仅在参数上:

call 要求传入函数的参数，必须单独传入逗号分隔

apply 要求传入函数的参数，必须放入数组中整体传入。*apply 可自动打散数组类型的参数*

bind: 基于一个现有函数*创建*一个功能完全相同的新函数,并*永久*绑定 this 为指定对象，还可永久绑定部分固定的参数值。

强调: 用 bind 绑定的 this，不能再被 call/apply 替换

总结: 只要函数中的 this 不是想要的:

call/apply: 如果立刻执行函数

call: 如果参数单独传入

apply: 如果参数放在数组中传入

bind: 希望创建一个新函数作为回调函数给别人用时

数组 API:
判断: 判断数组中的元素是否符合要求
返回值: bool
1. every: 判断数组中是否每个元素都满足要求
```
arr.every(function(val,i,arr){
  return 判断条件
})
```
2. some: 判断数组中是否包含满足要求的元素
```
arr.some(function(val,i,arr){
  return 判断条件
})
```
强调: 数组 API 的回调函数中 this 默认->window
所以，不能用 this 指代当前元素值
比如用 arr[i]或 val
遍历: 对数组中每元素执行相同的操作:
1. forEach: 对原数组中每个元素执行相同的操作
```
arr.forEach(function(val,i,arr){
  对当前元素执行的操作
})
```
2. map: 取出原数组中每个元素，执行相同操作后，放入新数组返回
```
arr.map(function(val,i,arr){
  return 对当前元素执行操作后的返回值
})
```
过滤和汇总:
1. 过滤: 复制出原数组中符合条件的元素组成新数组
```
var subArr=arr.filter(function(val,i,arr){
  return 判断条件
})
```
筛选出 arr 中符合判断条件的元素值，放入新数组返回。
2. 汇总: 将数组中所有元素，统计出一个汇总结果
返回值: 值
```
var r=arr.reduce(function(prev,val,i,arr){
  //prev: 截止目前的临时汇总值
  return prev+val
},startVal);
```
将 arr 数组中每个值累加后，求和。
其实: reduce 不一定非要从 0 开始累加
可从任意 startVal 开始累加

严格模式:
什么是: 比普通 js 运行模式更严格的模式
为什么: 解决普通 js 运行模式中固有的弊端和缺陷
何时: 2 种:
1. 新项目，建议必须使用严格模式
2. 旧项目，建议逐个模块向严格模式迁移
如何: 2 种:
1. 整个代码段启用严格模式:
在 script 元素或 js 文件的开头加入:"use strict";

2. 仅对单个函数启用严格模式
在 function 内，函数体的顶部加入:"use strict";
要求:
1. 不允许对未声明的变量赋值
2. 静默失败升级为错误
3. 不建议使用 arguments 和 arguments.callee 实现递归

2. ES6:

模板字符串: 对字符串拼接的简化:
何时: 只要字符串拼接时，需要动态执行表达式
如何: `反引号——ESC 键的正下方
模板字符串必须用``包裹
``中支持: 换行, 变量, 表达式
强调: 每个变量和表达式都要用${...}包裹

let: 创建一个仅在当前块内有效的变量
为什么:
1. js 中没有块级作用域，导致块内的变量会污染外部。
2. js 的声明提前会打乱程序的正常执行顺序
何时: 今后强烈建议用 let 代替 var
如何:
强调: let 必须配套严格模式使用
1. 解决声明提前: 检查 let a 之前不允许提前使用 a
变相强制将所有变量的声明放在当前作用域的顶部
2. 解决块级作用域:
let 声明的变量，仅在当前{}内有效，出了块失效。

箭头函数: 简化回调函数:
何时: 今后，几乎所有的回调函数都可用箭头函数简化
如何:
1. 所有回调函数都可: 去 function 改=>
2. 如果函数体只有一句话: 可省略{}
如果这一句话还是 return，可省略 return
3. 如果只有一个参数: 可省略()
但是,如果没有参数，必须保留空()
更大用途: 箭头函数内外共用同一个 this——取代 bind
特殊: 如果不希望内外共用 this，就不能用箭头函数
比如事件处理函数:

```
elem.addEventListener("click",function(){this->elem})
elem.addEventListener("click",()=>{ this->不是 elem})
变通解决:
elem.addEventListener("click",e=>{ e.target->elem})
```

for of: 简化普通 for 循环:
何时: 直接获得每个元素值时
如何: for(var val of arr){ val//自动获得当前元素值}
其中: of 会自动取出 arr 中每个元素的值，保存到 val

局限: 1. of 只能依次遍历每个元素
2. 按值传递,无法修改原数组中的值
如果要直接修改原数组中的值，还要用普通 for 循环
3. 只能遍历索引数组和类数组对象。不能遍历关联数组和对象。
如果遍历关联数组和对象，还要用 for in 循环

****class: 为了简化面向对象
1. 定义类型:
一个类型的所有属性和方法都集中包含在 class{} 中
类型名提升到 class 之后，称为类名
构造函数一律用 constructor 定义,其余保持不变
共有方法可直接放入 class 内，等效于自动放入原型对象中。

```
class Flyer {
  constructor(fname,speed){
    this.fname=fname;
    this.speed=speed;
}
fly (){//Flyer.prototype.fly
  console.log(...);
}
  }
```

2. 继承:

```
//让 Plane 继承 Flyer
class Plane extends Flyer{
 constructor(fname,speed,score){
  //super()指代父类型构造，且自动替换 this
      super(fname,speed);
  this.score=score;
 }
 getScore (){...}
  }
```

DAY4 项目课

DAY5 项目课

JQuery

DAY1

liwenhua@tedu.cn

复习:
第一阶段：HTML(4)、CSS(6)、Bootstrap(5)
第二阶段：JS(7)、DOM&BOM(8)、jQuery(6.5)

DOM 分为三部分：
(1)核心 DOM：操作任意标签树
(2)HTML DOM：操作 HTML 标签树
(3)XML DOM：操作 XML 标签树

常用的核心 DOM 操作
(1)查找元素的方法
document.getElementById('p1')
document.getElementsByName('uname')
document.getElementsByTagName('div')
document.getElementsByClassName('btn')
document.querySelector('选择器')
document.querySelectorAll('选择器')
遍历 DOM 节点：
node.parentNode
parent.childNodes、parent.children
node.nextSibling、node.previousSibling
(2)修改元素的属性
node.setAttribute('title','值')
node.getAttribute('title')
(3)修改元素的内容
element.innerHTML
element.textContent/innerText
(4)修改元素的样式
element.style.color = 'red';
element.className = 'btn btn-danger'
(5)修改元素的值
inputElement.value
(6)添加新元素
var newElement = document.createElement('div');
parent.appendChild(newElement)
(7)删除已有元素
parent.removeChild(node)
(8)替换旧元素
parent.replaceChild(oldChild, newChild)
(9)元素克隆
element.cloneNode()

总结：核心 DOM 操作的问题
- 方法名普遍比较长
- 操作比较僵硬
- 方法存在浏览器兼容性

1.jQuery 概述

jQuery **是一个 DOM 操作的函数库**，简化了常用的 DOM 操作。
理念：Write Less，Do More

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers.

jQuery 提供了四类函数：

(1)DOM 操作 —— 重点

(2)事件处理

(3)动画函数

(4)AJAX

jQuery 的版本问题：

(1)jQuery 1.x：比较大、功能偏弱、兼容老 IE

(2)jQuery 2.x：比较小、功能强大、不兼容老 IE

(3)jQuery 3.x：比较小、功能更强大、不兼容老 IE

面试题：jQuery3 的新特性有哪些?

HTML 中使用 jQuery，只需要使用 SCRIPT 标签引入 jquery-*.js 即可，会为 window 添加两个新的成员：

window.$

window.jQuery

2.jQuery 函数的特点

(1) $或者 jQuery 是一个函数，返回值是一个 jQuery 类数组对象

(2)即使没有查找到需要的元素，jQuery 类数组对象也不会是 null/undefined，调用 jQuery 函数不会报错!

(3)jQuery 类数组对象提供的函数都自带 for 循环遍历每个查找到的元素

(4)jQuery 函数底层都是 DOM 操作，所以可和原生的 DOM 操作组合使用

(5)原生 DOM 对象不能调用 jQuery 提供的函数;jQuery 函数返回的类数组对象也不能调用核心 DOM 成员

(6)原生 DOM 对象和 jQuery 对象间如何转换：

原生 DOM 对象封装到一个 jQuery 类数组对象

```
$(domObject)
```

jQuery 类数组中取出封装的 DOM 对象：

```
$('button')[index]
```

(7)jQuery 对象方法的返回值一般还是当前选定的类数组对象，可以实现“链式调用”

练习：使用原生 DOM 和 jQuery 两种方式实现“按钮的点击计数”：

```
<button>0</button>
```

原生：　obj.onclick = fn

jQuery：　$(...).on('click', fn)

3.jQuery 函数第一部分：DOM 操作函数 —— 查找元素

```
$( '选择器' )
```

jQuery 的选择器语法支持所有的 CSS 选择器语法！并屏蔽了浏览器兼容性；同时还扩展了一些新的选择器语法。

面试题：jQuery 中的哪些选择器是 CSS 中没有的? —— 执行效率较低

第一组：基本选择器　　—— 重点

(1)#id

练习：点击一个 button#bt1，让下方的一个 p#p1 隐藏起来

提示：jQuery 类数组对象提供的.hide()函数可以隐藏一个元素

(2).className

练习：点击一个 button#bt2，让下方所有的 span.badge 字体变大为 2em

提示：jQuery 类数组对象提供的.css('color', 'red')函数可以修改当前选定元素的指定样式

(3)标签名

练习：li 元素上发生鼠标进入事件时，当前 li 显示出下边框

提示：使用.on('mouseover', fn)监听鼠标进入事件

(4)*

练习：单击 button#bt3 后，让所有的元素的 box-sizing 变为 border-box，所有元素的 margin 变为 0

(5)div, p, #c3

练习：页面加载完成后，为所有的 button、.btn、role="button"的元素，添加事件监听，点击后打印出当前系统时间

第二组：层级选择器　　—— 重点

(6) ancestor descendant　后代选择器

练习：为所有#alert1 中的.btn 添加事件监听，单击后，隐藏#alert1

(7) parent > child　直接子代选择器

鱼香肉丝：为所有#alert2 中的直接子元素.btn 添加事件监听，单击后，隐藏#alert2

青椒肉丝 **+ next**　　下一个相邻兄弟选择器

京酱肉丝]：为 UL 中的 LI 添加上边框，实现如下效果：

番茄肉丝

$('li + li').....

$('li:not(:first-child)')....

(9) prev ~ siblings　后续的所有兄弟选择器

第三组：基本过滤选择器　　—— 重点

注意：基本过滤选择器把所有满足选中的元素放在一个大集合中进行排序，不论是否在同一个父元素中与否。下标从 0 开始！

(10) :first

练习：把所有列表中的第一个 LI 字体加粗显示

(11) :last:

练习：把所有列表中的最后一个 LI 字体斜体显示

(12) :eq(index)

练习：把所有列表中的第 index 个 LI 添加有边框

(13) :gt(index)

练习：把所有列表中的下标大于 index 的 LI 添加删除线

(14) :lt(index)

练习：把所有列表中的下标小于 index 的 LI 添加下划线

(15) :odd

练习：把所有列表中的下标为奇数的 LI 背景颜色变为淡黄色

(16) :even

练习：把所有列表中的下标为偶数的 LI 背景颜色变为淡蓝色

(17) :not(selector)

第四组：子元素过滤选择器　　—— 重点

注意：在每个父元素中进行分组，查找指定的子元素。下标从 1 开始。

(18) :first-child

$('li:first-child');

练习：把每个列表中的第一个子 LI 字体变浅蓝

(19) :last-child

练习：把每个列表中的第最后一个子 LI 字体变红色

(20) :nth-child(index)

练习：把每个列表中的第 2 个子 LI 字体添加绿色背景

$('li:nth-child(2)')...

$('li:nth-child(odd)')　　$('li:nth-child(2n+2)')

$('li:nth-child(even)')　　$('li:nth-child(2n)')

(21) :only-child

$('li:only-child')

练习：实现一个表中 TBODY 中的 TR 隔行变色，使用基本过滤选择器或者子元素过滤选择器的区别是什么？

提示：要想看出区别，必须至少创建 2 个 TABLE！且第一个 TABLE 的行数必须是奇数！

第五组：属性选择器

(22) [attribute]

选中所有具备 title 属性的 a 元素：　$('a[title]')...

(23) [attribute=value]

选中所有具备 data-toggle 属性且值为 dropddown 的元素：

$('[data-toggle="dropdown"]')....

(24) [attribute!=value]

(25) [attribute^=value]

(26) [attribute$=value]

(27) [attribute*=value]　　具备指定的属性，且值中包含指定字符

第六组：可见性选择器

(28) :visible　　$(':visible')选中所有可见元素

(29) :hidden　　$(':hidden')选中所有隐藏元素

测试：哪些是:hidden 可以选中的？

- display: none　　可以
- visibility: hidden　　不可以
- opacity: 0　　不可以
- input[type="hidden"] 可以

第七组：内容过滤选择器

(30) :contains(txt)

练习：选中文本中包含“提交”字的 button 元素，让它们变为绿色按钮

(31) :has(selector)

练习：选中包含.close 按钮的.alert 元素，让它们变为红色的警告框；

选中不包含.close 按钮的.alert 元素，让它们变为红色的警告框

(32) :empty

练习：选中内容为空的警告框　　$('.alert:empty').....

(33) :parent

练习：选中内容不为空的警告框，即包含子元素或文本内容

$('.alert:parent').....

第八组：表单元素选择器

: input :text :password :radio

:checkbox :submit :image :reset

:button :file :hidden

:enabled :disabled :checked :selected

<input type="image" src="img/1.jpg">

DAY2

复习：

W3C DOM 把 HTML 文档看做树型结构，并提供操作节点方法；

jQuery 是一个函数库，用于简化 DOM 操作，屏蔽了浏览器兼容性问题。函数分为四类：

(1)DOM 操作

(2)事件处理

(3)动画

(4)AJAX

jQuery DOM 操作——查找节点

jQuery('选择器') =》 类数组对象

基本选择器：

#id、.class、div、*、button,.btn

层级选择器：

parent child
parent > child
prev + next
prev ~ siblings

基本过滤选择器：

:first :last :eq(i) :gt(i) :lt(i)
:odd :even :not(selector)

子代过滤选择器：

:first-child :last-child :nth-child(i)
:nth-child(odd) :nth-child(even)

属性选择器：

[attr="value"]

可见性选择器：

:visible :hidden

内容选择器：

:contains(txt) :has(selector)
:empty :parent

表单元素选择器：

:text :radio :submit
:disabled :enabled :checked :selected

今日目标：

(1)操作元素的属性、内容、样式、值 —— 重点
(2)增删替换克隆元素 —— 重点
(3)事件处理 —— 重点

1.操作元素的属性

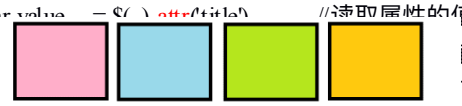
<a href="" title="" data-toggle="dropdown">

核心 DOM：

element.getAttribute('title')
element.setAttribute('title', 'abc')

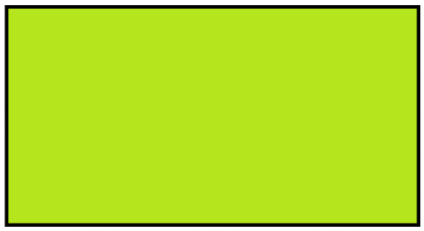
jQuery：

var value = $(...).attr('title') //读取属性的值



值

可以使用 attr() 方法，也可以使用 data() 方法，如： <a data-n



大图
">

2.操作元素的内容

核心 DOM：

```
var h = element.innerHTML;
element.innerHTML = h;
var t = element.innerText/textContent;
element.innerText/textContext = t;
```

jQuery：

```
$(..).html( )        //读取 innerHTML
$(..).html( 'html' )        //设置 innerHTML
$(..).text( )        //读取 innerText
$(..).text( 'txt' )        //设置 innerText
```

练习：点击一个 Button，上方显示出已点击的次数

练习：再次点击即显示下一张图片

```
1 再次点击即显示下一张图片
2<img src="1.jpg">
3 再次点击即显示下一张图片
4<img src="2.jpg">
5 再次点击即显示下一张图片
6 ....
```

3.操作元素的样式

核心 DOM：

```
var c = element.style.color          //读取行内样式
element.style.color = 'red'          //设置行内样式
var n = element.className            //读取 ClassName
element.className = n                //设置 ClassName
```

jQuery：

```
$(..).css('color')          //读取指定样式的值
$(..).css('color', 'red')   //设置行内样式
$(..).addClass('alert')     //添加一个 class
$(..).removeClass('alert')      //删除一个 class
$(..).hasClass('alert')     //判断选定元素是否具有指定 class
```

练习：实现一个“双态按钮”，点击一次变为深色，再点击变为浅色

```
.up {   }                   .down {   }
```

4.操作表单元素的值

核心 DOM：

```
    var v = input.value             //读取值
    input.value = 'v'           //设置值
```

jQuery：

```
    $(...).val( )               //读取值
    $(...).val('value')  //设置值
```

练习：“用户名”和“密码”输入框，点击“提交注册信息”按钮后，读取用户的两个输入，打印出来，在输入框中清除用户的输入

面试题：在操作元素的相关属性时，使用 attr()、val()、prop()、data()有何区别?

attr()一般只用于操作元素的 HTML 字面属性，如 src、href、name...

val()操作的是 HTML 元素对应的 JS 对象的 value 属性

prop()操作的是 HTML 元素对应的 JS 对象的 disabled、readyonly、selected、checked 等 Boolean 类型属性

data()操作的是 HTML 元素对应的 JS 对象的扩展数据属性（对象缓存数据），而 attr('data-xx')读取/修改的 HTML 元素字面属性

5.遍历 DOM 树上的节点

核心 DOM：

```
element.parentNode              //寻找父节点
element.childNodes/children     //获取子节点
elemnet.nextSibling             //获取下一个兄弟
element.previousSibling             //获取上一个兄弟
```

jQuery：

```
$(..).parent()              //返回选定元素的父节点
$(..).children()            //返回所有子节点
$(..).next()            //返回下一个兄弟
$(..).prev()            //返回上一个兄弟
$(..).siblings()            //返回所有的同辈兄弟
```

十元套餐　二十元套餐　三十元套餐

```
<ul class="tabs">
    <li><a>十元套餐</a></li>
    <li class="active"><a>二十元套餐</a></li>
    <li><a>三十元套餐</a></li>
</ul>
```

6.添加新的元素

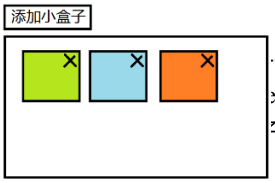
核心 DOM：

```
var li = document.createElement('li')     //创建子节点
//修改 li 的属性....                          //设置其属性
ul.appendChild( li );                       //添加到父节点
```

jQuery：

```
var li = $('<li class="item">内容</li>')  //创建子节点
//li.click(fn)
$('ul').append( li );                       //在父节点最后追加子节点
```

添加小盒子　　//子节点追加到父节点最后



----

em">内容
')

```
function rc(){                    //Random Color:   返回一个随机的颜色
    var r = Math.floor(Math.random()*256)
    var g = Math.floor(Math.random()*256)
    var b = Math.floor(Math.random()*256)
    return `rgb(   ,   ,   )`;
}
```

7.删除已有的元素

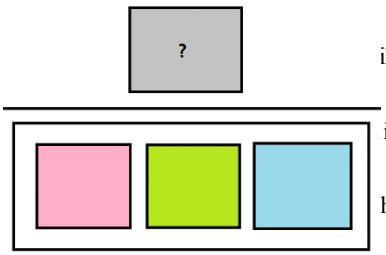
核心 DOM：

```
ul.removeChild( li )              //由父元素删除孩子
```

jQuery：

```
$('li').remove()                  //删除当前选定元素
```

8.替换已有元素



ild);

ild) //已有节点用新节点替换，返回被删除的旧节点

//新节点替换所有的旧节点，返回新节点

hosen div#list

9.克隆节点

核心 DOM：

```
var copy = element.cloneNode( )
```

jQuery：

```
var copy = $(..).clone()          //返回选定元素的副本
var copy = $(..).clone(copyListener)   //参数指示是否复制选定元素绑定的监听函数，默认为 false，不复制监听函数
```

练习：改进“英雄选择”应用，要求点击某个小飞机后，下方仍然有该飞机，上方选中区域出现当前飞机的一个副本

10.jQuery 函数第二部分：事件处理函数

jQuery 的历史上先后出现了若干事件处理函数：

(1)bind() / unbind() 已废弃

(2)one(事件名称, fn)仅对指定事件监听一次

(3)live() / die() 已废弃

(4)delegate() / undelegate() 已废弃

(5)on() / off()

(6)click() / mouseover() / mouseout() / keyup() ...

$(..).click(fn) <=> $(..).on('click', fn)

on()函数的第一种使用方法——直接绑定在事件源上：

$('事件源').on('事件名称', fn)　　//绑定监听函数

$('事件源').off('事件名称')　　//取消所有监听函数

练习：点击“开始抽奖”按钮，命令行中输出“抽奖中”，按钮上的文字也变为“抽奖中”；要求此后此按钮再被点击无任何处理函数了。

on()的第一种用法有两个限制：

(1)若选中元素很多，每个都会有一个监听函数

(2)无法为后添加的元素执行绑定

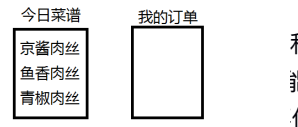
on()函数的第二种使用方法——委托给父元素进行事件代理：

$('parent').on('事件名称', '子元素选择器', fn)

DOM 中为元素绑定监听函数： btn.onclick = function(){　} btn.addEventListener('click', function(){　})
jQuery 中的 on()函数底层是 addEventListener

课后练习：

(1)jQuery DOM 操作中未讲解的函数：jQuery 手册“文档处理”章节中未讲函数



select>option

V1: 在“今日菜谱”中点击后，从“今日菜谱”跑到“我的订单”

V2：在“今日菜谱”中点击后，复制一份到“我的订单”

和事件绑定

能

件委托给父元素，从而减少监听函数的数量

DAY3

复习：

jQuery 是一个函数库，主要用于简化 DOM 操作，屏蔽浏览器的兼容性；思路仍旧是“先查找元素，再操作元素”；所有函数分为四类：

(1)DOM 操作

(2)事件处理

(3)动画函数

(4)AJAX 操作

DOM 操作——增删改查
(1)查找元素： $('选择器') jQuery('选择器')
(2)操作元素的属性： $(..).attr()
(3)操作元素的内容： $(..).html() $(..).text()
(4)操作元素的样式： $(..).css()
$(..).addClass() $(..)removeClass()
$(..).hasClass() $(..).toggleClass('.btn')
(5)操作元素的值： $(..).val()
(6)遍历元素：
$(..).parent()
$(..).children()-直接子代
$(..).find()-所有子元素
$(..).next() $(..).nextAll()
$(..).prev() $(..).prevAll()
$(..).siblings()
(7)插入节点： $(..).append() $(..).prepend()
(8)删除节点： $(..).remove() $(..).empty()
(9)替换节点： $(..).replaceWith() $(..).replaceAll()
(10)克隆节点： $(..).clone()

事件处理函数：
(1)$('.btn').one('click', fn)
(2)$('.btn').click(fn) ....
(3)$('.btn').**on**('click', fn) off(..)
$('.container').**on**('click', '.btn', fn) off(..)

让 jQuery 放弃使用 $ ：
jQuery.noConflict()

目标：
(1)补充事件处理面试题
(2)jQuery 中的动画函数
(3)jQuery 中的插件

1.面试题：window.onload 和$(document).ready()的异同?

window.onload 是核心 DOM 的写法：
window.onload = function(){ ... }
只能为绑定一次；
只有全部的网页内容(html/css/js/图片....)加载完成才能触发。

$(document).ready()是 jQuery 的写法：
$(document).ready(function(){ ... })
底层是 addEventListener('DOMContentLoaded', fn)，可以先后绑定多次；
只要“DOM 内容(只包括 html/js)加载完成”即可触发。

2.补充：jQuery 中的 hover()函数

jQuery 监听“鼠标进入+鼠标离开”有如下三种方法：

(1) $(..).mouseover(fn) + $(..).mouseout(fn)

(2) $(..).mouseenter(fn) + $(..).mouseleave(fn)

(3) $(..).hover(fn, fn)

注意：方法 3 等同于方法 2

3.补充：jQuery 中的 trigger()函数

使用 JS 代码代替用户触发指定的事件，调用之前绑定的监听函数。

$('.btn').trigger('click')

可以简写为：

$('.btn').click( )

4.jQuery 中的函数第三部分：动画函数 —— 隐藏和显示动画

隐藏和显示函数通过使用定时器修改目标元素的 **width / height / opcaity** 三个样式的值来实现动画：

$(..).show() $(..).show('slow/normal/fast') $(..).show(3000)

$(..).hide() $(..).hide('slow/normal/fast') $(..).hide(3000)

练习：隐藏和显示之间切换

练习：函数实现“手风琴”组件

西游记简介

红楼梦简介

水浒传简介

5.jQuery 中的函数第三部分：动画函数 —— 折叠展开/收起动画

折叠展开/收起动画函数通过使用定时器修改目标元素的 **height** 一个样式的值来实现动画：

$(..).slideUp( ) $(..).slideUp('slow/normal/fast') $(..).slideUp(300)

$(..).slideDown( ) ....

$(..).slideToggle( ) ....

练习：仿写 Bootstrap 中的响应式导航条在手机中的效果

```
<div class="navbar">
<div class="navbar-header">
    <a class="navbar-brand">TARENA</a>
    <a class="navbar-toggle">三</a>
</div>
<div class="navbar-collapse">
    ul>li>a
</div>
</div>
```

6.jQuery 中的函数第三部分：动画函数 —— 淡入/淡出动画

淡入/淡出动画函数通过使用定时器修改目标元素的 **opacity** 一个样式的值来实现动画：

$(..).fadeIn( )　$(..).fadeIn('slow/normal/fast')　$(..).fadeIn(300)

$(..).fadeOut( )

$(..).fadeToggle( )

提示：上述六个动画函数都可以在最后接收一个参数——函数

$(..).fadeOut(300,　function(){ //在动画结束时的回调函数 })

练习：点击小星星后，闪动 3 次，最后消失

6.jQuery 中的函数第三部分：动画函数 —— animate()

$(..).animate({

　　属性 1: 值 1;

...

}, 300, fn)

动画排队：执行完一个动画后，再执行另一个

动画并发：同时执行多个属性的动画效果

animate({})可以对哪些 CSS 属性执行动画?

(1)width、height、opacity、fontSize....等等有可渐变属性值（数值型）的样式可以执行动画

(2)display、fontFamily、transform、颜色类属性等没有渐变属性值的样式不能执行动画

练习：

(1)在屏幕左上角的小星星，　点击后从左移动到屏幕右边

position: ... ;　　left: 0;

(2)在屏幕左上角的小星星，点击后从左移动到屏幕右边，再移动到下边——走直角

(3)在屏幕左上角的小星星，点击后从左上角移动到屏幕右下边，走斜线

(4)点击小星星，旋转的同时变大、变淡....　直至消失

Web 应用中可用的动画技术：

(1)CSS3 Transition

(2)CSS3 Keyframes

(3)定时器 + 属性修改　　jQuery1/2 动画函数

(4)requestAnimationFrame　　jQuery3

7.jQuery 类数组对象的操作

window.$　<=> window.jQuery

$()函数或 jQuery()返回值是一个“类数组对象”—— 有点像数组，但不是 Array 类型的实例，其中封装着查找到的所有 DOM 元素。该对象称为“jQuery 对象”，其类数组相关操作：

$(..).length　　获取类数组中封装的 DOM 对象的数量

$(..)[index]　　获取类数组中封装的第 index 个 DOM 对象

$(..).get(index)　　获取类数组中封装的第 index 个 DOM 对象

$(..).each(fn)　　遍历类数组中封装的每一个 DOM 对象，针对每个 DOM 元素执行一次指定的回调函数

$(..).index(domObj)　返回指定的 DOM 元素在当前类数组中的下标

练习：假设有如下成绩列表，请给每个不足 60 分的成绩+10 分，并将超过 90 分的成绩用绿色背景标

识出来

```
<ul>
<li>98</li>
<li>33</li>
<li>52</li>
</ul>
```

请打分：  的全部改变背景色

练 听

用 jQuery 中的 DOM 操作实现省/市联动下拉菜单

要求：(1)点击“添加商品”按钮后，可以在表格中显示一个新的商品，价格随机生成，数量随机生成(整数)，可以自动显示出“小计”；

(2)“数量”显示在一个输入框中，只要用户的输入改变了，后续的“小计”随之改变；

(3)只要“添加商品”、“删除商品”被点击，或者“数量”被改变，必须重新计算出“购物车总金额”。

DAY4

复习：

见思维导图

1.补充：页面 DOM 内容加载完成后执行指定的函数

```
$(document).ready(fn)
    $().ready(fn)
    $(fn)
```

中的插件函数

，在现有的功能基础上添加更多的功能，扩展整体的应用。

jQuery 中的插件（即函数）分为两类：

(1)jQuery 全局插件函数

原本要声明的工具函数（如 max()/min()）如果声明为全局函数，会造成“全局对象(window)的污染；为了避免污染全局对象，可以把这些函数纳入到 jQuery 对象的名下——与 jQuery 没有必然的关系：

声明方式：
```
jQuery.max = function(arr){  }
jQuery.extend({ max: fn,  min: fn})
```
调用方式：
```
jQuery.max([10,38,50])
```

(2)jQuery 对象插件函数

jQuery 对象插件函数就是为所有的 jQuery 对象（$()函数的返回值）添加的公共函数，用于操作当前选定的 DOM 元素。

声明方式：
```
jQuery.fn.max = function( ){  }
jQuery.fn.extend({ max: fn,  min: fn})
```
调用方式：
```
$('li').max( )
```

jQuery(..) 或 $(..) 的返回值是一个类数组对象——“jQuery 对象”，所有的 jQuery 对象的原型：jQuery.fn;

若想给所有的 jQuery 对象都添加一个扩展函数，只需要加给 jQuery.fn 即可

练习：为所有的 jQuery 对象添加扩展的插件函数：

$('p').sum( )　　返回当前选定的所有元素的内容数字和

$('li').avg( )　　返回当前选定的所有元素的内容数字平均值

练习：仿 Bootstrap 中下拉菜单编写一个 jQuery 插件函数：

```
<div class="dropdown">
    <a href="#">产品大全</a>
    <ul class="dropdown-menu">
        <li><a>冰箱</a></li>
        <li><a>洗衣机</a></li>
```

十元套餐　二十元套餐　三十元套餐

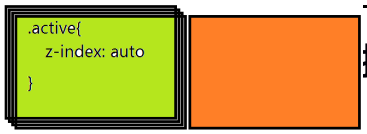
二十元套餐的详情.....

own( );

'一个 jQuery 插件函数：

```
    <ul class="nav">
    <li><a href="#tc10">十元套餐</a></li>
    <li class="active"><a href="#tc20">二十元套餐</a></li>
    <li><a href="#tc30">三十元套餐</a></li>
</ul>
<div class="tab-content">
    <div id="tc10" class="tab-pane">十元套餐详情...</div>
    <div id="tc20" class="tab-pane active">二十元套餐详情...</div>
    <div id="tc30" class="tab-pane">三十元套餐详情...</div>
</div>
```

JS 调用：
```
$('.nav > li > a').tab()
```



播广告

后面几张：z-index:-1

$('..').carousel( );

4.复杂插件的编写：滚动监听

$('..').scrollspy( );

课后练习：

仿写上述的滚动监听插件(scrollspy)，实现如下效果：随着页面滚动到特定的楼层，该楼层起始处的灰色气泡变绿

AJAX

DAY1

今天学习的内容

程涛 chengtao@tedu.cn

1:整体课内容

2:ajax 内容安排

3:今日目标

3.1:服务器与数据库服务器概述

3.2:sql -- 重点&难点

#----------

1:整体课内容

第一个阶段:html(4)+css(5)=静态页面

第二个阶段:js(7)+dom(6.5)=用户交互(特效)

第三个阶段:ajax(9)+h5 高级(6.5)=动态页面(全栈)

第四个阶段:框架

2:ajax 内容安排

数据服务器 -- sql:重点难点

web 服务器 -- php:重点

http 协议

原生 ajax -- ajax 流程:(抽象)

jquery 中的 ajax

3:今日目标

3.1:服务器与数据库服务器概述

思考问题:开发网站(网站如何工作)

第一个?上万或百万商品信息

(图片,价格,简介,)保存在哪里?

?如何保存

?如何获取

?如何显示

解决问题:

第一个问题:软件(数据库)
第二个问题:如何保存 表
创建一张表:商品表
商品图片 商品价格 商品名称 简介 ...
01.jpg 1000.00 运动鞋 跑鞋
02.jpg 330.00 运动鞋 拖鞋
...
第三个问题:如何从数据库中将数据表中数据
抓取出来并且显示.
解决方法:语言 php 负责从数据库中抓取数据
并且显示给用户.

a:什么是服务器
能够在网站中提供各种(浏览网页,收发邮件
视频,语言)等服务器的软件与硬件集合。
硬件服务器:高可用,高性能计算机
软件服务器(通常):(WEB/FTP/EMAIL/DB/DNS..)
web:apache
db:mysql
b:数据库服务器(软件)
数据库软件特点：所有其它软件所没有
1:永久海量存储数据
2:高速查询
c:数据库服务器分类
网状数据库:
树型数据库:
关系型数据库:
MySQL***/Oracle/SQLServer/DB2
NOSQL 数据库:Redis
d:什么是关系型数据库
以横行竖列的方式保存数据,
mysql
e:关系型数据库层次
数据库软件(mysql)-->库-->表-->行-->列
f:mysql 之父:马丁
MYSQL AB(瑞典)->SUN->Oracle
g:mysql 版本:
主版本: 企业版(收费)社区版(免费)
小版本:5.0 5.1 5.2 5.4 5.5 5.6 5.7 ..
建议:下载哪个版本
?如果开发全新网站项目:新版本 5.6
?如果二次开发选旧版本.
g:下载
https://www.oracle.com 官网
https://www.mysql.com/downloads/
搜狐镜像
http://mirrors.sohu.com/
mysql windows(64/32)(zip/exe)

linux

unix

mysql-5.6.34-win32.msi (32)

i:使用软件包(XAMPP)

(包含:mysql/php/apache/..)

如何使用 xampp

1:双击 xampp 图标

2:点击 apache [start]启动(饭店)

点击 mysql [start]启动

3:点击 右侧[shell]

4:输入命令

mysql -uroot -p

#mysql 指令专门进入数据库系统

#-u 指定用户名

#root root 是 mysql 软件最高级别用户

#-p root 用户密码

5:退出软件

#exit

#点击右上角[X]

练习:进入 mysql 10:45--10:47

3.2:sql -- 重点&难点(熟练)

sql(结构化查询语言)专用于(增删改查)数据库

中数据语言.

使用 sql 两种方式:

a:交互模式(上课)

一行一行执行指令

用户输入一个 sql 指令,mysql 执行一个指令.

b:脚本模式(做项目)

把所有 sql 指令保存在一个.sql 文件中

一次执行在 mysql 中执行

3.2.1 sql 语句的分类

1:DDL:数据定义指令

CREATE/DROP/ALTER/TUNCATE

2:DML:数据操作指令

INSERT/UPDATE/DELETE

3:DQL:数据查询指令

SELECT

4:DCL:权限指令

GRANT/REVOKE

熟练掌握:

[CREATE/INSERT/UPDATE/DELETE/SELECT]

3.2.2: #CREATE 指令功能创建(库)创建表

指令格式:

CREATE DATABASE 库名;(库)

建议:(库名/表名/列名)

:英文不要数字或特殊字符开头

:中文空格不要

test01 ok

2017test error

test 02 error

大旭之家 error

示例:创建库 test001

第一步创建库:

第二步查看库是否创建成功:

CREATE DATABASE test001;

SHOW DATABASES;

练习:11:25--11:30

创建二个库 test002 test003

注意:

a:通常 sql 指令大写，库名、表名、列名小写

b:出错处理:如何处理常见错误

语法错误:SQL syntax

3.2.3: #CREATE 指令功能创建表

a:列数据类型(常用列类型)

int	整型(年龄) 范围-21 亿~21 亿
varchar(10)	字符串(10 个字符[数字，字母，汉字])
double(10,2)	浮点(小数)总长 10 位其中 2 位小数
datetime	日期和时间

分析表:商品表

1:	商品编号	int 1 2 3 5
2:	商品名称	varchar(20)
3:	商品上架时间	datetime
4:	商品价格	double(10,2)

b:创建表语法

```
CREATE TABLE 表名(
    列名 1 列类型,
        列名 2 列类型,
列名 3 列类型
    );
```

C:指令 use 库名称;

D:查看表是否创建成功

SHOW TABLES;

```
use test001;
    CREATE TABLE t_product(
        id INT,
    name VARCHAR(20),
            ctime DATETIME,
            price DOUBLE(10,2)
    );
    SHOW TABLES;
```

练习:11:51--11:56 完成上述练习

小结上午重点

1:创建库

CREATE DATABASE 库名; 创建库

SHOW DATABASES; 查询库是否创建成功

2:创建表

USE 库名;

```
CREATE TABLE 表名(
 列名 列类型(int/varchar/double/datetime)
);
SHOW TABLES;
```

中午练习:

1:创建学生表(编号，学号，姓名，年龄)

```
USE test001;
CREATE TABLE t_stu(
    id INT,
      no VARCHAR(11),
      name VARCHAR(20),
      age INT
);
```

#建议:一张表总有一个编号 id 类型 INT

2:创建班级表(编号，班级名称，班级人数)

```
USE test001;
CREATE TABLE t_class(
    id INT,
      name VARCHAR(20),
      count INT
);
```

3.2.4:向数据库中添加记录(行)

希望:

t_stu

id	no	name	age
1	wsl001	赵大旭	19
2	wsl002	张东	18
3	wsl003	文华	17

a:标准语法

a.1:向所有列添加数据

```
INSERT INTO 表名 VALUES(列值 1,列值 2,列值 3,.);
INSERT INTO t_stu VALUES(1,'wsl001','xuxu',19);
INSERT INTO t_stu VALUES(2,'wsl002','dongdong',18);
INSERT INTO t_stu VALUES(3,'wsl003','wenhua',17);
INSERT INTO t_stu VALUES(4,'wsl004','wenhua1',17);
INSERT INTO t_stu VALUES(5,'wsl005','wenhua2',17);
INSERT INTO t_class VALUES(1,'WEB',100);
INSERT INTO t_class VALUES(2,'mysql',101);
```

INSERT INTO t_class VALUES(3,'php',100);

验证:

SELECT * FROM t_stu;

#字符串类型列加单引号

#小心中文/(英文)单引号

#小心日期写法

#SQL 指令结束使用分号

a.2:向部分列添加数据

INSERT INTO 表名(列名 1，列名 2)VALUES(列值 1,列值 2,..);

INSERT INTO t_stu(id,no)VALUES(4,'tao');

a.3:日期类型 '2017-04-12' 字符串

now() 函数(当前日期时间)

INSERT INTO t_product VALUES(10,'book','2010-10-10',100.99);

INSERT INTO t_product VALUES(20,'js','2017-04-12',99);

INSERT INTO t_product VALUES(30,'css',now(),98);

14:45--14:47 向学生表添加 5 条记录

向班级表 3 条记录

3.2.5:删除记录(行)

!!!恢复很困难

#标准语法:

条件: = < > <= != <>

DELETE FROM 表名 WHERE 条件;

示例:

DELETE FROM t_product id = 10; //ok

DELETE FROM t_product price = 100.99;

DELETE FROM t_product name = 'book';

3.2.6:更新记录(行)

#标准语法

UPDATE 表名 SET 列名 1=新值 1

,列名 2=新值 2 ..

WHERE 条件;

UPDATE t_stu SET age = 16,name='daxu'

WHERE id = 1;

练习 1:更新年龄 21 编号 id=1

练习 2:更新年龄 22 名称='daxu' id=2

练习 3:更新学生编号 wsl009 编号 id=3

15:15--15:18

综合练习一.txt

根据注释编写 SQL

3.2.7: 查询记录(行)

语法:SELECT 列名1,列名2,...
FROM 表名
WHERE 条件
ORDER BY 列名1 #排序

SELECT * FROM 表名; # *所有列
SELECT id,name FROM t_emp;
SELECT sal FROM t_emp;
SELECT * FROM t_emp;
SELECT id,name,sal,sal*1.2 FROM t_emp;

小结 SQL:
a:分类
DDL:数据定义语句
CREATE/DROP
DML:数据操作语句
DELETE/UPDATE/INSERT
DQL:数据查询语句
SELECT
DCL:
b:常用 SQL
CREATE DATABASE 库名; 创建库
SHOW DATABASES; 查询库名
CREATE TABLE 表名(
列名1 列类型,
...
);
列类型(INT/VARCHAR(10)/DOUBLE(10,2)/DATETIME)
USE 库名; 进入指定库中
SHOW TABLES; 查询表名
INSERT INTO 表名 VALUES(值1,值2,...);
INSERT INTO 表名(列1,列2)VALUES(值1);
字符串值:''
日期值 :'' now()
DELETE FROM 表名 WHERE 条件;
UPDATE 表名 SET 列名=新值1,列名=新值2
WHERE 条件;
SELECT * FROM 表名;#*所有列
SELECT id,name FROM 表名;
ORDER BY 列名;
ORDER BY 列名 DESC;
count(列)/sum(列)/max(列)/min(列)/avg(列)
子查询

用户表:

编号 id INT
用户名 name VARCHAR(20)
密码 pwd VARCHAR(32)
手机 phone VARCHAR(30)
00861-13999999999

作业一:
建议基础弱一些:
三遍:综合练习一.txt
作业二:
1:创建 jd.sql 文件
2:删除数据库 jd 如果存在的话
3:创建数据库 jd 指定编码 utf8
4:创建用户表
t_user(id/name/pwd/pic 头像/ctime 创建时间)
5:向用户表添加三条记录
6:创建用户评论表
t_review(id,title 标题/ctime 创建时间/content 内容)
7:为每个用户添加三条评论
8:查询所有评论
9:删除用户编号为 1 用户及其所有评论

DAY2

今天学习的内容
1:复习昨天知识重点
2:作业/完成综合练习
3:今天目标 web 服务器
3.1:web 服务器与 php 简介 了解
3.2:php 语法 必须掌握
3.3:使用 php 操作数据库 重点&难点
#----------
1:复习昨天知识重点
a:什么服务器
b:分类
硬件服务器:高性计算机
软件服务器:MySQL/Apache
c:数据库服务器
解决二个问题:
海量存储/高速查询
d:SQL
DDL:CREATE/DROP
DML:INSERT/DELETE/UPDATE
DQL:SELECT
DCL:
e:图:工作原理-饭店.jpg 电话订餐
2:作业/完成综合练习
jd.sql

综合练习一.txt
处理中文:
1:将所有指令写脚本文件中
2:打开 mysql 执行指令窗口
3:输入二个命令
SET NAMES utf8;
SOURCE d:/tao/ajax/day02/dangdang.sql
问题:看到窗口中文不正确
原因:库/表 正确(utf8)
windows{窗口} gbk
解决:SET NAMES GBK;{临时将文字转换 gbk 显示}
10:38--10:42
3:今天目标 web 服务器
3.1:web 服务器与 php 简介 了解
a:web 服务器(软件 apache)
b:作用:负责接收客户端请求,
理解请求内容,找到请求资源并且
返回客户端浏览器
(电话餐厅:大堂经理)
c:web 服务器种类
c.1:静态 web 服务器(apache)
提供的内容任何时间任何人访问
都是完全相同.
属于静态 web 技术范畴
HTML/CSS/JS/FLASH/视频/音频
c.2:动态 web 服务器(apache)
提供的内容在不同的时间不同人访问
可能变化属于动态范畴的技术
1:JSP = HTML+JAVA
2:ASP.NET = HTML+C#
3:PHP = HTML+PHP
4:Node.JS = HTML+NodeJS

#面试题:如何自学一门新的编程语言
1)了解背景,历史,现状,发展,特点,应用领域
2)建设环境=写出 HelloWorld
3)数据类型
4)变量常量
5)运算符
6)逻辑结构
7)通用小程序
8)函数,对象
9)常用函数库,类库,中间件,框架
10)实用项目

#php 背景特点

php 是一种运行在服务器端(apache)的编程序言,用于
生成动态网页内容.

2000,Zend 公司成立，维护 php 语言.发布 php 解释器环境
php 特点:开源，简单，(易上手),
跨平台(windows/linux),占用资源少
尤其适合中小型 web 应用开发
(微博/微信/论坛)...

#搭建软件环境
1:服务器端环境创建:下载选择 web 服务器
apache/Microsoft IIS/Nginx
c:/xampp/apache/
2:服务器端:下载并且安装 php 解析软件
c:/xampp/php/php.exe
3:服务器端:编写并保存 php 程序保存哪里
!!!
c:/xampp/htdocs/1.php
4:服务器端:启动 web 服务器
[start]
5:客户端:打开浏览器/直接输入服务器程序地址
按回车
http://127.0.0.1/1.php

练习:1:创建 2.php 输出你的名字和年龄
<?php
echo "";
?>
2:保存 c:/xampp/htdocs/
3:启动 apache[start]
4:访问 2.php
11:47--11:50
3.2:php 语法 必须掌握
a:程序位置 c:/xampp/htdocs/??
b:创建程序 后缀.php 1.php 2.php
c:代码
<?php
?>
d:声明一个变量
$变量名 = 值;
$name = "tom";
e:访问程序{浏览器}
http://127.0.0.1/1.php 回车
f:一个 php 程序由 html/css/js/php 代码混合写

练习:14:15--14:30
1:创建 3.php,使用 php 向客户端输出 1 个*
2:创建 4.php,使用 php 向客户端输出 50 个*
3:创建 5.php,使用 php 向客户端输出 5 行 10 列个*

4:创建 6.php,使用 php 向客户端输出"九九乘法表"

3.3:php 数据类型

1:值类型/标量类型

string 字符串类 ""

boolean/bool true/false

int/integer

float/double

2:复合类型

object php 面向对象 2005 年之后才出现

array php 索引/关联数组

3:特殊类型

null/NULL

resource 资源:数据库连接，查询结果

4.php 练习数据类型

5.php 练习数组

6.php 综合示例

3.10:使用 php 操作数据库 重点&难点

?数据库乱码问题处理

原因:脚本文件格式不正确

解决：保存脚本文件 utf-8

php 中操作 mysql 数据库函数

1:php 官方最初提供一套连接 mysql 函数:mysql_xxx();

2:php 官方提供增强版 mysql 函数: mysqli_xxx();

使用 php 操作 mysql 服务器步骤

1:创建到 mysql 服务器连接

$conn = mysqli_connect(...);

1:数据库服务器地址 ip 127.0.0.1

2:数据库用户名 root

3:数据库密码

4:选库 dangdang

2:向 mysql 服务器发送 sql 指令,

等待服务器执行

$sql = "....";//insert/delete/update

$result = mysqli_query($conn,$sql);

3:读取 mysql 服务器返回结果

if($result===false)

4:断开与 mysql 服务器连接--可以省略

mysqli_close($conn);

7.php 8.php 9.php

小结:

1:php 语法

```
$age = 10;
$stop = "tom $age";
$run = 'tom'.$age;
```

2:php 数组

```
$arr = [1,2,3,4];
$arr[] = 5;
$arr = ["name"=>"tom","age"=>10];
foreach($arr as $k=>$v){
}
```

3:php 连接 mysql

a:创建连接

```
$conn = mysqli_connect(数据库 ip,数据库用户名,密码,库);
$sql = "SET NAMES UTF8";
mysqli_query($conn,$sql);
```

b:发送 sql

```
$sql = "INSERT INTO dd_book VALUES(null,'d')";
$result = mysqli_query($conn,$sql);
```

c:判断结果

```
if($result === false)
```

作业一:基础稍弱一些同学 综合练习一.txt

三遍

作业二:4:输出九九乘法表

作业三:使用 php 实现新闻的添加和删除功能

1:创建 news.sql 脚本文件

2:创建数据库 ifeng 编码 utf8

3:创建新闻表 t_news

nid 新闻编号

title 新闻标题

count 浏览次数

content 内容

pubtime 发布时间

4:创建 news_add.php 向 t_news 表中添加记录

二条

5:创建 news_del.php 依据新闻 id 删除 t_news 表中记录

DAY3

今天学习的内容

1:复习昨天知识重点
2:作业/综合练习
3:问题{乱码/set names 作用}
4:今日目标
4.1: php mysql 查询{获取记录并且判断}-重点&难点
4.2: php 常用函数
4.3: HTTP 协议(知识点多，杂，乱，前后没有关系)
#------------------
1:复习昨天知识重点
php 语法/php 后端语言 {运行在 apache}
<?php
?>
php:
string $str = 'tom'; $str = "Hello $age";
$i = 10;
$str = $str.$i;
int
double
$arr = [10,20,30,100];
echo $arr[1];
for($i=0;$i<count($arr);$i++){
echo $arr[$i];
}
foreach($arr as $k=>$v){
echo $v;
}
$arr = ["name"=>"tom","age"=>10];
foreach($arr as $k=>$v){
echo $v;
}
1:连接数据库
$conn = mysqli_connect("127.0.0.1","root","","dangdang");
$sql = "SET NAMES utf8";
mysqli_query($conn,$sql);
2:发送 sql 语句
$sql = "INSERT INTO 表名....";
$result = mysqli_query($conn,$sql);
3:获取判断返回结果
if($result===false)
4:关闭连接
mysqli_close($conn);
2:作业/综合练习
2.1:小知识:
$_REQUEST['title'];获取表单中的参数
10:25--10:35
2.2:综合练习
论坛系统中的用户管理 10:30--10:45
1:创建 bbs.sql 创建数据库 bbs
创建表 t_user(uid,uname,upwd,pic,

score,regTime)
2:插入四行记录
3:创建 php user_add.php
获取用户表单中的 uname,upwd,pic,score
执行 insert 输出注册成功或失败
4:创建 user_add_input.html
创建一个表单请用户输入 uname,upwd,pic,score
5:创建 php user_del.php
获取用户表单中的 uid,删除记录
6:创建 user_del_input.html
创建一个表单请用户输入 uid

小结:
1: 发送 sql
解决:在 mysql 命令输入 sql-->copy
如果列是字符串类型一定 "
如果列是整型不用加"
2: 表单获取参数
html: <input type="text" name="uid"
php : $uid = $_REQUEST['uid']
中午时间:bbs
3:问题{乱码/set names 作用}
乱码:?????
解决:
a:顺序:
1:news.sql 按照 utf-8 保存
2:set names utf8;
source 不加分号
b:指令:
1:SET NAMES UTF-8;
2:创建库编码指定错误
CREATE DATABASE bbs CHARSET=UTF8;
C:1.php
SET NAMES UTF8;
INSERT
INSERT
INSERT

4:今日目标
4.1: php mysql 查询{获取记录并且判断}-重点&难点
a:php mysql 查询操作
1:连接数据库
2:发送查询 sql SELECT
SELECT * FROM t_user;
3:获取返回结果 不是 true/false
是:'结果集对象'
? 几条记录满足查询条件 5
? 一共几个字符段 4
4:抓取结果中数据并且转换数组

4.1:抓取一行数据{关联数组}
mysqli_fetch_assoc(结果集对象);
4.2:抓取一行数据{索引数组}
mysqli_fetch_row(结果集对象);
5:循环抓取数据
while()
结束条件:如果没有数据
assoc
row 返回 NULL
1_select.php

bbs/select_user.php
14:52--15:02

1:查询 bbs/t_user 表所有记录并且按表格
显示/用户编号/用户名称/操作
用户编号|用户名称|操作
10 | tom | 删除
20 | jerry | 删除

bbs/
select_user.php 查询
user_del.php 删除
user_add_input.html 添加
user_add.php
更新

小节知识点
1:php 操作数据库
a:INSERT/UPDATE/DELETE
1:连接数据库
2:发送 sql 语句
3:判断$result===false/true
b:SELECT
1:连接数据库
2:发送 sql 语句 SELECT
3:$result 结果集对象[几条/列]
4:抓取
4.1:索引:抓取一行
$row = mysqli_fetch_row($result);
4.2:关联:抓取一行
$row = mysqli_fetch_assoc($result);
5:while()
6:判断抓取结束
$row === NULL 结束

c:bbs:

user_select.php 查询用户表<tr>
user_add.php 添加用户 INSERT
user_del.php 删除用户 DELETE uid
user_update.php 更新用户
a:查询
b:输入新密码
c:update 更新数据

4.2: php 常用函数
4.3: HTTP 协议(知识点多，杂，乱，前后没有关系)

作业一:bbs(读程序/三遍)
a:查询用户列表 **
b:删除用户 **
c:更新用户 ***
d:添加新用户 **

DAY4

今天学习的内容
1:学习总结上周重点
2:今天学习目标
2.1:php 常用函数
2.2:php 工作原理
2.3:http 协议(零碎.没有太多相关性)重点&难点
2.4:阶段小项目(留言板 CRUD)
#-----------------
1:学习总结上周重点
1.1:mysql
数据库软件:
海量存储永久数据
快速查询
SQL:结构化查询语言(操作数据库)
DDL:DROP/CREATE (DATABASE/TABLE)
DML:INSERT/UPDATE/DELETE
DQL:SELECT
DCL:
SET NAMES UTF8; 设置 SQL 编码
SHOW TABLES; 查询当前库表名
SHOW DATABASES; 查询当前 MYSQL 库名
..
1.2:php
后台语言:运行在(服务器 apache)
<?php
?>
$age = 10;
$str = 'hello php';

$str = "hello $age php";
php mysql
a:INSERT/UPDATE/DELETE
1:连接数据库
$conn = mysqli_connect("127.0.0.1",'root','','dangdang');
2:发送 sql
$sql = "INSERT/UPDATE/DELETE";
$result = mysqli_query($conn,$sql);
3:if($result===true){
执行成功
}
b:SELECT
1:连接数据库
$conn = mysqli_connect("127.0.0.1","root","","dangdang");
2:发送 sql
$sql = "SELECT ...";
$result = mysqli_query($conn,$sql);
3:返回结果集对象
4:抓取
$row = mysqli_fetch_row($result); 索引
$row = mysqli_fetch_assoc($result);关联

2:今天学习目标
2.1:php 常用函数
1:#die($str); 终止当前 php 文件执行,
并且向客户输出一个终止原因说明.
2:#@ 压制住当前行代码警告消息.
3:#time() 返回当前系统时间,以秒为单位整数
4:#$id = mysqli_insert_id($conn);
返回连接上刚刚执行 insert 语句产生 id 值
CREATE TABLE t_user(
id INT PRIMARY KEY AUTO_INCREMENT,
);
INSERT INTO t_user VALUES(null,);
5:#$count = mysqli_affected_rows($conn)
返回刚刚连接上执行增删改语句影响行数
6:$rows = mysql_fetch_all($result,MYSQLI_ASSOC);
从结果集对象中抓取所有记录,
返回二维数组;

练习 1:(添加)
bbs/t_user(uid/uname/upwd/pic/score/regTime);
1:创建 add_user_02.php 功能新建用户
1:获取参数
:如何参数值不存在停止 php 执行,
:并且输出 用户名不存在
:不允许显示 php 错误消息
2:连接数据库并且添加 t_user 表中

:添加成功后显示当前用户编号值
:添国成功后显示当前 SQL 影响几行记录
2:表单 add_user_input_02.html
表单(用户名/密码/图片/分数)

2.2:php 工作原理(电话餐厅)
电话餐厅.jpg select_user_02.php
1:浏览器发送请求 apache (打电话)
地址栏输入地址回车(请求)
2:apache 将请求转发 php 引擎(经理厨师)
3:php 引擎处理 php 代码并且(!!!)
连接数据库查询结果生成(html)->(菜)
4:php 将生成 html->返回 apache
5:apache 将 html 发送客户端(快递小哥)
6:客户端浏览器显示 html

2.3:http 协议(零碎.没有太多相关性)重点&难点
2.3.1:学习 http 协议目标
1:调试 AJAX 应用"看不见摸不着"的错误
2:进行 web 访问优化---高阶面试题

2.3.1.1:URL
统一资源定位符:
互联网任何资源都有一个 URL 才能被访问
http://www.baidu.com 网站
https://www.baidu.com/img/bd_logo1.png 图片
http://127.0.0.1/01.php

<scheme>://<user>:<pwd>@<host>:
<port>/<path>;<params>?<query>

a:scheme:方案 指定以哪种协议从服务器获取指定资源
常见方案:http/https/ftp/mailto/file/telnet....
http:获取网络资源{明文}
https:获取网络资源{加密}
http://www.ccb.com/cn 建行
https://ibsbjstar.ccb.com.cn 建行登录

b:host 主机名:资源在服务器 ip 地址或者域名
http://127.0.0.1 ip 地址
http://www.baidu.com 域名(DNS 域名->ip)
http://tmooc.cn 域名

c:port 端口号*****(面试题)
每一项网络服务在服务器都对应一个端口号
ftp 21 文件上传下载
ssh 22 安全远程登录

telnet 23 远程登录
smtp 25 邮件传输发送
dns 53 域名解析
http 80 超文本传输***(apache)
pop3 110 邮件接收
https 443 加密传输

d:path
http://127.0.0.1/ajaxday03/01.php

e:?<query>
http://127.0.0.1/ajaxday03/01.php?id=10&age=10
$id = $_REQUEST['id'];
$age = $_REQUEST['age'];

#分清一个概念:URL/URN/URI(了解)
URL:统一资源定位符
URN:统一资源命名符
URI:统一资源标示符

URI 图.jpg
URI==URL+URN
URL:....
<img src="http://tmoo.cn/logo.jpg">....
URN:
..
..

2.3.2:http 协议概述
作用:传输网页
协议标准:生活标准(手机: 3g 4g 5g)
USB(2.0 3.0)
国际互联网任务组(IETF)制定 http 协议标准
1991: http/0.9 有严重缺陷
1996: http/1.0 正式版本
1999: http/1.1 当前主流版本(***)

面试题:
http/1.1 比 http/1.0 改进哪些地方?
http 协议工作原理方式:请求和响应
1:客户端浏览器发送请求 (google->apache)
2:服务器响应请求并且返回数据
(apache->google)
解答问题:
1:支持虚拟主机技术，在一个 web 服务器上
同时并存多个不同域名的网站
个人博客:500w(128 CPU 1T 2000 块)

3 网页 index.html photo.html review.html
20/年
d:/tmooc www.tmooc.cn 50w
e:/tts www.tts.cn 50w
...10
http: Host:tmooc.cn 虚拟主机域名
2:支持持久连接技术
不支持持久边接技术情况
(每次客户端与服务器数据传输)
固定流程 101(300 握 400 挥)
三次握手 3->1-->4
四次握手
http: keep-alive (3)--101--(4)
3:支持代理连接
Proxy:xxx

2.3.3: http 协议规定两种消息格式:
请求(Request) --客户端浏览器发送 web 服务器
响应(Response)--web 服务器发送客户端浏览器

Message:消息/报文,是在 http 客户端和服务器传递数据块
http 协议规定,消息必须符合特定格式才能版此理解..

消息内容 "思维导图"

示例:请求主体类型
15:57--16:02
input_03.html 表单 GET/POST
用户名/密码
input_03.php 接收表单

2.4:阶段小项目(留言板 CRUD)
功能:
1:发表留言:
用户在一个表单中提交:名称，电话,
留言内容，点击提交按钮，即可发表.
2:浏览所有留言:
可以查看留言，编号，名称，内容，发布时间
3:删除某个留言:
点击某个留言右上角 X 删除该留言

创建 SQL-> 创建 php-> 创建 html/js

需要文件
*0:创建文件夹 msg 16:40--16:50
*1:创建 sql 文件:tarena.sql
1:创建库 msg utf8
2:创建留言表 t_msg

留言编号:留言发布人名称:联系电话
发布时间：留言内容
3:添加三条留言
2:创建 php:发表留言
init.php
msg_add.php {接收参数:保存数据库 INSERT}
3:创建 html:创建表
msg_add_input.html 表单

#ANSI:windows--->GBK 编码

作业:
4:留言板:浏览所有留言(ul>li)
msg_select.php
[X] -》确定您是否要删除该留言?
-->是
5:留言板:删除留言
msg_delete.php
5.1:获取 id
5.2:发送 sql
DELETE FROM t_msg WHERE id=?
5.3:删除成功自动跳转
msg_select.php
location.href = "msg_select.php";

DAY5

今天学习的内容
1:复习昨天知识重点
2:作业:留言二个功能
3:http 协议-(响应)
4:今天目标
4.1:综合示例"搜狐新闻"
4.2:ajax 重点&难点
#----------
1:复习昨天知识重点
php 常用函数
die("提示信息"); 程序行首@
mysqli_insert_id($conn);
mysqli_fetch_all($result,MYSQLI_ASSOC);
http 协议:
HTTP 作用:传输网页
工作机制:请求和响应
请求几种:
1:地址栏输入地址按回车
http://127.0.0.1/1.php

2:表单 <form action="1.php">
<input type="submit"
3:超链接 ...
2:作业:留言二个功能
3:http 协议-(响应)
小结:
http:作用传输网页
http:工作方式 请求与响应
请求消息:
请求方式:{GET/POST/PUT/DELETE/HEAD/CONNECT/TRACE/OPTIONS}
响应消息:
响应状态码:
100-199 提示消息
200-299 响应成功 200
300-399 重定向 304
400-499 客户端请求错误 404
500-599 服务器错误 500
响应主体类型:
text/plain
text/html
text/css
application/javascript
application/json

1: 面试题:GET/POST 区别?
a:语义
GET:客户端获取服务器上资源
POST:客户端将数据提交服务器
b:安全级别
GET:不安全
POST:不安全 {https}
c:数据长度
GET:通过浏览器地址栏 请求请起启 1KB
汉字 20-30
POST:通过 http 响应主体 长度没限制
d:编码
GET:不会自动编码->可能出现中文乱码
POST:自动编码---->不会中文乱码
e:如何发起
GET:
1:浏览器地址栏输入地址回车
2:标签 href <a> 点击发 GET
src <img> 自动发 GET
css like 自动发 GET
script src 自动发 GET
3:js 自动跳转
location.href = "1.html";
4:表单

<form method="get" action="">
<form action="1.ph">
5:ajax GET
POST:
1:表单
<form action="" method="post">
2:ajax post

2：面试题:如何使用 http 协议相关知识进行 web 优化
提示:web 访问可以很多方面考虑:
:优化数据库
:优化 php
:优化 web 服务器(apache/nginx)
:网速
:传输数据
:浏览器解析速度(html/css/js)
下面仅从 http 请求和响应角度考虑
1:域名解析
尽可能减少域名解析次数--减少跨站外
资源引用.
DNS(www.tmmoc.cn->120.132.68.230)
tmooc.com
http://www.tmooc.cn/index.html
index.html
<img src="www.tmooc.cn/1.jpg">
<script src="www.tmooc.cn/1.js"></script>
2:创建连接
努力减少连接创建次数-Connection:keep-alive
启用持久连接
3:减少发送请求次数
尽量减少请求次数--合理进行资源合并,
合理使用缓存.
4:等待响应时间
提交服务器运行速度-提高数据运算及查询速度
www.csdn.net 程序员 300w windows 30 台
www.javaeye.com 150w linux 2 台
5:接收响应
尽可能减少响应数据长度--删除空白字符，启压缩
http://tool.oschina.net/jscompress/
4:今天目标
4.1:ajax 重点&难点 9
a:快速入门
ajax:异步 javascript and xml
google 推出技术-->酷
生活场景:ajax 微信餐厅
最大不同:(发送请求和接收数据都由 js 完成)

b:原理细节知识 14:50---15:00
*功能:添加新用户 msg/t_user(uname,upwd)

*1:创建 add_user.php
a:获取参数 uname,upwd
b:创建 sql insert 发送 sql
c:输出结果添加成功或失败
*2:创建 add_user_input.html
a:创建表单(uname,upwd)
? b:在表单<input type="button" value="新建用户"/>
c:为 button 按钮绑定点击事件
c.1:获取用户名和密码
d:通过 ajax 发送请求并且接收服务器
输出一句话"添加成功""添加失败"
以下 ajax 代码固定 js
1:创建 ajax 对象
var xhr = new XMLHttpRequest();
2:绑定事件
xhr.onreadystatechange = function(){}
3:打开连接(连接 php 程序)
xhr.open('GET','add_user.php',true);
4:发送请求
xhr.send(null);

重点:ajax 请求并且接收响应 4 步
1:创建 ajax 对象
var xhr = new XMLHttpRequest();
2:绑定事件:监听 xhr 对象状态
xhr.onreadystatechange = function(){
}
3:连接 web 服务器 php 程序
xhr.open(请求方式,请求地址,是否异步);
xhr.open('GET','add_user.php?',true);
4:发送请求消息
xhr.send(null);

练习:删除用户 15:40--15:55
1:del_user.php {获取用户名/创建 sql/成功}
2:del_user_input.html {表单/输入用户名}
ajax 将用户发送 php 删除

4.2:ajax 原理
a:2002 年由 Google 搜索引擎，提示建议
AJAX=HTML/CSS/JS/DOM/XML/HTTP
b:ajax 作用:
实现在'无刷新''无提交''无跳转'
的情况下完成页面局部更新.
c:ajax 应用场景
常见场合:聊天室，在线走势图,搜索建议..

d:ajax 异步 javascript and XML 异步:
1:ajax 异步请求:XHR
2: 同步请求:
1.1:地址栏输入地址回车
1.2:表单 submit
1.3:<a href="1.php">
1.4:<img src="1.jpg">
1.5:location.href = '1.php'

4.3:ajax 常用对象和属性事件方法
1:xhr 对象:作用，向 web 服务器发送请求，
并接收返回响应消息.

```
var xhr = new XMLHttpRequest();
```

#注意:产品兼容性
#老 IE IE8- 不支持 xhr

```
new ActiveXObject("Microsoft.XMLHTTP");
```

#兼容性写法

```
var xhr = null;
if(window.XMLHttpRequest){
xhr = new XMLHttpRequest();
}else{
xhr = new ActiveXObject("Microsoft.XMLHTTP");
}
alert(xhr);
```

2:xhr 对象常用属性*****
2.1:readyState xhr 对象当前状态
不能手工赋值，可以取值
其值会随着(请求-响应)过程进行
自动改变
0 UNSENT 请求消息沿未发送
1 OPENED XHR 已经打开 web 服务器连接
2 HEADERS_RECEVIED xhr 已经接收服务器响应消息头部
3 LOADING XHR 正在加载响应消息主体
4 DONE XHR 接收完成响应消息主体

2.2:xhr.readyState 状态改变
0 UNSEND
xhr.open();
1 OPENED
xhr.send();
2 HREADERS_RECEIVED
自然(不能控制)
3 LOADING
自然(不能控制)
4 DONE

2.3:status
2.4:statusText
2.5:responseText

4.4:综合示例"搜狐新闻"

作业:
功能:新闻添加/新闻删除/新闻查询
0:创建文件夹 sohu
1:创建 sohu.sql
创建库 sohu
t_user(id,uname,upwd)
t_news(id,title,ctime,uid,viewcount);
viewcount BIGINT 浏览器次数
2:新闻添加
add_news.php
获取参数并且数据添加数据库 t_news
add_news_input.html
创建表单,标题,用户编号
ajax
3:删除新闻
del_news.php
获取参数并且删除指定记录 标题
del_news_input.html
创建表单，标题
ajax
4:新闻查询
方案一:html/php select_news.php
查询所有新闻 标题创建时间
方案二:ajax
select_news.php
echo "
世界杯:2008-10-10
";
echo "
世界杯:2012-10-10
";
select_news_input.html
<ul id="menu">

js
menu.innerHTML = xhr.responseText;

DAY6

今天学习的内容
1:复习总结昨天知识重点
2:解决问题
2.1:多表查询
2.2:删除不成功
mysqli_affected_rows($conn);

2.3:php 工作原理图
4:作业
5:ajax 原理与 ajax 接收不同类型数据
5.1 text/html
5.2 javascript
5.3 xml ---- 很少(代码复杂)
5.4 json ---- 重点&难点
#----------
1:复习总结昨天知识重点
http 协议:
请求消息:请求 8 种方式
响应消息:响应状态码
响应数据类型
ajax:
2:解决问题
2.1:多表查询(今天/项目)
2.2:删除不成功(用户名)
?问题:删除用户-》用户不存在
?结果奇怪:删除成功
$num = mysqli_affected_rows($conn);
刚才执行 DELETE/INSERT/UPDATE 影响几行记录
2.3:php 工作原理图
4:作业
SOHU 新闻.txt 11:30---11:35
背下:多表查询语法
1:几张表，表名 2 t_user/t_news
2:表之间关系 = != < > <= ..
t_user.id=t_news.uid
3:表别名
t_user u/t_news n

```
SELECT n.title,n.ctime,u.uname
FROM t_user u,t_news n
WHERE u.id = n.uid;
```

5:ajax 原理与 ajax 接收不同类型数据
a:核心 XMLHttpRequest
-创建对象

```
var xhr = null;
if(window.XMLHttpRequest){
xhr = new XMLHttpRequest();
}else{
xhr = new ActiveXObject("Microsoft.XMLHTTP");
}
```

b:xhr 属性、方法、事件
b.1:xhr.readyState 表示 xhr 状态
0 未发送 UNSENT
1 打开连接 OPENED
2 已经接收服务器返回头信息

HEADERS_RECEIVED
3 接收服务器数据
LOADING
4 接收完成
DONE
B.2:status 表示服务器返回状态码
===200
B.3:responseText 表示服务器响应文本
B.4:responseXML 表示服务器响应 XML 文本
方法:
B.5:open(method,url.isAsyn)
表示:打开到服务器连接
method:请求方式 GET POST
url: 请求 url 地址(程序地址)
isAsyn:请求方式是异步 true
同步 false
B.6:send(data)
表示:把请求消息发送 web 服务器
data:请求消息主体内容
GET-->内容为 null
send(null);
POST-->请求数据放在里面
send('id=10&name=tom&age=19');
事件:onreadystatechange
xhr.readyState 每次改变时候触发
事件 0 1 2 3 4

5:ajax 接收很多数据类型
a:php echo "添加成功";
b:php echo "
";

综合示例‘当当网’:
14:25----14:40
mysql 开发工具(oracle/phpadmin)
*0:创建文件夹/dangdang/dangdang.sql
*1:创建库 dangdang/t_book
(id,name,pic,
price,pubDate)
*2:添加 10 行记录
*3:添加图书
* book_add.php 获取参数添加数据返回
book_add_input.html 创建表单
#AJAX POST 标准语法
1:创建 ajax 对象 xhr
2:绑定事件 xhr.onreadystatechange
*3:打开连接
xhr.open('POST','book_add.php',true);
*3.1:修改请求头信息(将参数编码)

```
xhr.setRequestHeader('Content-Type',
'application/x-www-form-urlencoded');
*4:xhr.send(`id=10&name=tom&age=19`);
```

如果 ajax 输出

1:创建 php 地址栏输入回车

http://127.0.0.1/ajaxday06/dangdang/book_add.php?name=1&pic=2.jpg&price=100&pubDate=2010-10-10

2:创建 html/js

```
console.log("1"+变量);
```

3:F12/network

#当当网功能二:搜索图书 15:40--15:55

功能:请用户输入图书名称

ajax 获取图书名称发送 post 请求 php

php 输出包含用户输入图名图书列表

#book_search.php

```
SELECT * FROM t_book WHERE name LIKE '%j%'
```

#book_search.html j

例:

1:用户输入 js

2:显示

<js 大全 1>

<js 大全 2>

5.0 text/plain 纯文本

php 第一行

```
header("Content-Type:text/plain");
```

html->ajax

xhr.responseText

5.1 text/html (默认)

php 第一行

```
header("Content-Type:text/html;charset=utf-8");
```

html->ajax

xhr.responseText;

5.2 javascript

php 第一行

```
header("Content-Type:application/javascript;charset=utf-8");
echo "var msg='hello';alert(msg)";
```

html->ajax

```
eval(xhr.responseText);
```

示例:国际化

当前浏览器 zh-CN 你好

en-US hello

a:编写 php i18n.php

读取客户端请求头部信息

Accept-Language 截最后二个字符
CN 你好
US hello
b:i18n.html
ajax 发送请求-->获取服务器返回信息
alert();
5.3 xml ---- 很少(代码复杂)
5.4 json ---- 重点&难点

作业:异步员工查询(ajax)
功能要求:
1:编写 tarena.sql
创建库 tarena/创建表 t_emp
(eid/ename/phone/salary/deptId 部门编号)
创建表 t_dept
(did 部门编号/dname 部门名称)
2:select_emp.php
2.1 选择不同的部门,
查询该部门下员工列表(ajax)
示例:
研发部 10
市场部 20 -->
运营部 30

员工编号 员工名称
1 tom [删除]
2 jerry [删除]
2.2:点击删除按钮->删除数据库表中记录
并且删除网页中 tr(ajax)

DAY7

今天学习的内容
1:复习总结 AJAX
2:作业
3:今天知识
3.1:ajax 处理不同数据类型
a:json
b:xml
#----------
1:复习总结 AJAX
a:工作流程
b:常用对象，属性，方法，事件

2:作业
解决方案:location.href
解决方案:ajax
3:今天知识
3.1:ajax 处理不同数据类型
a:json--->重点&难点
数据格式标准:软件项目
后台工程师---->前端工程师
用户列表(li 1 li name)A-->li
用户列表(1:tom:100)B-->li
用户列表(tr td)C-->li

a.1:标准语法 json 要求
1:一个 JSON 字符串有且只有一个根,
可以是{}表示一个对象
[]表示一个数组

员工信息 {"name":"tom","age":19}
一组整型 [10,20,90,100]
一组员工信息
[
{"name":"jerry","age":21},
{"name":"james","age":22},
{"name":"gogo","age":19}
]
错误 json
{"name":"tom1"}
{"name":"tom2"}
{"name":"tom3"}
2:JSON 中可以表示，数字，bool,null,字符串
注意:字符串必须用双引号
3:数组中可以包含多个值，使用逗号分隔
4:对象中可以包含多个键值，使用逗号分隔
不同值，键和值之间用分号分隔 ,
键必须是双引号.

如何处理 JSON 数据:
1:服务器端 php
1.1:header("Content-Type:application/json;charset=utf-8");
1.2:$str = json_encode($arr);
1.3:echo $str;
2:javascript 接收
var obj = JSON.parse(xhr.responseText);
obj-->js 数组 obj-->js 对象

练习一:
功能:
查询 t_dept 表中部门编号为 10 部门信息
转换 json 字符串发送客户端

客户端 ajax->接收 json->

```
<ul>
  <li>部门编号:10</li>
    <li>部门名称:研发部</li>
</ul>
```

tarena/t_dept

dept_json_01.php(关联一行)

dept_json_01.html

练习二:(关联多行)

dept_json_02.php

dept_json_02.html

#JSON 字符串格式概述

XML 是字符串数据格式，用于描述数据，
稍微有点麻烦--重量级数据格式.
10 字节->XML-->50 字节
JSON 是字符串数据格式，用于描述数据，
更加简单--轻量级数据格式.
10 字节->JSON->22 字节

JSON(JavaScript Object Natation)
是一种轻量级数据交换格式，
易于编写，同时易于程序解析生成
.语法基于 JS 语言.但是目前被各种语句
所支持.成为一种"异构系统交互数据
标准格式";
JAVA--- {JSON} html/js/css
C#---- {JSON} html/js/css
php---- {JSON} html/js/css

味多美蛋糕

功能 1:蛋糕类别列表(巧克力/芝士/木斯)
功能 2:当用户选择不同类别->
显示该类别下所有蛋糕列表
[ ] 价格 简介

操作流程:16:20--16:30

1:创建目录/库 weiduomei/表
2:dm_cake_type 蛋糕类别
did {类别编号}(巧克力/芝士/木斯)
dname{类别名称}
cakecount{该类别下蛋糕数量}
pic {类别图片}
3:dm_cake 蛋糕
cid/cname/pic/price/
typeid 类别 id

程序:

select.html 16:57--17:00

巧克力蛋糕
芝士蛋糕
慕斯蛋糕

<table>
...
</table>
type_select.php 完成查询类别列表
cake_list_select.php 完成查询{did=1}蛋糕列表
功能描述
1:页面加载成功,自动发送 ajax 请求
type_select.php-->json
显示所有类别蛋糕信息
2:点击某一种类别蛋糕图片发送 ajax 请求
cake_list_select.php 1
b:xml

作业:
:点击某一种类别蛋糕图片发送 ajax 请求
cake_list_select.php 1
:select.html ajax

DAY8

今天学习内容
1:作业
2:今天目标
2.1:ajax 处理 xml 数据格式
2.2:jquery 常用 ajax 函数
2.3:跨域 ajax
3:开发项目(mysql/php/ajax)
3.1:京东首页(html/css)
用户登录
产品列表(分页)
3.2:购物车
创建购物车
购物车修改 +-
#------------------
1:作业
2:今天目标
2.1:ajax 处理 xml 数据格式 -- 了解

xml:
html:超文本标记语言，所有标签都是预定义好的,
用于描述一个网页结构.
xml:可扩展的标签语言，所有的标签都是自定义的
用于描述一段数据--尤其是批量复合数据.

xml 语法要求:
1:xml 文档声明
<?xml version="1.0" encoding="utf-8"?>
2:整篇 xml 字符串有且只能一个根元素
<booklist>
<book></book>
<book></book>
</booklist>
3:标签有开始必须有结束
<book></book>
4:标签可能嵌套不能交叉
<book><name></name></book> ok
<book><name><id></name></id></book>
5:标签所有属性'值'必须用双引号括起来
<book id="1" name="tom"></book>
总结:xml 和 html 用途不同，xml 语法严格
综合示例:
服务器 php: book_xml_01.php
1:header("Content-Type:application/xml;charset=utf-8");
2:echo ""xml,,,,
客户 js: book_xml_01.html
1:ajax 请求接收数据;
2:var xmlDoc = xhr.responseXML;---
3:获取所有 book 标签
var book = xmlDoc.querySelectorAll("book");
4:获取 book 下 id/name
var name = book.querySelector("name");
5:获取书名
name.innerHTML;

2.2:jquery 常用 ajax 函数
a:$.get(url,callback);--简单
作用含义:
发起一个 ajax 的 GET 请求,如果服务器返回成功
响应消息，调用 callback 函数,
在方法中处理响应的数据
callback function(data){}
练习:weiduomei
select_jquery.html
$.get("cake_type.php",function(data){
//data json-->JSON.parse(data);
});
b:$.post(url,data,callback);
作用:
发起一个 ajax POST 请求，并在请求主体中
提交请求数据，如果服务器返回成功响应
消息，调用 callback，在 cllback 方法中

处理响应数据.

使用方法三种:

$.post(url,data,callback);

$.post(url,'id=1&name=tom',callback);

$.post(url,{id:1,name:tom},callback);

c:$.getJSON(url,callback);

d:$.load(url);

e:$.getScript(url,callback);

f:$.ajax({})--功能最全最强

万能 AJAX 封装函数，提供非常多的可选项,

可以处理各种情形，前面函数都是它的

简化版.

```
$.ajax({
        type:'GET'          //请求方式 post/pub/delete/head
        url:"x.php"         //请求地址
        data:'k=v&k1=v1'//请求服务器数据
        beforeSend:fn       //请求消息发送之前调用 fn
        success:fn          //响应完成并且成功调用 fn
        error:fn            //响应完成但有问题调用 fn
    complete:fn         //响应完成回调(无论成败)fn
    });
```

练习:味多美类别

1:添加新类别

add_cake_type.php 11:53--11:58

1:获取参数 dname/pic

2:添加成功 1

add_cake_type_input.html

1:表单 保存

2:发送 ajax 请求

##万寿路线路检修: 1:30 上课

##中午时间分析功能、创建对应表

了解:

c:$.getJSON(url,callback);

发起异步请求 GET，要求服务器返回 JSON

数据格式,会自 JSON.parse(data);

e:$.getScript(url,callback);

发起异步请求 GET, 要求服务器返回 Javascript

数据格式，会自动 eval(xhr.responseText);

#ajax 注意

对于异步请求成功后创建 DOM 元素,

不能进行直接事件绑定! click()/bind().

因为在执行此事件绑定时，这些元素在 DOM

还不存在。

必须将相关事件委托给 DOM 树上已经存在父元素!

2.3:跨域 ajax

3:开发项目(mysql/php/ajax)--重点&难点
3.1:京东首页(html/css)
用户登录
产品列表(分页)
3.2:购物车
创建购物车
购物车修改 +-

DAY9

今天学习的内容
01:上周京东项目重点小结
02:今天目标
上午: 01:添加购物车
02:显示购物车列表
03:删除购物车选项
04:修改购物车 +-
下午:
05:跨域
06:cookie
07:分页
/-----------------
01:上周京东项目重点回顾
02:今天目标
上午: 01:添加购物车
02:显示购物车列表
03:删除购物车选项 中午练习
04:修改购物车 +- 作业
开发项目_006.txt
下午:
06:cookie
cookie:小甜饼
cookie:保存客户端浏览器中一个纯文本文件
cookie 作用:
保存:[安全性要求不高]文字或数字数据
登录密码?
通用方案:
1:用户昵称/用户名/用户编号
2:浏览过商品
3:大型网站-->用户购物车中数据也保存 cookie

语法:!!!! 重点.
1:cookie_add.html 保存 cookie

document.cookie = '名=值';
document.cookie = 'uid=10';
2:cookie_read.html 读取 cookie
var str = document.cookie;
#
var str = "name=tom;age=10;sex=F";
var arrStr = str.split(";");
["name=tom","age=10","sex=F"]
name = arrStr[0].split("=")[0];
tom = arrStr[0].split("=")[1];

07:分页
mysql:
SELECT * FROM jd_product LIMIT ?,?
? 启动记录行 0
? 查询几行记录

1:SELECT * FROM jd_product LIMIT 0,8;
2:SELECT * FROM jd_product LIMIT 8,8;
3:SELECT * FROM jd_product LIMIT 16,8;
页数 启始记录
1---->0 (页数-1)*8; (1-1)*8=0
2---->8 (2-1)*8=8
3---->16 (3-1)*8=16

修改程序
1:php

05:跨域

开发项目 1

开发项目
1:开发流程{工作职位/负责}
小../中小/大(100>)
中小
a:需求分析-->技术经理--(美工)
文档(需求说明书)
b:美工--》图片-->切图
小图片
c:前端-->html/css/js 静态网页
d:后端-->php/mysql 动态网页
e:运维-->linux/mysql/redis/ip/www/服务器
动态网页部署服务器

f:测试-->功能
g:上线
i:调试->加新功->升级...

2:分析功能{功能模块}
功能模块一:用户登录
功能模块二:产品列表显示->分页
功能模块三:点击购物商品
功能模块四:浏览购物车{删除/+/-}

功能模块一:用户登录
1.1
依据功能创建库、表、记录
创建库:jd
创建表:登录表
t_login:
编号: id INT
用户名: uname VARCHAR(20)
密 码: upwd VARCHAR(32)
添加三条记录
1.2:创建 php
jd/data/login_do.php
a:获取参数 uname/upwd
b:查询数据库
SELECT * FROM t_login
WHERE uname='$uname' AND upwd='$upwd'
c:如果用户输入{uname/upwd}
d:-1 -2 -3 1

1.3:创建 js {分析 html/css}
a:productlist.html
b:jd/js/productlist.js
c:js
c.1:登录按钮 "提交登录信息" 绑定点击事件
productlist.html
 408 模态窗口
<p class="alert"> 412 出错信息
id="uname" 416
id="upwd" 417
id="bt-login" 418 按钮
c.2:获取用户名密码
c.3:发送 ajax 请求 login_do.php
c.4:-1 -2 -3 1
c.5:登录成功->隐藏登录框和半透明背景
c.6:登录失败->提示框{用户名或密码有误}
#----------

开发项目 2

开发项目
1:开发流程{工作职位/负责}
小../中小/大(100>)
中小
a:需求分析-->技术经理--(美工)
文档(需求说明书)
b:美工--》图片-->切图
小图片
c:前端-->html/css/js 静态网页
d:后端-->php/mysql 动态网页
e:运维-->linux/mysql/redis/ip/www/服务器
动态网页部署服务器
f:测试-->功能
g:上线
i:调试->加新功->升级...

2:分析功能{功能模块}
功能模块一:用户登录
功能模块二:产品列表显示->分页
功能模块三:点击购物商品
功能模块四:浏览购物车{删除/+/-}

功能模块二:产品列表显示
依据功能创建库、表、记录
创建库:jd
创建表:产品表
jd_product:
pid 产品编号
pname 产品名称
price 产品价格
pic 产品图片
添加多条记录
1.2:创建 php
jd/data/product_list.php
b:查询数据库
SELECT * FROM jd_product;
c:返回 json 格式字符串
1.3:创建{分析 html/css}
a:productlist.html
<section id="plist"> 32

 32--42

b:jd/js/productlist.js
c:js
c.1:页面加载成功显示产品列表

作业:下课
分页处理: Mysql limit 记录位置,几行;
记录位置从 0
SELECT * FROM jd_product LIMIT 0,8; 一页
SELECT * FROM jd_product LIMIT 8,8; 二页
SELECT * FROM jd_product LIMIT 16,8; 三页

#--------------------------

添加页头页尾
header.php
footer.php
小知识:jquery 函数
$("父元素").load("程序 1");
load()向程序发送请求(ajax)
返回的内容->会保存在父元素中
相当于 innerHTML;

开发项目 3

开发项目
1:开发流程{工作职位/负责}
小../中小/大(100>)
中小
a:需求分析-->技术经理--(美工)
文档(需求说明书)
b:美工--》图片-->切图
小图片
c:前端-->html/css/js 静态网页
d:后端-->php/mysql 动态网页
e:运维-->linux/mysql/redis/ip/www/服务器
动态网页部署服务器
f:测试-->功能
g:上线
i:调试->加新功->升级...

2:分析功能{功能模块}
功能模块一:用户登录
功能模块二:产品列表显示->分页
功能模块三:点击“加入购物车”
功能模块四:浏览购物车{删除/+/-}

功能模块四:浏览购物车{删除/+/-}

依据功能创建库、表、记录
创建库:jd
创建表:购物车表
jd_cart:

购物车编号	id	INT
用户编号	uid	INT
产品编号	productid	INT
购买数量	count	INT

1.2:创建 php
jd/data/cart_list.php {购物车的列表}
a:查询数据库
多表查询
1:购物车表 jd_cart{id/uid/productid/count}
2:产品表 jd_product{pid/pname/price/pic}
查询:mysql 标准语法
查结果{购物车 id/产品名称/产品图片/产品价格}
{购物车中该产品购买数量}
a:几张表，表名 jd_cart/jd_product
b:每一张表起一个别名 c/p
c:条件 = > < != c.productid=p.pid
11:10--11:20

```
SELECT c.id,p.pname,p.pic,p.price,c.count
FROM   jd_cart c,jd_product p
WHERE c.productid=p.pid
AND    c.uid = 10;
```

1.3:创建{分析 html/css}
a:shoppingcart.html

```
<table id="cart">        15          line
    <tr></tr>                30--43   line
```

b:jd/js/shoppingcart.js

开发项目 4

开发项目
1:开发流程{工作职位/负责}
小../中小/大(100>)
中小
a:需求分析-->技术经理--(美工)
文档(需求说明书)
b:美工--》图片-->切图
小图片
c:前端-->html/css/js 静态网页

d:后端-->php/mysql 动态网页
e:运维-->linux/mysql/redis/ip/www/服务器
动态网页部署服务器
f:测试-->功能
g:上线
i:调试->加新功->升级...

2:分析功能{功能模块}
功能模块一:用户登录
功能模块二:产品列表显示->分页
功能模块三:点击“加入购物车”
功能模块四:浏览购物车
功能模块五:删除浏览购物车中选项
功能模块六:修改购物车中数量 {+/-}

功能模块六:删除浏览购物车中选项
依据功能创建库、表、记录
创建库:jd
创建表:购物车表
jd_cart:

购物车编号	id	INT
用户编号	uid	INT
产品编号	productid	INT
购买数量	count	INT

1.2:创建 php
jd/data/cart_update_add.php //+
jd/data/cart_update_sub.php //-
header("content-type:application/json;...");
a:获取删除购物项 id 值
'{"code":-1,"msg":"修改编号不能为空"}'
b:连接数据
UPDATE jd_cart SET count=count+1
WHERE id = $id;
c:发送 sql
'{"code":1,"msg":"修改成功"}'

1.3:创建{分析 html/css}
a:shoppingcart.html
<td>1199.00</td>
<td>
<button>-</button>
<input type="text" value="8">
<button>+</button>
</td>
<td>￥9592</td>
b:jd/js/shoppingcart.js

开发项目 5

开发项目
1:开发流程{工作职位/负责}
小../中小/大(100>)
中小
a:需求分析-->技术经理--(美工)
文档(需求说明书)
b:美工--》图片-->切图
小图片
c:前端-->html/css/js 静态网页
d:后端-->php/mysql 动态网页
e:运维-->linux/mysql/redis/ip/www/服务器
动态网页部署服务器
f:测试-->功能
g:上线
i:调试->加新功->升级...

2:分析功能{功能模块}
功能模块一:用户登录
功能模块二:产品列表显示->分页
功能模块三:点击“加入购物车”
功能模块四:浏览购物车
功能模块五:删除浏览购物车中选项
功能模块六:修改购物车中数量 {+/-}

功能模块六:删除浏览购物车中选项
依据功能创建库、表、记录
创建库:jd
创建表:购物车表
jd_cart:

购物车编号	id	INT
用户编号	uid	INT
产品编号	productid	INT
购买数量	count	INT

1.2:创建 php
jd/data/cart_update_add.php //+
jd/data/cart_update_sub.php //-
header("content-type:application/json;...");
a:获取删除购物项 id 值
'{"code":-1,"msg":"修改编号不能为空"}'
b:连接数据
UPDATE jd_cart SET count=count+1
WHERE id = $id;
c:发送 sql
'{"code":1,"msg":"修改成功"}'

1.3:创建{分析 html/css}
a:shoppingcart.html
<td>1199.00</td>
<td>
<button>-</button>
<input type="text" value="8">
<button>+</button>
</td>
<td>￥9592</td>
b:jd/js/shoppingcart.js

开发项目 6

开发项目
1:开发流程{工作职位/负责}
小../中小/大(100>)
中小
a:需求分析-->技术经理--(美工)
文档(需求说明书)
b:美工--》图片-->切图
小图片
c:前端-->html/css/js 静态网页
d:后端-->php/mysql 动态网页
e:运维-->linux/mysql/redis/ip/www/服务器
动态网页部署服务器
f:测试-->功能
g:上线
i:调试->加新功->升级...

2:分析功能{功能模块}
功能模块一:用户登录
功能模块二:产品列表显示->分页
功能模块三:点击“加入购物车”
功能模块四:浏览购物车
功能模块五:删除浏览购物车中选项
功能模块六:修改购物车中数量 {+/-}
功能模块七:用户登录

功能模块七:用户登录
依据功能创建库、表、记录
创建库:jd
创建表:购物车表
t_login:

用户编号	id	INT
用户编号	uname	INT
用户密码	upwd	INT

功能分析:
当用户登录成功后.
a:获取用户 id 和登录名称保存 cookie
b:查看购物车-->uid
1.2:创建 php
jd/data/login_do.php
升级
jd/data/login_do_02.php

1.3:创建{分析 html/css} 17:05--17:15 升级
a:productlist.html
src
c:jd/js/productlist_02.js
95 line 完成添加商品到购物车
27 line 完成登录保存 uid/uname
28 line
d:升级:登录成功后 uid/uname 保存 cookie
jd/js/shoppingcart_02.js
13 line 完成查看购物车列表
e:jd/data/login_do_02.php
15 line 完成登录成功返回 id

NODE JS

DAY1

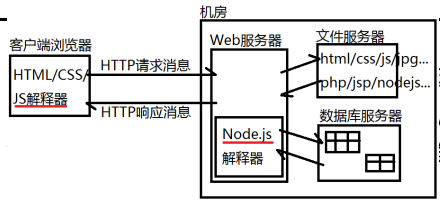
从 FTP 上下载最新版 Node.js 安装程序
64 位 Windows 下载：node-v6.10.2-x64.msi
32 位 Windows 下载：node-v6.10.2-x86.msi
无需提前安装，上课一起安装....

1.阿里面试题：用户在浏览器中输入 www.taobao.com 直到看到页面之间发生了什么?

(1)操作系统访问网络上的 DNS 服务器，把域名转换为 IP 地址
(2)浏览器发起 HTTP 请求消息
(3)Web 服务器接收并解析请求消息，查找指定的资源，可能访问数据库，构建并返回 HTTP 响应消息
(4)浏览器接收并解析响应消息
(5)浏览器缓存接收到响应内容，并解析和渲染响应内容

静态网页 和 动态网页?
静态：网页内容任何人在任何时间访问都是不变的
HTML/CSS/JS/Flash/视频音频....
动态：网页内容不同人在不同时间访问可能是不同的
DB/JSP/PHP/ASP.NET/Node.js

JSP=HTML+Java：功能强大可靠，适合大型企业级项目



网项目

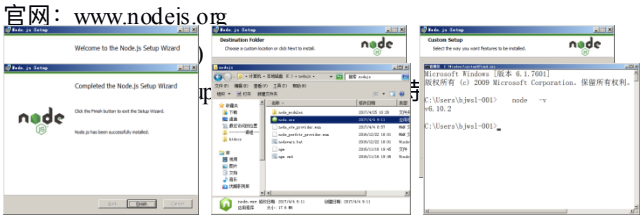
vs 平台

务器端 IO 密集型项目，不适合 CPU 密集型项目

2.Node.js 概述

Node.js 不是 JS，是一种服务器端技术，它的竞争对手是 PHP/JSP/ASP.NET！ 历史上第一次有一种语言可以通吃前后台！

官网：www.nodejs.org



3.Node.js 的两种运行模式

(1)交互模式——一般用于临时测试

REPL：Read Evaluate Print Loop，输入一行代码执行一行

注意：交互模式自带输出功能，不必写 console.log

node 回车

(2)脚本模式——正式项目中使用的方式

把要执行的所有语句编写的一个文本文件中(后缀名任意，没有都行)，一次性提交给 node 解释器执行

node 完整路径名/x.js 回车

提示：只要安装完 Node.js，重启一下 WebStorm，WS 可以自动发现 node.exe 解释器程序。记得新建的项目一定要修改默认的文件编码方式为 UTF-8

练习：创建一个 4.js 文件，打印出九九乘法表，以脚本模式在 WS 中运行

4.面试题：如何自学一门新语言 —— Node.js

(1)了解背景

——百度百科 Node.js

(2)搭建开发环境，编写 HelloWorld

下载并安装 node-v6.10.2-x64.msi

交互模式、脚本模式

(3)数据类型——重点面试题

前端 JS 中的数据类型：

1.基本/原生/值类型

string、number、boolean、null、undefined

2.引用/对象类型

ES 对象类型：String、Number、Boolean、Math、Date、RegExp、Object、Function、Error....

BOM 对象类型：window、document、screen、history、location、navigator、event...

DOM 对象类型：Node、Element、Attr....

用户自定义对象类型： { }

后端 Node.js 中的数据类型：

1.基本/原生/值类型

string、number、boolean、null、undefined

2.引用/对象类型

ES 对象类型：String、Number、Boolean、Math、Date、RegExp、Object、Function、Error....

Node.js 特供对象：目前有几百个....

用户自定义对象类型： { }

(4)变量和常量

```
var age = 20;
const PI = 3.14;
```

(5)运算符

算术运算符

比较运算符

逻辑运算符

位运算符

三目运算符

赋值运算符

特殊运算符 . instanceof typeof

(6)逻辑结构

循环结构： while do..while for(; ;) for(.. in ..) for(.. of ..)

练习：创建 6.js，声明一个保存 5 个学生成绩的数组，使用三种 for 循环依次打印出每个成绩值

选择结构：if..else.. switch..case..

练习：创建一个变量 var path='/index'; 使用二种选择结构，判断 path 的值为哪种（/index、/search、/login），调用不同的执行函数

(7)通用小程序

九九乘法表、100 以内的质数、数组排序、...

练习：打印出 100 以内所有的质数

(8)函数和对象

(9)常用的组件、第三方工具、框架

(10)实际小项目

Modal：模态框

Model：模型

Module：模块

6.Node.js 中的特有概念——模块

一个 Web 项目功能可以分为很多不同的“模块”，如商品管理模块、用户管理模块、支付模块、促销模块、商家管理模块....

Node.js 按照功能的不同，可以把函数、对象分处到不同的文件、目录下，这些文件/目录在 Node.js 中就称为“Module”

Node.js 中每个模块都是一个独立构造函数，解释器会为每个.js 文件添加如下代码：

```
(function(exports, require, module, __filename, __dirname){
//exports：{}  用于声明向外部导出的自己的成员
//require：fn  用于导入其它的模块，创建指定模块对象
//module：
//__filename：
//__dirname：
//自己编写的文件内容
})
```

Node.js 的模块中 exports 和 module.exports 对象的区别是什么?
二者都可以用于向外界导出自己内部的成员。但:
module 变量指代当前模块对象，真正导出的是 modue.exports;
Node.js 底层有代码： exports = module.exports。
所以：若只是给 exports 对象添加新的成员，则等价于给 module.exports 添加新成员； 但是若修改了 exports 的指向，则不会产生实质的作用。

每个模块都可以使用自己的 require()函数引入另一个模块——底层本质就是创建了指定模块的一个对象实例。

```
require('./模块文件名')
```

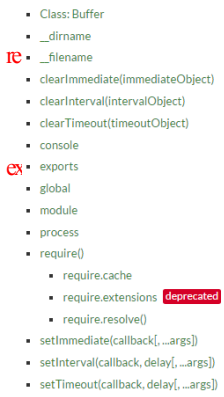
每个模块可以使用 exports 对象向外导出/公开一些自己内部的成员供其它模块使用。

```
exports.成员名 = 成员值;
```

练习：创建一个文件模块：11_Circle.js，其中定义个常量 PI，声明一个方法 getSize(r)，根据传入的圆形半径，返回其面积；以及一个方法 getPerimiter(r)，根据传入的圆形半径，返回其周长。 再创建一个应用程序的主模块 12_app.js，引入 11_Circle 模块，调用其公开的两个成员方法。

练习：编写一个模块 13_ArrayUtil，包含一个方法 sum(arr)，返回指定数组中所有数值的和；一个方法 avg(arr)返回指定数组中所有数值的平均值，使用 module.exports 导出上述两个成员；
再编写 14_app 模块，引入 13_ArrayUtil 模块，调用其公开的方法。

7.Node.js 中模块的分类



模块——安装在解释器内部

块文件名')

义模块 —— Global

使用，而无需 require('global')

exports：用于向外部导出当前模块内部的成员
module：用于指代当前模块
require：用于引入其他模块
__filename：返回当前模块的文件全名
__dirname：返回当前模块文件所在的目录全名
console：指代控制台对象，注意该对象与 Chrome 中 console 不同!

console.log/dir/time/timeEnd/assert….

setInterval(time, fn)

setTimeout(time, fn)

setImmediate(fn)　　等价于 setTimeout(0, fn)

今日练习：

仿写 Node.js 提供的一个模块：URL

创建模块文件 MyURL，向外导出一个方法：resolve(url)，该方法接收的参数形如：

http://www.jd.com:80/ad/index?uname=tom&pno=3

返回对象，形如：

```
{
    protocol:   'http',
    server: 'www.jd.com',
    port: 80,
    path: '/ad/index',
    query: {uname: 'tom', pno:'3'}
}
```

再编写一个主模块，引入上述模块并调用

DAY2

复习：

Node.js 适合于 IO 密集型应用，不适合于 CPU 密集型应用。

JS 和 Node.js 区别：

JS 运行于客户端浏览器中，存在兼容性问题；数据类型：值类型+引用类型(ES+DOM/BOM+自定义)

Node.js 运行于服务器端(V8 引擎)，不存在兼容性问题；数据类型：值类型+引用类型(ES+扩展对象+自定义)。

```
var s1 = {age:10};
var s2 = {age:20};
s2.age++;          //s1.age:10     s2.age:21
var s3 = s1;       //让 s3 也指向对象{age:10}
s3.age++;          //s1.age:11     s3.age:11
s3 = {age:30}          //s1.age:11     s3.age:30
```

Node.js 模块系统：

Node.js 中每个.js 文件都是一个“Module”，每个模块都可以引入其它模块；也可以导出自己的成员供其它模块来使用。

```
(function(exports, require, module, __filename, __dirname){
    exports = module.exports;     //{    }
    var age=20;
    var fn = function(){}

    //exports.age = 20;
    //exports.f = fn;
    //module.exports.age = 20;
    //module.exports.f = fn;
```

```
//exports = { age: age,    f: fn };          //无效语句
module.exports = {age: age, f: fn}       //有效导出语句
})
```

Node.js 中模块的类型：
(1)官方提供的原生模块
global、util、url、fs、http
(2)第三方模块
mysql、oracle、express...
(3)自定义模块
文件模块和目录模块

今日目标：
(1)模块和 NPM —— 有些乱但了解即可
(2)常用的原生模块 —— 重点&难点

1.自定义模块的两种形式

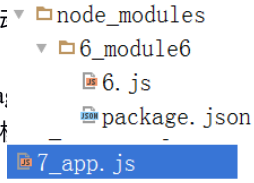
(1)文件模块

创建一个 js 文件，如 m3.js，导出需要公开的数据。其它模块可以 require('./m3')模块

(2)目录模块

方式 1：创建一个目录，假设名为 m4，其中创建名为 index.js 的文件，导出需要公开的数据。其它模块可以 require('./m4')模块

方式 2：创建一个目录，假设名为 m5，其中创建 package.json 文件，其中声明 main 属性指定默认执行的启动 JS 文件，如 6.js，其中导出需要公开的数据。其它模块可以 require('./m5')模块



必须名为 **node_modules**，其中再创建目录模块，假设名为 m6，其中创建 package.json 文件，其中声明 main 属性指定默认执行的启动 JS 文件，如 6.js，其中导出需要公开的数据。其它模块可以 require('m6')模块

练习：使用方式 3，创建两个目录模块，circle、rectangle，都对外公开两个方法 size()、perimeter()，返回指定图形的面积和周长。

最后在最外层的主模块中引入上述两个模块：

circle.size(r)　　circle.perimeter(r)

rectange.size(w, h)　　rectange.perimeter(w, h)

自己尝试：若有个文件名为 m7.js，还有个目录 m7/index.js，还有个 node_modules/m7/index.js；最后 app.js 中 ：

require('./m7')　引入的是哪个模块?

require('m7')　引入的是哪个模块?

2.NPM 包管理器

Node Package Manager：Node.js 的第三方模块/包管理器，可用于下载、更新、删除、维护包依赖关系的工具。

npm 工具默认到 www.npmjs.org 网站下载所需的第三方模块包。

使用 NPM 工具下载一个新的软件包：

npm install 包名

npm uninstall 包名

更多的 NPM 命令参数可以使用 npm -h 进行查看。

3.Node.js 官方提供的原生模块 —— querystring

querystring 模块用于处理 HTTP 请求 URL 中的查询字符串

var obj = qs.parse(str) 把查询字符串解析为 JS 对象

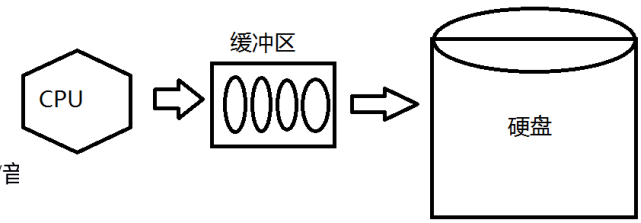
var str = qs.stringify(obj) 把 JS 对象转换为查询字符串

4.Node.js 官方提供的原生模块 —— url

url 模块用于解析一个 HTTP 请求地址，获取其中各个不同的部分

var obj = url.parse(str) 把一个 URL 字符串解析为一个对象

var obj = url.parse(str, true) 把一个 URL 字符串解析为一个对象，并把其中的查询字符串也解析为对象



Buffer

的数据（可能是数字、字符串、二进制图片/音

```
//分配一个指定大小的缓冲区
var buf1 = Buffer.alloc( 1024 );
//使用一个数字数组创建一个缓冲区
var buf2 = Buffer.from( [1, 3, 5] );
//使用一个字符串创建一个缓冲区
var buf3 = Buffer.from( 'abcdefg' );
//把一个缓冲区中的数据转换为字符串
var str = buf3.toString( );
```

6. Node.js 官方提供的原生模块 —— fs —— 重点

fs 模块提供了对文件系统中的文件/目录进行增删改查、读写的功能。

```
//同步读取文件中的内容
var data = fs.readFileSync( file );
//同步向文件中写出内容（删除已有内容）
fs.writeFileSync( file, str/buf )
//同步向文件中追加写出内容（不删除已有内容）
fs.appendFileSync( file, str/buf )
```

练习：使用上述方法，实现把 4.css 文件复制为 4.backup.css

//异步读取文件中的内容
fs.readFile(file, function(err, data){ });
//异步向文件中写出内容（删除已有内容）
fs.writeFile(file, str/buf, function(err){ })
练习：使用异步方法，实现把 4.css 文件复制为 44.backup.css

提示：后续的项目中，文件读写可以使用同步或异步，但推荐使用异步方法——最大限度的发挥 Node.js 的优势。

7. Node.js 官方提供的原生模块 —— http —— 重点

HTTP 模块可用于编写基于 HTTP 协议的客户端程序（即浏览器）；也可以编写基于 HTTP 协议的服务器端程序（即 Web 服务器）

用 http 模块编写一个 Web 服务器：

(1)接收客户端的 HTTP 请求消息

(2)解析客户端请求消息

(3)读取客户端请求的文件

(4)向客户端发送 HTTP 响应消息，主体就是客户端请求的文件

```
var server = http.createServer( );
server.listen(80);
server.on('request', function(req, res){
        //解析请求消息
        //向客户端写出响应消息
})
```

今日练习：

使用 Node.js 创建一个 Web 服务器，根据客户端请求的地址的不同，输出不同的 HTML 页面内容，如

http://127.0.0.1/login.html

服务器应该返回./public/login.html 中的内容；

http://127.0.0.1/news.html

服务器应该返回./public/news.html 中的内容；

若请求资源在 public 目录下不存在，则返回 404 的响应消息。

DAY3

复习：

Node.js 中的数据类型

(1)基本值类型

number、boolean、string、null、undefined

(2)引用/对象类型

1)ES 原生类型

2)Node.js 提供的类型

3)用户自定义的类型

Node.js 中模块(Module)系统

(1)Node.js 官方提供的模块

(2)第三方扩展模块——npm

(3)自定义模块：

文件模块

circle.js：exports.size = fn;

app.js： require('./circle');

目录模块

node_modules/mysql/package.json: { "main":"./index.js" }

node_modules/mysql/index.js: exports.conn = fn

app.js: require('mysql')

Node.js 官方提供的模块：

(1)global

global.console、global.parseInt、global.setInterval

(2)util

```
const util = require('util')
var str = util.inspect(obj)
```

(3)querystring

```
const qs = require('querystring')
var obj = qs.parse('uname=tom&age=20')
```

(4)url

```
const url = require('url');
var obj = url.parse('http://tmooc.cn:80/st/index.html?k=v#c1', true)
```

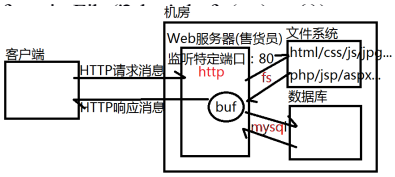
(5)Buffer

缓冲区就是一块内存，用于暂存数据

```
var buf = Buffer.alloc(1024);
var buf = Buffer.from('abcde');
```

(6)fs

```
const fs = require('fs')
//同步 IO 操作
var buf = fs.readFileSync('1.html')
fs.writeFileSync('2.log', buf)
fs.appendFileSync('3.log', buf)
//异步 IO 操作
fs.readFile('1.html', (err, buf)=>{...})
```




```
const http = require('http')
var server = http.createServer()
server.listen(80)
server.on('request', (req, res)=>{
    //读取 req 中的请求信息
    //输出响应消息
})
```

练习：静态 Web 服务器：创建 Web 服务器，接收客户端请求，一律向客户端输出一句<h1>Hello</h1>

练习：动态 Web 服务器：创建 Web 服务器，接收客户端请求，若客户端请求地址是/register，则向客户端输出 public/register.html 中的内容：

```
<form action="register.do ">
    <input name="uname">
    <input name="upwd">
    <input type="submit">
</form>
```

若客户端请求的 URL 是/register.do，则解析 URL 中的查询字符串，把客户端提交 uname 和 upwd 追加写出到 public/user.log 的文件中，并向客户端输出“注册成功”

1.MySQL 中的 SQL

SQL 语句的分类：

DDL:(Data Define Language) —— 定义数据列

CREATE、DROP、ALTER、TRUNCATE

DML:(Data Manipulate Language) —— 操作数据行

INSERT、DELETE、UPDATE

DQL:(Data Query Lanaguage) —— 数据查询

SELECT(最难)

DCL:(Data Control Language) —— 控制用户权限

GRANT、REVOKE

2.使用 Node.js 访问 MySQL 服务器

为了精简 Node.js 解释器，官方没有提供访问任何数据库相关模块，必须使用 npm 工具下载第三方模块。

在 www.npmjs.org 上搜索关键字 mysql，可以得到很多相关的模块，每个模块都有使用说明。

使用 npm 工具下载 mysql 模块：

```
npm  i  mysql
```

mysql 模块的使用步骤：

(1)创建到数据库服务器的连接

```
const mysql = require('mysql');
var conn = mysql.createConnection({...})
```

(2)发送 SQL 语句给数据库服务器来执行

```
conn.query('SQL...', function(err, result){....})
```

(3)关闭连接

```
conn.end()
```

练习：仿写上述代码，使用 Node.js，向 jd/emp 表添加一行新的记录

练习：仿写上述代码，使用 Node.js，将 jd/emp 表中编号为 1 的员工 ename 修改为汤米,工资修改为 6000，

入职时间修改为 1331234567890 —— 注意？占位符的使用

课后练习： 使用 http 和 mysql 模块
创建动态 Web 服务器，接收如下的请求 URL：
/register： 服务器向客户端返回 public/register.html 内容。
/register.do： 接收客户端提交的 uname 和 upwd，保存入 MySQL 数据库，返回“注册成功”。
/login：服务器向客户端返回 public/login.html 内容。
/login.do：接收客户端提交的 uname 和 upwd，查询数据库，返回“登录成功”或“用户名或密码错误”。
/userlist：服务器查询出数据库中所有的用户信息，在一个 TABLE 像客户端输出。

DAY4

复习：
Node.js 模块的分类：
(1)官方提供的模块
global、util、querystring、url、Buffer、fs、http
(2)第三方模块——npm
mysql
(3)自定义模块
文件模块 user.js require('user')
目录模块 node_modules/db/package.json require('db')

http 模块的使用：

```
const http = require('http');
var server = http.createServer()
server.listen(8080)
server.on('request', (req, res)=>{ })
```

mysql 模块的使用：

```
# npm i mysql ./node_modules/mysql
const mysql = require('mysql')
var pool = mysql.createPool({
 host: '127.0.0.1', user: 'root', password: '',
 database: 'jd', connectionLimit: 10
});
pool.getConnection((err, conn)=>{
 conn.query('SQL..?,?', [x, x], (err, result)=>{
        conn.realease()
 })
})
```

练习：

(1)编写 SQL： tmooc.sql，数据库名为 tmooc，表：

stu(sid, sname, score, schoolTime-入学时间)

插入 4 行记录

(2)创建 Node.js WebServer 应用： app.js，创建 http 服务器，监听端口，接收如下请求地址：

/stu/add：向客户端输出/public/stu/add.html 内容

/stu/add.do：接收客户端提交的 sname / score / schoolTime，保存入 tmooc 数据库，返回“添加成功！新记录在数据库中的编号为：5”

/stu/list：向客户端输出数据库中所有的学生信息，以 JSON 格式

Sever&&mysql

1.Day01

程序员键速： 350k/m
5 个月目标：高级前端

李文华
liwenhua@tedu.cn
QQ: 1759496573

1.课程概述
阶段 1(10 天)：DB+PHP 8’
阶段 2(20 天)：HTML+CSS+JS+AJAX 5’
阶段 3(20 天)：JS+DOM+jQuery 7’
阶段 4(20 天)：HTML5+Bootstrap+Vue.js 6’
阶段 5(20 天)：Ionic+AngularJS+React 9’
阶段 6(10 天)：微信+Node.js+项目管理(gulp、webpack、git) 6’

前两个月的项目内容： www.codeboy.com

2.学习方法
程序思维：—— 机器思维
85 95 67 53
---
85 95 67 53
85 67 95 53
85 67 53 95
---
67 85 53 95
67 53 85 95
---
53 67 85 95

编程学习的方法 —— 由人的思维方式转变为机器思维方式：
(1)理解思路/算法/逻辑
(2)仿写别人的代码
(3)默写之前写过的代码
(1)…
(2)...
(3)…
总结：勤思考多练习！

开发电脑：Windows、MacOS
企业服务器：Unix、Linux(Redhat、CentOS、Ubuntu...)

3.程序员常用的快捷键

Windows+E	调出资源管理器（Explorer）
Windows+R	调用“运行”窗口 cmd/calc/mspaint....
Windows+D	显示/隐藏桌面

Alt+Tab　　切换窗口
Alt+Shift+Tab　切换窗口(从后往前)
Alt+F4　　关闭当前窗口
==========
缩略词补全展开功能：
div+Tab　　补全为
h1#bt1+Tab　　补全为<h1 id="bt1"></h1>
h1.bt2+Tab　　补全为<h1 class="bt2"></h1>
h1>span+Tab　补全为<h1></h1>
#box+Tab　　补全为
.box+Tab　　补全为
img*3+Tab　　补全为<img><img><img>
lorem+Tab　　补全为一段随机假文
==========
Ctrl+Alt+↓　　复制当前行
Alt+↑/↓　　移动当前行
Ctrl+D　　删除当前行
Ctrl+/　　注释当前行

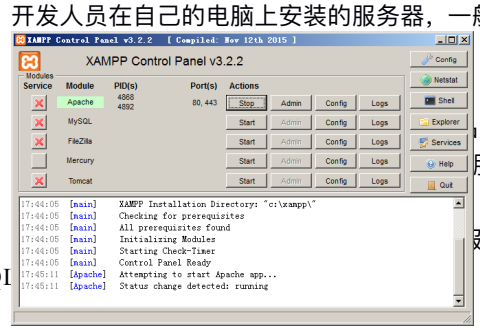
4.软件工程和开发流程 —— 贯穿到整个课程

面试题：软件的生命周期 —— 三个时期八个阶段

(1)可行性研究阶段
(2)需求分析阶段
(3)概要设计阶段　　架构师
(4)详细设计阶段　　设计师
(5)开发实现阶段　　前端/后台/UI 工程师
(6)测试阶段　　测试工程师
(7)部署阶段
(8)维护阶段

5.练习：在自己的（开发者）电脑上安装服务器软件

开发人员在自己的电脑上安装的服务器，一般仅在开发阶段供测试，调试使用，称为“Development Mode



、订单...）

服务器

MySQL

服务器软件的集合，包含常用的各种服务器，如 Apache、

使用 XAMPP 服务器的步骤：

(1)服务器端安装 XAMPP 套装，默认安装在 c:/xampp 目录下；

(2)服务器端编写网页，保存在 Web 服务器的专用目录下，c:/xampp/htdocs

(3)服务器端启动自己的 Web 服务器

---

(4)客户端在浏览器地址栏中输入　http://172.163.0.xx:80，就可以访问到 Web 服务器上提供的网页、图片、CSS、JS 等 Web 内容了。

如何查看自己电脑的 IP 地址
Windows+R => cmd => ipconfig

Day 02

复习：
(1)软件生命周期——三个时期八个阶段
软件定义时期：
可行性研究阶段
需求分析阶段
软件开发时期：
概要设计阶段
详细设计阶段
开发实现阶段
测试阶段
软件部署时期：
部署阶段
维护阶段
(2)快捷键
(3)服务器的概念
XAMPP(xampp.org) = Web 服务器 + 数据库服务器 + 辅助服务器 + ...

遇到问题/BUG 如何解决：
自己分析/尝试解决 => 同桌 => 项目经理 => 讲师 => 文华老师

今日目标
(1)什么是数据库
(2)如何操作数据库
(3)常用 SQL ——重点&难点(英文单词需要记忆)

date：日期
data：数据 database

1.数据库概述

Database Server：数据库服务器，专用于存储网页中的数据。关系型数据库管理系统分为两部分：
服务器端：负责永久存储数据、维护数据，人不能直接观看；服务器上数据的逻辑结构： **Server > Database > Table > Row > Column**
客户端：用于向服务器发起“增删改查”命令，呈现出操作的结果

Oracle MySQL：MySQL 的分支
MariaDB：MySQL 的分支

Deamon：精灵，守护者，守护程序，服务器程序

2.使用 MySQL 服务器的步骤

(1)服务器端：安装 MySQL 服务器端软件

c:/xampp/mysql/bin/mysqld.exe

(2)服务器端：启动服务器端软件

(3)客户端：安装一款 MySQL 客户端软件

c:/xampp/mysql/bin/mysql.exe

(4)客户端：运行客户端程序连接到远程的服务器

在命令行中敲入如下命令：

c:/xampp/mysql/bin/mysql.exe -uroot -p

或者在 XAMPP Shell 中敲入如下命令：

mysql -uroot

练习 1：启动 MySQL 服务器程序，在“任务管理器”中找到 mysqld.exe 程序；再停止 MySQL 服务器程序。

练习 2：使用 MySQL 客户端连接到服务器上，显示当前服务器上已有哪些数据库(show databases;)；退出连接(quit;)。再次重新登录一次。

3.MySQL 常用管理命令

提示：(1)MySQL 命令大小写都可以！推荐在编写关键字的时候用大写字母；非关键字可以小写。(2)所有命令必须以英文分号结束。

SHOW DATABASES;	显示服务器中当前所有的数据库名
USE 库名;	进入指定的数据库中
SHOW TABLES;	显示当前库中有哪些数据表
DESC 表名;	描述指定表的结构(有哪些列)

练习：连接到 MySQL 服务器，在不同的数据库中切换

练习：数一数五个默认库中各有多少个表

练习：查看 phpmyadmin 库中每个表中各有哪些列，找出列数最多的表

4.SQL 语言

结构化查询语言，专用于操作（增删改查）数据库服务器中的数据。是一门国际标准化语言，被各大数据库厂家所支持。

常用的 SQL 语句：

```
#丢弃一个已有的数据库(如果存在的话)
DROP DATABASE IF EXISTS 库名;
#创建新的数据库，其中保存的字符使用指定的字符集
CREATE DATABASE 库名 CHARSET=UTF8;
#进入指定的数据库
USE 库名;
#创建保存特定数据的表
CREATE TABLE 表名 (列名 1 类型, 列名 2 类型, ...);
```

练习 41：创建一个新的文本文件，其中编写 SQL 语句（3 行）——试着删除并重新创建一个用于保存

学生成绩的数据库（名 chengjiguanli），进入该库。

练习 42：创建一个新的文本文件，其中编写 SQL 语句（3 行）——试着删除并重新创建一个用于手机京东项目所需数据的库（名 jd_mobile），进入该库。

练习 43：创建一个新的文本文件，其中编写 SQL 语句（3 行）——试着删除并重新创建一个用于 taobao 网所有数据的库（名 taobao），进入该库。

课后练习：

创建一个文本文件，编写所需的如下语句：

(1)试着删除数据库，如果存在的话： xuezishangcheng；

(2)创建数据库：xuezishangcheng，字符集用 UTF8;

(3)进入上述数据库；

(4) 创建保存笔记本商品信息的表： xz_laptop ，需要的列请分析 http://www.codeboy.com/product_details.html?lid=1

(5)向笔记本商品信息表中插入 5 行数据，即 5 个不同的笔记本的数据；

(6)查看所有的笔记本商品信息；

Day03

复习：

数据库服务器：用于永久存储数据

数据库客户端：用于向服务器发送增删改查命令

Server > Database > Table > Row > Column

MySQL 专用管理命令：

```
SHOW DATABASES;
USE xuezi;
SHOW TABLES;
DESC xz_user;
SHOW CREATE TABLE xz_user;     显示创建指定表所用的 SQL
```

通用 SQL 操作命令：

```
DROP DATABASE IF EXISTS xuezi;
CREATE DATABASE xuezi CHARSET=UTF8;
USE xuezi;
-----------------------------------------
CREATE TABLE xz_user( ..... );
INSERT INTO xz_user VALUES(.....);
INSERT INTO xz_user VALUES(.....);
INSERT INTO xz_user VALUES(.....);
SELECT * FROM xz_user;
```

今日目标：

1)删除和修改语句 —— 重点

2)各种列类型 —— 次重点

3)各种列约束 —— 掌握

1.常用 SQL 命令

添加数据：

INSERT INTO 表名 VALUES (....);

删除数据：

DELETE FROM 表名; #删除所有记录行,慎用!

DELETE FROM 表名 WHERE 列=值; #删除满足条件的行

修改数据：

UPDATE 表名 SET 列=值, 列=值; #修改所有记录行,慎用!

UPDATE 表名 SET 列=值, 列=值 WHERE 列=值;#修改满足条件的行

查询数据：

SELECT * FROM 表名;

练习：试着删除第一行记录，查询所有记录；试着删除第三行记录，查询所有记录

DELETE FROM xz_laptop WHERE lid=15; SELECT * FROM xz_laptop;

练习：试着把第三行记录，笔记本主标题改为“ShenZhou”，价格改为 3000，是否特价改为“Y”，已售数量改为 123

UPDATE xz_laptop SET title='ShenZhou', price=3000, is_onsale='Y', sold_count=123 WHERE lid=18;

练习：创建一个新的文本文件，保存达内公司部门和员工数据

(1)丢弃数据库 danei，如果存在的话

(2)新建数据库 danei，字符集用 UTF8

(3)进入 danei 数据库

(4)创建保存部门信息的表 dn_dept(did, dname, addr)

(5)插入 3 个部门的信息

(6)创建保存员工信息的表 dn_emp(eid, ename, sex, age, dept_id)

(7)为每个部门添加 2 个员工信息

(8)修改编号为 1 的员工的姓名为 King，性别为男，年龄为 50，所在部门编号为 10

(9)删除 20 号部门，并删除其中所有的员工

(10)查询出所有的部门信息

(11)查询出所有的员工信息

2.MySQL 中的列类型

(1)数字类型

TINYINT: 占 1 个字节， -128~127

SMALLINT: 占 2 个字节，-32768~32767

INT: 占 4 个字节，-2147483648~2147483647

BIGINT：占 8 个字节，....

FLOAT(M,D)：单精度浮点型，占 4 个字节，M 表总有效位数，D 表小数点后面的有效位数

DOUBLE(M,D)：双精度浮点型，占 8 个字节

DECIMAL(M,D)：严格定点数，用于精确运算，如货币金额

(2)字符串类型

CHAR(M): M 不能超过 255

VARCHAR(M):M 不能超过 65535

TEXT(M): M 不能超过 2^32，即 40 亿个字符

(3)日期时间类型

DATE：日期类型，必须用引号括起来，采用'yyyy-mm-dd'格式

TIME：时间类型，必须用引号括起来，采用'hh:mi:ss'格式

DATETIME：日期时间类型，必须用引号括起来，采用'yyyy-mm-dd hh:mi:ss'格式

(4)布尔类型

BOOL/BOOLEAN：只能表示 TRUE(等价于 1)或 FALSE(等价于 0)

面试题：CHAR(8)和 VARCHAR(8)的区别

user_name　CHAR(8)：

定长字符串，可能产生空间浪费，但读取速度快

'a'　实际存储为：　'a\0\0\0\0\0\0'

'ab'实际存储为：　'ab\0\0\0\0\0\0'

'abc'实际存储为：　'abc\0\0\0\0\0'

'abcd'实际存储为：　'abcd\0\0\0\0'

'abcde'实际存储为：　'abcde\0\0\0'

'abcdef'实际存储为：　'abcdef\0\0'

'abcdefg'实际存储为：　'abcdefg\0'

'abcdefgh'实际存储为：　'abcdefgh'

'abcdefghi'实际存储为：　'abcdefgh'

user_name　VARCHAR(8)：　变长字符串

定长字符串，不会产生空间浪费，但读取速度稍慢

'a'　实际存储为：　'a\0'

'ab'实际存储为：　'ab\0'

'abc'实际存储为：　'abc\0'

'abcd'实际存储为：　'abcd\0'

'abcde'实际存储为：　'abcde\0'

'abcdef'实际存储为：　'abcdef\0'

'abcdefg'实际存储为：　'abcdefg\0'

'abcdefgh'实际存储为：　'abcdefgh'

'abcdefghi'实际存储为：　'abcdefgh'

3.列上的约束

Constraint：约束，列上的值往往是有限制的，必须满足特定的要求才是有效的，如：

·性别：只能取男或女

·政治面貌：只能取党员、团员、群众之一

·高考成绩：FLOAT(4,1)　取值有规则

·手机号码：有格式要求

·用户名：必须唯一

·登录密码：不能为空，且长度不能少于 N 位

·员工所在部门：可取值必须在部门表中存在

……

(1)主键约束(PRIMARY　KEY)

语法：列名　类型　PRIMARY KEY

声明为“主键”的列上不能出现 NULL 值，且不能重复，如商品编号；

表中所有的记录行会自动按照主键列上的值进行排序 —— 一个表至多只能声明一个主键列。

(2)唯一约束（ UNIQUE ）

语法：列名 类型 UNIQUE

声明为“唯一”约束的列上不能出现重复值，但可以出现多个 NULL

(3)非空约束（ NOT NULL ）

语法：列名 类型 NOT NULL

声明为“非空”约束的列上不能出现 NULL，但可以重复

(4)检查约束 —— MySQL 不支持

(5)默认值约束（ DEFAULT ）

语法：列名 类型 DEFAULT 值

声明了默认值的列若未指定值，则使用默认值；若指定的特定的值，则使用指定的值。

课后练习：创建一个新的文本文档，分析 codeboy.com 上的网页数据，按照如下要求，编写需要的 SQL 语句：

(1)设置后面的 SQL 语句字符编码为 UTF8

(2)删除数据库 xuezi，如果存在的话

(3)创建数据库 xuezi，存储数据所用编码为 UTF8

(4)进入数据库 xuezi

(5)创建笔记本型号表 xz_laptop_family(

fid, 型号编号

fname, 型号名称

laptop_count 该型号下笔记本产品的数量

)

(6)插入 3 个笔记本系列数据，如“Apple MacBook 系列”、“戴尔燃 7000 系列”、“联想小新系列”等

(7)创建保存笔记本信息的表 xz_laptop(

lid, 笔记本编号

title, 主标题

price, 价格

promise, 服务承诺，默认值为“*退货补运费 *30 天无忧退货 *48 小时快速退款 *72 小时发货”

spec, 笔记本规格

is_onsale, 是否特价

sold_count, 已售出数量，默认值为 0

shelf_time, 上架时间,

family_id 所属的型号编号

)

(8)为每种笔记本型号下插入 2~3 款笔记本记录

(9)删除编号为 1 的笔记本，注意：需要修改该笔记本所属型号下笔记本的数量。

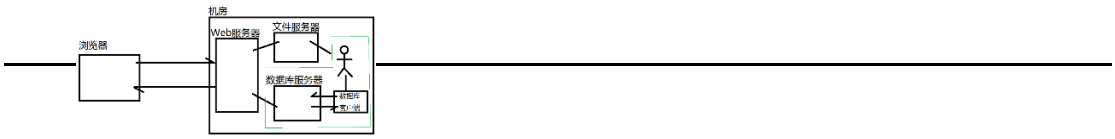
(10)将编号为 2 的笔记本由所属的型号改为另一个型号，注意：需要修改影响到的两个笔记本型号下笔记本的数量。

(11)查询出所有笔记本型号数据和笔记本数据。

Day04

复习：

SQL：结构化查询语言，是一种用于增删改查数据库服务器中数据的语言。




```
SET NAMES UTF8;
DROP DATABASE IF EXISTS xuezi;
CREATE DATABASE xuezi CHARSET=UTF8;
USE xuezi;
CREATE TABLE xz_user(....);
INSERT INTO xz_user VALUES(...);
DELETE FROM xz_user WHERE uid=?;
UPDATE xz_user SET uname=?,upwd=? WHERE uid=?;
SELECT * FROM xz_user;
```

```
12345.6789  = 1234.56789E1
            = 123.456789E2
            = 12.3456789E3
            = 1.23456789E4
```

MySQL 中的列类型 —— 查看参考手册

数字类型：

TINYINT、SMALLINT、INT、BIGINT

FLOAT、DOUBLE、DECIMAL

字符串类型：

CHAR(255)、VARCHAR(65535)、TEXT(4G)

日期时间类型：

DATE、TIME、DATETIME

布尔类型：

BOOL

MySQL 中的列约束

(1)主键约束　PRIMARY KEY

(2)非空约束　NOT NULL

(3)唯一约束　UNIQUE

(4)默认值约束　DEFAULT

(5)外键约束

今日目标：

(1)外键约束

(2)自增列

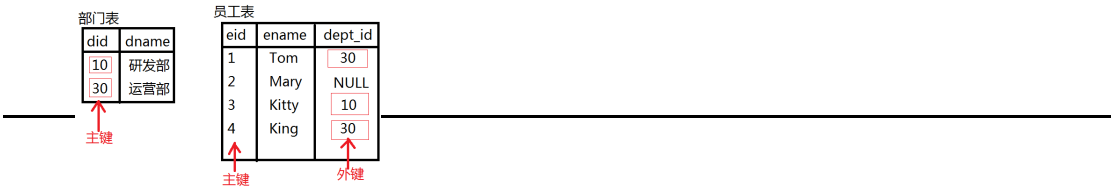
(3)简单查询 —— 重点

(4)复杂查询 —— 重点 & 难点

1.主键约束和外键约束

主键：PRIMARY KEY，唯一且非空约束，是排序依据；

外键：FOREIGN KEY，可重复可为空，外键列上出现的值必须在另外一个表的主键列上出现过



外键约束的语法：　　references

列名　类型,

FOREIGN KEY(列名) REFERENCES　表名(列名)

提示：如果为员工表添加了一个参考部门表的外键列，增加、修改员工都要查询一下部门表，操作效率会降低。老版本的 MySQL 默认是不支持此约束；新版本 MySQL 支持。

2.MySQL 数据库中专有的“自增列”

AUTO_INCREMENT，只有 MySQL 支持，用于实现自增列！自增列无需手工指定特定的值，只需要赋值为 NULL，MySQL 服务器会自动查询当前已有的最大整数，在此基础上+1.

语法：　CREATE TABLE xx (

　　主键列名　INT　　PRIMARY KEY　AUTO_INCREMENT,

　　...

　);

提示：自增列只能用于主键列，且必须是整数型主键。自增列也可以手工赋值。

午间练习：

删除并重新创建数据库　xuezi；

创建保存用户信息的表　xz_user(uid, uname, upwd, phone)；

插入三个用户信息；

创建保存用户收货地址表　xz_receive_addr (aid, rcv_name, addr, rcv_phone,　user_id)；

插入六个收货地址；

3.简单查询语句

(1)只查询特定的列

语法：SELECT　列名, 列名,..列名　FROM　表名;

示例：查询所有的员工姓名及其工资

　　SELECT ename, salary FROM emp;

练习：　　查询所有的部门的名称；

　　SELECT dname FROM dept;

　　查询所有的员工工资、生日、姓名

　　SELECT salary, birthday, ename FROM emp;

(2)查询所有的列

语法：SELECT　*　FROM　表名;

提示：* 指代“所有列”

(3)给列取别名，以简化或者更好的说明

语法：SELECT 列名 AS 别名, 列名 AS 别名 FROM 表名;

　　SELECT 列名 别名, 列名 别名 FROM 表名;

示例：查询所有的员工 birthday（生日）、salary（月薪）、ename（姓名）

　　SELECT birthday AS 生日, salary AS s, ename 姓名 FROM emp;

　　SELECT　birthday　AS　'生　日'　FROM emp;

(4)只显示列上不同的值（即合并相同的值）

语法：SELECT　DISTINCT　列名 FROM 表名;

示例：查看哪些部门有员工

　　SELECT　DISTINCT　dept_id FROM emp;

(5)按照指定列上的值排序

语法：SELECT ... FROM ... ORDER BY 列名 [ASC];

SELECT ... FROM ... ORDER BY 列名 DESC;

示例：查询员工的所有信息，记录行按照工资由小到大排序

SELECT * FROM emp ORDER BY salary;

SELECT * FROM emp ORDER BY salary DESC;

练习：查询员工的所有信息，按照员工姓名排序（升序）

SELECT * FROM emp ORDER BY ename;

查询员工的所有信息，按照员工姓名排序（降序）

SELECT * FROM emp ORDER BY ename DESC;

查询员工的所有信息，按照员工部门编号升序排序；若部门编号相同，再按生日降序排序

SELECT * FROM emp ORDER BY dept_id ASC, birthday DESC;

(6)查询时进行运算

示例：查询每个员工的姓名及其年薪

SELECT ename, salary*12 AS 年薪 FROM emp;

练习：公司给每人加薪 500，查询出每个员工姓名、月薪、年薪

SELECT ename, salary+500 AS 月薪, (salary+500)*12 AS 年薪 FROM emp;

(7)查询出满足特定条件的记录行 —— 重要

语法：SELECT ... FROM ... WHERE 条件;

其中的“条件”可以是多种表达式，如：

= > < >= <= !=

示例：查询出 30 号部门的员工的所有信息

SELECT * FROM emp WHERE dept_id=30;

练习：查询出工资在 18000 及以上的员工的所有信息

SELECT * FROM emp WHERE salary>=18000;

查询出在 1990-10-5 日以前出生的员工所有信息

SELECT * FROM emp WHERE birthday < '1990-10-5' ORDER BY birthday;

查询出不在 20 号部门的员工的所有信息

SELECT * FROM emp WHERE dept_id != 20;

SELECT * FROM emp WHERE dept_id <> 20;

其中的“条件”还可以是多种表达式的组合，如

AND(并且) OR(或者) NOT(非)

示例：查询出工资在 10000~20000 的员工的所有信息

SELECT * FROM emp WHERE (salary<=20000) AND (salary>=10000);

练习：查询出 1990 年 10 月出生的员工；

SELECT * FROM emp WHERE (birthday>='1990-10-1') AND (birthday<='1990-10-31');

SELECT * FROM emp WHERE birthday BETWEEN '1990-10-1' AND '1990-10-31';

查询出 20 和 30 号部门的员工的所有信息

SELECT * FROM emp WHERE (dept_id=20) OR (dept_id=30);

查询出 20、30、40、80 号部门的员工的所有信息

SELECT * FROM emp WHERE (dept_id=20) OR (dept_id=30) OR (dept_id=40) OR (dept_id=80);

SELECT * FROM emp WHERE dept_id IN (20, 30, 40, 80);

查询出部门不在 20、30、40、80 范围的员工的所有信息

SELECT * FROM emp WHERE (dept_id!=20) AND (dept_id!=30) AND (dept_id!=40) AND (dept_id!=80);

SELECT * FROM emp WHERE dept_id NOT IN (20, 30, 40, 80);

课后作业：

(1)把上述“简单查询”中的示例和练习中代码删除，自己编写出需要的语句

(2)创建一个文本文件，编写 SQL：

删除并重建数据库：xuezi

创建商品信息表：xz_laptop(lid, title, pic, price, spec)

创建用户信息表：xz_user(uid, uname, upwd)

创建购物车内容表：xz_shoppingcart_content(

cid 购物车内容编号

laptop_id 所够商品

buy_count 购买数量

user_id 该购物车所属的用户

)

插入适当的测试数据，并查询。

Day05

复习：

day01: 常用 SQL

DDL:(数据定义语言)——定义数据的结构(列)

CREATE / DROP / ALTER / TRUNCATE —— 必须掌握

DML:(数据操作语言)——操作数据的记录行

INSERT / DELETE / UPDATE —— 必须掌握

DQL:(数据查询语言)——对数据没有影响

SELECT —— 必须掌握

DCL:(数据控制语言)——控制用户权限

GRANT / REVOKE

day02: 列类型 —— 必须掌握

数字类型：TINYINT、SMALLINT、INT、BIGINT、FLOAT、DOUBLE、DECIMAL

字符串类型：CHAR、VARCHAR、TEXT

日期时间类型：DATE、TIME、DATETIME

布尔类型：BOOL

列约束

主键、外键、唯一、非空、默认值、检查

day03: 简单查询 —— 必须掌握

查询特定列、列取别名、取不相同的值、运算、排序、条件查询

复杂查询

分组查询、子查询、跨表查询、查询结果合并

```
CREATE TABLE emp (eid INT,  sex CHAR(1) DEFAULT '男');
INSERT INTO emp VALUES(1, '');      //错误!
INSERT INTO emp(eid) VALUES(2);     //正确!
INSERT INTO emp VALUES(3, DEFAULT);     //正确!
```

面试题： DELETE 和 TRUNCATE 二者的区别是什么?

今日目标：

(1)复杂查询 —— 重点&难点
(2)Web 服务器
(3)PHP 语言基础

补充小知识：如何查询部门编号为 NULL 的员工所有信息

SELECT * FROM emp WHERE dept_id=30;
NULL 不能用=和!=进行判定！
SELECT * FROM emp WHERE dept_id=NULL;　　//错误
SELECT * FROM emp WHERE dept_id　IS　NULL;　//正确

如何查询部门编号不为 NULL 的员工所有信息
SELECT * FROM emp WHERE dept_id　IS NOT NULL;

1.简单查询

(8)分页查询 —— LIMIT

语法：　SELECT ... FROM ... WHERE ... ORDER BY ... LIMIT start, count;
start: 表示从满足条件的哪一行记录开始读取，第一行称为 0
count：此次最多读取多少行记录
假设每一页显示 5 条记录：
第 1 页：　... LIMIT 0, 5;
　　第 2 页：　... LIMIT 5, 5;
　　第 3 页：　... LIMIT 10, 5;
第 4 页：　... LIMIT 15, 5;
第 5 页：　... LIMIT 20, 5;
...
第 N 页：... LIMIT (N-1)*5, 5;
练习：分页查询出所有的员工信息，只要第 1 页
　　SELECT * FROM emp LIMIT　0, 5;
　　分页查询出所有的员工信息，只要第 2 页
　　SELECT * FROM emp LIMIT　5, 5;
　　分页查询出所有的员工信息，只要第 3 页
　　SELECT * FROM emp LIMIT　10, 5;　　//3
　　分页查询出所有的员工信息，只要第 4 页
　　SELECT * FROM emp LIMIT　15, 5;　　//0
　　分页查询出 10 部门所有的员工信息，只要第 1 页
　　SELECT * FROM emp WHERE dept_id=10 LIMIT 0, 5;
　　分页查询出 10 部门所有的员工信息，只要第 2 页
　　SELECT * FROM emp WHERE dept_id=10 LIMIT 5, 5;

(9)模糊查询 —— LIKE

示例：查询出姓名包含字母 e 的所有员工信息
　　SELECT * FROM emp WHERE ename='e';　//错误
　　SELECT * FROM emp WHERE ename='%e%'; //错误
　　SELECT * FROM emp WHERE ename LIKE '%e%'; //正确

SQL 查询中，　**%** 代表任意多个任意字符；　_ 代表任意一个字符

练习：查询出姓名中以 m 结尾的员工的所有信息

```
SELECT * FROM emp WHERE ename LIKE '%m';
```
查询出姓名中倒数第二个字符是 i 的员工的所有信息
```
SELECT * FROM emp WHERE ename LIKE '%i_';
```

2.复杂查询

(1)分组函数

COUNT(列名/*): 查询出满足条件的记录行数

MAX(列名): 查询指定列上的最大值

MIN(列名): 查询指定列上的最小值

SUM(列名): 查询指定列上的所有数据的总和

AVG(列名): 查询指定列上的评价值(Average)

示例：查询所有员工的数量
```
SELECT  COUNT(eid)  FROM emp;
SELECT  COUNT(*)  FROM emp;
```
查询出每个部门的员工数量
```
SELECT COUNT(*),dept_id FROM emp GROUP BY dept_id;
```
查询出所有员工工资中的最大值
```
SELECT MAX(salary) FROM emp;
```
练习：

查询出 10 号部门员工的数量
```
SELECT  COUNT(*)  FROM emp WHERE dept_id=10;
```
查询出达内公司部门的数量
```
SELECT COUNT(*) FROM dept;
```
查询出所有员工的工资最小值
```
SELECT MIN(salary) FROM emp;
```
查询出所有员工的工资平均值
```
SELECT AVG(salary) FROM emp;
```
查询出每个部门的编号及该部门的工资最大值、最小值、平均值
```
SELECT dept_id, MAX(salary), MIN(salary), AVG(salary)
FROM emp
GROUP BY dept_id ;
```
查询出所有员工的工资总和
```
SELECT SUM(salary) FROM emp;
```
查询出每个部门的编号及该部门所有员工的工资总和
```
SELECT dept_id, SUM(salary) AS s
FROM emp
GROUP BY dept_id;
```

(2)子查询

在一条查询语句中再嵌入另一条查询，被嵌入的查询称为“子查询”。

示例：查询出 Market 部门的所有员工的所有信息

步骤 1：根据部门名称查询其编号
```
SELECT did FROM dept WHERE dname='Market';
```
步骤 2：根据部门编号查询员工信息
```
SELECT * FROM emp WHERE dept_id=10;
```
整合步骤 1 和步骤 2：
```
SELECT * FROM emp WHERE dept_id=(
SELECT did FROM dept WHERE dname='Market'
```

);
练习：查询出工资比 Tiger 少的员工的所有信息
步骤 1：查询 Tiger 的工资——根据员工姓名查工资
SELECT salary FROM emp WHERE ename='Tiger';
步骤 2：查询工资比 18000 少的员工信息
SELECT * FROM emp WHERE salary<18000;
整合：
SELECT * FROM emp WHERE salary<(
    SELECT salary FROM emp WHERE ename='Tiger'
);

查询出年龄比 Kate 大(即生日比 Kate 小)的员工的所有信息
步骤 1：查询 Kate 的生日——根据姓名查生日
SELECT birthday FROM emp WHERE ename='Kate';
步骤 2：查询生日比'1990-10-3'小的员工的所有信息
SELECT * FROM emp WHERE birthday<'1990-10-3';
整合：
SELECT * FROM emp WHERE birthday<(
  SELECT birthday FROM emp WHERE ename='Kate'
);

(3)跨表查询

跨表/多表查询，一次查询得到的结果集中的列来自于不同的表；跨表查询一定要防止出现笛卡尔积——必须指定两个表中的相等条件。

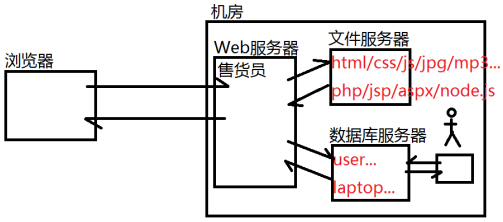
示例：查询每个员工的姓名及其所在部门的名称
SELECT ename, dname
FROM emp, dept;　　//错误，会出现“笛卡尔积”
----------SQL-92----------
SELECT ename, dept_id, dname, did
FROM emp, dept
WHERE emp.dept_id = dept.did; //防止出现笛卡尔积

提示：SQL-92 标准中的跨表查询无法显示连接条件为 NULL 的记录行，如员工的部门编号为 NULL，该记录无法显示

----------SQL-99——了解----------
SQL-99 中的内连接——作用等同于 SQL-92 中的跨表查询：
SELECT ename, dname
FROM emp INNER JOIN dept
ON emp.dept_id = dept.did;
SQL-99 中的左外连接——会显示出左侧表中所有的记录：
SELECT ename, dname
FROM emp LEFT OUTER JOIN dept
ON emp.dept_id = dept.did;
SQL-99 中的右外连接——会显示出右侧表中所有的记录：
SELECT ename, dname
FROM emp RIGHT OUTER JOIN dept
ON emp.dept_id = dept.did;
SQL-99 中的全连接——会显示出两侧表中所有的记录：
SELECT ename, dname
FROM emp FULL JOIN dept
ON emp.dept_id = dept.did;　　//MySQL 不支持全连接

提示：使用结果集的合并操作可以类似的实现“全连接查询”效果。

```
SELECT ename, dname
FROM emp LEFT OUTER JOIN dept
ON emp.dept_id = dept.did
```



3.Web 服务器

Web 服务器的作用：相当于商店里的“售货员”，功能是：

(1)接收并理解客户端请求

(2)查找客户端需要的资源（文件+数据）

(3)发送资源给客户端浏览器

Web 服务器分为两类：

(1)静态 Web 服务器

提供的内容任何人在任何时间访问都是相同的。

静态服务器软件：Apache Httpd、MS IIS、NginX

工作原理：浏览器发起请求，静态 Web 服务器解析请求地址，把特定目录下保存的文件(htdocs/)，发送给客户端——Web 服务器不会对文件内容特殊处理。

练习：访问同桌的 Web 服务器；停止后再次访问试试。

网络小知识：

1.如何查看本机的 IP 地址： ipconfig 命令

2.所有的计算机都默认有一个特殊 IP 地址： 127.0.0.1，永远指代当前计算机

3. 所有的计算机都默认有一个默认域名： localhost，永远指代当前计算机

(2)动态 Web 服务器

提供的内容不同人在不同时间访问都可能不同。

动态网页中的内容是可变的，往往需要数据库访问、其它服务器访问才能实现，只有一些动态的编程语言才能实现。主流的动态网页编程语言：

·JSP：使用 Java 编写网页

·ASP.NET：使用 C#.NET 编写网页

·PHP：使用 PHP 编写网页

·Node.js：使用服务器端 JS 编写网页

4.面试题：如何自学一门编程语言

(1)了解背景：历史、特点、应用场合

(2)搭建开发环境，输出 Hello World

(3)变量和常量

(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)常用组件、工具、框架

(10)实用小项目

5.PHP 简介

PHP： Personal Home Page，后改名为 PHP：Hypertext Preprocessor

练习：创建一个 4.php 文本文件，向客户端输出你的姓名、年龄、电话

课后练习：

(1)删除笔记中“复杂查询语句”部门中的示例代码，根据提示说明写出需要的 SQL 语句。

(2)创建一个文本文件，重名为 99.php，向客户端输出一个九九乘法表

1*1=1

2*1=2 2*2=4

3*1=3 3*2=6 3*3=9

....

(3)创建一个文本文件，其中编写 SQL 语句：

删除并重建数据库：xuezi

创建笔记本型号表：xz_laptop_family(fid, fname)

创建笔记本表：xz_laptop(lid, title, price, shelf_time, sold_count)

创建笔记本图片表： xz_laptop_pic(pid, sm, md, lg, laptop_id)

SM：Small MD：Medium LG：Large

Day06

复习：

所有 SQL 语句相关知识点，参见思维导图

Web 服务器：

作用：等待着客户端连接...接收客户端请求，解析请求内容，查找需要的内容(文件+数据)，发送给客户端响应消息。

分类：(1)静态 Web 服务器 (2)动态 Web 服务器

今日目标：
(1)PHP 中的变量和常量
(2)PHP 中的数据类型 —— 重要！！且不难
(3)PHP 中运算符 ——重要！！且不难

1.补充小知识

如何存储项目中的“日期/时间”数据
方式 1：DATE/DATETIME，不足：不同国家的人日期格式不同的！'2017-12-31'、'12-31-2017'、'31/12/2017'
方式 2：VARCHAR，不足：不方便比较大小 '578'>'2017'
方式 3：BIGINT，表示距离计算机元年经过了多少毫秒 1000 => 1970-1-1 0:0:1
1000*60=> 1970-1-1 0:1:0
1000*60*60=> 1970-1-1 1:0:0
1000*3600*24=> 1970-1-2 0:0:0
1000*3600*24*365=> 1971-1-1 0:0:0
....

计算机元年：1970-1-1 0:0:0

学习新语言的步骤
背景 => 搭建环境 => 变量/常量 => 数据类型 => 运算符 => 通用小程序 ....

2.PHP 中的变量
Variable：变量，指值可能发生改变的量。
数学中的变量：
a = 3;
b = 4;
c = ?
PHP 程序中声明变量的格式：
$变量名 = 值 ;

变量名中可以包含数字、字母、_，不能以数字开头，不能出现空白字符。若由多个单词组成，推荐使用“下划线命名法”或“小驼峰命名法”或“大驼峰命名法”

练习：
1、新建 php 页面，声明三个变量，用来保存一个笔记本商品的信息：
一个变量用来保存笔记本的名称
一个变量用来保存笔记本的价格
一个变量用来保存笔记本的已售数量
2、输出商品的上述三个信息。
3、尝试输出一个未声明过的变量的值，观察运行结果

3.PHP 中的常量
Constant：常量，指一旦赋值就不能再改变的量。
声明常量的格式：
const 常量名 = 值 ;

常量名无需以$开头！！其中可以包含数字、字母、_，不能以数字开头，不能出现空白字符。习惯

上，常量名都使用纯大写字母；若包含多个单词，使用“下划线命名法”，如 MY_FIRST_CONSTANT

练习：创建一个常量，名称为 PAGE_SIZE ，其值赋值为 9；输出该常量指；再试着修改其值，是否可行?

4.PHP 中的数据类型 —— 重点！！！

PHP 中声明变量或常量无需指定类型，但底层所有的数据都是有类型的。var_dump() 函数可以输出变量的类型和值。

PHP 中的数据类型分为“三大类八小种”：

(1)标量类型/值类型

int/integer 整数

float/double 浮点数，PHP 中的 float 等同于 double

string 字符串

bool/boolean 布尔类型

(2)复合类型

array 数组类型

object 对象类型

(3)其它类型

resource 资源类型

null/NULL 空类型

注意：PHP 中的字符串可以使用单引号或双引号括起来，二者区别在于：

'姓名：$ename' 值是 姓名：$ename

"姓名：$ename" 值是 姓名：变量的实际值

练习：创建变量保存用户的用户名、密码、积分、是否在线，输出到页面中。 形如：

用户名： dingding

密码： 123456

积分： 100

是否在线： true?ture? false? flase?

练习：创建多个变量保存商品的编号、标题、价格、是否特价、上架时间等数据，输出到页面中。

5.PHP 中的运算符

(1)算术运算符(结果是数字)

\+ - * / %(取余：只要除法的余数) ++ --

(2)比较运算符(结果是布尔)

> < >= <=

== != ===(全等比较) !==(不全等比较)

(4)赋值运算符(结果是终值) =

注意：

(1)PHP 中的+只能做算术加法运算，不能执行字符串拼接；

(2)+、-、*、/运算中，若参与运算的有字符串，则会自动隐式解析为数字；

(3)==会自动进行隐式类型转换；===若类型不同，直接判定为不等，不会进行隐式转换

(4)$n++是“先赋值再自加”；++$n 是“先自加再赋值”

课后任务：

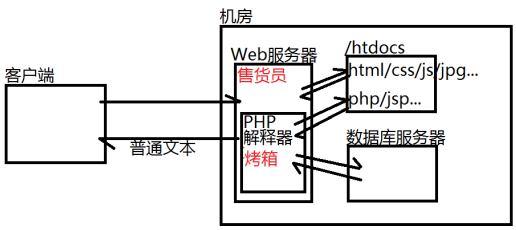
(1)画思维导图，整理出 SQL 所有知识点

(2)根据设计文档，编写 SQL 文件，创建“学子商城”所需的所有数据表和数据，数据从 www.codeboy.com 网站抓取即可。

(3)编写 PHP 网页，实现下面两个练习：

1. 买菜找零：买了 1.6 元的菜，给了老板 2 元钱，问:应找零多少钱?
2. 笔试题：判断输出结果:

```
$n=2;
$r=$n++ + ++$n + $n++;
//或  $r=++n + $n++ + ++$n;
echo($r);
```



学习新语言基本步骤：

(1)背景

(2)搭建环境　　httpd.exe + php.exe

(3)变量和常量

变量：　$变量名 = 值;　　echo $变量名;

常量：　const 常量名 = 值;　　echo 常量名;

(4)数据类型——三大类八小种

标量类型：　int、float、bool、string

复合类型：　array、object

其它类型：　null、resource

echo($age) 和 var_dump($age)的区别?

PHP 中单引号字符串和双引号字符串的区别?

'hello'　　"hello"

'hello $uname'　　"hello $uname"

(5)运算符

算术运算符：+ - * / %　　++　　--

比较运算符：> < >= <= == != === !==

(6)逻辑结构

(7)通用小程序

(8)函数和对象

(9)常用工具、组件、框架

(10)实用小项目

1 'xx' 'yy' 5000 '1.jpg' 3 5 4
2 'xx' 'yy' 5000 '1.jpg' 5 1 1
3 'xx' 'yy' 5000 '1.jpg' 1 4 null
4 'xx' 'yy' 5000 '1.jpg' 2 null 3
5 'xx' 'yy' 5000 '1.jpg' 4 null 2
6 'xx' 'yy' 5000 '1.jpg' null 3 5
7 'xx' 'yy' 5000 '1.jpg' 6 2 null
8 'xx' 'yy' 5000 '1.jpg' null null 6

作业提示：创建表的顺序：
首页轮播广告、首页商品、产品类型、产品、产品图片、用户、用户收货地址、购物车、订单、订单详情

练习：说出下面代码的输出结果

```
$i=5;
$j= 3 + 3 + 5;
echo $j;
```

今日目标：
(1)运算符 —— 比较重要
(2)逻辑结构——选择结构 —— 比较难&重要！
(3)逻辑结构——循环结构 —— 难&重要！

1.PHP 中的运算符

(1)算术运算符
+
(2)比较运算符
== === ...
(3)逻辑运算符
&&(与/并) ||(或) !(取反)
(4)赋值运算符
= += -= *= /= %= .=
(5)位运算符 —— 了解
<< >>
(6)字符串拼接运算符
.
(7)三目运算符
?:

面试题：如何计算一个数字*8 得到的结果

```
$num = $num * 8;
$num *= 8;
$num << 3 ;        //速度最快
```

如何计算一个数字*16 得到的结果

```
$num << 4;
```

如何计算一个数字*32 得到的结果

```
$num << 5;
```

练习： 最后输出$num 的值为多少？ 10:53

```
$num=10;
$num=$num +3;
$num=$num-2;
$num=$num*2;
$num=$num/5;
$num=$num%4;
$num=?
```

练习：

(1)创建一个变量保存用户的年龄，输出此用户是否未退休的成年人。

(2)创建一个变量表示年份，页面中输出这个年份是否为闰年。

注：闰年需要满足下列两个条件之一：

·年份能被 4 整除，且不能被 100 整除的是闰年。

·年份能被 400 整除的是闰年。

午间练习：

1、价格打折：创建变量保存消费金额：如果金额>=100 元，则享受八折。输出实际应收金额

2、创建变量保存用户的一段留言：如果有内容，变量值就是留言的内容；否则就赋值为“主人很懒，什么也没留下”

练习：创建三个变量保存笔记本的编号、标题字、单价，把上述信息输出为如下字符串：

"编号：101 标题字：戴尔燃 7000 单价：￥5888"

要求：用两种语法实现 (1)""字符串形式 (2)使用.运算符

练习：创建三个变量保存用户的用户名、积分、当前是否在线，把上述信息输出为如下字符串：

"用户名：tom 积分：350 当前是否在线：true"

要求：用两种语法实现 (1)""字符串形式 (2)使用.运算符

2.运算符——三目运算符

语法： 表达式 1？表达式 2：表达式 3

含义：若表达式 1 值为 true，则整个式子的总结果为表达式 2 的值；否则整个式子的总结果为表达式 3 的值.

示例：

```
$num1 = 10;
$num2 = 200;
$result = $num1>$num2  ?  $num1  :  $num2 ;
```

练习：创建一个变量保存一个语文成绩，输出一个判定结果，若>=60 分，则输出“及格”，否则输出“不及格”

练习：创建一个变量表示员工工资，按如下规则在页面中输出该工资金额的级别：

(1)大于等于 20000，显示“高工资”

(2)小于 20000 大于等于 8000，显示“中高工资”

(3)小于 8000，显示“普通工资”

练习：体重健康指数（克莱托指数）计算公式如下：

体重(kg)÷身高²(m)

20-25 正常，20 以下偏瘦，25 以上偏胖。

例如：某人是 60kg，1.7m，那就是：

60÷(1.7*1.7)=20.76，属于“正常”体重；

创建变量保存体重和身高，输出判定结果。

3.程序的逻辑结构

程序的三种基本逻辑结构：

(1)顺序执行

(2)选择执行

(3)循环执行

4.选择执行 —— if ... else ....

语法：

```
if( 表达式 ){
    要执行的语句;
}
```

含义： 若“表达式”值为 true，则执行“要执行的语句”

练习：创建变量表示单价、购买数量；若总价超过了￥500，则总价打八折，输出应付的总价。

语法：

```
if( 表达式 ){
    条件满足时执行的语句；
} else {
    条件不满足时执行的语句;
}
```

含义： 若“表达式”值为 true，则执行“条件满足时执行的语句”，否则，执行“条件不满足时执行的语句”

示例：在刚才程序基础上，再创建一个变量表示用户已经支付的金额，若该金额大于等于商品总金额，则计算找零；否则提示“已支付的金额不足”

练习：创建变量保存一个学生的语文成绩，如果大于等于 60，输出及格，否则输出不及格

语法：

```
if( 表达式 1 ){
    条件满足表单式 1 时执行的语句；
} else if( 表达式 2 ){
    条件满足表达式 2 时执行的语句;
} else if( 表达式 3 ){
```

```
        条件满足表达式 3 时执行的语句;
    } else {
        上述条件都不满足时执行的语句;
    }
```

练习：使用 if ... else if ... else ... 实现“科莱托指数”判定

体重 / (身高*身高)　　... 25 ... 20 ...

练习：电话语音系统中，创建一个变量保存用户的语音选项，1-则输出“账户余额查询中，请稍候...”，2-则输出“流量信息查询中，请稍候”，3-则输出“正在转接人工服务...”，其它-则输出“不存在的选项，请重试”

5.选择结构 —— switch ... case ...

语法：

```
switch( 变量名 ){
    case 值 1:                    //变量名===值 1
        满足场景 1 时执行的语句;
        break;
    case 值 2:                    //变量名===值 2
        满足场景 2 时执行的语句;
        break;
    ....
    default:                      //变量名不等于上述值
        上述场景都不满足时执行的语句;
}
```

提示：if...else...能够完成 switch...case...所有的功能；只是在判定某个变量全等情形下，后者看上去更简单些。

课后练习：

(1)继续完善“学子商城”必需的 SQL 语句。

(2)**默写**笔记中所有的“示例”和“练习”代码。

(3)学子商城中使用 bool 类型变量保存一个商品是否特价，但是输出在页面中需要显示出“特价商品”或者“非特价商品”等提示信息，请根据 bool 类型变量的值，进行输出。

要求：使用三目运算符和和 if..else...两种方式实现

(4)学子商城中用户的订单状态在数据库中使用一个整数来存储，比如数值 1 表示“等待付款”，2 表示“备货中”，3 表示“运输中”，4 表示“派货中”，5 表示“订单完成”，请创建一个变量保存这样的一个整数，在页面中输出其对应的提示文字。

要求：使用三目运算符和 if..else...和 switch...case...三种方式实现

Day08

复习：
掌握一门新语言的基本步骤
(1)背景　　PHP、JSP、ASP.NET、Node.js
(2)搭建环境
(3)变量和常量
$变量名 = 值;
const 常量名=值;　define('常量名', 值);
(4)数据类型——三大类八小种
　　标量类型/值类型：int、float、bool、string
复合类型：array、object
其它类型：null、resource
(5)运算符
算术：+ - * / %　　++　--
　　比较：> < >= <= == === != !==
逻辑： && || !
　　赋值：= += -= *= /= %= .=
位： << >>
字符串拼接： .
三目： 表达式 1 ? 表达式 2 : 表达式 3
(6)逻辑结构
顺序执行
选择执行　if..else...　　switch...case...
循环执行

练习：数据库保存员工的政治面貌，使用一个数字，如 1-党员、2-团员、3-群众、其它-未知；创建一个整型变量保存一个政治面貌，输出对应的字符串
要求：使用三种方式实现

今日目标：
(1)循环结构 —— while —— 掌握
(2)循环结构 —— do..while —— 掌握
(3)循环结构 —— for —— 重点
(4)循环结构 —— foreach —— 重点
(5)数组 —— 重点&难点

1.循环结构 —— while 循环

语法：
```
while( 循环条件 ){
    循环主体
}
```
含义：当“循环条件”为 true 时，就执行一遍“循环主体”；再次判定....
练习 1：使用 while 循环从 0 输出到 9
练习 2：使用 while 循环在一行中输出 50 个※
练习 3：使用 while 循环输出 10 行 50 列的※

※※※※※※※※※※※※※※※※※※※※※※※
※※※※※※※※※※※※※※※※※※※※※※※
方法 1： 逢 50 输出一个
 while + if
方法 2： 把"一行+
"循环 10 遍 while+while

2.循环结构 —— do...while 循环

语法： do{
 循环主体;
 }while(循环条件);
含义： 执行循环主体，再判定循环提交是否满足，若满足则再次执行循环主体，否则就退出循环
区别：do...while... 循环主体至少执行一次；while... 循环的主体则可能一次都不执行。
练习 1：使用 do..while 循环从 0 输出到 9
练习 2：使用 do..while 循环在一行中输出 50 个※
练习 3：使用 do..while 循环输出 10 行 50 列的※

3.循环结构 —— for 循环

语法： for(表达式 1; 表达式 2; 表达式 3){
 循环主体;
 }

```
$i=0;
while( $i<10 ){
    echo $i;
    $i++;
}
```

```
for( $i=0; $i<10; $i++){
    echo $i;
}
```

含……十数器赋初始值。再执行表达式 2 的判定，若为 true……3——一般是计数器改变；然后再次执行表达式 2 的……

练习 1：使用 for 循环从 0 输出到 9
练习 2：使用 for 循环在一行中输出 50 个※
练习 3：使用 for 循环输出 10 行 50 列的※
练习 4：计算 1/5+1/10+1/15+1/20+...1/50=?
 分析： $i： 5、10、15、20....50
练习 5：计算 8/9+7/8+6/7+5/6+4/5+3/4+2/3+1/2=?
 分析： $i: 8、 7、 6、 5、.... 1
练习 6：通用小程序：使用※ 打印如下的图形：

```
※           1-1
※※          2-1/2
※※※         3-1/2/3
※※※※        4-1/2/3/4
※※※※※       5-1/2/3/4/5
```

练习 7：通用小程序：使用※ 打印如下的图形：

```
※※※※※       1-1/2/3/4/5
※※※※        2-1/2/3/4
※※※         3-1/2/3
※※          4-1/2
※           5-1
```

4.数组 —— Array —— 重要

保存一个学生的成绩： $score = 730;

保存 5 个学生的成绩： $arr = [710, 690, 550, 430, 720];

注意：

(1)数组和普通变量不同

```
$x = 100;          //int(100)
$y = [100];        //array()
```

(2)数组变量不能使用 echo 进行输出！

(3)保存在数组变量的中每个值都需要有一个下标(序号)，访问某个特定的元素，需要使用其下标

语法：

```
$变量名 = [ 值 1,  值 2, .... ];
```

访问第$i 个元素：

```
$变量名[ $i ]
```

获得数组中元素的个数：

```
count($变量名)
```

添加新元素：

```
$变量名[ ] = 新的元素值;
```

练习：创建一个空数组，向其中保存一个员工的工资，再向其中保存一个员工的工资，再向其中保存一个员工的工资，再向其中保存一个员工的工资，使用循环输出所有的工资值

课后练习：

(1)通用小程序：使用 while、do..while、for 循环三种方法输出“九九乘法表”

(2)通用小程序：打印出所有的“水仙花数”。 水仙花数是指一个 3 位正整数，它的每个位上的数字的三次方之和等于它本身。（例如：1*1*1 + 5*5*5+ 3*3*3 = 153）

(3)通用小程序：输出 100 以内的质数/素数——只能被 1 和它自身整除的数 —— 思路：

外层循环$i 从 2~100

内层循环$j 从 2~$i-1，每个都试着整除$i，若都不能整除则说明$i 是质数

Day9

复习：

掌握新语言的步骤：

(1)背景

(2)搭建环境

(3)变量和常量

(4)数据类型

标量类型：int、float、string、bool

复合类型：array、object

其它类型：null、resource

(5)运算符

算术、比较、逻辑、赋值、位、三目、特殊

(6)逻辑结构

顺序执行
选择执行　if...else...　　switch...case...break...
循环执行　while(){ }　do{ }while()　for(; ;){ }
(7)通用小程序
※ 打印三角、九九乘法表、水仙花数、质数....
(8)函数和对象
(9)常用组件、工具、框架
(10)实用小项目

练习：输出 100 以内所有的整数
练习：输出 100 以内偶数的和
练习：输出 100 以内奇数的和
练习：输出 Hello0、Hello1、Hello2...Hello10

今日目标：
(1)数组 —— 重点
(2)预定义数组 —— 重点
(3)函数 —— 掌握
(4)实用数据库连接函数 —— 重点&难点

1.PHP 中的两种数组

(1)索引数组(Indexed Array):每个元素的下标都是一个数字，形如 0、1、2.....N，数组的总长度为 N+1
声明方式：　$arr = [95, 78, 66];　　$arr = ['丁丁', '当当','豆豆']
访问元素：　echo $arr[$i];
添加元素：　$arr[ ] = 80;
获取长度：　echo count($arr);
练习：创建一个数组，保存购物车中的三个商品的编号。
再向添加一个新的商品编号；再向添加一个新的商品编号；
输出现有的商品的数量；循环输出所有的商品编号
练习：创建一个数组，保存购物车中的三个商品的名称。
再向添加一个新的商品名称；再向添加一个新的商品名称；
输出现有的商品的数量；循环输出所有的商品名称

思考：一个 PHP 数组中可以放置纯数字、纯字符串，可以放置多个不同类型的值吗?
$arr = [8, '戴尔燃 7000', 5388.00, true, 1490123456789];
语法上可以！但实际项目中上述数组表意不清！

(2)关联数组(Assocation Array)：每个元素的下标都是一个自定义的字符串，系统不会自动生成下标
声明语法：　$arr = ['lid'=>8 , 'title'=>'戴尔燃 7000', 'price'=>5388.00, 'isOnsale'=>true];
元素数量：　echo　count($arr);
获取元素：　echo　$arr['title'];
添加元素：　$arr['shelfTime'] = 1490132455679;

注意：关联数组每个元素的下标都是自定义的字符串，不是 0/1/2/3...，不能使用传统 for 循环进行遍历。

**$user['uname']."
"　其中 uname 的单引号不能省略!**
**"$user[uname]
"　其中的必须省略 uname 的单引号**

练习：创建关联数字，保存一个用户的所有信息，编号、用户名、密码；

再向上述关联数字中添加新的下标：积分
再向上述关联数字中添加新的下标：是否在线
输出数组中元素的个数；
输出数字中每个元素的下标和值

2.逻辑结构——循环结构 —— foreach
foreach：对于数组中的每一对元素，都看作一个下标遍历指向一个值变量，都执行一次循环主体。
语法：
```
foreach( $数组名  as  $key=>$value  ){
        echo  $key . $value;
}
foreach( $数组名  as  $value  ){
        echo  $value;
}
```
含义：依次把数组中的每一对下标=>值，赋值为指定的两个变量，执行一遍循环体。
提示：foreach 循环既可以遍历关联数组，也可以遍历索引数组！

练习：创建一个关联数组，保存一个商品的信息，包括编号、名称、单价、上架时间，使用 foreach 遍历其中的每个元素。

练习：创建一个关联数组，保存一个商品的信息，包括编号、名称、单价、上架时间；
再创建一个关联数组，保存一个商品的信息，包括编号、名称、单价、上架时间；
再创建一个关联数组，保存一个商品的信息，包括编号、名称、单价、上架时间；
把上述三个关联数组保存在一个大的索引数组中，下标分别为 0/1/2，使用循环输出整个大数组中所有的数据

练习：
创建一个关联数组，保存一个用户的信息，编号、用户名、电话、头像；
创建一个关联数组，保存一个用户的信息，编号、用户名、电话、头像；
创建一个关联数组，保存一个用户的信息，编号、用户名、电话、头像；
再创建一个索引数组，包含上述三个关联数组。
尝试使用 for 循环输出上面的大数组中的所有数据；
尝试使用 foreach 循环输出上面的大数组中的所有数据；

练习：
学子商城中每个用户的“收货地址列表”中都可能包含多个收货地址，而每个收货地址又包含编号、接收人姓名、地址、联系电话四个属性。
试创建一个收货地址列表变量(二维数组)，其中包含三个收货地址。
使用 for 输出所有的收货地址信息；
使用 foreach 输出所有的收货地址信息。

3.函数 —— 了解

```
$num1 = 10;
$num2 = 20;
$max = $num1>$num2 ? $num1 : $num2;
echo $max;
```
若上述代码需要反复使用，多次复制粘贴会导致今后的代码维护工作很麻烦！
getMax {

```
$num1 = 10;
    $num2 = 20;
    $max = $num1>$num2 ? $num1 : $num2;
    echo $max;
}
```

上述代码就是在创建一个“函数”！

定义：是一段预定义好，有名称的，并可以被反复使用的代码块，其中可以包含多条可执行语句。

语法： function 函数名(形式参数列表) {
函数执行主体;
return 返回值;
}

调用： 函数名(实参列表); 函数名(实参列表);

练习：使用函数封装“需要反复调用”的代码

编写一段代码，从 0 输出到 9；

把上述代码封装为一个函数：pringNumber；

调用上述函数；再次调用上述函数；再次调用上述函数；

函数名中可以包含数字、字母、下划线，不能为数字开头。 若包含多个单词，可以使用下划线法则、或者大/小驼峰法则。
函数声明时，参数列表中可以声明 0~N 个参数，用逗号分隔。
函数可以声明一个返回值，把自己内部计算后的结果返回给函数的调用者： return 值； 函数体内此句后面不能再有其它的语句了。

练习：使用函数封装“比较难以编写”的代码

编写一段代码，输出一个九九乘法表；

把上述代码封装为一个函数：print99

调用上述函数；再次调用上述函数；再次调用上述函数；

练习：创建一个函数计算两个整数的和

创建一个整型变量$num1，值为 10；

创建一个整型变量$num2，值为 20；

创建一个整型变量$sum，值为$num1 和$num2 的和；

输出$sum 的值；

把上述四行代码封装为一个函数；

调用该函数。

练习：创建一个函数 leiJia( )，接收一个整数做参数$n，函数体内计算出 1+2+3+...$n 的累加和，输出该和。

调用该函数，计算 1 到 10 的累加和；

调用该函数，计算 1 到 100 的累加和；

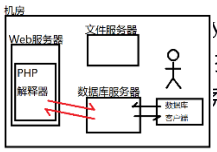
练习：创建一个函数：arrAdd，接收一个整型数的数组做参数，函数体内计算该数组中所有元素的和，返回该和。

调用该函数，实参为：[1, 3, 5]，得到返回的结果并输出；

调用该函数，实参为：[2, 4, 6, 8]，得到返回的结果并输出；

Improved： 改进的，改良后的

4.1



ySQL 服务器

操作 MySQL： mysql_xxx()；

秀的操作 MySQL 的函数库： mysqli_xxx()；

使用 MySQLi 函数库连接 MySQL 服务器的步骤：

提示：操作过程与“命令行客户端连接 MySQL 服务器”过程是一样的。

(1)连接到 MySQL 服务器

```
$conn = mysqli_connect('服务器地址', '用户名', '密码', '数据库名', 端口号);
```

(2)向 MySQL 服务器发送 SQL 命令

```
$sql = "....";
$result = mysqli_query($conn, $sql);
```

(3)查看执行的结果

```
if($result===true){              //执行成功
}else {                          //执行失败
}
```

(4)断开到 MySQL 服务器的连接——可以省略

```
mysqli_close($conn);
```

课后练习：

(1)创建一个函数名为 arrMax，接收一个数组做参数，函数体内查找出数组参数中的最大值，并返回该值。

调用上述函数，传递参数[10, 5, 30, 8]，输出函数的返回值

(2)创建一个函数名为 arrMin，接收一个数组做参数，函数体内查找出数组参数中的最小值，并返回该值。

调用上述函数，传递参数[10, 5, 30, 8]，输出函数的返回值

(3)创建一个函数名为 arrAvg，接收一个数组做参数，函数体内查找出数组参数中所有数值的平均值，并返回该值。

调用上述函数，传递参数[10, 5, 30, 8]，输出函数的返回值

(4)创建一个 SQL 文件，其中编写如下的 SQL 语句：

删除并重建数据库：xuezi9，字符集使用 UTF8；

创建表 xz_user(uid, uname, upwd, email, phone)

(5)创建一个 PHP 文件，使用 mysqli 函数库中提供的 mysqli_connect()函数连接到 MySQL 服务器，再使用 mysqli_query()函数提交一条 INSERT 语句，向 xz_user 表中插入一条新的用户记录。

提示：上述操作由于涉及到数据库服务器和 Web 服务器，非常容易出错，需要慢慢的积累错误调试技巧。

Day10

复习：

学习新语言的步骤

(1)背景

(2)搭建环境

(3)变量和常量

(4)数据类型

标量类型：int、float、string、bool

复合类型：array、object

其它类型：null、resource

(5)运算符

算术

比较
逻辑
赋值
拼接
位
三目
特殊
(6)逻辑结构
顺序执行
选择执行 if..else... switch..case..break
循环执行 while(){} do{}while() for(){} foreach(){}
(7)通用小程序
九九乘法表、水仙花数、打星星、质数判定...
(8)函数和对象
(9)组件、工具、框架
(10)实用项目

为什么要用函数：封装“反复多次执行的代码”、“不易编写的代码”
如何声明函数： function 函数名(...){要执行的代码}
如何调用函数： 函数名(...);
什么是形参： function fl($n1, $n2, $n3){ .... }
什么是实参： fl(10, 30, 50)
什么是返回值：function fl(){... return 值;}
$rtnValue = fl();

使用 PHP 提供函数访问 MySQL 数据库
(1)连接到数据库
```
$conn = mysqli_connect('','','','','');
```
(2)向数据库发送 SQL 命令
```
$sql = "...";
$result = mysqli_query( $conn, $sql );
```
(3)查看执行结果
```
if($result===true){
}else {
}
```

今日目标：

使用 PHP 提供函数实现增删改查（CRUD）——重点&难点

Create/Retrieve/Update/Delete

1.学子商城中的数据结构

参照设计图，完整实现必须的表和记录

$result = mysqli_query($conn,$sql);
上述函数的返回值：
(1)对于 DML(insert/delete/update)语句：
执行失败，就返回 false
执行成功，就返回 true
(2)对于 DQL(select)语句：
执行失败，就返回 false
执行成功，就返回 查询结果的描述对象

2.实现学子商城中“用户注册”功能点必需的服务器端页面

data/register.php：接收客户端提交的注册信息，保存入 MySQL 服务器，向客户端返回成功或失败

```
//1 连接数据库
$conn = mysqli_connect(...);
//2 提交 SQL 语句
$result = mysqli_query($conn, "INSERT INTO ....");
//3 查看执行结果
if($result === true){
    //执行成功，自增编号为: mysqli_insert_id($conn);
}else {
    //执行失败！最大可能 SQL 语法错误！
}
```

3.实现学子商城中“用户删除”功能点必需的服务器端页面

data/user_delete.php：根据用户编号 uid，从数据库中删除指定的用户

```
//1 连接数据库
$conn = mysqli_connect(...);
//2 提交 SQL 语句
$result = mysqli_query($conn, "DELETE FROM....");
//3 查看执行结果
if($result === true){
    //执行成功，影响在行数: mysqli_affected_rows($conn);
}else {
    //执行失败！最大可能 SQL 语法错误！
}
```

4.客户端浏览器向服务器端 PHP 页面传递数据

请求地址：

http://主机地址/页面名?k1=v1&k2=v2&k3=v3

浏览器地址栏中的页面名后面，以?开头，拼接了很多 key-value 对，彼此间用&符号拼接起来！——这样的字符串称为“查询字符串(Query String)”——用于向服务器端页面传递数据的。

服务器端 PHP 如何接收上述数据?

使用 **$_REQUEST** 这个预定义的数组变量！

```
$n = $_REQUEST['uname'];
或者
@$n = $_REQUEST['uname'];            //屏蔽警告消息
if($n===null){                       //客户端未提交数据
    die( 'uname require' );
}
```

练习：创建一个 add.php，客户端浏览器访问该页面，传递两个数据：num1 和 num2，add.php 读取这两个数据，在页面输出这两个请求数据的和。

练习：创建一个 leijia.php，客户端浏览器访问该页面，传递一个数据：num，leijia.php 读取这个数据，在页面输出从 1 到该数字的累加和。 1+2+3+....num

```
PHP 预定义的变量 —— 无需声明可以直接使用
$_SERVER
$_GET
$_POST
$_FILES
$_COOKIE
$_SESSION
$_REQUEST   该数组中保存着客户端请求中提交的数据
$_ENV
```

5.PHP 中的页面包含

有 1.php：

```
$num1 = 10;
echo $num1;
```

有 2.php：

```
$num2 = 20;
echo $num2;
#echo $num1;   #错误！变量不能跨页面使用！
require('1.php');   #在当前位置包含指定的文件
echo $num1;    #正确！
```

require()在项目中常用于包含多个页面都需要的公共页面。

6.实现学子商城中“修改用户基础信息”功能点必需的服务器端页面

data/update_basic.php：根据客户端提交的 uid，以及修改后的 user_name、gender、email、phone 修改数据库中的对应记录。提示：UPDATE xz_user SET user_name='?',gender='?',email='?',phone='?' WHERE uid=?

#1 接收客户端提交的请求数据

#2 连接到数据库

#3 向数据库提交 SQL 语句

#4 查看执行结果

7.实现学子商城中“用户登录”功能点必需的服务器端页面

data/login.php：接收客户端提交的 uname 和 upwd，执行数据库查询，若能查询到一行记录，则返回 login succ，否则 uname or upwd err。 提示：SELECT * FROM xz_user WHERE uname='?' AND upwd='?';

#1 接收客户端提交的请求数据

#2 连接到数据库

#3 向数据库提交 SQL 语句

#4 查看执行结果

从查询结果集中抓取记录，有三个方法可用：

```
$row = mysqli_fetch_row($result); //抓取一行，索引数组
$row = mysqli_fetch_assoc($result);//抓取一行，关联数组

$rowList = mysqli_fetch_all($result, MYSQLI_ASSOC);    //抓取所有行，二维数组
```

课后练习：

(1)完成“学子商城”中“查询所有用户”功能所需的服务器端页面 user_list.php，向客户端输出所有的用户信息。

(2)完成“学子商城”中“修改用户密码”功能所需的服务器端页面 update_password.php：接收客户端提交的用户 uid、oldpwd 和 newpwd， 根据 uid 和 oldpwd 先 SELECT 该记录是否存在，若不存在则向客户端输出‘原始密码错误’；否则执行 UPDATE，把该用户的密码修改为 newpwd，返回‘update succ’。