**Day1**

**一 服务器和客户端**

通俗的讲，能够提供某种服务的机器（计算机）称为服务器。

**1.服务器软件**

使计算机具备提供某种服务能力的应用软件，称为服务器软件，通过安装相应的服务软件，然后进行配置后就可以使计算具备了提供某种服务的能力。

**2 HTTP服务器**

即网站服务器，主要提供文档(文本、图片、视频、音频)浏览服务，一般安装Apache、Nginx服务器软件。

HTTP服务器可以结合某一编程语言处理业务逻辑，由此进行的开发，通常称之为服务端开发。

常见的运行在服务端的编程语言包括 PHP、Jsp、Asp、Python、Ruby、Perl等。

HTTP 服务器软件：Apache、Nginx、IIS、Tomcat、NodeJS等；

**3客户端**

具有向服务器索取服务能力的终端，如比如手机、电脑等，通过安装不同的客户端软件，可以获取不同的服务，比如通过浏览器获得www服务。

常见的HTTP客户端软件：浏览器。

以浏览器为宿主环境，结合 HTML、CSS、Javascript等技术，而进行的一系列开发，通常称之为前端开发。

**二 网络基础**

**1 IP地址**

所谓IP地址就是给每个连接在互联网上的主机分配的一个32位地址。(就像每部手机能正常通话需要一个号码一样)

查看本机IP地址 ping、ipconfig、ifconfig

**2 域名**

由于IP地址基于数字，不方便记忆，于是便用域名来代替IP地址，域名是一个IP地址的“面具”

**3 DNS服务**

DNS（Domain Name System）因特网上作为域名和[IP地址](http://baike.baidu.com/view/3930.htm" \t "_blank)相互映射的一个[分布式数据库](http://baike.baidu.com/view/68389.htm" \t "_blank)，能够使用户更方便的访问[互联网](http://baike.baidu.com/view/6825.htm" \t "_blank)，而不用去记住能够被机器直接读取的IP数串。

简单的说就是记录IP地址和域名之间对应关系的服务。

**4 端口**

端口号是计算机与外界通讯交流的出口，每个端口对应不同的服务。

现实生活中，银行不同的窗口办理不同的业务。

查看端口占用情况 netstat -a

常见端口号 80、8080、3306、21、22

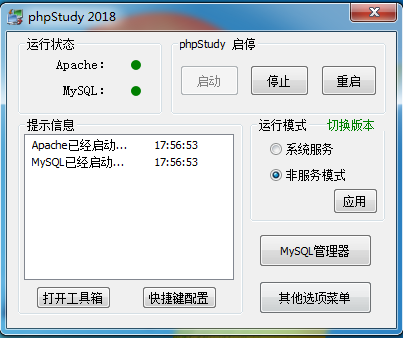
一般用到的是1到65535,其中0不使用,1-1023为系统端口,也叫BSD保留端口

**三 搭建HTTP服务集成环境**

**1安装phpStudy**

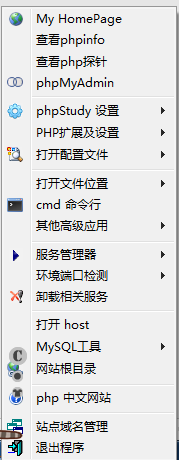
安装phpStudy，和普通软件安装无差别，除指定安装路径外，其它默认安装。

**2管理HTTP服务**

****

小圆点为绿色为正常启动状态

通过图形控制更改设置:



**注意事项：**

1、检查网络是不是通的 ping 对方IP

2、检查防火墙是否开启，如果开启将不能正常被访问

3、检查访问权限 Allow from all

4、理解默认索引

5、确保端口没有被其它程序占用

6、“#”表示注释

7、修改配置要格外小心，禁止无意修改其它内容

**3网站部署**

将我们制作好的网页拷贝到www根目录下，浏览器访问127.0.0.1即可,如果要访问其他用户则使用其局域网地址,如:10.36.150.176

**四PHP基础**

文件以.php后缀结尾，所有程序包含在<?php /\*\* 这里是代码 \*\*/ ?>



**避免使用中文目录和中文文件名**

**1变量**

1)变量以$开头 字母/数字/下划线 不能以数字开头

2)大小写敏感（区分大小写）：变量名、配置项

3)大小写不敏感（不区分大小写）：函数名、方法名、类名 不区分大小写，但推荐使用与定义时相同的名字

**2数据类型**

字符型、整型、浮点型、布尔型、数组、对象、NULL

索引数组、关联数组（了解即可）

**3内容输出**

echo：输出简单数据类型，如字符串、数值

print\_r()：输出复杂数据类型，如数组

var\_dump()：输出详细信息，如对象、数组（了解）



**4运算符**

基本与Javascript语法一致

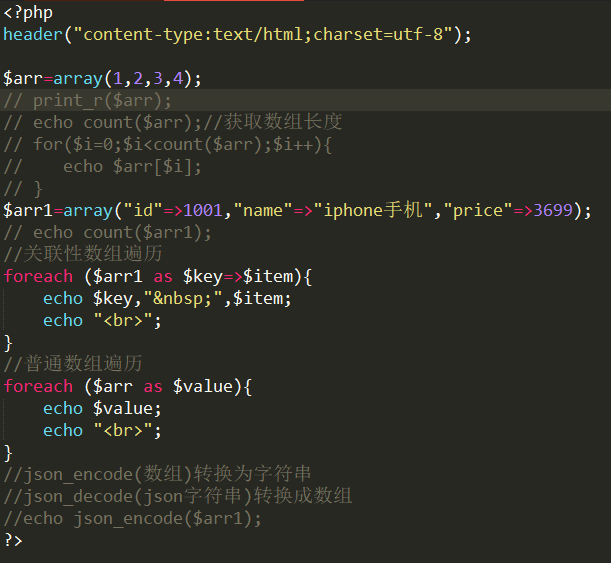
**. 号**表示字符串拼接符**，**Javascript中为+号 

**5函数**

与Javascript基本一致

**6分支、循环语句**

与Javascript基本一致



**7文件引入**

include、require

**8表单处理**

表单name属性的是用来提供给服务端接收所传递数据而设置的

表单action属性设置接收数据的处理程序

表单method属性设置发送数据的方式

$\_GET接收 get 传值

$\_POST接收 post 传值

$\_REQUEST 接收get/post传值

## 9常用PHP函数

in\_array(key, $arr) 搜索数组中是否存在指定的值

count($arr) 计算数组长度

array\_key\_exists (key,$arr)检测数组$arr中是否存在key

unset($arr[key]) 删除数组中一个

## 五 PHP操作数据库

1 数据库概念（数据库，表，主键）

\* MySQL 是最流行的关系型数据库管理系统（非关系型数据库简略介绍）

\* 关系数据库管理系统(Relational Database Management System)的特点

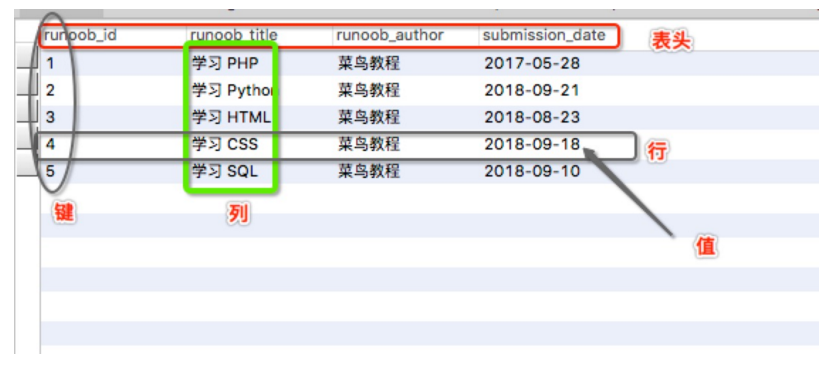
\* 数据以表格的形式出现

\* 每行为各种记录名称

\* 许多的行和列组成一张表单

\* 若干的表单组成database

\* 主键：主键是唯一的。一个数据表中只能包含一个主键。你可以使用主键来查询数据。



\* 数据类型（了解即可）







2 Navicat 数据库操作

* Navicat下载安装
* 链接数据库
* 新建数据库
* 数据导入

5.10.3 php操作数据库常用方法

\* mysqli\_connect() ：数据库链接

\* 接受五个参数（数据库主机名、数据库用户名、数据库密码、库名,端口号）

\* mysqli\_connect\_error() ：数据库的报错

\* mysqli\_num\_rows(): 结果的行数

\* mysqli\_query() ：执行SQL代码

\* 接受二个参数（数据库链接，数据库查询语句——SQL语句）

\* mysqli\_fetch\_assoc() 将数据转换为数组格式

\* 接受一个参数（数据库查询结果）

常用的SQL语句：

* 查询语句：SELECT 要显示的字段FROM 表名
* 插入语句：INSERT INTO 表名 (字段名1 , 字段名2) VALUES (‘字段名1的值’,‘字段名2的值’)
* 更新语句：UPDATE 表名 SET 字段名1='字段名1的值',字段名2='字段名2的值' WHERE 字段名1= '字段名1的值'
* 删除语句DELETE FROM 表名 WHERE 字段名1= '字段名1的值'`
* 创建数据库: CREATE DATABASE 数据库名
* 创建表: CREATE TABLE wq\_list (

wq\_id INT(6) UNSIGNED NOT NULL AUTO\_INCREMENT PRIMARY KEY,

wq\_title VARCHAR(300) NOT NULL,

wq\_detail VARCHAR(30) NOT NULL,

wq\_idea VARCHAR(30) NOT NULL,

submission\_date TIMESTAMP

)

【作业：用户注册】



【作业：用户登陆】



【思考题】

1. 本次登陆后，下次服务器是否还认识我，我下次是否需要重新登陆？如果需要有什么办法让服务器在下次还能认识我。
2. 登陆和注册重复代码过多，可以提取公共代码吗？require

**Day2网络传输协议**

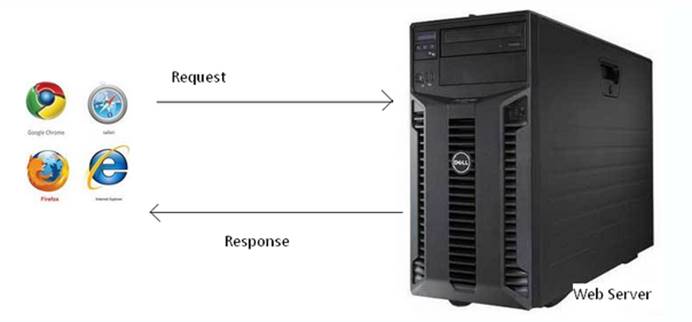
指服务器和客户端间进行通信时的约束和规范，客户端与服务端的数据交互并不是杂乱无章的，需要遵照（基于）一定的规范进行。

**一 HTTP协议**

即超文本传输协议，网站是基于HTTP协议的，例如网站的图片、CSS、JS等都是基于HTTP协议进行传输的。

HTTP协议是由从客户机到服务器的请求(Request)和从服务器到客户机的响应(Response)进行了约束和规范。

即HTTP协议主要由请求和响应构成。



常用请求方法 POST、GET、PUT、DELETE

**1请求/请求报文**

请求由客户端发起，其规范格式为：请求行、请求头、请求主体。

1、请求行

由请求方式、请求URL和协议版本构成

例如：GET /index.php HTTP/1.1

2、请求头

Host：localhost请求的主机

Accept：\*/\* 接受的文档MIME类型

User-Agent：很重要

Referer：从哪个URL跳转过来的

例如：host:www.baidu.com

3、请求主体

即传递给服务端的数据

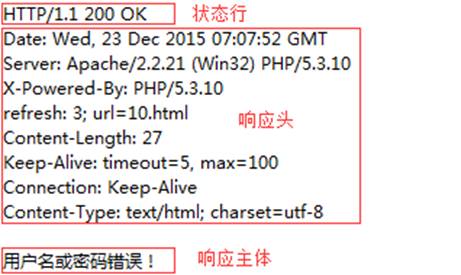
请求头和请求主体之间要空一行

**注：当以post形式提交表单的时候，请求头里会设置**

**Content-Type: application/x-www-form-urlencoded，以get形式当不需要**

**2响应/响应报文**

响应由服务器发出，其规范格式为：状态行、响应头、响应主体。

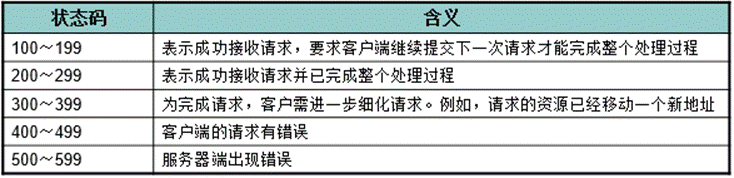


1)状态行

C:\Users\qf\Documents\My Knowledge\temp\3b743a51-0378-4b1e-af53-358b2978f006\128\index_files\clip_image0369bf4605c-1047-4e1c-8e1a-01b3efc31f07.jpg

由协议版本号、状态码和状态信息构成

状态码



常见的有200代表成功、304文档未修改、403没有权限、404未找到、500服务器错误

2)响应头

Content-Length：响应主体长度

Content-Type：响应资源的MIME类型

**MIME是标识文件类型的，文件后缀并不能正确无误的标识文件的类型。**

**作业:百度常见文件的MIME类型?**

**思考:客户端与服务器间传递数据时，是以什么形式传输的？**

**答案:客户端与服务器在进行数据传输的时候都是以字节形式进行的，咱们可以理解成是以文本形式传输，这时浏览器就需要明确知道该怎么样来解析这些文本形式的数据，MIME就是明确告知浏览器该如何来处理。**

3、响应主体

即服务端返回给客户端的内容；

**二 Cookie**

### 1 什么是cookie

\* cookie是每个客户身份的通行证，相当于一个身份证

\* cookie 特点:

\* 文件有大小限制：4KB(文件若没有大小限制，比如身份牌的重量是140斤，挂脖子能不能累死？)

\* 数量限制：（一般浏览器，限制大概在50条左右）

\* 读取有域名限制：不可跨域读取，只能由来自 写入cookie的 同一域名 的网页可进行读取。简单的讲就是，谁写的cookie，谁才有权利读取。

\* 时效限制：每个cookie都有时效，最短的有效期是，会话级别：就是当浏览器关闭，那么cookie立即销毁，如果没有指定Cookie的时效，那么默认的时效是：\*会话级别\*

### 2 使用方法

\* 使用 JavaScript 读取 Cookie :

document.cookie;

\* 使用 JavaScript 设置 Cookie :

document.cookie="username=John Doe; expires=Thu, 18 Dec 2043 12:00:00 GMT";

\* 使用 JavaScript 删除 Cookie:

document.cookie = "username=; expires=Thu, 01 Jan 1970 00:00:00 GMT";

**作业:cookie的封装 (document.cookie)**

**作业:** **cookie模拟登陆**

### 3 cookie和session的区别和用法(熟读并背诵)

\* cookie：在网站中，http请求是无状态的。也就是说即使第一次和服务器连接后并且登录成功后，第二次请求服务器依然不能知道当前请求是哪个用户。cookie的出现就是为了解决这个问题，第一次登录后服务器返回一些数据（cookie）给浏览器，然后浏览器保存在本地，当该用户发送第二次请求的时候，就会自动的把上次请求存储的cookie数据自动的携带给服务器，服务器通过浏览器携带的数据就能判断当前用户是哪个了。cookie存储的数据量有限，不同的浏览器有不同的存储大小，但一般不超过4KB。因此使用cookie只能存储一些小量的数据。

\* session:session和cookie的作用有点类似，都是为了存储用户相关的信息。不同的是，cookie是存储在本地浏览器，而session存储在服务器。存储在服务器的数据会更加的安全，不容易被窃取。但存储在服务器也有一定的弊端，就是会占用服务器的资源，但现在服务器已经发展至今，一些session信息还是绰绰有余的。

\* cookie和session结合使用：

web开发发展至今，cookie和session的使用已经出现了一些非常成熟的方案。在如今的市场或者企业里，一般有两种存储方式：

1. 存储在服务端：通过cookie存储一个session\_id，然后具体的数据则是保存在session中。如果用户已经登录，则服务器会在cookie中保存一个session\_id，下次再次请求的时候，会把该session\_id携带上来，服务器根据session\_id在session库中获取用户的session数据。就能知道该用户到底是谁，以及之前保存的一些状态信息。这种专业术语叫做server side session。

2. 将session数据加密，然后存储在cookie中。这种专业术语叫做client side session。

### 4 掌握cookie 的安全策略

\* Cookie具体工作过程描述如下：

1）Web 客户端通过浏览器向 Web 服务器发送连接请求， 通过 HTTP 报文请求行中的 URL 打开某一 Web页面。

2）Web 服务器接收到请求后，根据用户端提供的信息产生一个 Set-Cookies Head

3）将生成的Set-Cookies Header通过 Response Header存放在 HTTP 报文中回传给 Web 客户端，建立一次会话连接。

4）Web 客户端收到 HTTP 应答报文后，如果要继续已建立的这次会话，则将 Cookies 的内容从 HTTP 报文中取出，形成一个 Cookies文本文件储存在客户端计算机的硬盘中或保存 在客户端计算机的内存中。

5）当 Web 客户端再次向 Web 服务器发送连接请求时， Web 浏览器首先根据要访问站点的 U R L 在本地计算机上寻找对应的 Cookies文本文件或在本地计算机的内存中寻找对应的 Cookies 内容。如果找到，则将此 Cookies 内容存放在 HTTP 请求报文中发给 Web 服务器。

6）Web 服务器接收到包含 Cookies 内容的 HTTP 请求后， 检索其 Cookies 中与用户有关的信息，并根据检索结果生成一个客户端所请求的页面应答传递给客户端 。

\* COOKIE安全分析(**了解即可**)

\* Cookie 泄漏（欺骗）

cookie在http协议中是明文传输的，并且直接附在http报文的前面，所以只要在网络中加个嗅探工具，获取http包，就可以分析并获得cookie的值。

此时，当我们获取到别人的cookie的值，就产生了一种攻击漏洞，即cookie欺骗。我们将获取到的cookie值加在http请求前，服务器就会把我们当作是该cookie的用户，我们就成功的冒充了其他用户，可以使用其他用户在服务器的资源等。

既然明文cookie不安全，那么我们就使用加密传输cookie。这样即使数据包被截取，cookie也不会被泄漏。把http协议改成加密的https。

在HTTPS也不安全的情况下，我们考虑cookie 自身属性。

Cookie 中有个属性secure，当该属性设置为true时，表示创建的 Cookie 会被以安全的形式向服务器传输，也就是只能在 HTTPS 连接中被浏览器传递到服务器端进行会话验证，如果是 HTTP 连接则不会传递该cookie信息，所以不会被窃取到Cookie 的具体内容。就是只允许在加密的情况下将cookie加在数据包请求头部，防止cookie被带出来。

另一个是 HttpOnly属性，如果在Cookie中设置了"HttpOnly"属性，那么通过程序(JS脚本、Applet等)将无法读取到Cookie信息，这样能有效的防止XSS攻击。

secure属性是防止信息在传递的过程中被监听捕获后信息泄漏，HttpOnly属性的目的是防止程序获取cookie后进行攻击。

\* COOKIE防护

1. ”重名检查“:但是这个有难度。因为cookie 时RFC定义的，而修改RFC中的cookie 标准是有困难的。

2. 清理cookie:Cookie是由于便利性而存在的，那么总是在使用后清除cookie，其实就达不到cookie的便利作用了。

3. 不在 Cookies 中存放敏感信息:这是一个 理想化的思路，其实质就是抛弃 Cookies，但明显违背安全平台的设计思路，不能因为 Cookies 可能存在欺骗攻击而废止它的便利性。

#### 三 localStroage/sessionStorage

\* 浏览器的大小不统一，并且在IE8以上的IE版本才支持localStorage这个属性

\* 目前所有的浏览器中都会把localStorage的值类型限定为string类型，这个在对我们日常比较常见的JSON对象类型需要一些转换

\* localStorage在浏览器的隐私模式下面是不可读取的

\* localStorage本质上是对字符串的读取，如果存储内容多的话会消耗内存空间，会导致页面变卡

\* localStorage不能被爬虫抓取到

\* 使用方式:

window.localStorage.setItem()

window.localStorage.clear()

window.localStorage.removeItem("name")

window.localStorage.getItem()

window.localStorage.name

window.localStorage.key(0)

新的事件: 

**Day3 AJAX编程**

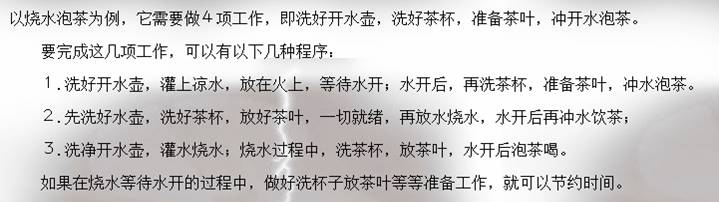
即 Asynchronous Javascript And XML，AJAX 不是一门的新的语言，而是对现有持术的综合利用。

本质是在HTTP协议的基础上以异步的方式与服务器进行通信。

**一 异步**

指某段程序执行时不会阻塞其它程序执行，其表现形式为程序的执行顺序不依赖程序本身的书写顺序，相反则为同步。

其优势在于不阻塞程序的执行，从而提升整体执行效率。





现实生活中的一个例子

打电话时同步 发消息是异步

XMLHttpRequest可以以异步方式的处理程序。

**二 XMLHttpRequest**

浏览器内建对象，用于在后台与服务器通信(交换数据) ，由此我们便可实现对网页的部分更新，而不是刷新整个页面。

下面是一个简单的例子



由于XMLHttpRequest本质基于HTTP协议实现通信，所以结合HTTP协议和上面的例子我们分析得出如下结果：

**三 请求**

HTTP请求3个组成部分与XMLHttpRequest方法的对应关系

1、请求行

C:\Users\qf\Documents\My Knowledge\temp\3b743a51-0378-4b1e-af53-358b2978f006\128\index_files\clip_image050f1990e5f-12c7-410c-83c9-41abfaff7544.jpg

2、请求头

C:\Users\qf\Documents\My Knowledge\temp\3b743a51-0378-4b1e-af53-358b2978f006\128\index_files\clip_image0524a56b307-ea30-4826-a9c0-fd47d0f187f6.jpg

**get请求可以不设置**

3、请求主体

C:\Users\qf\Documents\My Knowledge\temp\3b743a51-0378-4b1e-af53-358b2978f006\128\index_files\clip_image054071b71f8-1ed1-4c72-a87f-13e1655ae279.jpg

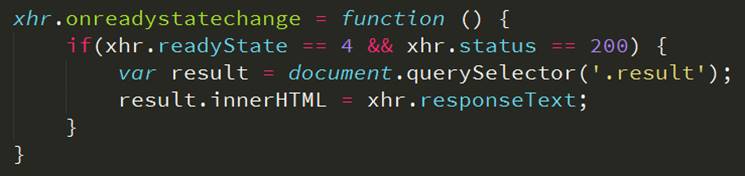
**注意书写顺序**

**四 响应**

HTTP响应是由服务端发出的，作为客户端更应关心的是响应的结果。

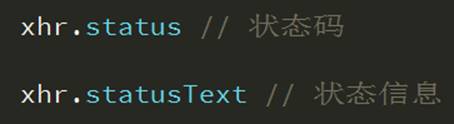
HTTP响应3个组成部分与XMLHttpRequest方法或属性的对应关系。

由于服务器做出响应需要时间（比如网速慢等原因），所以我们需要监听服务器响应的状态，然后才能进行处理。

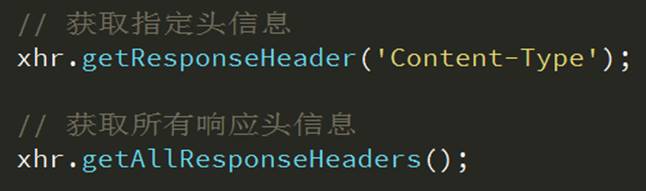


**onreadystatechange是Javascript的事件的一种，其意义在于监听XMLHttpRequest的状态**

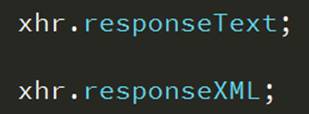
1、获取状态行（包括状态码&状态信息）



2、获取响应头



3、响应主体



我们需要检测并判断响应头的MIME类型后确定使用request.responseText或者request.responseXML

**五 XMLHttpRequest的API详解**

xhr.open() 发起请求，可以是get、post方式

xhr.setRequestHeader() 设置请求头

xhr.send() 发送请求主体get方式使用xhr.send(null)

xhr.onreadystatechange = function () {} 监听响应状态

xhr.readyState = 0时，UNSENT open尚未调用

xhr.readyState = 1时，OPENED open已调用

xhr.readyState = 2时，HEADERS\_RECEIVED 接收到头信息

xhr.readyState = 3时，LOADING 接收到响应主体

xhr.readyState = 4时，DONE 响应完成

不用记忆状态，只需要了解有状态变化这个概念

xhr.status表示响应码，如200

xhr.statusText表示响应信息，如OK

xhr.getAllResponseHeaders() 获取全部响应头信息

xhr.getResponseHeader('key') 获取指定头信息

xhr.responseText、xhr.responseXML都表示响应主体

**作业：详述GET和POST请求方式的差异（面试题）**

1、GET没有请求主体，使用xhr.send(null)

2、GET可以通过在请求URL上添加请求参数

3、POST可以通过xhr.send('name=itcast&age=10')

4、POST需要设置

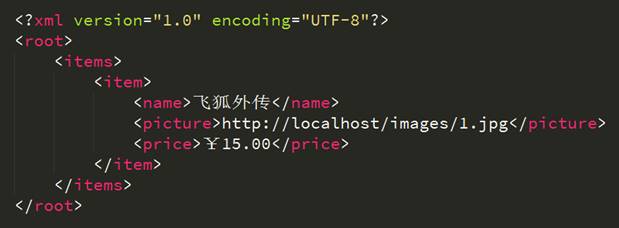
C:\Users\qf\Documents\My Knowledge\temp\3b743a51-0378-4b1e-af53-358b2978f006\128\index_files\clip_image06456150333-ee57-4e88-9dfc-cf44e75eaeac.jpg

5、GET效率更好（应用多）

6、GET大小限制约4K，POST则没有限制

**六 XML(了解即可)**

XML是一种标记语言，很类似HTML，其宗旨是用来传输数据，具有自我描述性（固定的格式的数据）。



**1 语法规则**

1、必须有一个根元素

2、不可有空格、不可以数字或.开头、大小写敏感

3、不可交叉嵌套

4、属性双引号（浏览器自动修正成双引号了）

5、特殊符号要使用实体

6、注释和HTML一样

虽然可以描述和传输复杂数据，但是其解析过于复杂并且体积较大，所以实现开发已经很少使用了。

**2 XML示例**

详见代码

**七 JSON**

即 JavaScript Object Notation，另一种轻量级的文本数据交换格式，独立于语言。

**1 语法规则**

1、数据在名称/值对中

2、数据由逗号分隔(最后一个健/值对不能带逗号)

3、花括号保存对象方括号保存数组

4、使用双引号

**2 JSON解析**

JSON数据在不同语言进行传输时，类型为字符串，不同的语言各自也都对应有解析方法，需要解析完成后才能读取

**1)Javascript 解析方法**

eavl()、JSON对象JSON.parse()、JSON.stringify()；

**2)PHP解析方法**

json\_encode()、json\_decode()

总结：JSON体积小、解析方便且高效，在实际开发成为首选。

**八 XMLHttpRequest兼容性**

IE5、IE6中使用 ActiveObject("Microsoft.XMLHTTP")

如下图



关于IE的兼容方面，了解即可。

**作业:封装AJAX工具函数**

为了提升我们的开发效率，我们自已将XMLHttpRequest封装成一个函数。

**接口化开发**

请求地址即所谓的接口，通常我们所说的接口化开发，其实是指一个接口对应一个功能，并且严格约束了请求参数和响应结果的格式，这样前后端在开发过程中，可以减少不必要的讨论，从而并行开发，可以极大的提升开发效率，另外一个好处，当网站进行改版后，服务端接口只需要进行微调。

**Day4**

**1 同源**

同源策略是浏览器的一种安全策略，所谓同源是指，域名，协议，端口完全相同。

**2 跨域**

不同源则跨域

例如http://www.example.com/

|  |  |  |
| --- | --- | --- |
| http://api.example.com/detail.html | 不同源 | 域名不同 |
| https//www.example.com/detail.html | 不同源 | 协议不同 |
| http://www.example.com:8080/detail.html | 不同源 | 端口不同 |
| http://api.example.com:8080/detail.html | 不同源 | 域名、端口不同 |
| https://api.example.com/detail.html | 不同源 | 协议、域名不同 |
| https://www.example.com:8080/detail.html | 不同源 | 端口、协议不同 |
| http://www.example.com/detail/index.html | 同源 | 只是目录不同 |

**3 跨域方案(了解)**

1)顶级域名相同的可以通过domain.name来解决，即同时设置 domain.name = 顶级域名（如example.com）

2)document.domain + iframe

3)window.name + iframe

4)location.hash + iframe

5)window.postMessage()

**4 JSONP(重点掌握)**

JSON with Padding

**原理剖析**

其本质是利用了<script src=""></script>标签具有可跨域的特性，由服务端返回一个预先定义好的Javascript函数的调用，并且将服务器数据以该函数参数的形式传递过来，此方法需要前后端配合完成。

只能以GET方式请求

Day5 Bootstrap(见官方文档)