**PHP与ajax复习**

### **1-表单提交**

**前端页面**：

action :

method :

name：

enctype: multipart/form-data

**后面处理表单常用的超全局变量**

$\_GET：

$\_POST ：

$\_FILES ：

$\_SESSION ：

$\_COOKIE ：

### **2-HTTP协议**

1.**请求 request**

请求行 请求方式 地址 协议版本

请求头 记录了浏览器相关信息

请求主体 post提交的数据

1. **响应 response**

状态行 协议版本 状态码 状态文本

响应头 记录了服务器相关信息

响应主体： 服务器发送给浏览器数据

**常见状态码：** 200 成功 302 页面重定向404 页面未找到 500 服务器错误

**3.get请求和post请求的区别 put delete**

1- get会在url后面拼接数据 （默认情况下都是get请求）

get 不安全 数据量小 4k 速度快

get请求请求主体为空

2-post

相对安全，post对上传文件大小没有限制

post有请求主体

### **3-sql 语法**

1.插入 insert into 表名 （字段。。。） values (值， 值)

2.删除 delete from 表名 where id = $id;

3.修改 update 表名 set 字段 = 值 ；

4.查询 select 字段列表 from 表名 where 条件

5.where 的用法​

and or ! 与或非

id in (3,5,9) 一对多匹配

模糊匹配

where name like '%春%';

select count(\*) as total from 表名；

1. 表的截取(分页) limit 起始索引， 截取长度

起始索引 = (当前页面 - 1）\* 每页数据条数

7 .排序 order（升序） by id desc 降序

1. 联合查询 select 字段列表 from 表 A join 表B on 关联条件 where 条件

### **4 - PHP操作mySql**

1.连接数据库 mysqli\_connect();

2.准备sql(外双内单)

3.执行sql mysqli\_query() 增删改 true false 查询 结果集 false

（1）显示最近的错误信息 mysqli\_error();

（2）获取结果集的行数 mysqli\_num\_rows($res);

（3）读取结果集中数据

while ( row = mysqli\_fetch\_assoc($res) ) {

​ $data[] = row

4.关闭数据库连接 mysqli\_close();

### **5 - cookie 和Session**

cookie 浏览器端存储数据的容器

操作cookie的常用方法，及cookie的特点：

$.cookie(k,v); 设置

$.cookie(k)； 获取

$.cookie(k, v , {expires: 1}) 设置有效期 设置有效期为负数 就会自动删除

php 操作cookie:

setcookie(k, v, 有效期) 设置cookie

$\_COOKIE超全局变量 只能获取cookie

特点：

1- 容量小 4Kb左右

2- 被同一个网站（同源）的多个页面共享数据

3- 每次请求服务器，浏览器会自动将cookie添加到请求报文中，发送给服务器

4- 服务器可以设置响应报文，在响应报文中setCookie = name =zs 进行设置，告诉浏览器，浏览器接受到响应报文后，会自行去设置cookie

ession 服务器端存储数据的容器

操作session的常用方法，及session的特点：

**使用之前先开启 session\_start();**

$\_SESSION['name'] = 'zs';

unset($\_SESSION['name']);

$\_SESSION= [];

**session\_start做的三件事：**

1-动态生成sessionID

2-创建一个session用于存储数据，会以sessionID进行命名

3-通过响应报文让浏览器的cookie中存储这个sessionID

session 中数据可以被服务器中多个页面共享；

cookie 和 session 配合 实现登录状态保持

1- 在用户登录成功时，

在服务器中记录一个秘钥，这个秘钥会给到浏览器cookie , 在session中记录登录用户信息

session\_start();

$\_SESSION['user\_id'] = $data['id'];

2-下次判断用户是否登录：

1- 判断该用户浏览器中是否有秘钥，如果没有去登录

2- 根据用携带sessionID去找对用应session文件，判断里面数据是否存在；

### **6 - 原生Ajax(XMLhttpRequest对象)**

var xhr = new XMLHttpReqeust(); //创建一个可以发送ajax请求实例

//设置http请求报文

xhr.open('get', '0.php');

xhr.setReqeustHeader('content-type', 'text/html'); // application/x-www-form-urlencoded

xhr.send(null);

//监听服务器

xhr.onreadystatechange = function () {

if （xhr.readyState == 4 && xhr.status == 200 ) {

var r = xhr.responsText; //获取响应主体

//JSON.parse();

}

}

### **7 - $.ajax()**

$.ajax({

type: 'get',

url: '0.php',

data: {name: 'zs', age: 18}, // name=zs&age=18

dataType: 'json', // 设置为json 会自动调用JSON.parse(); xml text

timeout: 1000, //超时 毫秒

beforeSend: function () {}, //请求发送之前逻辑

success:function () {}, //成功响应后逻辑

error: function () {}, //请求出错是执行

complete: function () {} //请求完成时一定执行的回调函数

})

//用return false 终止请求

### **8 - 模板引擎 (template-web.js)**

**使用步骤**：

1- 引入插件

2- 准备模板

3- 准备数据

4- 将数据和模板组装生成结构（html)

**注意：**

1- 数据必须是对象 （不是对象就包装成对象）

2- 模板中直接使用对象的属性

语法：

将js逻辑代码放在{{ }}中

遍历

{{ each list v i }}

{{ /each }}

判断

{{ if num >= 100}}

{{ else if num < 50 }}

{{ /if }}

### **9 - XMLHttpRequest2.0**

//1-超时

xhr.timeout = 2000,

xhr.ontimeout = function () { alert('超时')};

//2-formData

FormData 管理表单数据对象

var formData = new FormData( form );

xhr.open('post', '0.php');

xhr.send(formData);

//注意点： 1必须是用post方式 2-不能设置请求头

表单序列化 $('form').serialize() 可以快速将表单数据 拼成 name=zs&age=18&sex=m

$.ajax({

type: 'post',

data: formData,

contentType: false, //不用设置请求头了

processData: false, //不用在处理数据

});

//3-上传文件的进度

xhr.upload.onprogress = function (e) {

//获取文件进度

var value = e.loaded / e.total ; //比值就进度

}

### **10 - JSONP的原理**

1-如果跨域 前端解决办法

$.ajax({

dataType: 'jsonp'

})

jsonp : json with padding 用json数据进行填充

**本质**： 利用script标签src属性进行的跨域

**实现步骤**：

1- 前端要先定义好一个方法，让利用script的src 将方法名字传给后台 callback=say

2-后台获取方法名，在方法名后拼接括号，并且在方法括号中填充json数据,并返回

echo say({"name":"zs"})

3-前端将返回字符串 当做js执行 ,就可以获取方法参数中数据；

<script src="01.php?callback=say"></script>

1-jsonp要求前后daunt配合

2-jsonp只能发送get请求

### **11- 浏览器向服务器发送请求的过程**

1-如果请求的是静态文件，服务器直接返回，交给浏览器进行解析

2-如果请求的是php文件（后台处理程序）， 先在服务器进行执行，在服务器中只执行php标签内部代码，执行完成后，会将执行的结果和 php标签外面代码一起返回给浏览器，在由浏览器进行解析；

### **12 -常用的各种插件**

1-分页： jquery.pagination.js

2-时间格式化： moment.js moment(时间).format('YYYY-MM-DDTHH:mm')

3-富文本编辑器：wangEditor.js

### **13 - 其他知识**

1-当用户输入的事件：

box.oninput = function () {}

表单状态：

input.onchange = function () {}

2-控制文件上传的类型：

accpet = "image/\*"

3-图片本地预览：

input.onchange = function () {

var file = this.files[0]; //文件

var url = URL.createObjectURL(file);

img.src = url;

}

4-表单重置：

form.reset(); ==== 》 DOM方法

5-常见的jq表单状态选择器（可用，禁用，被选中）：

$('input:checked')

$('input:disabled')

$('input:enabled')

6-jq对象和DOM对象的区别：

jq: $('div') [div,div,div,div]

dom: div

7-事件委托的使用场景：

给动态生成的子元素绑定事件

$('存在父元素').on(事件类型, 子元素, function () {})

8-如果拓展jq插件：

给$.fn 添加方法即可

$.fn.waterFall = function (){}

$('div').waterFall();