## **一．服务器环境的搭建**

1. 安装wampserver 2.1a 多语言集成包 32位(F:\wamp)，将站点文件放在www文件夹下(F:\wamp\www)。
2. 双击图标，打开服务器—单机桌面右下角图标—点击www directory可以进入www文件夹。
3. 地址栏输入127.0.0.1或者localhost访问服务器网页

注意:(1)双击www文件夹中的html文件，访问的仍然是本地文件，而不是服务器页面。(2)为避免端口冲突，在运行服务器时，其他其他软件(迅雷等)应关闭。

## **二．第一个ajax程序**

1. Ajax—Asynchronous JavaScript and XML(异步JavaScript和XML)

用JavaScript异步的形式去操作XML

注意: Ajax用来数据交互。(类似于浏览网页时用户在浏览器中输入地址，向服务端发送请求，请求文件，完成数据交互。)

优点:节约用户操作，时间，提高用户体验，减少数据请求。

传输获取数据。

1. 第一个Ajax程序

<input type="button" id="btn" value="按钮">

<script>

window.onload=function(){

var oBtn=document.getElementById('btn');

oBtn.onclick=function(){

//打开浏览器

//1. 创建ajax对象

var xhr=new XMLHttpRequest();

//在地址栏中输入地址

//2.open方法参数:(1)打开方式;(2)地址;(3)是否异步;

xhr.open('get','1.txt',true);

//回车提交

//3. 发送请求

xhr.send();

//4.等待服务器返回内容

//readyState:ajax工作状态

//status服务器状态码

//responseText:ajax请求返回的内容就被存放到这个属性下面

xhr.onreadystatechange=function(){

if(xhr.readyState==4){

if(xhr.status==200){

alert(xhr.responseText);

}else{

alert('出错了，Err：'+xhr.status);

}

}

}

}

}

</script>

注意:使用Ajax进行数据交互一定要在服务器环境中，本地html页面无法访问后台数据。

## **三. 对象的创建和兼容处理**

1. 第一步：创建ajax对象

var xhr=new XMLHttpRequest();

1. 在IE6及IE6一下，new XMLHttpRequest()未定义。
2. 兼容IE6

<input type="button" id="btn" value="按钮">

<script>

window.onload=function(){

var oBtn=document.getElementById('btn');

oBtn.onclick=function(){

var xhr=null;

if(window.XMLHttpRequest){

xhr=new XMLHttpRequest();

}else{

xhr=new ActiveXObject('Microsoft.XMLHTTP');

}

xhr.open('get','1.txt',true);

xhr.send();

xhr.onreadystatechange=function(){

if(xhr.readyState==4){

alert(xhr.responseText);

}

}

}

}

</script>

1. try catch异常捕获

<script>

alert(a);//报错

alert(2);//不会弹出2

</script>

代码尝试执行try块中的内容，如果有错误，则会执行catch{},并且传入错误信息参数。

<script>

try{

alert(a);

}catch(e){

alert(1);//弹出ReferenceError: a is not defined

}

alert(2);//弹出2

</script>

1. 使用try catch方法兼容XMLHttprequest()对象

try{

xhr=new XMLHttpRequest();

}catch(e){

xhr=new ActiveXObject('Microsoft.XMLHTTP');

}

## **open方法和表单**

1. 第二部：open方法

xhr.open('get','1.txt',true);

open方法的参数:(1)打开方式;(2)地址;(3)是否异步;

1. 表单：数据提交

属性：action=数据提交的地址，默认是当前页面

method=数据提交的方式，默认是get方式

content-type:提交的数据格式，默认

'application/x-www-form-urlencoded'

注意: form表单提交与Ajax中的get提交方式无需设置content-type(声明发送数据的编码类型)；而Ajax中的post提交方式需要设置content-type(声明发送数据的编码类型)。

1. 表单method提交方式

(1)get;

把数据名称和数据值用=连接，如果有多个数据，则把多个数据组合用&连接，放到url?后传到指定页面。

例：http://localhost/miaov/ajax-1/1.get.php?username=jake&age=20

注意:由于url长度限制的原因(IE浏览器小于2000多个字符，不同浏览器存在差异)，不能通过get方式传递过多的数据。

(2)post

传递数据大小理论上无限制。

1. 表单数据提交get方式举例

ajax.html页面代码:

<form action="1.get.php">

<input type="text" name="username">

<input type="text" name="age">

<input type="submit" value="提交">

</form>

1. get.php页面代码:

<?php

header('content-type:text/html;charset="utf-8"');

error\_reporting(0);

$username=$\_GET['username'];

$age=$\_GET['age'];

echo "你的名字{$username},年龄{$age}";

?>

过程: 输入用户名和年龄并提交后，跳转到1.get.php页面，该页面输出“你的名字leo,年龄30”。

注意:①method属性默认为get方式。②发生数据后1.get.php页面地址为

http://localhost/miaov/ajax-1/1.get.php?username=jake&age=20

1. 表单数据提交post方式举例

ajax.html页面代码:

<form action="1.post.php" method="post">

<input type="text" name="username">

<input type="text" name="age">

<input type="submit" value="提交">

</form>

1.get.php页面代码:

<?php

header('content-type:text/html;charset="utf-8"');

error\_reporting(0);

//$\_REQUEST

$username=$\_POST['username'];

$age=$\_POST['age'];

echo "你的名字{$username},年龄{$age}";

?>

过程: 输入用户名和年龄并提交后，跳转到1.get.php页面，该页面输出“你的名字leo,年龄30”。

注意: ①数据提交方式(get/post)与后台接收数据的方式需要对应。② 1.post.php页面地址不变。

1. 第三个参数：是否异步xhr.open('get','1.txt',true);

true为异步，false为同步。

注意:异步不阻塞下面的代码执行。

## **数据的获取**

1. html页面代码

<input type="button" id="btn" value="按钮">

<script>

window.onload=function(){

var oBtn=document.getElementById('btn');

oBtn.onclick=function(){

var xhr=null;

try{

xhr=new XMLHttpRequest();

}catch(e){

xhr=new ActiveXObject('Microsoft.XMLHTTP');

}

xhr.open('get','getlist.php',true);

xhr.send();

xhr.onreadystatechange=function(){

if(xhr.readyState==4){

if(xhr.status==200){

alert(xhr.responseText);

}else{

alert('出错了，Err：'+xhr.status);

}

}

}

}

}

</script>

getlist.php页面代码

<?php

header('content-type:text/html;charset="utf-8"');

error\_reporting(0);

$arr1=array('leo','momo','zhangsen');

$arr2=array('username'=>'leo','age'=>32);

echo json\_encode($arr1);

//echo json\_encode($arr2);

?>

结论：后台输出arr1，点击按钮弹出['leo','momo','zhangsen']；若后台输出arr2，则点击按钮弹出{'username':'leo','age':32}。

## **获取数据的处理和实例**

1. Ajax通过new XMLHttpRequest()对象的responseText属性获取后台发送的数据，数据类型是String。

JSON.stringify(对象)方法可以把一个对象转化为对应的字符串。

JSON.parse(字符串)可以把字符串转化为对象。

注意：对象有两种，一种是JSON类型数据，一种是字符串类型数据。

1. JSON数据类型中的键名必须是双引号

<script>

var str1="{left:100}";

var str2="{'left':100}";

var str3='{"left":100}';

// alert(JSON.parse(str1)); 报错

// alert(JSON.parse(str2)); 报错

alert(JSON.parse(str3));//弹出[object Object]

</script>

1. 定时刷新与Ajax函数的简单封装

Html页面代码:

<input type="button" id="btn" value="按钮">

<ul id="ul1"></ul>

<script>

function ajax(method, url, data, success) {

var xhr = null;

try {

xhr = new XMLHttpRequest();

} catch (e) {

xhr = new ActiveXObject('Microsoft.XMLHTTP');

}

if (method == 'get' && data) {

url += '?' + data;

}

xhr.open(method,url,true);

if (method == 'get') {

xhr.send();

} else {

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

xhr.send(data);

}

xhr.onreadystatechange = function() {

if ( xhr.readyState == 4 ) {

if ( xhr.status == 200 ) {

success && success(xhr.responseText);

} else {

alert('出错了,Err：' + xhr.status);

}

}

}

}

</script>

<script>

window.onload=function(){

var oBtn=document.getElementById('btn');

oBtn.onclick=function(){

ajax('get','getNews.php','',function(data){

var data=JSON.parse(data);

var oul=document.getElementById('ul1');

var html='';

for(var i=0;i<data.length;i++){

html+="<li><ahref=''>"+data[i].title+"</a><span>"+data[i].date+"[</span></li>";

};

oul.innerHTML=html;

})

setInterval(function(){

ajax('get','getNews.php','',function(data){

var data=JSON.parse(data);

var oul=document.getElementById('ul1');

var html='';

for(var i=0;i<data.length;i++){

html+="<li><a href=''>"+data[i].title+"</a><span>"+data[i].date+"[</span></li>";

};

oul.innerHTML=html;

})

},1000);

}

}

</script>

Php页面代码

<?php

header('content-type:text/html;charset="utf-8"');

error\_reporting(0);

$news=array(

array('title'=>'田径首金即创世界纪录', 'date'=>'2016-8-12'),

array('title'=>'王军霞23年前万米纪录被破','date'=>'2016-8-12'),

array('title'=>'林丹2-0横扫小组两连胜','date'=>'2016-8-12'),

array('title'=>' 徐晨/马晋遭逆转告负','date'=>'2016-8-12'),

array('title'=>'中国女排0-3完败塞尔维亚','date'=>'2016-8-12'),

array('title'=>'中国游泳选手药检呈阳性','date'=>'2016-8-12'),

);

echo json\_encode($news);

?>

## **应用中get和post的区别处理**

1. Get方式进行数据交互

Html页面

xhr.open('get','2.get.php?username='+encodeURI('刘伟')+'&age=30&'+new Date().getTime(),true);

xhr.send();

Php页面

$username = $\_GET['username'];

$age = $\_GET['age'];

echo "你的名字：{$username}，年龄：{$age}";

注意:①**get方式如何向后台发送数据:多个数据组合用&连接，放到url?后**。

②缓存问题:发送字符串尾部加上**&**时间戳。(&不可以省略)

③中文乱码问题：encodeURI进行编码。

④与form表单提交类似，后台php页面的键与html发送数据的键要一致。

⑤后台返回的数据与后台页面用浏览器直接打开的页面显示一致。

⑥不管使用form表单提交数据还是用ajax数据交互，后台书写格式类似。

⑦form表单提交数据会刷新页面，ajax可以实现局部刷新、异步加载。

问题: 使用form表单提交数据，如何接收后台返回的数据？

1. Post方式进行数据交互

Html页面

xhr.open('post','2.post.php',true);

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

xhr.send('username=刘伟&age=20');

Php页面

$username = $\_POST['username'];

$age = $\_POST['age'];

echo "你的名字：{$username}，年龄：{$age}";

注意:①post方式通过请求头发送数据，需要设置请求头发送数据的编码类型。

②post方式，需要发送的数据放在send()里面作为参数传递。

③get方式与post方式数据的拼接格式相同。

④post方式并不存在缓存问题。(post方式是为了向服务器提交数据，提交数据并不存在缓存问题)

⑤与get方式不同，post方式在设置编码类型后，就不再有编码问题。

## **瀑布流布局原理**

1. 瀑布流的特点:①列宽相等；②分批加载；

瀑布流的分类:①固定列数；②不固定列数；