ajax学习笔记

js代码中，通过 obj.setAttribute('src','./zmh.jpg');

为标签添加属性

file\_get\_contents();

file\_put\_contents();

document.createElement('img');创建一个标签

document.getElementById('id').appendChild('child');

把子元素添加到父元素中

Http 204 无内容 可以实现 无刷新效果

注意：页面无刷新效果可以用多种方法实现，Ajax只是其中的一种方法

注意：虽然Http协议没有限制URL的长度，但是浏览器和服务器对URL的长度是是有限制的

所以更多使用post来传递数据

Window对象、Parent对象、Frame对象、Document对象和Form对象的阶层关系：Windwo对象→Parent对象→Frame对象→Document对象→Form对象

远古时代的Ajax，主要是想实现 页面没有刷新的状态，但是还是在后台发送了Http请求

有三种方法可以实现这种效果

第一：利用Http协议头信息 204 no contents

<?php

$cnt = file\_get\_contents("./vote.txt");//这个文件必须事先，如果有内容，他会自动清开

$cnt+=1;

file\_put\_contents("./vote.txt", $cnt);

header("Http/1.1 204 no contents");

?>

第二：利用image标签 结合js

function add(){

var img = document.createElement('img');

img.setAttribute("src","./demo1.jpg");

document.getElementById('div').appendChild(img);

}

只要标签后面有链接，就会向服务器请求资源 比如a img

第三：使用iframe框架，把需要刷新的内容放在框架内，页面其他部分不刷新

<form action="demo3.php" method="post" target="regzone">

<p>用户名：<input type="text" name="username"></p>

<p>密码：<input type="password" name="password"></p>

<p><input type="submit" value="提交"></p>

</form>

<!-- 这是一个内联框架标签，name值与表单或者超链接的target属性值相对应 -->

<iframe frameborder="1" name="regzone"></iframe>

<!-- 以下是把超链接指向的页面显示在iframe框架内 -->

<a href="demo1.html" target="demo1">显示页面一</a>

<iframe src="" name="demo1" frameborder="0" width="500px" height="600px"></iframe>

以上例子说明：无刷新只是视觉上的效果，本质上还是要通过某种方式发送http请求，比如 Http 204 js动态添加图片(img src) 用框架iframe将两个网页隔开，也就是将刷新和不刷新部分隔离开

Ajax快速入门案例

核心XMLHttpRequest对象 简称XHR 他主要用于代替浏览器发送和接收htt请求

XHR就是介于浏览器和服务器之间的第三者，所有浏览器与服务器之间的通信都要经过 xhr 他既可以发请求，也可以接收请求 通过js操作dom的方法实现取出xhr里面所带回来的内容，使他呈现在浏览器页面上

js的网络化

响应的格式有 xml 和 json 两种主流格式

var variable=new XMLHttpRequest();

老版本的 Internet Explorer （IE5 和 IE6）使用 ActiveX 对象：

var variable=new ActiveXObject("Microsoft.XMLHTTP");

false 表示同步

true 表示异步

同步和异步的区别：

send();

xxx 普通代码

只要体现在以上两句的执行顺序，如果是同步，要等服务器响应结束后才执行send后面的普通代码，如果是异步，不用等到服务器响应，直接马上就可以执行xxx普通代码

但是如何让浏览器知道服务器已经返回了响应信息呢？

xhr对象有一个 readyState 属性，他有多个状态值

对象.responseText就代表返回的文本

status 获取状态码

statusText 获取状态码所对应的文本

getResponseHeader("content-type")

getAllResponseHeaders()

setRequestHeader("Content-type:application/x-form-www-urlencoded");

设置请求头部信息

responseXML属性，表示接收回来的XML格式的文本

responseText属性 响应的文本信息，也就是PHP中用echo输出的部分

post数据详细解释

主要是要设置好content-type:application/x-form-www-urlencoded

注意:请求完成，未必表示请求成功了，比如请求一个本身不存在的URL地址，服务器返回一个错误信息，这就表明 请求成功并且还返回了信息 但不是我们想要的信息，所以说请求并未成功，仅仅是请求完成

所以在响应的回调函数中

xhr.onreadystatechange=function(){

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

}

}

只要状态有变化，这个函数就会被调用 状态有0-4之间变换，if只是起到过滤的作用

头信息在服务器之间的交互式很重要的

暂停、中断与服务器之间的交互用 xhr.abort();方法，可以实现大文件上传的中断操作，以便续传

表单如果不写提交方法，默认是以 get方式发送的 所以一般要声明 method='post'

xhr.setRequestHeader("Content-type","application/x-www-form-urlencoded");

利用Ajax注册无刷新提交

<script>

//把表单的默认提交动作屏蔽，然后利用Ajax后台提交

var xhr = new XMLHttpRequest();

// console.log(xhr);

function ajaxreg(){

xhr.open("POST","demo5.php",true);

//下面这句很重要

xhr.setRequestHeader("Content-type","application/x-www-form-urlencoded");

xhr.onreadystatechange = function(){

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

var text = xhr.responseText;

console.log(text);

}

}

var username = document.getElementsByName('username')[0].value;

var pwd = document.getElementsByName('pwd')[0].value;

// alert('username='+username+'&pwd='+pwd);

//以键值对形式传参，与地址栏中get方式传参一样

xhr.send('username='+username+'&pwd='+pwd);

return false;//阻止表单自动提交到指定页面

}

在HTML表单中 onsubmit = "return ajaxreg()"

<!-- 函数的返回值是false，这里相当于return false 所以不会自动提交到action中指定的页面 -->

可以到网上引用别人提供的jQuery库，这样可以节省不少流量

http://code.jquery.com/jquery-latest.js

返回值类型：

XML类型：xhr.responseXML;属性

普通文档：responseText 属性

作为普通文档返回又有以下几种变通形式：

1.后台返回大段的HTML代码，在js代码中直接用 对象.innerHTML= xhr.responseText;即可瞬间生成丰富的页面，例如在后台通过一个foreach循环生成一个table或者列表

2.返回的文档格式正好符合js能识别的js对象，json

如果返回的是json格式，需要用括号引起来，表示是一个表达式

例如：var json = eval('('+xhr.responseText+')');

这样就可以直接通过操作json对象，用 . 的方式很方便取出返回来的各种数据

在后端如何组织json格式的字符串

在PHP中有 json\_encode($array);可以将一个关联数组直接转换成json格式的字符串

相反：也有json\_decode($json\_stringt,true);可以将一个json格式的字符串直接转化成关联数组,当该参数为 TRUE 时，将返回 array 而非 object 。

Ajax跨域：

何为跨域：

百度的jsonp接口

https://sp0.baidu.com/5a1Fazu8AA54nxGko9WTAnF6hhy/su?wd=Apple&json=1&p=3&cb=fruit&t

jsonp压根就不是现代意义上的Ajax，他并没有new对象，只是利用在你的页面内，动态创建一个script标签，利用这个标签的src属性中的URL地址 来间接向第三方服务器发请求，然后对方给你返回一段类似json的对象，包含很多有用信息，最后在你的js中通过URL中指定的回调函数取出这个对象 function callback(res) {

console.log(res);

}

<!DOCTYPE html>

<html lang="en">

<head>

<title>ajax跨域</title>

<script type="text/javascript">

function sear() {

var kw = document.getElementsByName('keyword')[0].value;

var url = 'https://sp0.baidu.com/5a1Fazu8AA54nxGko9WTAnF6hhy/su?wd='+kw+'&json=1&p=3&cb=test';

//创建script标签

var scr = document.createElement('script');

//用setAttribute(key,value);的方式设置属性值

scr.setAttribute('src',url);

//下面与img标签不同，必须要appendChild()

document.getElementsByTagName('body')[0].appendChild(scr);

}

function test(res) {

// console.log(res);

var results = res.s;//取出返回结果中我们所需要的信息

var szone = document.getElementById('szone');

var str = '';//循环之前定义一个空字符串进行拼接信息

for(var i in results) {

//alert(i);

str += '<p>'+ results[i] + '<p/>';

}

//将拼接好的信息一次性塞到HTML标签中

szone.innerHTML = str;

}

</script>

</head>

<body>

<input type="text" name="keyword" /><br />

<input type="button" value="搜索" onclick="sear();" />

<div id="szone">

</div>

</body>

</html>

异步原理

readyState 存有 XMLHttpRequest 的状态。从 0 到 4 发生变化。

0: 请求未初始化 对象刚刚创建时

1: 服务器连接已建立 open()成功后，为1 也就是建立连接成功

2: 请求已接收 接收响应头信息为2

3: 请求处理中 接收body主体信息为3

4: 请求已完成，且响应已就绪 成功结束后为4

js就是单线程的 异步非阻塞理解：在js中有执行队列的概念，一般情况下，代码由上至下按顺序执行，当采用了异步，有个回调函数，程序始终不停的执行，直到有回调函数时，马上插队 执行 。如果是同步则会出现停滞阻塞后面代码的执行

Ajax最大的好处就是异步不阻塞，比如一个登陆页面，在你登陆验证时，时间比较长，采用Ajax异步到服务器验证，同时不阻塞下面页面的加载，用户体验好

iframe文件上传

<script type="text/javascript">

/\*

分析:

1:捕捉表单提交的动作

2:创建一个iframe

3:把表单的 target修改 指向 该iframe

4:去掉这个iframe

\*/

function ajaxup() {

var ifname = 'up' + Math.random();

$('<iframe name="' + ifname + '" width="0" height="0" frameBorder="0"></iframe>').appendTo($('body'));

$('form:first').attr('target',ifname);

$('#progress').html('<img src="./loading.gif" />');//点击提交后触发函数，图片一直在转，当上传完成，服务器返回一段js代码改变了这个div

//return false;

}

</script>

//如果表单中有文件上传，必须在form中加 enctype="multipart/form-data"

<form action="up.php" method="post" enctype="multipart/form-data" onsubmit="return ajaxup();">

<p>

<input type="file" name="pic" />

</p>

<p>

<input type="submit" value="提交" />

</p>

</form>

如果文件上传成功，$\_FILES['file']['error']=0

$error = $\_FILES['pic']['error']==0 ? 'succ' : 'fail';

FormData 表单数据

var fd = new FormData($form);自动打包表单对象的 name value,并且可以在超全局数组$\_POST 中取出值

xhr.send(fd);

还不用加 xhr.setRequestHeader();

fd.append('key','value');新增表单不存在的键值对

打印上传文件的信息

Array ( [pic] => Array (

[name] => 2017年湛江东兴公司招聘面试点按排 (2).xlsx

[type] => application/vnd.openxmlformats-officedocument.spreadsheetml.sheet

[tmp\_name] => C:\Windows\php6DC1.tmp

[error] => 0 //表示上传成功

[size] => 14641 )

)

只有真正使用xhr对象提交表单才要设置 return false; 来阻止自动提交，其他的用iframe，确实是真正提交到另一个网页，只是这个网页右显示在这个窗口

问题：如何解决文件上传，文件名乱码问题呢？

xhr.open('POST','formdata.php',true);

//必须要打开连接才能设置请求头

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

无法取得表单的formdata，问题是出现在jQuery中，不能直接$('form:first'),不存在这样的写法 要用原生的方法 document.getElementById()

总结：如果是用submit来提交表单，而又要实现无刷新效果，必须要在js中return false;

如果是 表单中 内部的 button 绑定一个函数提交，则不用在js

函数中设置 return false

你还可以使用jQuery来发送FormData,但必须要正确的设置相关选项:

var fd = new FormData(document.getElementById("fileinfo"));

fd.append("CustomField", "This is some extra data");

$.ajax({

url: "stash.php",

type: "POST",

data: fd,

processData: false, // 告诉jQuery不要去处理发送的数据

contentType: false // 告诉jQuery不要去设置Content-Type请求头

});

HTML5 file API

在PHP中可以方便的通过超全局数组 $\_FILES 来取得通过表单post过来的文件，包括二进制文件 multipart/form-data

html5更是给我们提供了files接口，只需要在js中取得file表单对象，通过

file.files;返回filelist 文件列表 是一个数组，数组每个元素表示一个文件

filelist[0]就是文件对象

可以取得 name size lastModified type

可以把获取到的文件 append到formdata中，进行文件上传

bool move\_uploaded\_file ( string $filename , string $destination )

举例:move\_uploaded\_file($\_FILES['pic']['tmp\_name'],路径字符串)

上传文件 后端PHP处理思路：先判断有没有上传 if(empty($\_FILES)){exit('no file select!');}

再判断上传失败 if($\_FILES['pic']['errno']!=0){

exit('fail);

}

走到这里说明成功，返回信息给前端页面

echo "<script>document.getElementById(ss).innerHTML='".$info.'</script>";

图片预览效果

img.src = window.URL.createObjectURL(blob);

把二进制对象直接读成浏览器能识别的资源路径

Ajax文件上传进度条

可以用 xhr2 标准的 xhr.upload.onprogress = function(ev){}

ev.loaded ev.total

这个函数会在上传的过程中不断被触发，ev记录的是整个上传过程中各个瞬间鼠标位置，时间，上传进度...信息

num.toFixed(2)控制一个数字四舍五入后的位数

HTML5中对跨域的新定义

？为什么可以上传文件到对方服务器

能否跨域取决于对方的应答，如果对方的服务器文件设置了

access-Control-Allow-Origin:\* baidu.com

或者在接收请求的PHP文件中设置了响应头

header ( "Cache-Control: no-cache, must-revalidate" ); // HTTP/1.1

header ( "Expires: Sat, 26 Jul 1997 05:00:00 GMT" );

header ( "Location: http://www.example.com/" ); /\* Redirect browser \*/

说明

void header\_remove ([ string $name ] )

Removes an HTTP header previously set using header() .