使用header方式实现文件下载功能

1. PHP发送http header

header(string,replace,http\_response\_code)

|  |  |
| --- | --- |
| string | 必需。规定要发送的报头字符串。 |
| replace | 可选。指示该报头是否替换之前的报头，或添加第二个报头。  默认是 true（替换）。false（允许相同类型的多个报头）。 |
| http\_response\_code | 可选。把 HTTP 响应代码强制为指定的值。（PHP 4 以及更高版本可用） |

* 1. 常见的header功能

header('HTTP/1.1 200 OK'); // ok 正常访问

header('HTTP/1.1 404 Not Found'); //通知浏览器 页面不存在

header('HTTP/1.1 301 Moved Permanently'); //设置地址被永久的重定向 301

header('Location: http://www.test.con/'); //跳转到一个新的地址

header('Refresh: 10; url=http://www.test.con/'); //延迟转向 也就是隔几秒跳转

header('X-Powered-By: PHP/7.0.0'); //修改 X-Powered-By信息

header('Content-language: en'); //文档语言

header('Content-Length: 1234'); //设置内容长度

header('Last-Modified: '.gmdate('D, d M Y H:i:s', $time).' GMT'); //告诉浏览器最后一次修改时间

header('HTTP/1.1 304 Not Modified'); //告诉浏览器文档内容没有发生改变

###内容类型###

header('Content-Type: text/html; charset=utf-8'); //网页编码

header('Content-Type: text/plain'); //纯文本格式

header('Content-Type: image/jpeg'); //JPG、JPEG

header('Content-Type: application/zip'); // ZIP文件

header('Content-Type: application/pdf'); // PDF文件

header('Content-Type: audio/mpeg'); // 音频文件

header('Content-type: text/css'); //css文件

header('Content-type: text/javascript'); //js文件

header('Content-type: application/json'); //json

header('Content-type: application/pdf'); //pdf

header('Content-type: text/xml'); //xml

header('Content-Type: application/x-shockw\*\*e-flash'); //Flash动画

######

###声明一个下载的文件###

header('Content-Type: application/octet-stream');

header('Content-Disposition: attachment; filename="ITblog.zip"');

header('Content-Transfer-Encoding: binary');

readfile('test.zip');

######

###对当前文档禁用缓存###

header('Cache-Control: no-cache, no-store, max-age=0, must-revalidate');

header('Expires: Mon, 26 Jul 1997 05:00:00 GMT');

######

###显示一个需要验证的登陆对话框###

header('HTTP/1.1 401 Unauthorized');

header('WWW-Authenticate: Basic realm="Top Secret"');

######

###声明一个需要下载的xls文件###

header('Content-Disposition: attachment; filename=abc.xlsx');

header('Content-Type: application/vnd.openxmlformats-officedocument.spreadsheetml.sheet');

header('Content-Length: '.filesize('./test.xls'));

header('Content-Transfer-Encoding: binary');

header('Cache-Control: must-revalidate');

header('Pragma: public');

readfile('./test.xls');

* 1. 下载所要用的的请求头

header("Content-type:application/octet-stream");

header("Accept-Ranges:bytes");

header("Accept-Length:".$file\_Size);

header("Content-Disposition: attachment; filename=".$filename);

content-type：文件类型

Accept-Ranges：表示接收数据的类型或者范围，图片属于二进制的东西所以需要使用字节的方式传输

Accept-Length：表示接收的文件大小，php文件下载需要告诉浏览器下载的文件有多大

Content-Disposition：附件只需要把文件名给过去就可以，这个名称就是下载时显示的文件名称

window系统的文件名是GB2312的，根据不同的情况可能需要转码：

$filename=iconv("UTF-8","GB2312",$filename);

* 1. PHP代码

<?php

ob\_clean();

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

$filename = base64\_decode($action);//传的参数encode了

$filepath = '/data/www/www.test.com/'.$filename;

if(!file\_exists($filepath)){

echo "<script>alert('文件不存在')</script>";

exit;

}

$fp=fopen($filepath,"r");

$filesize=filesize($filepath);

header("Content-type:application/octet-stream");

header("Accept-Ranges:bytes");

header("Accept-Length:".$filesize);

header("Content-Disposition: attachment; filename=".$filename);

$buffer=1024;

$buffer\_count=0;

while(!feof($fp)&&$file\_Size-$buffer\_count>0){

$data=fread($fp,$buffer);

$buffer\_count+=$buffer;

echo $data;

}

fclose($fp);

?>

1. JS实现改变地址栏URL而不跳转
   1. History

window.history.back(); //这个方法会像用户点击了浏览器工具栏上的返回键一样

window.history.forward();

* + 1. 移动到历史记录特定的位置

window.history.go(-1); //相当于 back()

window.history.go(1); //相当于 forward()

检查浏览器历史记录的length属性来找到历史记录堆栈中的页面总数。

var numberOfEntries = window.history.length;

* 1. 添加和修改 history 实体

Html5引入了 histtory.pushState() 和 history.replaceState() 这两个方法，他们允许添加和修改 history 实体。同时，这些方法会和 window.onpostate 事件一起工作。

使用 history.pushState() 方法来修改 referrer，这种方法可以被用在经过修改状态后而为 xmlhttpRequest 对象创建的 http header 中。这个 referrer 会是创建 XMLHttpRequest 时 document 的URL。

pushState 用于向 history 添加当前页面的记录，而 replaceState 和 pushState 的用法完全一样，唯一的区别就是它用于修改当前页面在 history 中的记录。

* + 1. pushState方法

history.pushState(state, title, url)

将当前URL和history.state加入到history中，并用新的state和URL替换当前。不会造成页面刷新。

state对象：state对象是一个JavaScript对象，它关系到由pushState()方法创建出来的新的history实体。用以存储关于你所要插入到历史 记录的条目的相关信息。State对象可以是任何Json字符串。因为firefox会使用用户的硬盘来存取state对象，这个对象的最大存储空间为640k。如果大于这个数 值，则pushState()方法会抛出一个异常。如果确实需要更多的空间来存储，请使用本地存储。

title：firefox现在回忽略这个参数，虽然它可能将来会被使用上。而现在最安全的使用方式是传一个空字符串，以防止将来的修改。或者可以传一个简短的标题来表示state

URL：这个参数用来传递新的history实体的URL，注意浏览器将不会在调用pushState()方法后加载这个URL。但也许会过一会尝试加载这个URL。比如在用户重启了浏览器后，新的url可以不是绝对路径。如果是相对路径，那么它会相对于现有的url。新的url必须和现有的url同域，否则pushState()将抛出异常。这个参数是选填的，如果为空，则会被置为document当前的url。

* 1. replaceState()方法

用新的state和URL替换当前。不会造成页面刷新。

history.replaceState() 用起来很像pushState()，除了replaceState()是用来修改当前的history实体而不是创建一个新的。这个方法有时会很有用，当 你需要对某些用户行为作反应而更新一个state对象或者当前history实体时，可以使用它来更新state对象或者当前history实体的url。

若仅是修改当前URL可以使用

replaceState (null,null,"url")

* 1. popstate事件

当history实体被改变时，popstate事件将会发生。如果history实体是有pushState和replaceState方法产生的，popstate事件的state属性会包含一份来自history实体的state对象的拷贝。

window.onpopstate

history.go和history.back（包括用户按浏览器历史前进后退按钮）触发，并且页面无刷的时候（由于使用pushState修改了history）会触发popstate事件，事件发生时浏览器会从history中取出URL和对应的state对象替换当前的URL和history.state。通过event.state也可以获取history.state。

* 1. 读取当前的state

当页面加载时，它可能会有一个非空的state对象。这可能发生在当页面设置一个state对象(使用pushState或者replaceState)之后用户重启了浏览器。当页面重新加载，页面将收到onload事件，但不会有popstate事件。然而，如果你读取history.state属性，将在popstate事件发生后得到这个state对象。

1. js准确获取当前页面url网址信息

在WEB开发中，时常会用到javascript来获取当前页面的url网址信息，在这里是我的一些获取url信息的小总结。

下面我们举例一个URL，然后获得它的各个组成部分：

http://i.cnblogs.com/EditPosts.aspx?opt=1

1、window.location.href(设置或获取整个 URL 为字符串)

var test = window.location.href;

alert(test);

返回：http://i.cnblogs.com/EditPosts.aspx?opt=1

2、window.location.protocol(设置或获取 URL 的协议部分)

var test = window.location.protocol;

alert(test);

返回：http:

3、window.location.host(设置或获取 URL 的主机部分)

var test = window.location.host;

alert(test);

返回：i.cnblogs.com

4、window.location.port(设置或获取与 URL 关联的端口号码)

var test = window.location.port;

alert(test);

返回：空字符(如果采用默认的80端口(update:即使添加了:80)，那么返回值并不是默认的80而是空字符)

5、window.location.pathname(设置或获取与 URL 的路径部分（就是文件地址）)

var test = window.location.pathname;

alert(test);

返回：/EditPosts.aspx

6、window.location.search(设置或获取 href 属性中跟在问号后面的部分)

var test = window.location.search;

alert(test);

返回：?opt=1

PS：获得查询（参数）部分，除了给动态语言赋值以外，我们同样可以给静态页面，并使用javascript来获得相信应的参数值。

7、window.location.hash(设置或获取 href 属性中在井号“#”后面的分段)

var test = window.location.hash;

alert(test);

返回：空字符(因为url中没有)

* 1. js获取url中的参数值
     1. 正则法

function getQueryString(name) {

var reg = new RegExp('(^|&)' + name + '=([^&]\*)(&|$)', 'i');

var r = window.location.search.substr(1).match(reg);

if (r != null) {

return unescape(r[2]);

}

return null;

}

// 这样调用：

alert(GetQueryString("参数名1"));

alert(GetQueryString("参数名2"));

alert(GetQueryString("参数名3"));

* + 1. split拆分法

function GetRequest() {

var url = location.search; //获取url中"?"符后的字串

var theRequest = new Object();

if (url.indexOf("?") != -1) {

var str = url.substr(1);

strs = str.split("&");

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

theRequest[strs[i].split("=")[0]] = unescape(strs[i].split("=")[1]);

}

}

return theRequest;

}

var Request = new Object();

Request = GetRequest();<br>// var id=Request["id"];

// var 参数1,参数2,参数3,参数N;

// 参数1 = Request['参数1'];

// 参数2 = Request['参数2'];

// 参数3 = Request['参数3'];

// 参数N = Request['参数N'];

1. JS动态修改单网页web应用在微信浏览器中的title

document.title = '需要设置的值';

document.getElementsByTagName("title")[0].innerText = '需要设置的值';

$('title').html(''); //jquery的方法

$('title').text('')

上面这两种方法，在安卓里面是可以运行的，但是在ios上并不生效，第三种方法为ios上的方法

window.setDocumentTitle = function(title) {

document.title = 'the title you want to set';

var i = document.createElement('iframe');

i.src = '../favicon.ico';

i.style.display = 'none';

i.onload = function() {

setTimeout(function(){

i.remove();

}, 9)

}

document.body.appendChild(i);

}

OR 依赖jq

var $body = $('body');

document.title = 'the title you want to set';

var $iframe = $("<iframe style='display:none;' src='/favicon.ico'></iframe>");

$iframe.on('load',function() {

setTimeout(function() {

$iframe.off('load').remove();

}, 0);

}).appendTo($body);