Search

Keyboard Shortcuts

?
: This help

j
: Next menu item

k
: Previous menu item

g p
: Previous man page

g n
: Next man page

G
: Scroll to bottom

g g
: Scroll to top

g h
: Goto homepage

g s
: Goto search
(current page)

/
: Focus search box

file:// »
« Zlib context options

- Руководство по PHP
- Справочник языка

Change language: Russian

Submit a Pull Request Report a Bug

Поддерживаемые протоколы и обёртки

PHP поставляется со множеством встроенных обёрток для различных URL-протоколов для использования с функциями файловой системы, таких как fopen(), copy(), file_exists() и filesize(). В дополнение к этим обёрткам, можно регистрировать собственные обёртки, используя функцию stream_wrapper_register().

> **Замечание**: URL синтаксис, используемый для описания обёртки, может быть только вида `scheme://...`. Варианты синтаксиса `scheme:/` и `scheme:` не поддерживаются.

Содержание

User Contributed Notes 31 notes

26
fabacrans at nospamhotmail dot com ¶
10 years ago

```
You can use "php://input" to accept and parse "PUT", "DELETE", etc. requests.

<?php
// Example to parse "PUT" requests
parse_str(file_get_contents('php://input'), $_PUT);

// The result
print_r($_PUT);
?>

(very useful for Restful API)
```


19
sander at medicore dot nl ¶
16 years ago

```
to create a raw tcp listener system i use the following:

xinetd daemon with config like:
service test
{
disable = no
type = UNLISTED
socket_type = stream
protocol = tcp
bind = 127.0.0.1
port = 12345
wait = no
user = apache
```

```
group = apache
instances = 10
server = /usr/local/bin/php
server_args = -n [your php file here]
only_from = 127.0.0.1 #gotta love the security#
log_type = FILE /var/log/phperrors.log
log_on_success += DURATION
}
```

now use fgets(STDIN) to read the input. Creates connections pretty quick, works like a charm.Writing can be done using the STDOUT, or just echo. Be aware that you're completely bypassing the webserver and thus certain variables will not be available.

up
down
14

ben dot johansen at gmail dot com ¶

17 years ago

Example of how to use the php://input to get raw post data

```
//read the raw data in
$roughHTTPPOST = file_get_contents("php://input");
//parse it into vars
parse_str($roughHTTPPOST);
```

if you do readfile("php://input") you will get the length of the post data

up
down
14

ben dot johansen at gmail dot com ¶

17 years ago

In trying to do AJAX with PHP and Javascript, I came upon an issue where the POST argument from the following javascript could not be read in via PHP 5 using the $_REQUEST or $_POST. I finally figured out how to read in the raw data using the php://input directive.

Javascript code:

```
=============
//create request instance
xhttp = new XMLHttpRequest();
// set the event handler
xhttp.onreadystatechange = serviceReturn;
// prep the call, http method=POST, true=asynchronous call
var Args = 'number='+NbrValue;
xhttp.open("POST", "http://<?php echo $_SERVER['SERVER_NAME'] ?>/webservices/ws_service.php", true);
// send the call with args
xhttp.send(Args);
```

PHP Code:

```
//read the raw data in
$roughHTTPPOST = file_get_contents("php://input");
//parse it into vars
parse_str($roughHTTPPOST);
```

up
down
11

sebastian dot krebs at kingcrunch dot de ¶

13 years ago

The stream php://temp/maxmemory:$limit stores the data in memory unless the limit is reached. Then it will write the whole content the a temporary file and frees the memory. I didnt found a way to get at least some of the data back to memory.

up
down
11

aidan at php dot net ¶

19 years ago

The contants:

* STDIN
* STDOUT
* STDERR

Were introduced in PHP 4.3.0 and are synomous with the fopen('php://stdx') result resource.

up
down
11

php at rapsys dot eu ¶

11 years ago

Here is a snippet to read compressed raw post data without enabling global variables.

I needed it to read xml posted data submitted by ocs agent. The data was sent as Content-Type: application/x-compressed (zlib compressed data).

It seems related to an old bug which still seems broken :
https://bugs.php.net/bug.php?id=49411

The important part is the default window set to 15 instead of -15.

Code snippet

```
<?php
$data = '';
$fh = fopen('php://input', 'rb');
stream_filter_append($fh, 'zlib.inflate', STREAM_FILTER_READ, array('window'=>15));
while(!feof($fh)) {
$data .= fread($fh, 8192);
}
?>
```

up
down
12

Hayley Watson ¶

6 years ago

Even though their names will be the same, you can have more than one //memory or //temp stream open concurrently; each time you fopen() such a stream, a NEW stream will be opened independently of the others.

This is hinted at by the fact you don't add any unique identifier to the path when creating such streams, but isn't said explicitly.

```
<?php

$hello = fopen('php://memory', 'r+'); // $hello, $php, $world are all different streams.
$php = fopen('php://memory', 'r+');
$world = fopen('php://memory', 'r+'); // They're not the same stream opened three times.

fputs($hello, "Hello ");
fputs($php, "PHP ");
rewind($php);
fputs($world, "World!");
rewind($hello);
rewind($world);

echo '[', stream_get_contents($hello), '][', stream_get_contents($php), '][', stream_get_contents($world), ']';
// If they were the same stream the output would be "[World!][World!][World!]".
?>
```

up
down
11

heitorsiller at uol dot com dot br ¶

17 years ago

```
For reading a XML stream, this will work just fine:
<?php

$arq = file_get_contents('php://input');

?>

Then you can parse the XML like this:

<?php

$xml = xml_parser_create();

xml_parse_into_struct($xml, $arq, $vs);

xml_parser_free($xml);

$data = "";

foreach($vs as $v){

if($v['level'] == 3 && $v['type'] == 'complete')
$data .= "\n".$v['tag']." -> ".$v['value'];
}

echo $data;

?>

PS.: This is particularly useful for receiving mobile originated (MO) SMS messages from cellular phone companies.
```

up
down
7

vibhavsinha91 at gmail dot com ¶

9 years ago

```
While writing to error stream, error_log() function comes as a shorthand to writing to php://stderr . This function also
allows writing to web server log when running through a web server such as apache.
```

up
down
2

kazdotkanso at geeemail dot com ¶

3 years ago

```
The php://fd/ wrapper is only supported in the cli tool.
```

up
down
11

gjaman at gmail dot com ¶

15 years ago

```
You can decompress (gzip) a input stream by combining wrappers:

eg: $x = file_get_contents("compress.zlib://php://input");

I used this method to decompress a gzip stream that was pushed to my webserver
```

up
down
8

nargy at yahoo dot com ¶

19 years ago

```
When opening php://output in append mode you get an error, the way to do it:
$fp=fopen("php://output","w");
```

```
fwrite($fp,"Hello, world !<BR>\n");
fclose($fp);
```

up
down
7

Anonymous ¶

6 years ago

If you want to filter incoming data through php://input use this:

```
file_get_contents("php://filter/read=string.strip_tags/resource=php://input");
```

I couldn't find any documentation to explain how to do this. All the examples I came across suggested that a full and actual URL had to be used (which didn't work for me).

This seems to work though.

up
down
6

Justin Megawarne ¶

10 years ago

If my understanding of the implementing code is correct, every time you open a php://memory stream, you get new storage allocated. That is to say, php://memory isn't a shared bank of memory.

up
down
7

sam at bigwig dot net ¶

20 years ago

[Editor's Note: There is a way to know. All response headers (from both the final responding server and intermediate redirecters) can be found in $http_response_header or stream_get_meta_data() as described above.]

If you open an HTTP url and the server issues a Location style redirect, the redirected contents will be read but you can't find out that this has happened.

So if you then parse the returned html and try and rationalise relative URLs you could get it wrong.

up
down
3

Anonymous ¶

5 years ago

Be forewarned:

the file:// protocol used in file_get_contents is used as the default for "any unrecognized protocol." Thus:

```
aldfjadlfadfladfl://whatever
```

will deliver the same as

```
file://whatever
```

up
down
5

leonid at shagabutdinov dot com ¶

12 years ago

For https for windows enable this extension:

```
extension=php_openssl.dll
```

up
down
4

aaron dot mason+php at thats-too-much dot info ¶

11 years ago

Be aware of code injection, folks - like anything else you take from the user, SANITISE IT FIRST. This cannot be stressed

enough - if I had a dollar for each time I saw code where form input was taken and directly used (by myself as well, I've been stupid too) I'd probably own PHP. While using data from a form in a URL wrapper is asking for trouble, you can greatly minimise the trouble by making sure your inputs are sane and not likely to provide an opening for the LulzSec of the world to cause havoc.

up
down
4

nyvsld at gmail dot com ¶

18 years ago

php://stdin supports fseek() and fstat() function call,
while php://input doesn't.

up
down
4

ben dot johansen at gmail dot com ¶

17 years ago

followup:

I found that if I added this line to the AJAX call, the values would show up in the $_POST

```
xhttp.setRequestHeader('Content-Type',
'application/x-www-form-urlencoded');
```

up
down
4

lupti at yahoo dot com ¶

20 years ago

I find using file_get_contents with php://input is very handy and efficient. Here is the code:

```
$request = "";
$request = file_get_contents("php://input");
```

I don't need to declare the URL filr string as "r". It automatically handles open the file with read.

I can then use this $request string to your XMLparser as data.

up
down
3

oliver at codeinline dot com ¶

10 years ago

A useful way to handle large file uploads is to do something like:

```
copy(("php://input"),$tmpfile);
```

as this avoids using lots of memory just to buffer the file content.

The correct mime type for this should be "application/octet-stream" however if you set this or any other recognised mime type other than "multipart/form-data" on your POST then $HTTP_RAW_POST_DATA is populated and the memory is consumed anyway.

Setting the mime type to "multipart/form-data" raises “PHP Warning: Missing boundary in multipart/form-data POST data in Unknown on line 0” however it seems to work without a problem.

up
down
3

Anonymous ¶

11 years ago

For php://filter the /resource=foo part must come last. And foo needs no escaping at all.
php://filter/resource=foo/read=somefilter would try to open a file 'foo/read=somefilter' while
php://filter/read=somefilter/resource=foo will open file 'foo' with the somefilter filter applied.

up
down

3

chris at free-source dot com ¶

18 years ago

If you're looking for a unix based smb wrapper there isn't one built in, but I've had luck with http://www.zevils.com/cgi-bin/viewcvs.cgi/libsmbclient-php/ (tarball link at the end).

up

down

3

jerry at gii dot co dot jp ¶

16 years ago

Not only are STDIN, STDOUT, and STDERR only allowed for CLI programs, but they are not allowed for programs that are read from STDIN. That can confuse you if you try to type in a simple test program.

up

down

2

dave at 4mation dot com dot au ¶

10 years ago

The use of php://temp/maxmemory as a stream counts towards the memory usage of the script; you are not specifying a new memory pool by using this type of stream.
As noted in the documentation however, this stream type will start to write to a file after the specified maxmemory limit is exceeded. This file buffer is NOT observed by the memory limit.
This is handy if you want your script to have a reasonably small memory limit (eg 32MB) but but still be able to handle a huge amount of data in a stream (eg 256MB)

The only works if you use stream functions like fputs(); if you use $buffer .= 'string'; or $buffer = $buffer . 'string'; you're calling your stream data back into PHP and this will hit the limiter.

As a practical example:

```
<?php
// 0.5MB memory limit
ini_set('memory_limit', '0.5M');
// 2MB stream limit
$buffer = fopen('php://temp/maxmemory:1048576', 'r+');
$x = 0;
// Attempt to write 1MB to the stream
while ($x < 1*1024*1024) {
fputs($buffer, 'a');
$x++;
}
echo "This will never be displayed";
?>
```

However, change fopen to use php://temp/maxmemory:1 (one byte, rather than one megabyte) and it will begin writing to the unlimited file stream immediately, avoiding memory limit errors.

up

down

1

Anonymous ¶

8 years ago

Note that STDIN and similar are defined only in CLI

up

down

1

ohcc at 163 dot com ¶

8 years ago

```
<?php
//enable $HTTP_RAW_POST_DATA when necessary
ini_set('always_populate_raw_post_data',-1);
$HTTP_RAW_POST_DATA = file_get_contents('php://input');
echo $HTTP_RAW_POST_DATA;
?>
```

up
down
1
Anonymous ¶
10 years ago
In PHP 5.4+ you can read multipart data via php://input if you set enable_post_data_reading to Off.

Of course if you set it to off, the $_POST and $_FILES superglobals won't be populated at all. It's entirely up to you to parse the data now.
up
down
-2
nguyenanthuan at gmail dot com ¶
9 years ago
Each stream pointer to php://memory and php://temp has its own memory allocation, so you can open many stream pointers to store your separated values.

```
<?php
$fp = fopen("php://temp", "r+");
$fp2 = fopen("php://temp", "r+");

fwrite($fp, "line1\n");
fwrite($fp2, "line4\n");
fwrite($fp, "line2\n");
fwrite($fp2, "line5\n");
fwrite($fp, "line3\n");
fwrite($fp2, "line6\n");

var_dump(memory_get_usage());

rewind($fp);
while(!feof($fp)) {
var_dump(fread($fp, 1024));
}
fclose($fp);
var_dump(memory_get_usage());

rewind($fp2);
while(!feof($fp2)) {
var_dump(fread($fp2, 1024));
}
fclose($fp2);
var_dump(memory_get_usage());
?>
```

Closing their stream handles will also free the allocated memory.

php://memory stream type is MEMORY, while php://temp stream type is STDIO FILE*.