Last updated: Fri, 26 Feb 2010

List of Supported Protocols/Wrappers

Table of Contents

PHP comes with many built-in wrappers for various URL-style protocols for use with the filesystem functions such as fopen() and copy(). In addition to these wrappers, you can as of PHP 4.3.0, write your own wrappers using the stream_wrapper_register() function.

Note: The URL syntax used to describe a wrapper only supports the scheme://... syntax. scheme:/ and scheme: syntaxes are not supported.

List of context options is available in the chapter Context options and parameters.

User Contributed Notes

List of Supported Protocols/Wrappers

travis at blackpulp dot com
15-Nov-2008 07:07

```
Not exactly sure why this was the solution, but if you are trying to fopen("php://input", "r") and
then wanting to print out the contents of the stream, you cannot do this:

<?
$putdata = fopen("php://input", "r");
while ($data = fread($putdata, 1024))
{
  print $data;
}
?>

On the above code, for some reason seemed to be stuck in an endless loop. Im a noob, so I don't know
why this was happening....

BUT, this is what worked for me
<?
$fp = fopen('php://input','r');
print stream_get_contents($fp);
?>
```

gjaman at gmail dot com
15-May-2008 09:15

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

marcus at synchromedia dot co dot uk
18-Apr-2008 12:36

If you want to talk to serial ports, on Linux or windows, there is some good discussion of it here:

http://blogs.vinuthomas.com/2007/04/09/php-and-serial-ports/

jerry at gii dot co dot jp
17-Aug-2007 05:11

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.

sander at medicore dot nl
14-Jun-2007 11:25

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.

ben dot johansen at gmail dot com
25-Oct-2006 09:57

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');
```

ben dot johansen at gmail dot com
29-Aug-2006 06:02

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

ben dot johansen at gmail dot com
29-Aug-2006 07:33

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);
```

heitorsiller at uol dot com dot br
07-Jul-2006 02:55

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.

opedroso at NOSPAMswoptimizer dot com
12-Apr-2006 06:07

php://input allows you to read raw POST data. It is a less memory intensive alternative to $HTTP_RAW_POST_DATA and does not need any special php.ini directives.

Example use:

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

When reading a base64 encoded stream using php://input, be aware that you do not need to decode it, it will automatically be done for you.

nyvsld at gmail dot com
27-Nov-2005 06:28

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

drewish at katherinehouse dot com
25-Sep-2005 06:50

Be aware that contrary to the way this makes it sound, under Apache, php://output and php://stdout don't point to the same place.

```
<?php
$fo = fopen('php://output', 'w');
$fs = fopen('php://stdout', 'w');

fputs($fo, "You can see this with the CLI and Apache.\n");
fputs($fs, "This only shows up on the CLI...\n");
```

```
fclose($fo);
fclose($fs);
?>

Using the CLI you'll see:
You can see this with the CLI and Apache.
This only shows up on the CLI...

Using the Apache SAPI you'll see:
You can see this with the CLI and Apache.
```

chris at free-source dot com
26-Apr-2005 07:52

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).

nargy at yahoo dot com
24-Sep-2004 10:16

```
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);
```

aidan at php dot net
27-May-2004 10:34

The contants:

* STDIN
* STDOUT
* STDERR

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

lupti at yahoo dot com
29-Nov-2003 10:04

```
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.
```

sam at bigwig dot net
15-Aug-2003 03:02

```
[ 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.
```