Generic Wrapper Requests

This page is maintained to keep the documentation smaller when it comes to generic requests that can be used in all wrapper modules. Methods at this page is normally supported by CurlWrapper, NetWrapper, SimpleStreamWrapper.

A big difference between 6.0 and 6.1 is the chaining, which did not exist in the old versions as it supported PHP 5.3 in the "good old (horrible) times". Almost everything in netcurl 6.1 is chained. There are however limits in different kind of places with makes requests with the obsolete PHP versions impossible. All autotests are running with PHP 5.6 or higher. PHP 5.4 does not work with autotests anymore, but *could* still be compatible in rare cases (I really don't know). All wrappers are based on same requests, so regardless if you use the curlWrapper directly or the generic netWrapper, the outcome will be much the same.

Table of contents

getParsed(), getBody(), getHeader(), getCode()

By using getBody(), you can also fetch the data unparsed. Likewise, you can also fetch other data from the request, like for example the returned headers from the server with a getHeader()-method. By running getHeader() without an argument, you'll ge the entire header, so if you'd like to fetch specific header rows, you can add a parameter like the example below:

Responses

```
// Returns the server name. Example: Apache/2.4.29 (Ubuntu)
$serverName = $wrapper->getHeader('Server');
// Returns full HTTP header response with code. Example: 200 OK
$httpHeadResponse = $wrapper->getHeader('HTTP');
// Returns the HTTP header response code only. Example: 200
$httpCode = $wrapper->getCode();
```

Basic Requests

The most basic request is done with simply using request(). The only argument it actually needs is the URL, and then you can start using the above get-methods to retrieve what you need from the site. There are however a bunch of arguments to use here, including multi-requests (i.e. for curl and the wrappers that supports multiple requests in the same call).

require()-argument	Format	Description
url	string or array	The only one you need.
data	array	Data to include in the request as an array. Formatted as default postdata if nothing else is given.
method	RequestMethod::	RequestMethod is **GET, POST, PUT, DELETE, HEAD, REQUEST, PATCH**
dataType	DataType::	How data is posted in the request. Can be **NORMAL (?var=val&var1=val1), JSON, SOAP, XML, SOAP_XML, RSS_XML**

String or array?

As you can see, you can add urls in the format of an array. By doing this you do an important one step up, which transform netcurl very much in a multirequest tool. Referring to the different wrappers it can look like anything from:

```
            (new NetWrapper())->request(
                [
                    'http://ipv4.netcurl.org/http.php?code=200&message=Funktionsduglig',
                    'http://ipv4.netcurl.org/http.php?code=500&message=Kass',
                    'http://ipv4.netcurl.org/http.php?code=201&message=Mittemellan',
                ]
            );
```

Where the multirequet is very simple built. Or if you want to advance into something more complex:

```
$wrapperFirst = (new CurlWrapper())->request([
            [
                'url' => 'https://ipv4.netcurl.org/',
                'requestMethod' => RequestMethod::POST,
                'dataType' => DataType::NORMAL,
                'data' => [
                    'dataRequestMethod' => 'FIRST',
                ],
                'headers' => [
                    'XHeaderFirst' => 'yes',
                    'X-Real-IP' => '255.255.255.0',
                    'Client-IP' => '127.0.0.255',
                    'X-Forwarded-For' => '127.255.0.0',
                ],
                'headers_static' => [
                    'HeaderIsForever' => 'only-in-non-multi-curls',
                ],
            ],
            [
                'url' => 'https://ipv4.netcurl.org/',
                'requestMethod' => RequestMethod::POST,
                'dataType' => DataType::NORMAL,
                'data' => [
                    'dataRequestMethod' => 'SECOND',
                ],
                'headers' => [
                    'XHeaderSecond' => 'yes',
                ],
            ],
            [
                'url' => 'https://ipv4.netcurl.org/',
                'requestMethod' => RequestMethod::GET,
                'data' => [
                    'dataRequestMethod' => 'THIRD',
                ],
            ],
        ]);
```

Each time the wrappers get this kind of request it always checks if curl_multi is available before using the simpler methods. So you can use this urllist as a simple array, or associative array.

This is some of the arguments in a associative array.

Argument	Description
url	See above
requestMethod	See above
dataType	See above
data	See above
headers	Extra headers to include into the request.
headers_static	Extra headers to include into the request, but they will restored even if the wrapper is reset between the requests.