Commonly Used PHP Extensions

Disclaimer: вы смотрите просто запись лекции, это **НЕ** специально подготовленный видеокурс!

Intro

While PHP has a lot of inbuilt capabilities, even more features are available through extensions – special libraries that distributed along with PHP itself, or available with PECL (PHP Extension Community Library, a repository for PHP extensions).

Setup and configuration

The general approach is the next:

1. If an extension is NOT distributed with PHP itself, download it from PECL and put it to PHP extensions directory.
2. Uncomment (or add) proper “extension=extension_name” string in php.ini.
3. (If necessary) add / uncomment / change extension settings in php.ini.

Important! Under Windows “extension_dir” parameter in php.ini usually should contain full path to the corresponding directory. And for many extensions to work the path to PHP main directory should be added to PATH system variable.

ImageMagick (Imagick)

ImageMagick is a software suite to create, edit, and compose bitmap images. It can read, convert and write images in a variety of formats (over 100) including DPX, EXR, GIF, JPEG, JPEG-2000, PDF, PhotoCD, PNG, Postscript, SVG, and TIFF.

Setup:

1. Download: https://pecl.php.net/package/imagick
2. Unpack php_imagick.dll and php_imagick.pdb to PHP extensions directory.
3. Unpack all other files to ImageMagic subdirectory in PHP extensions directory.
4. Add the directory from step 3 to PATH system variable.
5. Create MAGICK_CONFIGURE_PATH system variable and the full path to ImageMagic\config there.
6. Add “extension=imagic” to php.ini.

ImageMagick (Imagick) sample 1: thumbnail

This is how easy to create an image thumbnail with ImageMagic:

```
<?php

$imagick = new Imagick();
$imagick->readImage( './01_imagick_thumbnail_cat_src.jpg');
$imagick->thumbnailImage( 200, null );
$imagick->writeImage( './01_imagick_thumbnail_cat_dst.jpg');
$imagick->destroy();
```

ImageMagick (Imagick) sample 2: watermark

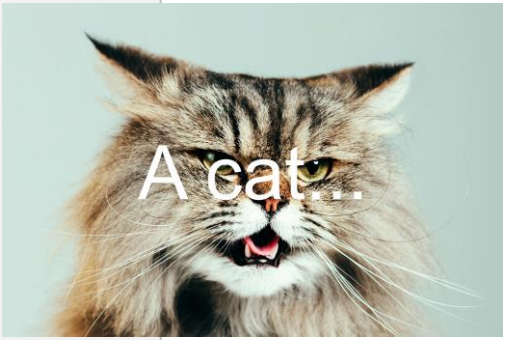
This is how easy to add a label (watermark) with ImageMagic:

```
<?php

$imagick = new Imagick();
$imagickDraw = new ImagickDraw();
$imagickDraw->setFontSize( 500 );
$imagickDraw->setFillColor('#ffffff');
$imagick->readImage( './02_imagick_watermark_cat_src.jpg' );
$imagickDraw->setGravity( Imagick::GRAVITY_CENTER );
$imagick->annotateImage( $imagickDraw, 4, 20, 0, 'A cat...');
$imagick->setImageFormat( 'jpg' );
$imagick->writeImage( './02_imagick_watermark_cat_dst.jpg');

// Output to HTTP
// header( "Content-Type: image/{$imagick->getImageFormat()}" );
// echo $imagick->getImageBlob();

$imagick->destroy();
```



libcurl (cURL)

Libcurl is library that allows one to connect and communicate to many different types of servers with many different types of protocols. It supports http, https, ftp, gopher, telnet, dict, file, and ldap protocols. It also supports https certificates, http post, http put, ftp uploading, http form based upload, proxies, cookies, and user+password authentication.

Setup:

1. Uncomment “extension=curl” in php.ini.
2. (Under Windows) make sure that your PHP directory is included in PATH system variable.

libcurl (cURL) sample

Here is a sample of how to send a file with cURL and get the progress:

```php
<?php

$data = array(
    'file' => new CurlFile('./02_imagick_watermark_cat_src.jpg', 'image/jpeg', 'cat.jpg'),
    'anyOtherParameter' => 'anyOtherValue'
);

$curl = curl_init();
curl_setopt($curl, CURLOPT_URL, "http://192.168.56.101//03_curl_file_upload_test.php");
curl_setopt($curl, CURLOPT_HEADER, false);
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
curl_setopt($curl, CURLOPT_USERAGENT, "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:101.0) Gecko/20100101 Firefox/101.0");
curl_setopt($curl, CURLOPT_POST, true);
curl_setopt($curl, CURLOPT_POSTFIELDS, $data);
curl_setopt($curl, CURLOPT_FOLLOWLOCATION, true);
curl_setopt($curl, CURLOPT_NOPROGRESS, false);
curl_setopt($curl, CURLOPT_PROGRESSFUNCTION, 'progressCallback');
curl_setopt($curl, CURLOPT_BUFFERSIZE, 128);
curl_setopt($curl, CURLOPT_CONNECTTIMEOUT, 160);
curl_setopt($curl, CURLOPT_TIMEOUT, 160);
$result = curl_exec($curl);
curl_close($curl);
echo $result;

function progressCallBack($curl, $dltotal, $dlnow, $uptotal, $upnow)
{
    echo "DLt = " . $dltotal . ", DLn = " . $dlnow . ", UPt = " . $uptotal . ", UPn = " . $upnow ."\n";
}
```

```php
<?php

print_r($_FILES);
print_r($_POST);
```

```
DLt = 0, DLn = 0, UPt = 0, UPn = 0
DLt = 0, DLn = 0, UPt = 542202, UPn = 0
DLt = 0, DLn = 0, UPt = 542202, UPn = 65536
DLt = 0, DLn = 0, UPt = 542202, UPn = 131072
DLt = 0, DLn = 0, UPt = 542202, UPn = 196608
DLt = 0, DLn = 0, UPt = 542202, UPn = 196608
DLt = 0, DLn = 0, UPt = 542202, UPn = 262144
DLt = 0, DLn = 0, UPt = 542202, UPn = 327680
DLt = 0, DLn = 0, UPt = 542202, UPn = 393216
DLt = 0, DLn = 0, UPt = 542202, UPn = 393216
DLt = 0, DLn = 0, UPt = 542202, UPn = 458752
DLt = 0, DLn = 0, UPt = 542202, UPn = 524288
DLt = 0, DLn = 0, UPt = 542202, UPn = 542202
DLt = 308, DLn = 308, UPt = 542202, UPn = 542202

Array
(
    [file] => Array
        (
            [name] => cat.jpg
            [full_path] => cat.jpg
            [type] => image/jpeg
            [tmp_name] => C:\Windows\Temp\phpD54D.tmp
            [error] => 0
            [size] => 541898
        )

)

Array
(
    [anyOtherParameter] => anyOtherValue
)
```

OPcache

OPcache improves PHP performance by storing precompiled script bytecode in shared memory, thereby removing the need for PHP to load and parse scripts on each request.

Setup:

1. Uncomment “zend_extension=opcache” in php.ini.
2. Make the following adjustments in php.ini:

```
[opcache]
; Determines if Zend OPCache is enabled
opcache.enable=1

; The OPcache shared memory storage size.
opcache.memory_consumption=128

; The amount of memory for interned strings in Mbytes.
opcache.interned_strings_buffer=8

; The maximum number of keys (scripts) in the OPcache hash table.
; Only numbers between 200 and 1000000 are allowed.
opcache.max_accelerated_files=10000

; When disabled, you must reset the OPcache manually or restart the
; webserver for changes to the filesystem to take effect.
opcache.validate_timestamps=1

; How often (in seconds) to check file timestamps for changes to the shared
; memory storage allocation. ("1" means validate once per second, but only
; once per request. "0" means always validate)
opcache.revalidate_freq=60

; If disabled, all PHPDoc comments are dropped from the code to reduce the
; size of the optimized code.
opcache.save_comments=1
```

OPcache demo

Here is a small demo on OPcache efficiency:

```php
<?php

// We have to create a huge file
// that will take PHP a lot of time to compile.

$fileName = './04_opcache_big_file.php';

if (is_file($fileName)) {
    unlink($fileName);
}

file_put_contents($fileName, '<?php' . "\n", FILE_APPEND);
file_put_contents($fileName, '$hugeArray = array();' . "\n", FILE_APPEND);
for ($i = 0; $i < 1000; $i++) {
    file_put_contents($fileName, '$hugeArray[] = ' . $i . ";\n", FILE_APPEND);
}
```

Run this with (opcache.enable=1) and without (opcache.enable=0) OPCache enabled. The “ab” utility is a part of Apache httpd.

```
ab -n 1000 -c 20 http://127.0.0.1/04_opcache_big_file.php
```

OPcache demo

Here are some results:

```
Concurrency Level:      20
Time taken for tests:   0.547 seconds
Complete requests:      1000
Requests per second:    1828.80 [#/sec] (mean)
Time per request:       10.936 [ms] (mean)
Time per request:       0.547 [ms] (mean, across all concurrent requests)
Transfer rate:          357.19 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   1.3      0      24
Processing:     0   10   9.0     10      62
Waiting:        0    9   8.9      9      62
Total:          0   11   9.0     10      62

Percentage of the requests served within a certain time (ms)
  50%     10
  66%     14
  75%     17
  80%     18
  90%     22
  95%     28
  98%     31
  99%     34
  100%    62 (longest request)
```

```
Concurrency Level:      20
Time taken for tests:   0.257 seconds
Complete requests:      1000
Requests per second:    3887.13 [#/sec] (mean)
Time per request:       5.145 [ms] (mean)
Time per request:       0.257 [ms] (mean, across all concurrent requests)
Transfer rate:          759.21 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   1.4      0      18
Processing:     0    5   7.6      0      48
Waiting:        0    4   7.3      0      48
Total:          0    5   7.7      0      48

Percentage of the requests served within a certain time (ms)
  50%      0
  66%      4
  75%      8
  80%     13
  90%     18
  95%     20
  98%     25
  99%     25
  100%    48 (longest request)
```

Even with a 20KB “application” enabling OPcache gives 1.5-2 times speedup.

~~Mcrypt~~, Sodium and OpenSSL

Mcrypt is an interface to the mcrypt library, which supports a wide variety of crypto-algorithms. It is “semi-deprecated” now, so let’s use Sodium and OpenSSL instead.

Setup:

1. If you still want Mcrypt:
   a. Download: https://pecl.php.net/package/mcrypt
   b. Unpack php_mcrypt.dll and php_mcrypt.pdb to PHP extensions directory.
   c. Add “extension=mcrypt” to php.ini.
2. Uncomment “extension=sodium” in php.ini.
3. Uncomment “extension=openssl” in php.ini.

Mcrypt file crypt/decrypt sample

Here is a small sample of a file encryption / decryption with Mcrypt:

```php
<?php
// --- ENCRYPTION ---

// The key should be random binary. Use scrypt, bcrypt or PBKDF2 to
// convert a string into a key; key is specified using hexadecimal.
$key = pack('H*', "bcb04b7e103a0cd8b54763051cef08bc55abe029fdebae5e1d417e2ffb2a00a3");

// Show key size use either 16, 24 or 32 byte keys for AES-128, 192
// and 256 respectively.
$keySize =  strlen($key);
echo "Key size: " . $keySize . "\n";

$plainText = file_get_contents('./05_mcrypt.php');

// Create a random IV (Initialization Vector) to use
// with CBC (Cipher Block Chaining) encoding.
$ivSize = mcrypt_get_iv_size(MCRYPT_RIJNDAEL_128, MCRYPT_MODE_CBC);
$iv = mcrypt_create_iv($ivSize, MCRYPT_RAND);

// Create a cipher text compatible with AES (Rijndael block size = 128).
// Only suitable for encoded input that never ends with value 00h
// (because of default zero padding)!
$ciphertext = mcrypt_encrypt(MCRYPT_RIJNDAEL_128, $key,
    $plainText, MCRYPT_MODE_CBC, $iv);

// Prepend the IV for it to be available for decryption.
$ciphertext = $iv . $ciphertext;

// Encode the resulting cipher text so it can be represented as a string.
$ciphertextBase64 = base64_encode($ciphertext);

file_put_contents('./05_mcrypt.php_enc', $ciphertextBase64);

// --- DECRYPTION ---

$ciphertextDec = base64_decode(file_get_contents('./05_mcrypt.php_enc'));

// Retrieve the IV, iv_size should be created using mcrypt_get_iv_size():
$ivDec = substr($ciphertextDec, 0, $ivSize);

// Retrieves the cipher text (everything except the $ivSize in the front):
$ciphertextDec = substr($ciphertextDec, $ivSize);

// Decrypt.
$plaintextDec = mcrypt_decrypt(MCRYPT_RIJNDAEL_128, $key,
    $ciphertextDec, MCRYPT_MODE_CBC, $ivDec);

file_put_contents('./05_mcrypt.php_dec', $plaintextDec);
```

99.99% this will NOT work under any PHP version starting from 8.0.

Sodium file crypt/decrypt sample

Here is a small sample of a file encryption / decryption with Sodium:

```php
<?php

function encrypt($fileName, $key)
{
    $data = file_get_contents($fileName);

    $nonce = random_bytes(
        SODIUM_CRYPTO_SECRETBOX_NONCEBYTES
    );

    $cipher = base64_encode(
        $nonce .
        sodium_crypto_secretbox(
            $data,
            $nonce,
            $key
        )
    );
    sodium_memzero($data);
    sodium_memzero($key);
    return $cipher;
}
```

```php
function decrypt($fileName, $key)
{
    $decoded = base64_decode(file_get_contents($fileName));

    if (mb_strlen($decoded, '8bit') < (SODIUM_CRYPTO_SECRETBOX_NONCEBYTES +
SODIUM_CRYPTO_SECRETBOX_MACBYTES)) {
        throw new Exception('Error in encoded data.');
    }
    $nonce = mb_substr($decoded, 0, SODIUM_CRYPTO_SECRETBOX_NONCEBYTES, '8bit');
    $ciphertext = mb_substr($decoded, SODIUM_CRYPTO_SECRETBOX_NONCEBYTES, null, '8bit');

    $plain = sodium_crypto_secretbox_open(
        $ciphertext,
        $nonce,
        $key
    );
    if ($plain === false) {
        throw new Exception('Error in encoded data.');
    }
    sodium_memzero($ciphertext);
    sodium_memzero($key);
    return $plain;
}
```

```php
$key = random_bytes(SODIUM_CRYPTO_SECRETBOX_KEYBYTES);
$enc = encrypt('./06_sodium.php', $key);
file_put_contents('./06_sodium.php_enc', $enc);

$dec = decrypt('./06_sodium.php_enc', $key);
file_put_contents('./06_sodium.php_dec', $dec);
```

OpenSSL file crypt/decrypt sample

Here is a small sample of a file encryption / decryption with OpenSSL:

```php
<?php

function encrypt($fileName, $key)
{
    $key = substr(sha1($key, true), 0, 16);
    $iv = openssl_random_pseudo_bytes(16);
    return $iv . openssl_encrypt(file_get_contents($fileName), 'AES-128-CBC', $key, OPENSSL_RAW_DATA, $iv);
}

function decrypt($fileName, $key)
{
    $key = substr(sha1($key, true), 0, 16);
    $data = file_get_contents($fileName);
    $iv = substr($data, 0, 16);
    $data = substr($data, 16);
    return openssl_decrypt($data, 'AES-128-CBC', $key, OPENSSL_RAW_DATA, $iv);
}

$key = sha1('Just some password');
$enc = encrypt('./07_openssl.php', $key);
file_put_contents('./07_openssl.php_enc', $enc);

$dec = decrypt('./07_openssl.php_enc', $key);
file_put_contents('./07_openssl.php_dec', $dec);
```

SOAP

The SOAP extension can be used to write SOAP Servers and Clients. It supports subsets of SOAP 1.1, SOAP 1.2 and WSDL 1.1 specifications.

Setup:

1. Uncomment “extension=soap” in php.ini.

Sorry, this course does not cover SOAP principles, but now we’ll see a quick sample…

SOAP sample

Here is a small sample of a SOAP client and server (service):

```xml
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="Library"
                  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                  targetNamespace="Library"
                  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
                  xmlns:tns="Library"
                  xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
    <xsd:documentation></xsd:documentation>
    <wsdl:types>
        <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetNamespace="Library">
            <xsd:complexType name="book">
                <xsd:sequence>
                  <xsd:element name="name" type="xsd:string"></xsd:element>
                  <xsd:element name="year" type="tns:integer"></xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:schema>
    </wsdl:types>
    <wsdl:message name="bookYearRequest">
        <wsdl:part name="book" type="xsd:book"></wsdl:part>
    </wsdl:message>
    <wsdl:message name="bookYearResponse">
        <wsdl:part name="year" type="tns:integer"></wsdl:part>
    </wsdl:message>
    <wsdl:portType name="Library">
        <wsdl:operation name="bookYear">
            <wsdl:input message="tns:bookYearRequest"/>
            <wsdl:output message="tns:bookYearResponse"/>
        </wsdl:operation>
    </wsdl:portType>
    <wsdl:binding name="Library" type="tns:Library">
        <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
        <wsdl:operation name="bookYear">
            <soap:operation soapAction="http://127.0.0.1/08_soap_server.php"/>
            <wsdl:input>
                <soap:body use="literal" namespace="Library"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" namespace="Library"/>
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>
    <wsdl:service name="Library">
        <wsdl:port binding="tns:Library" name="BookLibrary">
            <soap:address location="http://127.0.0.1/08_soap_server.php"/>
        </wsdl:port>
    </wsdl:service>
</wsdl:definitions>
```

```php
<?php

// https://gist.github.com/umidjons/f3de2533c51495a9c557

// Turn off WSDL caching:
ini_set("soap.wsdl_cache_enabled", "0");

// Model, which uses in web service functions as parameter:
class Book
{
    public $name;
    public $year;
}

/**
 * Determines published year of the book by name.
 * @param Book $book book instance with name set.
 * @return int published year of the book or 0 if not found.
 */
function bookYear($book)
{
    // List of the books:
    $_books = [
        ['name' => 'test 1', 'year' => 2011],
        ['name' => 'test 2', 'year' => 2012],
        ['name' => 'test 3', 'year' => 2013],
    ];

    // Search book by name:
    foreach ($_books as $bk) {
        if ($bk['name'] == $book->name)
            return $bk['year'];
    }

    return 0;
}

// Initialize SOAP Server:
$server = new SoapServer("./08_soap_wsdl.wsdl", [
    'classmap' => [
        'book' => 'Book', // 'book' complex type in WSDL file mapped to
the Book PHP class
    ]
]);

// Register available functions:
$server->addFunction('bookYear');

// Start handling requests:
$server->handle();
```

```php
<?php

// https://gist.github.com/umidjons/f3de2533c51495a9c557

class Book
{
    public $name;
    public $year;
}

// Create instance and set a book name:
$book = new Book();
$book->name = 'test 2';

// Initialize SOAP client and call web service function:
$client = new
SoapClient('http://127.0.0.1/08_soap_server.php?wsdl',
           ['trace' => 1, 'cache_wsdl' => WSDL_CACHE_NONE]);
$resp = $client->bookYear($book);

// View response:
var_dump($resp);
```

Tideways XHProf

Tideways XHProf (hierarchical profiler) is a light-weight profiler. During the data collection phase, it keeps track of call counts and inclusive metrics for arcs in the dynamic callgraph of a program. It computes exclusive metrics in the reporting/post processing phase, such as wall (elapsed) time, CPU time and memory usage. A functions profile can be broken down by callers or callees.

If this extension doesn't work with your latest PHP version, try using it with older PHP (i.e., 8.0 instead of 8.1).

Setup:

1. Go to: https://github.com/tideways/php-xhprof-extension
2. Follow instructions, i.e. (usually):
    1. Download pre-compiled SO (for Linux) or DLL (for Windows).
    2. Extract SO/DLL file to your PHP extensions directory.
    3. Rename the file to php_tideways_xhprof.so or php_tideways_xhprof.dll.
    4. Add "extension=tideways_xhprof" to php.ini.

Tideways XHProf

Here is a small sample of a Tideways XHProf usage:

```php
<?php

tideways_xhprof_enable(TIDEWAYS_XHPROF_FLAGS_MEMORY | TIDEWAYS_XHPROF_FLAGS_CPU);

// Your code to profile here:
for ($i = 0; $i < 10; $i++) {
    echo fibonacci($i) . "\n";
}

// This file name format, i.e. time().AppName.xhprof is IMPORTANT for most xhprof viewers!
file_put_contents('./' . time() . '.fibonacci.xhprof', serialize(tideways_xhprof_disable()));

function fibonacci($number)
{
    if ($number == 0) {
        return 0;
    } else
        if ($number == 1) {
            return 1;
        } else {
            return (fibonacci($number - 1) + fibonacci($number - 2));
        }
}
```

Tideways XHProf

The result for now is just a JSON-file, but we may follow this steps on Linux to "beautify" that result:

1. Go to https://github.com/sugarcrm/xhprof-viewer
2. Read instructions ☺.
3. Or just do the following:
    1. "git clone https://github.com/sugarcrm/xhprof-viewer.git"
    2. "composer update"
    3. Install Graphviz for Windows (or "sudo apt install graphviz" for Linux).
    4. Move all contents of "xhprof-viewer" to your web server DocumentRoot.
    5. In "config.php" point "profile_files_dir" to any empty folder.
    6. Copy your .xhprof files to that folder.
    7. Open http://127.0.0.1 in browser.
    8. Enjoy ☺.

Tideways XHProf

SugarCRM XHProf Viewer

Back To List

Function Calls | SQL Queries 0 | Elastic Queries

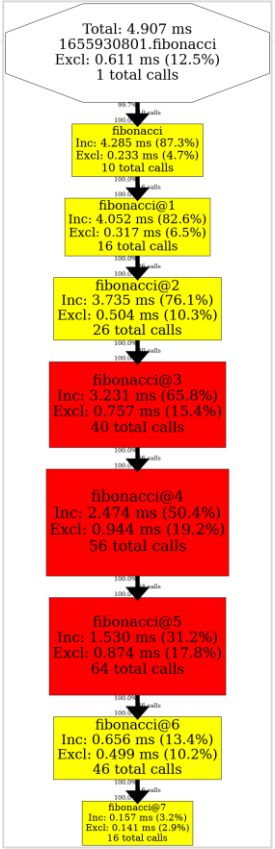
Top-Level Report

Search Functions Here

Display All | View Full Callgraph

Function Name	SQL (total / per func.)		Calls		Incl. Wall Time (µs)		Excl. Wall Time (µs)		Incl. CPU (µs)		Excl. CPU (µs)		Incl. MemUse (B)		Excl. MemUse (B)		Incl. PeakMemUse (B)		Excl. PeakMemUse (B)	
main()	0	0	1	0.4%	4,907	100.0%	611	12.5%	0	N/A%	0	N/A%	2,160	100.0%	240	11.1%	0	N/A%	0	N/A%
fibonacci	0	0	10	3.6%	4,285	87.3%	233	4.7%	0	N/A%	0	N/A%	1,792	83.0%	208	9.6%	0	N/A%	0	N/A%
fibonacci@1	0	0	16	5.8%	4,052	82.6%	317	6.5%	0	N/A%	0	N/A%	1,584	73.3%	208	9.6%	0	N/A%	0	N/A%
fibonacci@2	0	0	26	9.4%	3,735	76.1%	504	10.3%	0	N/A%	0	N/A%	1,376	63.7%	208	9.6%	0	N/A%	0	N/A%
fibonacci@3	0	0	40	14.4%	3,231	65.8%	757	15.4%	0	N/A%	0	N/A%	1,168	54.1%	208	9.6%	0	N/A%	0	N/A%
fibonacci@4	0	0	56	20.1%	2,474	50.4%	944	19.2%	0	N/A%	0	N/A%	960	44.4%	208	9.6%	0	N/A%	0	N/A%
fibonacci@5	0	0	64	23.0%	1,530	31.2%	874	17.8%	0	N/A%	0	N/A%	752	34.8%	208	9.6%	0	N/A%	0	N/A%
fibonacci@6	0	0	46	16.5%	656	13.4%	499	10.2%	0	N/A%	0	N/A%	544	25.2%	208	9.6%	0	N/A%	0	N/A%
fibonacci@7	0	0	16	5.8%	157	3.2%	141	2.9%	0	N/A%	0	N/A%	336	15.6%	208	9.6%	0	N/A%	0	N/A%
fibonacci@8	0	0	2	0.7%	16	0.3%	16	0.3%	0	N/A%	0	N/A%	128	5.9%	128	5.9%	0	N/A%	0	N/A%
tideways_xhprof_disable	0	0	1	0.4%	11	0.2%	11	0.2%	0	N/A%	0	N/A%	128	5.9%	128	5.9%	0	N/A%	0	N/A%

Display All



Commonly Used PHP Extensions

Disclaimer: вы смотрите просто запись лекции, это **НЕ** специально подготовленный видеокурс!