

Installing the MongoDB PHP Extension

In this section we cover the steps to install the MongoDB PHP extension on Windows and Linux/Mac. It is highly recommended for both Linux and Mac users to use [PECL](#) to install ext/mongodb. *PECL* stands for PHP Extension Community Library. It is a repository which contains PHP extensions written in C language which must be compiled and enabled in the php.ini file. For a Windows installation, on the other hand, all you need to do is to install the appropriate *.dll file, and enable the extension in the php.ini file.

ext/mongodb Installation Using PECL

The [pecl](#) command is used for low level PHP engine packaging and distribution. This system is shared with [PEAR](#) (PHP Extension and Application Repository). The latter contains libraries of functions written in PHP and is usually included in a standard PHP installation.

If, for some reason, the pecl command is not available on your system, [install PEAR](#), and also the PHP development library appropriate for your server (e.g. php-dev for Debian/Ubuntu). It should be noted that when installing PHP extensions written in C, *pecl* will download the source and compile. This means you also need to have a C compiler available (e.g. gcc).

To compile ext/mongodb using PECL, enter this command:

```
sudo pecl install mongodb
```

The results are shown here:


```
fred@fred-linux: ~  
fred@fred-linux:~$ sudo pecl install mongodb  
[sudo] password for fred:  
WARNING: channel "pecl.php.net" has updated its protocols, use "pecl  
to update  
downloading mongodb-1.5.0.tgz ...  
Starting to download mongodb-1.5.0.tgz (1,054,840 bytes)  
....done: 1,054,840 bytes  
423 source files, building  
running: phpize  
Configuring for:  
PHP Api Version:          20160303  
Zend Module Api No:       20160303  
Zend Extension Api No:    320160303  
building in /tmp/pear/temp/pear-build-root1Uuvoi/mongodb-1.5.0  
running: /tmp/pear/temp/mongodb/configure --with-php-config=/usr/lo  
checking for grep that handles long lines and -e... /bin/grep  
checking for egrep... /bin/grep -E  
checking for a sed that does not truncate output... /bin/sed  
checking for cc... cc  
checking whether the C compiler works... yes  
checking for C compiler default output file name... a.out  
checking for suffix of executables...  
checking whether we are cross compiling... no  
checking for suffix of object files... o  
checking whether we are using the GNU C compiler... yes  
checking whether cc accepts -g... yes  
checking for cc option to accept ISO C89... none needed  
checking how to run the C preprocessor... cc -E  
checking for icc... no  
checking for suncc... no  
checking whether cc understands -c and -o together... yes  
checking for system library directory... lib  
checking if compiler supports -R... no
```

When the compile process is finished, make a note of where the new extension binary is placed:

```
Build process completed successfully
Installing '/usr/local/lib/php/extensions/no-debug-non-zts-20160303
install ok: channel://pecl.php.net/mongodb-1.5.0
configuration option "php_ini" is not set to php.ini location
You should add "extension=mongodb.so" to php.ini
fred@fred-linux:~$
```

Look in the *php.ini* file for your installation. If the directory indicated by the `extension_dir` directive does not match, you will first need to move the newly installed driver to that directory. If you are not sure where your *php.ini* file resides, use `phpinfo()` or `php -i` as mentioned above.

To activate the driver, add the following line to the `php.ini` file:

```
 fred@fred-linux: ~  
extension=mongodb.so  
extension=sqlsrv.so  
extension=pdo_sqlsrv.so  
extension=yaml.so  
display_errors=on  
error_reporting=E_ALL  
  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~
```

"usr/local/lib/php.ini" 6 lines, 123 characters

You can then confirm that the driver is operational by again running `php -i` from the command line, or `phpinfo()` from a PHP script on a web server. The output from `php -i | grep mongo` is shown

here:

```
fred@fred-linux: ~  
fred@fred-linux:~$ php -i |grep mongo  
mongodb  
libmongoc bundled version => 1.11.0  
libmongoc SSL => enabled  
libmongoc SSL library => OpenSSL  
libmongoc crypto => enabled  
libmongoc crypto library => libcrypto  
libmongoc crypto system profile => disabled  
libmongoc SASL => disabled  
libmongoc ICU => enabled  
libmongoc compression => enabled  
libmongoc compression snappy => disabled  
libmongoc compression zlib => enabled  
mongodb.debug => no value => no value  
fred@fred-linux:~$
```

ext/mongodb Installation on Windows

Installing the MongoDB PHP driver on Windows is actually quite simple:

1. Download the correct pre-compiled *.dll file from <http://pecl.php.net/package/mongodb>.
2. Add this line to your installation's php.ini file:
extension=php_mongodb

The most difficult part is deciding *which* of the many pre-compiled *.dll files to use. From the <http://pecl.php.net/package/mongodb> page, choose the most appropriate MongoDB PHP Windows driver version (most likely the latest), and click on the link for *DLL*.



- [Home](#)
- [News](#)

Documentation:

- [Support](#)

Downloads:

- [Browse Packages](#)
- [Search Packages](#)
- [Download Statistics](#)

[Top Level](#) :: [Database](#)

Package Summary	
Summary	MongoDB driver for PHP
Maintainers	Derick Rethans < derick@php.net > (lead) [wishlist] Jeremy Mikola < jmikola@gmail.com > (lead) [details] Hannes Magnusson < bjori@php.net > (lead) [details] Katherine Walker (developer) [details]
License	Apache License
Description	The purpose of this driver is to provide exceptional performance for PHP and MongoDB, implementing only fundamental and performance-critical features necessary to build a fully-functional MongoDB driver.
Homepage	http://docs.mongodb.org/ecosystem/drivers/php/

[[Latest Tarball](#)]

[[Browse Source](#)]

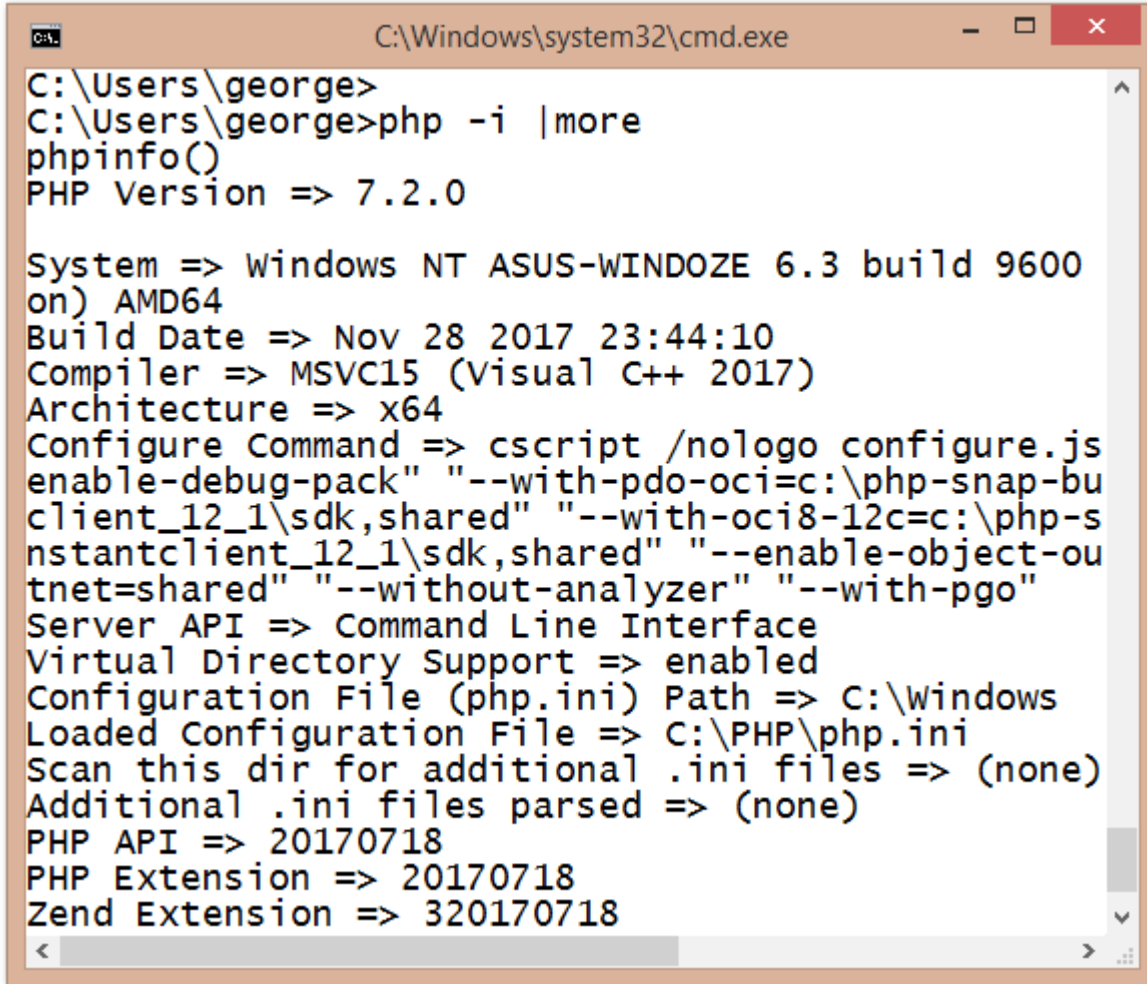
Available Versions			
Version	State	Release Date	Download
1.5.1	stable	2018-07-09	mongodb-1.5.1.tar.gz
1.5.0	stable	2018-06-26	mongodb-1.5.0.tar.gz

If you scroll down to the bottom of the DLL page, you are presented with several different versions of the *.dll file. In order to select the correct version, you need to find out some configuration information on your PHP/Windows installation:

- What version of PHP are you running (5.6, 7.0, 7.1, 7.2, etc.)?
- Is your PHP architecture 32 bit or 64 bit?
- Is [thread safety](#) enabled in the version of PHP that is installed?

You can find out the PHP version and architecture from either the command line, or from a browser. If you plan to access MongoDB through a web server running on your Windows server, create a script as follows: `<?php phpinfo(); ?>`. If you plan to access MongoDB from the command line only,

open a command prompt and enter `php -i |more`, which should look something the illustration shown here:



```
C:\Users\george>
C:\Users\george>php -i |more
phpinfo()
PHP Version => 7.2.0

System => Windows NT ASUS-WINDOZE 6.3 build 9600
on) AMD64
Build Date => Nov 28 2017 23:44:10
Compiler => MSVC15 (Visual C++ 2017)
Architecture => x64
Configure Command => cscript /nologo configure.js
enable-debug-pack" "--with-pdo-oci=c:\php-snap-bu
client_12_1\sdk,shared" "--with-oci8-12c=c:\php-s
nstantclient_12_1\sdk,shared" "--enable-object-ou
tnet=shared" "--without-analyzer" "--with-pgo"
Server API => Command Line Interface
Virtual Directory Support => enabled
Configuration File (php.ini) Path => C:\Windows
Loaded Configuration File => C:\PHP\php.ini
Scan this dir for additional .ini files => (none)
Additional .ini files parsed => (none)
PHP API => 20170718
PHP Extension => 20170718
Zend Extension => 320170718
```

Once you've determined the version of PHP and its architecture, here is a simple thread safety guideline which will assist you in choosing which version of the MongoDB Windows PHP Driver to use:

<i>Running MongoDB from a Web Server?</i>	<i>Running PHP using FastCGI?</i>	<i>Version to Use</i>
Yes	Yes	Non Thread Safe
Yes	No	Thread Safe
No	N/A	Non Thread Safe

Now you can add the line mentioned above to the `php.ini` file:

```
extension=php_mongodb
```

If you get an error when trying to use the newly installed MongoDB driver, try another driver.

The syntax `extension=xxx.dll` or `extension=xxx.so` has been deprecated. The new syntax is to omit `dll` or `so`. In the case of `ext/mongodb`, all you need to enter in the `php.ini` file is `extension=php_mongodb`. To confirm the driver is working, as mentioned above, from a command prompt simply run this command:

```
php -i |more
```

If the MongoDB driver is not working, an error message will be displayed immediately at the top.

If you get this error ...

Warning: PHP Startup: Unable to load dynamic library 'php_mongodb.dll'

(tried: C:\path\to\php\ext\php_mongodb.dll (The specified module could not be found.) ...

... try different versions of the driver. If still not successful try older versions of the driver.

Installing the PHP Library for MongoDB (PHPLIB)

Once you have installed the low level extension ext/mongodb, the next step is to install the [PHP Library for MongoDB](#) which we will abbreviate as *PHPLIB*. Although it is not mandatory to install this library, directly using the classes provided by ext/mongodb would be tedious and require many lines of code. Why not use a library which is expressly designed to leverage ext/mongodb and provides a rich set of classes and functionality which would otherwise take you days or weeks to accomplish?

Although you could simply download the library directly from <https://github.com/mongodb/mongo-php-library>, it's probably best to install using *Composer*. There three distinct advantages to using Composer:

- Any dependencies this library has are automatically resolved
- Composer can also perform updates
- Includes an *auto-loader* which saves you having to use *include* or *require* statements in your PHP code

To obtain a copy of Composer, follow this link: <https://getcomposer.org/composer.phar>. To install *PHPLIB* proceed as follows:

1. Open a terminal window (or command prompt)
2. Change to the directory containing your application source code
3. Use *Composer* to install PHPLIB:

```
php composer.phar require mongodb/mongodb
```

You will notice, from the screenshot below, several things happen:

- A vendor folder is created
- *PHPLIB* is installed
- The Composer control file composer.json is created for future reference


```
fred@fred-linux: ~/Desktop/Repos/MongoDB-Quick-Start-Guide-Doug/Source
fred@fred-linux:~/Desktop/Repos/MongoDB-Quick-Start-Guide-Doug/Source$
php composer.phar require mongodb/mongodb
Using version ^1.4 for mongodb/mongodb
./composer.json has been created
Loading composer repositories with package information
Updating dependencies (including require-dev)
Package operations: 1 install, 0 updates, 0 removals
  - Installing mongodb/mongodb (1.4.0): Downloading (100%)
Writing lock file
Generating autoload files
fred@fred-linux:~/Desktop/Repos/MongoDB-Quick-Start-Guide-Doug/Source$
```

In order to start using *PHPLIB*, all you need to do is to add this command at the beginning of your code:

```
require /path/to/source/vendor/autoload.php;
```