How to build PHP on Windows

for PHP 7.2 or newer

Requirements

Microsoft Visual Studio 2017

Creater workspace folder

C:\php-sdk

Download PHP SDK from GitHub

https://github.com/Microsoft/php-sdk-binary-tools

Extract PHP SDK to C:\php-sdk

C:\php-sdk\bin
C:\php-sdk\lib
C:\php-sdk\msys2
C:\php-sdk\pgo
...

Download PHP Source

https://windows.php.net/download/

Invoke VS 2017 x64 Native Tools Command Prompt with Administrator Privileges

C:\Windows\System32> **cd C:\php-sdk**

C:\php-sdk> **phpsdk-starter.bat -c vc15 -a x64**
[vcvarsall.bat] Environment initialized for: 'x64'

PHP SDK 2.1.10

OS architecture: 64-bit
Build architecture: 64-bit
Visual C++: 15
PHP-SDK path: C:\php-sdk

C:\php-sdk
$ **phpsdk_buildtree php-dev**

Extract PHP Source

C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src

Download Dependencies

C:\php-sdk
$ **cd C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src**

C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src
$ **phpsdk_deps -u**

Configuration: 7.2-vc15-x64-stable

Processing package ICU-63.1-vc15-x64.zip
Processing package apache-2.4.28-vc15-x64.zip
…
Processing package zlib-1.2.11-vc15-x64.zip
Processing package libsodium-1.0.16-vc15-x64.zip
Updates performed successfully.
Old dependencies backed up into 'C:\php-sdk\php-dev\vc15\x64\deps.201902091501'.

Compile

C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src
$ **C:\php-sdk\phpsdk-vc15-x64.bat**
[vcvarsall.bat] Environment initialized for: 'x64'

PHP SDK 2.1.10

OS architecture: 64-bit
Build architecture: 64-bit
Visual C++: 15
PHP-SDK path: C:\php-sdk

C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src
$ **buildconf**
Rebuilding configure.js
Now run 'configure --help'

C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src

$ **configure --disable-all --enable-cli**
$ **configure --disable-all --enable-cli --disable-zts**
PHP Version: 7.2.15

Saving configure options to config.nice.bat
Checking for cl.exe ... <in default path>
Detected compiler MSVC15 (Visual C++ 2017)
Detected 64-bit compiler
Checking for link.exe ... C:\Program Files (x86)\Microsoft Visual Studio\2017\Enterprise\VC\Tools\MSVC\14.16.27023\bin\HostX64\x64
Checking for nmake.exe ... <in default path>
Checking for lib.exe ... <in default path>
Checking for bison.exe ... <in default path>
Checking for sed.exe ... <in default path>
Checking for re2c.exe ... <in default path>
Detected re2c version 1.1.1
Checking for zip.exe ... <in default path>

```
Checking for lemon.exe ... <in default path>
Checking for mc.exe ... C:\Program Files (x86)\Windows
Kits\10\bin\10.0.17763.0\x64
Checking for mt.exe ... C:\Program Files (x86)\Windows
Kits\10\bin\10.0.17763.0\x64
Enabling multi process build

Build dir: C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src\x64\Release_TS
PHP Core: php7ts.dll and php7ts.lib

Checking for wspiapi.h ... <in default path>
Enabling IPv6 support
Enabling SAPI sapi\cli
Checking for library edit_a.lib;edit.lib ... ..\deps\lib\edit_a.lib
Checking for editline/readline.h ... C:\php-sdk\php-
dev\vc15\x64\deps\include
Enabling extension ext\date
Enabling extension ext\pcre
Enabling extension ext\reflection
Enabling extension ext\spl
Checking for timelib_config.h ... ext/date/lib
Enabling extension ext\standard

Creating build dirs...
Generating files...
Generating Makefile
Generating main/internal_functions.c
Generating main/config.w32.h
Generating phpize
Done.

...

Type 'nmake' to build PHP
```

C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src
$ **nmake**

Output

C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src\x64\Release_TS

How to build PHP Extension on Windows

For PHP 7.2+

Follow this post andmake sure you can build PHP on Windows

https://medium.com/@erinus/how-to-build-php-on-windows-a7ad0a87862a

(This is the above post)

Take APCu for Example

Download from PECL
https://pecl.php.net/get/APCu

Extract Source to extensions folder

C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src\ext\apcu

Invoke VS 2017 x64 Native Tools Command Prompt with Administrator Privileges

C:\Windows\System32> **cd C:\php-sdk**

C:\php-sdk> **phpsdk-starter.bat -c vc15 -a x64**
[vcvarsall.bat] Environment initialized for: 'x64'

PHP SDK 2.1.10

OS architecture: 64-bit
Build architecture: 64-bit
Visual C++: 15
PHP-SDK path: C:\php-sdk

C:\php-sdk
$ **cd C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src**

C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src
$ **nmake clean**

C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src
$ **buildconf --force**
Rebuilding configure.js
Now run 'configure --help'

C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src
$ **configure --disable-all --enable-cli --enable-apcu=shared**

PHP Version: 7.2.15

Saving configure options to config.nice.bat
Checking for cl.exe ... <in default path>
Detected compiler MSVC15 (Visual C++ 2017)
Detected 64-bit compiler
Checking for link.exe ... C:\Program Files (x86)\Microsoft Visual Studio\2017\Enterprise\VC\Tools\MSVC\14.16.27023\bin\HostX64\x64
Checking for nmake.exe ... <in default path>
Checking for lib.exe ... <in default path>
Checking for bison.exe ... <in default path>
Checking for sed.exe ... <in default path>
Checking for re2c.exe ... <in default path>
Detected re2c version 1.1.1

```
Checking for zip.exe ... <in default path>
Checking for lemon.exe ... <in default path>
Checking for mc.exe ... C:\Program Files (x86)\Windows
Kits\10\bin\10.0.17763.0\x64
Checking for mt.exe ... C:\Program Files (x86)\Windows
Kits\10\bin\10.0.17763.0\x64
Enabling multi process build

Build dir: C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src\x64\Release_TS
PHP Core: php7ts.dll and php7ts.lib

Checking for wspiapi.h ... <in default path>
Enabling IPv6 support
Enabling SAPI sapi\cli
Checking for library edit_a.lib;edit.lib ... ..\deps\lib\edit_a.lib
Checking for editline/readline.h ... C:\php-sdk\php-
dev\vc15\x64\deps\include
Enabling extension ext\apcu [shared]
Enabling extension ext\date
Enabling extension ext\pcre
Enabling extension ext\reflection
Enabling extension ext\spl
Checking for timelib_config.h ... ext/date/lib
Enabling extension ext\standard

Creating build dirs...
Generating files...
Generating Makefile
Generating main/internal_functions.c
Generating main/config.w32.h
Generating phpize
Done.

...
```

Type ‘nmake’ to build PHP

C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src
$ **nmake**

Output

C:\php-sdk\php-dev\vc15\x64\php-7.2.15-src\x64\Release_TS**php_apcu.dll**

Installing a PHP extension on Windows ¶

There are two ways to load a PHP extension on Windows: either compile it into PHP, or load the DLL. Loading a pre-compiled extension is the easiest and preferred way.

To load an extension, it has to be available as a *.dll* file on the system. All the extensions are automatically and periodically compiled by the PHP Group (see next section for the download).

To compile an extension into PHP, please refer to the building from source documentation.

To compile a standalone extension (aka a DLL file), please refer to the building from source documentation. If the DLL file is available neither with the PHP distribution nor in PECL, it may be necessary to compile it before the extension can be used.

Where to find an extension? ¶

PHP extensions are usually called *php_*.dll* (where the star represents the name of the extension), and they are located under the *PHP\ext* folder.

PHP ships with the extensions most useful to the majority of developers. They are called *bundled* extensions.

However, if the bundled extensions do not provide the needed functionality, one extension that does may still be found in »° PECL. The PHP Extension Community Library (PECL) is a repository for PHP Extensions, providing a directory of all known extensions and hosting facilities for downloading and developing PHP extensions.

If an extension has been developed for particular uses, it may be hosted on PECL so that others with the same needs can benefit from it. A nice side effect is that it's a good chance to receive feedback, (hopefully) thanks, bug reports and even fixes/patches. Before submitting an extension for hosting on PECL, please read »° PECL submit.

Which extension to download? ¶

Many times, there will be several versions of each DLL available:

- Different version numbers (at least the first two numbers should match)
- Different thread safety settings
- Different processor architecture (x86, x64, ...)
- Different debugging settings
- etc.

Keep in mind that the extension settings should match all the settings of the PHP executable being used. The following PHP script will tell *all* about the PHP settings:

Example #1 phpinfo() **call**

```
<?php
phpinfo();
?>
```

Or from the command line, run:

```
drive:\path\to\php\executable\php.exe -i
```

Loading an extension ¶

The most common way to load a PHP extension is to include it in the *php.ini* configuration file. Please note that many extensions are already present in the *php.ini* and that the semicolon only needs to be removed to activate them.

Note that, as of PHP 7.2.0, the extension name may be used instead of the extension's file name. As this is OS-independent and easier, especially for newcomers, it becomes the recommended way of specifying extensions to load. File names remain supported for compatibility with prior versions.

```
;extension=php_extname.dll
extension=php_extname.dll
; As of PHP 7.2.0, prefer:
extension=extname
zend_extension=another_extension
```

However, some web servers are confusing because they do not use the *php.ini* located alongside the PHP executable. To find out where the actual *php.ini* resides, look for its path in phpinfo():

```
Configuration File (php.ini) Path  C:\WINDOWS
Loaded Configuration File   C:\Program Files\PHP\8.2\php.ini
```

After activating an extension, save *php.ini*, restart the web server, and check phpinfo() again. The new extension should now have its own section.

Resolving problems ¶

If the extension does not appear in phpinfo(), the logs should be checked to learn where the problem comes from.

If PHP is being used from the command line (CLI), the extension loading error can be read directly on the screen.

If PHP is being used with a web server, the location and format of the logs vary depending on the software. Please read the web server documentation to locate the logs, as it has nothing to do with PHP itself.

Common problems are the location of the DLL and the DLLs it depends on, the value of the "extension_dir" setting inside *php.ini* and compile-time setting mismatches.

If the problem lies in a compile-time setting mismatch, probably the DLL downloaded is not the right one. Try downloading the extension again with the proper settings. Again, phpinfo() can be of great help.