Tue 18 Sep 2007

Creating a PHP 5 Extension with Visual C++ 2005

Posted by Matthew E under Productivity, Programming

This article describes the steps to create a custom PHP extension DLL for the Windows platform. The Zend API documentation that comes with PHP 5 on Windows (see php_manual_en.chm) does a good job explaining how to write extension methods, parse method parameters, and return values. But there is not currently a good step-by-step tutorial on how to get your first extension project up and running on Windows. The aim of this article is to fill that gap.

Prerequisites

- **Visual Studio 2005**
  You can alternately use the free Visual C++ Express Edition or the VC++ 8 compiler in the Windows SDK v6.0 if you’re a makefile master. Visual Studio 2003 (VC++ 7) will probably work just fine, but some of the project configuration steps will be different than what is explained here
- **A web server**
  For this article I used Sambar Server 7.0. Any HTTP server that can run the PHP 5.x ISAPI extension (php5isapi.dll) will do
- **PHP 5 binaries**, installed and configured for your server
  I used PHP 5.2.4. Using the windows installer package (.msi file) makes configuration easier
- **PHP 5 source code**
  DONT PANIC! You do not need to build PHP from source, just get the source that matches your binary version. You will need an extraction utility like WinRAR or ZipGenius that can handle .tar archives.

Conventions

Italics denote file paths and names. *C:\Server\PHP\dev*
Screenshot icons provide links to images showing how dialogs are configured.
Document icons provide links to sample code files.

Configuring the Environment

I will not discuss installing and running the HTTP web server. If you do not already have IIS available on your machine, I recommend the Sambar Server.

If you do not already have PHP 5 installed, download it from php.net. If you download and run the MSI installer package, it will configure your web server.

After you’ve got PHP 5 installed and configured for your server, download and extract the complete PHP 5 source code, also from php.net. **Caution:** Do NOT extract the source archive over top of your existing binary installation. This article’s example setup includes the PHP 5 binaries installed to *C:\Server\PHP5*, and the

source code extracted to *C:\Server\PHP5Src*.

Creating the Project

In Visual Studio 2005, create a new Visual C++ Win32 project using the project template. For this example, I named the project CustomExt. When the Win32 Application Wizard appears, click **Application Settings** (on the left) and select DLL as the application type. Click Finish to create the project.

I recommend following along step-by-step, but you can also download the complete project source (zipped). You’ll still need to follow the instructions for changing the directories to match your system.

Change Default C++ Options

Bring up the Project Properties dialog, and make sure the Debug configuration is active. Under **Configuration Properties > General**, change the Character Set to “Use Multi-Byte Character Set”.

Then under **Configuration Properties > C/C++ > Code Generation**, change the following options:

- Enable String Pooling to “Yes (/GF)”
- Enable Minimal Rebuild to “No”
- Basic Runtime Checks to “Default”
- Runtime Library to “Multi-threaded Debug (/MTd)”

Under **Configuration Properties > C/C++ > General**, change the following:

- Debug Information Format to “Program Database (/Zi)”
- Detect 64-bit Portability Issues to “No”

Be sure to click **Apply** to save the settings.

Set the INCLUDE Paths

Still on the Project Properties dialog unders **Configuration Properties > C/C++ > General**, in the Additional Include Directories, add the following paths, replacing *C:\Server\PHP5Src* with the location on your machine where you extracted the source code.

```
C:/Server/PHP5Src/main
C:/Server/PHP5Src/Zend
C:/Server/PHP5Src/TSRM
C:/Server/PHP5Src/regex
C:/Server/PHP5Src
```

Again, be sure to click Apply to save the settings as you go along.

Set the Preprocessor Definitions

On the Project Properties dialog, under **Configuration Properties > C/C++ > Preprocessor**, add the following definitions:

```
ZEND_DEBUG=0

ZTS=1
```

```
ZEND_WIN32

PHP_WIN32
```

Note: Do not be tempted to change ZEND_DEBUG to 1, even for this Debug build, as this will prevent your extension from being loaded into the pre-built PHP binaries installed on your system.

Set the Linker Options

On the Project Properties dialog, under **Configuration Properties > Linker > General** add the path to the *C:\Server\PHP\dev* directory to the Additional Library Directories. This is the directory underneath your installed **PHP binaries** (not the extracted source). This directory should contain *php5ts.lib*.

Under **Configuration Properties > Linker > Input**, add *php5ts.lib* to the Additional Dependecies.

By convention, php extensions start with php_, so under **Configuration Properties > Linker > General**, for the Output File option, I changed the output name to *php_custom_ext.dll*. This is entirely optional.

The Extension Code

Replace the contents of stdafx.h with the following, or download it:

```
#pragma once
/* PHP Extension headers */
/* include zend win32 config first */
#include "zend_config.w32.h"
/* include standard header */
#include "php.h"
```

Replace the contents of *ProjectName*.cpp (in this example CustomExt.cpp) with the following (or download it):

```
#include "stdafx.h"/* declaration of functions to be exported */
ZEND_FUNCTION(DoubleUp);
/* compiled function list so Zend knows what's in this module */
zend_function_entry CustomExtModule_functions[] = {
   ZEND_FE(DoubleUp, NULL)
    {NULL, NULL, NULL}
};

/* compiled module information */
zend_module_entry CustomExtModule_module_entry = {
    STANDARD_MODULE_HEADER,
    "CustomExt Module",
    CustomExtModule_functions,
    NULL, NULL, NULL, NULL, NULL,
    NO_VERSION_YET, STANDARD_MODULE_PROPERTIES
};

/* implement standard "stub" routine to introduce ourselves to Zend */
ZEND_GET_MODULE(CustomExtModule)
/* DoubleUp function */
/* This method takes 1 parameter, a long value, returns
   the value multiplied by 2 */
ZEND_FUNCTION(DoubleUp){
    long theValue = 0;
    if (zend_parse_parameters(ZEND_NUM_ARGS() TSRMLS_CC,
                              "l", &theValue) == FAILURE){
```

```
        RETURN_STRING("Bad parameters!", true);
    }
    theValue *= 2;
    RETURN_LONG(theValue);
}
```

You should now be able to build the project without any errors. Be careful when renaming **CustomExtModule** or **DoubleUp**, and you must replace every instance in the file, or the Zend macros will yield nearly indecipherable compiler errors.

Enable the extension

The first step is to locate the *\ext* directory beneath your **PHP 5 binaries** (not the directory where you extracted the PHP source code). On my sample system this is *C:\Server\PHP\ext*. Copy the DLL that you just built to this directory. If you are going to attach a debugger to your extension, you will also want to copy the .pdb files.

The second step is to configure PHP to load your extension DLL. You do this by modifying PHP.ini, adding a

```
extension=php_ext_name.dll
```

line to the Dynamic Extensions section, like the following.

```
;;;;;;;;;;;;;;;;;;;;;
; Dynamic Extensions ;
;;;;;;;;;;;;;;;;;;;;;
[CustomExt]
extension=php_custom_ext.dll
```

Verify that the extension can be loaded

Once you’ve edited and saved php.ini, you may need to restart the HTTP server in order to pick up the new module. After the server is restarted, browse to the phpinfo.php web page. If you do not already have a phpinfo.php page, create one using the following text, and save it to one of your virtual directories:

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

Now browse to http://localhost/*virtualpath*/phpinfo.php, and look for the **Additional Modules** section. Your new extension should now be seen in the listing. If your extension is not listed, you may need to resave php.ini and restart the server. Also, make sure that the php.ini file that your server is using is the same ini file that you just edited. On most servers, php.ini is located in the PHP 5 installation directory, but your server may be different. To make certain, look at the information at the top of the page for the **Loaded Configuration File** entry, which shows the full path to the ini file PHP is currently using.

Write a test page

Copy the following code to testext.php (or download it) , save it under one of your web server’s virtual directories.

```
<?php
$value = 14;
$result = DoubleUp($value);
print "Calling DoubleUp($value) returned $result";
```

```
?>
```

You can now browse to http://localhost/*virtualpath*/testext.php.

Debugging the extension

In order to debug the extension you need to attach the debugger to the running instance of the web server process. For IIS, this is w3wp.exe. For the Sambar server, this is sambar70\bin\server.exe. On the Project Properties dialog, under **Configuration Properties > Debugging** , set the following values:

- Command to the full path to the server executable
- Attach to “Yes”
- Debugger Type to “Native Only”

Stop you web server and make certain that you have copied the latest version of the extension DLL and the .pdb files to the *\ext* directory, then restart the web server. Set a breakpoint in the DoubleUp method and execute **Debug > Start Debugging**. You may get a message that the web server executable was not built with debugging information. Ignore this warning and click Yes to continue debugging.

Open a web browser and browse to your test page that calls the DoubleUp method. Your breakpoint should be hit. Step through the code, and be sure to use **Debug > Continue** (F5) so that the web server doesn’t hang.

To stop debugging, go to **Debug > Detach All** or stop the web server.

Where to from here?

The Zend API section of the PHP documentation provides a wealth of detailed information on the macros and functions available for extension development. And now that you’ve got your own extension up and running, you can finally make use of it! Some areas to explore are enabling your extension to accept configuration parameters from php.ini, and creating methods that accept multiple parameters, and/or optional parameters.

42 Responses to “ Creating a PHP 5 Extension with Visual C++ 2005 ”

Comments:

1. *Peter Rudloff* says:
   September 20th, 2007 at 9:27 am

   I will try it and hope best. Thanks for your work
   Peter

2. *Peter Rudloff* says:
   October 4th, 2007 at 6:13 am

   .. and today I did it. No problem, just worked. Thanks a lot. Your work saved hours!

3. *Matthew E* says:

October 4th, 2007 at 8:30 am

I’m glad it helped. The whole ZEND_DEBUG=0 vs ZEND_DEBUG=1 issue cost me half a morning.

4. *MinHe* says:
   October 6th, 2007 at 4:29 am

   Thanks a lot!
   It works!

5. *ci* says:
   October 7th, 2007 at 7:36 am

   learning

6. *aling* says:
   October 14th, 2007 at 8:01 am

   why use STL, VC++2005 linker gives a lot of ERROR LINK 2005 as follows:

   libcpmtd.lib(string.obj) : error LNK2005: “public: void __thiscall std::_Container_base::_Orphan_all(void)const ” (?_Orphan_all@_Container_base@std@@QBEXXZ) already defined in AutoClasssify.obj
   libcpmtd.lib(locale0.obj) : error LNK2005: “void * __cdecl operator new(unsigned int,void *)” (??2@YAPAXIPAX@Z) already defined in AutoClasssify.obj
   libcpmtd.lib(locale0.obj) : error LNK2005: “void __cdecl operator delete(void *,void *)” (??3@YAXPAX0@Z) already defined in AutoClasssify.obj
   libcpmtd.lib(locale0.obj) : error LNK2005: “public: void __thiscall std::_Container_base::_Orphan_all(void)const ” (?_Orphan_all@_Container_base@std@@QBEXXZ) already defined in AutoClasssify.obj

7. *Jonathon* says:
   October 15th, 2007 at 5:24 pm

   Excellent tutorial! This is the first time I’ve ever gotten a PHP extension to compile and work, kudos!

   Now time to learn the ins and outs of extension writing. 

8. *Peter Rudloff* says:
   October 16th, 2007 at 7:52 am

   Next time write an example with 2 functions and transfer 2 parameters (at least in one of these 2 functions). This would save even more time. Do you have experience in the speed of the data exchange with a PHP-Extension. I want to use existing grafics code to create .jpg’s and could pass one string containing all commands rather than calling the extension 100 times. Is this a good idea? It would not need the example mentioned above! Thanks a lot
   Peter

9. *Matthew E* says:
   October 16th, 2007 at 8:23 am

   @Jonathon: Glad it helped you.

   @Peter: I don’t have any intimate knowledge of the speed of the extension interface. I usually use

extensions that take and return a small amount of data, but have to do a lot of work.
I also do not have experience using extensions to craft images. Back in my ASP.NET days, I used what are known as “HTTP Handlers” to generate on-the-fly GIF or PDF content. The src attribute of the IMG tag is set to a special URL, which is captured by the handler, returning the image content. You could probably take a similar approach with your PHP code. The main benefit of this is that the browser can process multiple simultaneous requests to the different image URLs, independent of the main page URL.

10. *Matthew E* says:
October 16th, 2007 at 9:42 am

@aling: There may be a better way to clean this up, but the following worked for me:

Just before your first STL header,
#define _HAS_ITERATOR_DEBUGGING 0
This will clear up 2 of your 4 link errors.

Then, go to Project Properties, Linker, Command Line, and add the following to the Additional options text field:
/FORCE:MULTIPLE

That’s not pretty, as it gives you a bunch of link warnings, but the resulting DLL did work just fine.

11. *Alex Solé* says:
November 23rd, 2007 at 6:44 pm

Great code. Fantastic and very useful.
Thanks a lot!

12. *dekoo* says:
November 24th, 2007 at 4:45 am

and how to write PHP extensions with cpp on Linux ?

13. *Evan* says:
November 27th, 2007 at 8:54 pm

I got the error “Invalid library (maybe not a PHP library)” until I added this line to the linker command line (**Configuration Properties** -> **Linker** -> **Command Line**):

```
/EXPORT:get_module
```

14. *Doug Logan* says:
December 1st, 2007 at 3:11 pm

Thanks for this tutorial!

I was able to get this code working with VC++ 2008 Express by downloading and updating the project.

I had to make one modification to zend_config.w32.h commenting out the line:
#define vsnprintf _vsnprintf

Other than that everything was good ;-).

I originally had some headaches trying to get it to work on VC++ 2008 following the directions. While all

the screens referenced in 2005 could easily be translated to 2008, something is definitely different (or I missed a step). I ended up with a ton of compilation errors.

15. *rasa* says:
December 7th, 2007 at 5:25 am

Nice, but why it's not working with 2 functions in dll? Or it should follow some different coding maner than that of for Unix? How to code properly more tnhan 1 function in dll?

16. *Matthew E* says:
December 7th, 2007 at 9:26 am

@rasa:
If you need to export more than 1 function, you have to make sure you add it to the array of zend_function_entry structs (CustomExtModule_functions array in this example).

@evan:
The macro definition of ZEND_GET_MODULE, in conjunction with the preprocessor defines being set up, should be handling the export of the get_module fn for you. But I'm glad you found a workaround.

17. *Digilee* says:
December 15th, 2007 at 11:21 pm

The example looks like exactly what I need, except that I need it for PHP4. Will it work if I just substitute PHP4 where you had PHP5? I'm going to try it, but I hope that you might have some suggestions.

Thanks,

Lee Mulcahy

18. *Digilee* says:
December 17th, 2007 at 3:49 am

Well, I tried it, and although it compiled without any errors, it causes my web page to hang. I tried similar instructions from another web site for PHP4/Visual Studio6, and translated them to VS 2003, and it worked fine, but it is C, not C++. I would really like to use C++ throughout, so if you have any suggestions, I would appreciate it

Lee

19. *Burkhard* says:
January 22nd, 2008 at 11:59 am

Thanks for the great and very helpful article!

Yet two suggestions:
1. If you get the compiler error "fatal error C1083: Cannot open precompiled header file: …", change the project property **Configuration Properties > C/C++ > Precompiled Headers > Create/Use Precompiled Headers** into **Not Using Precompiled Headers**.
2. If you get lots of linker errors like "error LNK2005: _getwchar already defined in CustomExt.obj", remove the file *stdafx.cpp* (created automatically by the wizard) from the project.

20. *trecool999* says:
January 25th, 2008 at 2:44 am

Compiling…
cl : Command line error D8016 : ‘/MTd’ and ‘/clr’ command-line options are incompatible
Build log was saved at “file://c:\Documents and Settings\Chris\My Documents\Visual Studio 2008\Projects\HW\HW\Debug\BuildLog.htm”
HW - 1 error(s), 0 warning(s)
========== Build: 0 succeeded, 1 failed, 0 up-to-date, 0 skipped ==========

I’ll I’ve had with this tutorial is errors… Different ones each time :’(.

21. *trecool999* says:
January 25th, 2008 at 2:57 am

I give up :(.
I’m using Microsoft Visual C++ 2008 Express BTW.

22. *Matthew E* says:
January 25th, 2008 at 9:28 am

You are getting the /MTd and /clr conflict error because the project has been set to compile managed (.NET) C++ code (aka C++/CLI). I haven’t used the express edition, so I don’t know what project types you are limited to. But you do *not* want to do this in managed code. Even if you started with a managed C++ project, you should still be able to turn it off in the project properties pages.

23. *Akhil* says:
February 10th, 2008 at 12:23 pm

Hi,

I got the error “Invalid library (maybe not a PHP library)” until I added this line to the linker command line (Configuration Properties -> Linker -> Command Line):
/EXPORT:get_module

Now I am not getting this error. But the dll is not loading. I checked with the test PHP application provided with the project. Can someone hlep me plz?

Thanks,
Akhil

24. *Akhil* says:
February 10th, 2008 at 12:33 pm

Hi,

I have posted my problem (last one - 23). I FORGOT TO MENTION THAT AM USING PHP4.

Any suggestion would be very helpful.

Thanks,
Akhil

25. *John* says:
February 20th, 2008 at 7:28 am

I got DLL to work with VS++2008 on my laptop running XP PRO. But when I copy the DLL to a work server running server 2003 RC2 it fails. Does the DLL only work on the computer you compile it on?

Thanks

26. *Matthew E* says:
February 20th, 2008 at 9:17 am

@John:
You may want to make sure you’ve got all the required DLLs on your server 2003 machine. The easiest way is to use the dependency walker in the SDK tools (depends.exe). If you haven’t installed VS2008 on your server, some stuff may be missing.

27. *Matthew E* says:
February 20th, 2008 at 9:21 am

@Akhil:
First make sure your function is being declared as a PHP-callable function with the proper macros. And double-check your phpinfo.php page to make sure the module is showing up under the Additional Modules section.

28. *Burkhard* says:
February 20th, 2008 at 11:17 am

Hi again.
I compiled, linked and tested the CustomExt example - everything was fine.
Then I added another source code file to the project and got a long list of ” already defined…” by the linker, all of math functions like _acosl etc., although I don’t use any of them.

Solution: Include the cmath header before all with
#include .

Probably, the php engine includes some math funtions (defined as inline) without a real definition (?), so the compiler is forced to generate the bodies automatically in every .obj file (thus twice in my case).

29. *Michael* says:
February 22nd, 2008 at 4:00 pm

This script is very good. Thanks for your work

30. *Michael* says:
February 22nd, 2008 at 4:02 pm

This script is very good. Thanks for your work!

31. *Nancy* says:
March 31st, 2008 at 2:10 pm

HI:

I TRY THE SAMPLE AND IT WORKS FINE !! I USED APACHE AND VISUAL STUDIO 2005

NOW I WANT TO LINK THIS EXTENSION WITH A LIBRARY CONTAINING FUNCTIONS I WANT TO USE. I GET A LINKER ERROR FOR “UNRESOLVED REFERENCES”. I CHECK MY SETTINGS AND DON’T FOUND CAUSES FOR THESE CONFLICTS.

ANY HELP?

32. *Alex* says:
April 10th, 2008 at 3:48 pm

Hello,

first of all thank you very much for this great tutorial!
The Extension itself, using it as is works fine on Windows Server 2003 with IIS and/or Apache as webserver.
But, when I create a method wich expects more than one parameter and call this method with the correct parameter type and count I got the errormessage "PHP Access Violation at *******". Is the parameter type wrong I'm getting the correct errormessage that it's the wrong parametertype but not if it's correct.
On IIS the errormessage appears, on Apache the server just hangs for a while and than I got another errormessage but it's at the end the same problem.

Is there any way to solve this problem?

33. *Matthew E* says:
April 11th, 2008 at 8:47 am

Alex:
I recommend referencing Chapter 46 of the PHP manual (Chapter 46. Zend API: Hacking the Core of PHP), especially the sections "Source Discussion" and "Accepting Arguments". Separate the call to ZEND_NUM_ARGS() out of the zend_parse_parameters() call so that you can assign the value to a variable. The you can follow the instructions for locally debugging the extension to see what is coming across. You may also want to use zend_get_parameters_array_ex in conjunction with the conver_to_xxx functions. (There are examples for this in the "Accepting Arguments" section)

34. *Jacques* says:
April 14th, 2008 at 2:54 pm

Your instructions are spot-on! Everything worked first time round. Great work!

35. *KP* says:
May 14th, 2008 at 11:51 am

Hi Matt,
php_custom_ext.dll is compiled without any problems from vis studio 2005.

I have added the extension to php.ini as instructed, and restarted the apache 2.2 server running on windows XP (uses fast-cgi, as per default zend core installation).

But, phpinfo() does not find this additional module at all. I know 100% that it is loading this php.ini file, as ive checked in phpinfo and also have made other config changes which have taken effect. Ive also checked apaches error logs and there are no references to php_custom_ext.

Any thoughts?

36. *KP* says:
May 14th, 2008 at 12:37 pm

Hi Matt,

I've realized that my php_custom_ext.dll is working fine if Apache is configured with PHP as a module, but it doesn't work when using FastCGI.

How can I get php_custom_ext.dll to work when PHP is running under FastCGI?

Thanks

37. *Andrey Wolk* says:
May 28th, 2008 at 1:48 am

It works!
Many thanks!
Very useful article.

38. *Cyberjupie* says:
June 10th, 2008 at 9:20 pm

Thank your for your article, Matthew. It works fine.

But, I have a litle troble,

I try to create some custom function that contain php function (ex. base64_decode, eval, etc..) , but i do not know how to write it properly. I just got error that i dont understand.

simply i just want to write the php extension that contain function like:

```
function my_custom($str) {
eval(base64_decode($str));
}
```

can you help me? (or any body here?)

thanks.

39. *Matthew E* says:
June 11th, 2008 at 10:58 am

@Cyberjupie:
You’ll need to change the arguments in zend_parse_parameters(). The example I provided takes a long value, hence the “l” argument list. For a single string, you need to pass “s”, and then provide output parameters for both the string and it’s length.

```
char* stringArg;
int stringArgLen;
if (zend_parse_parameters(ZEND_NUM_ARGS() TSRMLS_CC, “s”, &stringArg, &stringArgLen) ==
FAILURE)
```

40. *Cyberjupie* says:
June 16th, 2008 at 9:04 pm

Thanks, Mat. It helps me so far.

I have written some .dll extension contain internal function like php_printf, php_date, base64_encode, etc… It works fine.

But, now I get a little problem. Why I can’t find where is internal ‘eval’ source code that works like PHP eval() function, so I can use it in my code?
I know, ‘eval’ is not function, but if I want to write .dll that works similary with:

what should I use?

To explain more clearly, what my question is, so far I have this:
…
ZEND_FUNCTION(my_custom_eval)
{
char *str;
int str_len;

if (zend_parse_parameters(ZEND_NUM_ARGS() TSRMLS_CC, “s”, &str, &str_len) == FAILURE) {
return;
}

}

Something like php_printf, php_date, etc.., What function should I use to act like PHP ‘eval’ ?

Please, any one help me.

Thanxs.

41. *Cyberjupie* says:
June 16th, 2008 at 9:10 pm

upsss…., sory, I use php tags so the code doesn’t appear.
I want to write .dll that works similary with:

…. PHP
function my_custom_eval($str) {
eval($str);
}

…. PHP

thnx.

42. *Deserved* says:
October 21st, 2008 at 5:29 am

Bravo, very good work.

Leave a Reply

Name

Mail (will not be published)

Website

Security code (Required)*
To prove that you're not a bot, enter this code

XHTML: You can use these tags: <a href="" title=""> <abbr title=""> <acronym title=""> <b> <blockquote cite=""> <code> <i> <strike>

Submit Comment

Archived Entry

- **Post Date :**
- Tuesday, Sep 18th, 2007 at 8:08 am
- **Category :**
- Productivity and Programming
- **Do More :**
- You can leave a response, or trackback from your own site.

Free Trials

- **SlickEdit® 2008**
- **SlickEdit Core for Eclipse™**
- **SlickEdit Tools for Microsoft® Visual Studio®**

Recent Posts

- SlickEdit Macros for Everybody
- “Please” and “Thank You” Macro
- Our Nation’s First CTO
- The Politics of Features
- Too Busy Bailing to Plug the Leak

Categories

- Code Editors
- Dev Management
- Other
- Productivity
- Programming

- SlickEdit Games
- SlickEdit Products

Blogroll

- Blog Archives
- SlickEdit Forum
- SlickEdit Games
- SlickEdit Website

Search

Google Search

Feeds

Feed Subscriptions

Book