

Thursday, December 20, 2007

Getting tired of Home? Emmigrate to PHP City in Jamaica!

Yes, Cal Evans has [setup the paradise of PHP City](#) on the island of Jamaica. Granted, it's a Flash version of Jamaica, but Jamaica all the same. Haven't you ever wanted to visit Jamaica??? Then do it now!

<http://phpcity.myminicity.com>

If you really want to lend a hand, then visit this link everyday so we can overtake all the other piddly little settlements in Jamaica (like Rastafarie) and continue on to overpopulate the entire island. Commune with fellow PHP City citizens like Mayor McCal (our revered leader) or everyone else who commutes to PHP City daily (it has a 0 rate of Criminality, and 0 Unemployment!).

If you want to really help out, also visit the following:

<http://phpcity.myminicity.com/ind> (Get yourself a job and build up PHP City's industry!)

<http://phpcity.myminicity.com/tra> (Let's get the Transportation sector up to scratch!)

<http://phpcity.myminicity.com/sec> (Whip out a BFG and fight crime!)

<http://phpcity.myminicity.com/env> (Save the environment and tell Bush where to stick it!)

...more links to click with fervour when the City's population increases more.

As a really simple click-daily web game, the core concept of MyMiniCity.com is quite good. It's a simple ranking system relying on the obsessive visiting of URLs daily to increase the score of your adopted City. As I've seen before, it's the simple things leading to a little pride, achievement and sense of superiority that are weirdly attractive to people. No doubt this simple flash game gets a surprising number of visitors who, attracted first by the simple concept of growing a Mini City, are then diverted to the other stable of games advertised on the site.

But, hey, we're talking Jamaica - don't disappoint Mayor McCal. If you do, some polite gentlemen may drop by, strap you to a chair, and force you to memorise all those PHP functions everyone relies on the manual to recall. Do you want that? I wouldn't.


Posted by Pádraic Brady in PHP General, PHP Security at 15:30

Wednesday, December 19, 2007


Complex Views with the Zend Framework - The Final Chapter: ZFE and Zend_Layout released to Core!

As [announced earlier by Matthew Weier O'Phinney](#) the Zend_View Enhanced and Zend_Layout components have migrated to the Zend Framework Core.


These two (now much more polished!) components were both designed to solve the concerns a lot of developers were having in achieving truly complex, structured and modular Views using Zend_View. Indeed there are months of blog postings, debates, experimental and not-so-experimental code, proposals, IRC sessions and countless emails pouring over how to accomplish the goals of these components. The end result is something I feel will serve Zend Framework users faithfully for months and years to come.

Those following the long running "Complex Views With The Zend Framework" series on this blog will be overjoyed to see the adoption of partials, placeholders and controller actions that I discussed at the time. All of which were explored in general terms before being combined into the Zend_View Enhanced proposal. Ralph Schindler's Zend_Layout solution (the one I had many a disagreement over 

) has also made Core which resolves any layout aggregation issues that are hugely troublesome otherwise.

I'd like to thank Matthew and Ralph who did most of the coding and committing for this one. I was unfortunately tied up to a tree called "Life" after the Summer and unable to contribute in any remarkable detail except by providing the odd remark and cutting criticism when the mood took me 

. I can be a harsh complainer at times! Great job from both you guys!

While I sit here staring at some C code wondering if windows.h is the devil, I hope users of the Zend Framework will quickly adopt and try out these Cored components. They reduce the solutions to very common and concrete problems with composing a View to a simple API that's easy to learn, and easy to apply. If you are still using older versions of either component, there's no time like January 7 (let's let you have the Christmas and New Year ) to migrate.

Posted by Pádraic Brady in PHP General, PHP Security at 17:23

Compiling PHP for Windows Vista using Visual C++ Express 2008 - Seriously!

Over the past week or so I've been figuring out how to compile PHP on Windows, specifically Windows Vista. It's been an interesting ride since I'm not very familiar with compiling on Windows to start with. However with some perseverance, and the help of all **two** useful online sources of information for the task of compiling PHP on Windows, I got around to accomplishing it.

The main source of information available is an excellent guide written by [Elizabeth Marie Smith](#) in December 2006 about [compiling PHP5 and PHP-GTK2 using Visual C++ Express](#). The other basically refers back to Elizabeth's article and adds some details on needing ICU to compile PHP6. If you are currently using Windows XP and Visual C++ Express 2005, then you need look no further. This is the single most informative source about compiling PHP on Windows that I could locate.

If you are using Windows Vista, or intend updating/using Visual C++ Express 2008, then my blog entry is largely another coat of sugar on top of Elizabeth's guide to clear up any difference between the two approaches. There certainly are differences, but they are not overwhelming - esp. since I'll lay them out below



. This set of instructions assumes you are building for Vista (32bits). If you want to compile for 64bits for Vista, see [http://msdn2.microsoft.com/en-us/library/x4d2c09s\(VS.90\).aspx](http://msdn2.microsoft.com/en-us/library/x4d2c09s(VS.90).aspx) which may help.

You need to download, and install:

1. [Virtual Clone Drive 5](#)
2. [7-Zip](#) or [WinRAR](#)
3. [Microsoft Visual Studio 2008 Express](#)
4. [Microsoft Windows SDK for Vista Update](#)
5. [Microsoft .NET Framework SDK 2.0](#)

Installing The Windows Vista Compilation Environment

The reason for downloading Virtual Clone Drive is to allow us to mount ISO disk images. The downloads for VCE2008 and the Windows SDK for Vista are especially large, and I just found it handy to download the ISO versions so I could install multiple times without going through the web install options multiple times, or for multiple machines. WinRAR or 7-Zip should be available to decompress standard gzip/bzip tarballs where a zip download option does not exist. Download and install these at your leisure.

So, the first major download is [Microsoft Visual Studio 2008 Express](#). This was released in the last month (as of writing) so I chose to use it. You can use VCE2005 on Vista but VCE2008 skips a lot of the setup tasks VCE2005 required to integrate with the Windows SDK.

At the bottom of the download page you can find the all-in-one ISO download. It packs quite a punch but includes the C++, C#, VB and Web Developer editions of Visual Studio Express including SQL Server, .NET Framework 3.5 (required!) and Silverlight. Actually MS seem to be pushing Silverlight (sort of a Flash alternative) since the page has Silverlight elements that prompt you to download and install the runtime.

If you took the web install option, just follow the setup instructions and accept the default installation path.


You don't require Silverlight or SQL Server to compile PHP so feel free to deselect those options. If you downloaded the ISO you need to mount the ISO disk image using Virtual Clone Drive. To do so open My Computer, right-click the virtual drive, select "mount...", find the ISO image to mount, and accept. It should now autorun and execute the setup program - see the instructions at the beginning of this paragraph. Note installation may take some time. Start up VCE at least once (sometimes wants to setup the environment) and make sure the register it (you'll need an MS Passport or Email account to do this, e.g. Hotmail).

A word of interest. VCE2008 will also install v6.0A of the Windows SDK. I tried compiling PHP using this SDK version but it didn't seem to work out well. If you have any success doing so let me know via a comment. Otherwise let's press on and install v6.0 of the Windows SDK which VCE2008 seems to prefer using anyway.

The second major download is [Microsoft Windows SDK for Vista Update](#). Please be aware this is updated from time to time and the current update is dated March 2007. New versions are generally commented on in the attached Additional Information of the download page. Again, this the ISO download which is pretty heavy. There's an alternative web install option if you have a speedy reliable connection. Download and mount the ISO using Virtual Clone Drive as described earlier. During setup accept the default options and proceed. As with VCE2008, installation may take some time to complete.

The third download is the [Microsoft .NET Framework SDK 2.0](#). Download and install using the default options. I'm not completely certain where PHP needs the .NET SDK - I can compile PHP with some standard options without it installed. Giving the benefit of the doubt, it's probably needed somewhere for something for someone so there's no harm installing it and avoiding any future obscure problems. Unlike the Windows SDK which VCE2008 will automatically detect and include, you need to integrate the .NET SDK into VCE2008. This is accomplished by editing VCE's vcvars32.bat file - try looking for it in:
C:\Program Files\Microsoft Visual Studio 9.0\VC\bin\vcvars32.bat

To edit, you first need to copy it to your desktop (you'll find other Vista editing VCE files is a task in futility even if you have an administrator account). Open in a text editor like notepad, and locate the lines setting the environmental variables PATH, INCLUDE and LIB. At the end of each, you should append the relevant path to the .NET Framework SDK BIN, INCLUDE and LIB directories just before the closing variable. Make sure you have semi-colons (;) between each added path. There's nothing else needed - vcvars32.bat is executed automatically whenever the VCE2008 Command Line tool is started, and it will detect other needed paths for the Windows SDK installed earlier.

If you're still here, give yourself a pat on the back. The torture is almost over. The very last step is to fix some problem PHP has finding a file called winres.h which doesn't exist. What you need to do is locate the file called WinResrc.h in C:\Program Files\Microsoft SDKs\Windows\v6.0\Include, copy it to the desktop, rename it to winres.h, and copy it back into the Windows SDK v6.0 include directory. You won't be able to copy&rename it on the spot, because Vista really doesn't like that 


Downloading PHP and Libraries To Allow Compilation

You have three main choices about what to compile. PHP 5.2.X where the source code for the latest stable version is available. PHP 5.2/5.3-dev which has regular snapshots of the source code for this development branch published to <http://snaps.php.net/>. Or PHP 6 development which also has published snapshots. I'm using PHP 5.3-dev.

Note: Under Windows Vista, possibly given the updated from XP's Windows Platform SDK to Vista's

Windows SDK (no "Platform), there is a bug in PHP 5.2.X which causes a compilation error in source code related to ipv6 (presumably) in ./main/network.h of the source code. You can comment out the offending line reported in the compile error to force a successful compile of PHP 5.2 but it will probably break ipv6 which may not be a good thing. The error is not evident when compiling PHP 5.3-dev so it will likely not be a problem by the time PHP 5.3.0 is released.

Whatever your choice, download the source code.

Create a root directory on C:\ called "PHPDEV". Copy the PHP source code archive into this directory and extract it using WinRAR or 7-Zip. Since I despise command line typing I usually rename it to "php-x.x" so it's easier to navigate there without remembering the timestamp 

Next you need to download the libraries and header files PHP and it's extensions may need. These are not included in the source code package but available from individual websites and download locations. Luckily, [Edin Kadriba](#) keeps an archive of these in a single easy to update download which includes the standard ones and those needed for PECL extensions. So download <http://files.edin.dk/php/win32/zip.zip>, copy it to C:/PHPDEV, and extract it there.

If you intend compiling PHP6, you will also require the ICU (International Components for Unicode) libraries available from <http://www.icu-project.org/download/>. The current release is 3.8.x, and if you check [the ICU4C download page](#) the downloads include binary distributions for Win32. The mscv8 prefix refers to these been built for the Microsoft MSCVR80 C runtime - which VCE2008 tends to use. Older downloads of ICU4C for version 3.6.x were built for MSCVR71 if required. Download and extract into C:/PHPDEV.

And we're done. If you were bothered by the number of directories don't worry - when compiling PHP, the required files will be looked for in sibling directories of the PHP source code automatically.

Compiling PHP on Windows Vista using Visual C++ Express 2008

Now we get to the good part, where we find out if this all worked or failed miserably. If it does fail don't get too dejected - I've given up on this working first time out. I find I have to restart my PC a few times before VCE gets the messages and starts using the right environmental variables. This might just be my PC though - registering VCE2008 failed a few times before I managed to get through.

Note: PHP compiled under this set of instructions will require the MSCVR80 Microsoft C runtime (also fondly known as mscvr80.dll) which is generally available on Windows Vista. Windows XP or earlier probably uses mscvr71.dll which generally means you should install MSCVR80 on any XP system (if not already present) if you intend deploying your newly compiled PHP there. Get it from

So! From the Start Menu, find the Visual Studio Express 2008 folder and click it. From Tools, select the VCE2008 Command Line tool. It should open a standard MS-DOS console window - the main thing is that it should also have executed vcvars32.bat automatically so this console is ready to start some heavy duty compiling.

Navigate to your PHP source code directory at C:\PHPDEV\php-5.3 (or whatever yours may be called). Compiling PHP from hereon out should be fairly typical - just bear in mind some options may require you to download additional libraries. For example, we already know PHP6 requires the additional ICU4C libraries. Many others will use the files we downloaded from Edin's site.

From the command line run:

```
buildconf.bat
```

This will generate a configure script containing all the possible options to configure PHP with. This includes the typical configure options, and others based on the available extensions.

You can run the following the list all the available options:


```
cscript /nologo configure.js --help
```

We'll test everything using a simple configure line. I'm re-reading Sara Goleman's book so I'll borrow the starting configure line from there since it seems suitable (as well as minimal!). This is a good first step before jumping into adding new configure options. It will only create a command line executable, adding configure options for Apache/MySQL or other exotics is left to your own devices.

```
cscript /nologo configure.js --without-xml --without-wddx --without-simplexml --without-dom --without-libxml  
--disable-zlib --without-sqlite --disable-odbc --disable-cgi --enable-cli --enable-debug --without-iconv
```

Once you've run the above, there's only one more step. Take a deep breath, relax and run:

```
nmake
```


This usually takes a while, and generates a ludicrous number of warnings. Warnings under Windows are par for the course so they're nothing to get worried about. Takes about 5-10 minutes on my PC, and slower on older machines. Once it's done (hopefully without experiencing any compile errors; a bit more fatal than mere warnings 

) you should find it's created a new directory in your PHP source code location called Debug_TS. Inside there's a sparkly new php.exe file waiting for you. Unlike a traditional Windows PHP binary, don't expect to see many .dll files for extensions - .dll extensions are only generated if an extension has been compiled "shared", otherwise it's just lumped into PHP. You may also notice the absence of a php.ini file (or it's dist/recommended templates). This is normal. I'll explain

Change to the Debug_TS directory where the PHP binary is located, and run the command:


```
php -v
```

You should now have executed PHP using the command line, and received a few lines of output giving the

PHP version, and it's compile time. From here you can experiment with configure options to get exactly what you need and delve ever deeper into the dark world of compiling PHP on Windows... 

Last bit - to install PHP somewhere after installation, set the installation path using the `--enable-prefix=C:\path\to\php` configure option before compiling. Once compiled, call:

```
nmake install
```


That's all I have for now. If anything more needs explaining I'll edit this blog entry for it. In the meantime I hope this small article helps someone out who like me is crazy enough to need to compile PHP on Windows for some reason 

. Good luck!

Posted by Pádraic Brady in Irishisms, PHP General, PHP Security at 15:22

Thursday, December 13, 2007

Namespaces (or Yet Another Pointless Opinion Piece)

Seems like everyone else is displaying opinions on Namespaces in PHP. Yet another milestone in PHP's quest to match every other language's concept of OOP 

. But seriously, like every other proposal that looks off-the-wall from the point of view of a developer, you have to keep in mind PHP is merely getting around to implementing a standard solution from other languages. Java has namespaces, C++ has namespaces, Ruby has modules, Perl has... You get the picture. PHP doesn't have to follow the herd, but sometimes the herd-brain does have its good points.

The problem isn't that namespaces are bad, it's that these arcane structures seem to have so little impact on development in PHP today for many many people. Does that mean it's unimportant? Appearances are deceiving - when intelligent people get something like namespaces into CVS it's a signal that the community has decided it has value. Eventually everyone will see it has value, and then everyone will use it, and then all the protests will get relegated to some back country forum in the hinterlands of the internet.

I remember a few years ago the feedback on the idea of type hinting in function/method declarations. The chorus of "it will ruin dynamic typing" style protests resounded across the PHP landscape and continues to echo through forums even today. Honestly, if you can't bother to apply some level of reasoning to a new solution then give the community a break and cease prattling like a manic prophet of the oncoming apocalypse standing on a street corner. Of course today every Joe and his dog is writing type-hinted code for PHP5, and dynamic typing appears to have survived the assault of this Javatastic invasion.

It's tiresome enough that messages on the PHP Internals mailing list (I lurk - never saw a need to post until the day I might submit dodgy patches to irritate everyone...hehehe) [deteriorate into tit-for-tat off-topic comments that spark a deluge of more emails I have to skip past](#) or deductions worked out laboriously in the forebrains of strange people who don't know what "C" is. At least sufficiently few people read the list not to get too misinformed, discouraged or plain put off by Joe Blogg's latest diatribe on folk discussing proposals on IRC/IM when they're asleep and not posting stuff publicly (or whatever). Get serious - emailing everything in detail is a horrible idea. It would add to the internals flood anyway! My inbox is bad enough!

My opinion, is that namespaces are a neat solution to an ever increasing problem - those freaking PEAR CS class names. Greg Beaver's earlier input added the nails, and conjured up a nailgun, for the opposition's coffin. There are other potential benefits of course like possible namespace level access modifiers, but screw those - I want to type less, and not have to read

`My_Cool_Freakin_Ever_Increasing_Length_Class_Name_Because_I_Love_Typing_Way_Too_Mucho` all over my otherwise clean and awesome code. Roll on, Mucho::Gratias, and give my wrist muscles a wee break.

Posted by PÃdraic Brady in PHP General, PHP Security at 13:33

Friday, December 7, 2007

RubyConf 2007 Videos Online

For those who find some level of enjoyment out of watching Ruby talks, most of the RubyConf talks from the past week are available online at:


<http://rubyconf2007.confreaks.com/>

Marcel Molina's look at "What Makes Code Beautiful?" is always worth watching, if only to contrast with PHP for a little deeper insight. Both IronRuby and JRuby have their own talks - as does Rubinius.


Of particular interest since I'm developing the PHP port, David Chelimsky and Dave Astels presentation of "Behaviour Driven Development with RSpec" is also online. It's Ruby specific of source, but the core ideals represented by BDD are worth being informed about.

Posted by Pádraic Brady at 17:51

Astrum Futura Redux

For the cool folk who have followed the Solar Empire legacy through thick and thin since 1999, those cute .php3 file suffixes, Moriarty's continuing denegration of all other developers, the revolution that open source brought, the sudden push towards OOP, and the...err...security and bug population... Working on anything hitting it's 9th anniversary in PHP is really interesting - 9 times the fun and games 

Astrum Futura was envisaged as a desperately needed update to the long running open source Solar Empire franchise, where players engage each other across a galaxy of 500+ star systems mining resources, building colonies, and generally screaming bloody murder at each other. It's not the most impressive or ambitious of online games, but it's always been tenacious and attracted roaming users who like a quick dash of destruction in their daily diet.

In January 2008, the oft delayed development process will once again creak into action. The current tracer code was built originally using the Zend Framework 0.2-0.6 and famously led that Spring to my long running "Complex Views With The Zend Framework" blog series that gave birth to the Zend_View Enhanced proposal, and maybe gave Ralph Schindler a few headaches 

. Building anything more complex than a login page was pure heart ache previously.


Now that time is available, the Zend Framework significantly more mature, and we have a lot of legwork carried over from 2006 in the Quantum Game Library (thanks to Jacob Santos), it's about time we got something concrete done. The usual suspects, if interest levels are high enough, can report to the shiny updated phpBB3 forums at <http://forums.astrumfutura.com> - same domain as this blog you're reading.

And we will be applying XP this time - the random running process that usually prevails just doesn't work out well. So get your Selenium gear in place.

Posted by Pádraic Brady in Astrum Futura, PHP Game Development, PHP General, PHP Security at 17:01

Monday, December 3, 2007

PHPMutagen: Mutation Testing for PHP5

I swear - this is the last bdd/testing related idea I'm having for a long time 

. There's (so far) PHPSpec, PHPMock and now PHPMutagen. The first is a Behaviour-Driven Development framework, the other two are unrelated libraries intended for use with PHPSpec, SimpleTest, PHPUnit or any other conceivable testing framework.

As I said in my previous entry I was thinking about how to write a Mutation Testing engine. The "braindump" was to use PHP's built in Tokenizer to break down a class file into digestable pieces which could be mutated, and then reconstructed into a mutated file. Once you allow for a working copy of the original source (let's not mutate the original!) it turned out to be a reasonable approach.


The source code is still rough, but essentially operational. I wouldn't get too excited until I find some time to scan through it again (after I get over this cold) and get the spec coverage into shape. For the moment I'll tease you all with some sample output.

Here's what you really want to see after running PHPMutagen (note this was only possible with PHPSpec, but a PHPUnit or PHPT adapter would be equally simple to add later):

```
.....
7 Mutants born out of the mutagenic slime!

7 Mutants exterminated!

No Mutants survived! Muahahahaha!
```

Here's what you probably want not to see 

. The comment at the end is important - in Mutation Testing there will always be a few elusive ghosts which aren't a real problem because you can switch operators in code without altering behaviour in some circumstances. Using `- $this->_actualCallCount++;`
`+ $this->_actualCallCount--;`
`if (!is_null($this->_exceptionToThrow)) {`
`if (is_array($this->_exceptionToThrow)) {`
`$class = $this->_exceptionToThrow[0];`
`$message = $this->_exceptionToThrow[1];`

Happy Hunting! Remember that some Mutants may just be Ghosts (or if you want to be boring, false positives).

The diff output for a mutant is courtesy of PEAR::Text_Diff's unified diff renderer and a little extra decorating.

Posted by Pádraic Brady in PHP General, PHP Security at 13:38