PHP6 – The Next Level

Santa Clara, April 26\textsuperscript{th} 2007
MySQL User Conference
Lukas Kahwe Smith (lsmith@optaros.com)
http://pooteewet.org/files/mysqliuserconf07/php_the_next_level.pdf
About Myself

- Lukas Kahwe Smith
- Started using PHP in the late PHP3 days
- Joined PEAR sometime in 2002
- Began posting on internals@lists.php.net in 2003
- Started the unofficial official PHP todo list in 2005
- Makes his living as a consultant for Optaros
  - OSS consulting and system integration firm
  - We are always hiring
- Can be reached
  - Via Email: lsmith@optaros.com
  - On IRC: “lsmith” on efnet, freenode, ircnet ..
  - In Person: Here or at Ultimate Frisbee tournaments
Agenda

- **BC breaking changes in HEAD**
- Additions available in HEAD
- Other planned changes on the horizon
- What's not going to happen
- Current state of PHP6
- References
BC breaking changes in HEAD

Register Globals

- Automatically define variables from user parameters
- Removed since it promote writing of insecure code
- Obviously all functions using register globals were removed such as session_register() etc.
- BC hack
  - Use import_request_variables()
- Code change
  - Use $_POST, $_GET, $_COOKIE, $_REQUEST instead
BC breaking changes in HEAD

◆ Register long variable
  • Used to be controlled via the php.ini setting “register_long_arrays”
  • $HTTP_*_VARS => $_*
    - $HTTP_POST_VARS => $_POST
    - $HTTP_SERVER_VARS => $_SERVER
  • BC Hack
    - $HTTP_POST_VARS &= $_POST; etc.
  • Code change
    - Search/replace in your code via a regular expression
BC breaking changes in HEAD

All “magic”^^^evil automatic string escaping

- Used to be controlled via php.ini setting “magic_quotes_gpc”, “magic_quotes_sybase” and “magic_quotes_runtime”
- Calling set|get_magic_quotes_gpc() raises E_CORE_ERROR
  - Gave developers a false sense of security
    - Escaping needs to be context sensitive
    - Made writing portable applications a hassle
- BC hack
  - Iterate over superglobals and call addslashes()
- Code change
  - Use the relevant context sensitive alternative functions when switching context
BC breaking changes in HEAD

允许传递引用参数会抛出E_STRICT错误
- 通过php.ini设置“allow_call_time_pass_reference”控制
- 自PHP 5起已弃用
- BC补丁
  - 禁用E_STRICT

- 代码更改
  - 将&符号移到函数/方法签名中
    - foo(&$bar) => function foo(&bar) { .. }
BC breaking changes in HEAD

- **dl() disabled by default**
  - Disabled for multi-threaded SAPI's already
    - It was never thread safe
  - Enabled on a per SAPI basis
    - For now only in CLI and Embed in the bundled SAPI's
    - Disabled for Apache etc because of unloading issues
      - Cry loud enough and someone might fix these
  - BC hack
    - None available
  - Code change
    - Add the extension into your php.ini
BC breaking changes in HEAD

- Removed triggering of __autoload() in various places
  - instanceof, catch blocks, is_a() and is_subclass_of()
    - Triggering __autoload() for these was a bad idea
    - Not an intended feature
  - BC hack
    - None available
  - Code change
    - If your code relied on this, then you better rethink what you were trying to do!
BC breaking changes in HEAD

- Dynamic break operand support removed
  - Does not really work
  - There should be no dynamic jumping in PHP
    - for (..) { for (..) { $foo = rand(1, 2); break $foo; } } 
  - BC hack
    - None available
  - Code change
    - for (..) { for (..) { if (..) break 2; else break 1; } } 

BC breaking changes in HEAD

◆ Assign "new" by reference will throw an E_STRICT error
◆ zend.ze1_compatibility mode removed
◆ safe_mode removed
  • Only gave a false sense of security
  • Easily by-passed with various extensions
  • Made the life of developers miserable
    - For example problems accessing files cached in the FS
  • Many hosters blindly enabled it without thinking
◆ Support for Freetype1 and GD1 removed
  • Use GD2 instead
◆ ncurses extension moved to PECL
◆ Output buffering changes
  • Internal rewrite, so beware of hidden BC breaks
  • Expect some BC breaks in ob_start() and ob_get_status()
Agenda

- BC breaking changes in HEAD
- **Additions available in HEAD**
- Other planned changes on the horizon
- What's not going to happen
- Current state of PHP6
- References
Available changes in HEAD

◆ Jump label aka “limited goto”
  - May be helpful for writing parsers and handling errors
    - Static labels only
    - Limited to current scope
    - May not jump into loops or switch statements
      - But may jump out of them
  - Example code:
    - $n = 1;
    - L1:
      - echo "${n: ok}
      - $n++;
    - if ($n <= 2) goto L1;
  - Example output:
    - 1: ok
    - 2: ok
Available changes in HEAD

- Native Unicode support
  - Currently PHP is essentially a binary preprocessor
  - Adds native unicode string type (UTF-16 encoding)
  - Based on the ICU library
    - Text processing, collation, encoding conversions, portable locale support
  - Enabled as needed (unicode.semantics php.ini setting)
    - Makes internal functions unicode aware (strlen() for example)
  - Script level encoding
    - script_encoding php.ini or declare() at run time setting
    - Unicode characters in identifiers
  - Flexible input (lazy loading) and output encoding as well as conversion error handling mechanisms
  - set|get_default_locale() will deprecate POSIX style setlocale()

- ext/mbstring will stay for now
Available changes in HEAD

- ifsetor() like operator
  - $a ?: $b == $a ? $a : $b
  - Not the real thing since $a needs to be defined
- __construct() may be part of an interface
- E_STRICT added to E_ALL
  - By then we might also have E_DEPRECATED split off from E_STRICT (currently planned for 5.3)
- Added runtime JIT auto-globals fetching and caching
- Added ext/hash support to the ext/session ID generator
- Mark ext/ctype as deprecated in favor of ext/filter
Agenda

- BC breaking changes in HEAD
- Additions available in HEAD
- Other planned changes on the horizon
- What's not going to happen
- Current state of PHP6
- References
Other planned changes on the horizon

- XML related improvements
  - ext/soap enabled by default
    - PECL axis2 package is maturing
      - But is too large and unwieldy for wide use
  - Add support for XML security
    - Add hooks into ext/soap to make use of XML security
Other planned changes on the horizon

- Late static binding
  - "Fixing" inheritance for static methods
  - Enables optional resolution of inheritance at run time instead of at compile time
  - Example code
    - class A {
      - static function staticA() { static::static2(); }
      - static function static2() { echo "A::static2\n"; }
    }
    - class B extends A {
      - static function static2() { echo "B::static2\n"; }
    }
    - B::staticA();
  - Example output:
    - B::static2

- Enables optional resolution of inheritance at run time instead of at compile time
- Example code
  - class A {
    - static function staticA() { static::static2(); }
    - static function static2() { echo "A::static2\n"; }
  }
  - class B extends A {
    - static function static2() { echo "B::static2\n"; }
  }
  - B::staticA();
- Example output:
  - B::static2
Other planned changes on the horizon

◆ Type hinting
  • Add support for type hinted return values
    – No final syntax chosen at this point
  • Add support for optional type hinted parameters (DONE)
  • Add support for type hinting of primitive types

◆ Property overloading with __get__()/__set()
  • To document “virtual” properties
  • To be able to check the existence of a “virtual” property
  • To be able to generate a clear error message when the “virtual” property does not exist
    – Current userland solution is to define a property containing an array of virtual properties
    – Alternatively add
      – Callback method could be created
      – Keyword „abstract“ to mark a property as “virtual“
Other planned changes on the horizon

**Namespaces**

- Backslash seems to be the most likely separator
  - Only a single character alternative not yet taken
  - Other candidates was “:::
- Might be limited to OO constructs
- Still issues with how import works

**Alternatively follow the PHP userland naming guide**
Other planned changes on the horizon

◆ mysqlnd
  • Drop in libmysql replacement (more or less)
  • Currently only supported in ext/mysqli
    – Support for PDO will be added eventually
  • BSD licensed, currently in beta
  • Performance is as good or slightly better than libmysql
    – Especially for large result sets (less memory too)
    – Zval cache still needs tweaking (feedback wanted)
  • Tightly coupled with PHP6 (and PHP5)
    – Streams, memory management, client side query cache, statistics, unicode, persistent connections*

◆ MySQL_Proxy*
  • Load balancing, persistent connections, query analysis, filtering and auditing, XA transaction manager
  • Independent of PHP (but interesting none the less)
Other planned changes on the horizon

◆ Other potential BC breaks
  • Move ext/mime_magic to PECL in favor of ext/fileinfo
  • Remove ASP tag support and disallow mixing of PHP tags
  • Dynamic call on a static method throws E_FATAL (and vice versa)
  • Synchronize with relevant patches from hardend-php
  • microtime() will return a float by default
  • ext/ereg will be moved to PECL
    – Disabling PCRE will no longer be allowed
  • BC breaks to casting of objects to primitive types
Other planned changes on the horizon

◆ Additions and other improvements
  • Add native 64 bit integer support
  • Ready only variable and persistent variable support
  • Bundle the APC byte code cache (disabled by default)
  • Speed up @-operator and heredoc (maybe already in 5.3)
  • Add sand boxing support
    – Maybe beefed up runkit with (semi-)persistent instances
    – Calling legacy code
    – Calling a trimmed down PHP (templating)
What's not going to happen?

- Function API alignments
- Mass removal of deprecated functions or non standard aliases
- Removal of ext/mysql
- Named parameters
- Type hinted properties
- Fatal errors to become exceptions or support for “finally”
- Improvements to garbage collection for long running scripts
- Addition of a debugger extension
- Support for `<?php='foo' ?>`
- Multiple Inheritance
- Removal of `{}` in favor of `[]` character referencing in strings
- Ability to flag E_STRICT compliance on a class or file basis
Current state of PHP6

- Very close to a first preview/alpha release
  - Finishing up JIT decoding/error handling
  - Maybe also
    - Namespaces
    - Late static binding

- Still a lot of work on unicode
  - Almost 61% of all functions have been updated and tested for unicode support
    - Want to get your feet wet in the PHP code?

- Final release
  - Probably in the first half of 2008, maybe late 2007
  - If you need unicode you might be desperate to use a beta
References

- Installing PHP6
  - http://snaps.php.net/

- NEWS file in HEAD

- Unofficial PHP todo wiki
  - http://oss.backendmedia.com/PhP60

- Paris meeting notes
  - http://www.php.net/~derick/meeting-notes.html

- Zend weeklies
References

- **PEAR::PHP_Compat**

- **Unicode**

- **Namespaces discussion summary**

- **mysqlnd**
  - [http://blog.ulf-wendel.de/?p=138](http://blog.ulf-wendel.de/?p=138)

- **mysql_proxy**
  - [http://jan.kneschke.de/projects/mysql/mysql-proxy](http://jan.kneschke.de/projects/mysql/mysql-proxy)