

PHP 5: So now what?

New Toys!



Larry "Crell" Garfield
Senior Programmer,
Palantir.net

PHP 5 Zend Certified
Engineer



Special guest heckler

Karoly "chx" Negyesi,
Now Public

PHP 5.2

- Released 2 November 2006
- Only 5 minor releases
- Fast uptake

XML

- SimpleXML
 - XML parsing as it was meant to be
- DOM
 - Standard, language-agnostic traversal
- XSLT
 - XML Transformations
- *Thursday, 16:00...*

PDO

- PHP 5.1+
- Unified Database Access API
- Prepared Statements
- Iterators
- *Friday, 10:00...*

New functions!

- Lots of array functions
 - `$assoc = array_combine($keys, $values);`
- `$dirs = scandir('.');`
- `http_build_query($args);`
- `file_put_contents();`
- Many more...
 - <http://php.net/manual/en/migration5.functions.html>

JSON

- Replaces drupal_to_js()

```
echo json_encode($data);
```

```
$data = json_decode($json);
```

E_STRICT

- New error reporting level, 2048
 - "...run-time PHP suggestions on your code interoperability and forward compatibility."
- Warns on deprecated usage (for now)
- Let's do...

Exceptions

- Out-of-band error handling
- Used by some internal components
- try-catch-throw pattern
 - See also: C++, Java

Streams

- Generalized "file" wrappers
- Filters
 - compress.zlib
 - https
 - ssh2.tunnel
- <http://php.net/manual/en/ref.stream.php>

Streams

```
$session = ssh2_connect('example.com', 22);  
ssh2_auth_pubkey_file($session, 'username',  
    '/home/username/.ssh/id_rsa.pub', '/home/username/.ssh/id_rsa',  
    'secret');  
  
$stream = fopen("ssh2.tunnel://$session/remote.example.com:1234",  
    'r');  
  
readfile("php://filter/read=string.toupper|  
    string.rot13/resource=http://www.example.com");
```

Streams

```
function parse_csv($str, $delimiter = ',', $enclosure = '"', $len = 4096) {  
    $fh = fopen('php://memory', 'rw');  
    fwrite($fh, $str);  
    rewind($fh);  
    $result = fgetcsv( $fh, $len, $delimiter, $enclosure );  
    fclose($fh);  
    return $result;  
}  
  
$tags = 'a tag, another tag, "some, tag";'  
$tags = array_map('trim', parse_csv($tags));
```

ext/filter

- Rasmus' latest idea
- Type-specific filtering
- Replaces `$_GET`, `$_POST`, etc. access
 - `INPUT_GET`, `INPUT_SERVER`, `INPUT_SESSION`
 - `input_get(INPUT_POST, 'foo', $filter);`
- Not just for input data!
 - `filter_data($data, $filter);`

ext/filter

```
$search_html = input_get(INPUT_GET, 'search',  
    FILTER_SANITIZE_SPECIAL_CHARS);
```

```
// returns (int)5 (CHECK THIS!)
```

```
filter_data("5", FILTER_VALIDATE_INT);
```

```
// returns "bob@example.com"
```

```
filter_data('bob@example.com', FILTER_VALIDATE_EMAIL);
```

```
// returns FALSE
```

```
filter_data('example.com', FILTER_VALIDATE_URL,  
    FILTER_FLAG_SCHEME_REQUIRED);
```

Date and time

- Native time zone support
- DateTime class
 - 64-bit; Handles more than Epoch dates
- DateTimeZone class
 - Master list of timezones
- API docs sadly slim
 - <http://maetl.coretxt.net.nz/datetime-in-php>

DateTime

```
$date = new DateTime('1885-11-5');  
echo $date->format(DateTime::W3C);  
// 1885-11-05T00:00:00-06:00  
echo $date->format(DateTime::RSS);  
// Thu, 05 Nov 1885 00:00:00 -0600  
echo $date->format("Y-m-d h:i:s");  
// 1885-11-05 12:00:00  
$date->modify('next Tuesday');  
echo $date->format("Y-m-d h:i:s");  
// 1885-11-10 12:00:00
```

DateTime

```
$date = new DateTime();  
$date->setDate(2007, 9, 31);  
echo $date->format("Y-m-d h:i:s");  
// 2007-10-01 06:38:27
```

```
$date->setISODate(2007, 5);  
echo $date->format("Y-m-d h:i:s");  
// 2007-01-29 06:38:27
```

DateTimeZone

```
print_r(DateTimeZone::listAbbreviations());  
$zone = new DateTimeZone('Europe/Madrid');  
$date = new DateTime(NULL, $zone);  
echo $date->format(DateTime::ISO8601);  
// 2007-09-17T01:38:27+0200  
$date->setTimezone(new  
    DateTimeZone('America/Chicago'));  
echo $date->format(DateTime::ISO8601);  
// 2007-09-16T18:38:27-0500
```

Date and Time applied

```
$user->timezone = 'America/Chicago';
```

```
$server->timezone = 'Europe/Madrid';
```

```
$input = getdate();
```

```
$date = new DateTime(NULL, new DateTimeZone($user->timezone));
```

```
$date->setDate($input['year'], $input['mon'], $input['mday']);
```

```
$date->setTimezone(new DateTimeZone($server->timezone));
```

```
echo $date->format(DateTime::ISO8601);
```

```
$date->setTimezone(new DateTimeZone('GMT'));
```

```
echo $date->format(DateTime::ISO8601);
```

```
echo $date->format('U'); // UNIX timestamp
```

SOAP

- Super-XML-RPC
- RMI
- WSDL files
- More robust than XML-RPC

Objects

- Super-resources
- Abstract classes, Interfaces
- Magic methods
- Auto-loading

SPL

- Language assistance
 - `ArrayAccess`
 - `Countable`
- Iterators
 - `FilterIterator`
 - `RecursiveIterator`
- Utility classes
 - `DirectoryIterator`

Good OOP practices

- Do not fear the object!
- Do not worship the object!
- Not namespaces
- Shallow hierarchy, Visitor pattern
- Interfaces
- Protected, not Private

Good OOP practices

- Where not to use objects
 - Modules
 - Menu callbacks
- Where objects may be useful
 - Data Obj: Node, Comment, User, Field, File
 - Logic Obj: Forms, Widget, Action, View
 - Renderable: Page, Block, View

The OO challenge

- Procedural is easy to hack when needed
- A good OO framework is far more flexible and powerful
- A bad OO framework is unrecoverable

- Plan well

Intermission

Run, before Larry
starts talking again

Coding standards

- PHP internal convention:
 - `function_name()`
 - `ClassName::methodName();`
- Drupal convention:
 - `function_name()`
 - ?
- Adopt PHP standard to avoid confusion
 - (Keep `_` for dynamic child classes?)

Discussion

So now that we have PHP 5.2,
what do we do with it?