PharScape

Introducing PHAR

Marcus Börger

php|a Chicago 2007

Introducing Phar

- ☑ Archive format like JAR, TAR, ZIP
- ☑ Extension is not really required
- ☑ Normal PHP scripts that can be executed with PHP
- ☑ Command line tool which is itself a PHAR

Phar

- ☑ Entries are accessible as streams, phar://
- ☑ PHAR stub can contain PEAR package PHP_Archive
- ☑ Archive can be given an alias name
- ☑ Entries can be referenced by archive alias
- ☑ Entries can be compressed
- ☑ Extracted archives can still be accessed as phar://
- ☑ No need to change extracted files in any way
- ☑ Global and entry wide metadata supported

How to get PHAR

- ☑ Using PEAR

```
$> pear install pecl/phar
```

How to get PHAR

- ☑ Using PEAR

```
$> pear install pecl/phar
```

- ☑ Building your own from CVS

```
$> cvs –d :pserver:cvsread@cvs.php.net:/repository
      login
      phpfi
$> cvs –d :pserver:cvsread@cvs.php.net:/repository
      co –d phar pecl/phar
$> cd phar
$> phpize && ./configure && make
$> sudo make install
$> vi /etc/php.ini

$> make phar.phar
$> sudo cp phar.phar /usr/bin
```

How to get PHAR

- ☑ Using PEAR

```
$> pear install pecl/phar
```

- ☑ Building your own from CVS

```
$> wget http://pecl.php.net/get/phar-1.2.0.tgz
$> tar –xzvf phar-1.2.0.tgz
$> cd phar
$> phpize && ./configure && make
$> sudo make install
$> vi /etc/php.ini

$> make phar.phar
$> sudo cp phar.phar /usr/bin
```

Phar

- ☑ Packing files in a directory, using copy()

```
<?php
$phar = new Phar($argv[1], 0, 'newphar');
$dir  = new RecursiveDirectoryIterator($argv[2]);
$dir  = new RecursiveIteratorIterator($dir);


foreach($dir as $file)
{
   echo $file . "\n";
   copy($file, 'phar://newphar/' . $file);
}


?>
```

Phar

- ☑ Packing files in a directory, just a bit more

```
<?php
$phar = new Phar($argv[1], 0, 'newphar');
$dir  = new RecursiveDirectoryIterator($argv[2]);
$dir  = new RecursiveIteratorIterator($dir);
$dir  = new RegexIterator($dir, '/'.$argv[3].'/');
$phar->startBuffering();
foreach($dir as $file)
{
   echo $file . "\n";
   copy($file, 'phar://newphar/' . $file);
}
$phar->compressAllFilesBZIP2();
$phar->stopBuffering();
?>
```

Phar

- ☑ Packing files in a directory, using ArrayAccess

```php
<?php
$phar = new Phar($argv[1], 0, 'newphar');
$dir  = new RecursiveDirectoryIterator($argv[2]);
$dir  = new RecursiveIteratorIterator($dir);
$dir  = new RegexIterator($dir, '/'.$argv[3].'/');
$phar->startBuffering();
foreach($dir as $file) {
   $f = $dir->getSubPathName();
   echo $f . "\n";
   $phar[$f] = file_get_contents($file);
}
$phar->compressAllFilesBZIP2();
$phar->stopBuffering();
?>
```

Self referencing PHARs

- ☑ PHAR stubs can reference themselves

```
<?php
Phar::mapPhar('myphar.phar', 'myphar');
include 'phar://myphar/main.php';
__HALT_COMPILER();
?>

$> php myphar.phar
```

Self referencing PHARs

- ☑ PHAR stubs might still be used when extracted

```
<?php
$l = Phar::getExtractList();
if(!isset($l[__FILE__]) && !isset($l['phar://myphar'])) {
   Phar::mapPhar('myphar.phar', 'myphar');
}
include 'phar://myphar/main.php';
__HALT_COMPILER();
?>

$> php myphar.phar
```

Self referencing PHARs

- ☑ PHAR stubs can use metadata for bootstrapping

```
#!/usr/bin/php
<?php
$phar = new Phar(__FILE__);
$meta = $phar->getMetaData();
if (isset($meta['boot'])) {
   require_once 'phar://'.__FILE__.'/'.$meta['boot'];
}
__HALT_COMPILER();
?>

$> phar.phar meta-set –f myphar.phar –k boot –m main.php
$> ./myphar.phar
```

Self referencing PHARs

☑ PHAR stubs can implement __autoload

```
#!/usr/bin/php
<?php
function __autoload($cn) {
   $f = 'phar://'.__FILE__.'/'.strtolower($cn).'.inc';
   if (file_exists($f)) {
      require($f);
   }
}
new MyPhar();
__HALT_COMPILER();
?>

$> phar.phar meta-set –f myphar.phar –k boot –m main.php
$> php myphar.phar
```

phar.phar

- ☑ Command line tool to deal with PHAR packages

```
$> phar.phar
No command given, check /usr/bin/phar.phar help

$> phar.phar help-list
compress extract help help-list info list meta-del
meta-get meta-set pack sign stub-get stub-set tree

$> phar.phar help help
help    This help or help for a selected command.

        Optional arguments:

        ...   Optional command to retrieve help for.
```

At Last some Hints

- ☑ List of all SPL classes — PHP 5.0.0

```
php –r 'print_r(array_keys(spl_classes()));'
```

- ☑ Reflection of a built-in class — PHP 5.1.2

```
php --rc <Class>
```

- ☑ Reflection of a function or method — PHP 5.1.2

```
php --rf <Function>
```

- ☑ Reflection of a loaded extension — PHP 5.1.2

```
php --re <Extension>
```

- ☑ Extension information/configuration — PHP 5.2.2

```
php --ri <Extension>
```

THANK YOU

- ☑ This Presentation
  http://somabo.de/talks/
- ☑ SPL Documentation
  http://php.net/~helly
- ☑ SPL_Types
  http://pecl.php.net/package/spl_types
- ☑ Phar
  http://pecl.php.net/packages/phar
  http://php.net/phar