PEAR, PECL, Composer, SPL

Disclaimer: вы смотрите просто запись лекции, это **НЕ** специально подготовленный видеокурс!

Intro

Sometimes, when we need some typical solution, that solution already exists. Like many other programming languages, PHP has a set of libraries and tools that includes:

- PEAR (PHP Extension and Application Repository) – a framework and distribution system for reusable PHP components.
- PECL (PHP Extension Community Library) – a repository for all known PHP extensions.
- Composer – a dependency manager for PHP.
- SPL – standard PHP Library.

While these two solutions slowly become outdated (e.g., PEAR is being replaced by Composer) sometimes you may still find here what you are looking for.

PEAR: quick installation guide and usage sample

1. Go to https://pear.php.net/manual/en/installation.getting.php.
2. Get https://pear.php.net/go-pear.phar (and place it in a folder for PEAR installation).
3. Run “php go-pear.phar” (with Administrator permissions).
4. Add that folder to your PATH system variable.
5. Find a component you want: https://pear.php.net/packages.php.
6. Run (e.g.) “pear install Numbers_Words-0.18.2”.
7. Use the package in your application.

```php
<?php

require_once('../../WebSoft/PEAR/pear/Numbers/Words/Locale/en/US.php');
$sourceNumber = 784513245178;
$transformer = new Numbers_Words_Locale_en_US();
echo $finalString = $transformer->toWords($sourceNumber);
// seven hundred eighty-four billion five hundred thirteen million
// two hundred forty-five thousand one hundred seventy-eight
```

PECL: quick installation guide and usage sample

1. Find a component you want: https://pecl.php.net/packages.php.
2. Download it (follow instructions in the package descriptions). E.g., you want a RAR archives support (https://pecl.php.net/package/rar).
3. Add it to your php extensions set. (Copy *php_rar.dll* and *php_rar.pdb* files to your PHP extension dir and add *extension=rar* to your *php.ini* file.)
4. Use the extension in your application.

```php
<?php

$rarFile = rar_open('02_PECL_usage.rar') or exit("Can't open Rar archive");
$entries = rar_list($rarFile);

foreach ($entries as $entry) {
    echo 'Filename: ' . $entry->getName() . "\n";
    echo 'Packed size: ' . $entry->getPackedSize() . "\n";
    echo 'Unpacked size: ' . $entry->getUnpackedSize() . "\n";
}

rar_close($rarFile);
```

```
/*
Filename: cat1.bmp
Packed size: 187513
Unpacked size: 327414
Filename: cat2.bmp
Packed size: 1419119
Unpacked size: 2160054
Filename: cat3.bmp
Packed size: 1154682
Unpacked size: 2107254
*/
```

Composer: a more detailed overview

Composer is a tool for dependency management in PHP. It allows you to declare the libraries your project depends on and it will manage (install/update) them for you.

You’ll need Composer to run multiple open source PHP projects from GitHub and other repositories, and you should use Composer to create your own projects.

Composer: quick installation guide

1. Go to https://getcomposer.org.
2. Download https://getcomposer.org/Composer-Setup.exe.
3. Run it and follow instructions ☺.

P.S. There's nice documentation for each and every occasion.

Composer: quick usage guide

1. Imagine, you want quick and easy CLI progress bars.
2. And you’ve decided to use this solution: https://github.com/guiguiboy/PHP-CLI-Progress-Bar.
3. Create a new directory.
4. Run “composer require guiguiboy/php-cli-progress-bar” in that directory.
5. Now – just add it to your project and use it.

```php
<?php

require 'vendor/autoload.php';

$progressBar = new \ProgressBar\Manager(0, 10);

for ($i = 0; $i <= 10; $i++) {
    $progressBar->update($i);
    sleep(1);
}
// 4/10 [====================>--------------------------------] 40.00% 00:00:06
```

SPL: a more detailed overview

The Standard PHP Library (**SPL**) is a collection of interfaces and classes that are meant to solve common problems.

SPL provides a set of standard data structures, a set of iterators to traverse over objects, a set of interfaces, a set of standard Exceptions, a number of classes to work with files, and it provides a set of functions like spl_autoload_register().

SPL is a part of PHP core, so no installation required.

Details: https://www.php.net/manual/en/book.spl.php

SPL: a quick sample

This course is not about SPL (that may really need a separate training), so – let’s just look at a couple of samples.

```php
<?php

// A simple recursive directory traverser

$directory = new \RecursiveDirectoryIterator("../");
$iterator = new \RecursiveIteratorIterator($directory);
foreach ($iterator as $info) {
    if ($info->getType() === 'file') {
        echo $info->getBasename() . " = " . $info->getSize() . "\n";
    }
}

// 03_php_comments.php = 238
// 04_php_case_sensitivity.php = 33
// 05_variables_declaration.php = 438
// ...
```

SPL: a quick sample

SPL provides a variety of standard solutions for commonly used data structures, e.g., stack and queue.

```php
<?php

// Stack
$stack = new \SplStack();
$stack->push(1);
$stack->push(2);
$stack->push(3);
echo  $stack->pop() . "\n"; // 3
echo  $stack->pop() . "\n"; // 2
echo  $stack->pop() . "\n"; // 1
```

```php
// Queue
$queue = new \SplQueue();
$queue->enqueue(1);
$queue->enqueue(2);
$queue->enqueue(3);
echo  $queue->dequeue() . "\n"; // 1
echo  $queue->dequeue() . "\n"; // 2
echo  $queue->dequeue() . "\n"; // 3
```

Conclusion

These tools are not something one should “learn by heart”, they’re just tools. Useful ones. Just keep them in mind, read documentation and use existing well-tested solutions to avoid “re-inventing the wheel”.

PEAR, PECL, Composer, SPL

Disclaimer: вы смотрите просто запись лекции, это **НЕ** специально подготовленный видеокурс!