

# Interoperabilidad

Ricardo Pérez López

IES Doñana, curso 2019/2020

## Índice general

<b>1. Versionado semántico</b>	<b>2</b>
<b>2. Composer</b>	<b>2</b>
2.1. Paquetes . . . . .	2
2.2. Packagist . . . . .	2
2.3. Dependencias . . . . .	2
2.3.1. <code>composer.json</code> y <code>composer.lock</code> . . . . .	2
2.4. Versiones y restricciones . . . . .	2
2.4.1. Versión exacta . . . . .	2
2.4.2. Rango ( <code>&gt;</code> , <code>&gt;=</code> , <code>&lt;</code> , <code>&lt;=</code> , <code>!=</code> , <code>,</code> , <code> </code> ) . . . . .	2
2.4.3. Guión ( <code>-</code> ) . . . . .	3
2.4.4. Asterisco ( <code>*</code> ) . . . . .	3
2.4.5. Tilde ( <code>~</code> ) . . . . .	3
2.4.6. Gorrito ( <code>^</code> ) . . . . .	3
2.4.7. Nombres de rama . . . . .	3
2.4.8. Estabilidad mínima . . . . .	3
2.4.9. Comprobador online de restricciones . . . . .	3
2.5. Comandos básicos . . . . .	3
2.5.1. <code>require</code> . . . . .	3
2.5.2. <code>install</code> . . . . .	4
2.5.3. <code>update</code> . . . . .	4
2.6. Entornos de desarrollo y producción . . . . .	4
<b>3. Autocarga de clases</b>	<b>4</b>
3.1. <code>spl_autoload_register()</code> . . . . .	4
3.2. PSR-4 . . . . .	4
3.3. Autoloader de Composer . . . . .	4
<b>4. Ejemplos</b>	<b>4</b>
4.1. <code>mpdf/mpdf</code> . . . . .	4
4.2. <code>ramsey/uuid</code> . . . . .	4
4.3. <code>doctrine/inflector</code> . . . . .	4

<b>5. Recomendaciones PSR del PHP-FIG (Framework Interop Group)</b>	<b>5</b>
5.1. PSR-1: Basic Coding Standard . . . . .	5
5.2. PSR-2: Coding Style Guide . . . . .	5
5.3. PSR-4: Autoloading Standard . . . . .	5
5.4. PSR-5: PHPDoc Standard (borrador) . . . . .	5
5.5. PSR-11: Extended Coding Style Guide (borrador) . . . . .	5
5.6. PSR-19: PHPDoc tags (borrador) . . . . .	5
<b>6. Paquetes de Atom y herramientas externas #opcional</b>	<b>5</b>
6.1. PHP_CodeSniffer . . . . .	5
6.2. PHP-CS-Fixer . . . . .	5
6.3. Yii2-Shell . . . . .	5
<b>7. Ejercicios</b>	<b>5</b>
7.1. De versionado semántico . . . . .	5
7.2. De versiones y restricciones . . . . .	5
7.3. De uso básico de Composer . . . . .	5
7.4. De buscar paquetes en Packagist que tengan una funcionalidad concreta y usarlos en un ejemplo . . . . .	6

## 1. Versionado semántico

## 2. Composer

### 2.1. Paquetes

### 2.2. Packagist

### 2.3. Dependencias

#### 2.3.1. `composer.json` y `composer.lock`

### 2.4. Versiones y restricciones

#### 2.4.1. Versión exacta

ricpelo's note: - `x.y.z` es exactamente esa versión

- `x.y` equivale a `x.y.0`

- `x` equivale a `x.0.0`

- Se usa en los siguientes tipos de restricciones

#### 2.4.2. Rango (`>`, `>=`, `<`, `<=`, `!=`, `,`, `|`)

### 2.4.3. Guión (-)

### 2.4.4. Asterisco (\*)

ricpelo's note: -  $x.y.*$  equivale a  $\geq x.y < x.(y+1)$   
-  $x.*$  equivale a  $\geq x < (x+1)$   
-  $*$  equivale a cualquiera

### 2.4.5. Tilde (~)

ricpelo's note: Especifica una versión mínima y permite que avance el último dígito indicado pero no el primero.

ricpelo's note: -  $\sim x.y.z$  equivale a  $\geq x.y.z < x.(y+1)$   
-  $\sim x.y$  equivale a  $\geq x.y < (x+1)$   
-  $\sim x$  equivale a  $\geq x < (x+1)$

### 2.4.6. Gorrito (^)

ricpelo's note: También permite especificar una versión mínima, pero impide actualizaciones que rompen la compatibilidad hacia atrás (es decir, permite que aumenten todos los números excepto el *mayor*).

ricpelo's note: -  $\wedge x.y.z$  equivale a  $\geq x.y.z < (x+1)$   
-  $\wedge x.y$  equivale a  $\geq x.y < (x+1)$   
-  $\wedge x$  equivale a  $\geq x < (x+1)$

### 2.4.7. Nombres de rama

**2.4.7.1. dev-master** ricpelo's note: Equivale al último commit de la rama `master`.

**2.4.7.2. 5.1.x-dev** ricpelo's note: Equivale al último commit de la rama `5.1`.

### 2.4.8. Estabilidad mínima

### 2.4.9. Comprobador online de restricciones

ricpelo's note: Probar con `laravel/laravel`.

## 2.5. Comandos básicos

### 2.5.1. require

### 2.5.2. install

### 2.5.3. update

## 2.6. Entornos de desarrollo y producción

# 3. Autocarga de clases

## 3.1. spl\_autoload\_register()

ricpelo's note: - Cuando se llama sin argumentos, registra el manejador predeterminado (`spl_autoload()`), el cual autocarga el archivo `.php` a partir del nombre de la clase **en minúsculas** (la clase `A` la busca en `a.php`).

```
- spl_autoload_register(function ($c) { include "$c.php"; });  
- spl_autoload_register(function ($c) { include strtolower(str_replace('\\',  
'/', $c) . '.php'); });
```

## 3.2. PSR-4

## 3.3. Autoloader de Composer

# 4. Ejemplos

## 4.1. mpdf/mpdf

ricpelo's note:

```
<?php  
  
// composer require mpdf/mpdf  
  
require 'vendor/autoload.php';  
  
$mpdf = new Mpdf\Mpdf();  
$mpdf->WriteHTML('<h1>¡Hola, mundo!</h1>');  
$mpdf->Output('salida.pdf');
```

## 4.2. ramsey/uuid

## 4.3. doctrine/inflector

## 5. Recomendaciones PSR del PHP-FIG (Framework Interop Group)

ricpelo's note: PHP Standard Recommendation

### 5.1. PSR-1: Basic Coding Standard

### 5.2. PSR-2: Coding Style Guide

### 5.3. PSR-4: Autoloading Standard

### 5.4. PSR-5: PHPDoc Standard (borrador)

### 5.5. PSR-11: Extended Coding Style Guide (borrador)

### 5.6. PSR-19: PHPDoc tags (borrador)

## 6. Paquetes de Atom y herramientas externas #opcional

ricpelo's note:

```
$ composer global require --prefer-dist friendsofphp/php-cs-fixer "squizlabs/php_codesniffer"
$ sudo ln -sf /opt/composer ~/.config/composer
$ apm install linter-phpcs php-cs-fixer
$ phpcs --config-set default_standard /opt/composer/vendor/yiisoft/yii2-coding-standards/Yii2
```

ricpelo's note: <https://github.com/ricpelo/conf>

### 6.1. PHP\_CodeSniffer

### 6.2. PHP-CS-Fixer

### 6.3. Yii2-Shell

## 7. Ejercicios

### 7.1. De versionado semántico

### 7.2. De versiones y restricciones

### 7.3. De uso básico de Composer

**7.4. De buscar paquetes en Packagist que tengan una funcionalidad concreta y usarlos en un ejemplo**