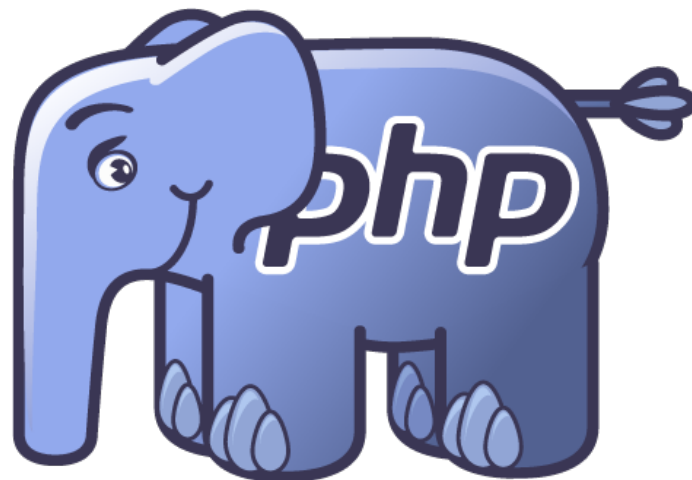


PHP Avanzado

Federico Ferreri

Agosto 2015



Unidad 3

Componentes

- # Por que usar componentes?
- # Que son los componentes?
- # Buscar y encontrar componentes
- # Composer
- # Usando componentes en PHP

Por que usar componentes?

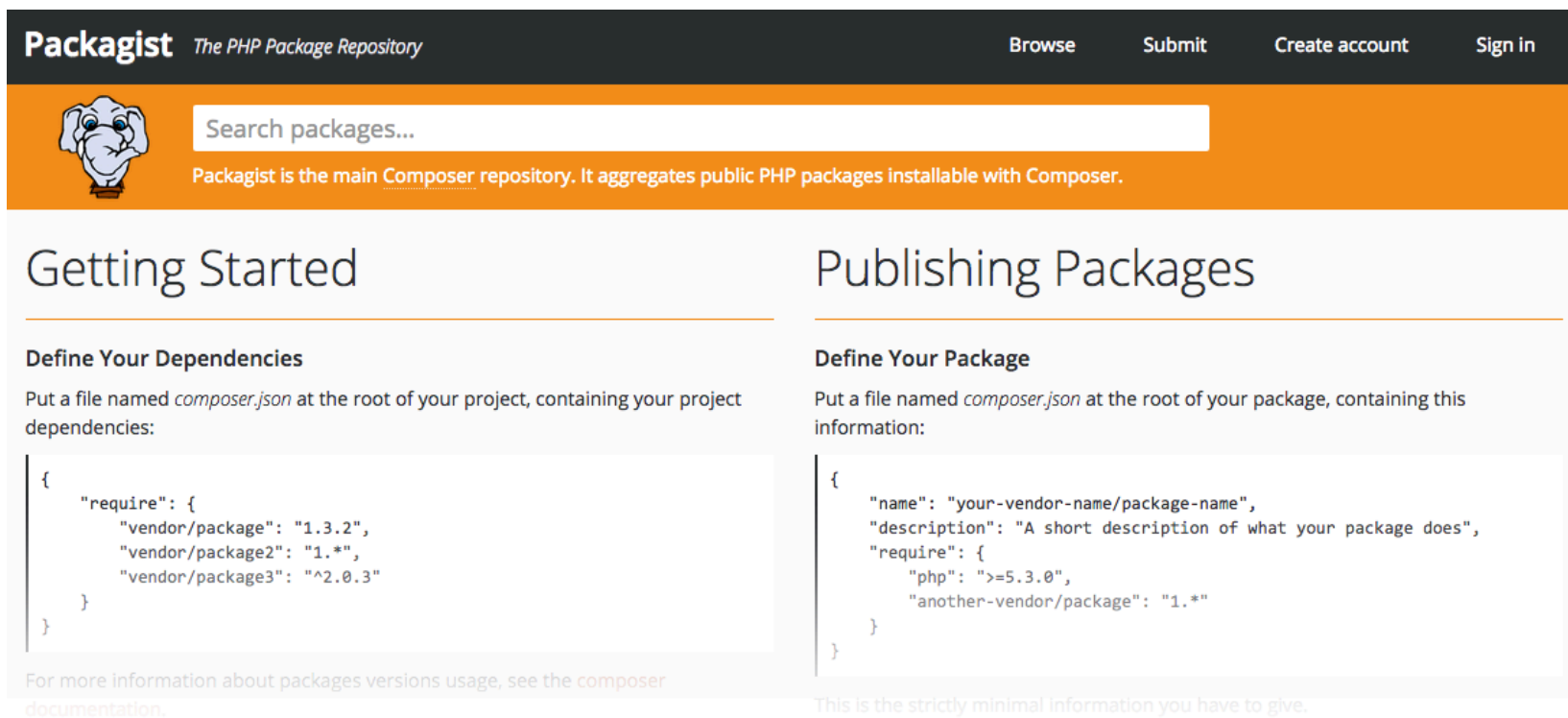
- ▶ Previo a la aparición masiva de componentes escritos en PHP, el panorama para el programador era mas complejo. La falta de standards y formas de integración hacían que cualquier funcionalidad no provista por el framework tenía que ser programada.
- ▶ Hoy día y gracias a la aparición PHP 5.3+ y manejadores de componentes como Composer mucho mas sencillo resolver el problema anterior.
- ▶ Dada la gran calidad de los componentes existentes, sería una tontería ponerse a re inventar la rueda tratando de resolver problemas como routing, acceso a archivos o acceso a recursos en la web.
- ▶ El uso de componentes acelera los tiempos de desarrollo y mejoran la calidad del producto final.
- ▶ El uso de componentes permite la reutilización de código.

Que son los componentes?

- ▶ Es un combo de código y otros recursos (imágenes, JS, etc.) que resuelven un problema específico. Por ejemplo, enviar mensajes vía HTTP, parsear un archivo CSV o ejecutar un comando en un servidor remoto usando SSH.
- ▶ Como todo en la vida, hay componentes buenos y componentes malos. Como características de un buen componente se pueden nombrar:
 - ▶ Que sea específico: que resulta un problema solo y bien. Provee una interface simple.
 - ▶ Pequeño: usa la menor cantidad de código posible.
 - ▶ Cooperativo: debe interactuar bien con otros componentes.
 - ▶ Bien testado: un componente bien específico y pequeño es fácilmente testeable. Los componentes de buena calidad proveen sus propios juegos de tests y cubren la mayoría de la funcionalidad.
 - ▶ Bien documentado: debe incluir información de como instalarlo, configurarlo y utilizarlo. La documentación no solo debe ser de la funcionalidad, también un buen componente tiene su código bien comentado (DocBlocks por ejemplo)

Buscar y encontrar componentes

- ▶ El lugar para buscar y encontrar componentes PHP es <https://packagist.org/>
- ▶ Packagist es el repositorio principal de Composer
- ▶ Es un índice de componentes, que incluye información adicional como estadísticas de uso del componente y feedback de los usuarios.



The screenshot shows the Packagist website interface. At the top, there's a navigation bar with links: Browse, Submit, Create account, and Sign in. Below this is an orange banner with the Packagist logo (an elephant) and a search bar labeled 'Search packages...'. A text line below the banner states: 'Packagist is the main Composer repository. It aggregates public PHP packages installable with Composer.'

The main content area is divided into two columns. The left column is titled 'Getting Started' and contains a section 'Define Your Dependencies' with the instruction: 'Put a file named `composer.json` at the root of your project, containing your project dependencies:'. Below this is a code block showing a JSON snippet for dependencies:

```
{
  "require": {
    "vendor/package": "1.3.2",
    "vendor/package2": "1.*",
    "vendor/package3": "^2.0.3"
  }
}
```

Below the code block, it says: 'For more information about packages versions usage, see the [composer documentation](#).'

The right column is titled 'Publishing Packages' and contains a section 'Define Your Package' with the instruction: 'Put a file named `composer.json` at the root of your package, containing this information:'. Below this is a code block showing a JSON snippet for package information:

```
{
  "name": "your-vendor-name/package-name",
  "description": "A short description of what your package does",
  "require": {
    "php": ">=5.3.0",
    "another-vendor/package": "1.*"
  }
}
```

Below the code block, it says: 'This is the strictly minimal information you have to give.'

Composer

- ▶ Composer es un manejador de dependencias para PHP
- ▶ Permite la instalación de componentes PHP y sus dependencias. Se encarga de bajar e instalar todo lo necesario (dependencias) para el correcto funcionamiento del componente.
- ▶ Composer resuelve el problema de autoloading generando automáticamente un autoloader PSR-4 compatible en forma automática.
- ▶ De <https://getcomposer.org/> se puede bajar Composer y también está toda la documentación necesaria para poder usarlo, tanto como usuario final que baja e instala componentes dentro de sus proyectos, como para los que desarrollan componentes.
- ▶ Composer requiere PHP 5.3.2+ para funcionar (namespaces).

Composer: instalación

- ▶ La instalación es sencilla:

- ▶ `curl -sS https://getcomposer.org/installer | php`

- ▶ Para una instalación mas avanzada:

- ▶ `curl -sS https://getcomposer.org/installer | php -- --install-dir=bin --filename=composer`

- ▶ En Windows se puede instalar utilizando el paquete de instalación [Composer-Setup.exe](#) disponible en el sitio

Usando componentes en PHP

- ▶ Lo primero que hay que hacer es instalar los componentes que creemos vamos a utilizar en nuestro proyecto.
- ▶ Paso a paso:
 1. Crear un directorio vacío
 2. Ejecutar ``composer require vendor/componente``
 3. Incluir "vendor/autoload.php" en nuestro PHP
 4. Escribir nuestro código!
- ▶ Ejemplo:
 - ➔ `composer require nesbot/carbon`
Using version ^1.20 for nesbot/carbon
./composer.json has been created
Loading composer repositories with package information
Updating dependencies (including require-dev)
 - Installing symfony/translation (v2.7.3)
Loading from cache
 - Installing nesbot/carbon (1.20.0)
Loading from cacheWriting lock file
Generating autoload files