EBook Gratis

APRENDIZAJE composer-php

Free unaffiliated eBook created from
Stack Overflow contributors.

#composer-php

Tabla de contenido

Acerca de

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: composer-php

It is an unofficial and free composer-php ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official composer-php.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Capítulo 1: Empezando con composer-php

Observaciones

Esta sección proporciona una descripción general de qué es composer-php y por qué un desarrollador puede querer usarlo.

También debe mencionar cualquier tema grande dentro de composer-php, y vincular a los temas relacionados. Dado que la Documentación para composer-php es nueva, es posible que deba crear versiones iniciales de esos temas relacionados.

Examples

Visión general

Composer es una herramienta para la gestión de dependencias en PHP. Le permite declarar las bibliotecas de las que depende su proyecto y las administrará (instalará / actualizará) por usted.

Composer no es un gestor de paquetes en el mismo sentido que Yum o Apt. Sí, se trata de "paquetes" o bibliotecas, pero los administra en función de cada proyecto, instalándolos en un directorio (por ejemplo, un proveedor) dentro de su proyecto.

Composer requiere PHP 5.3.2+ para ejecutarse. También se requieren algunos ajustes confidenciales de php y banderas de compilación, pero al usar el instalador se le advertirá sobre cualquier incompatibilidad.

Para instalar paquetes desde fuentes en lugar de simples archivos zip, necesitará git, svn, fossil o hg dependiendo de cómo se controle la versión del paquete.

Instalando Composer en Ubuntu

Antes de descargar e instalar Composer, debemos asegurarnos de que nuestro servidor tenga todas las dependencias instaladas.

Primero, actualice el caché del administrador de paquetes ejecutando:

```
sudo apt-get update
```

Ahora, instalemos las dependencias. Necesitaremos `curl` para descargar Composer y `php5-cli` para instalarlo y ejecutarlo. `git` es utilizado por Composer para descargar dependencias de proyectos. Todo se puede instalar con el siguiente comando:

```
sudo apt-get install curl php5-cli git
```

Ahora vamos a instalarlo:

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

Esto descargará e instalará Composer como un comando de todo el sistema llamado composer, en / usr / local / bin. La salida debería verse así:

```
Output
#!/usr/bin/env php
All settings correct for using Composer
Downloading...

Composer successfully installed to: /usr/local/bin/composer
Use it: php /usr/local/bin/composer
```

Para probar su instalación, ejecute:

```
composer
```

Y deberías obtener una salida similar a esta:

```
Output
   ______
  / ____/___  ____ ___  ____  ____  ________  _____
 / /   / __ \/ __ `__ \/ __ \/ __ \/ ___/ _ \/ ___/
/ /___/ /_/ / / / / / / /_/ / /_/ (__  )  __/ /
\____/\____/_/ /_/ /_/ .___/\____/____/\___/_/
                    /_/
Composer version 1.0-dev (9859859f1082d94e546aa75746867df127aa0d9e) 2015-08-17 14:57:00

Usage:
 command [options] [arguments]

Options:
 --help (-h)           Display this help message
 --quiet (-q)          Do not output any message
 --verbose (-v|vv|vvv) Increase the verbosity of messages: 1 for normal output, 2 for more
verbose output and 3 for debug
 --version (-V)        Display this application version
 --ansi                Force ANSI output
 --no-ansi             Disable ANSI output
 --no-interaction (-n) Do not ask any interactive question
 --profile             Display timing and memory usage information
 --working-dir (-d)    If specified, use the given directory as working directory.

....
```

Instalación en Windows

Aquí simplemente utilizaremos el instalador.

Esta es la forma más fácil de configurar Composer en su máquina.

Descargue y ejecute Composer-Setup.exe . Instalará la última versión del compositor y configurará su `PATH` para que pueda llamar al `composer` desde cualquier directorio en su línea de

comando.

Nota : Cierre su terminal actual. Pruebe el uso con un nuevo terminal: esto es importante ya que la RUTA solo se carga cuando se inicia el terminal.

Nota-2 : Configurar `PATH` en Windows 10

1. Haga clic con el botón derecho en inicio (logotipo de Windows) -> `system ->Advance system settings->Environment variables->System variables[below box] ->` seleccione `Path` y haga clic en `Edit`
2. Haga clic en Nuevo y agregue este valor `C:\ProgramData\ComposerSetup\bin`
3. Ahora abre tu terminal [cmd] y prueba el `composer --version`

Lea Empezando con composer-php en línea: https://riptutorial.com/es/composer-php/topic/3267/empezando-con-composer-php

Capítulo 2: Auto carga con compositor

Examples

Autocarga

Para las bibliotecas que especifican información de carga automática, Composer genera un archivo proveedor / autoload.php. Simplemente puede incluir este archivo y obtendrá la carga automática de forma gratuita.

```
require __DIR__ . '/vendor/autoload.php';
```

Esto hace que sea realmente fácil de usar código de terceros. Por ejemplo: si su proyecto depende de Monolog, puede comenzar a usar las clases y se cargarán automáticamente.

```
$log = new Monolog\Logger('name');
$log->pushHandler(new Monolog\Handler\StreamHandler('app.log', Monolog\Logger::WARNING));
$log->addWarning('Foo');
```

Incluso puede agregar su propio código al autocargador agregando un campo de `autoload` a `composer.json`

```
{
    "autoload": {
        "psr-4": {"Acme\\": "src/"}
    }
}
```

Composer registrará un autocargador PSR-4 para el espacio de nombres Acme.

Se define una asignación de espacios de nombres a directorios. El directorio `src` estaría en la raíz de su proyecto, en el mismo nivel que el directorio del proveedor. Un ejemplo de nombre de archivo sería `src/Foo.php` contiene una clase `Acme\Foo` .

Después de agregar el campo de carga automática, debe volver a ejecutar `dump-autoload` para volver a generar el archivo `vendor/autoload.php` .

Incluir ese archivo también devolverá la instancia del autocargador, por lo que puede almacenar el valor de retorno de la llamada de inclusión en una variable y agregar más espacios de nombres. Esto puede ser útil para las clases de carga automática en un conjunto de pruebas, por ejemplo.

```
$loader = require __DIR__ . '/vendor/autoload.php';
$loader->add('Acme\\Test\\', __DIR__);
```

Además de la carga automática de PSR-4, Composer también admite PSR-0, classmap y carga automática de archivos.

Lea Auto carga con compositor en línea: https://riptutorial.com/es/composer-php/topic/5915/auto-carga-con-compositor

Capítulo 3: Cómo utilizar repositorios privados con Composer

Parámetros

Parámetros	Detalles
repositorios	Indica a Composer dónde puede descargar los paquetes requeridos.
tipo: vcs	Le dice a Composer cómo tratar el repositorio.
url: http: // ...	Le dice a Composer dónde está el repositorio.

Observaciones

Use la sintaxis de `type: "vcs"` para usar repositorios privados .

Para administrar el acceso al repositorio privado mientras se desarrolla en una máquina local, use un archivo `auth.json` y no lo `auth.json` en su repositorio de proyectos. En su lugar, dé acceso a cada desarrollador individual al repositorio privado, de modo que, utilizando cada uno su propio archivo de `auth.json` NO COMPROMETIDOS, pueden obtener el repositorio remoto con la `composer install composer update` o la `composer update` .

Consejo: Coloque el archivo `auth.json` en el archivo `.gitignore` de su repositorio `git` .

Si está utilizando un sistema de integración continua, use la variable de entorno `COMPOSER_AUTH` .

Examples

sintaxis composer.json

```
{
    "name": "your/package",
    "license": "proprietary",
    "type": "project",
    "description": "How to load an external private Composer package.",
    ...
    "require": {
        "your/private_package": "*"
    },
    ...
    "repositories": [
        {
            "type": "vcs",
            "url": "https://example.com/Your/private-package.git"
        }
    ]
```

```
}
```

Lea Cómo utilizar repositorios privados con Composer en línea: https://riptutorial.com/es/composer-php/topic/3554/como-utilizar-repositorios-privados-con-composer

Creditos

S. No	Capítulos	Contributors
1	Empezando con composer-php	Abdelaziz Dabebi, Atiqur, Community, Nic Wortel
2	Auto carga con compositor	Adil Abbasi
3	Cómo utilizar repositorios privados con Composer	Aerendir