

fundacionunirioja.adrformacion.com © ADR Infor SL
Héctor García González

Extensiones y librerías © ADR Infor SL

fundacionunirioja.adrformacion.com © ADR Infor SL
Héctor García González

fundacionunirioja.adrformacion.com © ADR Infor SL
Héctor García González

Indice

Extensiones y librerías	3
Extensiones	4
Búsqueda de extensiones	5
Activación de extensiones	10
Instalación de extensiones	11
Carpeta de extensiones	12
Instalación de extensiones en Windows	12
Instalación de extensiones en Linux	15
Librerías	16
Ejemplo práctico: Librería PHPMailer	19
Descarga de PHPMailer	19
Añadir PHPMailer a nuestro proyecto	20
Enviar un e-mail con PHPMailer	20
Hemos aprendido	22
Ejercicios	24
Ejercicio 1: Instala la extensión para MongoDB	24
Pasos a seguir	24
Si tu servidor es Windows	24
Si tu servidor es Linux	24
Ejercicio 2: Envía un correo con PHPMailer	24
Lo necesario para comenzar	25
Pasos a seguir	25
Recursos	26
Enlaces de Interés	26
Preguntas Frecuentes	26
Glosario.	26

Extensiones y librerías



Al finalizar esta unidad el alumno será capaz de buscar, instalar y activar extensiones de PHP, así como de utilizar librerías de terceros en sus desarrollos.

PHP es un lenguaje con gran cantidad de utilidades y funciones nativas, que nos permiten realizar multitud de procesos complejos de forma sencilla. No obstante, la potencia de PHP no reside en ser un lenguaje muy completo (que lo es), sino en ser capaz de **integrar dentro de él nuevas funcionalidades** cuando nuestros desarrollos así lo requieran.

Es posible que necesitemos realizar algún proceso, relativamente estándar, para el que PHP no tenga ninguna función o utilidad implementada. Sin embargo, será muy sencillo encontrar dicha funcionalidad, ya desarrollada por otros programadores e integrarla dentro de nuestro proyecto.



Necesitamos generar un informe en **Excel** para que el usuario pueda descargarlo.

No encontramos en PHP ninguna funcionalidad para generar un archivo Excel. Pero **encontramos utilidades desarrolladas por terceros** con facilidad.

Integrar estas utilidades en nuestros desarrollos de PHP es muy sencillo y las tenemos disponibles en el acto.

En esta capacidad de añadir rápidamente cualquier funcionalidad al lenguaje, reside la verdadera fuerza de PHP, haciendo que prácticamente **nada sea imposible de implementar**.

Los pilares en los que se sustenta esta habilidad de PHP son los siguientes:

Facilidad de integración de código externo

PHP dispone de **mecanismos específicos para integrar funcionalidades** dentro del lenguaje de forma sencilla. Su propia estructura **favorece la modularidad** y la facilidad de que el código pueda ser distribuido.

No olvidemos, que PHP es un lenguaje open source que **ha sido desarrollado por la propia comunidad de desarrolladores**. En muchos casos, desarrollos externos de ciertas funcionalidades, han tenido tanto éxito que se han convertido casi en un estándar, llegando a integrarse de forma definitiva con PHP y siendo distribuidos, de base, en las siguientes versiones del lenguaje.

Una comunidad de desarrolladores muy extensa y activa

Gracias a que PHP, **es uno de los lenguajes más utilizados del mundo**, cuenta con una extensa comunidad de programadores que lo utilizan de forma activa a diario.

Esto significa que, con toda seguridad, cuando te encuentres un problema que debes resolver, muchas más personas ya se han encontrado con el mismo problema y lo han resuelto.

Podemos decir que **el alma de PHP reside en su comunidad**, gracias a la cual adquiere la potencia y se mantiene siempre actualizado con respecto a las nuevas necesidades tecnológicas.

Existen dos formas de añadir funcionalidades a PHP:

Extensiones

Las extensiones son módulos que se integran dentro del propio lenguaje de programación.

Necesitan ser **instaladas y configuradas** dentro de PHP para funcionar y suelen utilizarse en casos en que la funcionalidad a añadir es casi un estándar.

Librerías

Las librerías son simplemente **archivos con código PHP** que anexamos a nuestros proyectos mediante un **include**.

Es la forma más simple y generalizada de distribuir código ya que cualquier programador puede distribuir sus códigos de esta forma.

Extensiones

Las extensiones son **módulos que pueden ser integrados de forma directa** dentro de PHP.

Contienen funcionalidades que estarán disponibles como si formasen parte del propio núcleo de PHP. Por lo general, suelen ser códigos muy testados y libres de bugs, especialmente si son extensiones oficiales.

Cuando necesitamos utilizar una extensión de PHP, podemos encontrarnos con cualquiera de los siguientes casos:

La extensión está instalada y activa

Podremos ver que, muchas extensiones, ya son **distribuidas y activadas** por defecto con la instalación básica de PHP, pasando a formar parte del propio lenguaje.

Suele tratarse de extensiones que ofrecen funcionalidades esenciales, pero conservan su formato de extensión para poder ser desactivadas.

Podemos **desactivar una extensión** para eliminar funcionalidades del lenguaje que no vamos a utilizar, mejorando así el rendimiento de PHP.

La extensión está instalada y desactivada

Comprobaremos que hay muchas extensiones que son **distribuidas** con la instalación básica de PHP pero que, por defecto, están **desactivadas**.

Esto suele ser debido a que sus funcionalidades no se utilizan de forma habitual en todos los desarrollos y mantenerlas activas penalizaría el rendimiento del motor de PHP.

Podemos **activar una extensión** desactivada que ya se encuentra instalada para comenzar a utilizar sus funcionalidades en cualquier momento.

Necesitamos instalar la extensión

Habrán muchos casos en los que la extensión **no haya sido distribuida** con la instalación de PHP.

En estos casos deberemos **descargar e instalar** la extensión para, posteriormente, **activarla** y comenzar a usar sus funcionalidades.

Búsqueda de extensiones

Todo comienza con una **necesidad**.

Nos encontraremos durante nuestro desarrollo la necesidad de realizar cierto proceso. Desconocemos por completo si ya existe la forma de realizar dicho proceso en PHP.



Dentro de nuestro proyecto, nos surge la necesidad de utilizar el protocolo SOAP y desconocemos si ya existe la forma de manejarlo en PHP.

Lo primero que necesitaremos saber es, **si existe una extensión** para realizar el proceso que necesitamos y **qué extensión es**. Para esto nos serviremos de las siguientes utilidades:

php.net

Al primer lugar al que debemos acudir es a la página oficial de PHP que, como sabemos, dispone de una documentación muy completa de las funcionalidades del lenguaje.



<http://www.php.net/>

A través de su buscador, podemos localizar funcionalidades específicas para resolver nuestro problema.

The screenshot shows the PHP website with a navigation bar at the top containing links for Downloads, Documentation, Get Involved, and Help. A search bar is located on the right. The main content area features three release announcements, each dated 06 Jul 2017:

- PHP 5.6.31 Released »**: The PHP development team announces the immediate availability of PHP 5.6.31. This is a security release. Several security bugs were fixed in this release. All PHP 5.6 users are encouraged to upgrade to this version. For source downloads of PHP 5.6.31 please visit our [downloads page](#). Windows source and binaries can be found on [windows.php.net/download/](#). The list of changes is recorded in the [Changelog](#).
- PHP 7.1.7 Released »**: The PHP development team announces the immediate availability of PHP 7.1.7. This is a security release with several bug fixes included. All PHP 7.1 users are encouraged to upgrade to this version. For source downloads of PHP 7.1.7 please visit our [downloads page](#). Windows source and binaries can be found on [windows.php.net/download/](#). The list of changes is recorded in the [Changelog](#).
- PHP 7.2.0 Alpha 3 Released »**: The PHP development team announces the immediate availability of PHP 7.2.0 Alpha 3. This release contains fixes and improvements relative to Alpha 2. All users of PHP are encouraged to test this version carefully, and report any bugs and incompatibilities in the [bug tracking system](#). **THIS IS A DEVELOPMENT PREVIEW - DO NOT USE IT IN PRODUCTION!**

On the right side of the page, there are sections for Download (listing versions 5.6.31, 7.0.21, and 7.1.7 with links to Release Notes and Upgrading), Upcoming conferences (listing php[world] 2017, PHP Developer Day 2017, Forum PHP 2017, and Madison PHP Conference 2017), Conferences calling for papers (listing php Central Europe Conference 2017, ZendCon 2017, and Madison PHP Conference 2017), User Group Events, Special Thanks, and Social media (listing @official_php).

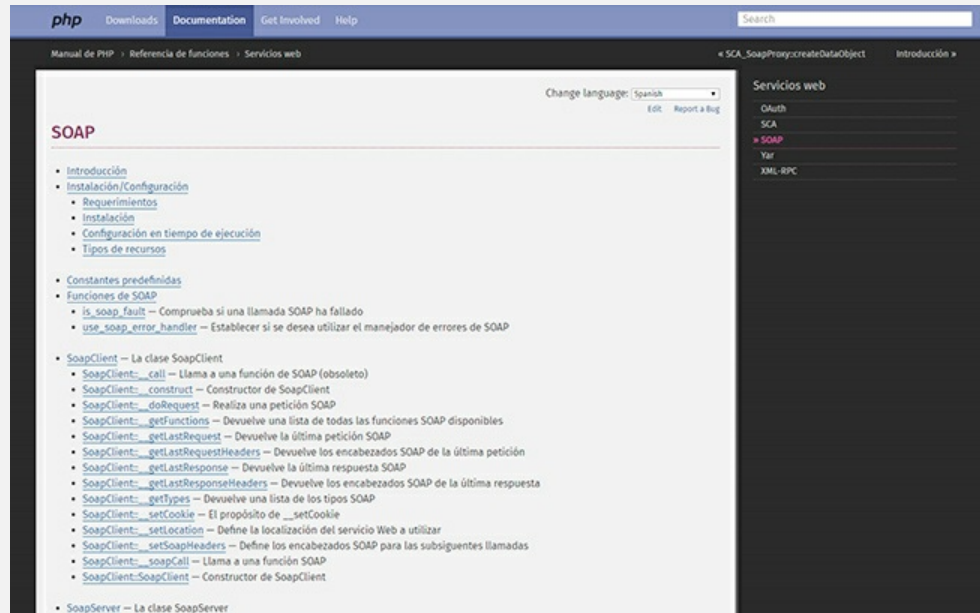
Existen muchas extensiones **oficiales**, que se encuentran documentadas dentro de **php.net**.

Las extensiones descritas aquí, están **mucho más testadas y mejor estructuradas** que cualquier otro código que podamos encontrar en otros lugares. Reciben actualizaciones frecuentes para todas las versiones de PHP con soporte, por lo que, utilizarlas siempre debería ser nuestra primera opción.



Buscando SOAP en **php.net**, encontramos una extensión específica para manejar dicho protocolo.

En la documentación encontraremos una descripción completa de las clases y funcionalidades contenidas en la extensión, ejemplos de su utilización e instrucciones de instalación.



PECL

PECL son las siglas de *PHP Extension Community Library*, o, traducido al castellano: **Librería de extensiones PHP de la comunidad**.

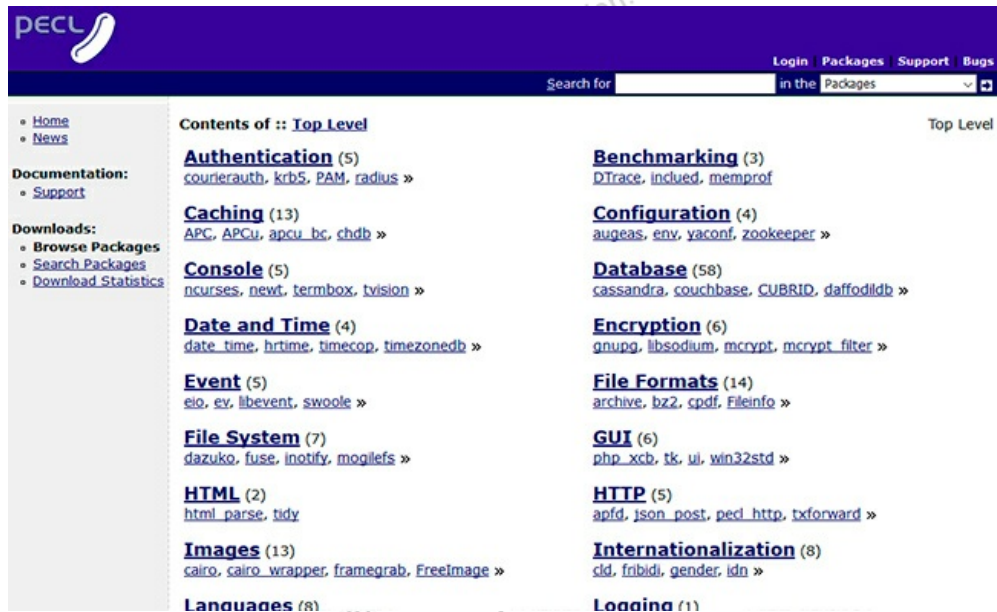
Éste es el repositorio de extensiones de la comunidad de programadores de PHP. En él podrás encontrar gran cantidad de extensiones para realizar infinidad de procesos.



Enlaces: PHP Extension Community Library (PECL)

<https://pecl.php.net/>

Utilizaremos el buscador de PECL para encontrar la extensión adecuada para resolver nuestra necesidad, o pulsaremos sobre **"Browse All Packages"** para obtener una vista categorizada de las extensiones contenidas en la librería.



The screenshot shows the PECL (PHP Extension Community Library) website. The header is purple with the PECL logo and navigation links: Home, News, Documentation, Support, Downloads, Browse Packages, Search Packages, and Download Statistics. A search bar is located in the top right. The main content area is titled 'Contents of :: Top Level' and lists various extension categories with their respective counts and sub-packages.

Category	Count	Sub-packages
Authentication	(5)	courierauth, krb5, PAM, radius »
Caching	(13)	APC, APCu, apcu_bc, chdb »
Console	(5)	ncurses, newt, termbox, tvision »
Date and Time	(4)	date_time, hrtimer, timecop, timezonedb »
Event	(5)	eio, ev, libevent, swoole »
File System	(7)	dazuko, fuse, inotify, mogilefs »
HTML	(2)	html_parse, tidy »
Images	(13)	cairo, cairo_wrapper, framegrab, FreeImage »
Languages	(8)	
Benchmarking	(3)	DTrace, included, memprof »
Configuration	(4)	augeas, env, yaconf, zookeeper »
Database	(58)	cassandra, couchbase, CUBRID, daffodildb »
Encryption	(6)	gnupg, libsodium, mcrypt, mdecrypt_filter »
File Formats	(14)	archive, bz2, cpdf, Fileinfo »
GUI	(6)	php_xcb, tk, ui, win32std »
HTTP	(5)	apfd, json_post, ped_http, txforward »
Internationalization	(8)	cid, fribidl, gender, idn »
Logging	(1)	

Todas las extensiones incluidas en PECL están **ampliamente testadas** y **libres de bugs**, por lo que podemos confiar en ellas para utilizarlas en nuestros desarrollos.



Tenemos la necesidad de conectarnos a una base de datos **Oracle** desde PHP.

Buscamos en PECL y encontramos la extensión **oci8** que se encarga de esta funcionalidad.

The screenshot shows the PECL website interface. On the left, there are navigation links: Home, News, Documentation (Support), and Downloads (Browse Packages, Search Packages, Download Statistics). The main content area displays the package information for **oci8**.

oci8

Package Information

Summary	Extension for Oracle Database
Maintainers	Antony Dovgal (lead) [wishlist] [details] Christopher Jones <christopher.jones@oracle.com> (lead) [details] Wez Furlong <wez@php.net> (lead) [details] Andi Gutmans <andi@zend.com> (lead) [details]
License	PHP
Description	Use the OCI8 extension to access Oracle Database. Use 'ped install oci8' to install for PHP 7. Use 'ped install oci8-2.0.12' to install for PHP 5.2 - PHP 5.6. Use 'ped install oci8-1.4.10' to install for PHP 4.3.9 - PHP 5.1. The OCI8 extension can be linked with Oracle client libraries from Oracle Database 12.1, 11, or 10.2. These libraries are found in your database installation, or in the free Oracle Instant Client from http://www.oracle.com/technetwork/database/features/instant-client/ . Oracle's standard cross-version connectivity applies. For example, PHP OCI8 linked with Instant Client 11.2 can connect to Oracle Database 9.2 onward. See Oracle's note "Oracle Client / Server Interoperability Support" (ID 207303.1) for details.

[Latest Tarball] [Changelog] [View Statistics]
[Browse Source] [Package Bugs] [View Documentation]

Available Releases

Version	State	Release Date	Downloads
2.1.4	stable	2017-04-12	oci8-2.1.4.tgz (187.5kB) DLL [Changelog]
2.1.3	stable	2016-11-11	oci8-2.1.3.tgz (187.4kB) DLL [Changelog]
2.1.2	stable	2016-08-18	oci8-2.1.2.tgz (187.3kB) DLL [Changelog]
2.0.12	stable	2016-08-18	oci8-2.0.12.tgz (187.5kB) DLL [Changelog]
2.1.1	stable	2016-04-18	oci8-2.1.1.tgz (187.9kB) DLL [Changelog]
2.0.11	stable	2016-04-18	oci8-2.0.11.tgz (187.5kB) DLL [Changelog]
2.1.0	stable	2015-12-12	oci8-2.1.0.tgz (186.7kB) DLL [Changelog]
2.0.10	stable	2015-12-12	oci8-2.0.10.tgz (186.9kB) DLL [Changelog]

Desde PECL podremos descargar las extensiones para todas las **versiones de PHP** y todos los **sistemas operativos** en los que está disponible.

Buscadores de internet

Por último, podemos utilizar cualquier **buscador** de internet para buscar extensiones que resuelvan nuestras necesidades.

Instalar extensiones que no estén recogidas en **php.net** o **PECL** puede suponer un riesgo, ya que no está testado por la comunidad. No obstante, si confiamos en el editor de la extensión, podemos hacerlo sin problema.

Es recomendable consultar antes **php.net** o **PECL** antes de instalar una extensión desconocida, por si hubiese otra extensión aprobada por la comunidad.

Activación de extensiones

Una vez localizada la extensión que necesitamos, debemos comprobar si ya está **instalada** y si está **activada**.



Los tres casos que se pueden dar a la hora de requerir una extensión eran:

- Está instalada y activada
- Esta instalada y desactivada
- No está instalada

Los pasos a seguir para comenzar a utilizar una extensión son:

Comprobar si la extensión ya está activa

Para comprobar si una extensión ya está activa, utilizaremos la función `phpinfo()` para mostrar la configuración completa de PHP.

Si la extensión está activada, tendremos un bloque dentro de `phpinfo()` que tendrá su nombre e indicará su configuración.

soap

Soap Client	enabled	
Soap Server	enabled	
Directive	Local Value	Master Value
soap.wsdl_cache	1	1
soap.wsdl_cache_dir	/tmp	/tmp
soap.wsdl_cache_enabled	1	1
soap.wsdl_cache_limit	5	5
soap.wsdl_cache_ttl	86400	86400

Si este es el caso, podemos comenzar a utilizarla sin más.

Comprobar si la extensión está instalada

Para comprobar si la extensión ya está instalada, debemos abrir nuestro archivo `php.ini`. En él buscaremos una línea similar a la siguiente:

```
extension=archivo_extensión
```

Las extensiones se almacenan en archivos. En Windows serán archivos **DLL** y en Linux archivos **SO**.

Dentro del archivo `php.ini` existirá una línea similar a `extension=archivo_extensión` por cada extensión activa. De esta forma PHP **activa o desactiva** extensiones.

Es posible que encontremos esta línea con un punto y coma (;) delante:

```
;extension=archivo_extensión
```

El carácter punto y coma (;) simboliza un **comentario** en php.ini. Si una extensión ha sido instalada, pero no activada, aparecerá la línea comentada dentro de php.ini para que podamos activarla si es necesario.

Si **descomentamos la línea** y **reiniciamos el servidor web**, nuestra extensión estará activa y podemos comenzar a utilizarla.



Ejemplo: Activar extensión soap

Queremos utilizar la extensión **soap** de PHP.

Observamos que no aparece dentro de `phpinfo()` por lo que abrimos nuestro **php.ini** y encontramos la siguiente línea:

```
;extension=php_soap.dll
```

Descomentamos la línea que queda de la siguiente forma:

```
extension=php_soap.dll
```

Reiniciamos el servidor Apache y nuestra extensión **soap** ya está disponible para su uso.

La extensión no está instalada

Si no encontramos el archivo de la extensión dentro de nuestro **php.ini**, o al descomentarlo la extensión no aparece en `phpinfo()`, significa que la extensión **no está instalada**.

Para poder utilizar la extensión debemos proceder a **descargarla e instalarla**.

Instalación de extensiones

La instalación de extensiones en PHP es relativamente sencilla, aunque debemos tener conocimientos de administración del sistema operativo en el que estamos trabajando.

Para instalar una extensión, solo tenemos que realizar tres pasos:

1

Colocar el **archivo de la extensión** en la **carpeta de extensiones** de PHP.

2

Asegurarnos de que están instaladas todas sus **dependencias**.

3

Añadir la línea `extension=archivo_extensión` en **php.ini**

Carpeta de extensiones

La carpeta de extensiones es el lugar en el que **se almacenan todos los archivos de extensiones**.

Si el archivo de la extensión que queremos activar se encuentra en esta carpeta, y tiene todos los recursos y programas que utiliza instalados, al añadir la línea de extensión correspondiente en **php.ini** y reiniciar el servidor web, la extensión se activará.

Para conocer cuál es la carpeta de extensiones, debemos buscar en **php.ini** la directiva **extension_dir**.



Tenemos el archivo de extensión **php_extension.dll**

Al abrir **php.ini** encontramos la siguiente línea

```
extension_dir="C:\xampp\php\ext"
```

Si colocamos un archivo el archivo en **C:\xampp\php\ext\php_extension.dll** y escribimos en **php.ini** la línea:

```
extension=php_extension.dll
```

Al **reiniciar** nuestro servidor Apache, la extensión está activada.

Instalación de extensiones en Windows



Importante: Limitaciones de Windows

Si el sistema operativo en el que estamos intentando instalar nuestras extensiones es Windows, **tendremos un gran handicap con respecto a un servidor Linux**.

Por lo general, **PHP está especialmente pensado para funcionar con servidores Linux**. Esto es lógico ya que Linux es mucho más eficiente a la hora de actuar como servidor que Windows y es un sistema open source como lo es PHP.

Esto hace que, el desarrollo de extensiones para Linux sea prioritario en todos los casos y Windows sea secundario. En muchos casos veremos, que la extensión está únicamente disponible para Linux.

Si en tus desarrollos, comienzas a necesitar extensiones, es posible que haya llegado el momento de plantearte utilizar un servidor Linux.

Los archivos de extensiones de Windows, tienen **extensión DLL**.

Lo más habitual es utilizar la librería de extensiones de la comunidad **PECL** para localizar y descargar dicho archivo.

Veremos un listado de versiones de la extensión. Si la extensión está disponible para Windows, veremos que a su derecha aparece el **logotipo de Windows** con la palabra **DLL**.

The screenshot shows the PECL website interface. The top navigation bar includes links for Home, News, Documentation, Support, Downloads, Browse Packages, Search Packages, and Download Statistics. The main content area displays the package information for 'mongodb'. The package information table includes Summary, Maintainers, License, Description, and Homepage. Below this, there are links for Latest Tarball, Changelog, View Statistics, Browse Source, Package Bugs, and View Documentation. The 'Available Releases' table lists various versions of the MongoDB driver, including stable and alpha releases, with columns for Version, State, Release Date, Downloads, and a link to the Changelog.

Package Information					
Summary	MongoDB driver for PHP				
Maintainers	Derick Rethans <derick@php.net> (lead) [wishlist] [details] Jeremy Mikola <jmikola@gmail.com> (lead) [details] Hannes Magnusson <bjori@php.net> (lead) [details]				
License	Apache License				
Description	The purpose of this driver is to provide exceptionally thin glue between MongoDB and PHP, implementing only fundamental and performance-critical components necessary to build a fully-functional MongoDB driver.				
Homepage	http://docs.mongodb.org/ecosystem/drivers/php/				

Available Releases					
Version	State	Release Date	Downloads		
1.2.9	stable	2017-05-04	mongodb-1.2.9.tgz (767.6kB)	DLL	[Changelog]
1.2.8	stable	2017-03-20	mongodb-1.2.8.tgz (692.0kB)	DLL	[Changelog]
1.2.7	stable	2017-03-15	mongodb-1.2.7.tgz (766.8kB)	DLL	[Changelog]
1.2.6	stable	2017-03-07	mongodb-1.2.6.tgz (767.1kB)	DLL	[Changelog]
1.2.5	stable	2017-01-31	mongodb-1.2.5.tgz (765.8kB)	DLL	[Changelog]
1.2.4	stable	2017-01-30	mongodb-1.2.4.tgz (765.9kB)	DLL	[Changelog]
1.2.3	stable	2017-01-17	mongodb-1.2.3.tgz (691.7kB)	DLL	[Changelog]
1.2.2	stable	2016-12-13	mongodb-1.2.2.tgz (836.2kB)	DLL	[Changelog]
1.2.1	stable	2016-12-07	mongodb-1.2.1.tgz (686.8kB)	DLL	[Changelog]
1.1.10	stable	2016-12-07	mongodb-1.1.10.tgz (646.8kB)	DLL	[Changelog]
1.2.0	stable	2016-11-29	mongodb-1.2.0.tgz (689.5kB)	DLL	[Changelog]
1.1.9	stable	2016-10-20	mongodb-1.1.9.tgz (787.6kB)	DLL	[Changelog]
1.2.0alpha3	alpha	2016-09-21	mongodb-1.2.0alpha3.tgz (821.7kB)	DLL	[Changelog]
1.2.0alpha2	alpha	2016-09-14	mongodb-1.2.0alpha2.tgz (810.7kB)	DLL	[Changelog]

Al pinchar sobre **DLL** podremos elegir entre las diferentes DLL para esta versión de la extensión.

Tendremos diferentes DLL según estos criterios:

Versión de PHP

Debemos seleccionar la versión de PHP que utiliza nuestro servidor.

Tipo de procesador

Debemos seleccionar la DLL adecuada para el tipo de procesador del servidor.

Los tipos de procesador pueden ser de **32 bits** (indicados como x86) o **64 bits** (indicados como x64).

Thread safe

Todas las DLL pueden ser **thread safe** (TS) o **non thread safe** (NTS).

Se dice que una DLL es thread safe cuando funciona correctamente durante la ejecución simultánea de múltiples hilos.

Por lo general, preferiremos que la DLL sea **thread safe**.

The screenshot shows the PECL website interface. The main content area displays the 'mongodb 1.2.9' package page. On the left, there are navigation links for Home, News, Documentation, Downloads, and Search Packages. The main content area includes a search bar, a breadcrumb trail (Top Level :: Database :: mongodb :: Windows), and the package title 'mongodb 1.2.9'. Below this, there is a 'Package Information' table with fields for Summary, Maintainers, License, Description, Homepage, and Release notes. The 'Release notes' section lists several bugs (PHPC-940, PHPC-948, PHPC-949). Below the package information, there is a 'DLL List' table showing the available DLLs for different PHP versions (7.1, 7.0, 5.6, 5.5) and their architectures (x86, x64) and thread safety (NTS, TS).

Package Information	
Summary	MongoDB driver for PHP
Maintainers	Derick Rethans <derick@php.net> (lead) [wishlist] [details] Jeremy Mikola <jmikola@gmail.com> (lead) [details] Hannes Magnusson <bjori@php.net> (lead) [details]
License	Apache License
Description	The purpose of this driver is to provide exceptionally thin glue between MongoDB and PHP, implementing only fundamental and performance-critical components necessary to build a fully-functional MongoDB driver.
Homepage	http://docs.mongodb.org/ecosystem/drivers/php/
Release notes	Version 1.2.9 (stable) ** Bug * [PHPC-940] - php_phongo_free_ssl_opt() attempts to free interned strings * [PHPC-948] - BSON encoding should throw on circular references * [PHPC-949] - Memory leak if Serializable::bsonSerialize() returns keys with null bytes

DLL List	
PHP 7.1	7.1 Non Thread Safe (NTS) x86 7.1 Thread Safe (TS) x86 7.1 Non Thread Safe (NTS) x64 7.1 Thread Safe (TS) x64
PHP 7.0	7.0 Non Thread Safe (NTS) x86 7.0 Thread Safe (TS) x86 7.0 Non Thread Safe (NTS) x64 7.0 Thread Safe (TS) x64
PHP 5.6	5.6 Non Thread Safe (NTS) x86 5.6 Thread Safe (TS) x86 5.6 Non Thread Safe (NTS) x64 5.6 Thread Safe (TS) x64
PHP 5.5	5.5 Non Thread Safe (NTS) x86 5.5 Thread Safe (TS) x86 5.5 Non Thread Safe (NTS) x64 5.5 Thread Safe (TS) x64

At the bottom of the page, there are links for [Latest Tarball], [Changelog], and [View Statistics].

Una vez descargada la DLL correspondiente, la colocaremos en la **carpeta de extensiones** y la incluiremos en **php.ini** para activarla.



Importante: Dependencias

Si tras seguir estos pasos, la extensión no se activa, puede ser que la DLL necesite otras DLL, programas o recursos para funcionar correctamente.

En tal caso debemos informarnos sobre qué requisitos necesita la extensión para funcionar y aplicarlos.

Es muy posible que encuentres los requisitos necesarios en la documentación de la extensión en <http://www.php.net/>



Instalando la extension MongoDB en Windows

Instalación de extensiones en Linux

La instalación de extensiones es mucho más sencilla en servidores Linux ya que tenemos un comando específico para su instalación.

El comando **pecl** se encarga de instalar las extensiones contenidas en librería de la comunidad PHP. Para que este comando funcione correctamente necesitamos únicamente tres requisitos:

1

Tener **PHP** instalado en el servidor.

2

Tener **PEAR** instalado en el servidor.

PEAR es un sistema de **distribución de componentes de PHP** en el que se basa el comando **pecl**.

Si no lo tienes instalado, puedes hacerlo con los comandos habituales de instalación de paquetes de tu distribución Linux.

3

Tener instalado un **compilador de C**.

El compilador de código más habitual para Linux es **gcc**.

Si no lo tienes instalado, puedes hacerlo con los comandos habituales de instalación de paquetes de tu distribución Linux.

El uso de este comando es tremendamente sencillo. Para realizar una instalación, sólo debemos ejecutar el siguiente comando:

`pecl install nombre_extension`



Importante: Dependencias

Es posible que, durante la instalación, se produzca algún tipo de error.

Esto suele ser debido a que, nuestra extensión, **necesita paquetes o recursos adicionales** que no están instalados en el servidor.

El comando **pecl** nos avisará de qué paquete falta y simplemente tendremos que instalarlo mediante los comandos habituales de instalación de paquetes de nuestra distribución Linux.



Instalando la extension MongoDB en Linux

Librerías

Aparte de las extensiones, tenemos otra forma de añadir funcionalidades adicionales a nuestros desarrollos en PHP: Las **librerías**.

Una librería, es un **archivo php** con el código necesario para realizar cierto proceso.

Con el comando **include**, podemos añadir el código contenido en dicho archivo a nuestro proyecto y utilizarlo.

Vamos a ver un ejemplo:

Descargamos de internet una librería que realiza cierto proceso. La descarga es un archivo llamado libreria.php con el siguiente contenido:

```
<?php

class nombre_libreria {
    // Código que realiza el proceso deseado
}

?>
```


En el código de nuestro proyecto añadiremos la librería de la siguiente forma:

```
<?php

include('libreria.php');
$proceso = new nombre_libreria();
// Ya podemos utilizar todos los métodos de la librería

?>
```



Anotación: Utilización de clases en librerías

Aunque no es estrictamente necesario, en la gran mayoría de las librerías que descarguemos, veremos que el código contenido **es una o varias clases**.

Esto suele ser así debido a que la mejor forma de distribuir un fragmento de código que realiza un proceso concreto independiente es **encapsularlo en un objeto**.

También es la mejor forma de **no mezclar funciones o variables** que puedan llamarse de forma idéntica en nuestro proyecto, evitando así incompatibilidades.

Como vemos, esta es una forma de distribuir código **mucho más simple** que las extensiones.

Ventajas

Utilizar librerías tiene algunas **ventajas** frente al uso de extensiones:

Portabilidad

Como el código de la librería **va anexado a nuestro proyecto**, no tenemos que preocuparnos de si nuestro servidor tiene instalado o no cierto componente.

En muchas ocasiones, tendremos que subir nuestros desarrollos a servidores de terceros, en los cuales **no somos libres de instalar todo lo que queramos**.

Utilizar librerías nos asegura que nuestro código funcionará correctamente con independencia de los módulos instalados en el servidor.

Facilidad de uso

Integrar una librería en nuestro proyecto, es **mucho más sencillo** que añadir una extensión, si esta no está instalada.

Instalar extensiones **puede requerir ciertos conocimientos de administración del sistema operativo** del servidor que es posible que todos los programadores no posean.

No obstante, cualquier programador será capaz de utilizar una librería.

Elemento

Crear una librería **está al alcance de cualquier programador**.

No es necesario tener ningún conocimiento adicional para crearla, ya que utilizaremos la misma sintaxis que usaríamos para un desarrollo propio.

Podemos subir nuestra librería a cualquier web para promocionarla y distribuirla por nuestra cuenta.

Diversidad

Gracias a esta **facilidad de creación**, y a la **amplia comunidad de desarrolladores** en PHP, encontraremos una cantidad enorme de librerías.

En comparación con las extensiones, la cantidad de de librerías nos asegura que encontremos el proceso que necesitemos, casi con seguridad.

Cualquier problema que necesitemos resolver, sin duda ya se lo han encontrado otros programadores con anterioridad y, en muchos casos, habrán desarrollado librerías para solventarlo.

Desventajas

Pero también tiene algunas **desventajas**:

Fallos

Debido a que cualquier programador puede crear y publicar una librería sin mucho esfuerzo, puede que la calidad del código no sea la mejor.

Si la fuente de la que hemos obtenido la librería no es muy confiable, podríamos encontrarnos con **código ineficiente** o incluso con **errores de ejecución** en el código.

Por lo general, antes de utilizar una librería, debemos comprobar si la fuente de donde la obtenemos es de confianza y si hay otros programadores utilizándola satisfactoriamente.

Mantenimiento

Si utilizamos una librería de la que nuestro código depende, podríamos encontrarnos con la desagradable sorpresa de que **el autor de la misma ha dejado de mantenerla**

Es posible que al intentar avanzar de versión en PHP o actualizar ciertas características, nos encontremos con que la librería se encuentra **obsoleta y no hay actualizaciones**.

No hay una forma sencilla de prever esta situación, pero intentar librerías de fuentes que nos den cierta seguridad, puede ahorrarnos más de un problema en el futuro.

Estandarización

Existe una cantidad enorme de librerías. Esto, que a priori es una ventaja, puede producir una situación en la que existen **múltiples librerías distintas tratando de resolver el mismo problema**

Cuando utilizamos extensiones, sobre todo si son oficiales, nuestro forma de resolver el problema es casi estándar. Pero al usar librerías, podemos encontrarnos con que en unos desarrollos se ha resuelto el problema con una librería y en otros con otra.

Es importante verificar que la librería que hemos elegido es la mejor alternativa posible para resolver nuestras necesidades y que **no existe otra mejor y más utilizada** para ello.



Truco: Utilizar librerías de forma segura

Si vas a utilizar una librería, comprueba siempre si **la fuente de donde la obtienes es fiable** y si es la **mejor alternativa** para cubrir tu necesidad.

Ejemplo práctico: Librería PHPMailer

Vamos a ver un ejemplo práctico de utilización de librerías de terceros: **PHPMailer**.

PHPMailer es una de las librerías más famosas de PHP. Su misión es realizar **envíos de correos electrónicos** mediante el protocolo SMTP.



Esta librería está **implantada en miles de aplicaciones web** y es utilizada por proyectos desarrollados en PHP tan relevantes como **WordPress, Drupal o Joomla**. Esto nos indica que es una librería muy robusta y estable y no tendremos problemas con ella.

Descarga de PHPMailer

Si buscas PHPMailer en cualquier buscador de Internet, obtendrás como primer resultado la página enGitHub de la librería PHPMailer.



Enlaces: Página en GitHub de PHPMailer

<https://github.com/PHPMailer/PHPMailer>

Desde su página de GitHub puedes descargar la última versión de PHPMailer.

Si lo prefieres, puedes descargar la misma librería guardada en este curso:



[Librería PHPMailer](#)

Añadir PHPMailer a nuestro proyecto

Integrar PHPMailer con nuestro proyecto PHP es muy sencillo, sólo debemos seguir los siguientes pasos:

1. **Descomprimir** el archivo ZIP descargado.
2. Copiar todos los archivos descargados a una **carpeta dentro de la carpeta pública** de nuestro servidor web.
3. Incluir en nuestro código mediante la instrucción **include** el archivo **PHPMailerAutoload.php**.
4. **Crear un objeto** de la clase PHPMailer.

Vamos a ver un ejemplo de cómo quedaría nuestro código después de realizar estos pasos:

```
<?php
```

```
include('phpmailer/PHPMailerAutoload.php');  
$mail = new PHPMailer();
```

```
?>
```

Enviar un e-mail con PHPMailer

Una vez generado el objeto PHPMailer, el uso de la librería es muy sencillo. Encontrarás un ejemplo muy orientativo en su página de GitHub.

Solo tenemos que establecer una serie de propiedades del objeto y llamar a una serie de métodos para configurar **el servidor SMTP** y el **correo electrónico a enviar**. Vamos a verlos:



Proceso: Configurar el servidor SMTP

Para configurar el servidor web ejecutaremos la siguientes instrucciones:

```
$mail -> isSMTP(); // Indica que vamos a utilizar el protocolo SMTP
$mail -> Host = 'smtp.gmail.com'; // Especificamos el servidor SMTP
$mail -> SMTPAuth = true; // Indicamos que nuestro servidor requiere autenticación
$mail -> Username = 'miusuario'; // Nombre de usuario
$mail -> Password = 'micontraseña'; // Contraseña
$mail -> SMTPSecure = 'tls'; // Tipo de seguridad de la conexión
$mail -> Port = 587; // Puerto de envío
$mail -> setFrom('correo@gmail.com','José'); // Dirección de correo y nombre
```

Estos datos deben ser suministrados por nuestro proveedor de correo.



Proceso: Configurar el correo electrónico a enviar

Para configurar el correo electrónico a enviar:

```
$mail -> isHTML(true); // Especifica si el cuerpo del mensaje está en HTML
$mail -> CharSet = 'UTF-8'; // Indica la codificación de caracteres
$mail -> Subject = 'Asunto'; // Asunto del correo
$mail -> Body = 'Mensaje'; // Texto del correo
$mail -> addAddress('destinatario@gmail.com', 'Pedro'); // Añadir un destinatario
```



Anotación: otros métodos para configurar el correo electrónico

Opcionalmente, podemos utilizar también los siguientes métodos:

```
$mail -> addReplyTo('info@gmail.com', 'Info'); // Dirección de respuesta
$mail->addCC('maria@gmail.com','María'); // Añadir destinatario de copia
$mail->addBCC('rosa@gmail.com','Rosa'); // Añadir destinatario de copia oculta
$mail->addAttachment('C:\imagen.jpg', 'foto.jpg'); // Añadir archivo adjunto
```

Por último enviaremos el correo llamando al método `send()`, que devolverá **true** o **false**, según se haya conseguido o no enviar el correo.

**Ejemplo: Código completo de envío con PHPMailer**

```
<?php
    include('phpmailer/PHPMailerAutoload.php');
    $mail = new PHPMailer();
    $mail -> isSMTP();
    $mail -> Host = 'smtp.gmail.com';
    $mail -> SMTPAuth = true;
    $mail -> Username = 'cursophpdr@gmail.com';
    $mail -> Password = '123456789';
    $mail -> SMTPSecure = 'tls';
    $mail -> Port = 587;
    $mail -> setFrom('cursophpdr@gmail.com', 'José');
    $mail -> isHTML(true);
    $mail -> CharSet = 'UTF-8';
    $mail -> Subject = 'Notificación';
    $mail -> Body = 'Tienes una <b>Alerta</b>';
    $mail -> addAddress('domingo@gmail.com', 'Domingo');
    if (!$mail -> send()) {
        echo 'Error en el envío: ' . $mail -> ErrorInfo;
    } else {
        echo 'Correo enviado';
    }
?>
```



Enviar correos electrónicos con PHPMailer

Hemos aprendido



En esta unidad hemos aprendido:

- Las extensiones son **módulos que se integran dentro del propio lenguaje de programación** que necesitan ser **instalados y activados**.
 - A la hora de utilizar una extensión, puede encontrarse:
 - Instalada y activada.
 - Instalada y desactivada.
 - No instalada
 - Los principales lugares en los que buscaremos extensiones son:
 - php.net
 - PECL
 - Buscadores de internet
 - Para activar una extensión, debemos asegurarnos de que **el archivo de extensión se encuentra en la carpeta de extensiones** de PHP y añadir la línea **extension=nombre_archivo** correspondiente en nuestro **php.ini**.
 - Para instalar extensiones en Windows, descargaremos la **DLL** y la ubicaremos en la **carpeta de extensiones**.
 - Para instalar extensiones en Linux, utilizaremos el comando **pecl**.
- Las librerías son **archivos con código PHP** que anexamos a nuestros proyectos mediante un **include**.
 - Antes de utilizar una librería, debemos asegurarnos de que **la fuente de donde la obtenemos es fiable** y de que **no existe una alternativa mejor y más estándar**.

fundacionunirioja.adrformacion.com © ADR Infor SL
Héctor García González

fundacionunirioja.adrformacion.com © ADR Infor SL
Héctor García González

Ejercicios

Ejercicio 1: Instala la extensión para MongoDB

Duración estimada del ejercicio



15
minutos



Para completar este ejercicio correctamente, deberás instalar la extensión para MongoDB en tu servidor.

Pasos a seguir

Si tu servidor es Windows

1. Busca la extensión **mongodb** en <https://pecl.php.net/>
2. Pulsa sobre **DLL** en la última versión de la misma.
3. Descarga el archivo adecuado para tu **versión de PHP** y tu **procesador**.
4. Copia el archivo en la carpeta de extensiones (Puedes consultar cual es tu carpeta de extensiones en la directiva **extension_dir** de **php.ini**)
5. Añade la línea **extension=nombre_archivo.dll** en tu **php.ini**.
6. Reinicia el servidor web.
7. Comprueba con `phpinfo()` que la extensión **mongodb** está operativa.

Si tu servidor es Linux

1. Utiliza el comando **pecl install mongodb** para instalar la extensión.
2. Instala todas las dependencias que el comando te vaya indicando.
3. Añade la línea **extension=nombre_archivo.dll** en tu **php.ini**.
4. Reinicia el servidor web.
5. Comprueba con `phpinfo()` que la extensión **mongodb** está operativa.

Ejercicio 2: Envía un correo con PHPMailer

Duración estimada del ejercicio



30
minutos



Para completar este ejercicio, deberás enviar un correo electrónico con PHPMailer.

Lo necesario para comenzar

Descarga la librería PHPMailer de su página en GitHub o desde el enlace del propio curso:



Enlaces: Página de GitHub de PHPMailer

<https://github.com/PHPMailer/PHPMailer>



PHPMailer-master.zip

Librería PHPMailer

Pasos a seguir

1. Obtén la información necesaria para **configurar el servidor SMTP** de tu correo electrónico. Contacta con tu proveedor si no estás seguro. Necesitarás:
 1. Servidor SMTP
 2. Usuario
 3. Contraseña
 4. Dirección de correo
 5. Puerto de envío
 6. Tipo de seguridad
2. Añade la librería PHPMailer a tu código y crea el objeto.
3. Configura los datos de tu servidor SMTP en el objeto.
4. Configura un correo de prueba
5. Envía el correo con el método **send**.
6. Comprueba que el correo ha llegado correctamente al destinatario.

Recursos

Enlaces de Interés



<http://www.php.net/>
<http://www.php.net/>

Página web oficial de PHP



<https://pecl.php.net/>
<https://pecl.php.net/>

PHP Extension Community Library (PECL)



<https://github.com/PHPMailer/PHPMailer>
<https://github.com/PHPMailer/PHPMailer>

Página en GitHub de PHPMailer.

Preguntas Frecuentes

1. He añadido la línea `extension=nombre_extension` en mi `php.ini` y el archivo correspondiente a la carpeta de extensiones, pero, tras resetear el servidor, la extensión no funciona ¿Que sucede?

Seguramente tu extensión necesite algún recurso o programa que no está instalado en tu servidor. Consulta la documentación de la extensión para conocer los requisitos de la misma.

2. ¿Qué es mejor? ¿Una extensión o una librería?

Ninguna es mejor que otra, todo depende de la situación.

Las extensiones son **más estables y estándar**, pero las librerías tienen **mayor portabilidad** y te facilitarán las migraciones de servidor.

Glosario.

- **Bugs:** Errores de software
- **Extensiones:** Las extensiones son módulos que se integran dentro del propio lenguaje de programación.
- **GitHub:** Es una plataforma de desarrollo colaborativo para alojar proyectos utilizando el sistema de control de versiones Git.
- **Librerías:** Las librerías son archivos con código PHP que anexamos a nuestros proyectos mediante un `include`.
- **MongoDB:** Sistema de base de datos NoSQL de código abierto.

Extensiones y librerías

- **Open source:** Código abierto. Hace referencia al software desarrollado y distribuido libremente.
- **PECL:** PHP Extension Community Library. Repositorio de extensiones de la comunidad de programadores de PHP.
- **SMTP:** Simple Mail Transfer Protocol. Es el protocolo habitual para el envío de correos electrónicos.
- **SOAP:** Simple Object Access Protocol. Es un protocolo estándar utilizado en servicios web que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.