COmanage Installation - PHP

PHP Versions

The minimum version of PHP varies according to the product and version, though in general use the latest version of PHP where possible.

⚠ It is strongly recommended to use a version of PHP that is not EOL (currently **7.4** or later).

COmanage Product	PHP Version	Notes
COmanage Directory	**PHP 5.2.8** through **5.6.x**	
COmanage Match v1.1.0 and later	**PHP 8.0.2** or later	
COmanage Match v1.0.0	**PHP 7.4** or later	• CakePHP requires PHP 7.2, but its dependencies effectively require PHP 7.4 • Doctrine DBAL requires PHP 7.1 • Otherwise Match code runs under PHP 7.0.x
COmanage Registry v4.1.0 or later (for v4.x)	**PHP 7.1** or later	
COmanage Registry v2.0.0 through v4.0.2	**PHP 5.4** or later	• Versions of PHP 7 prior to 7.0.10 may cause segmentation faults under Apache • PHP 7.1 and 7.2 are supported starting with COmanage Registry v3.1.0 • PHP 8 is supported starting with COmanage Registry v4.0.2 • Installations with more than a thousand or so CO Person records should consider using PHP 7 or later, which features improved memory utilization • The LDAP Source Plugin requires PHP 5.6 or later • The Password Authenticator Plugin requires PHP 5.5 or later
COmanage Registry v1.0.x and earlier	**PHP 5.2.8** through **5.6.x**	• ⚠ The GitHub Provisioning Plugin requires PHP 5.4. It may be necessary to delete this plugin to use an earlier PHP version.
COmanage Registry PE v5.0.0 MR1 or later	**PHP 8.0.2** or later	• See Registry Minimum Versions and Requirements for additional information

> ⊘ **PHP 5.4.0 Enables Strict Error Reporting**
>
> As of PHP 5.4.0, PHP ships by default with strict logging enabled. As of CakePHP 2.0.5, this will cause failures during setup. This can be disabled in `php.ini` by setting
>
> ```
> error_reporting = E_ALL & ~E_STRICT
> ```

PHP Build Options

Various PHP build options may be required, depending on which products and features are required.

PHP Functionality	Required?	Build Option	Debian Package 1	RHEL Package 2

An appropriate database option	Match: Postgres only Registry: MySQL or Postgres Registry PE: MySQL or Postgres	eg `--with-pgsql --with-pdo-pgsql` or `--with-mysql --with-pdo-mysql`[3]	php*-pgsql or php*-mysql	php-pgsql or php-mysql
GD2	Registry: To make use of QR codes exposed for enrollment flow starting points	`--with-gd`	php*-gd	php-gd
LDAP	Directory: Required Registry: In order to use any LDAP based plugin (including the LDAP Provisioning Plugin, LDAP Identifier Validator Plugin, or LDAP Source Plugin)	`--with-ldap`	php*-ldap	php-ldap
OpenSSL	Match: Required Registry: Required	`--with-openssl`	*Enabled by default*	php-openssl
mbstring	Registry: For multi-byte string support in data normalization Registry PE: Required	`--enable-mbstring`	php*-mbstring	php-mbstring
XSL	Registry: Required	`--with-xsl`	php*-xsl	php-xsl
Internationalization Functions	Match: Required Registry PE: Required	`--enable-intl`	php*-intl	php-intl
Process Control	Registry PE: Required	`--enable-pcntl`	*Enabled by default*	php-pcntl
SimpleXML	Registry PE: Required	*Enabled by default*	php-xml	php-xml

Notes

[1]Debian packages use the PHP version number (eg: 5 or 7.0) in place of the *.

[2]Not all packages are available in the default repos.

[3]Both regular and PDO are required, with the former used by ADOdB and the latter used by CakePHP.

> ⊘ **PCRE Bug May Cause Problems**
>
> There are known issues with earlier versions of the PCRE library that will cause COmanage Registry to be unable to set up its database tables. Version 6.6 and earlier are known to have problems, while versions 8.02 and later are known to work. You can check the version that PHP was built against by running this command:
>
> ```
> php -r 'phpinfo();' | grep PCRE
> ```
>
> If you are using an old version of PCRE, you'll first need to install a more recent version. Be sure to configure it with the `--enable-utf8 --enable-unicode-properties` flags. You'll then need to rebuild PHP against the newer version of the PCRE library.
>
> Alternately, you may be able to rebuild PHP using its own internal copy of PCRE. As of PHP v5.3.3, PCRE 8.02 or later is included by default.

Testing PHP Database Connectivity

> ⊘ If PHP is not built correctly for database connectivity, subsequent steps may throw errors like
>
> Error: Database connection "Mysql" is missing, or could not be created.
>
> You can test for this with `php -r 'phpinfo();' | grep pdo`:
>
> ```
> $ php -r 'phpinfo();' | grep pdo
> pdo_mysql
> pdo_mysql.default_socket => /var/lib/mysql/mysql.sock => /var/lib/mysql/mysql.sock
> pdo_sqlite
> ```
>
> If the PDO libraries show correctly but the error still occurs, you may be trying to connect via the default socket (ie: "localhost") instead of TCP (ie: "127.0.0.1"), and the PDO libraries may not be able to find the default socket.

For the following tests to succeed, a database server must already be available with the example database (`test` in the example) and user (`registry_user` in the example) created.

To test if PHP was built with support for MySQL create a file `mysql-test.php` with contents similar to

```
<?php
mysql_connect("localhost", "registry_user", "a password goes here") or die(mysql_error());
echo "Connected to MySQL<br/>";
?>
```

To test if PHP was built with support for PostgreSQL create a file `postgreSQL-test.php` with contents

```
<?php
pg_connect("host=localhost dbname=test user=registry_user password=password") or die(pg_last_error());
echo "Connected to PostgreSQL<br/>";
?>
```

Either run the command line `php` tool on the file or serve it from your webserver and make sure that the script can connect to your database server.

> ⚠️ Be sure to remove your test file after testing so it is not exposed on your web server.

Configuration Suggestions

As for all production PHP installations, `display_errors` should be set to `Off` in your [php.ini](#).

Next Step

[COmanage Installation - Web Server](#)