

# **PHP\_CompatInfo User Guide**

**Laurent Laville**

---

# PHP\_CompatInfo User Guide

Laurent Laville

---

# Table of Contents

1. Introduction .....	1
2. Features .....	2
3. System Requirements .....	3
4. Installing PHP_CompatInfo .....	4
4.1. Using PEAR installer .....	4
5. Getting Started .....	5
5.1. Introduction .....	5
5.2. A simple tutorial .....	6
6. API basic usage .....	8
6.1. Default options .....	8
6.2. Scanning Files and Folders .....	8
6.3. Specifying a Reference .....	9
7. Using PHP_CompatInfo from the command line .....	11
7.1. Differences to other SAPI .....	11
7.2. Options .....	11
7.3. Commands .....	12
8. The XML Configuration File .....	25
8.1. Main options .....	25
8.2. Cache options .....	26
8.3. Setting PHP INI settings .....	27
8.4. Excluding Files or Elements from parsing .....	27
8.5. Scan Listeners .....	28
8.6. Plugins .....	29
A. Postprocessing the report file with XSLT .....	30
A.1. Main options .....	30
B. UML classes diagram .....	32
C. Copyright .....	33

---

## List of Figures

A.1. Example with sources file list collapsed .....	30
A.2. Example with sources file list expanded .....	31

---

# Chapter 1. Introduction

PHP\_CompatInfo is a PHP library that parse any data source (file/folder/mixed) to find out the minimum version and extensions required for it to run. A CLI tool is available, giving results without to have to code a line of programming.

The major version 2 is a full rewrites to PHP5, that used exceptions to raise errors, and an autoloader to dynamically load required classes.

If you want a PHP4 version compatible, you should consider to have a look on the branch 1.x solution hosted on PEAR repository :

- PEAR::PHP\_CompatInfo [[http://pear.php.net/package/PHP\\_CompatInfo](http://pear.php.net/package/PHP_CompatInfo)] 1.9.0



I recommend to migrate to PHP5, because I don't gave anymore support for PHP4 versions

This manual documents the final stable version 2.2.5

---

# Chapter 2. Features

- Parse a single file
- Parse a directory recursively or not
- Parse a list of files and/or directories
- Ability to give a list of extensions to ignore when calculating the version needed
- Ability to give a list of interfaces to ignore when calculating the version needed
- Ability to give a list of classes to ignore when calculating the version needed
- Ability to give a list of functions to ignore when calculating the version needed
- Ability to give a list of constants to ignore when calculating the version needed.
- Ability to give a list of files to ignore when calculating the version needed
- Ability to give a list of directories to ignore when calculating the version needed
- Ability to use a custom extensions list or by default only extensions loaded, to parse PHP code
- Event driven and listeners system to audit parsing process

---

# Chapter 3. System Requirements

## Required resources :

- PHP [<http://www.php.net>] 5.2.0 or newer
- Base [<http://ezcomponents.org/>] 1.8 or newer from *components.ez.no* PEAR channel
- ConsoleTools [<http://ezcomponents.org/>] 1.6.1 or newer from *components.ez.no* PEAR channel
- Console\_CommandLine [[http://pear.php.net/package/Console\\_CommandLine](http://pear.php.net/package/Console_CommandLine)] 1.1.3 or newer from default PEAR channel
- PHP\_Reflect [<http://bartlett.laurent-laville.org/>] 1.2.0 or newer from *Bartlett* PEAR channel
- tokenizer [<http://www.php.net/manual/en/book.tokenizer.php>] extension
- pcre [<http://www.php.net/manual/en/book.pcre.php>] extension
- SPL [<http://www.php.net/manual/en/book.spl.php>] extension
- DOM [<http://www.php.net/manual/en/book.dom.php>] extension
- libxml [<http://www.php.net/manual/en/book.libxml.php>] extension
- PHP\_Timer [<http://pear.phpunit.de>] 1.0.0 or newer from *PHPUnit* PEAR channel

## Optional resources :

- PEAR [<http://pear.php.net>] 1.9.0 or newer
- Net\_Growl [[http://pear.php.net/package/Net\\_Growl](http://pear.php.net/package/Net_Growl)] 2.4.0 or newer
- PHPUnit [<http://pear.phpunit.de>] 3.5.0 or newer from *PHPUnit* PEAR channel
- an XSLT processor if you want to produce an xHTML report from a phpci xml report

---

# Chapter 4. Installing PHP\_CompatInfo



The current version of PHP\_CompatInfo requires **PHP 5.2.0 or newer** to run. If you don't already have an up-to-date version of PHP installed it can be downloaded from the official PHP website <http://www.php.net/>.

## 4.1. Using PEAR installer

PHP\_CompatInfo should be installed using the [PEAR Installer](<http://pear.php.net/>). This installer is the backbone of PEAR, which provides a distribution system for PHP packages, and is shipped with every release of PHP since version 4.3.0.

The PEAR channel (*bartlett.laurent-laville.org*) that is used to distribute PHP\_CompatInfo needs to be registered with the local PEAR environment. Furthermore, components that PHP\_CompatInfo depends upon is hosted on the eZ Components PEAR channel (*components.ez.no*), and on the PHPUnit PEAR channel (*pear.phpunit.de*).

```
$ pear channel-discover bartlett.laurent-laville.org
Adding Channel "bartlett.laurent-laville.org" succeeded
Discovery of channel "bartlett.laurent-laville.org" succeeded

$ pear channel-discover components.ez.no
Adding Channel "components.ez.no" succeeded
Discovery of channel "components.ez.no" succeeded

$ pear channel-discover pear.phpunit.de
Adding Channel "pear.phpunit.de" succeeded
Discovery of channel "pear.phpunit.de" succeeded
```

This has to be done only once. Now the PEAR Installer can be used to install packages from the Bartlett channel.

```
$ pear install bartlett/PHP_CompatInfo
downloading PHP_CompatInfo-2.2.0.tgz ...
Starting to download PHP_CompatInfo-2.2.0.tgz (754,629 bytes)
.....done: 754,629 bytes
install ok: channel://bartlett.laurent-laville.org/PHP_CompatInfo-2.2.0
```

After the installation you can find the PHP\_CompatInfo source files inside your local PEAR directory.



---

# Chapter 5. Getting Started

## 5.1. Introduction

PHP\_CompatInfo branch 1.x is still compatible with PHP4, that is from an old age now (unmaintained). You can find it on PEAR main site on its project page [[http://pear.php.net/package/PHP\\_CompatInfo](http://pear.php.net/package/PHP_CompatInfo)].

PHP\_CompatInfo branch 2.x is a full rewrite with PHP5. Since RC3 the PHP parser engine used is PHP\_Reflect [<https://github.com/llaville/php-reflect>]. It's an improved version with "callbacks to what ever token you want" feature from basic concept version PHP\_TokenStream [<https://github.com/sebastianbergmann/php-token-stream>].

### 5.1.1. What can PHP\_CompatInfo do for you ?

Depending of server API you will use, PHP\_CompatInfo main goal is to give you the minimum and maximum PHP versions, a script or a list of scripts (sources), are required to run code.

But PHP\_CompatInfo (alias `phpci`), may also provides on CLI (with the `phpci` command) :

- Reference informations (MIN and MAX PHP versions) about one or more extensions :
  - list all classes
  - list all interfaces
  - list all functions
  - list all constants
- Print multiple reports group by scanned files or components
  - **summary** : usage summary of each extension, interface, class, function, constant used by your sources code
  - **source** : lines code count summary with or without source code
  - **xml** : export result to a XML format easy to transform to xHTML or other format
  - **token** : language PHP5 special features such as try/catch/throw exception, etc
  - **extension** : usage, minimum php version and origin of each extension used by your sources code
  - **namespace** : usage, minimum php version and origin of each namespace used by your sources code
  - **trait** : usage, minimum php version and origin of each trait used by your sources code
  - **interface** : usage, minimum php version and origin of each interface used by your sources code
  - **class** : usage, minimum php version and origin of each class used by your sources code

- **function** : usage, minimum php version and origin of each function used by your sources code
- **constant** : usage, minimum php version and origin of each constant used by your sources code
- **global** : usage, minimum php version and origin of each global variable used by your sources code

## 5.2. A simple tutorial

### 5.2.1. Parse a single file with default options

To parse a file with the PHP5 known references of your extensions loaded, you have just to specify the file's location.

#### API example.

```
<?php
require_once 'Bartlett/PHP/CompatInfo.php';

$source = '/path/to/myFile.php';

try {
    $phpci = new PHP_CompatInfo();
    $phpci->parse($source);

    $allResultsAtOnce = $phpci->toArray();
} catch (PHP_CompatInfo_Exception $e) {
    die ('PHP_CompatInfo Exception : ' . $e->getMessage() . PHP_EOL);
}
```

**phpci tool example.** From your PEAR *bin\_dir* directory run this command

```
$ phpci --no-configuration print --reference PHP5 --report summary /path/to/myFile.php
```

It won't use the default XML configuration file `phpcompatinfo.xml` or `phpcompatinfo.xml.dist` If you are sure of XML configuration settings, remove the `--no-configuration` option.

Will print out a summary report like this one

```
PHP COMPAT INFO REPORT SUMMARY
-----
FILES                                EXTENSIONS  INTERFACES  CLASSES  FUNCTIONS  CONSTANTS
-----
BASE: /path/to
-----
DIR.:
myFile.php                            5            3           11        27          4
-----
A TOTAL OF
  5 EXTENSION(S) 3 INTERFACE(S) 11 CLASSE(S) 27 FUNCTION(S) 4 CONSTANT(S)
WERE FOUND IN 1 FILE(S)
WITH CONDITIONAL CODE LEVEL 32
REQUIRED PHP 5.1.3 (MIN)
```

```
Time: 2 seconds, Memory: 8.50Mb
```

## 5.2.2. Parse a directory with default options

If you wish to parse an entire directory, you can specify the directory location instead of a file.

### API example.

```
<?php
require_once 'Bartlett/PHP/CompatInfo.php';

$source = '/path/to/myFolder';

try {
    $phpci = new PHP_CompatInfo();
    $phpci->parse($source);

    $allResultsAtOnce = $phpci->toArray();
} catch (PHP_CompatInfo_Exception $e) {
    die ('PHP_CompatInfo Exception : ' . $e->getMessage() . PHP_EOL);
}
```

**phpci tool example.** From your PEAR *bin\_dir* directory and the default XML configuration file *phpcompatinfo.xml.dist* installed into PEAR *cfg\_dir*/PHP\_CompatInfo.

```
$ phpci print /path/to/myFolder
```

Will print out the summary report

```
PHP COMPAT INFO REPORT SUMMARY
-----
FILES                                EXTENSIONS INTERFACES CLASSES FUNCTIONS CONSTANTS
-----
BASE: /path/to/myFolder
-----
DIR.:
Cache.php                            3              0              2              4              3
CLI.php                              3              0              8             16             6
Configuration.php                    3              0              5             10             3
Exception.php                        1              0              2              0              0
Reference.php                         1              1              0              0              1
Report.php                            2              0              2             11             3
TokenParser.php                      2              0             14              9              3
TokenStream.php                      4              0              3             17             3
-----
A TOTAL OF
  5 EXTENSION(S) 1 INTERFACE(S) 31 CLASSE(S) 47 FUNCTION(S) 6 CONSTANT(S)
WERE FOUND IN 8 FILE(S)
WITH CONDITIONAL CODE LEVEL 32
REQUIRED PHP 5.1.2 (MIN)
-----
Time: 2 seconds, Memory: 12.00Mb
-----
```

---

# Chapter 6. API basic usage

## 6.1. Default options

Whether you use the CLI version with the *phpci* command, or directly API functions, `PHP_CompatInfo` has some default options you should learn if you want to understand results provided.

Depending on SAPI you will use, source of settings is different.

Default options printed below, may be changed by the `$options` parameter of `PHP_CompatInfo` class constructor.

Option	Default	Description
<code>recursive</code>	<code>false</code>	scan recursive subdirectories or just local files
<code>reference</code>	<code>PHP5</code>	data dictionary reference (all PHP4 and PHP5 informations)
<code>referencePlugins</code>	<code>PHP4[...], PHP5[...]</code>	adapters to connect to data dictionaries reference
<code>verbose</code>	<code>false</code>	output more information
<code>fileExtensions</code>	<code>[php, inc, phtml]</code>	list of file extensions to scan
<code>cacheDriver</code>	<code>file</code>	cache results to improve speed of next iteration
<code>cacheOptions</code>	<code>[save_path =&gt; /tmp]</code>	options specific to cache driver used
<code>listeners</code>	<code>[]</code>	none

## 6.2. Scanning Files and Folders

The simplest way of using `PHP_CompatInfo` is to provide the location of a file or folder for `PHP_CompatInfo` to scan. If a folder is provided, `PHP_CompatInfo` will scan all files it finds in that local folder.



If you want sub-folders scanned, use the `recursive` option.

**Example: do not use cache files, but parse directory recursively.**

```
<?php
require_once 'Bartlett/PHP/CompatInfo.php';

$source = '/path/to/myFolder';
$options = array(
    'cacheDriver' => 'null',
    'recursive'   => true
);
```

```

try {
    $phpci = new PHP_CompatInfo($options);
    $phpci->parse($source);

    $allResultsAtOnce = $phpci->toArray();
} catch (PHP_CompatInfo_Exception $e) {
    die ('PHP_CompatInfo Exception : ' . $e->getMessage() . PHP_EOL);
}

```

## 6.3. Specifying a Reference

PHP\_CompatInfo can have multiple references installed to allow a single installation to be used with multiple platform. When scanning PHP code, PHP\_CompatInfo can be told which reference to use. This is done using the `reference` option.

**Example: specify a PHP4 reference to parse only PHP 4 sources code.**

```

<?php
require_once 'Bartlett/PHP/CompatInfo.php';

$source = '/path/to/myFolder';
$options = array(
    'reference' => 'PHP4',
);

try {
    $phpci = new PHP_CompatInfo($options);
    $phpci->parse($source);

    $allResultsAtOnce = $phpci->toArray();
} catch (PHP_CompatInfo_Exception $e) {
    die ('PHP_CompatInfo Exception : ' . $e->getMessage() . PHP_EOL);
}

```



If you want to use your own reference, you should have (of course) to write it, but you must also tell where it is.

**Example: replaces default references provided in standard distribution.**

```

<?php
require_once 'Bartlett/PHP/CompatInfo.php';

$source = '/path/to/myFolder';
$options = array(
    'reference' => 'PHP5',
    'referencePlugins' => array(
        'PHP5' => array(
            'class' => 'myRefClass',
            'file' => '/path/to/file/hosting/myRefClass.php',
            'args' => array()
        ),
    ),
);

```

```
try {
    $phpci = new PHP_CompatInfo($options);
    $phpci->parse($source);

    $allResultsAtOnce = $phpci->toArray();
} catch (PHP_CompatInfo_Exception $e) {
    die ('PHP_CompatInfo Exception : ' . $e->getMessage() . PHP_EOL);
}
```

---

# Chapter 7. Using PHP\_CompatInfo from the command line

## 7.1. Differences to other SAPI

Remarkable differences of the CLI SAPI compared to other SAPI:

- gets responses in a seconde without code line to write
- printing components report
- printing references informations report
- configurable thru an easy to read XML file

## 7.2. Options

The list of command line options provided by the *phpci* command can be queried anytime by running *phpci* with the *-h* or *--help* switches.

```
PHPCompatInfo (cli) by Laurent Laville.
```

```
Usage:
```

```
phpci [options]
phpci [options] <command> [options] [args]
```

```
Options:
```

```
--configuration=xmlFile      Read configuration from XML file
--no-configuration           Ignore default configuration file
                              (phpcompatinfo.xml)
-d iniSet, --ini-set=iniSet  Sets a php.ini directive value
-v, --verbose                Output more verbose information
-h, --help                   show this help message and exit
--version                    show the program version and exit
```

```
Commands:
```

```
print                        Print a report of data source parsed.
list-references               List all extensions supported.
list                          List all "elements" referenced in the data base.
list-extensions               List all extensions referenced in the data base.
list-interfaces               List all interfaces referenced in the data base.
list-classes                  List all classes referenced in the data base.
list-functions                List all functions referenced in the data base.
list-constants                List all constants referenced in the data base.
```

Short Option	Long Option	Description
	--configuration	Specify a custom XML file (which does not need to be named phpcompatinfo.xml or located into PEAR cfg_dir)

Short Option	Long Option	Description
	--no-configuration	Ignore default configuration files <code>phpcompatinfo.xml</code> or <code>phpcompatinfo.xml.dist</code>
-d	--ini-set	This option allows you to set a custom value for any of the configuration directives allowed in <code>php.ini</code> . Example: <code>-d memory_limit=256M</code>
-v	--verbose	Sets the verbose level to print more informations
-h	--help	With this option, you can get information about the actual list of command line options and some one line descriptions about what they do.
	--version	Prints the version of PHP_CompatInfo and exits.

Unless you tell it to ignore the XML configuration file, default options are sets by the `phpcompatinfo.xml.dist` file if found into the PEAR `cfg_dir \PHP_CompatInfo` directory

```
<?xml version="1.0" encoding="utf-8" ?>
<phpcompatinfo
  reference="PHP5"
  report="summary"
  reportFileAppend="false"
  cacheDriver="file"
  recursive="false"
  fileExtensions="php, inc, phtml"
  consoleProgress="true"
  verbose="false"
  >
  <!-- ... -->
</phpcompatinfo>
```

## 7.3. Commands

They are two categories of command:

- printing scanned sources code results
- listing database references extensions informations

### 7.3.1. print command

This command allow to print results of scanned sources code. If you forget options, enter command below.

```
$ phpci print --help
```

That will show you this help screen:

```
Print a report of data source parsed.
```

```
Usage:
```



## Using PHP\_CompatInfo from the command line

```
phpci [options] print [options] <sourcePath>
```

### Options:

<code>--reference=reference</code>	The name of the reference to use
<code>--report=report</code>	Type of report
<code>--report-file=reportFile</code>	Write the report to the specified file path
<code>--exclude-pattern=excludeID</code>	Exclude components from list referenced by ID provided
<code>-R, --recursive</code>	Includes the contents of subdirectories
<code>--file-extensions=fileExtensions</code>	A comma separated list of file extensions to check
<code>--help-reference</code>	List of reference available
<code>--help-report</code>	List of report available
<code>-h, --help</code>	show this help message and exit

### Arguments:

`sourcePath` The data source to scan (file or directory).

Short Option	Long Option	Description
	<code>--reference</code>	This option tells what reference dictionary to use. May be either one providing by the package distribution or your own.
	<code>--report</code>	Specify what kind of report to print. See <code>--help-report</code> for full list.
	<code>--report-file</code>	This option identify the file where results will be written.
	<code>--exclude-pattern</code>	Identify a <code>&lt;exclude&gt;</code> element in the XML configuration file, that define what elements to exclude from scope.
<code>-R</code>	<code>--recursive</code>	This option allow to parse directories recursively.
	<code>--file-extensions</code>	When parsing directory, specify what file to scan.
	<code>--help-reference</code>	Remember you what are reference dictionary available in default distribution.
	<code>--help-report</code>	Remember you what are reports available in default distribution.
<code>-h</code>	<code>--help</code>	With this option, you can get information about the actual list of command line options.



You can give one or more report at the same time.

### Example: Printing only summary report.

```
$ phpci print --report summary /path/to/source
```

### Example: Printing both summary, extension, interface, class, function, and constant reports.

```
$ phpci print --report summary extension interface class function constant /path/to/source
```

### Example: Printing both function and constant reports.

```
$ phpci print --report function --report constant /path/to/source
```

## 7.3.2. list-references command

This command allow to print the list of documented extensions available with your version of PHP\_CompatInfo, and specify those which are currently loaded (**L** in front of their names)

If you forget options, enter command below.

```
$ phpci list-references --help
```

That will show you this help screen:

```
List all extensions supported.
```

```
Usage:
```

```
  phpci [options] list-references [options]
```

```
Options:
```

```
--report-file=reportFile  Write the report to the specified file path  
-h, --help                show this help message and exit
```

Short Option	Long Option	Description
	--report-file	This option identify the file where results will be written.
-h	--help	With this option, you can get information about the actual list of command line options.

### Example.

```
$ phpci list-references
```

```
-----  
PHP COMPAT INFO DATABASE REFERENCE  
-----  
EXTENSIONS                                EXTENSION ❶          VERSION ❷  
-----  
  apc                                     3.1.7                4.0.0  
L bcmath                                 4.0.0  
L bz2                                    4.0.4  
L calendar                               4.0.0  
L Core                                   4.0.0  
L ctype                                  4.0.4  
L curl                                   4.0.2  
L date                                   4.0.0  
L dom                                    20031129             5.0.0  
  enchant                                 1.1.0                5.3.0  
L ereg                                   4.0.0                5.3.0  
L fileinfo                               1.0.5-dev            4.0.0  
L filter                                 0.11.0               5.2.0  
L ftp                                    4.0.0  
L gd                                     4.0.0  
L gettext                                4.0.0  
  gmp                                    4.0.4  
L hash                                   1.0                  5.1.2  
L iconv                                  4.0.5  
L imap                                   4.0.0  
  intl                                   1.1.0                5.2.4
```

## Using PHP\_CompatInfo from the command line

L json	1.2.1	5.2.0
ldap		4.0.0
L libxml		5.0.0
L mbstring		4.0.6
L mcrypt		4.0.0
memcache	3.0.6	4.3.3
memcached	1.0.2	5.2.0
L mhash		4.0.0
L mysql	1.0	4.0.0
L mysqli	0.1	5.0.0
L OAuth	1.0-dev	5.1.0
L openssl		4.0.4
pcntl		4.1.0
L pcre		4.0.0
L PDO	1.0.4dev	5.1.0
pgsql		4.0.0
L Phar	2.0.1	5.2.0
posix	306939	4.0.0
readline	2.0.1	4.0.0
recode	2.0.1	4.0.0
reflection		5.0.0
L session		4.0.0
L shmop		4.0.0
L SimpleXML	0.1	5.0.0
snmp		4.0.0
L soap		5.0.0
L sockets		4.1.0
L SPL	0.2	5.0.0
L SQLite	2.0-dev	5.0.0
L sqlite3	0.7-dev	5.3.0
ssh2	0.11.0	5.0.0
L standard		4.0.0
sysvmsg	306939	4.3.0
sysvsem		4.0.0
sysvshm		4.0.0
L tidy	2.0	4.0.0
L tokenizer	0.1	4.2.0
L wddx		4.0.0
L xdebug	2.1.0	5.2.0
L xml		4.0.0
L xmlreader	0.1	5.0.0
L xmlrpc	0.51	4.1.0
L xmlwriter	0.1	5.1.2
L xsl	0.1	5.0.0
L zlib	1.1	4.0.0

-----

A TOTAL OF 66 EXTENSIONS WERE FOUND AND 48 LOADED

-----

Time: 0 seconds, Memory: 4.75Mb

-----

- ❶ This column specify the version of extension that was documented
- ❷ This column specify which are the php versions (minimum and maximum) supported by the extension

### 7.3.3. list command

This command allow to combine one or more list-\* command.



results may be huge

If you forget options, enter command below.

```
$ phpci list --help
```

That will show you this help screen:

```
List all "elements" referenced in the data base.

Usage:
  phpci [options] list [options] <element...>

Options:
  --reference=reference      The name of the reference to use
  --report-file=reportFile  Write the report to the specified file path
  --help-reference          List of reference available
  -h, --help                show this help message and exit

Arguments:
  element  May be either "extensions", "interfaces", "classes", "functions"
           or "constants"
```

Short Option	Long Option	Description
	--reference	This option tells what reference dictionary to use. May be either one providing by the package distribution or your own.
	--report-file	This option identify the file where results will be written.
	--help-reference	Remember you what are reference dictionary available in default distribution.
-h	--help	With this option, you can get information about the actual list of command line options.

**Example: list both interfaces and classes of all extensions supported by PHP\_Compatinfo.**

```
$ phpci list interfaces classes
```

```
-----
PHP COMPAT INFO INTERFACES REFERENCE
-----
```

INTERFACES	EXTENSION	VERSION
ArrayAccess	SPL	5.1.0
Countable	SPL	5.1.0
Iterator	SPL	5.1.0
IteratorAggregate	SPL	5.1.0
OuterIterator	SPL	5.1.0
RecursiveIterator	SPL	5.1.0
SeekableIterator	SPL	5.1.0
Serializable	SPL	5.1.0
SplObserver	SPL	5.1.0
SplSubject	SPL	5.1.0
Traversable	SPL	5.1.0

## Using PHP\_CompatInfo from the command line

```
-----  
A TOTAL OF 11 INTERFACES WERE FOUND  
-----
```

```
Time: 0 seconds, Memory: 6.25Mb  
-----
```

```
-----  
PHP COMPAT INFO CLASSES REFERENCE  
-----
```

CLASSES	EXTENSION	VERSION
AppendIterator	SPL	5.1.0
ArrayIterator	SPL	5.0.0
ArrayObject	SPL	5.0.0
BadFunctionCallException	SPL	5.1.0
BadMethodCallException	SPL	5.1.0
CachingIterator	SPL	5.0.0
Closure	Core	5.3.0
DOMAttr	dom	5.0.0

```
... <more results> ...
```

mysqli_warning	mysqli	5.0.0
php_user_filter	standard	5.0.0
stdClass	Core	4.0.0
tidy	tidy	4.0.0
tidyNode	tidy	5.0.1

```
-----  
A TOTAL OF 125 CLASSES WERE FOUND  
-----
```

```
Time: 0 seconds, Memory: 6.25Mb  
-----
```

### 7.3.4. list-extensions command

This command allow to print the list of documented extensions available with your version of PHP\_CompatInfo, and loaded on your platform.

If you forget options, enter command below.

```
$ phpci list-extensions --help
```

That will show you this help screen:

```
List all extensions referenced in the data base.
```

```
Usage:
```

```
  phpci [options] list-extensions [options] <extension>
```

```
Options:
```

```
--reference=reference      The name of the reference to use  
--report-file=reportFile  Write the report to the specified file path  
--help-reference          List of reference available  
-h, --help                show this help message and exit
```

```
Arguments:
```

```
  extension (optional) Limit output only to this extension
```

Using PHP\_CompatInfo  
from the command line

Short Option	Long Option	Description
	--reference	This option tells what reference dictionary to use. May be either one providing by the package distribution or your own.
	--report-file	This option identify the file where results will be written.
	--help-reference	Remember you what are reference dictionary available in default distribution.
-h	--help	With this option, you can get information about the actual list of command line options.

**Example.**

```
$ phpci list-extensions
```

```
-----
PHP COMPAT INFO EXTENSIONS REFERENCE
-----
```

EXTENSIONS	EXTENSION ❶	VERSION ❷
Core		4.0.0
PDO	1.0.4dev	5.1.0
Phar	2.0.1	5.2.0
SPL	0.2	5.0.0
SQLite	2.0-dev	5.0.0
SimpleXML	0.1	5.0.0
bcmath		4.0.0
bz2		4.0.4
calendar		4.0.0
ctype		4.0.4
curl		4.0.2
date		4.0.0
dom	20031129	5.0.0
ereg		4.0.0 5.3.0
fileinfo	1.0.5-dev	4.0.0
filter	0.11.0	5.2.0
ftp		4.0.0
gd		4.0.0
gettext		4.0.0
hash	1.0	5.1.2
iconv		4.0.5
imap		4.0.0
json	1.2.1	5.2.0
libxml		5.0.0
mbstring		4.0.6
mcrypt		4.0.0
mhash		4.0.0
mysql	1.0	4.0.0
mysqli	0.1	5.0.0
OAuth	1.0-dev	5.1.0
openssl		4.0.4
pcre		4.0.0
session		4.0.0
shmop		4.0.0
soap		5.0.0
sockets		4.1.0
sqlite3	0.7-dev	5.3.0

## Using PHP\_CompatInfo from the command line

```
standard                4.0.0
tidy                    2.0                4.0.0
tokenizer              0.1                4.2.0
wddx                   4.0.0
xdebug                 2.1.0              5.2.0
xml                    4.0.0
xmlreader              0.1                5.0.0
xmlrpc                 0.51               4.1.0
xmlwriter              0.1                5.1.2
xsl                    0.1                5.0.0
zlib                   1.1                4.0.0
```

-----  
A TOTAL OF 48 EXTENSIONS WERE FOUND  
-----

Time: 0 seconds, Memory: 6.25Mb  
-----

- ❶ This column specify the version of extension that was documented
- ❷ This column specify which are the php versions (minimum and maximum) supported by the extension



### You can filter result by extension

Example: if you want only SPL extension, enter command below.

```
$ phpci list-extensions SPL
```

## 7.3.5. list-interfaces command

This command allow to print the list of documented interfaces available with your version of PHP\_CompatInfo, from your extensions loaded (or defined into XML configuration file).

If you forget options, enter command below.

```
$ phpci list-interfaces --help
```

That will show you this help screen:

```
List all interfaces referenced in the data base.
```

Usage:

```
phpci [options] list-interfaces [options] <extension>
```

Options:

```
--reference=reference      The name of the reference to use
--report-file=reportFile  Write the report to the specified file path
--help-reference          List of reference available
-h, --help                show this help message and exit
```

Arguments:

```
extension (optional) Limit output only to this extension
```

Short Option	Long Option	Description
	--reference	This option tells what reference dictionary to use. May be either one providing by the package distribution or your own.

Short Option	Long Option	Description
	--report-file	This option identify the file where results will be written.
	--help-reference	Remember you what are reference dictionary available in default distribution.
-h	--help	With this option, you can get information about the actual list of command line options.

### Example.

```
$ phpci list-interfaces
```

```
-----
PHP COMPAT INFO INTERFACES REFERENCE
-----
INTERFACES                                     EXTENSION ❶   VERSION ❷
-----
ArrayAccess                                   SPL          5.1.0
Countable                                     SPL          5.1.0
Iterator                                       SPL          5.1.0
IteratorAggregate                             SPL          5.1.0
OuterIterator                                 SPL          5.1.0
RecursiveIterator                             SPL          5.1.0
SeekableIterator                             SPL          5.1.0
Serializable                                  SPL          5.1.0
SplObserver                                  SPL          5.1.0
SplSubject                                    SPL          5.1.0
Traversable                                   SPL          5.1.0
-----
A TOTAL OF 11 INTERFACES WERE FOUND
-----
Time: 0 seconds, Memory: 6.25Mb
-----
```

- ❶ This column specify the name of extension that provide these interfaces
- ❷ This column specify which are the php versions (minimum and maximum) supported by the interface



### You can filter result by extension

Example: if you want only SPL extension interfaces, enter command below.

```
$ phpci list-interfaces SPL
```

## 7.3.6. list-classes command

This command allow to print the list of documented classes available with your version of PHP\_CompatInfo, from your extensions loaded (or defined into XML configuration file).

If you forget options, enter command below.

```
$ phpci list-classes --help
```

That will show you this help screen:



## Using PHP\_CompatInfo from the command line

List all classes referenced in the data base.

### Usage:

```
phpci [options] list-classes [options] <extension>
```

### Options:

```
--reference=reference      The name of the reference to use
--report-file=reportFile  Write the report to the specified file path
--help-reference          List of reference available
-h, --help                show this help message and exit
```

### Arguments:

```
extension (optional) Limit output only to this extension
```

Short Option	Long Option	Description
	--reference	This option tells what reference dictionary to use. May be either one providing by the package distribution or your own.
	--report-file	This option identify the file where results will be written.
	--help-reference	Remember you what are reference dictionary available in default distribution.
-h	--help	With this option, you can get information about the actual list of command line options.

### Example.

```
$ phpci list-classes
```

```
-----
PHP COMPAT INFO CLASSES REFERENCE
-----
CLASSES                                     EXTENSION ❶      VERSION ❷
-----
AppendIterator                             SPL          5.1.0
ArrayIterator                              SPL          5.0.0
ArrayObject                                SPL          5.0.0
BadFunctionCallException                   SPL          5.1.0
... <more results> ...

mysqli_warning                             mysqli       5.0.0
php_user_filter                            standard    5.0.0
stdClass                                   Core        4.0.0
tidy                                        tidy        4.0.0
tidyNode                                   tidy        5.0.1
-----
A TOTAL OF 125 CLASSES WERE FOUND
-----
Time: 0 seconds, Memory: 6.25Mb
-----
```

❶ This column specify the name of extension that provide these classes

❷ This column specify which are the php versions (minimum and maximum) supported by the class



## You can filter result by extension

Example: if you want only SPL extension classes, enter command below.

```
$ phpci list-classes SPL
```

## 7.3.7. list-functions command

This command allow to print the list of documented functions available with your version of PHP\_CompatInfo, from your extensions loaded (or defined into XML configuration file).

If you forget options, enter command below.

```
$ phpci list-functions --help
```

That will show you this help screen:

```
List all functions referenced in the data base.
```

```
Usage:
```

```
phpci [options] list-functions [options] <extension>
```

```
Options:
```

```
--reference=reference      The name of the reference to use
--report-file=reportFile  Write the report to the specified file path
--help-reference          List of reference available
-h, --help                show this help message and exit
```

```
Arguments:
```

```
extension (optional) Limit output only to this extension
```

Short Option	Long Option	Description
	--reference	This option tells what reference dictionary to use. May be either one providing by the package distribution or your own.
	--report-file	This option identify the file where results will be written.
	--help-reference	Remember you what are reference dictionary available in default distribution.
-h	--help	With this option, you can get information about the actual list of command line options.

### Example.

```
$ phpci list-functions
```

```
-----
PHP COMPAT INFO FUNCTIONS REFERENCE
-----
FUNCTIONS                                EXTENSION ❶    VERSION ❷
-----
_                                           gettext      4.0.0
abs                                         standard     4.0.0
acos                                        standard     4.0.0
```

## Using PHP\_CompatInfo from the command line

```
acosh                                standard          4.0.7
... <more results> ...

zend_logo_guid                       standard         4.0.0
zend_thread_id                       Core            5.0.0
zend_version                         Core            4.0.0
zlib_get_coding_type                 zlib            4.3.2
-----
A TOTAL OF 1575 FUNCTIONS WERE FOUND
-----
Time: 1 second, Memory: 6.25Mb
-----
```

- ❶ This column specify the name of extension that provide these functions
- ❷ This column specify which are the php versions (minimum and maximum) supported by the function



### You can filter result by extension

Example: if you want only SPL extension functions, enter command below.

```
$ phpci list-functions SPL
```

## 7.3.8. list-constants command

This command allow to print the list of documented constants available with your version of PHP\_CompatInfo, from your extensions loaded (or defined into XML configuration file).

If you forget options, enter command below.

```
$ phpci list-constants --help
```

That will show you this help screen:

```
List all constants referenced in the data base.

Usage:
  phpci [options] list-constants [options] <extension>

Options:
  --reference=reference      The name of the reference to use
  --report-file=reportFile  Write the report to the specified file path
  --help-reference          List of reference available
  -h, --help                show this help message and exit

Arguments:
  extension (optional) Limit output only to this extension
```

Short Option	Long Option	Description
	--reference	This option tells what reference dictionary to use. May be either one providing by the package distribution or your own.
	--report-file	This option identify the file where results will be written.

## Using PHP\_CompatInfo from the command line

Short Option	Long Option	Description
	--help-reference	Remember you what are reference dictionary available in default distribution.
-h	--help	With this option, you can get information about the actual list of command line options.

### Example.

```
$ phpci list-constants
```

```
-----  
PHP COMPAT INFO CONSTANTS REFERENCE  
-----  
CONSTANTS                                EXTENSION ❶          VERSION ❷  
-----  
AF_INET                                   sockets      4.1.0  
AF_INET6                                   sockets      4.1.0  
AF_UNIX                                   sockets      4.1.0  
APACHE_MAP                                 soap        5.0.0  
ASSERT_ACTIVE                             standard     4.0.0  
  
... <more results> ...  
  
XSL_CLONE_NEVER                           xsl          5.0.0  
__CLASS__                                   Core         4.3.0  
__COMPILER_HALT_OFFSET__                   Core         5.1.0  
__DIR__                                     Core         5.3.0  
__FILE__                                    Core         4.0.0  
__FUNCTION__                               Core         4.3.0  
__LINE__                                    Core         4.0.0  
__METHOD__                                  Core         5.0.0  
__NAMESPACE__                              Core         5.3.0  
-----  
A TOTAL OF 1726 CONSTANTS WERE FOUND  
-----  
Time: 0 seconds, Memory: 6.25Mb  
-----
```

- ❶ This column specify the name of extension that provide these constants
- ❷ This column specify which are the php versions (minimum and maximum) supported by the constant



### You can filter result by extension

Example: if you want only SPL extension constants, enter command below.

```
$ phpci list-constants SPL
```

---

# Chapter 8. The XML Configuration File

## 8.1. Main options

The attributes of the `<phpcompatinfo>` element can be used to configure PHP\_CompatInfo's core functionality.

```
<?xml version="1.0" encoding="utf-8" ?>
<phpcompatinfo
  reference="PHP5"
  report="summary"
  reportFileAppend="false"
  cacheDriver="file"
  recursive="false"
  fileExtensions="php, inc, phtml"
  consoleProgress="true"
  verbose="false"
  >

  <!-- ... -->
</phpcompatinfo>
```

The XML configuration above corresponds to the default behaviour of the `phpcli` tool.

### reference

Data dictionary reference name. Defaults to PHP5 for all PHP4 and PHP5 components depending of your extensions loaded.

### report

Kind of report to produces. May be either *summary*, *source*, *xml*, *token*, *extension*, *namespace*, *interface*, *class*, *function*, *constant*, *global*

### reportFile

File that will contains the console results. Defaults output to console only.

### reportFileAppend

If you used the *reportFile* option, shall we replace its contents or not.

### cacheDriver

Either you use the *file* system cache, or don't want to cache results *null*.

### recursive

If you want to explore sub-directories of data source provided (*true*) or not (*false*).

### fileExtensions

A comma separated list of file extensions to parse.

### consoleProgress

Display (*true*) or not (*false*) a progress bar while scanning data source.

### verbose

Output more verbose information.

## 8.2. Cache options

```
<?xml version="1.0" encoding="utf-8" ?>
<phpcompatinfo
  ... >

  <cache id="file">
    <options>
      <save_path>/tmp</save_path>
      <gc_probability>1</gc_probability>
      <gc_maxlifetime>86400</gc_maxlifetime>
    </options>
  </cache>

</phpcompatinfo>
```

The `<cache>` element and its `<options>` child can be used to improve speed of parsing.

Default behavior will cache parsing results in a serialized data format on files backend of your local file system.

### save\_path

this is the path where the files (`pci_<md5Hash>`) are created.

### gc\_probability

*gc\_probability* is used to manage probability that the gc (garbage collection) routine is started. Defaults to 1

### gc\_maxlifetime

*gc\_maxlifetime* specifies the number of seconds after which data will be seen as *garbage* and potentially cleaned up. Garbage collection may occur after parsing data source. Defaults to 86400 (1 day)



To clean the cache, set the probability (*gc\_probability*) to 100, and reduce the life time (*gc\_maxlifetime*) to 1 second.

## References

```
<?xml version="1.0" encoding="utf-8" ?>
<phpcompatinfo
  ... >

  <references>
    <reference name="Core" />
    <reference name="standard" />
  </references>

</phpcompatinfo>
```

The `<references>` element and its `<reference>` children can be used to specify what extension you want to detect and none others.

Default behaviour will auto-detect all your extensions loaded [<http://www.php.net/manual/en/function.extension-loaded.php>].

## 8.3. Setting PHP INI settings

```
<?xml version="1.0" encoding="utf-8" ?>
<phpcompatinfo
  ... >

  <php>
    <ini name="memory_limit" value="140M" />
    <ini name="short_open_tag" />
    <ini name="zend.zel_compatibility_mode" value="false" />
  </php>

</phpcompatinfo>
```

The `<php>` element and its `<ini>` children can be used to configure PHP settings.



With **phpci** console tool, you can also sets a PHP directive value with `--ini-set` switch.

### Examples.

```
$ phpci --ini-set memory_limit=140M print /path/to/mySource

// both give same results
$ phpci --ini-set short_open_tag print /path/to/mySource
$ phpci --ini-set short_open_tag=true print /path/to/mySource
```

## 8.4. Excluding Files or Elements from parsing

```
<?xml version="1.0" encoding="utf-8" ?>
<phpcompatinfo
  ... >

  <excludes>
    <exclude id="demo">
      <directory name=".*\Zend\.*" />
      <file name=".*\.php5" />
      <extension name="xdebug" />
      <interface name="SplSubject" />
      <class name=".*Compat.*" />
      <function name="ereg.*" />
      <function name="debug_print_backtrace" />
      <constant name="T_USE" />
    </exclude>
  </excludes>

</phpcompatinfo>
```

The `<excludes>` element and its children can be used to configure what element to ignore from parsing. It may be a list of folders (`<directory>`), files, extensions, interfaces, classes, functions or constants.

Default behaviour ignore nothing.



With **phpci** console tool, you can invoke it with the following switch:

```
--exclude-pattern <id>
```

### Example.

```
$ phpci --exclude-pattern demo print /path/to/mySource
```

## 8.5. Scan Listeners

```
<?xml version="1.0" encoding="utf-8" ?>
<phpcompatinfo
  ... >

  <listeners>
    <listener class="className" file="/path/to/filename">
      <arguments>
        </arguments>
    </listener>
    <listener class="PHP_CompatInfo_Listener_File" />
    <listener class="PHP_CompatInfo_Listener_Growl">
      <arguments>
        <string>PHP_CompatInfo</string>
        <array>
          <element key="info">
            <array>
              <element key="display">
                <string>Information</string>
              </element>
              <element key="enabled">
                <boolean>>true</boolean>
              </element>
            </array>
          </element>
          <element key="warning">
            <array>
              <element key="enabled">
                <boolean>>true</boolean>
              </element>
            </array>
          </element>
        </array>
        <string>mamasam</string>
        <array>
          <element key="host">
            <string>192.168.1.2</string>
          </element>
          <element key="timeout">
            <integer>10</integer>
          </element>
          <element key="debug">
            <string>/path/to/logFile</string>
          </element>
        </array>
      </arguments>
    </listener>
  </listeners>
```



```
</phpcompatinfo>
```

The `<listeners>` element and its `<listener>` children can be used to attach additional observers to the parses process.

The **phpci** console tool know in standard distribution the **File** and **Growl** listeners.

Please refer to PEAR::Net\_Growl [<http://growl.laurent-laville.org/>] package for configuration options.

You may add your own observer. To do so, specify the class name (*class* attribute of `<listener>` element) hosted by a file (*file* attribute of the same `<listener>` element) that implement the SplObserver [<http://www.php.net/manual/en/class.spobserver.php>] interface.

## 8.6. Plugins

```
<?xml version="1.0" encoding="utf-8" ?>
<phpcompatinfo
  ... >

  <plugins>
    <reference name="MyReference"
      class="PEAR_CompatInfo"
      file="/path/to/PEARCompatInfo.php">
      <arguments>
      </arguments>
    </reference>
  </plugins>

</phpcompatinfo>
```

The `<plugins>` element and its `<reference>` children can be used to specify your own data dictionary references. Usefull when an extension data dictionary is not available in the standard distribution.

Default behaviour is to load all PHP4 and PHP5 known elements referenced by the `PHP_CompatInfo_Reference_PHP5` class. See the *reference* attribute of `<phpcompatinfo>` element.

You may add your own plugin. To do so, specify the class name (*class* attribute of `<reference>` element) hosted by a file (*file* attribute of the same `<reference>` element). Your class should inherit from `PHP_CompatInfo_Reference_PluginsAbstract` abstract class that implement the `PHP_CompatInfo_Reference` interface.



With **phpci** console tool, you can invoke it with the following switch:

```
--reference <name>
```

### Example.

```
$ phpci --reference MyReference print /path/to/mySource
```

# Appendix A. Postprocessing the report file with XSLT

## A.1. Main options

There are several XSLT scripts which can be used to transform the XML report into some nice html pages.

To do this, make sure you've first generate the XML report, i.e:

```
$ phpcli --report xml --report-file <report.xml> print <dataSource>
```

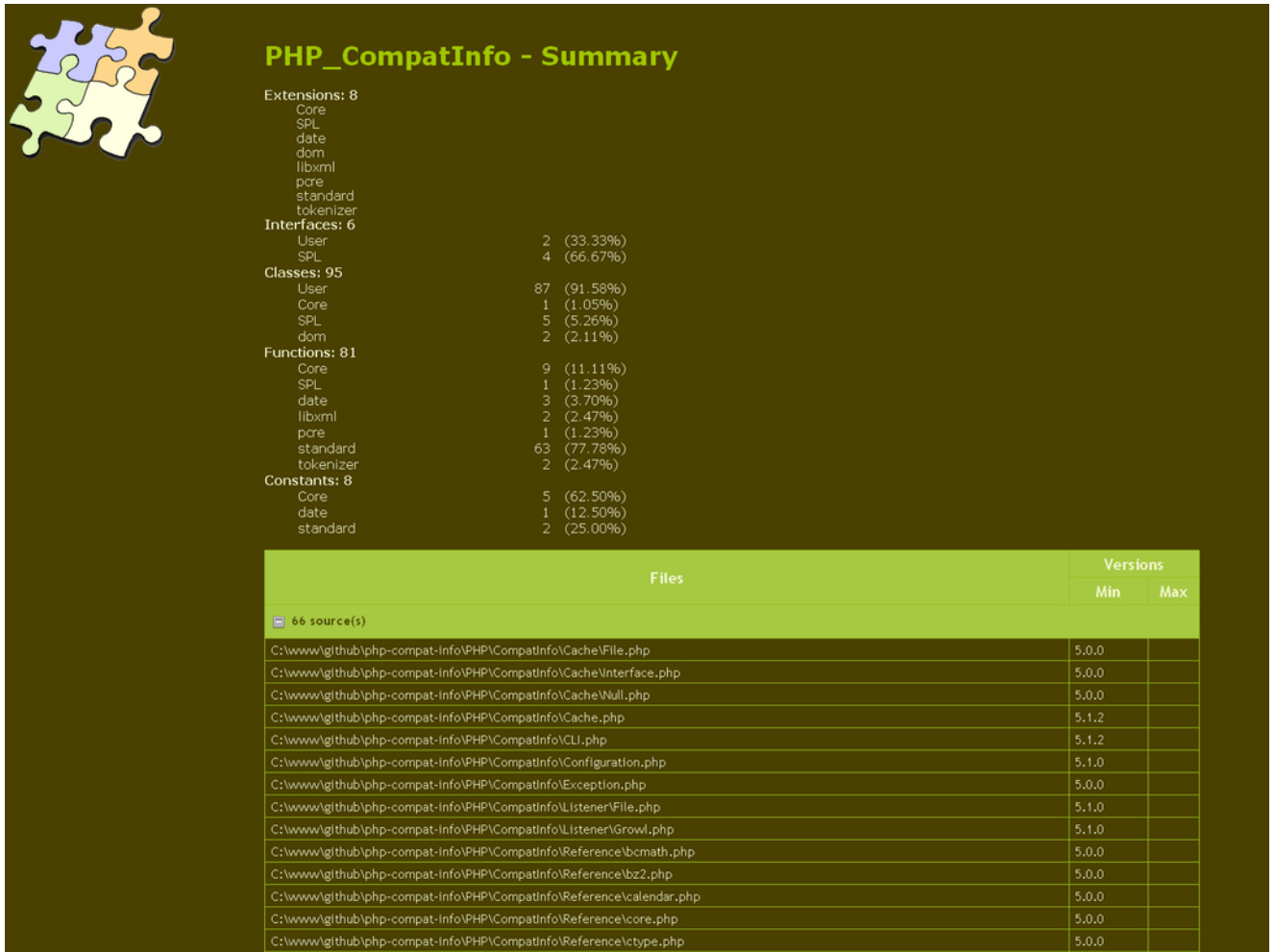
And then use the xslt processor to transform the XML to a beautiful XHTML page.

```
$ xsltproc -o <output_page.html> summary.xsl <report.xml>
```

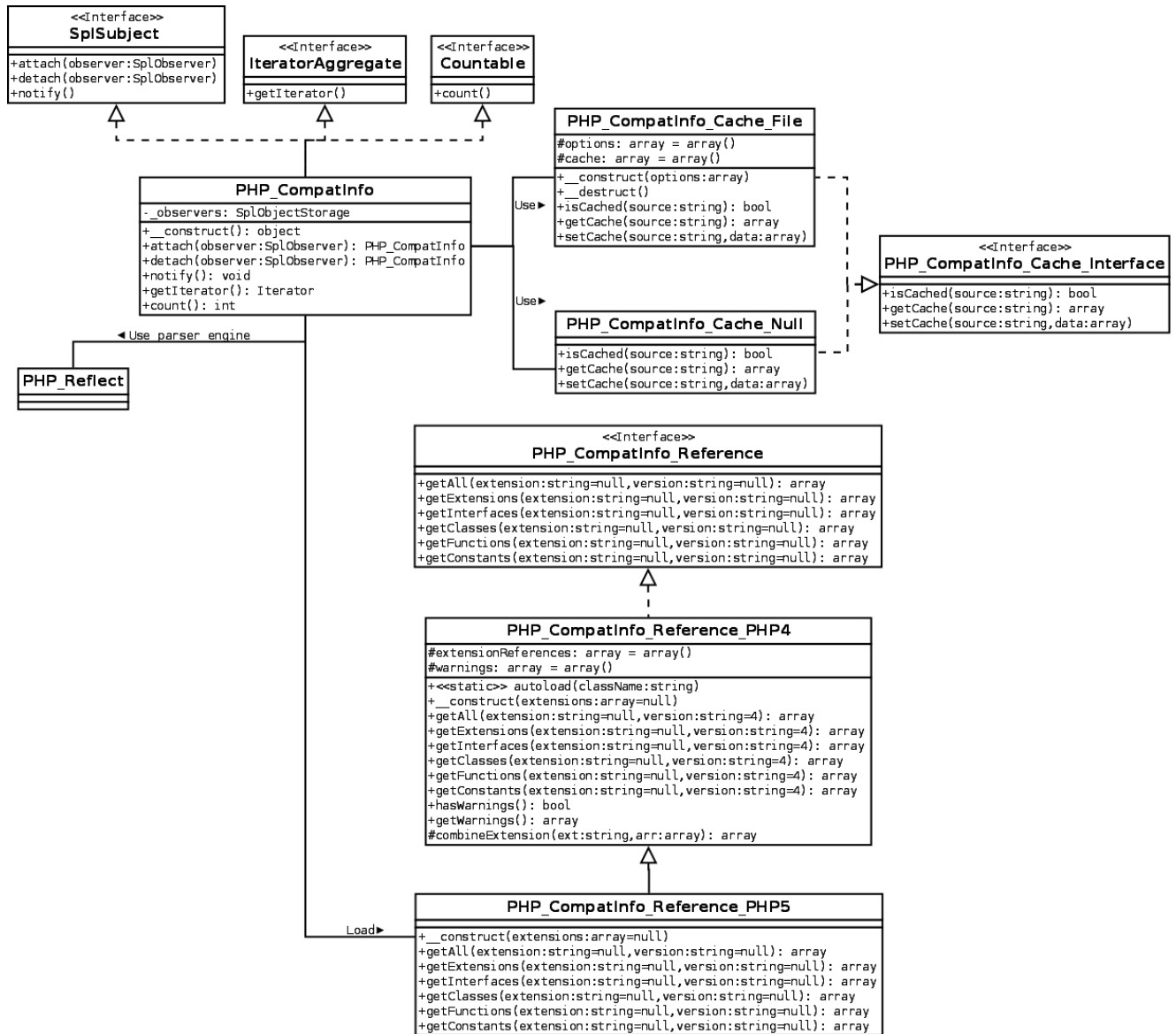
**Figure A.1. Example with sources file list collapsed**



**Figure A.2. Example with sources file list expanded**



# Appendix B. UML classes diagram



---

# Appendix C. Copyright

This work is licensed under the BSD License.

The full legal text of the license is given below.

```
Copyright (c) 2010-2012, Laurent Laville <pear@laurent-laville.org>
```

```
Credits to :
```

- \* Davey Shafik  
Original author, he introduced his proposal in 2004,  
that gave birth of a PEAR package named PHP\_CompatInfo.
- \* Remi Collet  
Contributor on many extensions and unit tests since version 2.0.0RC2

```
All rights reserved.
```

```
Redistribution and use in source and binary forms, with or without  
modification, are permitted provided that the following conditions  
are met:
```

- \* Redistributions of source code must retain the above copyright  
notice, this list of conditions and the following disclaimer.
- \* Redistributions in binary form must reproduce the above copyright  
notice, this list of conditions and the following disclaimer in the  
documentation and/or other materials provided with the distribution.
- \* Neither the name of the authors nor the names of its contributors  
may be used to endorse or promote products derived from this software  
without specific prior written permission.

```
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"  
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE  
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE  
ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS  
BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR  
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF  
SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS  
INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN  
CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)  
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE  
POSSIBILITY OF SUCH DAMAGE.
```