PHP CompatInfo Book

Laurent Laville

PHP CompatInfo Book

Laurent Laville

Table of Contents

List of Examples

This complete guide documents PHP CompatInfo 4.0.0-beta2, published on 2015-02-20.

Part I. Getting Started

Chapter 1. Download

We distribute a PHP Archive [http://www.php.net/phar] (PHAR) that contains all required dependencies of PHP CompatInfo bundled in a single file.

Download the latest version [http://bartlett.laurent-laville.org/get/phpcompatinfo-4.0.0-beta2.phar]

Make it executable, and put it into your `$PATH`.

```
$ chmod +x phpcompatinfo-4.0.0-beta2.phar
$ mv phpcompatinfo-4.0.0-beta2.phar /usr/local/bin/phpcompatinfo

$ phpcompatinfo --version
```

You can also immediately use the PHAR after you have downloaded it.

```
$ wget http://bartlett.laurent-laville.org/get/phpcompatinfo-4.0.0-beta2.phar
$ php phpcompatinfo-4.0.0-beta2.phar --version
```

With both methods then you have this output :

```
phpCompatInfo version 4.0.0-beta2 DB built Feb 20 2015 09:51:51 CET
```

Other alternative installations are possible. Please refer to the Chapter 7, *Installation* for details on how to do this.

Chapter 2. Configuration

With the minimalist JSON file `phpcompatinfo.json`.

```
{
    "source-providers": [
        {
            "in": ". as current",
            "name": "/\\.(php|inc|phtml)$/"
        }
    ],
    "plugins": [
    ],
    "analysers": [
    ],
    "services": [
    ]
}
```

Put it in your project's folder. Alternative locations are possible. Please refer to the Chapter 8, *The Json Configuration File* for details on how to do this.

The JSON configuration file is no more required for basic usage. Reserved to advanced users.

Chapter 3. Structure

source-providers
: this entry provide list of your data sources to parse.

plugins
: this entry list all plugins added to the core base code of PHP Reflect.

analysers
: this entry list all analysers that may be used with the `analyser:run` command.

services
: this entry list all services that may be used with this application.

Chapter 4. Execution

With the CompatInfo source code, invoke the following command :

```
$ phpcompatinfo analyser:run .
```

and you should obtain something like this :

```
Data Source Analysed

Directories                                          11
Files                                                13

Extensions Analysis

    Extension Matches REF      EXT min/Max PHP min/Max
    Core              Core     5.3.0       5.3.0
    PDO       1       PDO      5.1.0       5.1.0
    curl              curl     4.0.2       4.0.2
    intl              intl     2.0.0b1     5.3.7
    libxml            libxml   5.1.0       5.1.0
    openssl           openssl  5.2.0       5.2.0
    pcre              pcre     4.0.0       4.0.0
    spl               spl      5.1.0       5.1.0
    standard          standard 5.3.2       5.3.2
    Total [9]                              5.3.7

Namespaces Analysis

    Namespace                               Matches REF  EXT min/Max PHP min/Max
    +global                                         Core             4.0.0
    Bartlett\CompatInfo                             user             5.3.0
    Bartlett\CompatInfo\Analyser                    user             5.3.0
    Bartlett\CompatInfo\Api                         user             5.3.2
    Bartlett\CompatInfo\Api\V3                      user             5.3.0
    Bartlett\CompatInfo\Collection                  user             5.3.0
    Bartlett\CompatInfo\Console                     user             5.3.0
    Bartlett\CompatInfo\Console\Formatter           user             5.3.0
    Bartlett\CompatInfo\Output                      user             5.3.0
    Bartlett\CompatInfo\PhpParser                   user             5.3.0
    Bartlett\CompatInfo\Reference                   user             5.3.0
    Bartlett\CompatInfo\Util                        user             5.3.0
    Total [12]                                                       5.3.2

Interfaces Analysis

    Interface                                          Matches REF  EXT min/Max PHP min/Ma
    Bartlett\CompatInfo\Reference\ReferenceInterface 1       user             5.3.0
    Total [1]                                                                 5.3.0

No trait found

Classes Analysis

    Class                                                              Matches REF  EXT
    Bartlett\CompatInfo\Analyser\CompatibilityAnalyser                         user
    Bartlett\CompatInfo\Api\Reference                                          user
    Bartlett\CompatInfo\Api\V3\Reference                                       user
    Bartlett\CompatInfo\Collection\ReferenceCollection                 1       user
```

```
    Bartlett\CompatInfo\Console\Application                                  user
    Bartlett\CompatInfo\Console\Formatter\CompatibilityOutputFormatter       user
    Bartlett\CompatInfo\Environment                                  4       user
    Bartlett\CompatInfo\Output\Reference                                     user
    Bartlett\CompatInfo\PhpParser\ConditionalCodeNodeProcessor       1       user
    Bartlett\CompatInfo\Reference\ExtensionFactory                   2       user
    Bartlett\CompatInfo\Reference\SqliteStorage                      1       user
    Bartlett\CompatInfo\Util\Version                                 5       user
  U Bartlett\Reflect\Analyser\AbstractAnalyser                       1       user
  U Bartlett\Reflect\Api\BaseApi                                     1       user
  U Bartlett\Reflect\Api\V3\Common                                   1       user
  U Bartlett\Reflect\Console\Application                             1       user
  U Bartlett\Reflect\Console\Formatter\OutputFormatter               2       user
  U Bartlett\Reflect\PhpParser\NodeProcessorAbstract                 1       user
  U Doctrine\Common\Collections\AbstractLazyCollection               1       user
  U Doctrine\Common\Collections\ArrayCollection                      1       user
    PDO                                                              2       PDO  5.1.
  U PhpParser\Node                                                   21      user
  U PhpParser\Node\Name                                              1       user
    RuntimeException                                                 1       spl  5.1.
  U Symfony\Component\Console\Helper\TableSeparator                  3       user
  U Symfony\Component\Console\Output\OutputInterface                 4       user
    parent                                                           5       Core 5.0.
    self                                                             8       Core 5.0.
    stdClass                                                         1       Core 4.0.
    Total [29]

Functions Analysis

    Function                  Matches REF      EXT min/Max PHP min/Max
    array_key_exists          4       standard 4.0.7       4.0.7
    array_map                 1       standard 4.0.6       4.0.6
    array_merge               2       standard 4.0.0       4.0.0
    array_pop                 10      standard 4.0.0       4.0.0
    array_shift               2       standard 4.0.0       4.0.0
    array_slice               1       standard 4.0.0       4.0.0
    array_values              1       standard 4.0.0       4.0.0
    closure-71-73                     user                 5.3.0
    copy                      1       standard 4.0.0       4.0.0
    count                     11      standard 4.0.0       4.0.0
 C  curl_version              1       curl     4.0.2       4.0.2
    defined                   4       Core     4.0.0       4.0.0
    dirname                   3       standard 4.0.0       4.0.0
    each                      2       Core     4.0.0       4.0.0
    explode                   1       standard 4.0.0       4.0.0
    extension_loaded          1       Core     4.0.0       4.0.0
    file_exists               2       standard 4.0.0       4.0.0
    function_exists           1       Core     4.0.0       4.0.0
    in_array                  7       standard 4.0.0       4.0.0
    is_array                  1       standard 4.0.0       4.0.0
    is_callable               1       standard 4.0.6       4.0.6
    is_string                 10      standard 4.0.0       4.0.0
    ksort                     3       standard 4.0.0       4.0.0
    mkdir                     1       standard 4.0.0       4.0.0
    phpversion                2       standard 4.0.0       4.0.0
    preg_match                2       pcre     4.0.0       4.0.0
    sha1_file                 2       standard 4.3.0       4.3.0
    sprintf                   22      standard 4.0.0       4.0.0
    strcasecmp                2       Core     4.0.0       4.0.0
```

```
    stream_resolve_include_path 1       standard 5.3.2       5.3.2
    strpos                      1       standard 4.0.0       4.0.0
    strtolower                  1       standard 4.0.0       4.0.0
    substr                      2       standard 4.0.0       4.0.0
    sys_get_temp_dir            1       standard 5.2.1       5.2.1
    ucfirst                     5       standard 4.0.0       4.0.0
    version_compare             7       standard 4.0.7       4.0.7
    Total [36]                                               5.3.2

Constants Analysis

    Constant               Matches REF     EXT min/Max PHP min/Max
 C  INTL_ICU_VERSION       2       intl    2.0.0b1     5.3.7
    LIBXML_DOTTED_VERSION  1       libxml  5.1.0       5.1.0
    LIBXML_VERSION         1       libxml  5.1.0       5.1.0
 C  OPENSSL_VERSION_NUMBER 1       openssl 5.2.0       5.2.0
 C  OPENSSL_VERSION_TEXT   1       openssl 5.2.0       5.2.0
    PHP_EOL                5       Core    4.3.10      4.3.10
    PHP_VERSION            4       Core    4.0.0       4.0.0
    __DIR__                1       Core    5.3.0       5.3.0
    false                  12      Core    4.0.0       4.0.0
    null                   13      Core    4.0.0       4.0.0
    true                   8       Core    4.0.0       4.0.0
    Total [11]                                         5.3.0

Conditions Analysis

    Condition                       Matches REF     EXT min/Max PHP min/Max
    defined(INTL_ICU_VERSION)       2       intl    2.0.0b1     5.3.7
    defined(OPENSSL_VERSION_NUMBER) 1       openssl 5.2.0       5.2.0
    defined(OPENSSL_VERSION_TEXT)   1       openssl 5.2.0       5.2.0
    function_exists(curl_version)   1       curl    4.0.2       4.0.2
    Total [4]                                                   5.3.7

Requires PHP 5.3.2 (min)

Time: 11.58 seconds, Memory: 11.50Mb
```

Chapter 5. Summary

Let’s review what we’ve done :

- downloaded the latest stable PHAR version.
- prepared a minimalist JSON configuration file that is OPTIONAL to run CompatInfo commands.
- executed your first parse on the CompatInfo data source.

Chapter 6. Next

Choose your way depending of your skill level.

Read more

- Want to learn more about the command line interpreter (CLI) version, interface that do CompatInfo an easy tool without to write a line of PHP code, have a look on Part II, “User Guide”
- Want to learn more about CompatInfo architecture and /or you want to extends it to match your needs, have a look on Part IV, “Developer Guide”
- You are a user of previous version 2.26 that is really different, and want to upgrade to the new major version 3, and keep your old environment still running, have a look on Part III, “Migration Guide”

Part II. User Guide

First visit, you are highly recommended to follow chapters in following order.

1. Installing all necessary CompatInfo components. See Chapter 7, *Installation*
2. Configuring your project and get ready for your first parsing. See Chapter 8, *The Json Configuration File*
3. Running your first parses with the Command-Line interface. See Chapter 9, *The Command-Line*

All you have to know if you want to upgrade from a previous version 2.x easily.

See Part III, “Migration Guide”

Basic CompatInfo features does not match your needs. Learn how to extend or change some features/behaviors.

See Part IV, “Developer Guide”

Chapter 7. Installation

CompatInfo may be installed in several ways, choose your favorite.

Please read the Part III, “Migration Guide” in case you are upgrading from a version 2.x of PHP CompatInfo.

7.1. Requirements

Before you install PHP CompatInfo, you will need an operating system with PHP [http://www.php.net] 5.3.2 or later installed,

CompatInfo requires the json [http://www.php.net/manual/en/book.json.php], libxml [http://www.php.net/manual/en/book.libxml.php], pdo [http://www.php.net/manual/en/book.pdo.php], pcre [http://www.php.net/manual/en/book.pcre.php], and spl [http://www.php.net/manual/en/book.spl.php] extensions. These extensions are usually compiled and enabled by default.

7.2. Composer

Put a file named composer.json at the root of your project, with the content below:

```
{
    "require": {
        "bartlett/php-compatinfo": "4.0.0-beta2"
    }
}
```

And ask Composer [http://getcomposer.org/] to install the dependencies:

```
$ php composer.phar install
```

With composer `install` or `create-project` commands, if you want to disable installation of require-dev packages (`doctrine/cache`, `psr/log`, `monolog/monolog`, `bartlett/phpunit-loggertestlistener`), don’t forget to specify the `--no-dev` option.

You can also use Composer to create a new project from an existing CompatInfo package. This is the equivalent of doing a git clone checkout followed by a composer install of the vendors.

```
$ php composer.phar create-project bartlett/php-compatinfo /path/to/install 4.0
```

Where `/path/to/install` is your install directory.

7.3. PHAR

The recommended way for newbies, or just to have a look on features of this library, is to download a PHP Archive that contain all required dependencies of PHP CompatInfo bundled in a single file.

```
$ wget http://bartlett.laurent-laville.org/get/phpcompatinfo-4.0.0-beta2.phar
```

```
$ chmod +x phpcompatinfo-4.0.0-beta2.phar
$ mv phpcompatinfo-4.0.0-beta2.phar /usr/local/bin/phpcompatinfo
$ phpcompatinfo
```

You can also immediately use the PHAR after you have downloaded it.

```
$ wget http://bartlett.laurent-laville.org/get/phpcompatinfo-4.0.0-beta2.phar
$ php phpcompatinfo-4.0.0-beta2.phar
```

Chapter 8. The Json Configuration File

CompatInfo may used an optional config file in JSON [http://json.org/] format. It could be found either in the current, `$HOME/.config/`, or `/etc` directory.

By setting the `BARTLETTRC` environment variable it is possible to set the filename of `phpcompatinfo.json` to something else.

E.g: `BARTLETTRC=my-phpcompatinfo.json`

And by setting the `BARTLETT_SCAN_DIR` environment variable it is possible to change directories where to search for the json config file.

E.g: `BARTLETT_SCAN_DIR=.:/var/configs:/tmp/bartlett` (for Linux)

E.g: `BARTLETT_SCAN_DIR=.;\var\configs;\tmp\bartlett` (for Windows)

Take care of different PATH_SEPARATOR and DIRECTORY_SEPARATOR in each platform.

The minimalist JSON file `phpcompatinfo.json` is :

```
{
    "source-providers": [
        {
            "in": ". as current",
            "name": "/\\.(php|inc|phtml)$/"
        }
    ],
    "plugins": [
    ],
    "analysers": [
    ],
    "services": [
    ]
}
```

source-providers
: this entry provide list of your data sources to parse.

plugins
: this entry list all plugins added to the core base code of PHP CompatInfo.

analysers
: this entry list all analysers that may be used with the `analyser:run` command.

services
: this entry list all services that may be used with this application.

8.1. section Source Providers

There are lot of way to filter your data source. Each rule follow the syntax of Symfony Finder [http://symfony.com/doc/current/components/finder.html] Component.

The **Location** is the only mandatory criteria. It tells the Finder which directory to use for the search.

In a simple directory.

```
{
    "in": ". as current"
}
```

If you want to identify a data source easily by a short name, the alias (right of `as`) is compared with the `--alias` option constraint.

Search in several locations.

```
{
    "in": ". as current",
    "in": "src/"
}
```

Use wildcard characters to search in the directories matching a pattern:

```
{
    "in": "src/Bartlett/R*"
}
```

Search directly in archives (phar, zip, tar) with the `phar://` protocol.

```
{
    "in": "phar://path/to/archive.zip"
}
```

Restrict files by name and/or extension.

```
{
    "in": "phar://path/to/archive.zip",
    "name": "*.php"
}
```

Restrict files by size.

```
{
    "in": "phar://path/to/archive.zip",
    "name": "*.php",
    "size": "< 10K"
}
```

Restrict files by last modified dates.

```
{
    "in": ". as current",
    "date": "since yesterday"
}
```

By default, the Finder recursively traverse directories.

Restrict the depth of traversing.

```
{
```

```
        "in": ". as current",
        "depth": "< 3"
    }
```

Restrict location by only one directory.

```
    {
        "in": ". as current",
        "exclude": "vendor"
    }
```

Restrict location by 1 or more directories.

```
    {
        "in": ". as current",
        "exclude": ["vendor", "tests"]
    }
```

8.2. section Plugins

There are a number of optional plugins you can use along with CompatInfo to add more capabilities.

Take an example with the `Logger` plugin.

In your `phpcompatinfo.json` configuration file, add in `plugins` section the following entry:

```
    {
        "name": "Logger",
        "class": "Bartlett\\Reflect\\Plugin\\LogPlugin"
    }
```

- The `name` key is (since version 4.0.0-alpha1) comment only.
- The `class` key identify the name of the class that implement the plugin features (must be fully qualified).

The `LogPlugin` used by default the `Bartlett\Reflect\Plugin\Log\DefaultLogger` class that write results to error_log

8.2.1. Cache Plugin

Available only since version 3.3.0

In your `phpcompatinfo.json` configuration file, add in `plugins` section the following entry:

```
    {
        "name": "Cache",
        "class": "Bartlett\\Reflect\\Plugin\\CachePlugin",
        "options": {
            "adapter": "DoctrineCacheAdapter",
            "backend": {
```

```
                "class": "Doctrine\\Common\\Cache\\FilesystemCache",
                "args": [
                    "%{TEMP}/bartlett/cache"
                ]
            }
        }
    }
```

You may use any environment variable that will be replaced, at run-time, by their value. E.g: `TEMP`, `HOME`

Since release 3.3.0, the `HOME` syntax is compatible Linux/Windows.

Take care to use the same configuration as in PHP Reflect, or you should not share the cache results.

If you want to used the same options (Doctrine adapter with file cache) as above, you can used shortcut syntax like this.

```
    {
        "name": "Cache",
        "class": "Bartlett\\Reflect\\Plugin\\CachePlugin",
        "options": []
    }
```

In previous configuration we used the Doctrine Cache adapter and its File system backend. See the same configuration applied with other SAPI, in Section 16.3, “File cache”

8.2.2. Log Plugin

Available only since version 3.4.0

In your `phpcompatinfo.json` configuration file, add in `plugins` section the following entry:

```
    {
        "name": "Log",
        "class": "Bartlett\\Reflect\\Plugin\\LogPlugin"
    }
```

Where `options` key identify an optional class logger (fully qualified. E.g `YourNamespace\YourLogger`).

When `options` key is not provided, log plugin used the default Reflect logger bundled with distribution. See `Bartlett\Reflect\Plugin\Log\DefaultLogger` that write results to the error log system.

See the Developer Guide for definition examples of some loggers Section 17.3, “Using your private logger” or Section 17.4, “Using Monolog”

8.3. section Analysers

There are default analysers you can use, but you are free to add your owns.

In your `phpcompatinfo.json` configuration file, add in `analysers` section (for example) the following entry:

```
    {
        "name": "MyAnalyser",
        "class": "Your\\Analysers\\MyAnalyser"
    }
```

- The `name` key is (since version 4.0.0-alpha1) comment only.
- The `class` key identify the name of the class that implement your analyser (must be fully qualified).

Your analyser should implement both interfaces `Bartlett\Reflect\Analyser\AnalyserInterface` and `PhpParser\NodeVisitor`.

Then to use it in command line :

```
$ phpcompatinfo analyser:run /path/to/datasource my
```

`my` identify your analyser (prefix in lower case of `MyAnalyser` class)

Chapter 9. The Command-Line

The command-line interface is the easiest way to try and learn the basic CompatInfo features.

For all users.

9.1. Command-Line Options

Without `plugins` and `analysers` sections in your `phpcompatinfo.json` configuration file, when you invoke the `phpcompatinfo` command, you should obtain the following commands and options :

```
       _                 ____                                  _   ___          __
  _ __ | |__  _ __  ___ / ___|___  _ __ ___  _ __   __ _| |_|_ _|_ __  / _| ___
 | '_ \| '_ \| '_ \| |  / _ \| '_ ` _ \| '_ \ / _` | __|| || '_ \| |_ / _ \
 | |_) | | | | |_) | |__| (_) | | | | | | |_) | (_| | |_ | || | | |  _| (_) |
 | .__/|_| |_| .__/ \____\___/|_| |_| |_| .__/ \__,_|\__|___|_| |_|_|  \___/
 |_|         |_|                        |_|

phpCompatInfo version 4.0.0-beta2 DB built Feb 20 2015 09:51:51 CET

Usage:
 [options] command [arguments]

Options:
 --help (-h)           Display this help message
 --quiet (-q)          Do not output any message
 --verbose (-v|vv|vvv) Increase the verbosity of messages: 1 for normal output, 2 for mo
 --version (-V)        Display this application version
 --ansi                Force ANSI output
 --no-ansi             Disable ANSI output
 --no-interaction (-n) Do not ask any interactive question
 --no-plugins          Disables all plugins.
 --profile             Display timing and memory usage information.
 --progress            Show progress bar.
 --output              Write results to file or URL.
 --manifest            Show which versions of dependencies are bundled.

Available commands:
 help                  Displays help for a command
 list                  Lists commands
analyser
 analyser:list         List all analysers available.
 analyser:run          Analyse a data source and display results.
cache
 cache:clear           Clear cache (any adapter and backend).
config
 config:validate       Validates a JSON configuration file.
plugin
 plugin:list           List all plugins installed.
reference
 reference:list        List all references supported.
 reference:show        Show information about a reference.
reflection
 reflection:class      Reports information about a user class present in a data source.
```

```
 reflection:function   Reports information about a user function present in a data sourc
```

config:validate Validates an optional JSON config file.

```
$ phpcompatinfo config:validate
```

```
"/etc/phpcompatinfo.json" config file is valid.
```

plugin:list List all plugins configured (and correctly installed) in `plugins` section of your `phpcompatinfo.json` config file.

Without plugins, you will get.

```
$ phpcompatinfo plugin:list
```

```
No plugin installed.
```

With only `Cache` plugin configured, you will get.

```
$ phpcompatinfo plugin:list
```

```
 Plugin Class                              Events Subscribed
 Bartlett\Reflect\Plugin\CachePlugin       reflect.progress
                                           reflect.success
                                           reflect.complete
```

analyser:list List all analysers configured in `analysers` section of your `phpcompatinfo.json` config file, and available by default.

With only default analysers, you will get.

```
$ phpcompatinfo analyser:list
```

```
 Analyser Name Analyser Class
 loc           Bartlett\Reflect\Analyser\LocAnalyser
 reflection    Bartlett\Reflect\Analyser\ReflectionAnalyser
 structure     Bartlett\Reflect\Analyser\StructureAnalyser
 compatibility Bartlett\CompatInfo\Analyser\CompatibilityAnalyser
```

analyser:run Parse a data source and display results. May vary depending of the data source and analyser used.

With `compatibility` analyser and the CompatInfo source code, you will get something like.

```
$ phpcompatinfo analyser:run .
```

Possible alternative (if you use the default json config file).

```
$ phpcompatinfo analyser:run --alias current
```

```
Data Source Analysed

Directories                                          11
Files                                                13

Extensions Analysis

    Extension Matches REF      EXT min/Max PHP min/Max
    Core              Core     5.3.0       5.3.0
    PDO       1       PDO      5.1.0       5.1.0
```

```
    curl              curl     4.0.2       4.0.2
    intl              intl     2.0.0b1     5.3.7
    libxml            libxml   5.1.0       5.1.0
    openssl           openssl  5.2.0       5.2.0
    pcre              pcre     4.0.0       4.0.0
    spl               spl      5.1.0       5.1.0
    standard          standard 5.3.2       5.3.2
    Total [9]                               5.3.7

Namespaces Analysis

    Namespace                              Matches REF  EXT min/Max PHP min/M
    +global                                        Core             4.0.0
    Bartlett\CompatInfo                            user             5.3.0
    Bartlett\CompatInfo\Analyser                   user             5.3.0
    Bartlett\CompatInfo\Api                        user             5.3.2
    Bartlett\CompatInfo\Api\V3                     user             5.3.0
    Bartlett\CompatInfo\Collection                 user             5.3.0
    Bartlett\CompatInfo\Console                    user             5.3.0
    Bartlett\CompatInfo\Console\Formatter          user             5.3.0
    Bartlett\CompatInfo\Output                     user             5.3.0
    Bartlett\CompatInfo\PhpParser                  user             5.3.0
    Bartlett\CompatInfo\Reference                  user             5.3.0
    Bartlett\CompatInfo\Util                       user             5.3.0
    Total [12]                                                      5.3.2

Interfaces Analysis

    Interface                                          Matches REF  EXT min/Ma
    Bartlett\CompatInfo\Reference\ReferenceInterface 1       user
    Total [1]

No trait found

Classes Analysis

    Class                                                                  Match
    Bartlett\CompatInfo\Analyser\CompatibilityAnalyser
    Bartlett\CompatInfo\Api\Reference
    Bartlett\CompatInfo\Api\V3\Reference
    Bartlett\CompatInfo\Collection\ReferenceCollection                     1
    Bartlett\CompatInfo\Console\Application
    Bartlett\CompatInfo\Console\Formatter\CompatibilityOutputFormatter
    Bartlett\CompatInfo\Environment                                        4
    Bartlett\CompatInfo\Output\Reference
    Bartlett\CompatInfo\PhpParser\ConditionalCodeNodeProcessor             1
    Bartlett\CompatInfo\Reference\ExtensionFactory                         2
    Bartlett\CompatInfo\Reference\SqliteStorage                            1
    Bartlett\CompatInfo\Util\Version                                       5
  U Bartlett\Reflect\Analyser\AbstractAnalyser                             1
  U Bartlett\Reflect\Api\BaseApi                                           1
  U Bartlett\Reflect\Api\V3\Common                                         1
  U Bartlett\Reflect\Console\Application                                   1
  U Bartlett\Reflect\Console\Formatter\OutputFormatter                     2
  U Bartlett\Reflect\PhpParser\NodeProcessorAbstract                       1
  U Doctrine\Common\Collections\AbstractLazyCollection                     1
  U Doctrine\Common\Collections\ArrayCollection                            1
    PDO                                                                    2
  U PhpParser\Node                                                         21
```

```
  U PhpParser\Node\Name                                                  1
    RuntimeException                                                     1
  U Symfony\Component\Console\Helper\TableSeparator                      3
  U Symfony\Component\Console\Output\OutputInterface                     4
    parent                                                               5
    self                                                                 8
    stdClass                                                             1
    Total [29]

Functions Analysis

    Function                    Matches REF      EXT min/Max PHP min/Max
    array_key_exists            4       standard 4.0.7       4.0.7
    array_map                   1       standard 4.0.6       4.0.6
    array_merge                 2       standard 4.0.0       4.0.0
    array_pop                   10      standard 4.0.0       4.0.0
    array_shift                 2       standard 4.0.0       4.0.0
    array_slice                 1       standard 4.0.0       4.0.0
    array_values                1       standard 4.0.0       4.0.0
    closure-71-73                       user                 5.3.0
    copy                        1       standard 4.0.0       4.0.0
    count                       11      standard 4.0.0       4.0.0
 C  curl_version                1       curl     4.0.2       4.0.2
    defined                     4       Core     4.0.0       4.0.0
    dirname                     3       standard 4.0.0       4.0.0
    each                        2       Core     4.0.0       4.0.0
    explode                     1       standard 4.0.0       4.0.0
    extension_loaded            1       Core     4.0.0       4.0.0
    file_exists                 2       standard 4.0.0       4.0.0
    function_exists             1       Core     4.0.0       4.0.0
    in_array                    7       standard 4.0.0       4.0.0
    is_array                    1       standard 4.0.0       4.0.0
    is_callable                 1       standard 4.0.6       4.0.6
    is_string                   10      standard 4.0.0       4.0.0
    ksort                       3       standard 4.0.0       4.0.0
    mkdir                       1       standard 4.0.0       4.0.0
    phpversion                  2       standard 4.0.0       4.0.0
    preg_match                  2       pcre     4.0.0       4.0.0
    sha1_file                   2       standard 4.3.0       4.3.0
    sprintf                     22      standard 4.0.0       4.0.0
    strcasecmp                  2       Core     4.0.0       4.0.0
    stream_resolve_include_path 1       standard 5.3.2       5.3.2
    strpos                      1       standard 4.0.0       4.0.0
    strtolower                  1       standard 4.0.0       4.0.0
    substr                      2       standard 4.0.0       4.0.0
    sys_get_temp_dir            1       standard 5.2.1       5.2.1
    ucfirst                     5       standard 4.0.0       4.0.0
    version_compare             7       standard 4.0.7       4.0.7
    Total [36]                                               5.3.2

Constants Analysis

    Constant               Matches REF     EXT min/Max PHP min/Max
 C  INTL_ICU_VERSION       2       intl    2.0.0b1     5.3.7
    LIBXML_DOTTED_VERSION  1       libxml  5.1.0       5.1.0
    LIBXML_VERSION         1       libxml  5.1.0       5.1.0
 C  OPENSSL_VERSION_NUMBER 1       openssl 5.2.0       5.2.0
 C  OPENSSL_VERSION_TEXT   1       openssl 5.2.0       5.2.0
    PHP_EOL                5       Core    4.3.10      4.3.10
```

```
    PHP_VERSION             4       Core    4.0.0       4.0.0
    __DIR__                 1       Core    5.3.0       5.3.0
    false                   12      Core    4.0.0       4.0.0
    null                    13      Core    4.0.0       4.0.0
    true                    8       Core    4.0.0       4.0.0
    Total [11]                                          5.3.0

Conditions Analysis

    Condition                       Matches REF     EXT min/Max PHP min/Max
    defined(INTL_ICU_VERSION)       2       intl    2.0.0b1     5.3.7
    defined(OPENSSL_VERSION_NUMBER) 1       openssl 5.2.0       5.2.0
    defined(OPENSSL_VERSION_TEXT)   1       openssl 5.2.0       5.2.0
    function_exists(curl_version)   1       curl    4.0.2       4.0.2
    Total [4]                                                   5.3.7

Requires PHP 5.3.2 (min)

Time: 11.58 seconds, Memory: 11.50Mb
```

reflection:class Reports information about a user class present in a data source.

With the CompatInfo source code (`./src`), and `Bartlett\CompatInfo\Environment` class.

```
$ phpcompatinfo reflection:class Bartlett\CompatInfo\Environment ./src
```

```
Class [ <user> class Bartlett\CompatInfo\Environment ] {
  @@ C:\home\github\php-compat-info\src\Bartlett\CompatInfo\Environment.php

  - Constants [0] {
  }

  - Properties [0] {
  }

  - Methods [2] {
    Method [ <user> public method initRefDb ] {
      @@ C:\home\github\php-compat-info\src\Bartlett\CompatInfo\Environment.

      - Parameters [0] {
      }
    }

    Method [ <user> public method versionRefDb ] {
      @@ C:\home\github\php-compat-info\src\Bartlett\CompatInfo\Environment.

      - Parameters [0] {
      }
    }
  }
}
```

Chapter 10. Summary

Let’s review what we’ve learned about the command-line interface :

- It’s a Symfony Console Component [http://symfony.com/doc/current/components/console/index.html] that can be extended to infinite via plugins and analysers.
- You can examine inside a reference and filters elements.

Part III. Migration Guide

 **CLI**

Begin first with the commands of CompatInfo in CLI mode.

See Chapter 11, *CLI*

 **XML**

How you can customize CompatInfo in CLI mode only.

See Chapter 12, *Configuration file*

 **API**

Due to namespaces, API are incompatible in versions 2 and 3. Here are code to do the same things.

See Chapter 13, *Server API*

Chapter 11. CLI

11.1. Progress Bar

Since version 2.12, there is a new progress bar (ALA PHPUnit).

```
PHP_CompatInfo 2.12.0 by Laurent Laville

....CC.................................................  60 / 128 ( 46%)
....C..C.....................................C....C.... 120 / 128 ( 93%)
.C.....C
```

- each `C` character tell us that there is a source file with conditional code.

With version 3.2, the progress bar (ALA Symfony Console Progress Helper) is activated with the first verbose level (`-v`). For example :

```
$ phpcompatinfo -v analyser:run <SOURCE>
```

Renders

```
    3/128 [>---------------------------]   0% Elapsed: 2 secs
```

References

If we want to display list of references supported, here is how to do with both versions :

Version 2.26.

```
$ phpcompatinfo list-references
```

Version 3.2.

```
$ phpcompatinfo reference:list
```

Now, if you want to details each reference one by one identified by its name (`<REF>` in following examples), and got :

- Extensions

Version 2.26.

```
$ phpcompatinfo list --reference=ALL extensions
or
$ phpcompatinfo list-extensions --reference=ALL
```

Version 3.2.

```
$ phpcompatinfo reference:list
```

- Interfaces

Version 2.26.

```
$ phpcompatinfo list --reference=ALL interfaces <REF>
or
$ phpcompatinfo list-interfaces --reference=ALL <REF>
```

Version 3.2.

```
$ phpcompatinfo reference:show --interfaces <REF>
```

- Classes

Version 2.26.

```
$ phpcompatinfo list --reference=ALL classes <REF>
or
$ phpcompatinfo list-classes --reference=ALL <REF>
```

Version 3.2.

```
$ phpcompatinfo reference:show --classes <REF>
```

- Functions

Version 2.26.

```
$ phpcompatinfo list --reference=ALL functions <REF>
or
$ phpcompatinfo list-functions --reference=ALL <REF>
```

Version 3.2.

```
$ phpcompatinfo reference:show --functions <REF>
```

- Constants

Version 2.26.

```
$ phpcompatinfo list --reference=ALL constants <REF>
or
$ phpcompatinfo list-constants --reference=ALL <REF>
```

Version 3.2.

```
$ phpcompatinfo reference:show --constants <REF>
```

- INI entries

Feature not provided by version 2.26

Version 3.2.

```
$ phpcompatinfo reference:show --ini <REF>
```

And if you want to filter results on PHP version, do for example :

Version 2.26.

```
$ phpcompatinfo list --reference=ALL classes <REF> --filter-version="php_5.1.0" --filter
```

Version 3.2.

```
$ phpcompatinfo reference:show --classes <REF> --php=">= 5.1.0"
```

11.2. Print parses results

Where <SOURCE> identify the data source, directly in CompatInfo 2.26, and via the JSON configuration file in version 3.2

Version 2.26.

```
$ phpcompatinfo print --reference=ALL --recursive <SOURCE>
or
$ phpcompatinfo print --reference=ALL --recursive --report summary <SOURCE>
```

Version 3.2.

```
$ phpcompatinfo analyser:run <SOURCE>
or
$ phpcompatinfo analyser:run <SOURCE> summary
```

And with additionals reports :

- extension

Version 2.26.

```
$ phpcompatinfo print --reference=ALL --recursive --report extension <SOURCE>
```

Version 3.2.

```
$ phpcompatinfo analyser:run <SOURCE> extension
```

- namespace

Version 2.26.

```
$ phpcompatinfo print --reference=ALL --recursive --report namespace <SOURCE>
```

Version 3.2.

```
$ phpcompatinfo analyser:run <SOURCE> namespace
```

- trait

Version 2.26.

```
$ phpcompatinfo print --reference=ALL --recursive --report trait <SOURCE>
```

Version 3.2.

```
$ phpcompatinfo analyser:run <SOURCE> trait
```

- interface

Version 2.26.

```
$ phpcompatinfo print --reference=ALL --recursive --report interface <SOURCE>
```

Version 3.2.

```
$ phpcompatinfo analyser:run <SOURCE> interface
```

- class

Version 2.26.

```
$ phpcompatinfo print --reference=ALL --recursive --report class <SOURCE>
```

Version 3.2.

```
$ phpcompatinfo analyser:run <SOURCE> class
```

- function

Version 2.26.

```
$ phpcompatinfo print --reference=ALL --recursive --report function <SOURCE>
```

Version 3.2.

```
$ phpcompatinfo analyser:run <SOURCE> function
```

- constant

Version 2.26.

```
$ phpcompatinfo print --reference=ALL --recursive --report constant <SOURCE>
```

Version 3.2.

```
$ phpcompatinfo analyser:run <SOURCE> constant
```

- global

Version 2.26.

```
$ phpcompatinfo print --reference=ALL --recursive --report global <SOURCE>
```

Feature not provided by version 3.2

- condition

Version 2.26.

```
$ phpcompatinfo print --reference=ALL --recursive --report condition <SOURCE>
```

Version 3.4.

```
$ phpcompatinfo analyser:run <SOURCE> condition
```

- token

Version 2.26.

```
$ phpcompatinfo print --reference=ALL --recursive --report token <SOURCE>
```

Feature not provided by version 3.2

- xml

Version 2.26.

```
$ phpcompatinfo print --reference=ALL --recursive --report xml <SOURCE>
```

Feature not provided by version 3.2

- source

Version 2.26.

```
$ phpcompatinfo -v print --reference=ALL --report source <SOURCE>
```

Version 3.2.

```
$ phpcompatinfo provider:display <SOURCE>
```

11.3. Caching results

- Version 2.26 may cache results to speed-up later analysis.
- Version 3.2 is able (optionally) to cache results only with other SAPI than CLI.
- Version 3.3 is able to cache results on all API including CLI.

Chapter 12. Configuration file

12.1. Global options

- File extensions was restricted by default in both versions to `php`, `inc` and `html`.

In CompatInfo 3.2, the Finder recursively traverse directories, while it's not true in version 2.26

XML configuration 2.26.

```
<?xml version="1.0" encoding="utf-8" ?>
<phpcompatinfo
    recursive="false"
    fileExtensions="php, inc, phtml"
    >

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

JSON configuration 3.2.

```
{
    "source-providers": [
        {
            "in": ". as current",
            "name": "/\\.(php|inc|phtml)$/"
        }
    ],
}
```

- Progress bar

XML configuration 2.26.

```
<?xml version="1.0" encoding="utf-8" ?>
<phpcompatinfo
    consoleProgress="true"
    verbose="false"
    >

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

Use the first verbose level (`-v`) with `phpcompatinfo` while running the `analyser:run` command.

- Caching results

XML configuration 2.26.

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

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

Version 3.2 is able to cache parsing results only with other SAPI than CLI. See the Developer Guide.

12.2. Cache options

XML configuration 2.26.

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

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

</phpcompatinfo>
```

Version 3.2 does not provide yet ability to cache parsing results in CLI mode.

12.3. References options

XML configuration 2.26.

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

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

</phpcompatinfo>
```

Version 3.2 does not provide ability to load reference depending of rules in the configuration file.

All references are either pre-loaded (Prefetch Strategy) or loaded only when detected (AutoDiscover Strategy).

See Chapter 19, *References included*

12.4. PHP settings

XML configuration 2.26.

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

    <php>
        <ini name="memory_limit" value="140M" />
        <ini name="short_open_tag" />
        <ini name="zend.ze1_compatibility_mode" value="false" />
    </php>

</phpcompatinfo>
```

Version 3.2 does not provide ability to change PHP settings at run-time.

12.5. Excluding Files or Elements from parsing

XML configuration 2.26.

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

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

</phpcompatinfo>
```

Version 3.2 does not provide ability to exclude elements (class, trait, …), but you can exclude files or directories with the Finder. See `source-providers` in the JSON configuration file. See Chapter 8, *The Json Configuration File*

12.6. Listeners

XML configuration 2.26.

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

    <listeners>
        <listener class="className" file="/path/to/filename">
            <arguments>
            </arguments>
```

```
        </listener>
    </listeners>

</phpcompatinfo>
```

Version 3.2 provide this feature with the Symfony EventDispatcher component.

See Chapter 15, *Plugins*

12.7. Plugins options

XML configuration 2.26.

```
<?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>
```

Version 3.2 does not provide ability to select a custom References list.

All references are either pre-loaded (Prefetch Strategy) or loaded only when detected (AutoDiscover Strategy).

See Chapter 19, *References included*

Chapter 13. Server API

13.1. Parsing recursive directories

By default,

- `recursive` option is set to `false` in CompatInfo 2.26, while the Finder recursively traverse directories in version 3.2
- `cacheDriver` option is set to `file` in CompatInfo 2.26, while version 3.2 did not added the cache plugin.

Version 2.26.

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

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

$compatinfo = new PHP_CompatInfo($options);
$compatinfo->parse($source);
```

Version 3.2.

```
<?php
require_once 'vendor/autoload.php';

use Bartlett\CompatInfo;

use Bartlett\Reflect\ProviderManager;
use Bartlett\Reflect\Provider\SymfonyFinderProvider;

use Symfony\Component\Finder\Finder;

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

$finder = new Finder();
$finder->files()
    ->name('*.php')
    ->in($dirs);

$provider = new SymfonyFinderProvider($finder);

$pm = new ProviderManager;
$pm->set('dataSourceIdent', $provider);

$compatinfo = new CompatInfo;
$compatinfo->setProviderManager($pm);
$compatinfo->parse();
```

13.2. Using cache feature

Version 2.26.

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

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

$compatinfo = new PHP_CompatInfo($options);
$compatinfo->parse($source);
```

It’s not mandatory to specify `cacheDriver` option what is by default set to false.

Version 3.2.

```
<?php
require_once 'vendor/autoload.php';

use Bartlett\CompatInfo;

use Bartlett\Reflect\ProviderManager;
use Bartlett\Reflect\Provider\SymfonyFinderProvider;
use Bartlett\Reflect\Plugin\Cache\CachePlugin;
use Bartlett\Reflect\Plugin\Cache\DefaultCacheStorage;
use Bartlett\Reflect\Cache\DoctrineCacheAdapter;

use Doctrine\Common\Cache\FilesystemCache;

use Symfony\Component\Finder\Finder;

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

$finder = new Finder();
$finder->files()
    ->name('*.php')
    ->in($dirs);

$provider = new SymfonyFinderProvider($finder);

$pm = new ProviderManager;
$pm->set('dataSourceIdent', $provider);

$backend = new FilesystemCache(sys_get_temp_dir() . '/phpcompatinfo');
$doctrineCache = new DoctrineCacheAdapter($backend);
$cache = new DefaultCacheStorage($doctrineCache);

$compatinfo = new CompatInfo;
$compatinfo->setProviderManager($pm);
$compatinfo->addPlugin( new CachePlugin($cache) );
$compatinfo->parse();
```

13.3. Listeners

While CompatInfo 2.26 audit all events (does not provide ability to filter them, unless by writing a new listener), version 3.2 let you choose and connect a function by event. See ???

Version 2.26.

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

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

$fileListener = new PHP_CompatInfo_Listener_File();

$compatinfo = new PHP_CompatInfo($options);
$compatinfo->attach($fileListener);
$compatinfo->parse($source);
```

Version 3.2.

```
<?php
require_once 'vendor/autoload.php';

use Bartlett\CompatInfo;

use Bartlett\Reflect\ProviderManager;
use Bartlett\Reflect\Provider\SymfonyFinderProvider;

use Symfony\Component\Finder\Finder;

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

$finder = new Finder();
$finder->files()
    ->name('*.php')
    ->in($dirs);

$provider = new SymfonyFinderProvider($finder);

$pm = new ProviderManager;
$pm->set('dataSourceIdent', $provider);

$compatinfo = new CompatInfo;
$compatinfo->setProviderManager($pm);

$compatinfo->getEventDispatcher()->addListener(
    'reflect.progress',
    function (GenericEvent $e) {
        printf(
            'Parsing Data source "%s" in progress ... File "%s"' . PHP_EOL,
            $e['source'],
            $e['file']->getPathname()
        );
```

```
    }
);
$compatinfo->parse();
```

13.4. Exploring parsing results

Version 2.26.

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

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

$compatinfo = new PHP_CompatInfo($options);
$compatinfo->parse($source);

$versions   = $compatinfo->getVersions();
$classes    = $compatinfo->getClasses();
$functions  = $compatinfo->getFunctions();
$extensions = $compatinfo->getExtensions();
```

Version 3.2.

```
<?php
require_once 'vendor/autoload.php';

use Bartlett\CompatInfo;
use Bartlett\CompatInfo\Analyser;

use Bartlett\Reflect\ProviderManager;
use Bartlett\Reflect\Provider\SymfonyFinderProvider;
use Bartlett\Reflect\Plugin\Analyser\AnalyserPlugin;

use Symfony\Component\Finder\Finder;

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

$finder = new Finder();
$finder->files()
    ->name('*.php')
    ->in($dirs);

$provider = new SymfonyFinderProvider($finder);

$sourceId = 'dataSourceIdent';

$pm = new ProviderManager;
$pm->set($sourceId, $provider);

$compatinfo = new CompatInfo;
$compatinfo->setProviderManager($pm);
$compatinfo->addPlugin(
    new AnalyserPlugin(
        array(
```

```
            new Analyser\SummaryAnalyser(),
        )
    )
);
$compatinfo->parse();

$metrics = $compatinfo->getMetrics();

$versions   = $metrics[$sourceId]['sa.versions'];
$classes    = $metrics[$sourceId]['sa.classes'];
$functions  = $metrics[$sourceId]['sa.functions'];
$extensions = $metrics[$sourceId]['sa.extensions'];
```

`sa.` prefix corresponds to class constant METRICS_PREFIX of SummaryAnalyser.

Part IV. Developer Guide

 **API**

CompatInfo comes with a complete API.

See Chapter 14, *API*

 **Plugins**

CompatInfo uses a Symfony EventDispatcher [http://symfony.com/doc/current/components/event_dispatcher/index.html] Component to allow you to easily extend the features list.

See ???

 **Analysers**

CompatInfo uses analysers that implements the Visitor [http://en.wikipedia.org/wiki/Visitor_pattern] pattern in a simple and effective way to make the render of your results truly customizable.

See Chapter 18, *Build your Analysers*

Chapter 14. API

14.1. Data Source Identification

Basic or Complex Strategy to identify the Data Source.

Now, and for the following chapters, we will not mention how you load the classes. Depending of the install strategy you've adopted, Composer or other, don't forget to load your autoloader.

Compare to version 2, CompatInfo 4 offers two simple strategies to identify the data source.

First, is to give the relative or absolute path to file or directory to parse (without limitation).

Second, is to specify options to customize parsing process, to the Symfony Finder [http://symfony.com/doc/current/components/finder.html] Component.

14.1.1. Basic Strategy

With all SAPI, no JSON config file is required (as it was for CompatInfo 3). You have just to give the relative or absolute path to file or directory to parse.

It's also possible to specify any archive (phar, zip, tar, tgz, gz, rar) as file source.

Example with a simple file or directory (absolute path).

```
$ phpcompatinfo analyser:run /absolute/path/to/source
```

Example with a simple file or directory (relative path).

```
$ phpcompatinfo analyser:run ./relative/path/to/source
```

14.1.2. Complex Strategy

Still as it was with CompatInfo 3, you will need to configure your data source in a JSON file.

Syntax is closed to the Symfony Finder Component that is used to limit data source contents to parse.

Example to parse an archive.

```
{
    "source-providers": [
        {
            "in": "phar:///var/dist/owncloud-7.0.2.tar as owncloud7",
            "name": "*.php",
            "exclude": ["3rdparty"]
        }
    ],
    "plugins": [
    ],
    "analysers" : [
    ]
}
```

Do not forget the `phar://` protocol in front of archive identification.

Use alias named here `owncloud7` to identify data source entry in the JSON config file, rather than the full path `phar:///var/dist/owncloud-7.0.2.tar`.

Example to parse a directory.

```
{
    "source-providers": [
        {
            "in": "/home/github/phing/ as phing2",
            "path": ["bin", "classes"],
            "exclude": ["test"],
            "name": "*.php"
        }
    ],
    "plugins": [
    ],
    "analysers" : [
    ]
}
```

Learn more about directives, see Section 8.1, “section Source Providers”

Whatever SAPI you use, all metrics (for each analysers asked) are available at end of parse, in the same format.

With CLI, and CompatInfo source code, to get a compatibility report, you have to invoke the following command :

```
$ phpcompatinfo analyser:run /home/github/php-compat-info/src
```

With others SAPI, use example https://raw.githubusercontent.com/llaville/php-compat-info/4.0/examples/api_analyser_run.php

and you should obtain something like this :

```
Data Source Analysed

Directories                                          11
Files                                                13

Extensions Analysis

    Extension Matches REF      EXT min/Max PHP min/Max
    Core              Core     5.3.0       5.3.0
    PDO       1       PDO      5.1.0       5.1.0
    curl              curl     4.0.2       4.0.2
    intl              intl     2.0.0b1     5.3.7
    libxml            libxml   5.1.0       5.1.0
    openssl           openssl  5.2.0       5.2.0
    pcre              pcre     4.0.0       4.0.0
    spl               spl      5.1.0       5.1.0
    standard          standard 5.3.2       5.3.2
    Total [9]                              5.3.7
```

```
Namespaces Analysis

    Namespace                               Matches REF  EXT min/Max PHP min/Max
    +global                                         Core             4.0.0
    Bartlett\CompatInfo                             user             5.3.0
    Bartlett\CompatInfo\Analyser                    user             5.3.0
    Bartlett\CompatInfo\Api                         user             5.3.2
    Bartlett\CompatInfo\Api\V3                      user             5.3.0
    Bartlett\CompatInfo\Collection                  user             5.3.0
    Bartlett\CompatInfo\Console                     user             5.3.0
    Bartlett\CompatInfo\Console\Formatter           user             5.3.0
    Bartlett\CompatInfo\Output                      user             5.3.0
    Bartlett\CompatInfo\PhpParser                   user             5.3.0
    Bartlett\CompatInfo\Reference                   user             5.3.0
    Bartlett\CompatInfo\Util                        user             5.3.0
    Total [12]                                                       5.3.2

Interfaces Analysis

    Interface                                          Matches REF  EXT min/Max PHP min/Ma
    Bartlett\CompatInfo\Reference\ReferenceInterface 1       user             5.3.0
    Total [1]                                                                 5.3.0

No trait found

Classes Analysis

    Class                                                                Matches REF  EXT m
    Bartlett\CompatInfo\Analyser\CompatibilityAnalyser                           user
    Bartlett\CompatInfo\Api\Reference                                            user
    Bartlett\CompatInfo\Api\V3\Reference                                         user
    Bartlett\CompatInfo\Collection\ReferenceCollection                   1       user
    Bartlett\CompatInfo\Console\Application                                      user
    Bartlett\CompatInfo\Console\Formatter\CompatibilityOutputFormatter           user
    Bartlett\CompatInfo\Environment                                      4       user
    Bartlett\CompatInfo\Output\Reference                                         user
    Bartlett\CompatInfo\PhpParser\ConditionalCodeNodeProcessor           1       user
    Bartlett\CompatInfo\Reference\ExtensionFactory                       2       user
    Bartlett\CompatInfo\Reference\SqliteStorage                          1       user
    Bartlett\CompatInfo\Util\Version                                     5       user
  U Bartlett\Reflect\Analyser\AbstractAnalyser                           1       user
  U Bartlett\Reflect\Api\BaseApi                                         1       user
  U Bartlett\Reflect\Api\V3\Common                                       1       user
  U Bartlett\Reflect\Console\Application                                 1       user
  U Bartlett\Reflect\Console\Formatter\OutputFormatter                   2       user
  U Bartlett\Reflect\PhpParser\NodeProcessorAbstract                     1       user
  U Doctrine\Common\Collections\AbstractLazyCollection                   1       user
  U Doctrine\Common\Collections\ArrayCollection                          1       user
    PDO                                                                  2       PDO  5.1.
  U PhpParser\Node                                                       21      user
  U PhpParser\Node\Name                                                  1       user
    RuntimeException                                                     1       spl  5.1.
  U Symfony\Component\Console\Helper\TableSeparator                      3       user
  U Symfony\Component\Console\Output\OutputInterface                     4       user
    parent                                                               5       Core 5.0.
    self                                                                 8       Core 5.0.
    stdClass                                                             1       Core 4.0.
    Total [29]
```

```
Functions Analysis

    Function                    Matches REF      EXT min/Max PHP min/Max
    array_key_exists            4       standard 4.0.7       4.0.7
    array_map                   1       standard 4.0.6       4.0.6
    array_merge                 2       standard 4.0.0       4.0.0
    array_pop                   10      standard 4.0.0       4.0.0
    array_shift                 2       standard 4.0.0       4.0.0
    array_slice                 1       standard 4.0.0       4.0.0
    array_values                1       standard 4.0.0       4.0.0
    closure-71-73                       user                 5.3.0
    copy                        1       standard 4.0.0       4.0.0
    count                       11      standard 4.0.0       4.0.0
 C  curl_version                1       curl     4.0.2       4.0.2
    defined                     4       Core     4.0.0       4.0.0
    dirname                     3       standard 4.0.0       4.0.0
    each                        2       Core     4.0.0       4.0.0
    explode                     1       standard 4.0.0       4.0.0
    extension_loaded            1       Core     4.0.0       4.0.0
    file_exists                 2       standard 4.0.0       4.0.0
    function_exists             1       Core     4.0.0       4.0.0
    in_array                    7       standard 4.0.0       4.0.0
    is_array                    1       standard 4.0.0       4.0.0
    is_callable                 1       standard 4.0.6       4.0.6
    is_string                   10      standard 4.0.0       4.0.0
    ksort                       3       standard 4.0.0       4.0.0
    mkdir                       1       standard 4.0.0       4.0.0
    phpversion                  2       standard 4.0.0       4.0.0
    preg_match                  2       pcre     4.0.0       4.0.0
    sha1_file                   2       standard 4.3.0       4.3.0
    sprintf                     22      standard 4.0.0       4.0.0
    strcasecmp                  2       Core     4.0.0       4.0.0
    stream_resolve_include_path 1       standard 5.3.2       5.3.2
    strpos                      1       standard 4.0.0       4.0.0
    strtolower                  1       standard 4.0.0       4.0.0
    substr                      2       standard 4.0.0       4.0.0
    sys_get_temp_dir            1       standard 5.2.1       5.2.1
    ucfirst                     5       standard 4.0.0       4.0.0
    version_compare             7       standard 4.0.7       4.0.7
    Total [36]                                               5.3.2

Constants Analysis

    Constant               Matches REF     EXT min/Max PHP min/Max
 C  INTL_ICU_VERSION       2       intl    2.0.0b1     5.3.7
    LIBXML_DOTTED_VERSION  1       libxml  5.1.0       5.1.0
    LIBXML_VERSION         1       libxml  5.1.0       5.1.0
 C  OPENSSL_VERSION_NUMBER 1       openssl 5.2.0       5.2.0
 C  OPENSSL_VERSION_TEXT   1       openssl 5.2.0       5.2.0
    PHP_EOL                5       Core    4.3.10      4.3.10
    PHP_VERSION            4       Core    4.0.0       4.0.0
    __DIR__                1       Core    5.3.0       5.3.0
    false                  12      Core    4.0.0       4.0.0
    null                   13      Core    4.0.0       4.0.0
    true                   8       Core    4.0.0       4.0.0
    Total [11]                                         5.3.0

Conditions Analysis
```

```
    Condition                         Matches REF     EXT min/Max PHP min/Max
    defined(INTL_ICU_VERSION)         2       intl    2.0.0b1     5.3.7
    defined(OPENSSL_VERSION_NUMBER) 1       openssl 5.2.0       5.2.0
    defined(OPENSSL_VERSION_TEXT)   1       openssl 5.2.0       5.2.0
    function_exists(curl_version)   1       curl    4.0.2       4.0.2
    Total [4]                                                     5.3.7

Requires PHP 5.3.2 (min)

Time: 11.58 seconds, Memory: 11.50Mb
```

This is the default render. But, if you want to compare with other SAPI, activate the debug verbose mode (`-vvv`) to get the raw response. You should obtain something like this :

```
Raw response
Array
(
    [files] => Array
        (
            [0] => /home/github/php-compat-info/src/Bartlett/CompatInfo/Reference/Extens
            [1] => /home/github/php-compat-info/src/Bartlett/CompatInfo/Api/V3/Reference
            [2] => /home/github/php-compat-info/src/Bartlett/CompatInfo/Analyser/Compati
            [3] => /home/github/php-compat-info/src/Bartlett/CompatInfo/Api/Reference.ph
            [4] => /home/github/php-compat-info/src/Bartlett/CompatInfo/Collection/Refer
            [5] => /home/github/php-compat-info/src/Bartlett/CompatInfo/Console/Applicat
            [6] => /home/github/php-compat-info/src/Bartlett/CompatInfo/Console/Formatte
            [7] => /home/github/php-compat-info/src/Bartlett/CompatInfo/Environment.php
            [8] => /home/github/php-compat-info/src/Bartlett/CompatInfo/Output/Reference
            [9] => /home/github/php-compat-info/src/Bartlett/CompatInfo/PhpParser/Condit
            [10] => /home/github/php-compat-info/src/Bartlett/CompatInfo/Reference/Refer
            [11] => /home/github/php-compat-info/src/Bartlett/CompatInfo/Reference/Sqlit
            [12] => /home/github/php-compat-info/src/Bartlett/CompatInfo/Util/Version.ph
        )

    [Bartlett\CompatInfo\Analyser\CompatibilityAnalyser] => Array
        (
            [versions] => Array
                (
                    [php.min] => 5.3.2
                    [php.max] =>
                )

            [extensions] => Array
                (
                    ...

                )

            [namespaces] => Array
                (
                    ...

                )

            [interfaces] => Array
                (
                    ...

                )
```

```
            [traits] => Array
                (
                )

            [classes] => Array
                (
                    ...

                )

            [methods] => Array
                (
                    ...

                )

            [functions] => Array
                (
                    ...

                )

            [constants] => Array
                (
                    ...

                )

            [conditions] => Array
                (
                    ...

                )

        )

)


Time: 5.55 seconds, Memory: 11.50Mb
```

- First entry in array is the list of parsed `files`
- Second entry in array is the `compatibility` analyser result

Each analyser as its own data structure and results, but you will always get the fully qualified class name that identify origin of analyser used.

Example with two analysers (`structure` and `compatibility`).

```
$ phpcompatinfo analyser:run /home/github/php-compat-info/src structure compatibility
```

```
Raw response
Array
(
    [files] => Array
        (
            ...
```

```
        )

    [Bartlett\Reflect\Analyser\StructureAnalyser] => Array
        (
            ...
        )

    [Bartlett\CompatInfo\Analyser\CompatibilityAnalyser] => Array
        (
            ...
        )

)
```

Chapter 15. Plugins

15.1. Events

CompatInfo uses a Symfony EventDispatcher [http://symfony.com/doc/current/components/event_dispatcher/index.html] Component to allow you to easily extend the features list.

The EventDispatcher component allow CompatInfo components to communicate with each other by dispatching events and listening to them.

15.1.1. Event Dispatcher

CompatInfo implement interface `Bartlett\Reflect\Event\DispatcherInterface`. You can add event listeners and event subscribers to this object.

listeners	Callable functions that are registered on an event dispatcher for specific events.
subscribers	Classes that tell an event dispatcher what methods to listen to and what functions on the class to invoke when the event is triggered. Event subscribers subscribe event listeners to an event dispatcher.

15.1.2. Getting an EventDispatcher

You can get the EventDispatcher of `Bartlett\Reflect\Event\DispatcherInterface` by calling the `getEventDispatcher()` method.

Here is an example :

```
<?php

use Bartlett\Reflect\Client;

// creates an instance of client
$client = new Client();

// request for a Bartlett\Reflect\Api\Analyser
$api = $client->api('analyser');

$dispatcher = $api->getEventDispatcher();
```

15.1.3. Adding Event Listeners

After you have the event dispatcher, you can register event listeners that listen to specific events.

Example 15.1. Add a listener that will echo out files when they are parsed

```
<?php

use Bartlett\Reflect\Client;

use Symfony\Component\EventDispatcher\GenericEvent;
```

```
// creates an instance of client
$client = new Client();

// request for a Bartlett\Reflect\Api\Analyser
$api = $client->api('analyser');

$dispatcher = $api->getEventDispatcher();

$dispatcher->addListener(
    'reflect.progress',
    function (GenericEvent $e) {
        printf(
            'Parsing Data source "%s" in progress ... File "%s"' . PHP_EOL,
            $e['source'],
            $e['file']->getPathname()
        );
    }
);
```

15.1.4. Event Subscribers

Event subscribers are classes that implement interface `Symfony\Component\EventDispatcher\EventSubscriberInterface`. They are used to register one or more event listeners to methods of the class. Event subscribers tell event dispatcher exactly which events to listen to and what method to invoke on the class.

CompatInfo plugins follow the event subscribers behaviors. Have a look on NotifierPlugin :

```
<?php

use Bartlett\Reflect\Events;

class NotifierPlugin implements PluginInterface, EventSubscriberInterface
{
    public static function getSubscribedEvents()
    {
        $events = array(
            Events::PROGRESS => 'onNotification',
            Events::ERROR    => 'onNotification',
            Events::COMPLETE => 'onNotification',
        );
        return $events;
    }
}
```

This plugin registers event listeners to the `reflect.complete` event of a Reflect parse request.

When the `reflect.complete` event is emitted, the `onNotification` instance method of the plugin is invoked.

15.1.5. Events lifecycle

Event	Action	Informations available
reflect.progess	Before to parse a new file of the data source.	`source` data source identifier or its alias

Event	Action	Informations available
		`file` current file parsed in the data source
reflect.success	After parsing the current file (A cached request will not trigger this event)	`source` data source identifier or its alias `file` current file parsed in the data source `ast` the Abstract Syntax Tree [http://en.wikipedia.org/wiki/Abstract_syntax_tree] result of PHP-Parser [https://github.com/nikic/PHP-Parser]
reflect.error	When PHP Parser raise an error	`source` data source identifier or its alias `file` current file parsed in the data source `error` PHP Parser error message
reflect.complete	When a parse request is over.	`source` data source identifier or its alias

15.2. Register Plugins

In Reflect API 2, and other SAPI than CLI, you have to register a plugin, if you want to use it.

In Reflect API 3, it's no more necessary. All valid plugins defined in the JSON configuration file are automatically registered.

You must define environment variables `BARTLETT_SCAN_DIR` and `BARTLETTRC`, otherwise the JSON config file will not found it.

If you don't want to use any plugins, and de-activated all at once, follow this pattern.

```
<?php

use Bartlett\Reflect\Environment;
use Bartlett\Reflect\Client;

// set default values for BARTLETT_SCAN_DIR
Environment::setScanDir()

// set default value for BARTLETTRC
putenv("BARTLETTRC=phpcompatinfo.json");

// creates an instance of client
$client = new Client();
```

```
// request for a Bartlett\Reflect\Api\Analyser
$api = $client->api('analyser');

// de activate all plugins
$api->activatePlugins(false);

// perform request, on a data source with default analyser (compatibility)
$dataSource = dirname(__DIR__) . '/src';
$analysers  = array('compatibility');

// equivalent to CLI command `phpcompatinfo analyser:run ../src`
$metrics = $api->run($dataSource, $analysers);
```

Chapter 16. Cache Plugin

> This plugin is almost unnecessary if you unload the `xdebug` extension that slows down execution. Learn more on RFC [https://github.com/llaville/php-compat-info/issues/167].

16.1. Register Plugin

If you want to use the Doctrine cache component [https://github.com/doctrine/cache], skip this section.

If you want to use your own version of cache plugin, use following pattern.

```
<?php

namespace YourNamespace;

use Bartlett\Reflect\Plugin\CachePlugin as BaseCachePlugin;

class CachePlugin extends BaseCachePlugin
{
    // all additional code you need
}
```

And the configuration in JSON file :

```
{
    "source-providers": [
    ],
    "plugins": [
        {
            "name": "Cache",
            "class": "YourNamespace\\CachePlugin",
            "options": []
        }
    ],
    "analysers" : [
    ],
    "services": [
    ]
}
```

16.2. Doctrine Adapter

Use one of the most famous caching solution, provided by the Doctrine project.

Use this shortcut version that is strictly equivalent to next in section file cache.

```
{
    "source-providers": [
    ],
    "plugins": [
        {
```

```
            "name": "Cache",
            "class": "Bartlett\\Reflect\\Plugin\\CachePlugin",
            "options": []
        }
    ],
    "analysers" : [
    ],
    "services": [
    ]
}
```

16.3. File cache

Doctrine File backend to store your Reflect results in the local file system.

```
{
    "source-providers": [
    ],
    "plugins": [
        {
            "name": "Cache",
            "class": "Bartlett\\Reflect\\Plugin\\CachePlugin",
            "options": {
                "adapter": "DoctrineCacheAdapter",
                "backend": {
                    "class": "Doctrine\\Common\\Cache\\FilesystemCache",
                    "args": [
                        "%{TEMP}/bartlett/cache"
                    ]
                }
            }
        }
    ],
    "analysers" : [
    ],
    "services": [
    ]
}
```

In the source code above, we use the standard Doctrine File cache provider, and store results in the default system temporary directory (see php sys_get_temp_dir() [http://www.php.net/manual/en/function.sys-get-temp-dir.php] function).

Chapter 17. Log Plugin

17.1. Register Plugin

If you want to use default logger `Bartlett\Reflect\Plugin\Log\DefaultLogger`, skip this section.

If you want to use your own version of log plugin, use following pattern.

```
<?php

namespace YourNamespace;

use Bartlett\Reflect\Plugin\LogPlugin as BaseLogPlugin;

class LogPlugin extends BaseLogPlugin
{
    // all additional code you need
}
```

And the configuration in JSON file :

```
{
    "source-providers": [
    ],
    "plugins": [
        {
            "name": "Logger",
            "class": "YourNamespace\\LogPlugin",
            "options": []
        }
    ],
    "analysers" : [
    ],
    "services": [
    ]
}
```

17.2. Default logger

Use a solution similar to ErrorLogHandler of Monolog project.

```
{
    "source-providers": [
    ],
    "plugins": [
        {
            "name": "Logger",
            "class": "Bartlett\\Reflect\\Plugin\\LogPlugin",
            "options": []
        }
    ],
    "analysers" : [
    ],
    "services": [
```

```
    ]
}
```

It logs records at `Psr\Log\LogLevel::INFO` level or highter, identified by channel name `DefaultLoggerChannel`.

17.3. Using your private logger

Use your own logger, that must be compatible PSR-3 [http://www.php-fig.org/psr/psr-3/].

```
<?php

namespace YourNamespace;

use Psr\Log\AbstractLogger;

class YourLogger extends AbstractLogger
{
    public function log($level, $message, array $context = array())
    {
    }
}
```

And identify it in the JSON config file, as follow

```
{
    "source-providers": [
    ],
    "plugins": [
        {
            "name": "Logger",
            "class": "YourNamespace\\LogPlugin"
        }
    ],
    "analysers" : [
    ],
    "services": [
    ]
}
```

Or even

```
{
    "source-providers": [
    ],
    "plugins": [
        {
            "name": "Logger",
            "class": "Bartlett\\Reflect\\Plugin\\LogPlugin",
            "options": "YourNamespace\\YourLogger"
        }
    ],
    "analysers" : [
    ],
    "services": [
    ]
}
```

See full example at https://raw.githubusercontent.com/llaville/php-compat-info/4.0/examples/api_analyser_run_with_logger.php

17.4. Using Monolog

Use one of the most famous logging solution compatible PSR-3.

If you want to use Monolog with CompatInfo on CLI mode, then you should use a wrapper like this.

```
<?php

namespace YourNamespace;

use Monolog\Logger;
use Monolog\Handler\StreamHandler;

class YourLogger extends Logger
{
    public function __construct($name = 'YourLoggerChannel')
    {
        $stream = new StreamHandler('/var/logs/phpcompatinfo.log');
        parent::__construct($name, array($stream));
    }
}
```

And with JSON config file as follow

```
{
    "source-providers": [
    ],
    "plugins": [
        {
            "name": "Logger",
            "class": "Bartlett\\Reflect\\Plugin\\LogPlugin",
            "options": "YourNamespace\\YourLogger"
        }
    ],
    "analysers" : [
    ],
    "services": [
    ]
}
```

Chapter 18. Build your Analysers

Analysers implements the Visitor [http://en.wikipedia.org/wiki/Visitor_pattern] pattern in a simple and effective way to make the render of your results truly customizable.

18.1. Visitor pattern

Each `Analyser` class must implement interface `Bartlett\Reflect\Visitor\VisitorInterface`.

Abstract visitor is a component of Reflect, and not CompatInfo. Take care of namespace !

```
<?php

namespace Bartlett\Reflect\Visitor;

use Bartlett\Reflect\Model\Visitable;

interface VisitorInterface
{
    public function visit(Visitable $visitable);
}
```

Each element that need to be explored by your analyser should have a visit method accordingly.

- For packages, we need to implement a **visitPackageModel** method.
- For classes, we need to implement a **visitClassModel** method.
- For properties, we need to implement a **visitPropertyModel** method.
- For methods, we need to implement a **visitMethodModel** method.
- For functions, we need to implement a **visitFunctionModel** method.
- For constants, we need to implement a **visitConstantModel** method.
- For includes, we need to implement a **visitIncludeModel** method.
- For dependencies, we need to implement a **visitDependencyModel** method.

Abstract class `Bartlett\Reflect\Visitor\AbstractVisitor`, that implement interface `Bartlett\Reflect\Visitor\VisitorInterface`, holds a basic visitor.

```
<?php

use Bartlett\Reflect\Visitor\AbstractVisitor;

class Analyser extends AbstractVisitor
{
    public function visitPackageModel($package)
```

```
    {
    }

    public function visitClassModel($class)
    {
    }

    public function visitMethodModel($method)
    {
    }

    public function visitPropertyModel($property)
    {
    }

    public function visitFunctionModel($function)
    {
    }

    public function visitConstantModel($constant)
    {
    }

    public function visitIncludeModel($include)
    {
    }

    public function visitDependencyModel($dependency)
    {
    }
}
```

An abstract class `Bartlett\Reflect\Analyser\AbstractAnalyser` that implement all required interfaces may be used to initialize common data in a simple way.

Your analyser became as simple like that:

```
<?php

use Bartlett\Reflect\Analyser\AbstractAnalyser;

class Analyser extends AbstractAnalyser
{
}
```

18.2. Print results

Once you have used visit methods to explore parsing results, you will need to return formatted data ready to be print.

To do so, you should implement the `render()` method of `Bartlett\Reflect\Analyser\AnalyserInterface`. :leveloffset: 0

Chapter 19. References included

Statistics v2

- 2.0.0 support 61 references
- 2.1.0 support 63 references
- 2.2.0 support 65 references
- 2.3.0 support 67 references
- 2.5.0 support 75 references
- 2.8.0 support 80 references
- 2.10.0 support 83 references
- 2.13.0 support 84 references
- 2.15.0 support 86 references
- 2.16.0 support 95 references
- 2.23.0 support 98 references
- 2.25.0 support 99 references
- 2.26.0 support 100 references

Statistics v3

- 3.0.0 support 100 references
- 3.3.0 support 102 references

Statistics v4

- 4.0.0 support 103 references

Reference	CompatInfo
amqp 1.4.0 stable	2.8.0
apc 3.1.13 beta	2.0.0
apcu 4.0.7 beta	2.16.0
bcmath php5	2.0.0
bz2 php5	2.0.0
calendar php5	2.0.0
Core php5	2.0.0
ctype php5	2.0.0
curl php5	2.0.0

Reference	CompatInfo
date php5	2.0.0
dom 20031129 stable	2.0.0
enchant 1.1.0 stable	2.0.0
ereg php5	2.0.0
exif php5	2.5.0
fileinfo 1.0.5 stable	2.0.0
filter 0.11.0 stable	2.0.0
ftp php5	2.0.0
gd php5	2.0.0
gender 1.0.0 stable	2.16.0
geoip 1.1.0 beta	2.8.0
gettext php5	2.0.0
gmp php5	2.0.0
haru 1.0.4 stable	2.16.0
hash php5	2.0.0
htscanner 1.0.1 stable	2.23.0
http 2.1.4 stable	2.16.0
iconv php5	2.0.0
igbinary 1.2.1 stable	2.10.0
imagick 3.1.2 stable	2.10.0
imap php5	2.0.0
inclued 0.1.3 beta	2.8.0
intl php5	2.0.0
jsmin 1.0.0 stable	2.25.0
json 1.2.1 stable	2.0.0
ldap php5	2.2.0
libevent 0.1.0 beta	2.16.0
libxml php5	2.0.0
lzf 1.6.2 stable	2.5.0
mailparse 2.1.6 stable	2.5.0
mbstring php5	2.0.0
mcrypt php5	2.0.0
memcached 2.2.0 stable	2.1.0
memcache 3.0.8 beta	2.1.0
mhash php5	2.0.0

Reference	CompatInfo
mongo 1.5.8 stable	2.8.0
msgpack 0.5.5 beta	2.16.0
mssql php5	2.5.0
mysql 1.0 stable	2.0.0
mysqli 0.1 stable	2.0.0
OAuth 1.2.3 stable	2.2.0
odbc php5	2.10.0
openssl php5	2.0.0
pcntl php5	2.0.0
pcre php5	2.0.0
pdflib 3.0.4 stable	2.23.0
PDO 1.0.4dev stable	2.0.0
pgsql php5	2.0.0
phar 2.0.2 stable	2.0.0
posix php5	2.0.0
pthreads 2.0.10 stable	2.16.0
rar 3.0.2 stable	2.23.0
readline 2.0.1 stable	2.0.0
recode 2.0.1 stable	2.0.0
Reflection php5	2.3.0
riak 1.2.0 stable	2.26.0
session php5	2.0.0
shmop php5	2.0.0
SimpleXML 0.1 stable	2.0.0
snmp php5	2.0.0
soap php5	2.0.0
sockets php5	2.0.0
solr 2.0.0 stable	2.5.0
sphinx 1.3.2 stable	2.5.0
spl 0.2 stable	2.0.0
sqlite3 0.7-dev stable	2.0.0
sqlite 2.0-dev stable	2.0.0
ssh2 0.12 beta	2.0.0
standard php5	2.0.0
stomp 1.0.6 stable	2.16.0

Reference	CompatInfo
svn 1.0.2 stable	2.13.0
sync 1.0.1 stable	3.3.0
sysvmsg php5	2.0.0
sysvsem php5	2.0.0
sysvshm php5	2.0.0
tidy 2.0 stable	2.0.0
tokenizer 0.1 stable	2.0.0
uploadprogress 1.0.3.1 stable	2.16.0
varnish 1.1.1 stable	2.15.0
wddx php5	2.0.0
XCache 3.2.0 stable	2.8.0
xdebug 2.2.6 stable	2.0.0
xhprof 0.9.4 beta	2.5.0
xml php5	2.0.0
xmldiff 1.1.1 stable	4.0.0
xmlreader 0.1 stable	2.0.0
xmlrpc 0.51 stable	2.0.0
xmlwriter 0.1 stable	2.0.0
xsl 0.1 stable	2.0.0
yac 0.9.2 beta	3.3.0
yaml 1.1.1 stable	2.5.0
Zend OPcache 7.0.4-devFE beta	2.15.0
zip 1.12.4 stable	2.3.0
zlib 2.0 stable	2.0.0