ECPHP CAS Lib Documentation

Release 1.0.0

May 30, 2023

Contents

CAS Lib, a standard PHP library for CAS authentication.

The Central Authentication Service (CAS) is an Open-Source single sign-on protocol for the web. Its purpose is to permit a user to access multiple applications while providing their credentials only once. It also allows web applications to authenticate users without gaining access to a user's security credentials, such as a password. The name CAS also refers to a software package that implements this protocol.

For improving the flexibility and in order to maximize it, it is able to authenticate users and leaves the session handling up to the developer.

In order to foster a greater adoption of this library, it has been built with interoperability in mind. It only uses PHP Standards Recommendations interfaces.

- PSR-4 for classes autoloading,
- PSR-6 for caching,
- PSR-7 for HTTP messages (requests, responses),
- PSR-12 for coding standards,
- PSR-15 for HTTP Server Request Handlers,
- PSR-17 for HTTP messages factories,
- PSR-18 for HTTP client.

Therefore, this library is framework agnostic and can be integrated in any PHP project, with any framework.

CHAPTER 1

Requirements

1.1 PHP

PHP >= 7.4 is required for this library.

1.2 PHP Extensions

- json
- libxml
- simplexml

1.3 Packages

In order to get the CAS Lib library running, you will require some dependencies.

To give a maximum freedom to the users using CAS Lib, each required dependencies is a well defined standardized PHP class.

Dependency	PSR	Implementations	Example package
Cache	PSR-6	cache-implementation	symfony/cache
HTTP Message	PSR-7	http-message-implementations	nyholm/psr7
HTTP Factory	PSR-17	http-factory-implementations	loophp/psr17
HTTP Client	PSR-18	http-client-implementations	symfony/http-client

You are free to use any package you want, as long as they are implementing the proper requirement.

CAS Lib only returns standardized HTTP responses, you will need to emit the response back to the client.

You may use custom code for that, but you can also use any of the following packages for this

- zendframework/zend-httphandlerrunner
- http-interop/response-sender

CHAPTER 2

Installation

The easiest way to install it is through Composer

```
composer require ecphp/cas-lib
```

Based on the context this package is used, you might also need to install a package which provides PSR7 implementations.

There are many packages implementing PSR7, you can pick the one you prefer, exemple:

```
composer require nyholm/psr7
```

Next, you'll need an implementation of PSR17. PSR17 provides the required factories for the HTTP protocol. In order to facilitate the customizations, you can either implements your own PSR17 implementation or use loophp/psr17 which provides a default one:

```
composer require loophp/psr17
```

CHAPTER 3

Configuration

```
base_url: https://casserver.herokuapp.com/cas
protocol:
  login:
    path: /login
    default_parameters:
      foo: bar
  serviceValidate:
    path: /p3/serviceValidate
    default_parameters:
      pgtUrl: https://my-app/casProxyCallback
  logout:
    path: /logout
    default_parameters:
      service: https://my-app/homepage
  proxy:
    path: /proxy
    default_parameters:
      foo: bar
  proxyValidate:
    path: /proxyValidate
    default_parameters:
      pgtUrl: https://my-app/casProxyCallback
```

CHAPTER 4

Usage

Apereo already provides a demo CAS server without no proxy authentication mechanism enabled.

In order to test the libraries here, I've setup another CAS server with Proxy authentication enabled this time.

Feel free to use it for your tests.

> **Warning:** If your client application is not hosted on a public server and in HTTPS, this won't work.

Tip: See more on the page *Development*. if you want to have your own local CAS server.

The test login is *casuser*, password is: *Mellon*

4.1 Bare PHP

To get you started with CAS Lib in a simple bare PHP project (*without using any framework*), you can check the following project: drupol/psrcas-client-poc

Test the bare PHP demo application now.

4.2 Symfony

The CAS Lib library can be used in a Symfony project through the package ecphp/cas-bundle

Test the Symfony demo application now.

See the documentation of the ecphp/cas-bundle for more information.

CHAPTER 5

Tests, code quality and code style

Every time changes are introduced into the library, the continuous integration system run and validate the tests.

A PHP quality tool, Grumphp, is used to orchestrate all these tasks at each commit on the local machine, but also on the continuous integration tool in use.

To run the tests locally:

```
composer grumphp
```

CHAPTER 6

Contributing

See the file CONTRIBUTING.md but feel free to contribute to this library by sending Github pull requests.

CHAPTER 7

Development

In order to test efficiently, is to test the library against a real CAS server.

If you're not able to use one, the best is to work with a local CAS server.

If you want to setup your own local CAS server in less than 2 minutes, use the repo crpeck/cas-overlay-docker and you'll have something working really quickly.

Don't forget to setup the HTTPS certificates because the communication between the CAS server and your application MUST be in HTTPS, and I haven't found a way yet to disable this for testing purposes.

If you prefer to use your local machine, there are already some documentation on Github.

7.1 Maintainers

See the MAINTAINERS.txt file.

7.2 Contributors

See the Github insights page.