

# PSR

## PHP SPECIFICATION REQUEST





# Java Community Process

Java Specification Request



# **PHP Working Group**

PHP Specification Request



# **PHP Framework Interoperability Group**

PHP Specification Request



**<http://www.php-fig.org>**

<https://github.com/php-fig/fig-standards>



# Membres



## Standards

**PSR-0** Autoload

**PSR-1** Norme de codage

**PSR-2** Style d'écriture

**PSR-3** LoggerInterface

**PSR-4** Autoloader



**PSR-0**



## PSR-0

\<Nom du Vendor>\(<Espace de noms>\)\*<Nom de la Classe>



## PSR-0

\<Nom du Vendor>\(<Espace de noms>\)\*<Nom de la Classe>

\Prelinker\Fusion\Request\Http



## PSR-0

\<Nom du Vendor>\(<Espace de noms>\)\*<Nom de la Classe>

\Prelinker\Fusion\Request\Http

/vers/projet/vendor/prelinker/Fusion/Request/Http.php



```
http.php — projet

1  <?php
2  namespace \Prelinker\Fusion\Request;
3
4  /**
5   * Créer une requête en fonction des headers
6   * @param array $headers
7   */
8  class Http implements Http_Interface
9  {
10
11     /**
12     * @var String Nom du module
13     */
14     protected $_moduleName = null;
15
16     /**
17     * @var String Nom du contrôleur
18     */
19     protected $_controllerName = null;
20
21     /**
22     * @var array Nom de l'action
23     */
24     protected $ actionName = null;
```

Line: 30 | PHP | Tab Size: 4



http.php — projet

autoload.php — projet

```
1  <?php
2
3  function autoload($className)
4  {
5      $className = ltrim($className, '\\');
6      $fileName  = '';
7      $namespace = '';
8      if ($lastNsPos = strrpos($className, '\\')) {
9          $namespace = substr($className, 0, $lastNsPos);
10         $className = substr($className, $lastNsPos + 1);
11         $fileName  = str_replace('\\', DIRECTORY_SEPARATOR, $namespace) . DIRECTORY_SEPARATOR;
12     }
13     $fileName .= str_replace('_', DIRECTORY_SEPARATOR, $className) . '.php';
14
15     require $fileName;
16 }
17
```

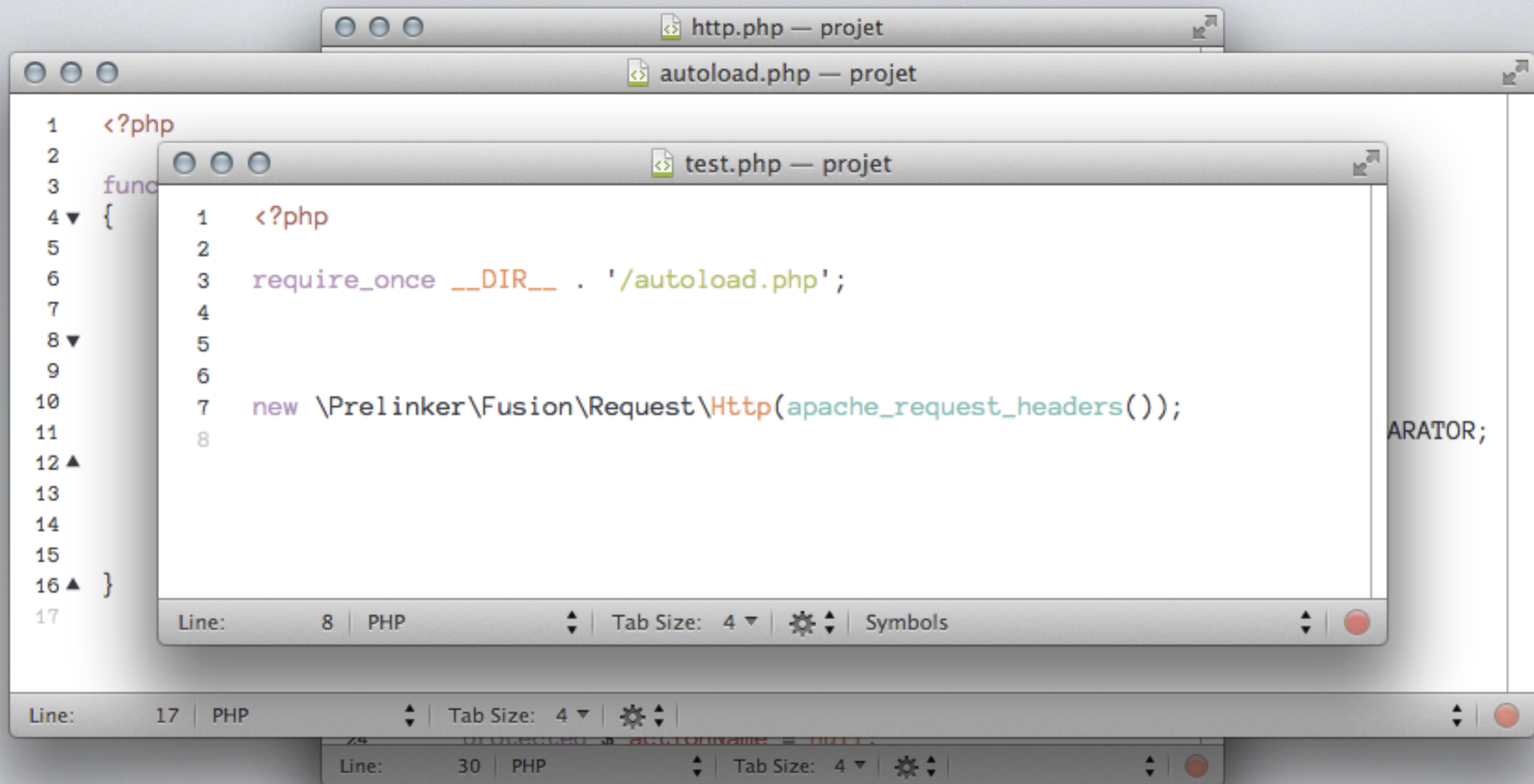
Line: 17 PHP

Tab Size: 4

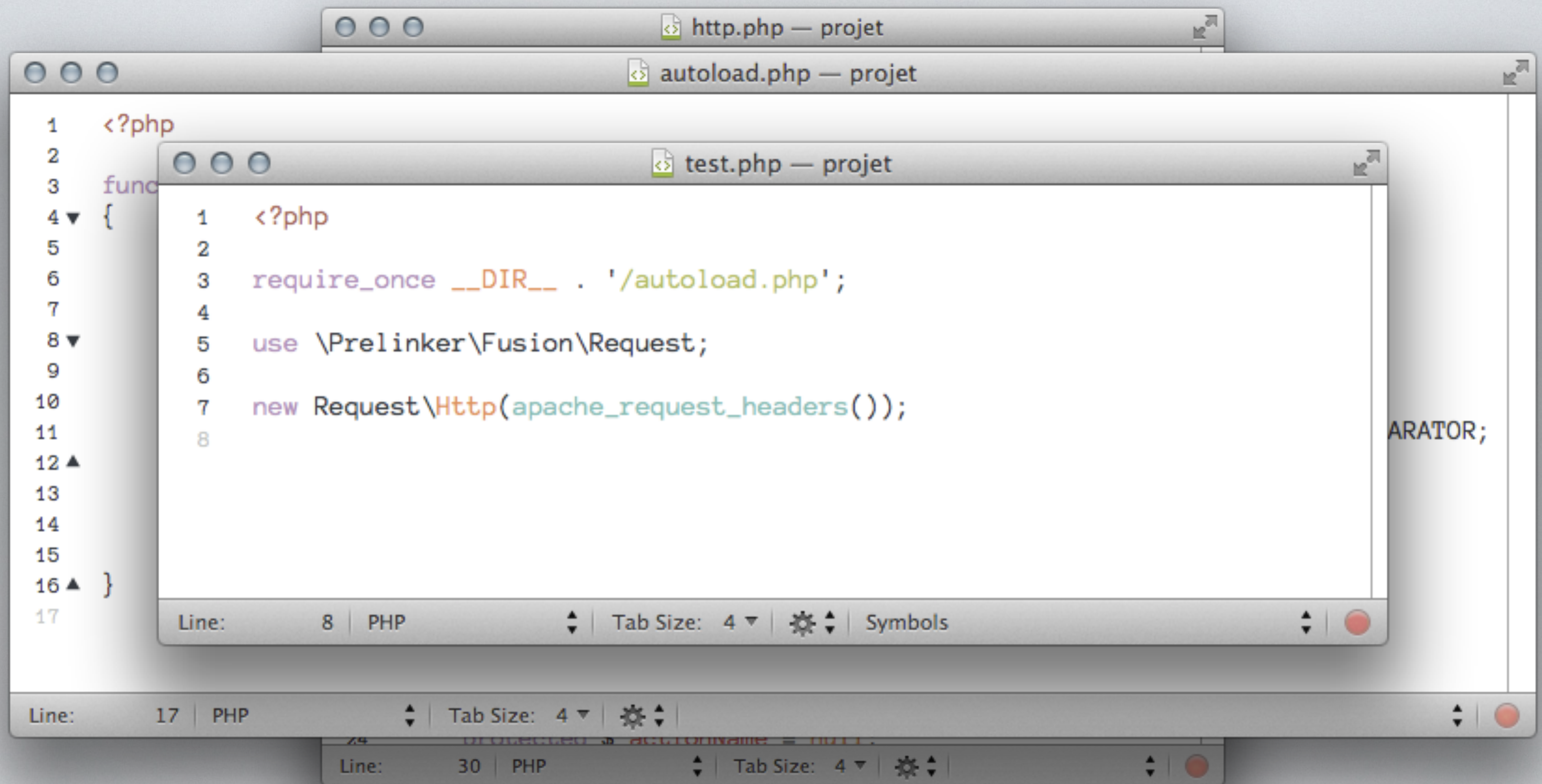
Line: 30 PHP

Tab Size: 4

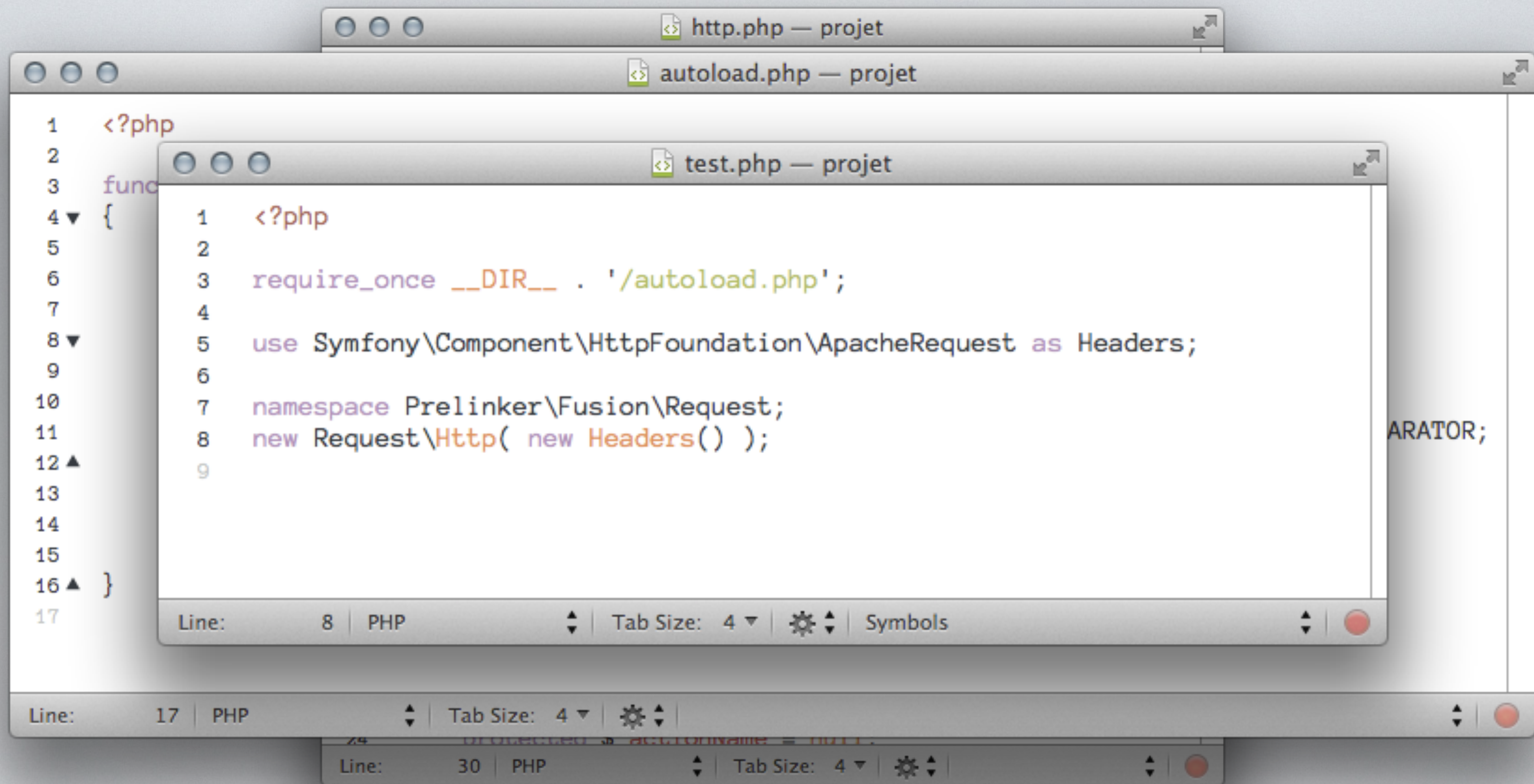














**PSR-I**



# PSR-I

Structure



# **PSR-I**

Structure

Champ d'action



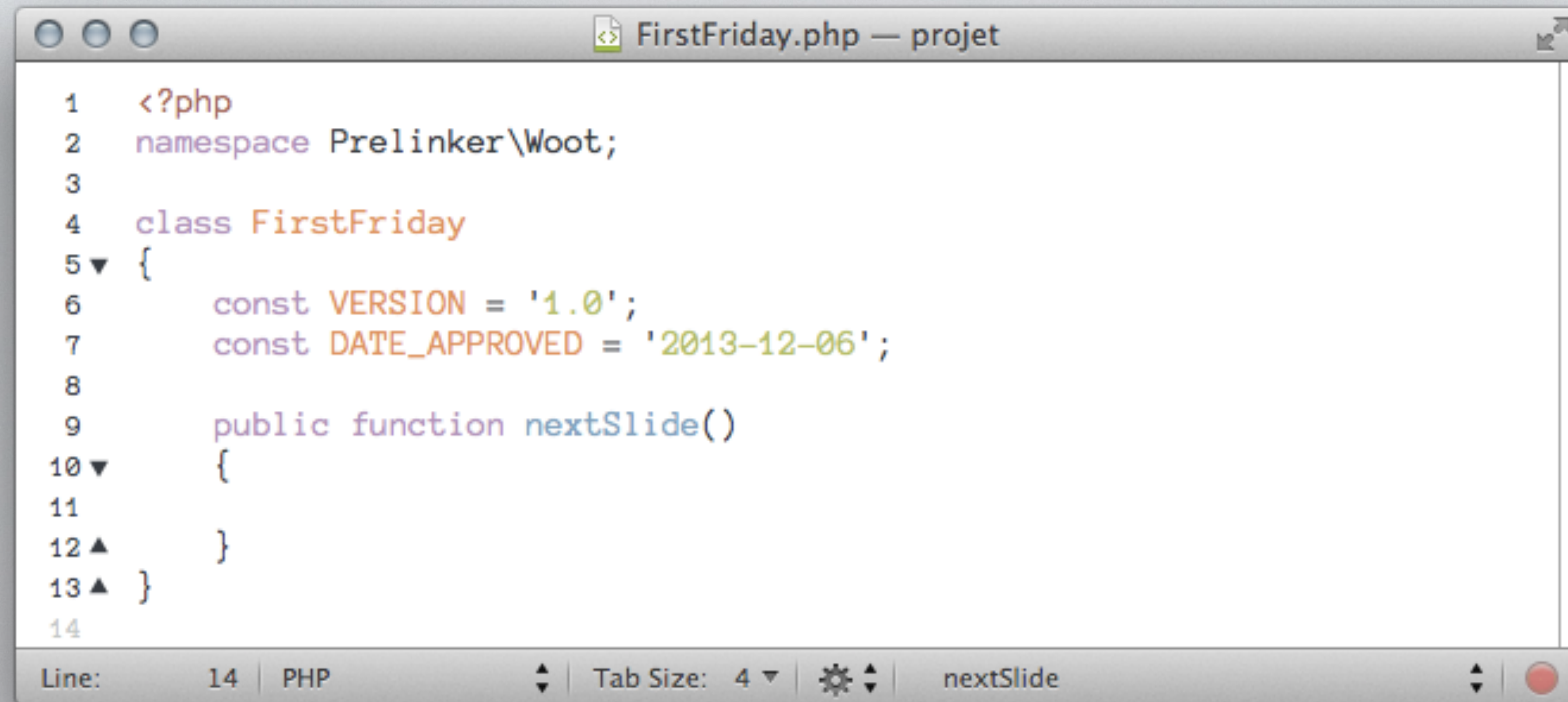
# **PSR-I**

Structure

Champ d'action

Syntaxe





```
1  <?php
2  namespace Prelinker\Woot;
3
4  class FirstFriday
5  {
6      const VERSION = '1.0';
7      const DATE_APPROVED = '2013-12-06';
8
9      public function nextSlide()
10     {
11
12     }
13 }
14
```

Line: 14 | PHP | Tab Size: 4 | nextSlide



**PSR-2**



## PSR-2

Guidelines

Guidelines

Guidelines



**PSR-3**



# PSR-3

Psr\Log\LoggerInterface



# **PSR-3**

Psr\Log\LoggerInterface

RFC 5424



Foo.php — projet

```
1  <?php
2
3  use Psr\Log\LoggerInterface;
4
5  class Foo
6  {
7      private $logger;
8
9      public function __construct(LoggerInterface $logger = null)
10     {
11         $this->logger = $logger;
12     }
13
14     public function doSomething()
15     {
16         if ($this->logger) {
17             $this->logger->info('Doing work');
18         }
19
20         // do something useful
21     }
22 }
23
```

Line: 23 | PHP | Tab Size: 4 | doSomething



**PSR-4**



**PSR-4**

PSR-0



**PSR-4**

PSR-0

Sans conflits



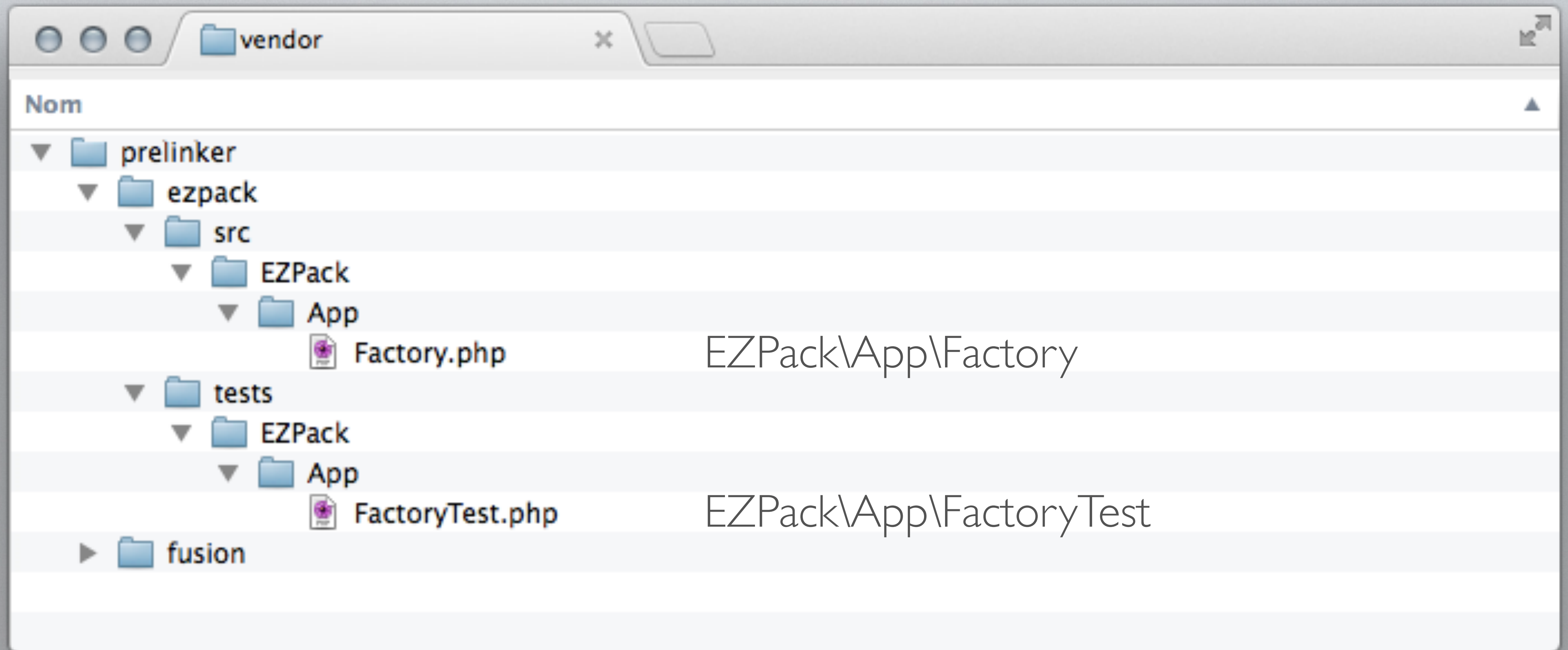
**PSR-4**

PSR-0

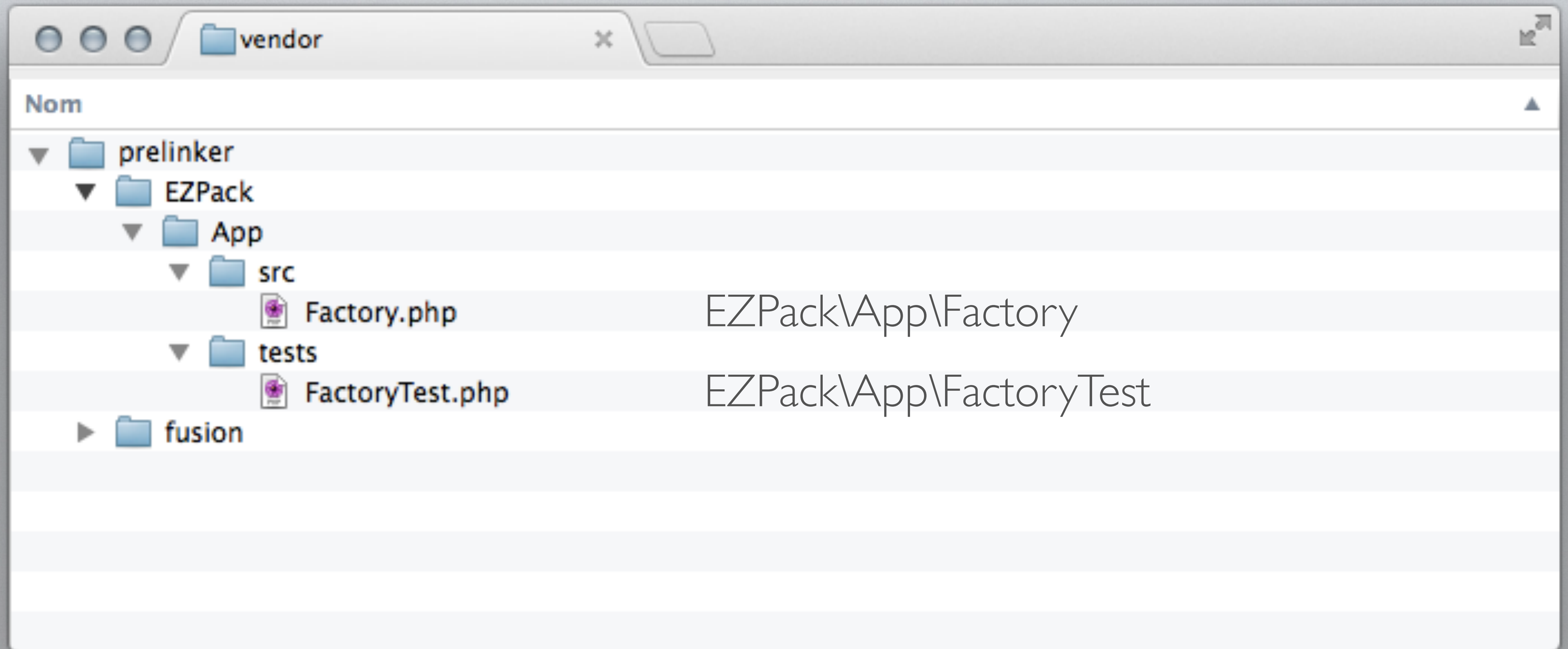
Sans conflits

Sans sur-nommage







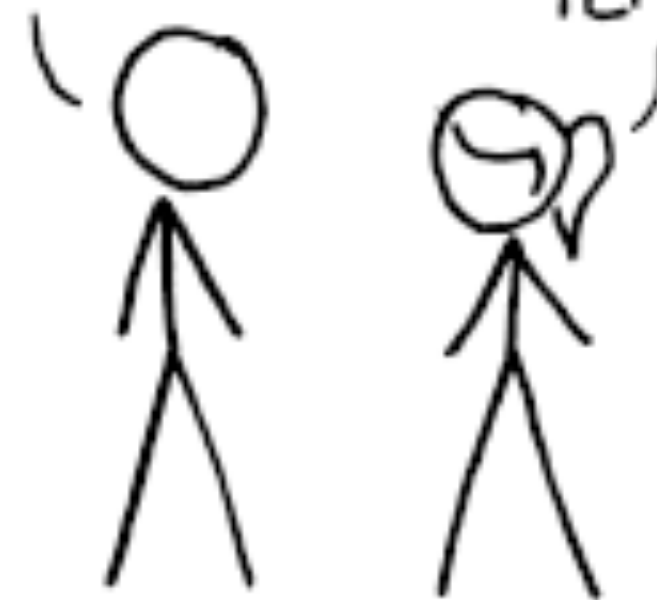




HOW STANDARDS PROLIFERATE:  
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC)

SITUATION:  
THERE ARE  
14 COMPETING  
STANDARDS.

14?! RIDICULOUS!  
WE NEED TO DEVELOP  
ONE UNIVERSAL STANDARD  
THAT COVERS EVERYONE'S  
USE CASES.



SOON:

SITUATION:  
THERE ARE  
15 COMPETING  
STANDARDS.