„Магически” или „вълшебни” методи

Виолета Божикова

„Магически” или „вълшебни” методи?

- Вълшебни методи са методи на класа, които са **достъпни за всички инстанция на един клас, с имена започващи с "__".**
- Например, такива са:

__construct ()

__destruct ()

- за създаване и за унищожаване на обект.

Други магически методи, използвани в PHP 5, са:

- **__set ()**
- **__get ()**
- **__call ()**
- **__toString ()**
- **__sleep ()**
- **__wakeup ()**
- **__isset ()**
- **__unset ()**
- **__autoload ()**
- **__clone ()**

Магическите методи __set() и __get()

- Нека разгледаме следния пример:

```
<?php
class Customer {
public $name;
}
$c = new Customer();
$c->name = "Ivan Ivanov";
// $name is set because its public
$c->email = "ivan@abv.bg";  //???
//assigning ivan@abv.bg to the $email variable.
echo $c->email;
?>
```

Резултат: **ivan@abv.bg**

Коментар:

- За всички, които познават други ОО езици това е **странен код**.
- Странен, защото се опитваме да пишем в **недефинирано свойство**, в случая – свойство email на обект $c. Това в други езици е груба грешка. Но тук всичко е наред. Php е слабо типизиран език, тоест ние може да създаваме недефинирани в класа свойства динамично и да извличаме стойности на такива недефинирани свойства.
- **Това може да е опасно**. Нарушава се капсулацията, всеки може да си създава свойства. Тази функционалност е полезна, ако все пак се контролира.

Решението: __set, __get

- Да използваме магически методи __set, __get. Магическите методи въобще са обикновени методи, но имат определен подпис и се извикват само, ако са дефинирани в класа. Тогава, когато види, че някой се опитва да пише в недефинирано свойство, PHP извиква магически метод __set. В него може да се вгради сложна логика, да има проверка за коректност и т.н. тоест серия от if-ове...чрез които програмистът решава какво ще правим с недефинираните в класа свойства, дали да ги създава въобще или изобщо да ги игнорираме, тоест магическите методи, в случая __set, __get са механизъм за контрол.
- За да контролираме писането в недефинирано свойство, то може да извикаме set – с 2 параметъра name (име) – value (стойност). Обратно с get, може да извличаме стойност на недефинирано свойство – с един параметър.

Решението: __set, __get

```
class Customer
{
public $name;
public function __get($field) {
        if($field == 'name') { return $this->name;   }  }
public function __set($field, $value)
       {  if($field == 'name') {  $this->name = $value;    }  }
}
$c = new Customer();$c->__set("name","Ivan Ivanov");
//$c->name = "Ivan Ivanov"; // $name is set because its public
echo "<br>"; print "It's name is ".($c->__get("name"));
$c->__set("email","ivan@abv.bg");
echo "<br>"; echo "<br>The email of ". $c->name." is ".($c
//echo "<br>"; echo $c->email;
$c->age=23;
echo "<br>The age is---".$c->age;
$c->__set("grade", 4.20);
echo "<br>"; print "It's age is ".($c->__get("age"));
echo "<br>"; print "It's grade is ".$c->grade;
```

It's name is Ivan Ivanov

The email of Ivan Ivanov is
The age is---
It's age is
It's grade is

Магически метод __toString()

От PHP 5.2.0 при преобразуването на обекти в низ без метода __toString ще се генерира фатална грешка!

```
<?php
class Customer
{private $firstName, $lastName, $email;
 public function __construct($firstName, $lastName, $email)
{$this->firstName = $firstName; $this->lastName = $lastName;
$this->email = $email; }
 public function __toString()
{return "The customer $this->firstName $this->lastName"." has e-mail address
    ". $this->email; }
}
$c = new Customer("Ivan","Ivanov","ivo@abv.bg");
echo $c;
?>  //The customer Ivan Ivanov has e-mail address ivo@abv.bg
```

Magic Methods – __isset() and __unset()

- Тези методи __isset() и __unset() се извикват автоматично когато се извикват isset() и unset() за не-декларирани членове на класа.
- Магическият метод __isset() получава един аргумент – стойността на тестваната променлива, и връща 1 (true), ако променливата е установена .
- Магически метод __unset() получава един аргумент – името на променливата, чиято стойност е нужно да се изчисти.

Magic Methods – __isset() and __unset()

```
<?php
class Customer
{
private $data = array();
public function __set($dt, $vl) {$this->data[$dt] = $vl;}
public function __get($dt) {return $this->data[$dt];}
 public function __isset($dt) {return isset($this->data[$dt]);}
 public function __unset($dt) { unset($this->data[$dt]);}
}
$c = new Customer();
$c->__set("name","ivan@abv.bg");
echo "The result of isset is ".isset($c->name)";
unset($c->name);
echo "After unset, the result of isset is ".isset($c->name);
?>
```

//The result of isset is 1 After unset, the result of isset is

magic method __call()

Магически метод **__call()** има 2 аргумента: името на не-деклариран метод, извикан от програмата и втори – един масив, който съдържа списък на параметрите на не-декларирания метод.

(Без **__call()**, недекларирани методи не могат да се викат).

```
<?php
class Customer {
    public function __call($n, $m) {
        var_dump($n);    echo "<br>";
        var_dump($m);   echo "<br>";
    }
}
$c = new Customer();
$c->My_Method("Name","email"); echo "<br>";
?>
```

Извежда:

```
string(9) "My_Method"
array(2) { [0]=> string(4) "Name" [1]=> string(5) "email" }
```

__invoke

- Методът __invoke се извиква, когато даден скрипт направи опит да извика обект, както се извиква функция.

```
<?php
class Customer {
    function __invoke($name) {
        var_dump($name);  }
}
$c = new Customer();
$c("Ivan Ivanov");
?>
```

Резултат:

string(11) "Ivan Ivanov"