

Запись на курсы по HTML, CSS, JavaScript, PHP, фреймворкам и CMS,  
а также: помощь в поиске работы и заказов, стажировка на реальных проектах→

урок 9 из 107

[Верстка](#) [JavaScript](#) [PHP](#) [NodeJs](#) [Vue](#) [React](#) [Laravel](#) [WordPress](#) [AJAX](#) [Парсинг](#)[Бесплатные курсы по React для новичков. Начало 4-го ноября→](#) [Конкурс CSS картинок. Тема: Хэллоуин. Призовой фонд: 100\\$. Подробности→](#)

# Свойства только для чтения

Сейчас мы с вами сделаем так, чтобы в объекте какое-то свойство было доступно только для чтения, но не для записи (*англ* read-only).

Это делается следующим образом: для такого свойства нужно сделать геттер, но не делать сеттер.

В этом случае свойство можно будет прочитать с помощью геттера, но нельзя будет записать, так как сеттер отсутствует.

При этом изначальное значение свойства будет задаваться в конструкторе при создании объекта.

Давайте попробуем реализовать описанное.

Пусть у нас дан вот такой класс **User**:

```
1  <?php
2      class User
3      {
4          private $name;
5          private $age;
6      }
7  ?>
```

Давайте сделаем так, чтобы свойство **name** было доступно только для чтения, а свойство **age** - и для чтения и для записи. Для этого свойству **name** сделаем только геттер, а свойству **age** - и геттер и сеттер:

```
1  <?php
2      class User
3      {
4          private $name;
5          private $age;
6
7          // Геттер для имени:
8          public function getName()
9          {
10             return $this->name;
11          }
12
13         // Геттер для возраста:
14         public function getAge()
15         {
16             return $this->age;
17         }
18
19         // Сеттер для возраста:
20         public function setAge($age)
21         {
22             $this->age = $age;
23         }
24     }
25 ?>
```

Давайте теперь добавим конструктор объекта, в котором будем задавать начальные значения наших свойств:

```
1  <?php
2      class User
3      {
4          private $name;
5          private $age;
6
```

```
7 // Конструктор объекта:
8 public function __construct($name, $age)
9 {
10     $this->name = $name;
11     $this->age = $age;
12 }
13
14 // Геттер для имени:
15 public function getName()
16 {
17     return $this->name;
18 }
19
20 // Геттер для возраста:
21 public function getAge()
22 {
23     return $this->age;
24 }
25
26 // Сеттер для возраста:
27 public function setAge($age)
28 {
29     $this->age = $age;
30 }
31 }
32 ?>
```

Все - наша задача решена, убедимся в этом:

```
1 <?php
2 $user = new User('Коля', 25); // создаем объект с начальными данными
3
4 // Имя можно только читать, но нельзя поменять:
5 echo $user->getName(); // выведет 'Коля'
6
7 // Возраст можно и читать, и менять:
8 echo $user->getAge(); // выведет 25
9 echo $user->setAge(30); // установим возраст в значение 30
10 echo $user->getAge(); // выведет 30
11 ?>
```

## Задача 9.1

Сделайте класс **Employee**, в котором будут следующие свойства: **name** (имя), **surname** (фамилия) и **salary** (зарплата).

## Задача 9.2

Сделайте так, чтобы свойства **name** и **surname** были доступны только для чтения, а свойство **salary** - и для чтения, и для записи.

