

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

урок 7 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: Хэллоун. Призовой фонд: 100\\$. Подробности→](#)

Конструктор объекта

Давайте рассмотрим следующий код:

```
1  <?php
2  // Класс с публичными свойствами name и age:
3  class User
4  {
5      public $name;
6      public $age;
7  }
8
9  // Создаем объект класса:
10 $user = new User;
11
12 // Записываем данные:
13 $user->name = 'Коля';
14 $user->age = 25;
15
16 // Прочитываем данные:
17 echo $user->name; // выведет 'Коля'
18 echo $user->age; // выведет 25
19 ?>
```

В данном коде не очень удобно то, легко можно забыть записать данные в какое-нибудь свойство объекта, особенно если этих свойств много.

Было бы удобно этот код:

```
1  <?php
2  // Создаем объект класса:
3  $user = new User;
4
5  // Записываем данные:
6  $user->name = 'Коля';
7  $user->age = 25;
8  ?>
```

Заменить на вот этот:

```
1  <?php
2  $user = new User('Коля', 25); // создадим обь
   ект, сразу заполнив его данными
3  ?>
```

То есть сделать так, чтобы поля объекта заполнялись при его создании - в этом случае мы никак не сможем забыть задать значения этих полей.

Для решения проблемы нам поможет так называемый *метод-конструктор* с названием `__construct` (в начале два символа подчеркивания).

Суть в следующем - если в коде класса существует метод с таким названием - он будет вызываться в момент создания объекта, смотрите пример:

```
1  <?php
2  class User
3  {
4      public $name;
```



```
5     public $age;
6
7     // Конструктор объекта:
8     public function __construct()
9     {
10         echo '!!!';
11     }
12 }
13
14 $user = new User; // выведет '!!!'
15 ?>
```

Конструктор в общем-то такой же метод, как и все остальные и может принимать параметры, смотрите на примере:

```
1 <?php
2 class User
3 {
4     public $name;
5     public $age;
6
7     public function __construct($var1, $var2)
8     {
9         echo $var1 + $var2; // найдем сумму параметров
10    }
11 }
12
13 $user = new User(1, 2); // выведет 3
14 ?>
```

Итак, давайте переделаем наш код, применив конструктор:

```
1 <?php
2 class User
3 {
4     public $name;
5     public $age;
6
7     // Конструктор объекта:
8     public function __construct($name, $age)
9     {
10         $this->name = $name; // запишем данные в с
            войство name
11         $this->age = $age; // запишем данные в с
            войство age
12    }
13 }
14
15 $user = new User('Коля', 25); // создадим объ
    ект, сразу заполнив его данными
16
17 echo $user->name; // выведет 'Коля'
18
19 echo $user->age; // выведет 25
20 ?>
```

Задача 7.1

Сделайте класс **Employee**, в котором будут следующие публичные свойства - **name** (имя), **age** (возраст), **salary** (зарплата). Сделайте так, чтобы эти свойства заполнялись в методе **__construct** при создании объекта.

Задача 7.2

Создайте объект класса **Employee** с именем 'Вася', возрастом 25, зарплатой 1000.



Задача 7.3

Создайте объект класса **Employee** с именем 'Петя', возрастом 30, зарплатой 2000.

Задача 7.4

Выведите на экран сумму зарплат Васи и Пети.

