

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

урок 67 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→](#)

Окончательный вариант класса Tag

В данном уроке я приведу окончательный вариант класса **Tag**, который мы и будем использовать в дальнейшем в следующих уроках.

Так как код класса достаточно большой и в нем много методов, было бы удобно сделать так, чтобы наш класс реализовывал некоторый интерфейс, в котором в компактном виде были бы прописаны все публичные методы, который должен иметь наш класс.

Добавим в интерфейс все публичные методы, описанные в уроках, а также те методы, которые были описаны в виде задач.

Итак, вот наш интерфейс:

```
1  <?php
2  interface iTag
3  {
4      // Геттер имени:
5      public function getName();
6
7      // Геттер текста:
8      public function getText();
9
10     // Геттер всех атрибутов:
11     public function getAttrs();
12
13     // Геттер одного атрибута по имени:
14     public function getAttr($name);
15
16     // Открывающий тег, текст и закрывающий тег:
17     public function show();
18
19     // Открывающий тег:
20     public function open();
21
22     // Закрывающий тег:
23     public function close();
24
25     // Установка текста:
26     public function setText($text);
27
28     // Установка атрибута:
29     public function setAttr($name, $value = true);
30
31     // Установка атрибутов:
32     public function setAttrs($attrs);
33
34     // Удаление атрибута:
35     public function removeAttr($name);
36
37     // Установка класса:
38     public function addClass($className);
39
40     // Удаление класса:
41     public function removeClass($className);
42 }
43 ?>
```

А вот код нашего класса **Tag**, реализующего интерфейс **iTag**:

```
1  <?php
2  class Tag implements iTag
3  {
4      private $name;
5      private $attrs = [];
6      private $text = '';
7
8      public function __construct($name)
9      {
10         $this->name = $name;
11     }
12
13     public function getName()
14     {
15         return $this->name;
16     }
17
18     public function getText()
19     {
20         return $this->text;
21     }
22
23     public function getAttrs()
24     {
25         return $this->attrs;
26     }
27
28     public function getAttr($name)
29     {
30         if (isset($this->attrs[$name])) {
31             return $this->attrs[$name];
32         } else {
33             return null;
34         }
35     }
36
37     public function show()
38     {
39         return $this->open() . $this->text . $this->close();
40     }
41
42     public function open()
43     {
44         $name = $this->name;
45         $attrsStr = $this->getAttrsStr($this->attrs);
46
47         return "<$name$attrsStr>";
48     }
49
50     public function close()
51     {
52         $name = $this->name;
53         return "</$name>";
54     }
55
56     public function setText($text)
57     {
58         $this->text = $text;
59         return $this;
60     }
61
62     public function setAttr($name, $value = true)
63     {
64         $this->attrs[$name] = $value;
65         return $this;
66     }
```

```
67
68 public function setAttrs($attrs)
69 {
70     foreach ($attrs as $name => $value) {
71         $this->setAttr($name, $value);
72     }
73
74     return $this;
75 }
76
77 public function removeAttr($name)
78 {
79     unset($this->attrs[$name]);
80     return $this;
81 }
82
83 public function addClass($className)
84 {
85     if (isset($this->attrs['class'])) {
86         $classNames = explode(' ', $this->attrs['class']);
87
88         if (!in_array($className, $classNames)) {
89             $classNames[] = $className;
90             $this->attrs['class'] = implode(' ', $classNames);
91         }
92     } else {
93         $this->attrs['class'] = $className;
94     }
95
96     return $this;
97 }
98
99 public function removeClass($className)
100 {
101     if (isset($this->attrs['class'])) {
102         $classNames = explode(' ', $this->attrs['class']);
103
104         if (in_array($className, $classNames)) {
105             $classNames = $this->removeElem($className, $classNames);
106             $this->attrs['class'] = implode(' ', $classNames);
107         }
108     }
109
110     return $this;
111 }
112
113 private function getAttrsStr($attrs)
114 {
115     if (!empty($attrs)) {
116         $result = '';
117
118         foreach ($attrs as $name => $value) {
119             if ($value === true) {
120                 $result .= " $name";
121             } else {
122                 $result .= " $name=\"$value\"";
123             }
124         }
125
126         return $result;
127     } else {
128         return '';
129     }
130 }
```



```
131
132     private function removeElem($elem, $arr)
133     {
134         $key = array_search($elem, $arr);
135         array_splice($arr, $key, 1);
136
137         return $arr;
138     }
139 }
140 ?>
```

