

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

урок 80 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→](#)

Класс FormHelper в ООП на PHP

Давайте теперь реализуем класс FormHelper, с помощью которого можно будет создавать формы. При этом унаследуем этот класс от класса TagHelper, созданного нами в предыдущем уроке.

Вот моя реализация класса FormHelper:

```
1  <?php
2  class FormHelper extends TagHelper
3  {
4      public function openForm($attrs = [])
5      {
6          return $this->open('form', $attrs);
7      }
8
9      public function closeForm()
10     {
11         return $this->close('form');
12     }
13
14     public function input($attrs = [])
15     {
16         if (isset($attrs['name'])) {
17             $name = $attrs['name'];
18
19             if (isset($_REQUEST[$name])) {
20                 $attrs['value'] = $_REQUEST[$name];
21             }
22         }
23
24         return $this->open('input', $attrs);
25     }
26
27     public function password($attrs = [])
28     {
29         $attrs['type'] = 'password';
30         return $this->input($attrs);
31     }
32
33     public function hidden($attrs = [])
34     {
35         $attrs['type'] = 'hidden';
36         return $this->open('input', $attrs);
37     }
38
39     public function submit($attrs = [])
40     {
41         $attrs['type'] = 'submit';
42         return $this->open('input', $attrs);
43     }
44
45     public function checkbox($attrs = [])
46     {
47         $attrs['type'] = 'checkbox';
48         $attrs['value'] = 1;
49
50         if (isset($attrs['name'])) {
51             $name = $attrs['name'];
52
53             if (isset($_REQUEST[$name]) and $_REQUEST[$na
```



```
54 |         me] == 1) {
55 |             $attrs['checked'] = true;
56 |         }
57 |         $hidden = $this->hidden(['name' => $name, 'value' => '0']);
58 |     } else {
59 |         $hidden = '';
60 |     }
61 |
62 |     return $hidden . $this->open('input', $attrs);
63 | }
64 | }
65 | ?>
```

Давайте применим наш класс для создания формы:

```
1 | <?php
2 |     $fh = new FormHelper();
3 |
4 |     echo $fh->openForm(['action' => '', 'method' => 'GET']);
5 |     echo $fh->input(['name' => 'year']);
6 |     echo $fh->checkbox(['name' => 'check']);
7 |     echo $fh->submit();
8 |     echo $fh->closeForm();
9 | ?>
```

В результате получится следующий HTML код:

```
1 | <form action="" method="GET">
2 |     <input name="year">
3 |     <input type="hidden" name="check" value="0">
4 |     <input type="checkbox" name="check" value="1">
5 |     <input type="submit">
6 | </form>
```

Задача 80.1

Изучите и разберите мой код описанного класса. Создайте с его помощью какую-нибудь HTML форму, применив как можно больше методов этого класса.

Задача 80.2

Самостоятельно, не подсматривая в мой код, реализуйте описанный класс.

Задача 80.3

Добавьте в вашу реализацию метод для создания тега `textarea`. Пусть этот тег сохраняет свое значение после отправки формы.



Задача 80.4

Добавьте в вашу реализацию метод для создания выпадающего списка. Пусть метод первым параметром принимает массив атрибутов тега `select`, а вторым - массив для создания тегов `option`. Пусть этот массив содержит ключ `'text'` для текста пункта списка и ключ `'attrs'` для массива атрибутов пункта списка.

Вот пример использования описанного метода:

```
1  <?php
2  $fh = new FormHelper();
3
4  echo $fh->select(
5      ['name' => 'list', 'class' => 'eee'],
6      [
7          ['text' => 'item1', 'attrs' => ['value' => '1']],
8          ['text' => 'item2', 'attrs' => ['value' => '1', 'selected' => true]],
9          ['text' => 'item1', 'attrs' => ['value' => '1', 'class' => 'last']],
10     ],
11 );
13 ?>
```

В результате должен получиться следующий HTML код:

```
1  <select name="list" class="eee">
2      <option value="1">item1</option>
3      <option value="2" selected>item2</option>
4      <option value="3" class="last">item3</option>
5  </form>
```

Сделайте так, чтобы выпадающий список сохранял свое значение после отправки формы.

