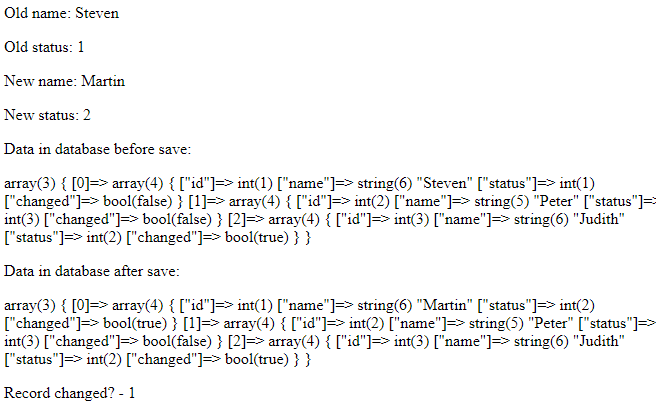
1. Класс Item
2. /\*\*
3. \* Класс Item для тестового задания
4. \* @author Kirill Strelkovskiy <kstr88@mail.ru>
5. \* @version  1.0
6. \* Класс не наследуется
7. \* @method  \_\_construct(int $id) форми
8. \* @method  \_\_get(string $property)
9. \* @method  \_\_set(string $property, $value)
10. \*/
11. **class** Item
12. {
13. /\*\* @access private\*/
14. /\*\* @type int|0 идентификатор\*/
15. **private** int $id = 0;
16. /\*\* @type string|'' наименование\*/
17. **private** string $name = '';
18. /\*\* @type int|0 статус\*/
19. **private** int $status = 0;
20. /\*\* @type bool|False изменение записи\*/
21. **private** bool $changed = False;
22. /\*\* @type array|data симуляция базы данных\*/
23. **private** **array** $db\_simulator = **array**(
24. **array**(
25. 'id'=> 1,
26. 'name'=> 'Steven',
27. 'status'=> 1,
28. 'changed'=> False
29. ),
30. **array**(
31. 'id'=> 2,
32. 'name'=> 'Peter',
33. 'status'=> 3,
34. 'changed'=> False
35. ),
36. **array**(
37. 'id'=> 3,
38. 'name'=> 'Judith',
39. 'status'=> 2,
40. 'changed'=> True
41. )
42. );
44. /\*\*
45. \* @param integer $id
46. \* @return void
47. \*/
48. **function** \_\_construct(int $id)
49. {
50. $this->id = $id;
51. $this->init();
52. }
54. /\*\*
55. \* инициализация параметров класса данными из базы
56. \* @access private
57. \* @return void
58. \*/
59. **private** **function** init()
60. {
61. $data = $this->get\_data\_from\_db();
62. $this->name = $data['name'];
63. $this->status = $data['status'];
64. $this->changed = $data['changed'];
65. }
67. /\*\*
68. \* получение информации из базы данных
69. \* @uses Item::$id
70. \* @return array|$data информация из базы данных по идентификатору
71. \*/
72. **private** **function** get\_data\_from\_db(): **array**
73. {
74. $data = **array**();
75. **foreach** ($this->db\_simulator **as** $value) {
76. **if** ($value['id'] == $this->id){
77. $data = $value;
78. }
79. }
80. **return** $data;
81. }
83. /\*\*
84. \* магический метод, возвращающий
85. \* закрытое свойство экземпляра класса
86. \* @param $property закрытое свойство объекта
87. \* @return Item::$property
88. \*
89. \*/
90. **public** **function** \_\_get($property)
91. {
92. **if** (property\_exists($this, $property)){
93. **return** $this->$property;
94. }
95. }
97. /\*\*
98. \* магический метод, меняющий
99. \* закрытое свойство экземпляра класса
100. \* с использованием проверки значения
101. \* @param $property закрытое свойство объекта
102. \* @param $value новое значение параметра
103. \* @return void
104. \*/
105. **public** **function** \_\_set($property, $value)
106. {
107. **if** (property\_exists($this, $property)){
108. **if** (strlen((string)$value) == 0){
109. **throw** **new** Exception("An empty value for this parameter is not allowed", 1);
110. }
111. **if** (gettype($value) != gettype($this->$property)){
112. **throw** **new** Exception("invalid value type", 1);
113. }
114. $this->$property = $value;
115. }
116. }
118. /\*\*
119. \* метод обновления данных в базе
120. \* @param $property закрытое свойство объекта
121. \* @uses Item::$id
122. \* @return bool возвращает True в случае успешного обновления данных
123. \*/
124. **private** **function** update\_db\_data($property): bool
125. {
126. try{
128. **foreach** ($this->db\_simulator **as** $key => $value) {
129. **if** ($value['id'] == $this->id){
130. $this->db\_simulator[$key][$property] = $this->$property;
131. }
132. }
133. } catch (Exception $e){
134. echo "Error: ". $e;
135. **return** False;
136. }
137. **return** True;
138. }
140. /\*\*
141. \* метод сохранения информации из свойств
142. \* экземпляра класса в базу данных
143. \* @uses Item::$changed
144. \* @return bool возвращает True, если все данные были обновлены в базе
145. \*/
146. **public** **function** save(): bool
147. {
148. **if** (!$this->update\_db\_data("name")) **return** False;
149. **if** (!$this->update\_db\_data("status")) **return** False;
150. $this->changed += True;
151. **if** (!$this->update\_db\_data("changed")) **return** False;
152. **return** True;
153. }
155. /\*\*
156. \* возвращает всю информацию из базы данных
157. \* @return array
158. \*/
159. **public** **function** get\_db\_data(): **array**
160. {
161. **return** $this->db\_simulator;
162. }
163. }

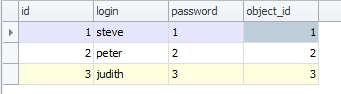
Результат работы с классом:



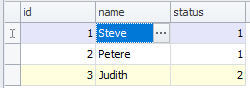
1. Запрос SQL

Таблицы для задания:

* Users
* **CREATE** **TABLE** devj.users (
* id **int**(11) **DEFAULT** NULL,
* login **varchar**(255) **DEFAULT** 'NULL',
* **password** **varchar**(255) **DEFAULT** 'NULL',
* object\_id **int**(11) **DEFAULT** NULL
* )



* Objects
* **CREATE** **TABLE** devj.objects (
* id **int**(11) **DEFAULT** NULL,
* **name** **varchar**(50) **DEFAULT** 'NULL',
* status **int**(11) **DEFAULT** NULL
* )



Запрос:

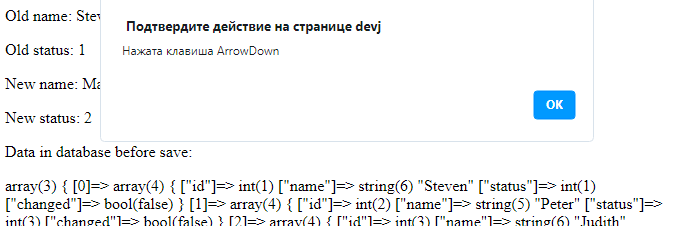
1. **SELECT** users.\* **FROM** users JOIN objects **ON** users.object\_id=objects.id **WHERE** objects.id=1;

Результат запроса:



1. Отслеживание клавиш JQuery
2. <script src="<https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js>" ></script>
3. <script type="text/javascript">
4. $(document).ready(**function**() {
5. $("body").keydown(**function**(e) {
6. **switch** (e.which) {
7. **case** 37:
8. alert("Нажата клавиша " + e.key);
9. **break**;
10. **case** 38:
11. alert("Нажата клавиша " + e.key);
12. **break**;
13. **case** 39:
14. alert("Нажата клавиша " + e.key);
15. **break**;
16. **case** 40:
17. alert("Нажата клавиша " + e.key);
18. **break**;
19. };
20. });
21. });
22. </script>

Результат нажатия клавиши "Вниз"



Продолжение действия после alert:

