Работа с конструкциями if-else, switch-case в PHP

Конструкция if-else

Для того, чтобы напрограммировать что-нибудь полезное, одних переменных далеко не достаточно. Нам нужен механизм, который позволит выполнять определенный код в зависимости от каких-либо условий.

То есть нам нужно иметь возможность спросить у PHP **'Если'**.

Например так: *если эта переменная меньше нуля, то вывести 'отрицательно', иначе (то есть если она больше нуля) вывести 'положительно'.*

В PHP для таких вопросов предназначена **конструкция if**, которая позволяет выполнять **определенный код при выполнении какого-либо условия**:

```
<?php
	if (логическое выражение) {
		Этот код выполниться,
		если логическое выражение верно (то есть равно true)
	}
	else
	{
		Этот код выполнится,
		если логическое выражение неверно (то есть равно false)
	}
?>
```

Обратите внимание на то, что **блок else не обязателен**.

Логическое выражение представляет собой тот самый вопрос, который мы хотим задать PHP. Например, чтобы спросить *'переменная **$a** больше нуля'* мы напишем так: **$a > 0**.

Примеры работы:

```
<?php
	$a = 3;

	/*
		Если переменная $a больше нуля, то выведи 'верно',
		иначе (если меньше или равна нулю) выведи 'неверно'
	*/
	if ($a > 0) {echo 'Верно!';} else {echo 'Неверно!';} //выведет 'Верно!'
?>
<?php
	$a = -3;

	/*
		Если переменная $a больше или равна нулю, то выведи 'верно',
		иначе (если меньше нуля) выведи 'неверно'
	*/
	if ($a >= 0) {echo 'Верно!';} else {echo 'Неверно!';} //выведет 'Неверно!'
?>
```

Сокращенный синтаксис

В случае, если в фигурный скобках **if** или **else** будет только одно выражение, можно эти фигурные скобки не писать:

```php
<?php
	//Полный вариант:
	if ($a == 0) {echo 'Верно!';} else {echo 'Неверно!';}

	//Уберем скобки после if:
	if ($a == 0) echo 'Верно!'; else {echo 'Неверно!';}

	//Уберем скобки после else:
	if ($a == 0) {echo 'Верно!';} else echo 'Неверно!';

	/*
		Уберем скобки и после if, и после else
		(обратите внимание на точку с запятой - она осталась):
	*/
	if ($a == 0) echo 'Верно!'; else echo 'Неверно!';
?>
```

Равенство по значению и типу

Для того, чтобы сравнить на **равенство** следует использовать оператор двойное равно **==**, а не одиночное **=**, как можно было подумать.

Почему так? Потому что одиночное равно зарезервировано за присваиванием. Смотрите пример:

```php
<?php
	$a = 0;

	/*
		Если переменная $a равна нулю, то выведи 'верно',
		иначе (если не равна нулю) выведи 'неверно'
	*/
	if ($a == 0) echo 'Верно!'; else echo 'Неверно!'; //выведет 'Верно!'
?>
```

А следующий пример работать будет **не так**, как мы думаем:

```php
<?php
	$a = 0;

	/*
		Мы думаем оно работает так:
			если переменная $a равна нулю, то выведи 'верно',
			иначе (если не равна нулю) выведи 'неверно'.

		На самом деле оно работает так:
			переменной $a присвоить 1,
			если удалось присвоить - то выведи 'верно',
			иначе (если не удалось присвоить) выведи 'неверно'.
	*/
```

```
        if ($a = 1) echo 'Верно!'; else echo 'Неверно!'; //всегда будет выводить 'Верно!'
?>
```

Кроме оператора == существует еще и оператор ===. Их различие в том, что === сравнивает не только по значению, но и **по типу**, а == сравнивает только по значению.

Чтобы полностью разобраться в этом, внимательно изучите примеры:

```
<?php
        $a = '0'; //переменная $a представляет собой строку, а не число 0
        if ($a == 0) echo 'Верно!'; else echo 'Неверно!';

        /*
                Выведет 'Верно!', так как проверяется только значение, но не тип.
                Поэтому '0' равен 0.
        */
?>
<?php
        $a = '0'; //переменная $a представляет собой строку, а не число 0
        if ($a === 0) echo 'Верно!'; else echo 'Неверно!';

        /*
                Выведет 'Неверно!', так как строка '0'
                не равна числу 0 при сравнении по типу.
        */
?>
```

Не равно

Для того, чтобы спросить **'не равно'**, существует операторы != и !==. Первый игнорирует различие в типах, а второй - нет.

```
<?php
        $a = 0;

        /*
                Если переменная $a НЕ равна нулю, то выведи 'верно',
                иначе (если не равна нулю) выведи 'неверно'
        */
        if ($a != 0) echo 'Верно!'; else echo 'Неверно!'; //выведет 'Верно!', так как $a равно 0
?>
<?php
        $a = 1;

        /*
                Если переменная $a НЕ равна нулю, то выведи 'верно',
                иначе (если не равна нулю) выведи 'неверно'
        */
```

```php
        if ($a != 0) echo 'Верно!'; else echo 'Неверно!'; //выведет 'Неверно!', так как $a равно 0
?>
<?php
        $a = '0';

        /*
                Если переменная $a НЕ равна нулю, то выведи 'верно',
                иначе (если не равна нулю) выведи 'неверно'
        */
        if ($a != 0) echo 'Верно!'; else echo 'Неверно!';

        /*
                Выведет 'Неверно!', так как $a равно '0',
                а различие в типах игнорируется.
        */
?>
<?php
        $a = '0';

        /*
                Если переменная $a НЕ равна нулю, то выведи 'верно',
                иначе (если не равна нулю) выведи 'неверно'
        */
        if ($a !== 0) echo 'Верно!'; else echo 'Неверно!';

        /*
                Выведет 'Верно!', так как $a равно '0',
                а это не равно 0 при сравнении по типу.
        */
?>
```

Все операции сравнения

Возможные операции сравнения, которые можно использовать внутри **if**:

$a == $b	**$a** равно **$b**
$a === $b	**$a** равно **$b** и они одинаковы по типу
$a != $b	**$a** не равно **$b**
$a !== $b	**$a** не равно **$b** или **$a** равно **$b**, но ни разные по типу
$a < $b	**$a** меньше **$b**
$a > $b	**$a** больше **$b**

$a <= $b	**$a** меньше или равно **$b**
$a >= $b	**$a** больше или равно **$b**

Функция empty

Зачастую в веб-программировании возникает следующая проблема: необходимо проверить переменную на *пустоту*.

Переменная будет **пустая**, если она равна нулю, '' (пустой строке), **false** или **null** (то есть не определена ранее).

Проверка на пустоту выполняется с помощью функции **empty()**:

```
<?php
	$a = null;
	//Если $a пустое, то...
	if (empty($a)) echo 'Верно!'; else echo 'Неверно!'; //выведет 'Верно!'
?>
<?php
	$a = 0;
	//Если $a пустое, то...
	if (empty($a)) echo 'Верно!'; else echo 'Неверно!'; //выведет 'Верно!'
?>
<?php
	$a = '';
	//Если $a пустое, то...
	if (empty($a)) echo 'Верно!'; else echo 'Неверно!'; //выведет 'Верно!'
?>
<?php
	$a = 'hi';
	//Если $a пустое, то...
	if (empty($a)) echo 'Верно!'; else echo 'Неверно!'; //выведет 'Неверно!', так как $a не пустая
?>
```

Чаще возникает обратная задача - проверка на то, что переменная является **НЕ пустой**. Отрицание НЕ можно сделать с помощью оператора '!':

```
<?php
	$a = null;
	//Если $a Непустое, то...
	if (!empty($a)) echo 'Верно!'; else echo 'Неверно!'; //выведет 'Неверно!', так как $a пустое
?>
```

Функция isset

Аналогом **empty** является функция **isset**. Она проверяет существует ли переменная (то есть то, что она не равна **null**). Смотрите примеры:

```
<?php
	$a = 'hello';

	//Если $a существует, то...
	if (isset($a)) echo 'Верно!'; else echo 'Неверно!';
```

```php
        /*
                Выведет 'Верно!', так как $a существует.
        */
?>
<?php
        $a = 0;

        //Если $a существует, то...
        if (isset($a)) echo 'Верно!'; else echo 'Неверно!';

        /*
                Выведет 'Верно!', так как $a существует.
        */
?>
<?php
        /*
                Пусть переменную $a вообще не определяли выше в коде
                (это все равно, что присвоить ей null).

                Если $a существует, то...
        */
        if (isset($a)) echo 'Верно!'; else echo 'Неверно!'; //выведет 'Неверно!'
?>
```

Аналогично проверятся на **НЕ существование** (через **!isset**):

```php
<?php
        $a = 3;

        //Если $a НЕ существует то...
        if (!isset($a)) echo 'Верно!'; else echo 'Неверно!';

        /*
                Выведет 'Неверно!', так как $a существует.
        */
?>
```

Несколько условий сразу: or и and

Иногда может быть нужно составить какое-то сложное условие, например, пользователь вводит месяц своего рождения и вам нужно проверить, что введенное число *больше или равно 1 и меньше либо равно 12* (так как в году 12 месяцев).

Для этого существуют операторы **and** (логическое И) и **or** (логическое ИЛИ).

```php
<?php
        $a = 3;
        $b = -3;
        //Если $a больше нуля и $b одновременно меньше нуля то...
```

```
        if ($a > 0 and $b < 0) echo 'Верно!'; else echo 'Неверно!'; //выведет
'Верно!'

        $a = 3;
        //Если $a больше или равно 1 и меньше или равно 12 то...
        if ($a >= 1 and $a <= 12) echo 'Верно!'; else echo 'Неверно!'; //выведет
'Верно!'

        $a = -3;
        $b = -3;
        /*
                Если $a больше нуля ИЛИ $b меньше нуля - хотя бы один из них, то...
                выведет 'Верно!', так как хотя $a и не больше нуля,
                но одно из условий -  $b < 0 - выполнится!
        */
        if ($a > 0 or $b < 0) echo 'Верно!'; else echo 'Неверно!';
?>
```

Вместо **and** можно писать **&&**, а вместо **or** - **||**.

Работа с логическими переменными

Многие функции PHP в результате своей работы возвращают либо **true** (истина), либо **false** (ложь). Эти значения довольно удобны при своей работе, но новичкам бывает довольно сложно понять их.

Представим себе, что переменная **$a** равна **true**. В таком случае конструкцию **if** можно записать так:

```
<?php
        $a = true;

        //Если $a равно true, то...
        if ($a == true) echo 'Верно!'; else echo 'Неверно!';

        /*
                Выведет 'Верно!', так как $a равно true.
        */
?>
```

Так как такие сравнения довольно распространены в PHP, то существует специальный прием, облегчающий работу (но не понимание, к сожалению).

Прием такой: конструкцию **$a == true** можно заменить на более простую: вместо **if ($a == true)**написать **if ($a)** и она будет работать аналогично.

Следует пользоваться второй конструкцией, так как она проще.

```
<?php
        /*
                Заменим  $a == true на более простую:
                вместо if ($a == true) напишем if ($a):
        */

        $a = true;
        //Если $a равно true, то...
```

```
        if ($a) echo 'Верно!'; else echo 'Неверно!'; //выведет 'Верно!', так как $a равно true


        $a = true;
        //Если $a НЕ true (то есть false!), то...
        if (!$a) echo 'Верно!'; else echo 'Неверно!'; //выведет 'Неверно!', так как $a равно true
?>
```

Также **обратите внимание** на следующие примеры:

```
<?php
        //Данное выражение всегда будет выводить 'Верно'
        if (true) echo 'Верно!'; else echo 'Неверно!';

        //Данное выражение всегда будет выводить 'Неверно'
        if (false) echo 'Верно!'; else echo 'Неверно!';

        //Данное выражение всегда будет выводить 'Неверно'
        if (!true) echo 'Верно!'; else echo 'Неверно!';

        //Данное выражение всегда будет выводить 'Верно'
        if (!false) echo 'Верно!'; else echo 'Неверно!';
?>
```

Работа вместе с HTML

PHP устроен таким образом, что его можно использовать в одном файле с HTML. Представьте, что у нас возникает такая задача: *если пользователь авторизован - вывести его логин, иначе ссылку на авторизацию.* Ее можно решить, смешивая PHP код и HTML код вместе.

Посмотрите примеры:

```
<?php
        $a = true;
        if ($a) {
                //Данный HTML код будет виден на экране только если $a равно true
                echo '<p>Здесь выведем какой-то HTML!</p>';
        }
?>
```

Однако, очень часто это может оказаться неудобным, особенно если размеры HTML кода очень большие. Поэтому можно воспользоваться следующим приемом с разрывом PHP:

```
<?php
        //Пример на работу вместе с HTML
        $a = true;
        if ($a) {
                /*
                        Мы оборвали PHP, но HTML код ниже
                        все равно будет виден на экране только если $a равно true.
                */
        ?>
```

```
                <p>Здесь выведем какой-то HTML!</p>
        <?php
                /*
                        Здесь мы продожаем PHP, открыв скобку <?php
                        и закрываем фигурную скобку от ифа.
                */
        }
?>
```

Вложенные if

Предположим, нам необходимо спросить у PHP такую вещь: *если переменная **$a** пуста, то вывести 'Введите $a', если не пуста, то проверить - больше нуля **$a** или нет. Если больше нуля - то вывести 'Больше нуля!', если меньше - вывести 'Меньше нуля'.*

Одной конструкцией **if** здесь не обойтись, нужно использовать две таким образом, чтобы одна была внутри другой:

```
<?php
        if (empty($a)) { //если переменная $a пуста
                echo 'Введите $a!';
        } else { //если переменная $a НЕ пуста
                if ($a > 0) { //спрашиваем, больше ли нуля переменная $a
                        echo 'Больше нуля!';
                } else {
                        echo 'Меньше нуля!';
                }
        }
?>
```

Конструкция elseif

Недостатком конструкции предыдущего примера является большое количество фигурных скобок. Поэтому была изобретена специальная конструкция **elseif** (пишется слитно!), которая представляет собой одновременно и **else**, и начало вложенного **if**:

```
<?php
        //Решение предыдущей задачи через конструкцию elseif
        if (empty($a)) {
                echo 'Введите $a!';
        } elseif ($a > 0) { //одновременно выполняется else для empty($a) и спрашивается больше ли $a нуля
                echo 'Больше нуля!';
        } else {
                echo 'Меньше нуля!';
        }
?>
```

Можно использовать **несколько elseif**, но злоупотреблять этим не стоит (лучше будет воспользоваться конструкцией **switch-case**, о которой ниже).

Несколько if

Пусть у нас есть такая задача: *сайт поддерживает 3 языка - русский, английский, немецкий. Переменная **$lang** может принимать 3 значения - 'ru', 'en' и 'de'. В зависимости от значения переменной **$lang** следует вывести фразу на одном из языков.*

Решение: можно было бы воспользоваться **вложенными ифами** или **elseif**. Выглядело бы это примерно так:

```
<?php
        //Решение задачи через elseif - не самое удачное
```

```
        if ($a == 'ru') { //фраза на русском
                echo 'Русский текст';
        } elseif ($a == 'en') { //фраза на английском
                echo 'Английский текст';
        } elseif ($a == 'de') { //фраза на немецком
                echo 'Немецкий текст';
        }
?>
```

Такое решение не слишком красивое - блок **else** здесь не нужен! Проще всего будет написать не один длинный **if** с несколькими **else**, а *несколько if вообще без else*:

```
<?php
        //Решение задачи через несколько if - оно намного лучше
        if ($a == 'ru') { //фраза на русском
                echo 'Русский текст';
        }
        if ($a == 'en') { //фраза на английском
                echo 'Английский текст';
        }
        if ($a == 'de') { //фраза на немецком
                echo 'Немецкий текст';
        }

        /*
                В данном коде сработает только один из ифов,
                так как $a может иметь только одно из значений
        */
?>
```

Однако, **это решение тоже не слишком удобно**. Представьте, что у вас будет не три языка, а 10 - вам придется написать 10 конструкций if.

Для таких случаев существует конструкция **switch-case**.

Конструкция switch-case

Конструкция **switch-case** представляет собой альтернативу **if-else**, ее рекомендуется использовать в случае множественного выбора (например, 10 различных языков, как в нашей задаче).

Изучите ее синтаксис:

```
<?php
        switch ($переменная) {
                case 'значение1':
                        здесь код, который выполнится в случае, если переменная имеет значение1;
                break;
                case 'значение2':
                        здесь код, который выполнится в случае, если переменная имеет значение2;
                break;
                case 'значение3':
```

```
                здесь код, который выполнится в случае, если переменная имеет значение3;
                break;
                default:
                        этот код выполнится в случае, если переменная не совпала ни с одним значением;
                break;
        }
?>
```

Решим нашу задачу с тремя языками с помощью данной конструкции:

```
<?php
        switch ($lang) {
                case 'ru':
                        echo 'Русский текст';
                break;
                case 'en':
                        echo 'Английский текст';
                break;
                case 'de':
                        echo 'Немецкий текст';
                break;
                default:
                        echo 'Данный язык не поддерживается';
                break;
        }
?>
```

Задачи на конструкции if-else, switch-case в PHP

Примеры решения задач

Задача

Задача. Если переменная **$a** равна 10, то выведите 'Верно', иначе выведите 'Неверно'.

Решение:

```
<?php
        $a = 10;
        if ($a == 10) {
                echo 'Верно';
        } else {
                echo 'Неверно';
        }
?>
```

Задача

Задача. В переменной **$min** лежит число от 0 до 59. Определите в какую **четверть часа** попадает это число (в первую, вторую, третью или четвертую).

Решение:

```php
<?php
	$min = 10;

	if ($min >= 0 and $min <= 14) {
		echo 'В первую четверть.';
	}

	if ($min <= 15 and $min >= 30) {
		echo 'Во вторую четверть.';
	}

	if ($min <= 31 and $min >= 45) {
		echo 'В третью четверть.';
	}

	if ($min >= 46 and $min <= 59) {
		echo 'В четвертую четверть.';
	}
?>
```

Задача

Задача. Переменная **$lang** может принимать два значения: 'ru' и 'en'. Если она имеет значение 'ru', то в переменную **$arr** запишем массив дней недели на русском языке, а если имеет значение 'en' – то на английском. Решите задачу через **2 if**, через **switch-case** и через **многомерный массив** без ифов и switch.

Решение:

Решение через 2 if:

```php
<?php
	$lang = 'ru';

	if ($lang == 'ru') {
		$arr = ['пн', 'вт', 'ср', 'чт', 'пт', 'сб', 'вс'];
	}
	if ($lang == 'en') {
		$arr = ['mn', 'ts', 'wd', 'th', 'fr', 'st', 'sn'];
	}

	var_dump($arr);
?>
```

Решение через switch-case:

```php
<?php
	$lang = 'ru';

	switch ($lang) {
		case 'ru':
			$arr = ['пн', 'вт', 'ср', 'чт', 'пт', 'сб', 'вс'];
```

```
                break;
                case 'en':
                        $arr = ['mn', 'ts', 'wd', 'th', 'fr', 'st', 'sn'];
                break;
        }

        var_dump($arr);
?>
```

Решение через многомерный массив:

```
<?php
        $lang = 'ru';

        $arr = [
                'ru'=>['пн', 'вт', 'ср','чт', 'пт', 'сб', 'вс'],
                'en'=>['mn', 'ts', 'wd', 'th', 'fr', 'st', 'sn'],
        ];

        var_dump($arr[$lang]);
?>
```

Задачи для решения

Работа с if-else

1. Если переменная **$a** равна нулю, то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при **$a**, равном 1, 0, -3.

2. Если переменная **$a** больше нуля, то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при **$a**, равном 1, 0, -3.

3. Если переменная **$a** меньше нуля, то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при **$a**, равном 1, 0, -3.

4. Если переменная **$a** больше или равна нулю, то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при **$a**, равном 1, 0, -3.

5. Если переменная **$a** меньше или равна нулю, то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при **$a**, равном 1, 0, -3.

6. Если переменная **$a** не равна нулю, то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при **$a**, равном 1, 0, -3.

7. Если переменная **$a** равна 'test', то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при **$a**, равном 'test', 'тест', 3.

8. Если переменная **$a** равна '1' и по значению и по типу, то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при **$a**, равном '1', 1, 3.

Работа с empty и isset

9. Если переменная **$a** пустая, то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при **$a**, равном 1, 3, -3, 0, null, true, '', '0'.

10. Если переменная **$a** НЕ пустая, то выведите 'Верно', иначе выведите 'Неверно'.

11. Если переменная **$a** существует, то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при **$a**, равном 3 и null.

12. Если переменная **$a** НЕ существует, то выведите 'Верно', иначе выведите 'Неверно'.

Работа с логическими переменными

13. Если переменная **$var** равна true, то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при **$var**, равном true, false. Напишите два варианта скрипта - с короткой записью и с длинной.

14. Если переменная **$var** НЕ равна true, то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при **$var**, равном true, false. Напишите два варианта скрипта - с короткой записью и с длинной.

Работа с OR и AND

15. Если переменная **$a** больше нуля и меньше 5-ти, то выведите 'Верно', иначе выведите 'Неверно'. Проверьте работу скрипта при **$a**, равном 5, 0, -3, 2.

16. Если переменная **$a** равна нулю или равна двум, то прибавьте к ней 7, иначе поделите ее на 10. Выведите новое значение переменной на экран. Проверьте работу скрипта при **$a**, равном 5, 0, -3, 2.

17. Если переменная **$a** равна или меньше 1, а переменная **$b** больше или равна 3, то выведите сумму этих переменных, иначе выведите их разность (результат вычитания). Проверьте работу скрипта при **$a** и **$b**, равном 1 и 3, 0 и 6, 3 и 5.

18. Если переменная **$a** больше 2-х и меньше 11-ти, или переменная **$b** больше или равна 6-ти и меньше 14-ти, то выведите 'Верно', в противном случае выведите 'Неверно'.

На switch-case

19. Переменная **$num** может принимать одно из значений: 1, 2, 3 или 4. Если она имеет значение '1', то в переменную **$result** запишем 'зима', если имеет значение '2' – 'лето' и так далее. Решите задачу через **switch-case**.

Задачи

20. В переменной **$day** лежит какое-то число из интервала от **1** до **31**. Определите в какую декаду месяца попадает это число (в первую, вторую или третью).

21. В переменной **$month** лежит какое-то число из интервала от **1** до **12**. Определите в какую пору года попадает этот месяц (зима, лето, весна, осень).

22. В переменной **$year** хранится год. Определите, является ли он **високосным** (в таком году есть 29 февраля). Год будет високосным в **двух случаях**: либо он делится на **4**, но при этом не делится на **100**, либо делится на **400**. Так, годы 1700, 1800 и 1900 не являются високосными, так как они делятся на **100** и не делятся на **400**. Годы 1600 и 2000 - високосные, так как они делятся на **400**.

23. Дана строка с символами, например, 'abcde'. Проверьте, что первым символом этой строки является буква 'a'. Если это так - выведите 'да', в противном случае выведите 'нет'.

24. Дана строка с цифрами, например, '12345'. Проверьте, что первым символом этой строки является цифра 1, 2 или 3. Если это так - выведите 'да', в противном случае выведите 'нет'.

25. Дана строка из 3-х цифр. Найдите сумму этих цифр. То есть сложите как числа первый символ строки, второй и третий.

26. Дана строка из 6-ти цифр. Проверьте, что сумма первых трех цифр равняется сумме вторых трех цифр. Если это так - выведите 'да', в противном случае выведите 'нет'.