**Лекция 8**

**Тема : Управляющие конструкции**

### Условные операторы

#### Оператор if

Это один из самых важных операторов многих языков, включая PHP. Он позволяет выполнять фрагменты кода в зависимости от условия. Структуру оператора *if* можно представить следующим образом:

if (выражение) блок\_выполнения

Здесь выражение есть любое правильное PHP-выражение (т.е. все, что имеет значение). В процессе обработки скрипта выражение преобразуется к *логическому типу*. Если в результате преобразования значение выражения истинно ( True ), то выполняется блок\_выполнения. В противном случае блок\_выполнения игнорируется. Если блок\_выполнения содержит несколько команд, то он должен быть заключен в фигурные скобки { }.

Правила преобразования выражения к *логическому типу*:

1. В FALSE преобразуются следующие значения:
   * логическое False
   * целый ноль ( 0 )
   * действительный ноль ( 0.0 )
   * пустая строка и строка "0"
   * массив без элементов
   * объект без переменных (подробно об объектах будет рассказано в одной из следующих лекций)
   * специальный тип NULL
2. Все остальные значения преобразуются в TRUE.

<?

$names = array("Иван","Петр","Семен");

if ($names[0]=="Иван") {

echo "Привет, Ваня!";

$num = 1;

$account = 2000;

}

if ($num) echo "Иван первый в списке!";

$bax = 30;

if ($account > 100\*$bax+3)

echo "Эта строчка не появится

на экране, так как условие не выполнено";

?>

Пример 3.1. Условный оператор if

#### Оператор else

Мы рассмотрели только одну, основную часть оператора *if* . Существует несколько расширений этого *оператора. Оператор* *else* расширяет *if* на случай, если проверяемое в *if* выражение является неверным, и позволяет выполнить какие-либо действия при таких условиях.

Структуру оператора *if* , расширенного с помощью оператора *else* , можно представить следующим образом:

if (выражение) блок\_выполнения

else блок\_выполнения1

Эту конструкцию if...else можно интерпретировать примерно так: если выполнено условие (т.е. выражение=true ), то выполняем действия из блока\_выполнения, иначе – действия из блока\_выполнения1. Использовать оператор *else* не обязательно.

Посмотрим, как можно изменить предыдущий пример, учитывая необходимость совершения действий и в случае невыполнения условия.

<?

$names = array("Иван","Петр","Семен");

if ($names[0]=="Иван") {

echo "Привет, Ваня!";

$num = 1;

$account = 2000;

} else {

echo "Привет, $names[0].

А мы ждали Ваню :(";

}

if ($num) echo "Иван первый в списке!";

else echo "Иван НЕ первый в списке?!";

$bax = 30;

if ($account > 100\*$bax+3)

echo "Эта строка не появится на экране,

так как условие не выполнено";

else echo "Зато появится эта строка!";

?>

Пример 3.2. Оператор else

#### Оператор elseif

Еще один способ расширения *условного оператора* *if* – использование оператора *elseif* . *elseif* – это комбинация *else* и *if* . Как и *else* , он расширяет *if* для выполнения различных действий в том случае, если условие, проверяемое в *if* , неверно. Но в отличие от *else* , альтернативные действия будут выполнены, только если *elseif*-условие является верным. Структуру оператора *if* , расширенного с помощью операторов *else* и *elseif* , можно представить следующим образом:

if (выражение) блок\_выполнения

elseif(выражение1) блок\_выполнения1

...

else блок\_выполненияN

Операторов *elseif* может быть сразу несколько в одном if-блоке. *Elseif*-утверждение будет выполнено, только если предшествующее if-условие является False, все предшествующие *elseif*-условия являются False, а данное *elseif*-условие – True.

<?

$names = array("Иван","Петр","Семен");

if ($names[0]=="Иван") {

// если первое имя в массиве Иван

echo "Привет, Ваня!";

}elseif ($names[0] == "Петр"){

// если первое имя

// не Иван, а Петр

echo "Привет, Петя!";

}elseif ($names[0] == "Семен"){

// если первое имя не

// Иван, не Петр, а Семен

echo "Привет, Сеня!";

}else {

// если первое имя не Иван,

// не Петр и не Семен

echo "Привет, $names[0]. А ты кто такой?";

}

?>

Пример 3.3. Оператор elseif

#### Альтернативный синтаксис

PHP предлагает *альтернативный синтаксис* для некоторых своих управляющих структур, а именно для *if* , *while* , *for* , *foreach* и *switch* . В каждом случае открывающую скобку нужно заменить на двоеточие ( :), а закрывающую – на endif;, endwhile; и т.д. соответственно.

Например, синтаксис оператора *if* можно записать таким образом:

if(выражение): блок\_выполнения endif;

Смысл остается тем же: если условие, записанное в круглых скобках оператора *if* , оказалось истиной, будет выполняться весь код, от двоеточия " :" до команды endif;. Использование такого синтаксиса полезно при встраивании php в html-код.

<?php

$names = array("Иван","Петр","Семен");

if ($names[0]=="Иван"):

?>

Привет, Ваня!

<?php

endif; ?>

Пример 3.4. Использование альтернативного синтаксиса

Если используются конструкции *else* и *elseif* , то также можно задействовать *альтернативный синтаксис*:

<?php

$a=1;

if ($a == 5):

print "a равно 5";

print "...";

elseif ($a == 6):

print "a равно 6";

print "!!!";

else:

print "a не равно ни 5, ни 6";

endif;

?>

#### Оператор switch

Еще одна конструкция, позволяющая проверять условие и выполнять в зависимости от этого различные действия, – это *switch* . На русский язык название данного оператора можно перевести как "переключатель". И смысл у него именно такой. В зависимости от того, какое значение имеет переменная, он переключается между различными блоками действия. *switch* очень похож на оператор if...*elseif*...else или набор операторов *if* . Структуру *switch* можно записать следующим образом:

switch (выражение или переменная){

case значение1:

блок\_действий1

break;

case значение2:

блок\_действий2

break;

...

default:

блок\_действий\_по\_умолчанию

}

В отличие от *if* , здесь значение выражения не приводится к логическому типу, а просто сравнивается со значениями, перечисленными после ключевых слов *case* ( значение1, значение2 и т.д.). Если значение выражения совпало с каким-то вариантом, то выполняется соответствующий блок\_действий – от двоеточия после совпавшего значения до конца *switch* или до первого оператора *break* , если таковой найдется. Если значение выражения не совпало ни с одним из вариантов, то выполняются действия по умолчанию ( блок\_действий\_по\_умолчанию ), находящиеся после ключевого слова default. Выражение в *switch* вычисляется только один раз, а в операторе *elseif* – каждый раз, поэтому, если выражение достаточно сложное, то *switch* работает быстрее.

Пример [3.3](https://intuit.ru/studies/courses/42/42/lecture/27179?page=1#example.3.3) можно переписать с использованием *switch* следующим образом:

<?

$names = array("Иван","Петр","Семен");

switch ($names[0]){

case "Иван":

echo "Привет, Ваня!";

break;

case "Петр":

echo "Привет, Петя!";

break;

case "Семен":

echo "Привет, Сеня!";

break;

default:

echo "Привет, $names[0].

А как Вас зовут?";

}

?>

Если в этом примере опустить оператор *break* , например, в *case* "Петр":, то, если переменная окажется равной строке "Петр", после вывода на экран сообщения "Привет, Петя!" программа пойдет дальше и выведет также сообщение "Привет, Сеня!" и только потом, встретив *break* , продолжит свое выполнение за пределами *switch* .

Для конструкции *switch* , как и для *if* , возможен *альтернативный синтаксис*, где открывающая *switch* фигурная скобка заменяется двоеточием, а закрывающая – endswitch; соответственно.