ТЕМА: СТРУКТУРЫ УПРАВЛЕНИЯ. УСЛОВНЫЙ ОПЕРАТОР.

Оператор вывода

Так, в PHP для того, чтобы вывести информацию на экран предусмотрены операторы **«print» и «echo».**

Оператор вывода

Когда нужно отобразить текст на веб-странице, то оператор echo является наиболее употребляемым оператором в PHP.

<?php echo 'Привет от PHP'; ?>

Для отображения текста можно использовать как двойные кавычки, так и одинарные. Для чисел кавычки можно не использовать:

<?php echo 2016; ?>

Оператор вывода

Оператор echo также может участвовать
в форматировании веб-страницы:

**<?php echo 'Петров Иван
Родился
...'; ?>**

Отображение в браузере:

Петров Иван
Родился
...

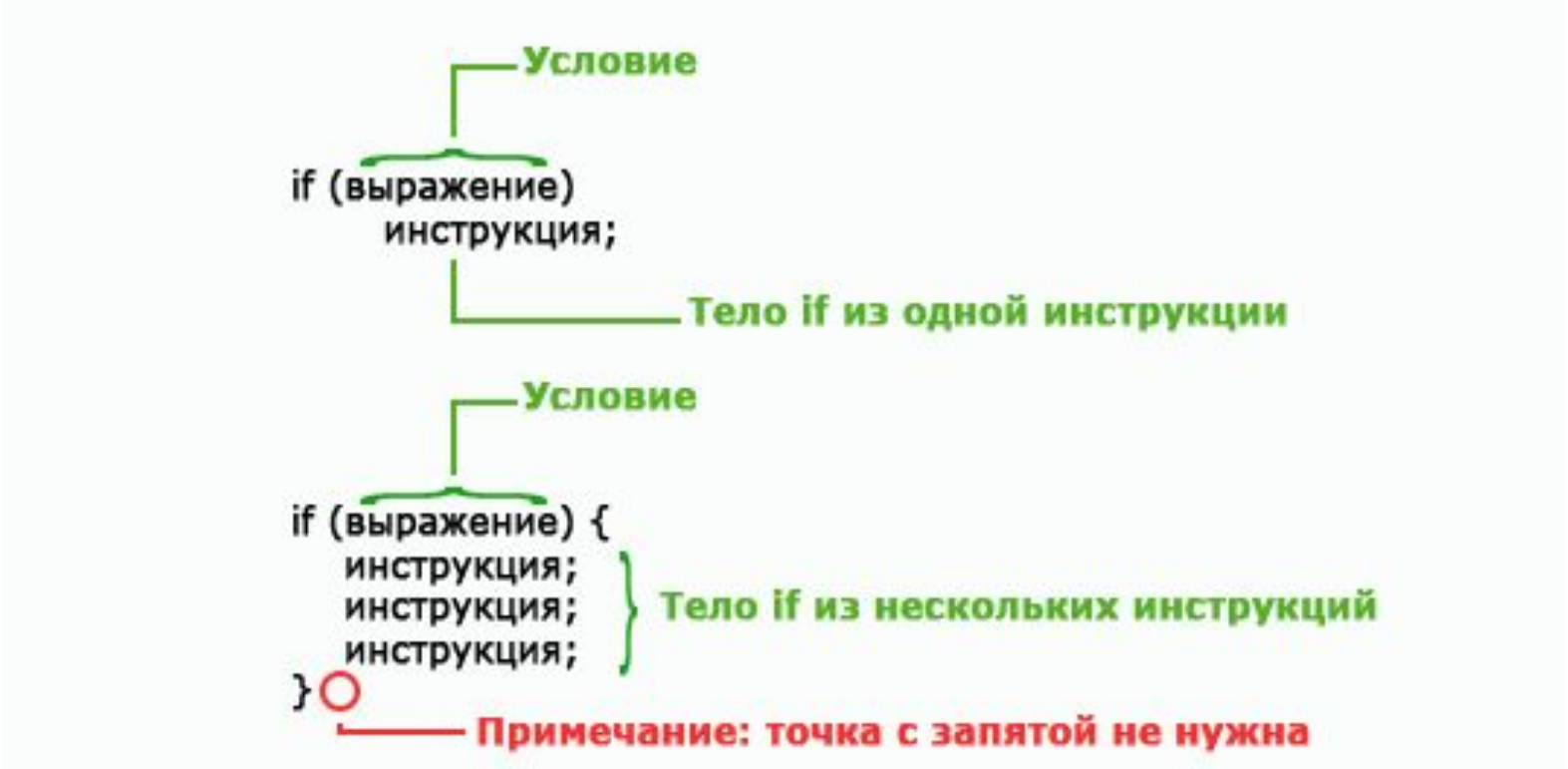
Условный оператор

Условный оператор позволяет пропустить или выполнить некоторый блок кода в зависимости от результата вычисления указанного выражения - условия.

Условный оператор if

Оператор if является наиболее простым из операторов ветвлений.

Синтаксис оператора if:



Условный оператор if

Оператор if сначала вычисляет условное выражение указанное в круглых скобках, результатом этого выражения является булево значение. Если полученный результат является истинным (true), то инструкция выполняется. Если выражение возвращает ложное значение (false), то инструкция не выполняется. В качестве условия может применяться выражение любой сложности.

Условный оператор if

Если в теле оператора if используется всего одна инструкция, то заключать ее в фигурные скобки можно, но не обязательно.

Если нужно выполнить в теле оператора if не одну инструкцию, а несколько, тогда эти несколько инструкций необходимо заключить в фигурные скобки. Обратите внимание на то, что после закрывающей фигурной скобки не следует ставить точку с запятой.

Пример оператора условия

```php
<?php

$a1 = 5;

// Если значение переменной меньше 100, выполнить инструкцию, следующую за оператором if
if ($a1 < 100)
  echo "Число $a1 меньше, чем 100";

// переменная $username никак не инициализирована: по умолчанию значение null
// с помощью оператора (Логическое НЕ) инвертируем значение в TRUE
if (!$username) {
  $username = "User";
  // и выводим имя пользователя на экран
  echo $username;
}

?>
```

Пример оператора условия

```php
<?php

  $a = 0;
  $b = 50;
  $i = 10;
  $j = 15;

  // Если выражение "i == 10" возвращает true
  if($i == 10)
    // то выполняется проверка условного выражения вложенного оператора if
    if ($j < 20) $a = $b;

?>
```

Оператор if else

Оператор if позволяет выполнять инструкции в том случае, если условие истинно. Если же условие оказывается ложно, то никаких действий не выполняется. Однако часто бывает необходимо выполнить одни инструкции, если определенное условие истинно, и другие инструкции, если условие ложно. Именно для таких случаев используется ветвление if else. Оно состоит из оператора if, за которым следует блок инструкций и ключевого слова else, за которым следует другой блок инструкций.

Синтаксис оператора if else:

Условие

```
if (выражение)
    инструкция;
else
    инструкция;
```

Тело if из одной инструкции

Тело else из одной инструкции

Условие

```
if (выражение) {
    инструкция;
    инструкция;
    инструкция;
}
else {
    инструкция;
    инструкция;
    инструкция;
}
```

Тело if из нескольких инструкций

Тело else из нескольких инструкций

Оператор if else

Оператор else не является обязательным.

Блок инструкций расположенный после else выполняется по умолчанию, т.е. когда условное выражение в if возвращает значение false.

Оператор else не может быть использован отдельно от оператора if.

Блок else должен располагаться только после оператора if, его можно рассматривать, как действие по умолчанию.

Пример оператора if else

```php
<?php

  $username = 'Holly';

  if ($username == 'Admin') {
    echo 'Добро пожаловать на страницу администратора.';
  }
  else {
    echo 'Добро пожаловать в гостевую страницу';
  }

?>
```

Пример оператора if else

```php
<?php

  if($i) {                  // внутренний блок if($i)
    if($a) инструкция;
    if($b) инструкция;
    else  инструкция; // этот else относится к if($b) - так как он ближе
  }
  else инструкция; // этот else относится к if($i)

?>
```

Конструкция elseif/else if

Оператор if/else вычисляет значение условного выражения и выполняет тот или иной фрагмент программного кода. Иногда требуется выполнить один из многих фрагментов. Если нужно проверить несколько условий подряд, то для этого подойдет конструкция elseif или else if (это одна и та же конструкция, просто по разному записана). Формально она не является самостоятельной конструкцией PHP; это лишь распространенный стиль программирования, заключающийся в применении повторяющихся операторов if/else. Она позволяет проверять дополнительные условия, пока не будет найдено истинное или достигнут блок else. Конструкция elseif/else if должна располагаться после оператора if и перед оператором else, если такой имеется.

Пример elseif/else if

```php
<?php

  $username = 'Арни';

  if ($username == 'Админ') {
    echo 'Добро пожаловать на страницу администратора.';
  }
  elseif ($username == 'Гость') {
    echo 'Доступ запрещен!';
  }
  else if ($username == 'Арни') {
    echo "Добро пожаловать $username!";
  }
  else {
    echo ('Добро пожаловать в гостевую страницу');
  }

?>
```

Здесь проверяется три условия, и, в зависимости от значения переменной $username, выполняются разные действия. Здесь последовательность операторов if, где каждый оператор if является частью конструкции else предыдущего if.

Пример elseif/else if

```php
<?php

  $username = 'Арни';

  if ($username == 'Админ') {
    echo 'Добро пожаловать на страницу администратора.';
  }
  else {
    if ($username == 'Гость') {
      echo 'Доступ запрещен!';
    }
    else {
      if ($username == 'Арни') {
        echo "Добро пожаловать $username!";
      }
      else {
        echo ('Добро пожаловать в гостевую страницу');
      }
    }
  }

?>
```

Тернарный оператор - ? :

Существует распространенная в программировании ситуация, когда в случае выполнения некоторого условия переменной необходимо присвоить одно значение и в случае невыполнения этого условия другое значение. В следующем примере переменной $min присваивается наименьшее из значений $foo и $bar с помощью конструкции if else:

```php
<?php

  if ($foo < $bar)
    $min = $foo;
  else
    $min = $bar;

?>
```

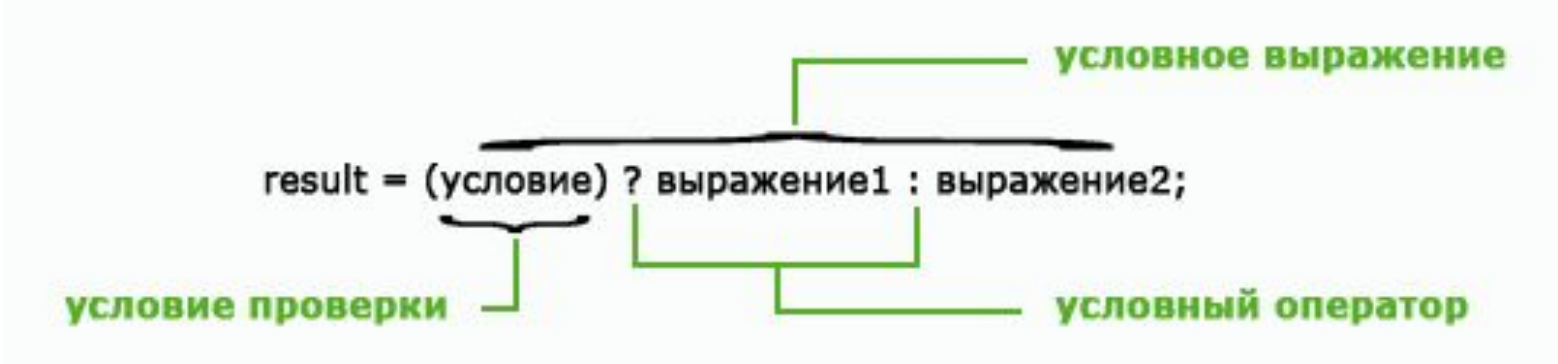
Тернарный оператор - ? :

На практике подобные действия оказались настолько распространенными, что был разработан специальный условный оператор, выполняющий данные действия - сокращенный способ присваивания значения переменной на основе выполнения того или иного условия – тернарный оператор.

```php
<?php

  $min = ($foo < $bar) ? $foo : $bar;

?>
```

Сначала проверяется результат работы условного выражения, если условие вернет значение true, то переменной $min присвоится значение переменной $foo, если false, значение переменной $bar.

Синтаксис тернарного оператора



Интерпретатор PHP вычисляет значение условия, если оно возвращает true, переменной присваивается значение выражения 1. Если условие возвращает false, переменной присваивается значение выражения 2.

Скобки вокруг условного выражения не обязательны, но их довольно часто употребляют для более легкого визуального восприятия тернарного оператора. Тернарные выражения вычисляются слева направо.

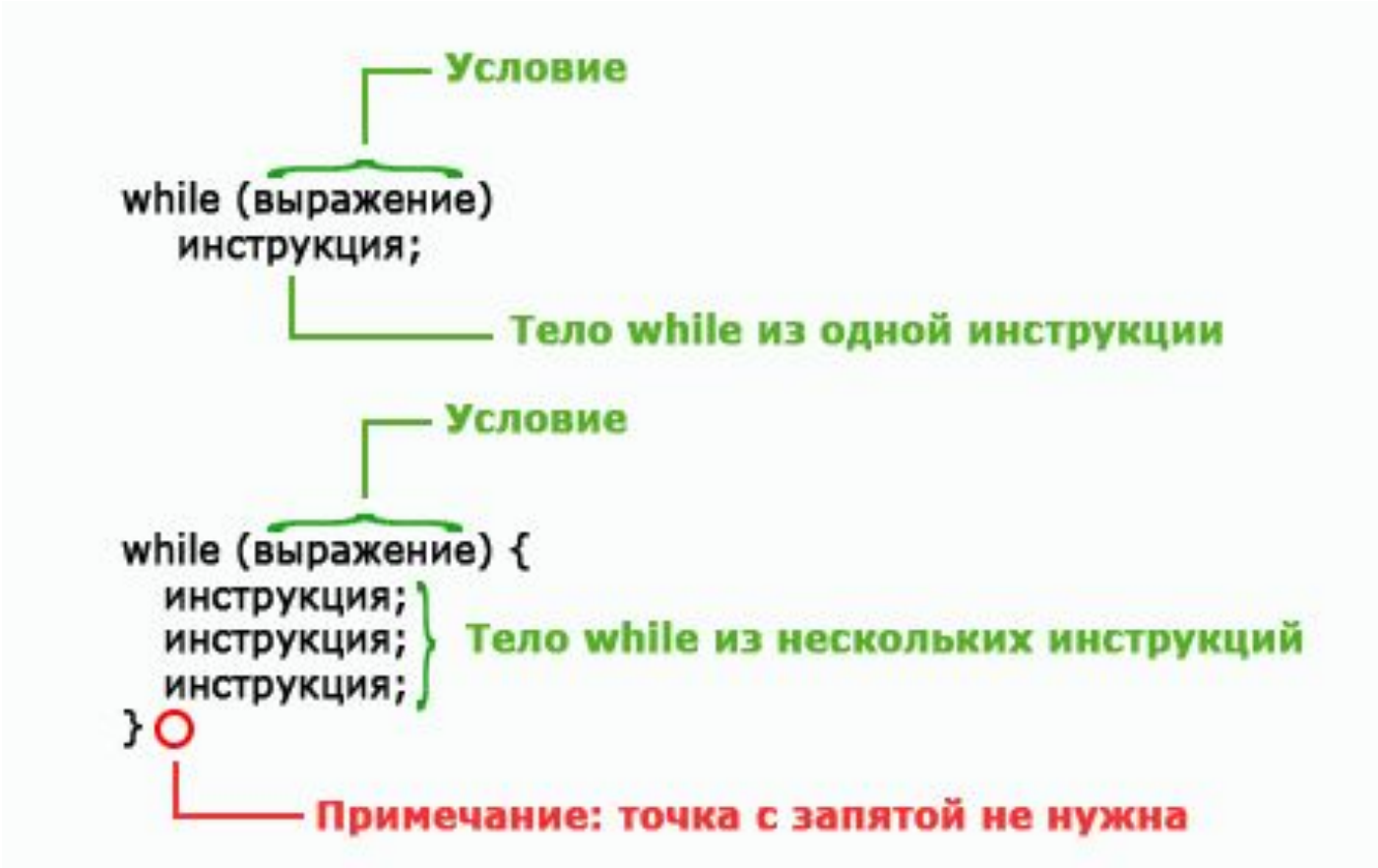
Циклы

Цикл - это блок кода, позволяющий повторять выполнение некоторых действий (инструкций) определенное количество раз. Каждое отдельное исполнение (одно или более повторений) последовательности инструкций в цикле называется - ***итерация.***

Каждый цикл состоит из двух основных частей. Первая определяет, когда должно быть остановлено исполнение цикла. Вторая - фрагмент программного кода, выполняющий необходимые действия, который может состоять из одной единственной инструкции или из нескольких инструкций, заключенных в фигурные скобки.

Оператор цикла while

Синтаксис:



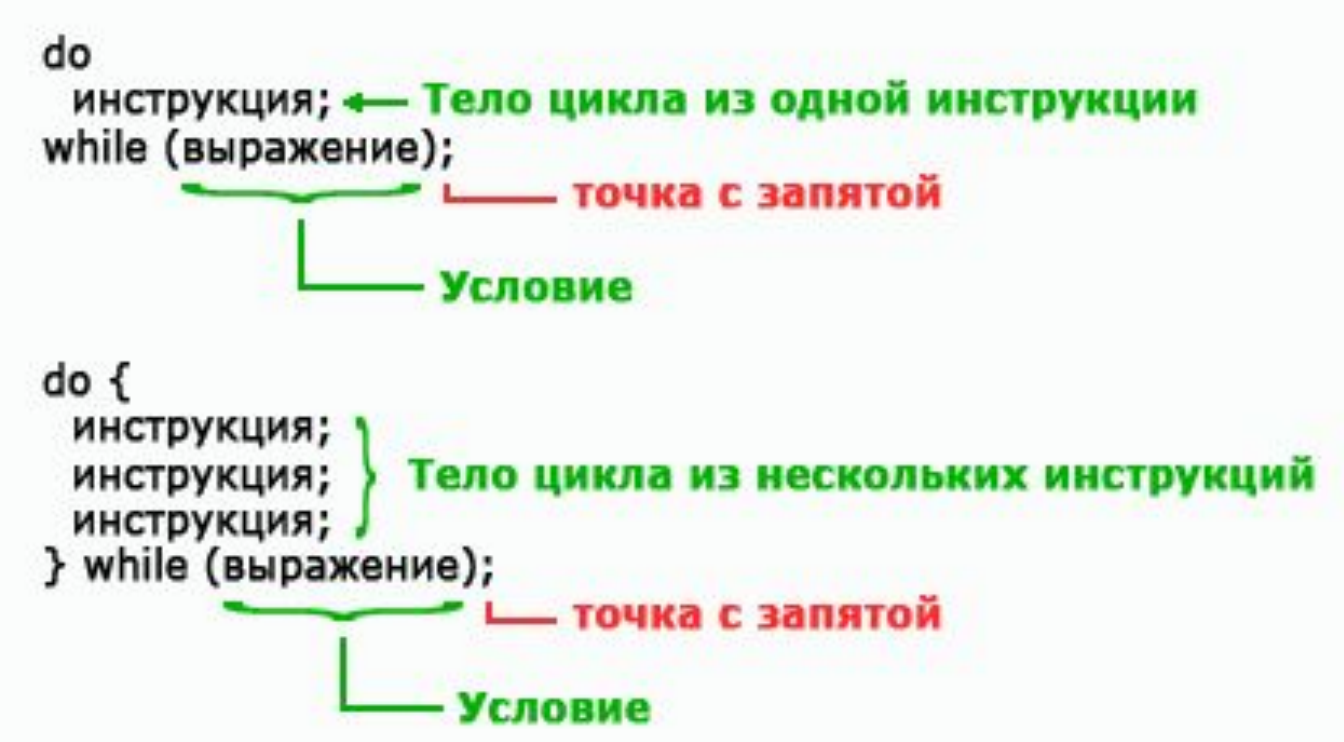
Оператор цикла while

```php
<?php

  $num = 1;

  while ($num <= 10) {
    echo "Итерация номер: $num<br>\n";

    $num++;
  }

?>
```

Оператор цикла do while

В цикле do while условное выражения проверяется в конце итерации, а не в начале.

Синтаксис цикла:



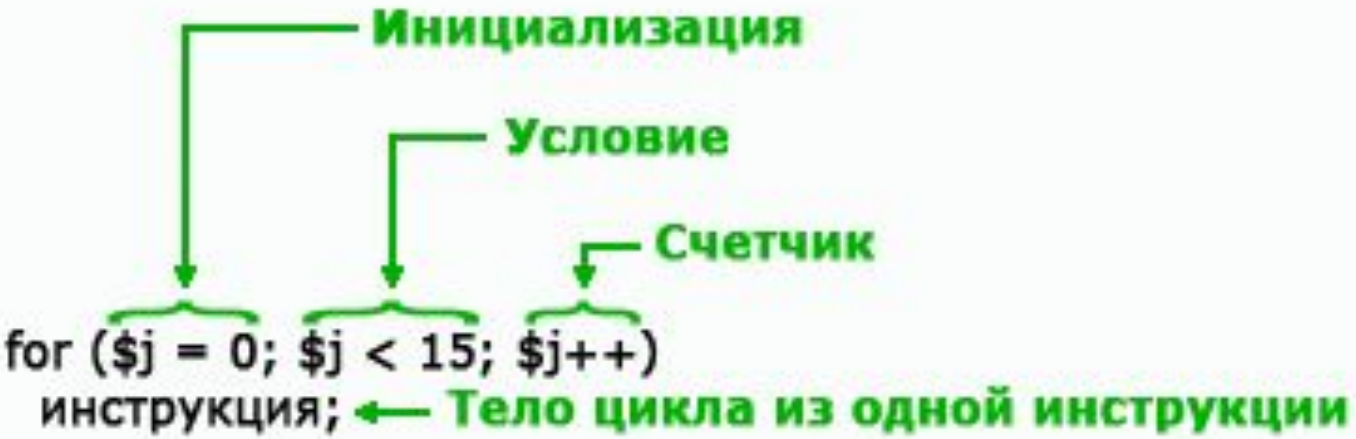
Оператор цикла do while

Между циклом do while и обычным циклом while имеется два отличия. Во-первых, цикл do требует как ключевого слова do (для отметки начала цикла), так и ключевого слова while (для отметки конца цикла и указания условия). Во-вторых, в отличие от цикла while, цикл do завершается точкой с запятой. Данная разновидность цикла полезна, когда тело цикла должно быть исполнено хотя бы один раз, независимо от значения условного выражения.

Оператор цикла do while

```php
<?php

  $num = 1;

  do {
    echo "Итерация номер: $num<br>\n";

    $num++;
  } while ($num <= 10);

?>
```

Оператор цикла for




```
for ($j = 0; $j < 15; $j++) {
  инструкция;
  инструкция;
  инструкция;
}
```

Тело цикла из нескольких инструкций

Оператор цикла for

Данный цикл состоит из ключевого слова for, за которым следуют круглые скобки, содержащие внутри три выражения, разделенных точками с запятой. Он имеет следующий порядок выполнения:

1. В начале цикла выполняется инициализирующее выражение, оно всегда вычисляется (выполняется) только один раз в самом начале.
2. Далее следует условное выражение. Перед началом каждой итерации вычисляется значение условного выражения (условие выполнения), если оно принимает значение TRUE, то тело цикла выполняется, если оно принимает значение FALSE, выполнение цикла завершается. Если при первой проверке условие оказывается ложным, тело цикла не выполнится ни разу.
3. В самом конце, после выполнения действий в теле цикла обрабатывается последнее выражение (следующее выражение после условия). В данном случае это инкрементирующее выражение - оно увеличивает с помощью инкремента значение переменной-счетчика.

Оператор цикла for

```php
<?php

  for ($k = 1; $k <= 10; $k++) {
    echo "$k ";
  }

?>
```

Закрепление материала

Записать инструкции скрипта:

1. **5 раз увеличить переменную A на 2 и вывести результат**
2. **Найти наибольшее из двух переменных A и C**
3. **Вывести значение переменной A, если оно больше 5**

Рефлексия

сегодня я узнал…
было интересно…
было трудно…
я понял, что…
я приобрел…
меня удивило…
мне захотелось…

Домашнее задание

Стивен Хольцнер, PHP в примерах, Пер. с англ.-М.: ООО «Бином-Пресс», 2007, стр. 51-70