Veebirakenduste loomise alused

JPTVR18

Jelena Kuzmina

PHP 7

- **PHP** (англ. PHP: Hypertext Preprocessor) — скриптовый язык программирования общего назначения, интенсивно применяемый для разработки веб-приложений.
- PHP отличается от JavaScript тем, что PHP-скрипты выполняются на сервере и генерируют HTML, который посылается клиенту.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Пример</title>
    </head>
    <body>

        <?php
        echo "Привет, я - скрипт PHP!";
        ?>

    </body>
</html>
```

- http://php.net/manual/ru/getting-started.php

PHP

- В файле, PHP скрипт начинается со слова - **<?php** и заканчивается на **?>**.
- Все, что между **<?php** и **?>** это PHP код.

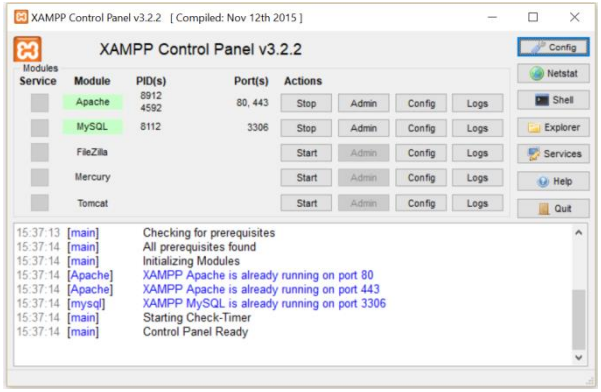
- Файлы, в котором записан PHP код нужно сохранять под расширением **.php**

Программное обеспечение

- Браузер
- Веб-сервер. Для локального тестирования вам нужно установить веб-сервер. Я рекомендую поставить XAMPP .
- Программа для редактирования PHP кода. Notepad++, NetBeans….

XAMPP

https://www.apachefriends.org/ru/index.html



- Запускаем сервер с помощью **xampp-control.exe**
- **htdocs** папка для хранения проектов (создать папку для своего проекта)
- В браузере набрать адресс проекта

http://**localhost**/papka_projekta/

Вывод текста на экран. Оператор echo.

- Когда нужно отобразить текст на веб-странице, то оператор echo является наиболее употребляемым оператором в PHP. Как его использовать - после слова echo нужно поместить строку текста в кавычки:

```
<?php
echo 'Привет от PHP';
?>
```

Echo

- Для отображения текста можно использовать как двойные кавычки, так и одинарные. Для чисел кавычки можно не использовать.
- Оператор echo также может участвовать в форматировании веб-страницы:

```
<?php
echo “Петров Иван<br>Родился<br>...”;
?>
```

Синтаксис HEREDOC

- Для отображения большого количества текста используют т. н. синтаксис heredoc. Он начинается с символов **<<<**, после которых может быть записан произвольный идентификатор. После располагаемого текста стоит указать тот самый идентификатор, что и в начале кода:

```
<?php
echo <<<END
<p>Для отображения большого <br>
количества текста используют
синтаксис heredoc</p>
END;
?>
```

Комментарии

- Комментарии нужны для описания написанного скрипта. Они нужны, если скрипт разрабатывается долгое время, или разрабатывается несколькими людьми, то невозможно запомнить всю структуру программы не оставляя описание в комментариях.
- В PHP существует 3 типа комментариев.

- **Первый** позволяет размещать комментарии в нескольких строках. Начинается такой тип комментариев с символов /* и заканчиваются */, например:

```
<?php
/* Тут может быть размещен любой текст,
даже в несколько строк */
?>
```

Комментарии

- Следующие два типа являются однострочными. Такие комментарии начинаются с символов // или # и продолжаются до конца строки. Пример:

```
<?php
// Тут может быть размещен любой текст
# Только в одной строке !
echo "Привет Всем !";
?>
```

Переменные в PHP

- Переменная - контейнер с данными. Каждая переменная содержит определенное значение.
- Синтаксис переменной состоит из знака доллара - **$** и "свободного" идентификатора которому присваивается какое-нибудь значение.
- Запомните, имя (идентификатор) переменной **не может начинаться с цифр и пробела**
- Имя переменной **чувствительно к регистру**

Создание переменной

- Переменная создается тогда, когда ей присваивают какое-нибудь значение. Для присвоения значения переменной используют оператор присвоения, который состоит из знака равенства =. Например:

```php
<?php
$surname = "Петров";
$number = 1269794645;
$pi = 3.14159265;
$hello = "Hi all";
?>
```

Вывод переменных

- Переменную можно вывести на экран с помощью оператора echo, вот так:

```php
<?php
$name = "Виктор";

echo "Ваше имя ", $name, "<br>";
?>
```

Пример

```php
<?php
$bann = 5; // Бананы
$lim = 10; // Лимоны
$together = $bann + $lim; // Всего

echo "Количество фруктов ", $together;
?>
```

- Если имя переменной заключено в двойные (не одинарные) кавычки, то переменная интерполируется. Например:

```php
<?php
$capital = "Paris";

echo "The capital of France is $capital <br />";
?>
```

Отображение в браузере:

```
The capital of France is Paris
```

Константы

- Когда не нужно менять заданное значение для переменной, то имеет смысл создать константу и потом использовать ее в любой части скрипта. Для описания константы используют функцию **define**, которой передается ее имя и значение, например:
-
```php
<?php
define("pi", 3.14);

echo "Математическая константа Пи равняется ", pi;
?>
```

Типы данных PHP

- **Boolean**. Это логический тип, который содержит значение true или false.
- **Integer**. Содержит значения целого числа (Например: 4 или 10 или другое целое число).
- **String**. Содержит значение текста произвольной длины (Например: Олег, Киев, Австрия).
- **Float**. Вещественное число (Например: 1.2, 3.14, 8.5498777).
- **Object**. Объект.
- **Array**. Массив.
- **Resource**. Ресурс (Например: файл).
- **NULL**. Значение NULL.

```php
<?php
$bool = true;    // Значение Boolean
$int = 100;    // Значение Integer
$string = "Переменная содержит текст";    // Значение String
$string2 = "5425";    // Значение String, так как число взято в кавычки !
$float = 44.122;    // Значение Float
?>
```

Математические операторы

- Числовые данные обрабатываются при помощи таких операторов PHP:

+ сумма двух чисел

- разность чисел

* умножение

/ частное от деления двух чисел

% остаток от деления

```php
<?php
echo "2 + 2 = ", 2 + 2, "<br>";
echo "5 - 2 = ", 5 - 2, "<br>";
echo "10 * 10 = ", 10 * 10, "<br>";
echo "100 / 2 = ", 100 / 2, "<br>";
echo "10 % 2 = ", 10 % 2, "<br>";
?>
```

Отображение в браузере

```
2 + 2 = 4
5 - 2 = 3
10 * 10 = 100
100 / 2 = 50
10 % 2 = 0
```

Математические функции

- **Abs**. Модуль числа.
- **Sin**. Синус.
- **Sinh**. Гиперболический синус.
- **Cos**. Косинус
- **Cosh**. Гиперболический косинус.
- **Acos**. Арккосинус
- **Acosh**. Гиперболический арккосинус.
- **Asin**. Арксинус.
- **Asinh**. Гиперболический арксинус.
- **Atan2**. Арктангенс частного двух переменных.
- **Tan**. Тангенс.
- **Tanh**. Гиперболический тангенс.
- **Atan**. Арктангенс.
- **Atan**. Гиперболический арктангенс

Математические функции

- **Base_convert**. Преобразование числа в строковом представлении из одной системы
счисления в другую.
- **Decoct**. Преобразование числа в восьмеричное представление в виде строки.
- **Bindec**. Преобразование строки, предоставленной в двоичном числе, в целое значение.
- **Octdec**. Преобразование строки, представляющей восьмеричное число, в целое число.
- **Hexdec**. Преобразование строки, которая представляет шестнадцатеричное число, в целое число.
- **Ceil**. Округление числа в большую сторону.
- **Floor**. Округление числа в меньшую сторону.
- **Deg2rad**. Градусы в радианы.
- **Exp**. Вычисление экспоненты числа.
- **Fmod**. Остаток от деления двух чисел.
- **Getrandmax**. Макс. значение, которое получают функцией ***rand()***
- **Hypot**. Вычисление гипотенузы по двум катетам.
- **Is_finite**. Проверка, является ли число конечным вещественным числом.
- **Is_infinite**. Проверка, является ли число бесконечностью.
- **Is_nan**. Проверка, является ли значение Не числом(Not-A-Number).

Математические функции

- **Lcg_value**. Генератор случайных чисел.
- **Log10**. Десятичный логарифм.
- **Log**. Натуральный логарифм.
- **Max**. Максимум заданных чисел.
- **Min**. Минимум заданных чисел.
- **Mt_getrandmax**. Макс. значение, которое можно получить функцией ***mt_rand***.
- **Mt_rand**. Генератор псевдослучайных чисел по алгоритму.
- **Pi**. Значение числа π.
- **Pow**. Возведение в степень.
- **Round**. Округляет число типа float.
- **Sqrt**. Квадратный корень.

```php
<?php
echo "round(4.2) = ", round(4.2), "<br>";
?>
```

Отображение в браузере

```
round(4.2) = 4
```

Операторы присвоения в PHP

- Основным оператором присвоения является знак равенства =. Он присваивает значение определенной переменной:

```php
<?php
$fruits = 14;
?>
```

```php
<?php
$n = $m = $p = 3;
echo $n, $m, $p;
?>
```

- Также в PHP есть комбинированные операторы, которые делают код более компактным. Вот их перечень:
- **+=**
- **-=**
- **/=**
- **.=**
- **%=**
- **&=**
- **|=**
- **^=**
- **<=**
- **>=**

- Например, если нужно прибавить 55 к значению переменной $number, это можно записать как: **$number = $number + 55**, а если использовать комбинированный оператор, то так: **$number += 55**.

Увеличение и уменьшение на 1

Оператор **++** называют инкрементом, а **--** декрементом.

++$a Пре-инкремент. Увеличивает значение на единицу, затем возвращает значение

$a++ Пост-инкремент. Возвращает текущее значение, после чего увеличивает его на единицу.

--$a Пре-декремент. Уменьшает значение на единицу, затем возвращает значение

$a-- Пост-декремент. Возвращает текущее значение, после чего уменьшает его на единицу.

```php
<?php
$a = $b = $c = $d = 2;

echo $a++, "<br>";
echo ++$b, "<br>";
echo $c--, "<br>";
echo --$d, "<br>";
?>
```

Отображение в браузере:

```
2
3
2
1
```

Приоритет операторов PHP

- Если вы используете несколько операторов одновременно в одном выражении, то нужно знать в каком порядке они будут выполняться.
- операторы с одним приоритетом выполняются слева на право
- Для изменения порядка выполнения операторов нужно использовать круглые скобки!

new
[
! ~ ++ -- (int) (float) (string) (array) (object)
@
* / %
+ - .
< >
< <= > >=
&
^
|
&&
||
? :
= += -= *= /= .= %= &= |= ^= <= >=
print
and
xor
or
,

Строковые операторы PHP

- PHP имеет два строковых оператора.
- **Первый** - оператор конкатенации **.**, который объединяет две строки в одну.
- **Второй** - конкатенирующий оператор присвоения **.=**, добавляет к строке нужное значение.
- Например:

```php
<?php
$d = "Hello";
$f = $d." world";   // Теперь $f = "Hello world"

echo $f;

echo "<br/>";

$f .= " ! ! !";   // Теперь $f = "Hello world ! ! !"

echo $f;
?>
```

Отображение в браузере:

```
Hello world
Hello world ! ! !
```

Условный оператор IF

- Во всех высокоуровневых языках программирования есть оператор if, в PHP синтаксис этого оператора такой:

if (exp)

statement

- **exp** (выражение) - логическое выражение, которое может быть истиной (TRUE) или ложью (FALSE). Например, выражение 100 > 1 это истина (TRUE).
- **statement** (инструкция) выполняется тогда, когда **exp** — истина, и не выполняется когда **exp** ложь!

- Например, если скорость машины будет больше 60 то это значит, что водитель превышает скорость

```php
<?php
$speed = 80;

if ($speed > 60)
    echo "Превышение скорости !";
?>
```

- Если нужно чтобы при выполнении условия выполнялись сразу несколько операторов, то нужно заключить их в фигурные скобки { и }:

```php
<?php
$speed = 80;

if ($speed > 60)
{    // Начало
  echo "Превышение скорости! <br>";
  echo "Пожалуйста, уменьшите скорость!";
}    // Конец
?>
```

Отображение в браузере:

```
Превышение скорости!
Пожалуйста, уменьшите скорость!
```

Операторы сравнения PHP

==	Равенство	Истина, если $a равно $b
===	Идентичность	Истина, если $a равно $b, и они одного и того же типа
!=	Неравенство	Истина, если $a не равно $b
<>	Неравенство	Истина, если $a не равно $b
!==	Неидентичность	Истина, если $a не равно $b, или они не одного типа
<	Меньше	Истина, если $a меньше $b
>	Больше	Истина, если $a больше $b
<=	Меньше или равно	Истина, если $a меньше или равно $b
>=	Больше или равно	Истина, если $a больше или равно $b

Стоит обратить внимание, что оператор сравнения записывается как `==`, а не просто `=`.

```php
<?php
$speed = 45;

if ($speed <= 60)
    echo "Скорость в пределах нормы";
?>
```

Логические операторы

and	Логическое "И"	Истина, если истинно $a и $b
&&	Логическое "И"	Истина, если истинно $a и $b
or	Логическое "ИЛИ"	Истина, если истинно $a или $b
||	Логическое "ИЛИ"	Истина, если истинно $a или $b
xor	Логическое "Исключающее ИЛИ"	Истина, если истинно $a или $b, но не оба одновременно
!	Логическое "НЕ"	Истина, если $a ложь

Как вы видите, в таблице присутствуют два оператора "И" и "ИЛИ". Это потому, что оператор `&&` или `||` имеет больший приоритет от "И" и "ИЛИ".

```php
<?php
$speed = 40;

if ($speed > 35 && $speed < 55) {
    echo "Скорость в пределах нормы";
}
?>
```

Оператор ELSE

- Очень часто нужно при истинном значении выполнить одно действие, а при ложном другое. Для этого в PHP есть оператор else. Синтаксис оператора:

if(exp)

Statement1

else

statement2

```php
<?php
$speed = 50;

if ($speed > 60)
    echo "Превышение скорости !";
else
    echo "Скорость в пределах нормы"
?>
```

Оператор ELSEIF

- Оператор if имеет еще одно расширение, это оператор elseif, он используется для последовательной проверки условий.
  Синтаксис:

if (exp)

  statement1

elseif (exp2)

  statement2

```php
<?php
$speed = 50;

if ($speed < 30)
    echo "Скорость в пределах нормы";

elseif ($speed >= 30 && $speed <= 60)
    echo "Ваша скорость {$speed} км/час";

else
    echo "Превышение скорости !";
?>
```

Тернарный оператор

- Тернарный оператор работает почти также как и оператор if, но при использовании тернарного оператора, мы вместо ключевых слов пишем ? и :.

Синтаксис:

$var = condition ? exp1 : exp2;

Если условие выполняется,
то переменной $var присваивается результат вычисления exp1, иначе exp2.

начиная с версии PHP 7.0

- можно использовать тернарный оператор в таком варианте:

```php
<?php
$var = $value ?? "Другое значение";

// еквивалентно
$var = isset($value) ? $value : "Другое значение";
?>
```

- Такой вариант полезно использовать, если нужно сначала проверить существует ли переменная. Т. е., если переменная не существует, то использовать какое-то другое значение.

Оператор SWITCH

```
switch (exp)
{
    case condition1:
    exp1;
    break;

    case condition2:
    exp2;
    break;

    case condition3:
    exp3;
    break;

    default:
    exp4;
    break;
}
```

```
<?php
$speed = 55;

switch($speed)
{
    case 30 :
        echo "Ваша скорость 30 км/час";
    break;

    case 58 :
        echo "Ваша скорость 50 км/час";
    break;

    case 70 :
        echo "Превышение скорости !";
    break;

    default :
        echo "Скорость в пределах нормы";
    break;

}
?>
```

Пример

```php
<?php
$speed = 55;

switch($speed)
{
    case 30 :
    case 58 :
        echo "Скорость в пределах нормы";
    break;

    case 70 :
        echo "Превышение скорости !";
    break;

    default :
        echo "Скорость в пределах нормы";
    break;

}
?>
```

Цикл FOR

- **for** (exp1; exp2; exp3) statement

- В выражение **exp1** вставляют начальное значение для счетчика цикла — переменная, которая считает количество раз выполнения тела цикла.
  exp2 — задает условие повторения цикла. Цикл будет выполнятся пока это условие будет true.
  exp3 — выполняется каждый раз после выполенения тела цикла. Обычно, оно используется для изменения (увеличение или уменьшение) счетчика.
  Пример:

```php
<?php
for ($i = 0; $i < 10; $i++)
{
    echo "Вывод строки. 10 раз <br>";
}
?>
```

Отображение в браузере:

Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз
Вывод строки. 10 раз

Циклы WHILE

- Цикл WHILE, вместо использования счетчика цикла проверяет некоторое условие до того, пока это условие Истина (TRUE). Синтаксис:
- **while** (exp) statement

```php
<?php
$counter = 0;
while ($counter < 5)
{
   echo "Эта строка выведется 5 раз <br>";
   $counter++;
}
?>
```

Отображение в браузере:

```
Эта строка выведется 5 раз
Эта строка выведется 5 раз
Эта строка выведется 5 раз
Эта строка выведется 5 раз
Эта строка выведется 5 раз
```

Цикл DO... WHILE

- Главное отличие цикла DO ... WHILE от WHILE в том, что первый сначала выполняется тело цикла, а потом проверяет условие. Т.е., если условие сразу Ложь, то цикл выполнится один раз.
  Синтаксис

do

statement

while (condition)

```php
<?php
$counter = 6;

do
{
   echo "Эта строка выведется 1 раз <br>";
   $counter++;
}
while ($counter < 5);

?>
```

Отображение в браузере:

```
Эта строка выведется 1 раз
```