Знакомство с PHP

Типы данных

- string (строки)
- integer (целые числа)
- double (дробные числа)
- boolean (логический тип)
- NULL
- array (массивы)

```php
$string = 'Строка';
$integer = 10;
$double = 10.15;
$boolean = true; // false
$null = null;
$array = [10, 'Привет', 11.2, false, null];
```

Условный оператор if

В сценариях PHP условные операторы обеспечивают основу для принятия решений. Условные операторы по существу управляют тем, будет ли выполняться часть сценария в зависимости от результата конкретного выражения (т.е. возвращает ли выражение логическое значение true или false).

```
if ($a > $b) {
    echo "a больше, чем b";
} elseif ($a == $b) {
    echo "a равен b";
} else {
    echo "a меньше, чем b";
}
```

Условный оператор switch

Оператор switch аналогичен серии операторов if с одинаковым условием. Часто возникает необходимость сравнивать одну и ту же переменную или выражение с массой различных значений, и выполнять разные сценарии в зависимости от того, какое значение принимает эта переменная или выражение. Это именно та ситуация, для которой удобен оператор switch.

```
switch(n){
    case label1:
        // код для выполнения, если n=label1
        break;
    case label2:
        // код для выполнения, если n=label2
        break;
    ...
    default:
        // код, выполняемый, если n отличается от всех меток labels
}
```

Операторы сравнения

Оператор	Имя	Пример	Результат
==	Равно	$x == $y	Возвращает true если $x равно $y
===	Идентичны	$x === $y	Возвращает true если $x идентично $y, и они того же типа
!=	Не равно	$x != $y	Возвращает true если $x не равно $y
<>	Не равно	$x <> $y	Возвращает true если $x не равно $y
!==	Не идентичны	$x !== $y	Возвращает true если $x не идентичны $y, или они не одного типа
>	Больше чем	$x > $y	Возвращает true если $x больше чем $y
<	Меньше чем	$x < $y	Возвращает true если $x меньше чем $y
>=	Больше чем или равно	$x >= $y	Возвращает true если $x больше чем или равно $y
<=	Меньше чем или равно	$x <= $y	Возвращает true если $x меньше чем или равно $y

Оператор spaceship (космический корабль)

Этот оператор предназначен для сравнения двух выражений. Он возвращает -1, 0 или 1 если $a, соответственно, меньше, равно или больше чем $b. Сравнение производится в соответствии с правилами сравнения типов PHP.

```php
// Целые числа
echo 1 <=> 1; // 0
echo 1 <=> 2; // -1
echo 2 <=> 1; // 1

// Числа с плавающей точкой
echo 1.5 <=> 1.5; // 0
echo 1.5 <=> 2.5; // -1
echo 2.5 <=> 1.5; // 1
```

```php
// Строки
echo "a" <=> "a"; // 0
echo "a" <=> "b"; // -1
echo "b" <=> "a"; // 1
```

Массивы

Массив — это ещё один тип данных, вроде числа или строки. Главное отличие массива от остальных типов данных заключается в его способности хранить в переменной больше одного значения.

```
$fav_shows = ['game of thrones', 'american horror story', 'walking dead'];
```

Массивы - простые операции

```php
// Создаём пустой массив
$array = [];

// Добавляем элемент
$array[] = 'Заяц';

// Создание массива с элементами [0] => Заяц [1] => Волк [2] => Белка
$arrayTwo = ['Заяц', 'Волк', 'Белка'];

// Выводим элемент с индексом 0
echo $arrayTwo[0]; // Заяц
```

Циклы

Цикл — это конструкция языка, которая позволяет выполнить блок кода больше одного раза.

Мы привыкли, что наши сценарии выполняются линейно: сверху вниз, строчка за строчкой, инструкция за инструкцией. Но что делать, если надо повторить какую-нибудь инструкцию несколько раз?

Например, как вывести на экран натуральные числа от 1 до 9? Есть очевидный способ:

```
echo 1;
echo 2;
echo 3;
// и так далее...
```

Инкремент/Декремент операторы

PHP Инкремент оператор, используются для увеличения значения переменной.

PHP Декремент оператор, используются для уменьшения значения переменной.

Оператор	Имя	Описание
++$x	Перед-инкрементом	Увеличивает $x на один, возвращает $x
$x++	После-инкремента	Возвращает $x, увеличивает $x на один
--$x	Перед-декрементом	Уменьшает $x на один, возвращает $x
$x--	После-декремента	Возвращает $x, уменьшает $x на один

Цикл While

PHP цикл while выполняет блок кода, пока заданное условие является истинным.

```
while (<условие цикла>) {
    <тело цикла>
}
```

```
$i = 0;
while ($i <= 10) {
    echo $i++;
    echo '<br>';
}
```

Цикл for

Цикл for используется, когда вы заранее знаете, сколько раз должен выполняться скрипт.

```
for (начальный счетчик; счетчик теста; счетчик увеличения){
    код для выполнения;
}
```

```
for ($x = 0; $x <= 10; $x++) {
    echo "Число: $x <br>";
}
```

Цикл foreach

Цикл foreach работает только с массивами и используется для прохождения каждой пары ключ/значение в массиве.

```
foreach ($array как $value) {
    код для выполнения;
}
```

```
$colors = array("Красный", "Зеленый", "Синий", "Желтый");

foreach ($colors as $value) {

    echo "$value <br>";

}
```

break

Часто бывает удобно при возникновении некоторых условий иметь возможность досрочно завершить цикл. Такую возможность предоставляет оператор break. Он работает с такими конструкциями как: while, do while, for, foreach или switch.

```
for ($x = 0; $x <= 3; $x++) {
    if ($arrayTwo[$x] == 'Tiger') {
        break;
    }
    echo $arrayTwo[$x];
    echo '<br>';
}
```

continue

Для остановки обработки текущего блока кода в теле цикла и перехода к следующей итерации можно использовать оператор continue. От оператора break он отличается тем, что не прекращает работу цикла, а просто выполняет переход к следующей итерации.

```
for ($x = 0; $x <= 3; $x++) {
    if ($arrayTwo[$x] == 'Tiger') {
        continue;
    }
    echo $arrayTwo[$x];
    echo '<br>';
}
```

Основные мета-теги (Раздел HEAD)

```
// Кодировка текста документа
<meta charset="utf-8">
// Автор
<meta name="author" content="Крис Миллс">
// Описание из <meta name="description"> используется на страницах поисковой выдачи.
<meta name="description" content="Наша задача - траляля траляля.">
```