Операторы PHP.

Оператор - это описание некоторого действия, которое интерпретатор должен сделать с указанными переменными.

Переменные, с которыми связан оператор, называются операндами.

Все операторы разделяются на 11 групп:

1. Арифметические
2. Присвоения
3. Битовые
4. Сравнения
5. Контроля ошибок
6. Выполнения
7. Инкремента/декремента
8. Логические
9. Строковые
10. Операторы массивов
11. Операторы типов

Арифметические операторы PHP

Пример	Название	Результат
-$a	Отрицание	Смена знака $a.
$a + $b	Сложение	Сумма $a и $b.
$a - $b	Вычитание	Разность $a и $b.
$a * $b	Умножение	Произведение $a и $b.
$a / $b	Деление	Частное от деления $a на $b.
$a % $b	Деление по модулю	Целочисленный остаток от деления $a на $b.

Операция деления ("/") всегда возвращает вещественный тип, даже если оба значения были целочисленными (или строками, которые преобразуются в целые числа).

Операторы инкремента и декремента

Булевые типы не подлежат инкрементированию и декрементированию.

Пример	Название	Действие
++$a	Префиксный инкремент	Увеличивает $a на единицу и возвращает значение $a.
$a++	Постфиксный инкремент	Возвращает значение $a, а затем увеличивает $a на единицу.
--$a	Префиксный декремент	Уменьшает $a на единицу и возвращает значение $a.
$a--	Постфиксный декремент	Возвращает значение $a, а затем уменьшает $a на единицу.

```php
<?php
    // Пост-инкремент/пост-декремент
    // $a сначала выводится, потом изменяется
    $a = 1000;
    echo $a++; // выведет 1000
    echo $a++; // выведет 1001
    echo $a--; // выведет 1002
 ?>

<?php
    // Пре-инкремент/пре-декремент
    // $a сначала изменяется, потом выводится
    $a = 1000;
    echo ++$a; // выведет 1001
    echo ++$a; // выведет 1002
    echo --$a; // выведет 1001
 ?>
```

Операторы присвоения

Базовый оператор присвоения обозначается как =

```
<?php
$a = ($b = 4) + 5; // результат: $a установлена значением 9, переменной $b присвоено 4.
?>
```

Комбинированные операторы

```
<?php
$a = 3;
$a += 5; // устанавливает $a значением 8, аналогично записи: $a = $a + 5;
$b = "Hello ";
$b .= "There!"; // устанавливает $b строкой "Hello There!", как и $b = $b . "There!";
?>
```

Побитовые операторы:

Пример	Название	Результат
$a & $b	Побитовое 'и'	Устанавливаются только те биты, которые установлены и в $a, и в $b.
$a | $b	Побитовое 'или'	Устанавливаются те биты, которые установлены либо в $a, либо в $b.
$a ^ $b	Исключающее или	Устанавливаются только те биты, которые установлены либо только в $a, либо только в $b
~ $a	Отрицание	Устанавливаются те биты, которые в $a не установлены, и наоборот.
$a << $b	Сдвиг влево	Все биты переменной $a сдвигаются на $b позиций влево (каждая позиция подразумевает 'умножение на 2')
$a >> $b	Сдвиг вправо	Все биты переменной $a сдвигаются на $b позиций вправо (каждая позиция подразумевает 'деление на 2')

Побитовые операторы:

```
<?php
    $a = 170; // 10101010 в бинарном виде
    $b = 240; // 11110000 в бинарном виде
    $and = $a & $b; // $and = 10100000
    $or = $a | $b;  // $or  = 11111010
    $xor = $a ^ $b; // $xor = 01011010
    $not = ~$a;     // $not = 01010101
    $L = $b << 1; // $L = 11100000
    $R = $b >> 1; // $R = 01111000
?>
```

Операторы сравнения

В PHP разрешается сравнивать только скалярные переменные. Массивы и объекты в PHP сравнивать нельзя.

Пример	Название	Результат
$a == $b	Равно	TRUE если $a равно $b.
$a === $b	Тождественно равно	TRUE если $a равно $b и имеет тот же тип. (Добавлено в PHP 4)
$a != $b	Не равно	TRUE если $a не равно $b.
$a <> $b	Не равно	TRUE если $a не равно $b.
$a !== $b	Тождественно не равно	TRUE если $a не равно $b или в случае, если они разных типов (Добавлено в PHP 4)
$a < $b	Меньше	TRUE если $a строго меньше $b.
$a > $b	Больше	TRUE если $a строго больше $b.
$a <= $b	Меньше или равно	TRUE если $a меньше или равно $b.
$a >= $b	Больше или равно	TRUE если $a больше или равно $b.

Операторы сравнения

```
<?php
    $a = 100;   // целочисленная
    $b = '100'; // строковая
    var_dump($a == $b);    // выведет bool(true)
    var_dump($a === $b);   // выведет bool(false)
?>
```

Если сравнивается числовое значение со строкой, то строка конвертируется в число, а если сравниваются две строки, состоящие только из цифр - они обе конвертируются в числа и сравниваются в числовом виде.

Оператор сравнения - тернарный оператор "? :". Смысл его в том, что возвращается одно из значений в зависимости от условия. В общем виде тернарный оператор записывается так:

[lvalue] = (условие)? выражение, если условие истинно : выражение, если условие ложно ;

Например:

```
<?php
    ...
    $a = ($b > 100)? 'Больше сотни' : 'Меньше или равно 100';
    echo $a;
 ?>
```

Если $b больше 100 - выведется "Больше сотни", а если меньше - 'Меньше или равно 100';

Логические операторы

Пример	Название	Результат
$a and $b	Логическое 'и'	TRUE если и $a, и $b TRUE.
$a or $b	Логическое 'или'	TRUE если или $a, или $b TRUE.
$a xor $b	Исключающее 'или'	TRUE если $a, или $b TRUE, но не оба.
! $a	Отрицание	TRUE если $a не TRUE.
$a && $b	Логическое 'и'	TRUE если и $a, и $b TRUE.
$a || $b	Логическое 'или'	TRUE если или $a, или $b TRUE.

Операторы инкремента (++) и декремента (--) не работают с логическими переменными.

Логические операторы

```
<?php

    $a = true;
    $b = false;

    $and = $a && $b; // $and = FALSE
    $or = $a || $b;  // $or  = TRUE
    $xor = $a xor $b; // $xor = TRUE
    $not = ~$a;      // $not = FALSE

?>
```

Приоритеты операторов PHP

Приоритет	Оператор	Порядок выполнения
13	(постфикс)++ (постфикс)--	слева направо
12	++(префикс) --(префикс)	справа налево
11	* / %	слева направо
10	+ -	слева направо
9	<< >>	слева направо
8	< <= > >=	слева направо
7	== !=	слева направо
6	&	слева направо
5	^	слева направо
4	|	слева направо
3	&&	слева направо
2	||	слева направо
1	= += -= *= /= %= >>= <<== &= ^= |=	справа налево

Строковые операторы

Оператор конкатенации ('.'), который возвращает объединение левого и правого аргумента

Оператор присвоения с конкатенацией, который присоединяет правый аргумент к левому.

```
<?php
$a = "Hello ";
$b = $a . "World!"; // $b содержит строку "Hello World!"

$a = "Hello ";
$a .= "World!";     // $a содержит строку "Hello World!"
?>
```

Операторы массивов

Название	Пример	Результат
Объединение	$a + $b или $a and $b	Объединение двух массивов
Эквивалентность	$a == $b или $a or $b	TRUE, если массивы имеют одинаковый набор пар "ключ/значение"
Идентичность	$a === $b	TRUE, если массивы имеют одинаковый набор пар "ключ/значение", в одинаковом порядке и одинакого типа
Неэквивалентность	$a != $b или $a <> $b	TRUE, если массивы имеют различный набор пар "ключ/значение"
Неидентичность	$a !== $b	TRUE, если массивы имеют различный набор пар "ключ/значение", в одинаковом порядке и одинакого типа

Массивы

Массив - это набор элементов, к каждому из которых можно обратиться по индексу или имени. Все массивы в PHP являются ассоциативными, т.е. остоят из пар "ключ"=" значение".

```
<?php
$arr = array();
$arr['car'] = 'Чайка';
?>
```

Массив $arr содержит одну пару значений. Ключом в ней будет значение 'car', а значением - 'Чайка'.

PHP может создавать ключи автоматически при добавлении элементов в массив. Ключи всегда будут числовыми и начинаться с нуля. Например:

```
<?php
// эти два объявления массива эквивалентны
$arr = array('первый', 'второй', 'третий');
// и
$arr = array(0=>'первый', 1=>'второй',
2=>'третий');
?>
```

Сочетание "=>" используется для отделения ключа от значения в элементе при объявлении массива.

Обращение к отдельному элементу массива производится по его индексу или ключу:

```
<?php
$arr = array('первый', 'второй', 'третий');
$first = $arr[0];
?>
```

А с помощью конструкции foreach можно быстро перебрать все элементы массива:

```
<?php
$arr = array('первый', 'второй', 'третий');
foreach($arr as $key=>$value)
echo "Элемент массива номер $key равен '$value'";
?>
```

Функции для работы с массивами

array_chunk()	**Служит для разделения массива на части заданного размера. Функция возвращает массив из фрагментов исходного массива.** **<?php** **$arr = array('а', 'б', 'в', 'г', 'д');** **$arr_chunked = array_chunk($arr, 3);** **// $arr_chunked[0] содержит 'а', 'б', 'в'** **// $arr_chunked[1] содержит 'г', 'д‘** **?>**

array_combine()	**Функция объединяет два массива так, что элементы первого становятся ключами, а элементы второго - значениями результирующего ассоциативного массива.** **<?php** **$keys = array(0, 1, 2);** **$vals = array('а', 'б', 'в');** **$res = array_combine($keys, $vals);** **// $res содержит 0=>'а', 1=>'б', 2=>'в'** **?>**

array_count_valu es()	**Подсчитывает количество уникальных значений в массиве и частоту их появления** **<?php** **$arr = array('а', 'б', 'в', 'а', 'в', 'а');** **$res = array_count_values($arr);** **// $res содержит 'а' => 3, 'б' => 1, 'в' => 2** **?>**

array_diff()	**Функция выделяет разницу двух массивов, т.е. элементы, которые есть в первом массиве и нет во втором.** **<?php** **$arr1 = array('а', 'а', 'в');** **$arr2 = array('а', 'б', 'в');** **$res = array_diff($arr1, $arr2);** **// $res содержит 'б'** **?>**

Работа со строками

- Лекция 6. Работа со строками.pptx