МДК 02.02. Web-программирование. Язык PHP

Введение в язык программирования PHP

PHP – это язык программирования, который используется для написания скриптов (сценариев), выполняемых на стороне сервера.

Не зависит от программного обеспечения клиента.

Последовательность инструкций (называемая программой или скриптом) выполняется **интерпретатором языка PHP.**

Язык и его интерпретатор разрабатываются группой энтузиастов в рамках проекта с открытым кодом.

Аббревиатура PHP означает «Hypertext Preprocessor (Препроцессор Гипертекста)».

Преимущества PHP

- ✔ является свободным программным обеспечением;
- ✔ легок в освоении на всех этапах;
- ✔ поддерживается большим сообществом пользователей и разработчиков;
- ✔ имеет развитую поддержку баз данных;
- ✔ имеется большое количество библиотек и расширений языка;
- ✔ может использоваться в изолированной среде;
- ✔ может быть развёрнут почти на любом сервере;
- ✔ поддерживается большим количеством аппаратных платформ и операционных систем.

Недостатки PHP

- ✔ не подходит для создания десктопных приложений или системных компонентов;
- ✔ имеет слабые средства для работы с исключениями;
- ✔ глобальные параметры конфигурации влияют на базовый синтаксис языка, что затрудняет настройку сервера и разворачивание приложений;
- ✔ возможны проблемы с безопасностью в разработанных веб-приложениях.

Д/З – История языка PHP

Редакторы кода PHP

1. Notepad ++
2. Sublime Text
3. PHP Storm
4. PHP Expert Editor
5. ByteRun Builder for PHP
6. ByteRun Editor for PHP
7. DzSoft PHP Editor
8. tsWebEditor
9. Davor's PHP Editor
10. PHP Coder

Способы размещения PHP-кода

1. Включение программного кода в HTML-документ.

<?php

программный код скрипта

?>

<?php – активирует php-интерпретатор

?> – деактивирует php-интерпретатор.

2. Размещение программного кода в отдельном файле с расширением .php.

Пример

```
<!DOCTYPE html>
<html>
   <head>
      <title>Пример</title>
   </head>
<body>
    <?php
         echo "Привет, я - скрипт PHP!";
    ?>
</body>
</html>
```

Запуск PHP-скрипта

В адресной строке браузера http://localhost/Primer/1.php

Комментарии

1. Однострочный комментарий

```
// Это комментарий
// echo "x равно $x";
$x += 10; // Увеличение значения $x на 10
```

2. Многострочный комментарий

```
<?php
/* Это область
многострочного комментария */
?>
```

Комментарии не могут быть вложенными друг в друга

Вывод результатов работы скрипта

1. **echo** – это специальная языковая конструкция, которая может принимать произвольное количество аргументов и выводить их

```
echo "Hello, world";
echo "Строка 1", "Строка 2";
```

2. **print** – специальная языковая конструкция, которая может принимать один параметр и выводить его.

```
print "Hello, world";
print 123;
```

Переменные

Переменные – это область оперативной памяти, используемая программой для хранения данных, доступ к которой осуществляется по имени.

Правила присваивания имен переменным

1. Все имена переменных в PHP начинаются со знака $.
2. Каждая переменная должна иметь уникальное имя в программе, состоящее из латинских букв, цифр и знаков подчеркивания.
3. Русские буквы лучше не использовать.
4. Имена переменных должны начинаться с буквы или с символа _ (подчеркивания).

Правила присваивания имен переменным

5. Имена переменных не должны включать в себя пробелы ($user_name).
6. Имена переменных чувствительны к регистру.
7. Не требуется явно описывать переменные и указывать их тип.

Пример

$x;

$strName;

$y1;

$_name;

Типы данных и инициализация переменных

I. Скалярные (примитивные)

- ✔ **boolean** – логический тип данных. Может содержать значения true или false;

$x = true;

Значения рассматриваются как FALSE:

- сам булев FALSE;
- целое 0 (ноль);
- число с плавающей точкой 0.0 (ноль);
- пустая строка и строка "0";
- массив с нулевыми элементами;
- объект с нулевыми переменными.

Все остальные значения рассматриваются как TRUE (включая любой ресурс).

- **integer** – целые числа

  (интервал от -2147483648 до 2147483647 и от 0 до 4294967295);
- **float** – числа с плавающей точкой

  $a = 1.234;

  $b = 1.2e3;

  $c = 7E-10;
- **string** – строка – набор символов любой длины

II. Смешанные типы

- ✔ **object** – для хранения экземпляров класса;
- ✔ **array** – используется для работы с массивами.

III. Специальные типы

- ✔ **resource** (ресурсы);

Ресурс – это специальная переменная, содержащая ссылку на внешний ресурс. Ресурсы создаются и используются специальными функциями.

- ✔ **NULL** ("пустой" тип).

Переменная считается NULL если:

- o ей была присвоена константа NULL;
- o ей еще не было присвоено какое-либо значение;
- o она была удалена с помощью unset().

IV. Псевдотипы

- ✔ **mixed** – смешанный тип;

  Параметр может принимать множество (но не обязательно все) типов.

- ✔ **number** – числовой;

  Параметр может быть либо integer, либо float.

Тип данных в PHP определяется значением переменной.

При инициализации переменной интерпретатор автоматически относит переменную к одному из типов данных.

Значение переменной присваивается с помощью **оператора =**

Пример

```
$number = 7;
$number2 = 7.8;
$string = "Строка";
$string2 = 'Строка';
$boolean = true;
```

Тип переменной может изменяться в соответствии с данными, хранящимися в ней

Функции для работы с переменными

1. Функция gettype(<Имя_переменной>) возвращает тип данных переменной

```
<?php
    $var = 7;
    echo gettype($var);       // Выведет: integer
    $str = 'Строка';
    echo gettype($str);       // Выведет: string
?>
```

2. Функции проверки конкретного типа переменных:

is_тип(<Переменая>)

is_int(<Переменная>)

is_integer(<Переменная>)

3. **Функция проверки существования переменной isset(<Переменная>)**
4. **Функция проверки наличия у переменной непустого, ненулевого значения empty(<Переменная>)**

Следующие значения воспринимаются как пустые:

- ✔ "" (пустая строка)
- ✔ 0 (целое число)
- ✔ 0.0 (число с плавающей точкой)
- ✔ "0" (строка)
- ✔ NULL
- ✔ FALSE
- ✔ array() (пустой массив)
- ✔ $var; (переменная объявлена, но не имеет значения)

5. **Функции изменения типа переменной**

- ✔ **intval**() – возвращает аргумент в виде целого числа integer;
- ✔ **floatval**() – возвращает аргумент в виде дробного числа float;
- ✔ **strval**() – возвращает аргумент в виде строки string;
- ✔ **settype**() – превращает первый аргумент в указанный во втором аргументе тип

6. **Функция удаления переменной unset(<Переменная>)**

Константы. Создание и использование констант

Константы – это именованная величина, используемая для хранения значений, которая не должна изменяться во время работы программы (скрипта).

Такими значениями могут быть математические константы, пути к файлам, пароли и т.д.

Создание константы

Функция define():

define(<Имя константы>,<Значение константы>[, <Регистр>]);

Пример

define("pi",3.14,true);

Правила работы с константами:

1. У констант нет приставки в виде знака доллара ($).
2. Константы можно определить только с помощью функции define(), а не присваиванием значения.
3. Константы могут быть определены и доступны в любом месте без учета области видимости.
4. Константы не могут быть определены или аннулированы после первоначального объявления.
5. Константы могут иметь только скалярные значения (логического, целого, плавающего и строкового типов).

Виды констант:

1. **Определенные разработчиком.**
2. **Предопределенные константы**
   A. **меняющие значение в зависимости от контекста, в котором они используются**

- ✔ **__FILE__** – содержит месторасположение скрипта и его имя;
- ✔ **__LINE__** – содержит номер строки, которую обрабатывает интерпретатор в данный момент;
- ✔ **__FUNCTION__** – имя функции;
- ✔ **__CLASS__** – имя класса;
- ✔ **__METHOD__** – имя метода класса.

B. Стандартные константы.

Объявлены в PHP по умолчанию.

C. Объявленные в ядре PHP.

Объявлены в ядре PHP.

✔ **PHPVERSION** – содержит версию PHP и другие.

Операторы PHP

Оператор состоит из одного или более выражений, которое можно вычислить как новое значение.

Операторы позволяют выполнить определенные действия с переменными.

Математические операторы

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

Операторы инкремента и декремента

++$a **Префиксный инкремент** – увеличивает $a на и возвращает значение $a.

$a++ **Постфиксный инкремент** – возвращает значение $a, а затем увеличивает $a на 1.

--$a **Префиксный декремент** – уменьшает $a на 1 и возвращает значение $a.

$a-- **Постфиксный декремент** – возвращает значение $a, а затем уменьшает $a на 1.

Пример

```
$a = 5;
echo $a++;
```

Логические операторы PHP

Предназначены для работы с логическими выражениями и возвращают **false** или **true**.

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

Операторы присваивания

1. **оператор «=»;**

Пример

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

2. **комбинированные операторы;**

+= $a+=1 $a=$a+1

- = $a-=1 $a=$a-1

/ = $a/=$b $a=$a/$b

***=** $a*=$b $a=$a*$b

Операторы сравнения

Позволяют сравнивать между собой два значения.

Возвращают одно из двух: **false** или **true**.

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.
$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
$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.

Оператор управления ошибками @

Оператор используется перед выражениями в PHP-коде, любые сообщения об ошибках, генерируемые этим выражением, будут проигнорированы.

Строковые операторы

Оператор конкатенации «.» служит для объединения строк.

Пример

```
<?php
$fam="Петров";
$fi="Ваша фамилия - ".$fam;
echo $fi;
?>
```