PHP

Hypertext Preprocessor

Язык программирования PHP

PHP: Hypertext Preprocessor

- Один из наиболее популярных и широко-распространённых языков программирования для Web
- PHP – язык программирования с открытым исходным кодом (open source programming language)
- PHP выполняется на сервере
- PHP можно бесплатно скачать и начать использовать

Интересные факты

- Многие крупные интернет проекты написаны на PHP

PHP файл

- Может содержать:
  - текст
  - HTML
  - CSS
  - JavaScript
  - PHP
- PHP выполняется на сервере
- Результат выполнения PHP скрипта – HTML код
- PHP файлы – файлы с расширением ***.php***

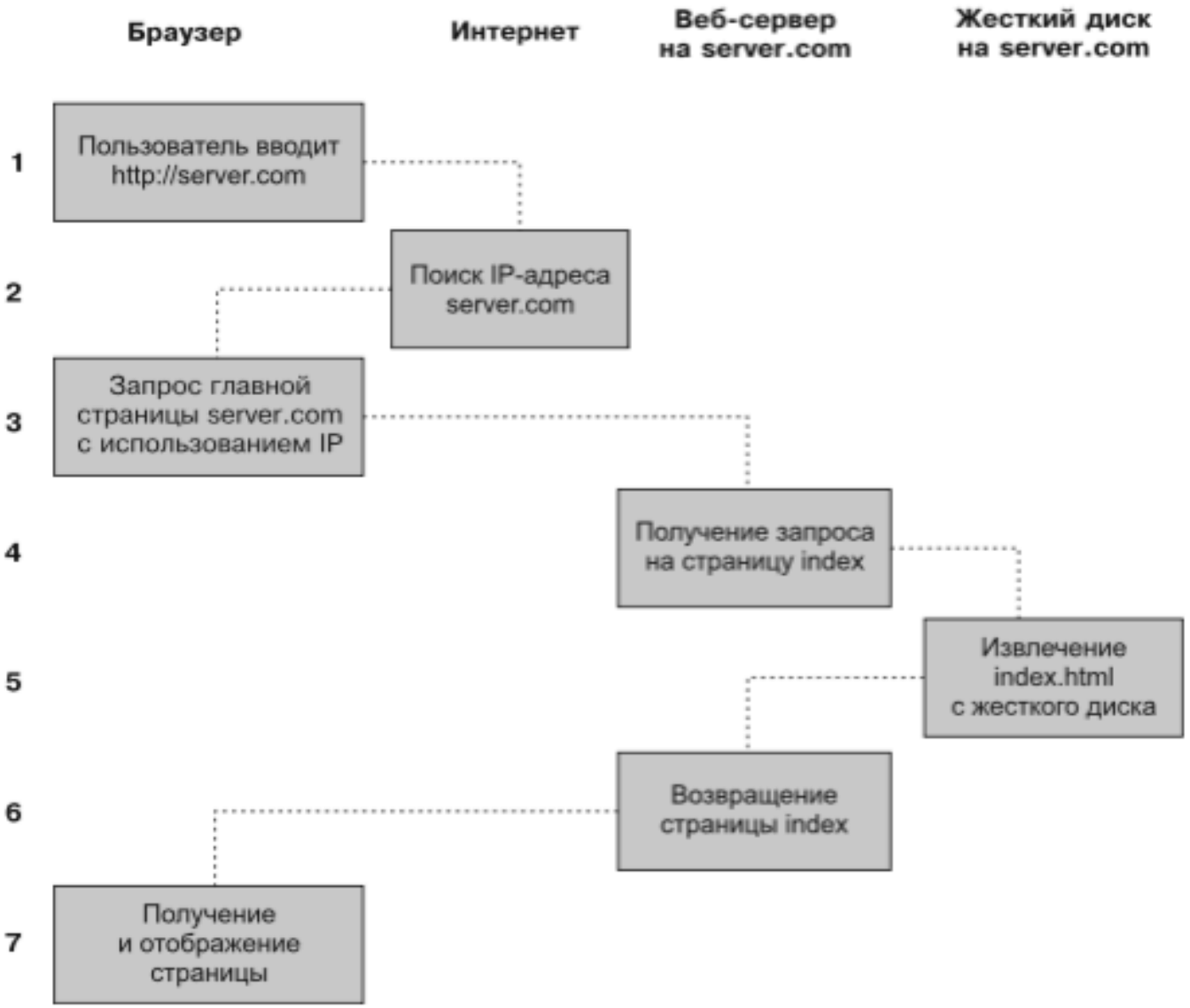
Возможности PHP

- Генерация динамического содержимого веб-страниц
- Создание, удаление, чтение и запись файлов на сервере
- Обработка данных из веб-форм
- Получение и отправка cookies
- Создание, удаление, чтение и изменение записей в базах данных
- Проверка доступа пользователей к контенту
- Шифрование данных

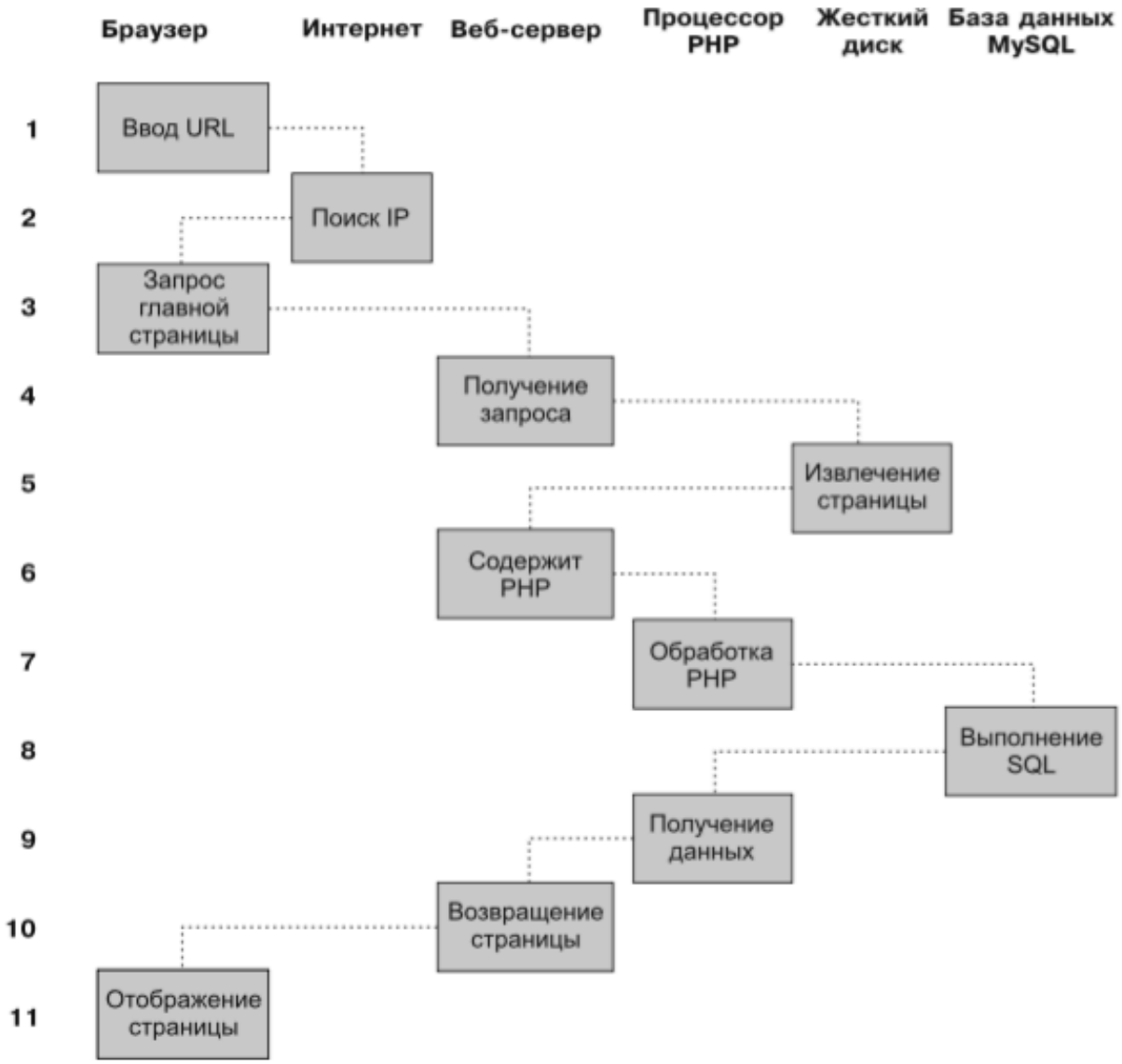
Преимущества PHP

- Кроссплатформенный (Windows, Linux, UNIX, Mac OS)
- Совместим с веб-серверами (Apache, IIS, Nginx)
- Поддерживает различные базы данных
- Бесплатный
- Открытый исходный код и большое сообщество
- Прост в изучении
- Достаточно эффективно выполняется на серверах

Процедура «запрос-ответ»



Процедура «запрос-ответ»



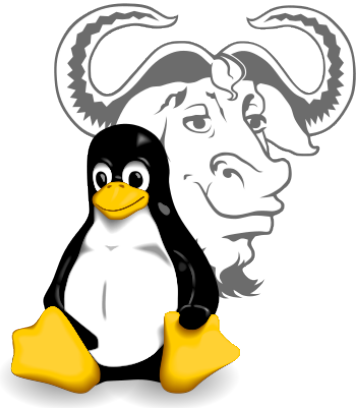
Установка PHP

- Варианты:
  - Использовать веб-хостинг с поддержкой PHP и MySQL
  - Установить веб-сервер, PHP и MySQL на компьютер

- Установка PHP на компьютере
  - Установить веб-сервер
  - Установить PHP
  - Установить MySQL

Сервер для разработки

- WAMP
  - Windows
  - Apache
  - MySQL
  - PHP
- LAMP
  - Linux
  - …
- MAMP
  - Mac OS
  - …



WAMP

- http://www.denwer.ru
- easyphp.org
- http://www.apachefriends.org/ru/xampp.html
- http://www.wampserver.com/ru/
- http://glossword.biz/glosswordwamp/
- http://www.microsoft.com/web/webmatrix/

Проверка установки

- http://localhost/
- http://127.0.0.1

Работа с удаленным сервером

- Вход в систему
  - Протокол ***ssh***
  - Программа ***Putty***

- Работа с файлами
  - Протокол ***ftp***
  - Программа ***TotalCommander*** и др.

Текстовые редакторы

- Notepad (Windows)

- Sublime

- Notepad++

- TextEdit (Mac)

Основы PHP синтаксиса

- PHP скрипт выполняется на сервере, результат в виде HTML страницы отправляется в браузер

```
<!DOCTYPE html>
<html>
    <body>

        <h1>Моя первая страница с PHP</h1>

        <?php
        echo "Hello World!";
        ?>

    </body>
</html>
```

Комментарии

```
<!DOCTYPE html>
<html>
    <body>
        <?php
            // Однострочный комментарий

            # Однострочный комментарий

            /*
            Многострочный комментарий
            */

            $x = 5 /* + 15 */ + 5;
            echo $x;
        ?>
    </body>
</html>
```

Чувствительность к регистру

- Ключевые слова не чувствительны к регистру

```
<?php
    ECHO "Hello World!<br>";
    echo "Hello World!<br>";
    EcHo "Hello World!<br>";
?>
```

- Имена переменных чувствительны к регистру

```
<?php
    $color = "red";
    echo "My car is " . $color . "<br>";
    echo "My house is " . $COLOR . "<br>";
    echo "My boat is " . $coLOR . "<br>";
?>
```

Определение переменной

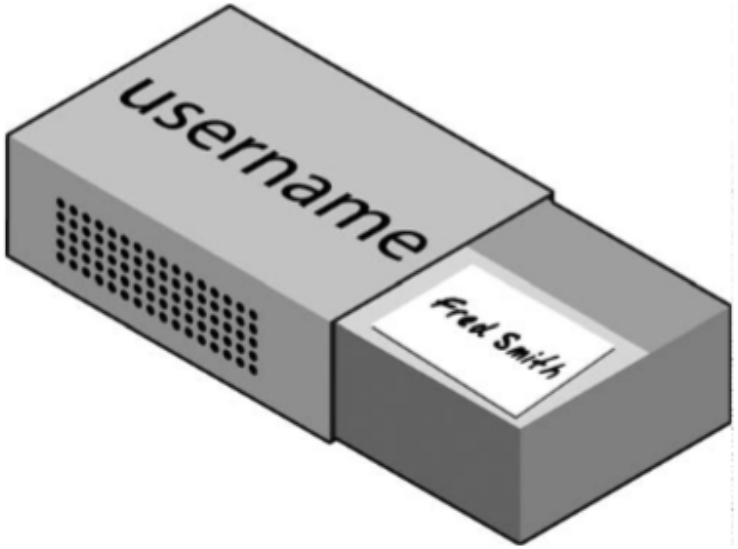
- **Переменная** – поименованная, либо адресуемая иным способом область памяти, адрес которой можно использовать для осуществления доступа к данным.

- **Значение переменной** – данные, находящиеся в переменной (то есть по данному адресу памяти).

Объявление переменной

- Переменная – «контейнер» для хранения информации
- Переменная начинается с символа ***$***, далее идет наименование переменной

```
<?php
    $txt = "Hello world!";
    $x = 5;
    $y = 10.5;
    $username = "Fred Smith";
?>
```



Правила присваивания имен переменных

- Переменная начинается с символа ***$***, далее идет наименование переменной
- Наименование переменной **начинается с буквы латинского алфавита** или **с символа «_»**
- Имена переменных могут **содержать символы «a-z, A-Z, 0-9, _»**
- Имена переменных не должны содержать пробелов, если **имя составляет два слова**, то в качестве разделителя используется «_» или «**camelCase**»
- Имена переменных **чувствительны к регистру**: $x и $X – разные переменные

Вывод значений переменных

```
<?php
    $txt = "Информатика";

    echo "Изучаемый предмет $txt!";

    echo "Изучаемый предмет" . $txt . "!";

    $x = 5;
    $y = 4;
    echo $x + $y;
?>
```

Вывод

- ***print***
  - принимает на вход 1 параметр
  - возвращает значение 1
  - работает медленнее, чем ***echo***

- ***echo***
  - принимает на вход множество параметров
  - не возвращает значение
  - работает быстрее, чем ***print***

Типы данных

- String – строки
- Integer – целые числа
- Float – вещественные числа
- Boolean – логическая переменная (***true*** или ***false***)
- Array – массивы
- Object – объекты
- NULL – переменная без значения

Строки

- Строка – последовательность символов

```
<?php
    $x = "Hello world!";
    $y = 'Hello world!';

    echo $x;
    echo "<br>";
    echo $y;
?>
```

Целые числа

- Размер: от -2 147 483 648 до +2 147 483 647
- Содержит хотя бы одну цифру
- Не содержит запятых, точек и пробелов
- Может быть положительным и отрицательным
- Формат
  - десятичные
  - шестнадцатеричные
  - восьмеричные

```
<?php
    $x = 5985;
    var_dump($x);
?>
```

Вещественные числа

- Формат: число с точкой или в формате с экспонентой

```
<?php
    $x = 10.365;
    var_dump($x);
?>
```

Логические переменные

- Используются в логических выражения и условных конструкциях

```
$x = true;
$y = false;
```

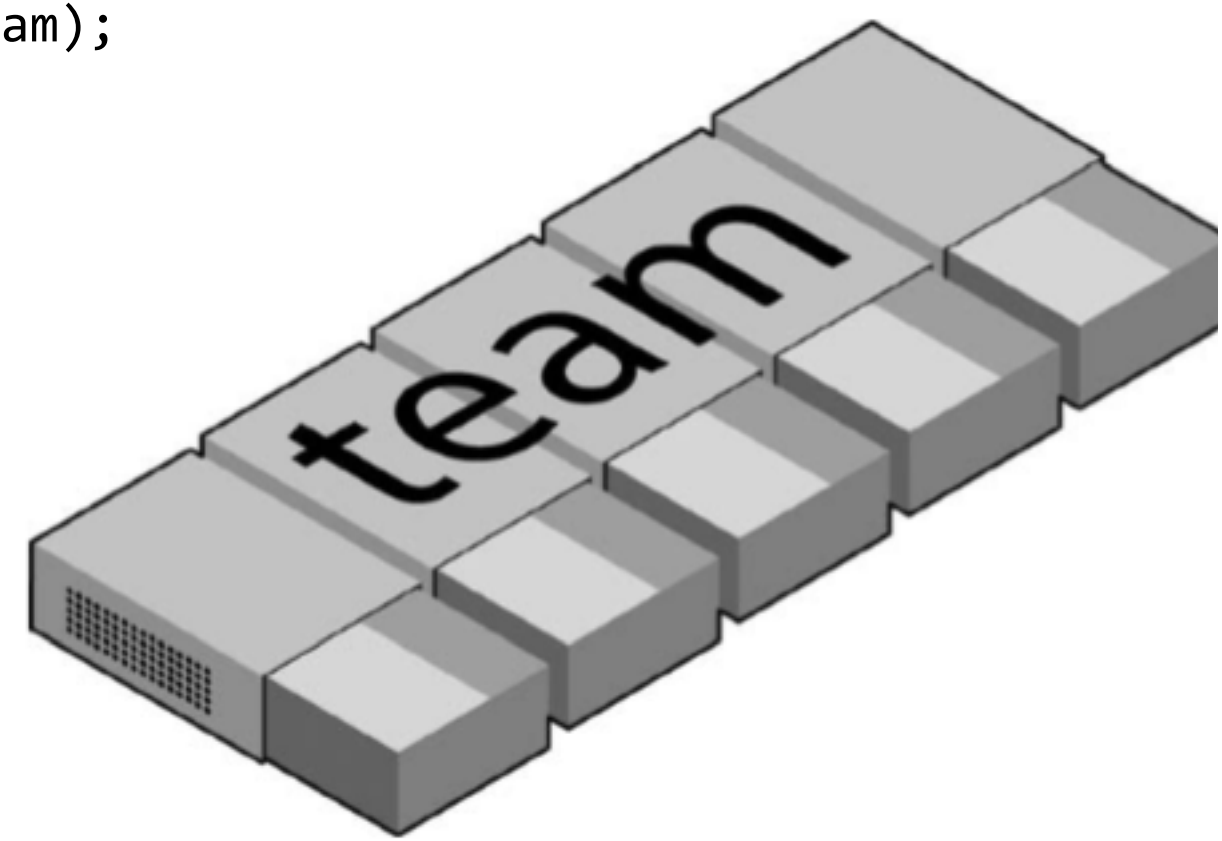
Осмысление переменных. Массивы

- **Массив** – именованный набор однотипных переменных, расположенных в памяти непосредственно друг за другом, доступ к которым осуществляется по индексу (в отличие от списка).
- **Индекс массива** — целое число, либо значение типа, приводимого к целому, указывающее на конкретный элемент массива.
- Массивы:
  - Одномерные
  - Двумерные
  - Многомерные

Массивы

- Множество значений в рамках одной переменной

```
<?php
    $team = array("Jim", "John", "Bill");
    var_dump($team);
?>
```



Функции для работы со строками

- Длина строки

```
echo strlen("Hello world!"); // 12
```

- Количество слов

```
echo str_word_count("Hello world!"); // 2
```

- Реверс строки

```
echo strrev("Hello world!"); // вывод !dlrow olleH
```

- Поиск текста

```
echo strpos("Hello world!", "world"); // 6
```

- Замена текста

```
echo str_replace("world", "Jim", "Hello world!");
// вывод Hello Jim!
```

Константы

- Синтаксис

```
define(name, value, case-insensitive)
```

- Пример

```
// Правильно
define("FOO",     "something");
define("FOO2",    "something else");
define("FOO_BAR", "something more");

// Ошибка
define("2FOO",    "something");
```

- Константы являются **глобальными** для всего скрипта

Предопределенные константы

- __LINE__
- __FILE__
- __DIR__
- __FUNCTION__
- __CLASS__
- __METHOD__
- __NAMESPACE__

Операторы. Operators

- **Оператор** – наименьшая автономная часть языка программирования.

Арифметические операторы. Arifmetical

Оператор	Операция	Пример
+	Сложение	`$x == $y`
-	Вычитание	`$x == $y`
/	Деление	`$x == $y`
*	Умножение	`$x == $y`
%	Целочисленное деление	`$x == $y`

Операторы присвоения. Assignment

Пример	Математический аналог
`$x = $y`	x = y
`$x += $y`	x = x + y
`$x -= $y`	x = x - y
`$x *= $y`	x = x * y
`$x /= $y`	x = x / y
`$x %= $y`	x = x % y

Операторы сравнения. Comparison

Оператор	Наименование	Пример
==	Равно	`$x == $y`
===	Тождественно равно	`$x === $y`
!=	Не равно	`$x != $y`
<>	Не равно	`$x <> $y`
!==	Тождественно не равно	`$x !== $y`
>	Больше	`$x > $y`
<	Меньше	`$x < $y`
>=	Больше или равно	`$x >= $y`
<=	Меньше или равно	`$x <= $y`

Операторы инкремента и декремента

Оператор	Наименование
`++$x`	Префиксная запись инкремента
`$x++`	Постфиксная запись инкремента
`--$x`	Префиксная запись декремента
`$x--`	Постфиксная запись декремента

Логические операторы. Logical

Оператор	Наименование	Пример
and	И	`$x and $y`
or	ИЛИ	`$x or $y`
xor	Исключающее ИЛИ	`$x xor $y`
&&	И	`$x && $y`
ll	ИЛИ	`$x || $y`
!	НЕ	`!$x`

Логические операторы. Таблица истинности

Входные данные		Операторы и результаты		
a	**b**	**AND**	**OR**	**XOR**
TRUE	TRUE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE	TRUE
FALSE	TRUE	FALSE	TRUE	TRUE
FALSE	FALSE	FALSE	FALSE	FALSE

Операторы для работы со строками

Оператор	Наименование	Пример
.	Конкатенация	`$x . $y`
.=	Конкатенация с присвоением	`$x .= $y`

Операторы для работы с массивами

Оператор	Наименование	Пример
+	Объединение	`$x + $y`
==	Равно	`$x == $y`
===	Тождественно равно	`$x === $y`
!=	Не равно	`$x != $y`
<>	Не равно	`$x <> $y`
!==	Тождественно не равно	`$x !== $y`

Приоритет операторов

Оператор (-ы)	Тип
()	Скобки
++ —	Инкремент/декремент
!	Логический
* / %	Арифметические
+ -	Арифметические и строковые
<< >>	Побитовые
< <= > >= <>	Сравнения
== != === !==	Сравнения
&	Поразрядный (и ссылочный)
^	Поразрядный
|	Поразрядный
&&	Логический
||	Логический
? :	Трехкомпонентный
= += –= *= /= .= %= &= != ^= <<= >>=	Присваивания
and	Логический
xor	Логический
or	Логический

Операторы

- Унарные операторы
  - Декремент
  - Инкремент
- Бинарные операторы
  - Арифметические
  - Логические
- Трехкомпонентный оператор
  - ? X : Y

Выражения

- **Выражение** – сочетание значений, переменных, операторов и функций, в результате которого вычисляется новое значение.

- **Литерал** – простейшее выражение, вычисляющееся само в себя.

Условия. Conditional Statements

- **Условный переход** – команда на изменение порядка выполнения программы в соответствии с результатом проверки некоторого условия
- **if выражение** – код выполняется, если условие истинно
- **if...else выражение** – если условие истинно выполняется один код, если ложно - другой
- **if...elseif....else выражение** – проверка условия, если предыдущее ложно
- **switch выражение** – выбор одного блока кода для выполнения

if выражение

- Синтаксис

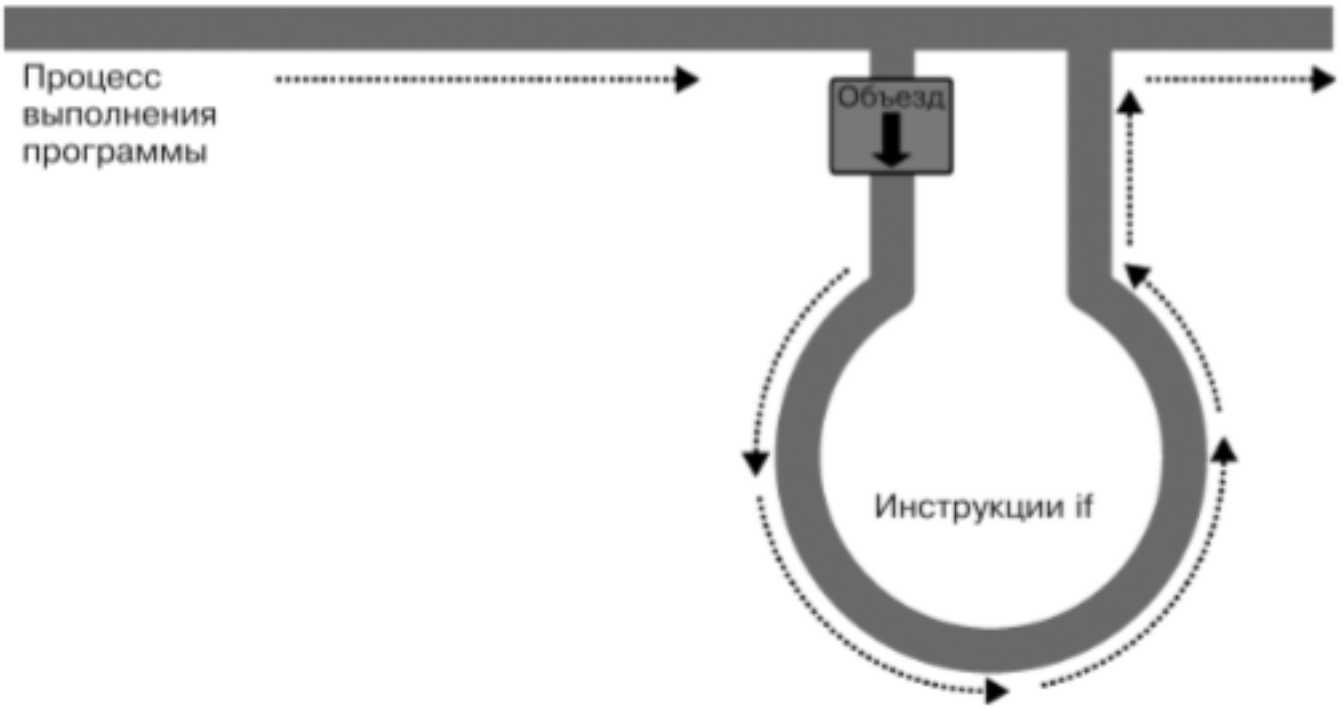
```
if (условие) {
    код выполняется, если условие истинно;
}
```

- Пример

```
$t = date("H");

if ($t < "20") {
    echo "Хорошего дня!";
}
```

if выражение

- Аналогия



if...else выражение

- Синтаксис

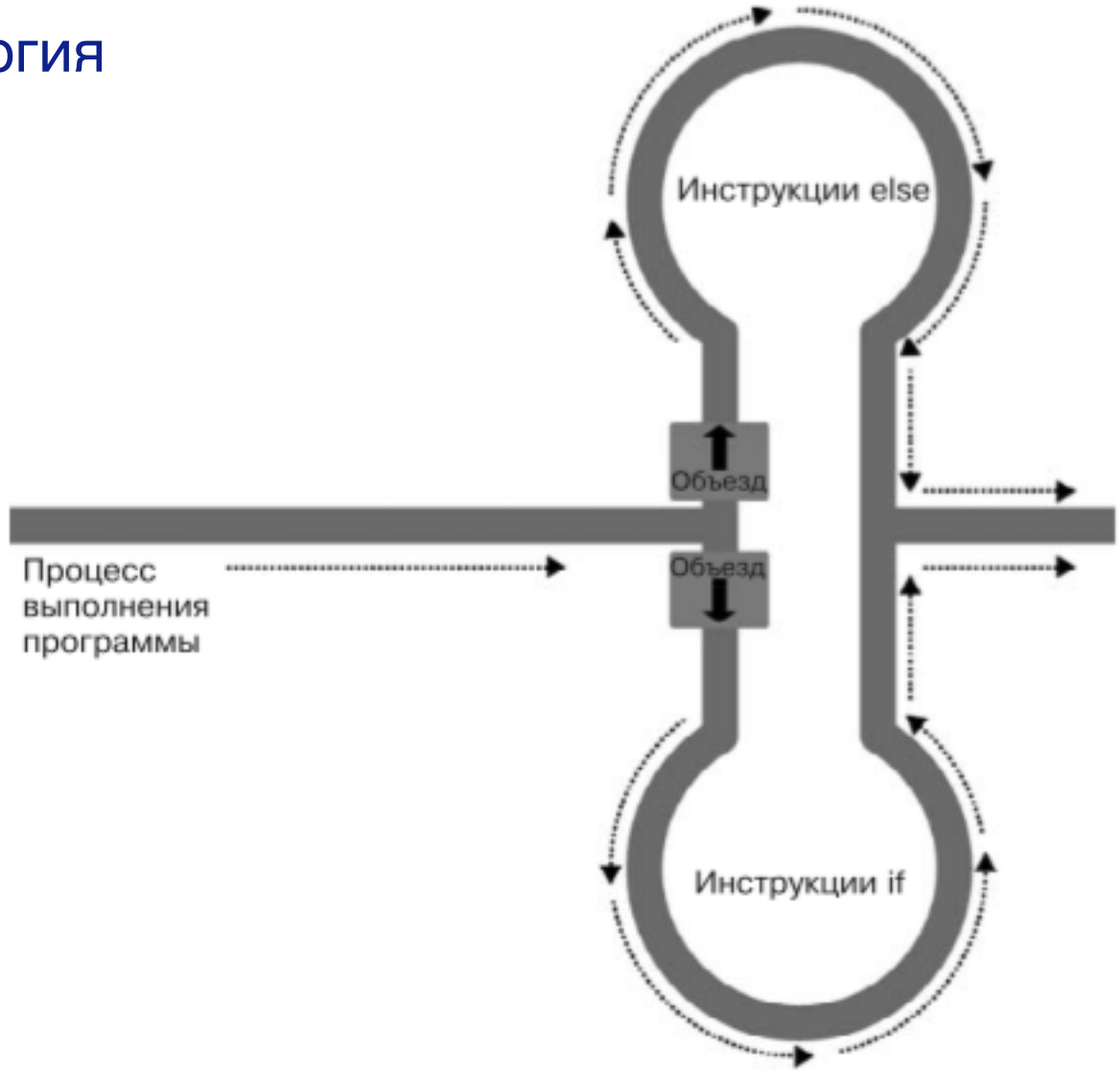
```
if (условие) {
    код выполняется, если условие истинно;
} else {
    код выполняется, если условие ложно;
}
```

- Пример

```
$t = date("H");

if ($t < "17") {
    echo "Добрый день!";
} else {
    echo "Добрый вечер!”;
}
```

if...else выражение

- Аналогия



if...elseif....else выражение

- Синтаксис

```
if (условие) {
    код выполняется, если условие истинно;
} elseif (условие) {
    код выполняется, если условие истинно;
}
} else {
    код выполняется, если условие ложно;
}
```

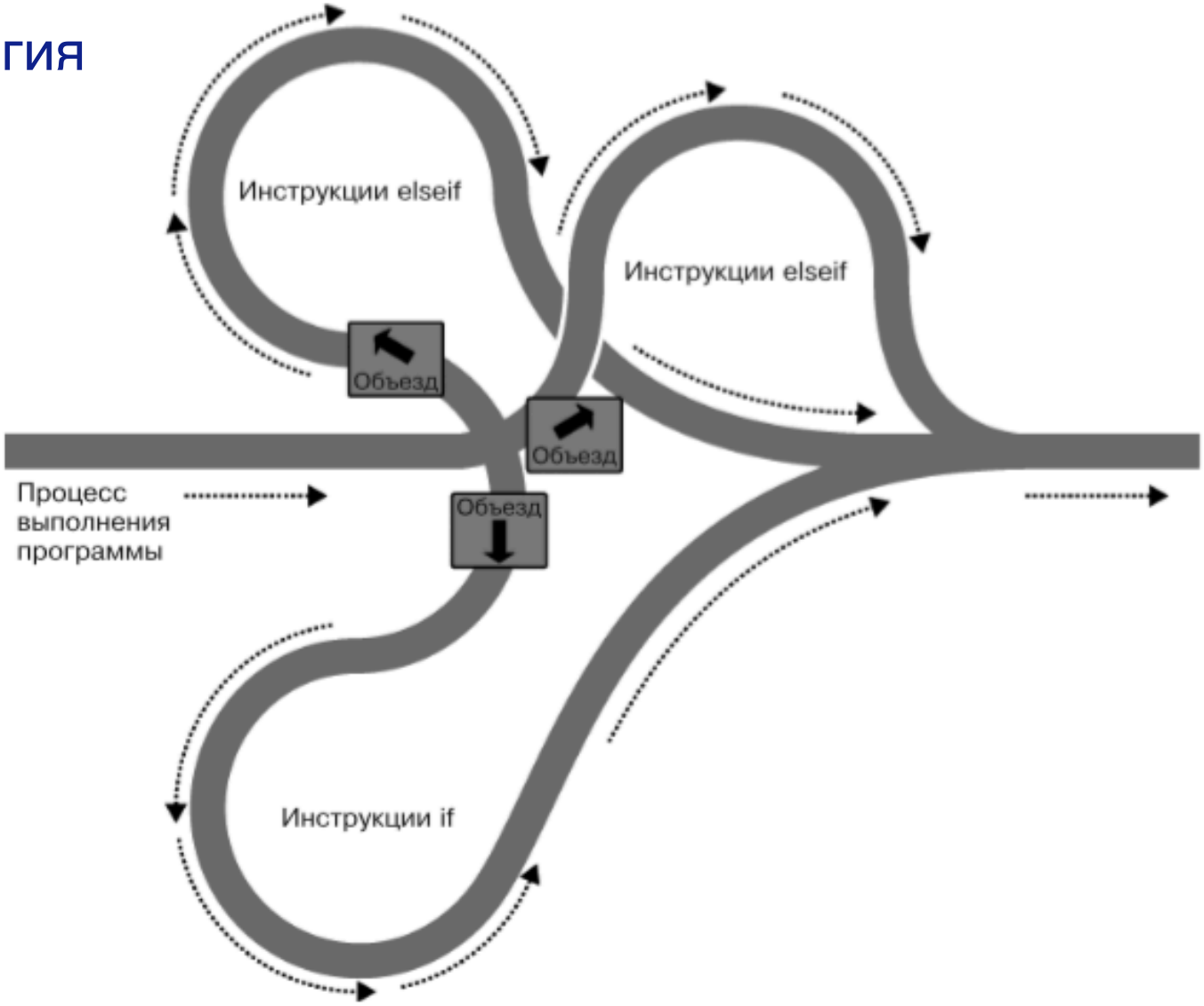
if...elseif....else выражение

- Пример

```
$t = date("H");

if ($t < "12") {
    echo "Доброе утро!”;
} elseif ($t < "17") {
    echo "Добрый день!";
} else {
    echo "Добрый вечер!”;
}
```

if...elseif....else выражение

- Аналогия



switch выражение

- Синтаксис

```
switch (n) {
    case значение1:
        код выполняется, если n=значение1;
        break;
    case значение2:
        код выполняется, если n=значение2;
        break;
    case значение3:
        код выполняется, если n=значение3;
        break;
    ...
    default:
        код выполняется, если n не совпало со значениями;
}
```

switch выражение

- Пример

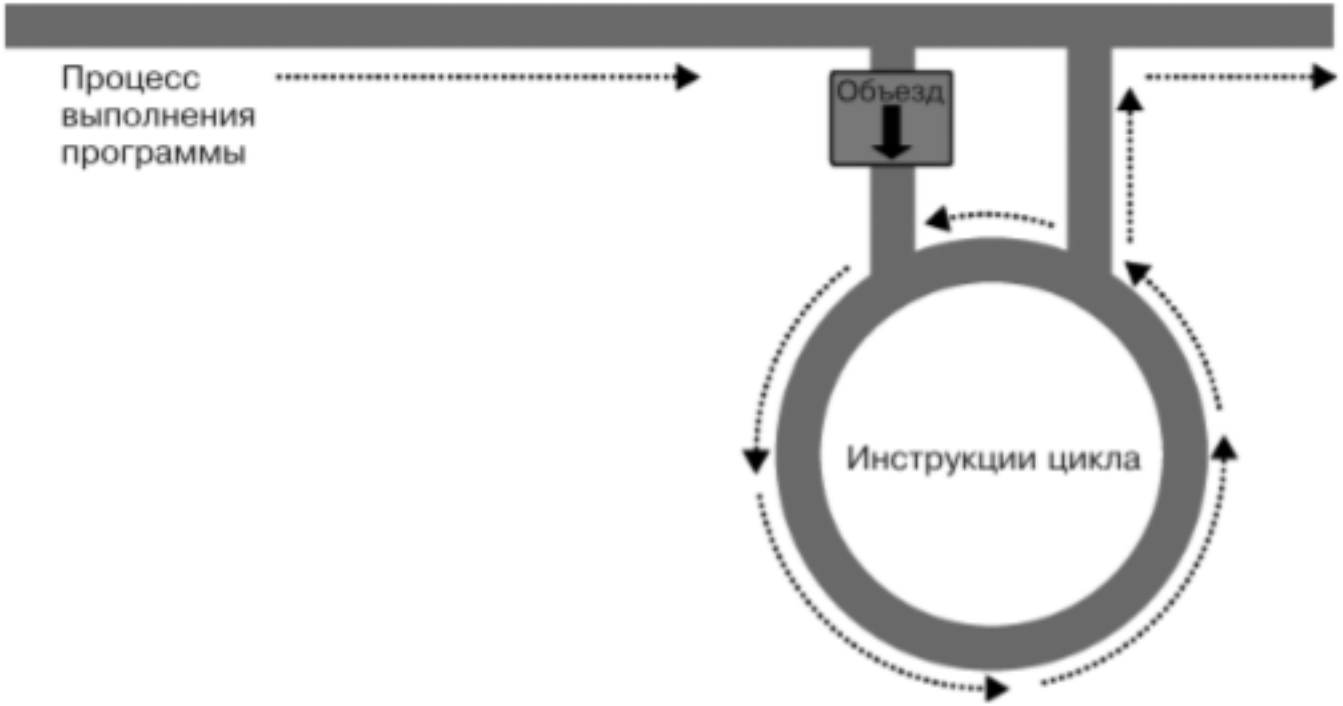
```
$favoriteColor = "красный";

switch ($favoriteColor) {
    case "красный":
        echo "Любимый цвет красный!";
        break;
    case "синий":
        echo "Любимый цвет синий!";
        break;
    case "зеленый":
        echo "Любимый цвет зеленый!";
        break;
    default:
        echo "Любимый цвет не известен!";
}
```

Циклы

- **Цикл** — разновидность управляющей конструкции, предназначенная для организации многократного исполнения набора инструкций.
- **Тело цикла** – последовательность инструкций, предназначенная для многократного исполнения
- **Итерация** – единичное выполнение тела цикла называется итерацией.
- **Условие выхода** – выражение, определяющее, будет в очередной раз выполняться итерация, или цикл завершится
- **Счетчик цикла** – переменная, хранящая номер текущей итерации цикла

Циклы

- Аналогия



Виды циклов

- **Безусловный цикл** – циклы, выход из которых не предусмотрен логикой программы.
- **Цикл с предусловием** – цикл, который выполняется пока истинно некоторое условие, указанное перед его началом.
- **Цикл с постусловием** – цикл, в котором условие проверяется после выполнения тела цикла.
- **Цикл со счетчиком** – цикл, в котором некоторая переменная изменяет своё значение от заданного начального значения до конечного значения с некоторым шагом, и для каждого значения этой переменной тело цикла выполняется один раз.

Виды циклов

- **while** – тело цикла выполняется, пока условие истинно
- **do...while** – тело цикла выполняется один раз и повторяется до тех пор, пока условие истинно
- **for** – тело цикла выполняется определенное количество раз
- **foreach** – тело цикла выполняется для каждого элемента в массиве

Цикл while

- Синтаксис

```
while (условие) {
    тело цикла;
}
```

- Пример

```
$x = 1;

while($x <= 5) {
    echo "Переменная x равна: $x <br>";
    $x++;
}
```

Цикл do…while

- Синтаксис

```
do {
    тело цикла;
} while (условие);
```

- Пример

```
$x = 6;

do {
    echo "Переменная x равна: $x <br>";
    $x++;
} while($x <= 5);
```

Цикл for

- Синтаксис

```
for (инициализация счетчика;
       проверка счетчика;
       инкремент счетчика) {
    тело цикла;
}
```

- Пример

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

Цикл foreach

- Синтаксис

```
foreach ($массив as $значение) {
    тело цикла;
}
```

- Пример

```
$colors = array("красный",
       "зеленый",
       "синий",
       "желтый");

foreach ($colors as $value) {
    echo "$value <br>";
}
```

Функции

- **Функция** – набор инструкций, который позволяет выполнять конкретную задачу и в дополнение к этому может вернуть какое-либо значение
- Функция – блок кода, который можно выполнить несколько раз
- Функции не выполняются при загрузки страницы
- Функции выполняется только при их вызове

Преимущества функций

- Экономия времени программирования
- Сокращение количества ошибок
- Сокращение времени выполнения
- Возможность использовать функции в разных случаях

Функции

- Синтаксис

```
function наименованиеФункции() {
    тело функции;
}
```

- Пример

```
function writeMsg() {
    echo "Hello world!";
}

writeMsg(); // вызов функции
```

- Наименование функции начинается с буквы или символа «_» и не чувствительно к регистру

Параметры функции

- Пример

```
function familyName($fname, $year) {
    echo "$fname Иванов. Родился в $year <br>";
}

familyName("Иван", "1975");
familyName("Петр", "1978");
familyName("Василий", "1983");
```

- ***$fname, $year*** – параметры функции ***familyName***

Параметры функции

- Пример

```
function setHeight($minheight = 50) {
    echo "Высота равна: $minheight <br>";
}

setHeight(350);

// будет использовано значение по умолчанию - 50
setHeight();

setHeight(135);
setHeight(80);
```

Возврат значений

- Пример

```
function sum($x, $y) {
    $z = $x + $y;
    return $z;
}

echo "5 + 10 = " . sum(5, 10) . "<br>";
echo "7 + 13 = " . sum(7, 13) . "<br>";
echo "2 + 4 = " . sum(2, 4);
```

- Функция `sum` возвращает результат сложения параметров `$x, $y`

Область видимости переменных

- Локальные переменные
- Глобальные переменные
- Статические переменные

Глобальные и локальные переменные

```
<?php
    $x = 5; // глобальная переменная

    function myTest() {
        // Ошибка
        echo "<p>Переменная x внутри функции $x</p>";
    }
    myTest();

    echo "<p>Переменная x вне функции: $x</p>";
?>
```

Глобальные и локальные переменные

```
<?php
    function myTest() {
        $x = 5; // локальная переменная
        echo "<p>Переменная x внутри функции $x</p>";
    }
    myTest();

    // Ошибка
    echo "<p>Переменная x вне функции: $x</p>";
?>
```

Глобальные переменные

```php
<?php
    $x = 5;
    $y = 10;

    function myTest() {
        global $x, $y;
        $y = $x + $y;
    }

    myTest();
    echo $y;
?>
```

```php
<?php
    $x = 5;
    $y = 10;

    function myTest() {
        $GLOBALS['y'] =
            $GLOBALS['x'] +
            $GLOBALS['y'];
    }

    myTest();
    echo $y;
?>
```

Статические переменные

- Статические переменные сохраняют свое значение между вызовами функции

```
<?php
    function myTest() {
        static $x = 0;
        echo $x;
        $x++;
    }

    myTest();
    myTest();
    myTest();
?>
```

Массивы. Arrays

- Хранят множество значений в рамках одной переменной
- Пример

```
$cars = array("Volvo", "BMW", "Toyota");
echo "Мне нравится" . $cars[0] . ", "
       . $cars[1] . " и " . $cars[2] . ".";
```

- Альтернатива

```
$cars1 = "Volvo";
$cars2 = "BMW";
$cars3 = "Toyota";
```

Виды массивов

- **Indexed arrays** – массивы с целочисленным индексом
- **Associative arrays** – ассоциативные массивы, индекс в массиве – некий ключ
- **Multidimensional arrays** – многомерные массивы, массивы элементы, которых могут быть массивами

Массив с численной индексацией

- Пример

```
$cars = array("Volvo", "BMW", "Toyota");

$cars[0] = "Volvo";
$cars[1] = "BMW";
$cars[2] = "Toyota";

echo "Мне нравится" . $cars[0] . ", "
      . $cars[1] . " и " . $cars[2] . ".";
```

- Размерность массива

```
echo count($cars);
```

Цикл по массиву с численной индексацией

- Пример

```
$cars = array("Volvo", "BMW", "Toyota");

$arrayLength= count($cars);

for($x = 0; $x < $arrayLength; $x++) {
    echo $cars[$x];
    echo "<br>";
}
```

Ассоциативный массив

- Пример

```
$age = array("Иван"=>"35", "Петр"=>"37", "Игорь"=>"43");
echo "Возраст Ивана " . $age['Иван'] . " лет.";
```

- Цикл по ассоциативному массиву

```
foreach($age as $key => $value) {
    echo "Ключ=" . $key . ", Значение=" . $value;
    echo "<br>";
}
```

Многомерные массивы

- Аналогия

Сортировка массива

- **sort()** – сортировка в возрастающем порядке
- **rsort()** – сортировка в убывающем порядке
- **asort()** – сортировка ассоциативного массива по возрастанию по значению
- **ksort()** – сортировка ассоциативного массива по возрастанию по ключу
- **arsort()** – сортировка ассоциативного массива по убыванию по значению
- **krsort()** – сортировка ассоциативного массива по убыванию по ключу

Суперглобальные переменные

- $GLOBALS
- $_SERVER
- $_REQUEST
- $_POST
- $_GET
- $_FILES
- $_ENV
- $_COOKIE
- $_SESSION