

PHP Уровень 1.

Основы создания сайтов

Игорь Борисов
<http://igor-borisov.ru>

Темы курса

- Установка/настройка веб-сервера и PHP
- Основы PHP
- Циклы
- Пользовательские функции
- Что внутри PHP?
- Изучаем HTTP: формы

Я ожидаю от вас...

- Знания HTML/CSS
- Умения работать с компьютером
- Умения создавать, перемещать и удалять файлы и папки
- Умения работать в текстовом редакторе
- Умения работать с клавиатурой
 - быстро
 - знание сочетаний клавиш
 - Ctrl+N, Ctrl+O, Ctrl+S
 - Ctrl+A, Ctrl+C, Ctrl+X, Ctrl+V, Ctrl+Z
 - Shift+Home, Shift+End, Shift+Ctrl+стрелки

PHP Уровень 1.

Установка/настройка

веб- сервера и PHP

Игорь Борисов
<http://igor-borisov.ru>

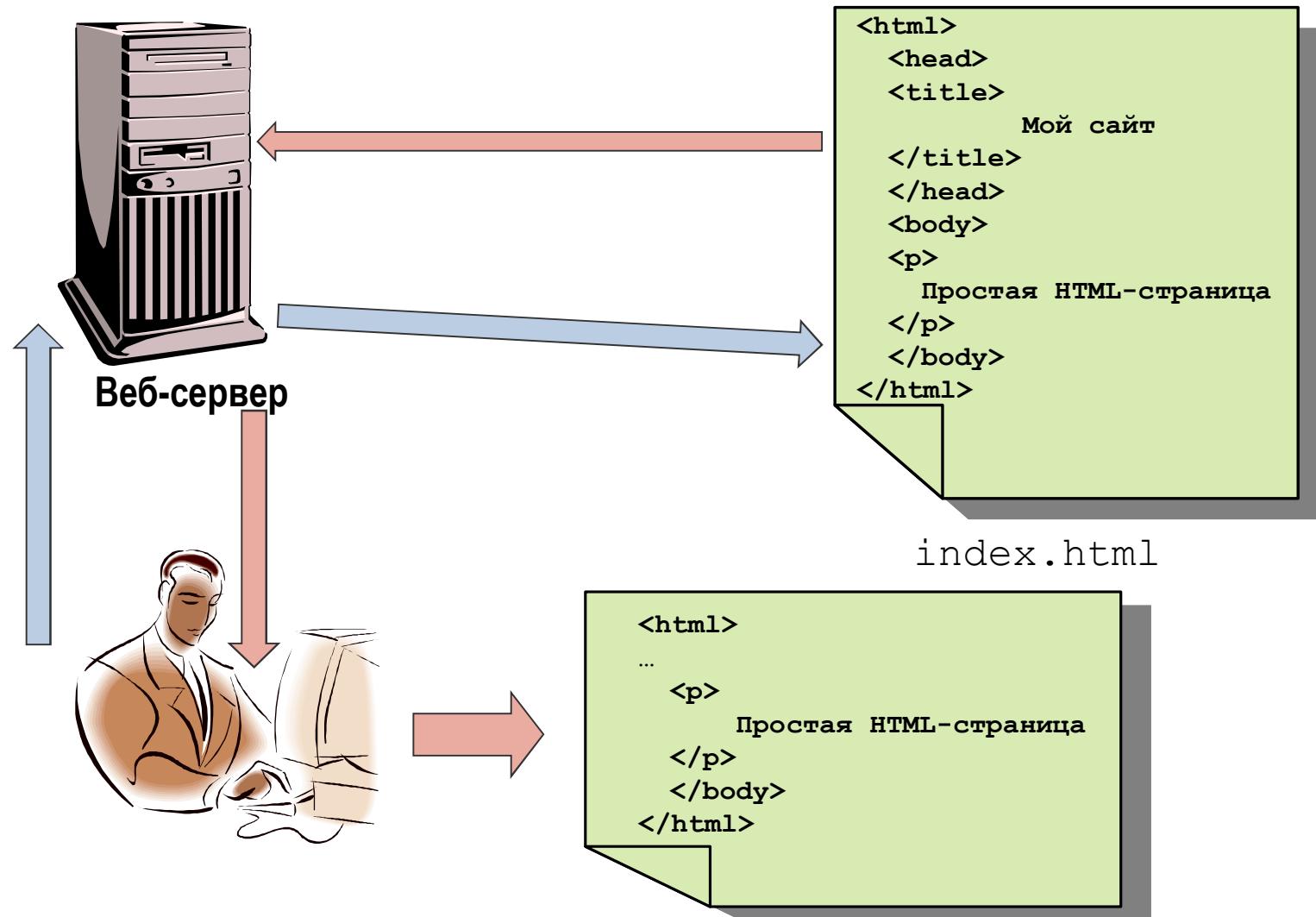
Темы модуля

- Вопросы, на которые надо ответить
- Как это работает?
- Установка веб-сервера
- Настройка веб-сервера
- Установка PHP
- Настройка PHP
- Первый скрипт на PHP

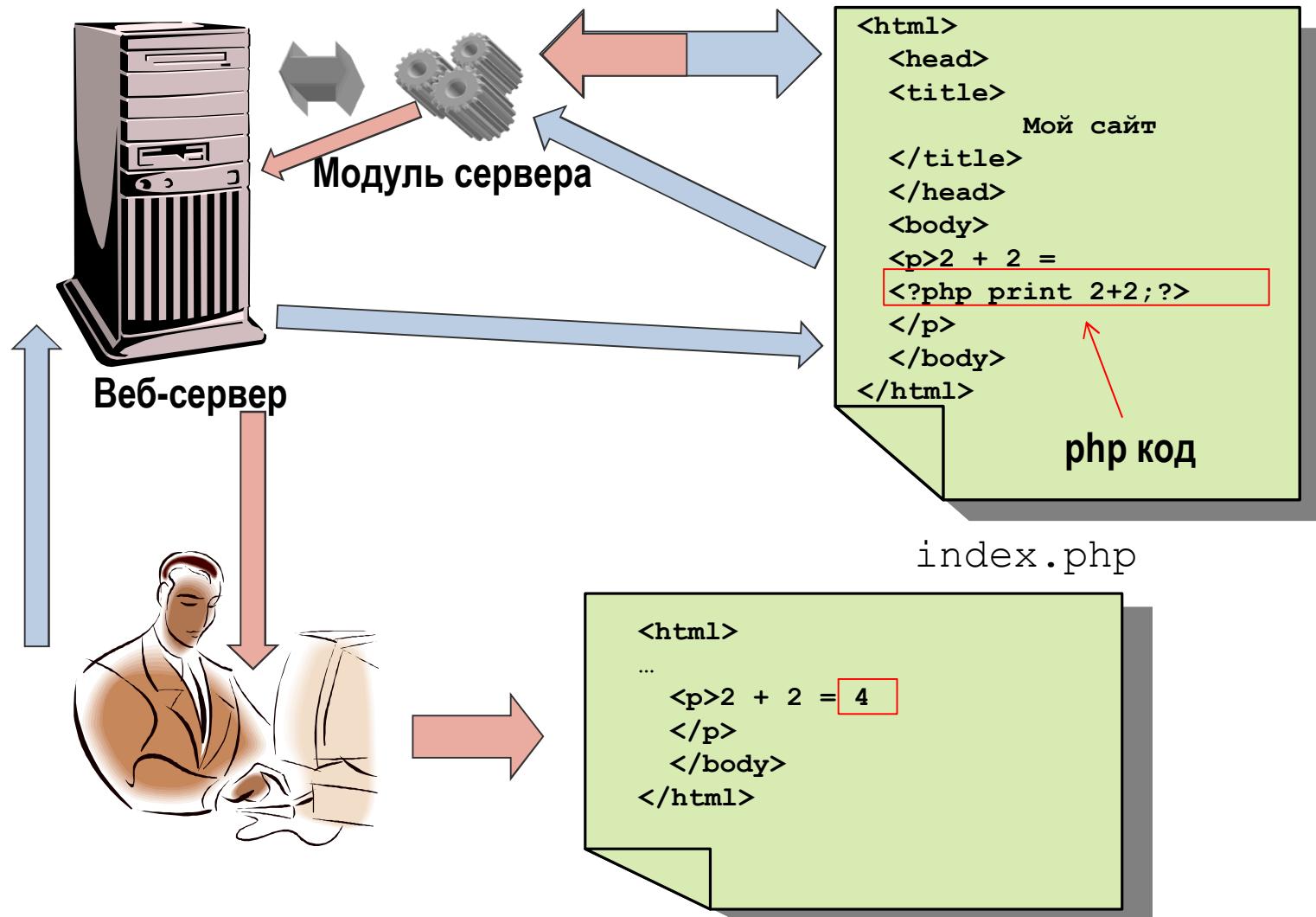
Вопросы, на которые надо ответить

- Только ли PHP?
- Почему PHP?
- Что такое "интерпретатор"?
- Что значит "встраиваемый язык"?
- Возможности PHP
- Зачем нужен сервер?
- Что такое "сервер"?
- Только ли Apache?
- Почему Apache?

Как это работает? (HTML)



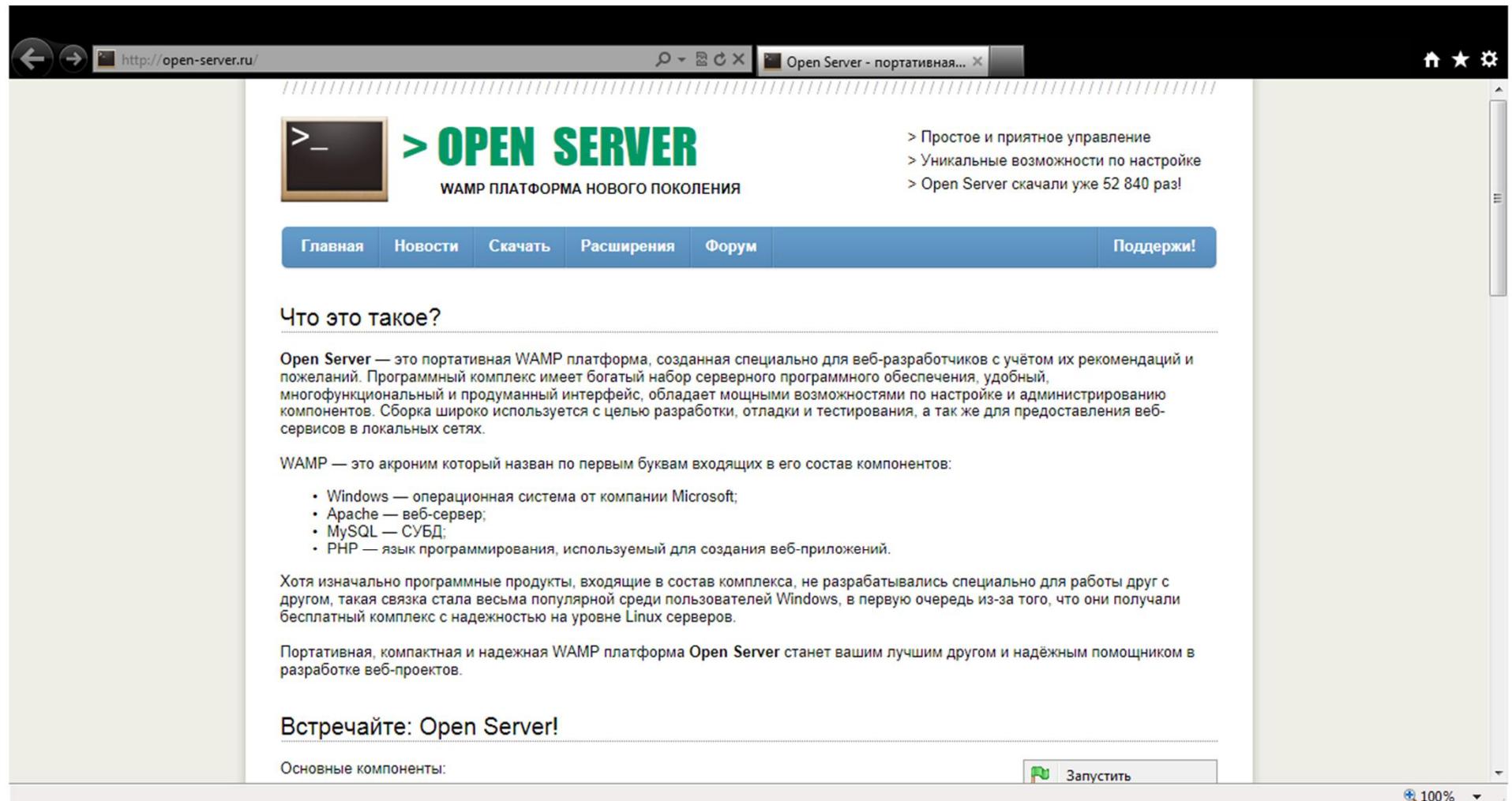
Как это работает? (PHP)



Что ставить?

- Сервер Apache
 - <http://httpd.apache.org/>
- Сервер Microsoft IIS
 - <http://www.iis.net/>
- Сборка Денвер
 - <http://www.denwer.ru/>
- Сборка XAMPP
 - <http://www.apachefriends.org/ru/xampp.html>
- Сборка Wamp Server
 - <http://www.wampserver.com>
- Сборка EasyPHP
 - <http://www.easypHP.org/>

Сборка Open Server



The screenshot shows a web browser window with the URL <http://open-server.ru/> in the address bar. The page content is as follows:

>_ > OPEN SERVER
WAMP ПЛАТФОРМА НОВОГО ПОКОЛЕНИЯ

> Простое и приятное управление
> Уникальные возможности по настройке
> Open Server скачали уже 52 840 раз!

[Главная](#) [Новости](#) [Скачать](#) [Расширения](#) [Форум](#) [Поддержки!](#)

Что это такое?

Open Server — это портативная WAMP платформа, созданная специально для веб-разработчиков с учётом их рекомендаций и пожеланий. Программный комплекс имеет богатый набор серверного программного обеспечения, удобный, многофункциональный и продуманный интерфейс, обладает мощными возможностями по настройке и администрированию компонентов. Сборка широко используется с целью разработки, отладки и тестирования, а так же для предоставления веб-сервисов в локальных сетях.

WAMP — это акроним который назван по первым буквам входящих в его состав компонентов:

- Windows — операционная система от компании Microsoft;
- Apache — веб-сервер;
- MySQL — СУБД;
- PHP — язык программирования, используемый для создания веб-приложений.

Хотя изначально программные продукты, входящие в состав комплекса, не разрабатывались специально для работы друг с другом, такая связка стала весьма популярной среди пользователей Windows, в первую очередь из-за того, что они получали бесплатный комплекс с надежностью на уровне Linux серверов.

Портативная, компактная и надежная WAMP платформа Open Server станет вашим лучшим другом и надежным помощником в разработке веб-проектов.

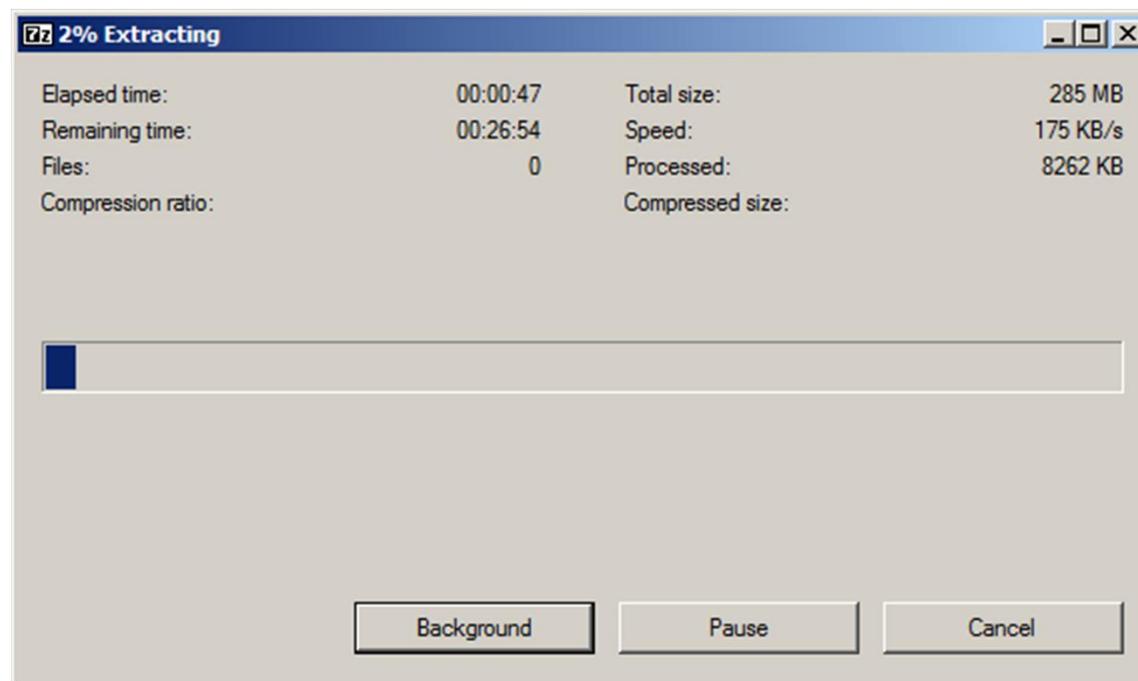
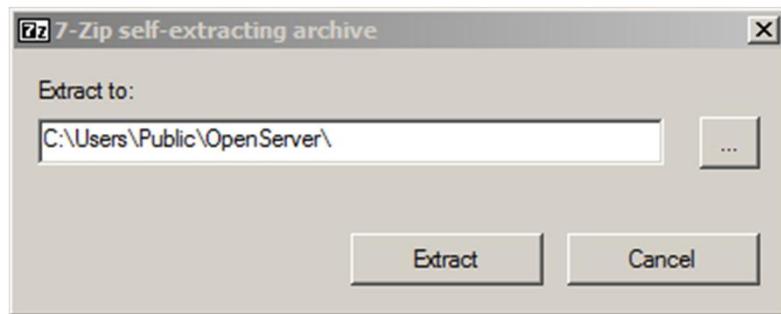
Встречайте: Open Server!

Основные компоненты:  Запустить

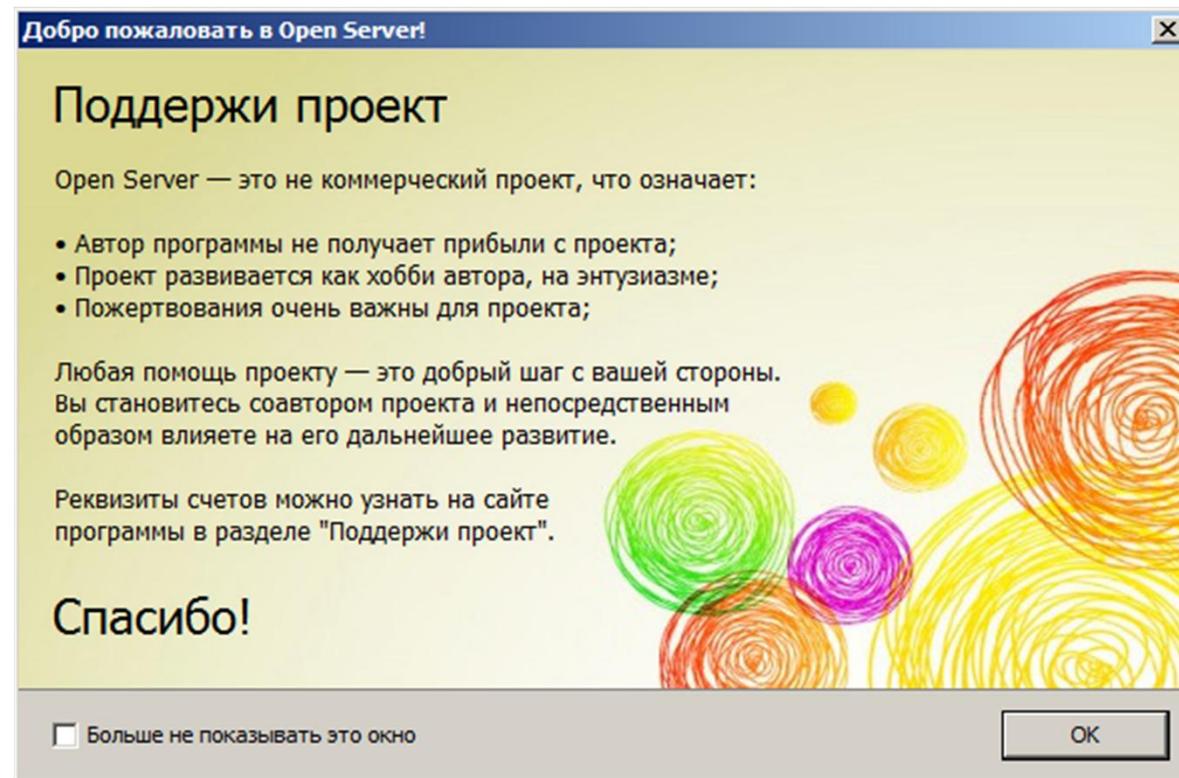
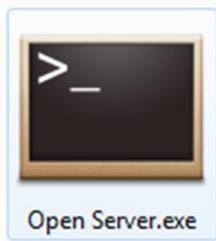
100%

<http://open-server.ru>

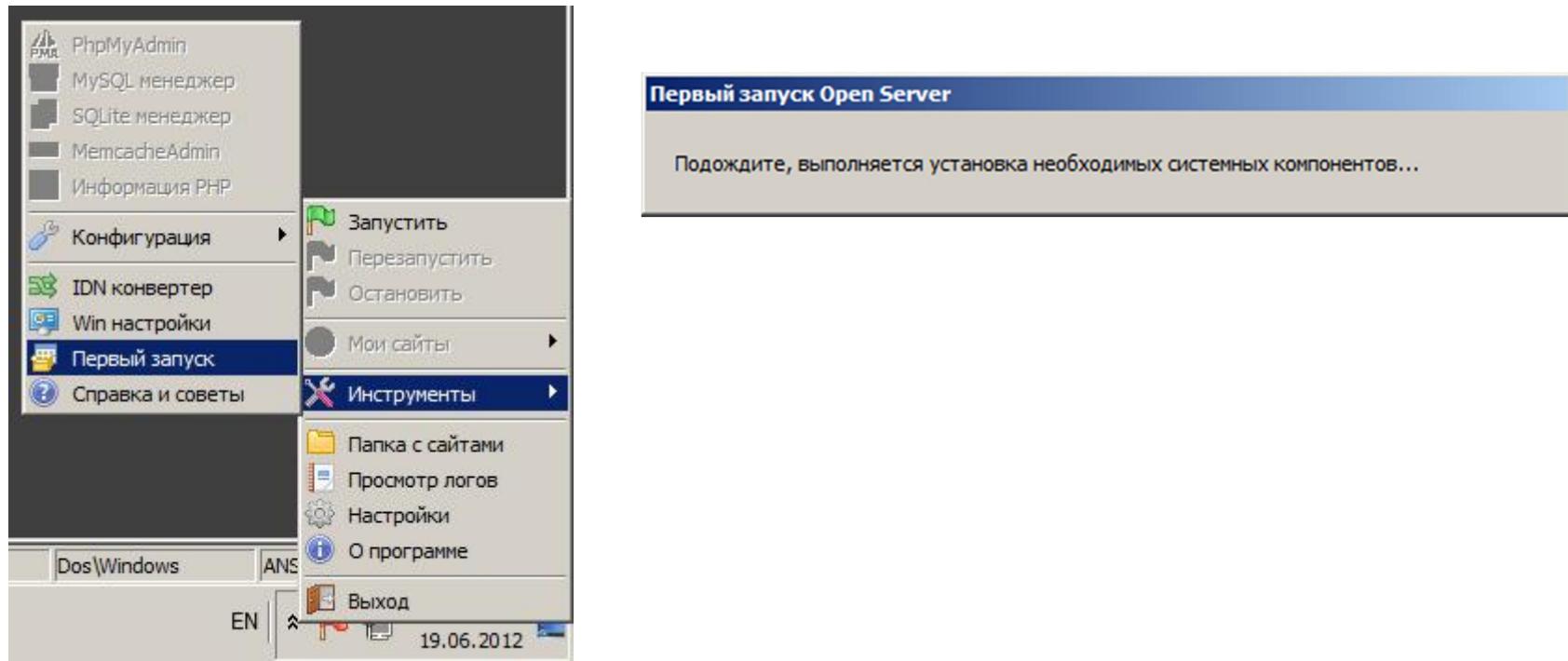
Установка Open Server: шаг 1



Установка Open Server: шаг 2



Установка Open Server: шаг 3



Проверка работы Open Server

Добро пожаловать в Open Server!

Вау! Он работает ;-)

[Перейти к стартовой странице](#)

Open Server: список сайтов и инструментов



[localhost](#)

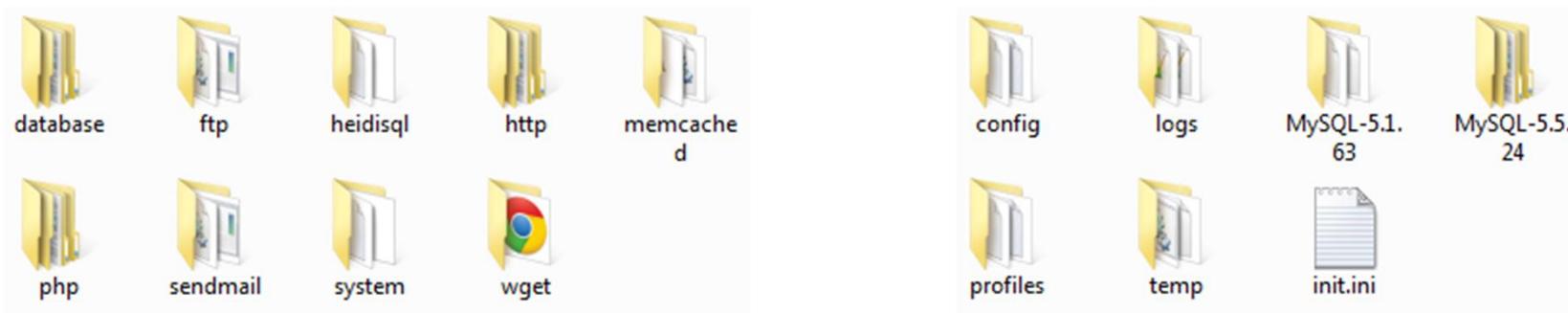
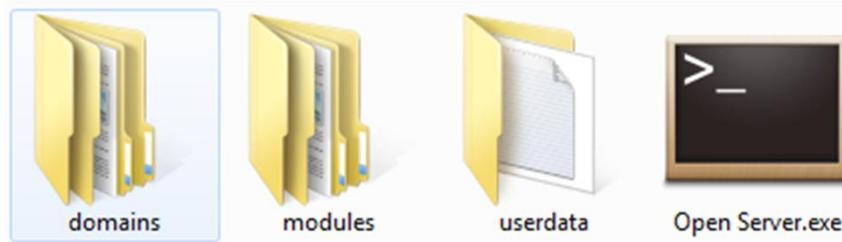


[localhost](#)

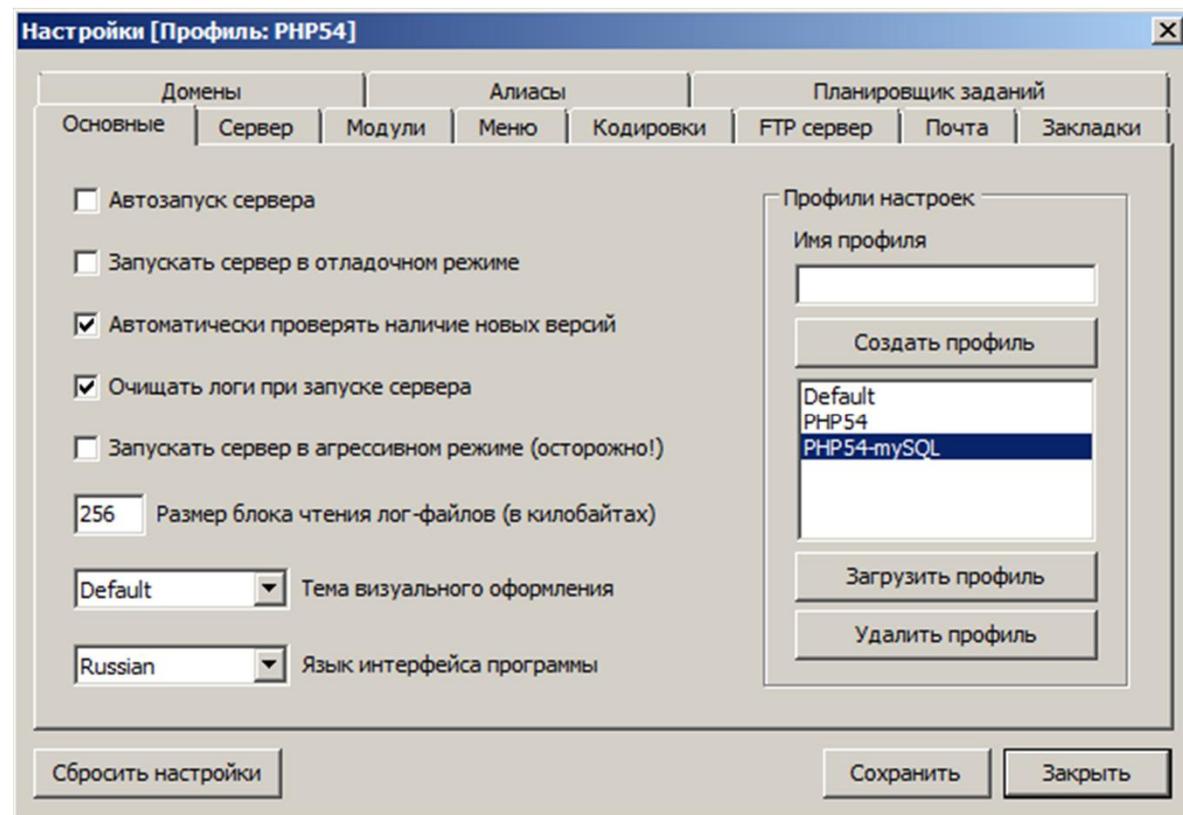


[PHPMyAdmin](#)
[PHP Info](#)
[Adminer](#)
[MemCachedAdmin](#)
[Webgrind](#)
[Server Status](#)

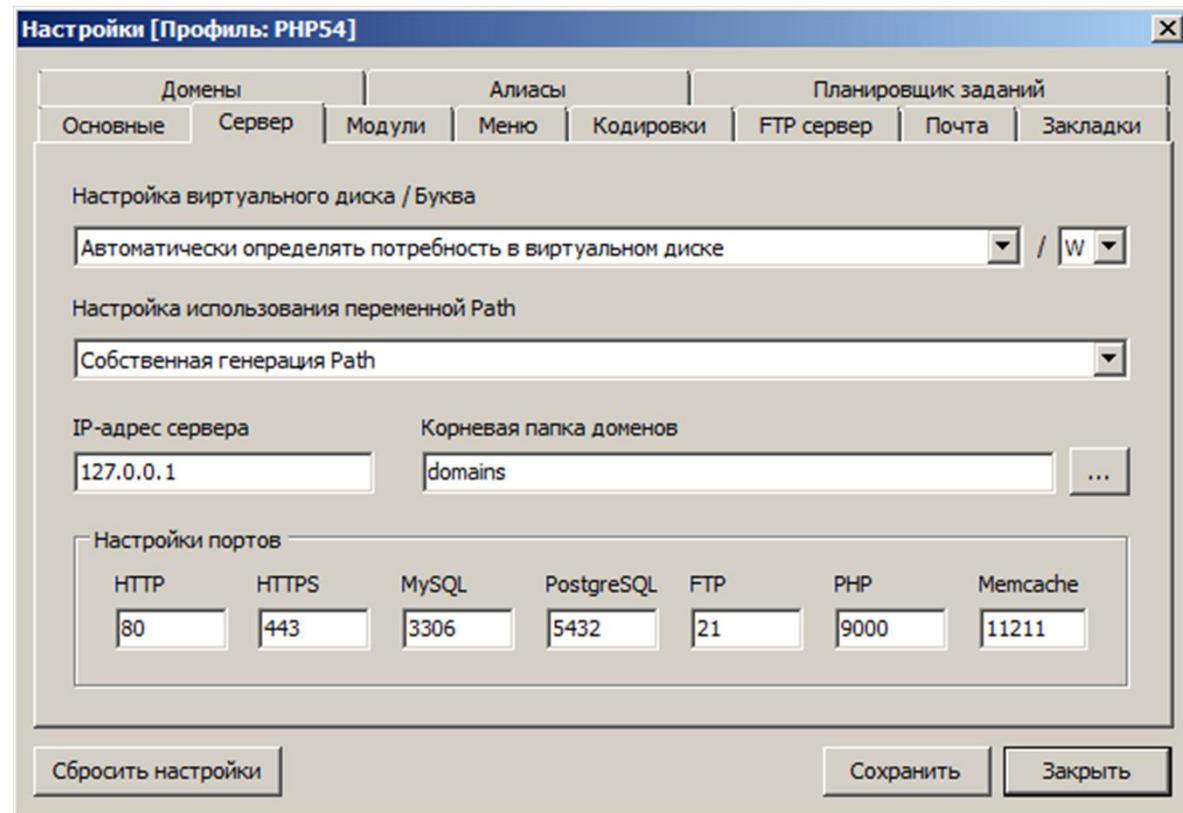
Open Server: что внутри?



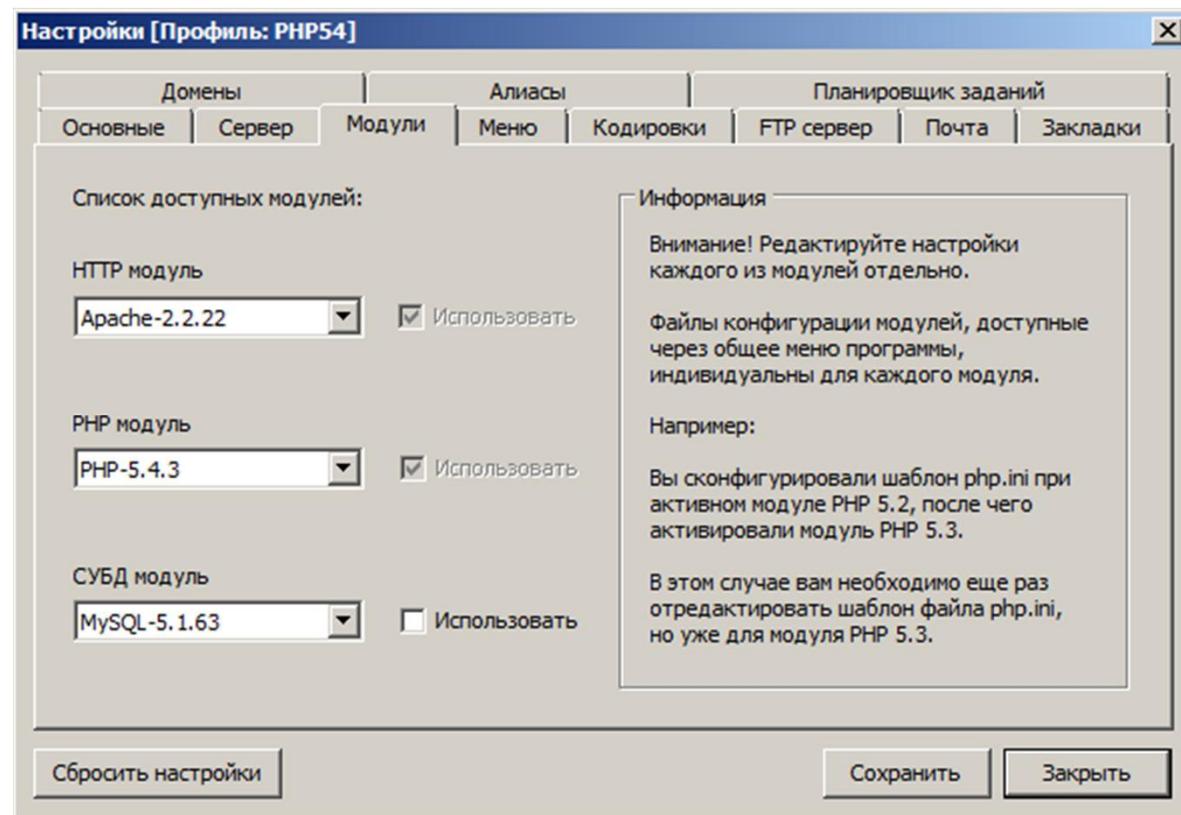
Настройка Open Server: Основные



Настройка Open Server: Сервер



Настройка Open Server: Модули



Лабораторная работа 1.1

Подготовка рабочего места

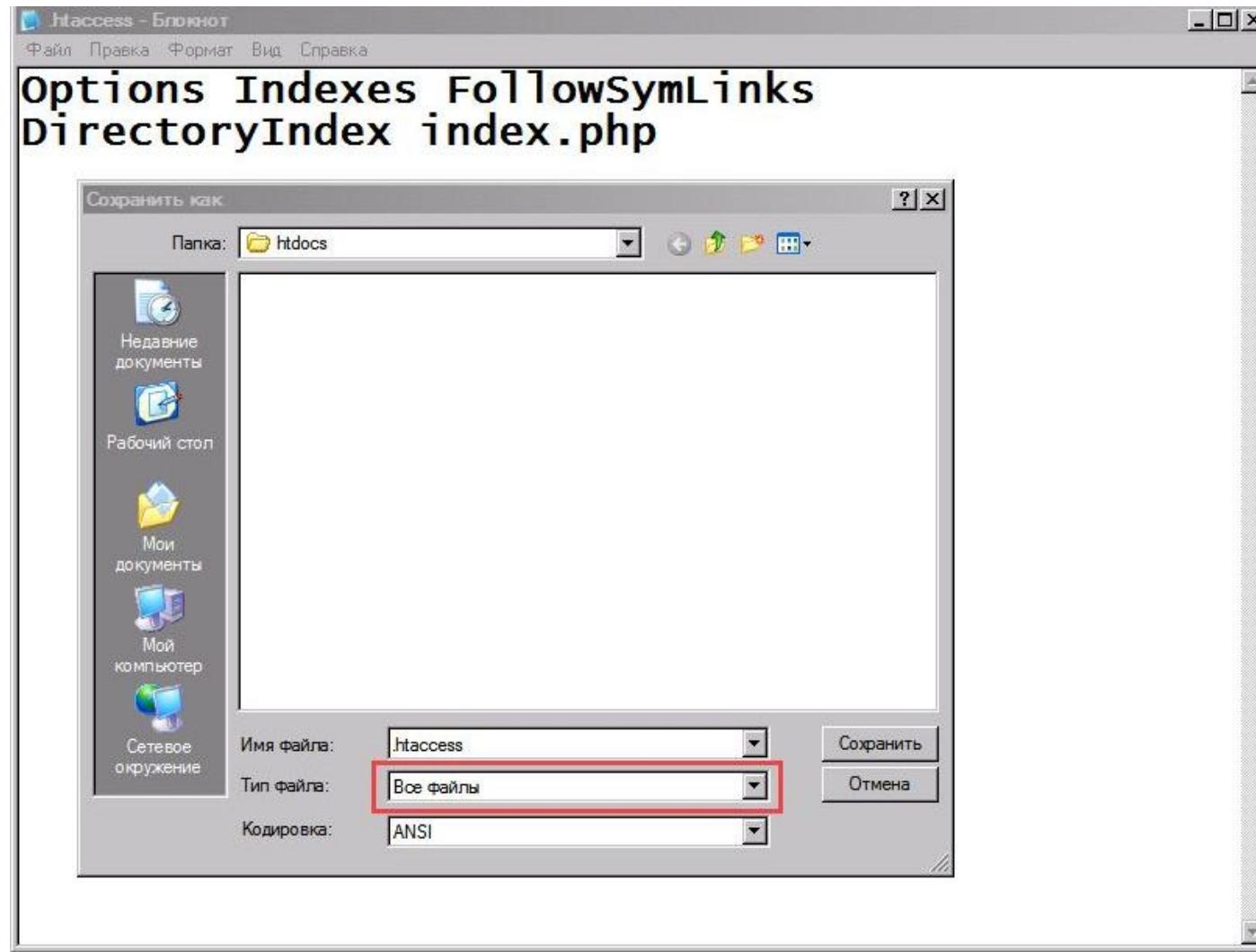
Упражнение 1: Создание виртуального хоста и запуск сервера

- Откройте проводник Windows
- Перейдите в директорию **C:\Пользователи\Общие\OpenServer\domains**
(Внимание! В некоторых ситуациях русскоязычному пути C:\Пользователи\Общие\ соответствует англоязычный путь C:\Users\Public\. Это одно и тоже.)
- В этой директории создайте папку **mysite.local**
- Запустите сервер. Для этого нажмите
[**Пуск -> Open Server**]
(На всякий случай, сама программа находится по пути C:\Пользователи\Общие\OpenServer\OpenServer.exe)
- В правом нижнем углу (рядом с часами) кликните по иконке с красным флагом
- В открывшемся меню выберите первый пункт **Запустить**
- Дождитесь пока цвет иконки с флагом изменится с желтого на зеленый
- Если запуск закончился неудачей - флагок опять стал красным, то кликните по иконке, выберите последний пункт **Выход** и повторите последние 4 пункта

Упражнение 2: Копирование необходимых файлов

- Получите у преподавателя архив с файлами для работы на курсе
- Скопируйте файл в созданную в предыдущем упражнении директорию **C:\Пользователи\Общие\OpenServer\domains\mysite.local**
- Распакуйте файл в **текущую** директорию
- Запустите браузер и в адресной строке наберите: **http://mysite.local/**
- Убедитесь, что сайт работает

Файл .htaccess



Лабораторная работа 1.2

Создание файла .htaccess

Упражнение 1: Создание файла конфигурации директории

- Откройте текстовый редактор
- В текстовом редакторе создайте новый файл и напишите следующий текст:
`# Настройки сервера Apache`

```
Options Indexes FollowSymLinks  
DirectoryIndex index.php
```

- Сохраните файл в папке **C:\Пользователи\Общие\OpenServer\domains\mysite.local** под именем **.htaccess**
- Запустите браузер и в адресной строке наберите: **http://mysite.local/**
- Убедитесь, что нет ошибок и сайт работает

Где живет PHP?

The screenshot shows the official PHP website at <http://php.net/>. The page features a large blue header bar with the PHP logo and navigation links like 'downloads', 'documentation', and 'faq'. Below the header, there's a main content area with sections for 'What is PHP?', 'Upcoming conferences', 'php.net security notice', 'PHP 5.3.6 Released!', and 'Security Enhancements and Fixes in PHP 5.3.6:'. On the right side, there are sidebar boxes for 'Stable Releases' (listing 'Current PHP 5.3 Stable: 5.3.6'), 'Upcoming Events [add]' (listing 'May' and 'Conferences' with links to 'phpDay 2011', 'International PHP Conference', and 'Internat'l PHP Conference 2011'), and 'User Group Events' (listing 'Wash DC PHP Developers Group', 'PHP User Group Stuttgart', and 'South FL PUG- Miami').

What is PHP?

PHP is a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into HTML. If you are new to PHP and want to get some idea of how it works, try the [introductory tutorial](#). After that, check out the online [manual](#), and the example archive sites and some of the other resources available in the [links section](#).

Ever wondered how popular PHP is? see the [Netcraft Survey](#).

Thanks To

... DNS

Upcoming conferences: [Dutch PHP Conference 2011](#) [Italian phpDay 2011](#)

php.net security notice

[19-Mar-2011] The wiki.php.net box was compromised and the attackers were able to collect wiki account credentials. No other machines in the php.net infrastructure appear to have been affected. Our biggest concern is, of course, the integrity of our source code. We did an extensive code audit and looked at every commit since 5.3.5 to make sure that no stolen accounts were used to inject anything malicious. Nothing was found. The compromised machine has been wiped and we are forcing a password change for all svn accounts.

We are still investigating the details of the attack which combined a vulnerability in the Wiki software with a Linux root exploit.

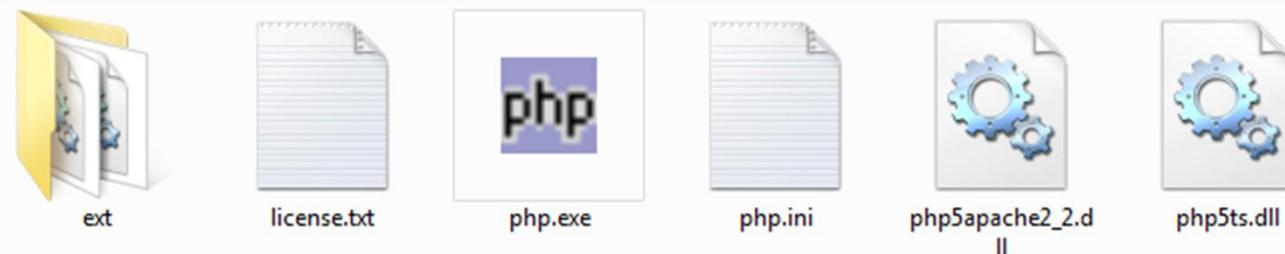
PHP 5.3.6 Released!

[17-Mar-2011] The PHP development team would like to announce the immediate availability of PHP 5.3.6. This release focuses on improving the stability of the PHP 5.3.x branch with over 60 bug fixes, some of which are security related.

Security Enhancements and Fixes in PHP 5.3.6:

<http://php.net>

PHP: ЧТО ВНУТРИ?

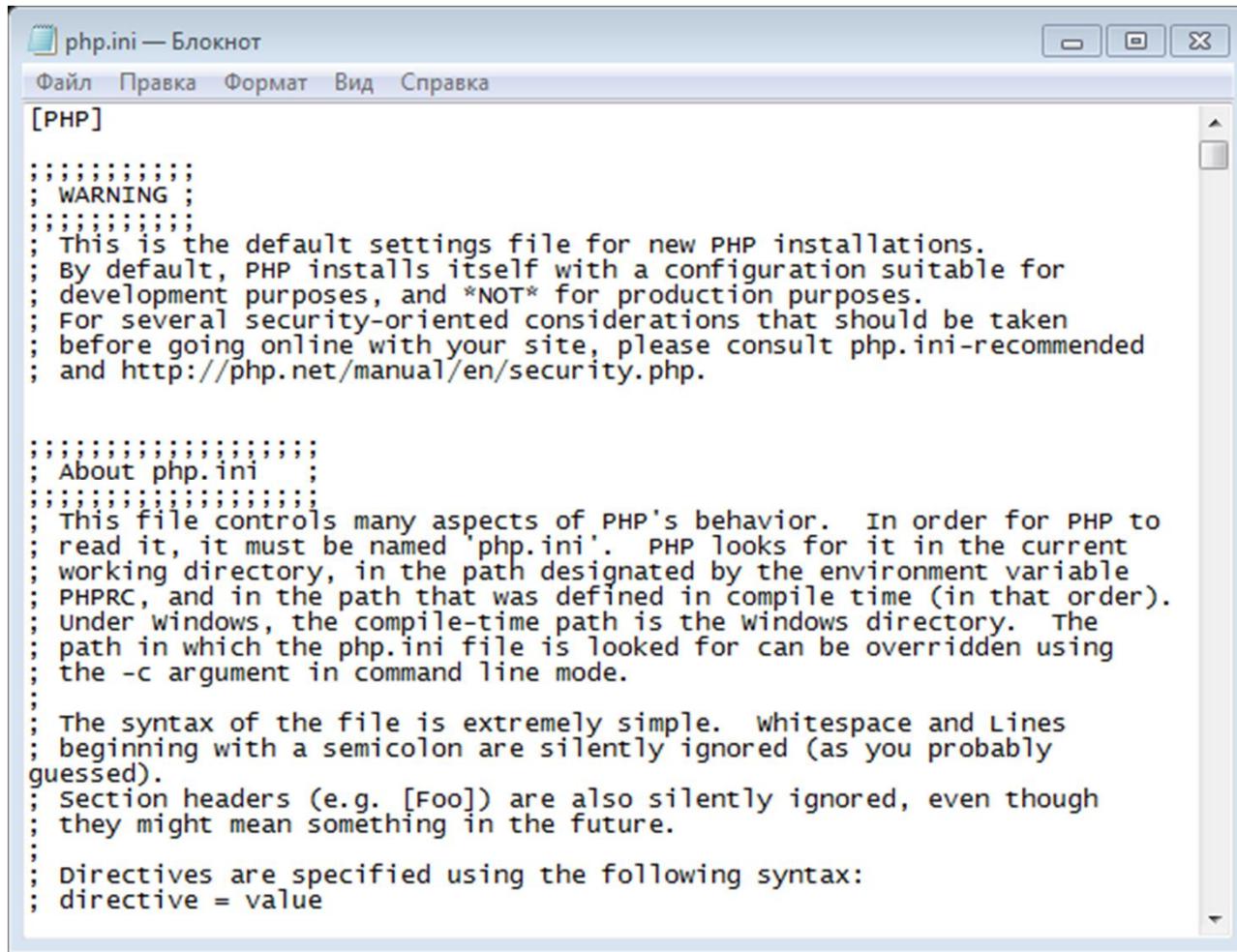


Как PHP привязан к Apache?

- **conf\httpd.conf**

- LoadModule php5_module
"путь-к-php/php5apache2_2.dll"
 - AddType application/x-httpd-php .php

Файл php.ini



php.ini — Блокнот

Файл Правка Формат Вид Справка

[PHP]

```
; ;;;;;;;;;;;;;;;;;;;;
; WARNING ;
; ;;;;;;;;;;;;;;;;;;;
; This is the default settings file for new PHP installations.
; By default, PHP installs itself with a configuration suitable for
; development purposes, and *NOT* for production purposes.
; For several security-oriented considerations that should be taken
; before going online with your site, please consult php.ini-recommended
; and http://php.net/manual/en/security.php.

; ;;;;;;;;;;;;;;;;;;;;
; About php.ini
; ;;;;;;;;;;;;;;;;;;;
; This file controls many aspects of PHP's behavior. In order for PHP to
; read it, it must be named 'php.ini'. PHP looks for it in the current
; working directory, in the path designated by the environment variable
; PHPRC, and in the path that was defined in compile time (in that order).
; Under windows, the compile-time path is the windows directory. The
; path in which the php.ini file is looked for can be overridden using
; the -c argument in command line mode.

; The syntax of the file is extremely simple. whitespace and Lines
; beginning with a semicolon are silently ignored (as you probably
; guessed).
; Section headers (e.g. [Foo]) are also silently ignored, even though
; they might mean something in the future.

; Directives are specified using the following syntax:
; directive = value
```

Лабораторная работа 1.3

Первый скрипт

Упражнение 1: Вывод системной информации

- Откройте текстовый редактор
- В текстовом редакторе создайте новый файл и напишите следующий текст:

```
<?php
    phpinfo();
?>
```
- Сохраните файл в папке **C:\Пользователи\Общие\OpenServer\domains\mysite.local** под именем **info.php**
- Запустите браузер и в адресной строке наберите: **http://mysite.local/info.php**
- Вы должны увидеть специальную страницу с логотипом PHP

Выводы

- Установка и настройка веб-сервера
- Создание виртуального хоста
- Создание файла .htaccess
- Проверка работы
- Просмотр информации

PHP Уровень 1.

Основы PHP

Игорь Борисов
<http://igor-borisov.ru>

Темы модуля

- Синтаксис
- Операторы
- Переменные
- Константы
- Типы
- Управляющие конструкции

PHP тэги

- Так PHP обрабатывает файл

```
<p>Это будет проигнорировано.</p>
<?php echo 'А это будет обработано.'; ?>
<p>Это тоже будет проигнорировано.</p>
```

PHP тэги

- Два всегда доступных набора тэгов
 - Рекомендуемый <?php ?>
 - И <script language="php"> </script>

```
<?php
echo 'если вы хотите работать с документами
ХНТМЛ или XML, делайте так';
?>
```

```
<script language="php">
echo 'некоторые редакторы (например, FrontPage)
не любят инструкции обработки';
</script>
```

PHP тэги

- Два других набора тэгов
 - Короткие тэги (доступны с директивой **short_open_tag** в php.ini)
 - Всегда доступен с PHP 5.4
 - Тэги в стиле ASP (доступны с директивой **asp_tags** в php.ini)

```
<?
echo 'это простейшая инструкция обработки';
?>
<%
echo 'Вы можете по выбору использовать тэги в
стиле ASP';
%>
```

Комментарии

- PHP поддерживает комментарии в стиле 'C', 'C++' и оболочки Unix (стиль Perl)

```
echo "Это тест"; // Это односторонний комментарий в стиле с++
/* Это многострочный комментарий
   еще одна строка комментария */
echo "Это еще один тест";
echo "Последний тест"; # Это тоже односторонний комментарий

/*
   echo "Это тест"; /* Этот комментарий вызовет проблему */
*/
```

Особенности PHP

- <?php какая-то инструкция;?>
- <?php инструкция 1; инструкция 2;?>
- <?php
 какая-то
 инструкция
 в
 несколько
 строк;
?>

Вывод данных

- echo "Привет мир!";
- print("Привет мир!");
- print "print можно использовать и без скобок.";
- echo 'Эта ', 'строка ', 'была ', 'создана ', 'с несколькими параметрами.';
- echo strftime('%d-%B-%Y, %A');

Лабораторная работа 2.1

Вывод данных

Упражнение 1: Вывод текущей даты

- Откройте в текстовом редакторе файл **index.php**
- В области основного контента **перед** строкой **<h3>Зачем мы ходим в школу?</h3>** напишите следующий текст:

```
<blockquote>
<?php
    echo strftime('Сегодня %d-%m-%Y');
?>
</blockquote>
```
- Сохраните файл **index.php**
- Запустите браузер и в адресной строке наберите: **http://mysite.local/**
- Убедитесь, что нет ошибок и результат выводится на страницу

Переменные

- Переменные в PHP начинаются со знака доллара (\$)
- Имя переменной **должно начинаться с** буквы или символа подчеркивания
- Последующие символы в имени переменной могут быть буквами, цифрами или символом подчеркивания в любом количестве
- Имя переменной **чувствительно к регистру**

Оператор присваивания

- Присвоить переменной `$x` значение **10**
 - `$x = 10;`
- Теперь прибавим к значению переменной `$x` значение **15**
 - `$x = $x + 15; // $x теперь равно 25`
- Выведем значение переменной `$x`
 - `print $x;`
- Удалим переменную `$x`
 - `unset($x);`

ВЫВОД ДАННЫХ

- `$name = "Вася";`
- `<h1>Привет <?php echo $name?></h1>`
- `<h1>Привет <?= $name?></h1>`
 - Этот сокращенный синтаксис допустим только когда включена директива конфигурации *short_open_tag*
 - Работает всегда с PHP 5.4

Лабораторная работа 2.2

Использование переменных

Упражнение 1: Вывод текущей даты используя переменные

- Откройте в текстовом редакторе файл **index.php**

- В самом начале файла введите следующий текст:

```
<?php  
    // Установка локали и выбор значений даты  
    setlocale(LC_ALL, "russian");  
    $day = strftime('%d');  
    $mon = strftime('%B');  
    $year = strftime('%Y');  
?>
```

- Внутри тэгов **<blockquote></blockquote>** вместо текста:

```
echo strftime('Сегодня %d-%m-%Y');
```

введите следующий текст:

```
echo 'Сегодня ', $day, ' число, ', $mon, ' месяц, ', $year, ' год.';
```

- Сохраните файл **index.php** и посмотрите результат в браузере

- Внизу файла в блоке **<!-- Нижняя часть страницы -->** вместо **2012** напишите:

```
<?= strftime('%Y')?>
```

- Сохраните файл **index.php**

- Посмотрите результат в браузере

Ошибки – наше всё

- Директивы php.ini
 - display_errors = On
 - error_reporting = E_ALL & ~E_NOTICE
- `ini_set('display_errors', 1);`
- Основные уровни ошибок
 - E_PARSE
 - E_ERROR
 - E_WARNING
 - E_NOTICE

Выставление уровней ошибок

- Включаем вывод всех ошибок
 - `error_reporting(E_ALL);`
- Отключаем вывод ошибок
 - `error_reporting(0);`
- Включаем определенные уровни ошибок
 - `error_reporting(E_ERROR | E_WARNING);`
 - `error_reporting(E_ALL & ~E_DEPRECATED);`

Копирование переменных

- `$x = 10;`
- Создаем копию переменной `$x`
 - `$y = $x;`

- `$y = 20;`
 - `echo $x; // 10`
 - `echo $y; // 20`

Ссылки

- `$x = 10;`
- Создаем ссылку на переменную `$x`
 - `$y =& $x;`
- `$y = 20;`
 - `echo $x; // 20`
 - `echo $y; // 20`

Переменные переменных

- \$a = 'hello';
 - \$\$a = 'world';
// hello world
 - echo \$a, ' ', \$hello;
-
- \$a = 'name';
 - \$\$a = 'Вася';
// Вас зовут: Вася
 - echo 'Вас зовут: ', \$name;

Типы

- PHP – язык с динамической типизацией (loosely typed)
- PHP поддерживает восемь простых типов
- Четыре скалярных типа:
 - boolean
 - integer
 - float (число с плавающей точкой)
 - string
- Два смешанных типа:
 - array
 - object
- Два специальных типа:
 - resource
 - NULL

Типы: boolean

- Это простейший тип
- Выражает истинность значения: `TRUE` или `FALSE`
- Ключевые слова `TRUE` и `FALSE` регистра-
независимы
- Присвоить `$foo` значение `TRUE`
 - `$foo = True;`

Типы: integer и float

■ Integer

- `$i = 1234; // десятичное число`
- `$i = -123; // отрицательное число`
- `$i = 0123; // 83 в восьмеричной системе`
- `$i = 0x1A; // 26 в шестнадцатеричной системе`
- `$i = 0b101; // 5 в двоичной системе (PHP 5.4)`

■ Float

- `$f = 1.234;`
- `$f = 1.2e3; // 1200`
- `$f = 7E-10; // 0.0000000007`

Типы: string (двойные кавычки)

- Если строка указывается в двойных кавычках, переменные внутри нее обрабатываются
 - `$juice = "apple";`
- Выводит: He drank some apple juice
 - `echo "He drank some $juice juice.;"`
- Выводит: Здесь будет перевод на новую строку
 - `echo "Здесь будет перевод на \n новую строку.;"`
- Выводит: Экранируем "внутри" двойных кавычек
 - `echo "Экранируем \"внутри\" двойных кавычек.;"`

Управляющие последовательности

- Если строка заключена в двойные кавычки, PHP распознает управляющие последовательности для специальных символов:
 - `\n` новая строка (LF или 0x0A (10) в ASCII)
 - `\r` возврат каретки (CR или 0x0D (13) в ASCII)
 - `\t` горизонтальная табуляция (HT или 0x09 (9) в ASCII)
 - `\\"` обратная косая черта
 - `\$` знак доллара
 - `\"` двойная кавычка

Типы: string (одинарные кавычки)

- В отличие от синтаксиса двойных кавычек, переменные и управляющие последовательности для специальных символов, заключенных в одинарные кавычки, не обрабатываются
 - `$juice = "apple";`
- Выводит: He drank some \$juice juice
 - `echo 'He drank some $juice juice.';`
- Выводит: Здесь будет перевод на \n новую строку
 - `echo 'Здесь будет перевод на \n новую строку.';`
- Выводит: Экранируем 'внутри' одинарных кавычек
 - `echo 'Экранируем \'внутри\' одинарных кавычек.';`

Лабораторная работа 2.3

Использование двойных кавычек

Упражнение 1: Вывод текущей даты используя подстановку значений переменных в двойных кавычках

- Откройте в текстовом редакторе файл `index.php`
- Переделайте строку:
`echo 'Сегодня ', $day, ' число, ', $mon, ' месяц, ', $year, ' год.';`
в строку:
`echo "Сегодня $day число, $mon месяц, $year год.";`
- Сохраните файл `index.php`
- Посмотрите результат в браузере

Типы: string (heredoc)

- echo <<<LABEL

Меня зовут "\$name". Переводим на новую строку.

Перед строкой стоит символ табуляции.

Это должно вывести заглавную букву 'A': \x41
LABEL;

- Идентификатор должен содержать только буквенно-цифровые символы и знак подчеркивания, и не должен начинаться с цифры (знак подчеркивания разрешается)
- После оператора <<< необходимо указать идентификатор, затем **только** перевод строки
- Закрывающий идентификатор **должен** стоять в первом столбце строки
- Стока с закрывающим идентификатором **не должна** содержать других символов, за исключением, возможно, точки с запятой (;

PHP 5.3: HEREDOC и NOWDOC

- `echo <<<"LABEL"`
Меня зовут "\$name". Переводим на новую строку.
Перед строкой стоит символ табуляции.
Это должно вывести заглавную букву 'A': \x41
`LABEL;`
- `echo <<<'LABEL'`
Меня зовут "\$name". Переводим на новую строку.
Перед строкой стоит символ табуляции.
Это должно вывести заглавную букву 'A': \x41
`LABEL;`

Типы: NULL

- Специальное значение NULL представляет собой переменную без значения
- NULL - это единственное возможное значение типа NULL
- Переменная считается null, если:
 - ей была присвоена константа `NULL`
 - ей еще не было присвоено никакого значения
 - она была удалена с помощью `unset()`
- `$var = NULL;`

Экранирование переменных

- `$juice = "apple";`
- Каков результат следующего кода?
 - `echo "He drank some $juice juice.";`
 - `echo "He drank some juice made of $juices.;"`
- Решение проблемы
 - `echo "He drank some juice made of {$juice}s.;"`
 - `echo "He drank some juice made of ${juice}s.;"`

Доступ к символу в строке

- Символы в строках можно использовать и модифицировать, определив их смещение относительно начала строки, начиная с нуля
- `$str = 'This is a test.';`
- Получение первого символа строки
 - `$first = $str{0};`
- Получение третьего символа строки
 - `$third = $str{2};`
- Получение последнего символа строки
 - `$last = $str{strlen($str)-1};`
- Изменение последнего символа строки
 - `$str[strlen($str)-1] = '!';`

Арифметические операторы

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

Строковый оператор

- Оператор конкатенации ('.') возвращает объединение левого и правого аргумента
 - `$a = "Hello ";`
 - `$b = $a . "World!" ;`
 - // \$b теперь содержит строку "Hello World!"
- `$a = "Hello"; $b = "World!" ;`
- `$c = $a . " " . $b;`
- // \$c теперь содержит строку "Hello World!"
- `$c = "$a $b"; // $c тоже "Hello World!"`

Комбинированные операторы

- В дополнение к базовому оператору присваивания имеются "комбинированные операторы" для всех бинарных арифметических и строковых операций, которые позволяют использовать некоторое значение в выражении, а затем установить его как результат данного выражения
- `$a = 3;`
`$a += 5; // устанавливает $a в 8, как если бы мы написали: $a = $a + 5;`
- `$b = "Hello ";`
`$b .= "World!"; // устанавливает $b в "Hello World!", как и $b = $b . "World!";`

Коротко о двоичной системе

2									10
128	64	32	16		8	4	2	1	
0	0	1	0		1	0	1	0	42
0	0	1	0		0	1	1	1	39
0	1	1	0		0	1	0	1	101
1	0	0	0		0	0	0	0	128
1	0	1	0		0	1	1	0	166
1	1	1	1		1	1	1	1	255

Побитовые операторы

Пример	Название	Результат
<code>\$a & \$b</code>	И	Устанавливаются только те биты, которые установлены и в <code>\$a</code> , и в <code>\$b</code> .
<code>\$a \$b</code>	Или	Устанавливаются те биты, которые установлены в <code>\$a</code> или в <code>\$b</code> .
<code>\$a ^ \$b</code>	Исключающее или	Устанавливаются только те биты, которые установлены либо только в <code>\$a</code> , либо только в <code>\$b</code> , но не в обоих одновременно.
<code>~ \$a</code>	Отрицание	Устанавливаются те биты, которые не установлены в <code>\$a</code> , и наоборот.
<code>\$a << \$b</code>	Сдвиг влево	Все биты переменной <code>\$a</code> сдвигаются на <code>\$b</code> позиций влево (каждая позиция подразумевает "умножение на 2")
<code>\$a >> \$b</code>	Сдвиг вправо	Все биты переменной <code>\$a</code> сдвигаются на <code>\$b</code> позиций вправо (каждая позиция подразумевает "деление на 2")

Побитовые операторы: примеры

- `$result = 1 & 5; // $result = 1`
- `$result = 1 | 5; // $result = 5`
- `$result = 1 ^ 5; // $result = 4`

- Побитовый сдвиг в PHP - это арифметическая операция
- Биты, сдвинутые за границы числа, отбрасываются
- Сдвиг влево дополняет число нулями справа, сдвигая в то же время знаковый бит числа влево, что означает что знак операнда не сохраняется
- Сдвиг вправо сохраняет копию сдвинутого знакового бита слева, что означает что знак операнда сохраняется
 - `$result = 4 >> 1; // $result = 2`
 - `$result = 4 << 1; // $result = 8`

Другие операторы

- Оператор управления ошибками
 - Знак (@). В случае, если он предшествует какому-либо выражению в PHP-коде, любые сообщения об ошибках, генерируемые этим выражением, будут проигнорированы
 - `echo @unknown_function();`
 - Не надо использовать этот оператор
- Оператор исполнения
 - Обратные кавычки (''). PHP попытается выполнить строку, заключенную в обратные кавычки, как консольную команду, и вернет полученный вывод
 - `$output = `ping 127.0.0.1`;`
`echo "<pre>$output</pre>";`
 - Подумайте, так ли необходим этот оператор?

Другие типы

- **Array (Массив)**
 - Массив в PHP - это упорядоченное отображение, которое устанавливает соответствие между значением и ключом
 - Рассматривается далее в теме, посвященной массивам
- **Resource (Ресурс)**
 - Это специальная переменная, содержащая ссылку на внешний ресурс
 - Ресурсы создаются и используются специальными функциями
 - Рассматривается в темах, посвященных работе с файлами, директориями и базами данных
- **Object (Объект)**
 - Сущность, обладающая определённым состоянием и поведением
 - Рассматривается в теме, посвященной объектно-ориентированному программированию

Полезные функции: `getTуре`

- `echo gettype($value);`
 - Возвращает тип переменной
 - Возможными значениями возвращаемой строки являются:
 - "boolean""integer"
 - "double" (по историческим причинам "double" возвращается в случае типа float, а не просто "float")
 - "string"
 - "array"
 - "object"
 - "resource"
 - "NULL"

Полезные функции: setType

- ```
$foo = "5bar"; // строка
settype($foo, "integer"); // $foo
теперь 5 (целое)
```
- Допустимыми значениями типа являются:
  - "boolean" (или "bool")
  - "integer" (или "int")
  - "float" (или "double")
  - "string"
  - "array"
  - "object"
  - "null"

# Приведение типов

- `$foo = 10; // $foo это целое число`  
`$bar = (boolean) $foo; // $bar это`  
`булев тип`
- Допускаются следующие приведения типов:
  - (int), (integer) - приведение к integer
  - (bool), (boolean) - приведение к boolean
  - (float), (double), (real) - приведение к float
  - (string) - приведение к string
  - (array) - приведение к array
  - (object) - приведение к object
  - (unset) - приведение к NULL (PHP 5)

# Константы

- Константы можно определить с помощью функции define(), а не присваиванием значения
  - `define("CONSTANT", "Здравствуй, мир.");`
  - `const CONSTANT = "Здравствуй, мир."; // PHP 5.3`
- У констант нет приставки в виде знака доллара (\$)
  - `echo CONSTANT; // выводит "Здравствуй, мир."`
- Константы могут быть определены и доступны в любом месте без учета области видимости
- Константы не могут быть определены или аннулированы после первоначального объявления
- Константы могут иметь только скалярные значения
- Объявляем константу регистра-независимой
  - `define("CONSTANT", "Здравствуй, мир.", true);`

# Лабораторная работа 2.4

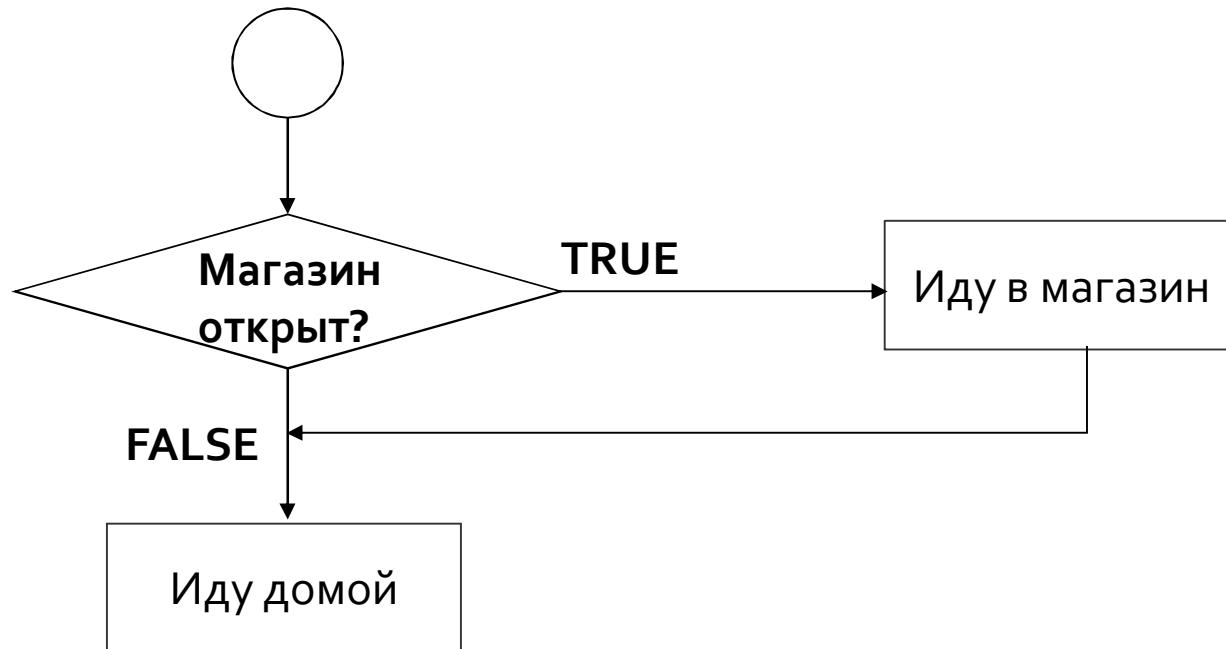
## Использование констант

### Упражнение 1: Вывод строки используя значение константы

- Откройте в текстовом редакторе файл **index.php**
- В самом начале файла в **php-блоке** напишите:  
`// Объявление константы  
define('COPYRIGHT', 'Супер Мега Веб-мастер');`
- Внизу файла в блоке **<!-- Нижняя часть страницы -->** вместо **Супер Мега Веб-мастер** напишите:  
`<?php echo COPYRIGHT?>`
- Сохраните файл **index.php**
- Посмотрите результат в браузере

# Управляющие конструкции: if

Надо зайти в магазин



# Управляющие конструкции: if

- Структура **if** реализована в PHP по аналогии с языком C:
  - if (условие)  
инструкция
- ```
if ($shop)
    echo "Иду в магазин";
■ if ($shop) {
    echo "Иду в магазин";
    echo "Покупаю хлеб";
}
■ if (defined("CONSTANT"))
    echo CONSTANT;
```

Преобразование в boolean

- При преобразовании в boolean, следующие значения рассматриваются как FALSE:
 - само значение boolean FALSE
 - integer 0 (ноль)
 - float 0.0 (ноль)
 - пустая строка и строка "0"
 - массив без элементов
 - особый тип NULL
- Все остальные значения рассматриваются как TRUE

Полезные функции: `isset` и `empty`

- `if (isset($var))`
 - Устанавливает, определена ли переменная
 - Возвращает TRUE, если переменная определена; FALSE в противном случае

- `if (empty($var))`
 - Определяет, считается ли переменная пустой
 - Возвращает FALSE, если переменная является непустой и ненулевым значением

Таблица сравнения типов

Выражение	gettype()	empty()	isset()	boolean : <i>if(\$x)</i>
<code>\$x = "";</code>	string	TRUE	TRUE	FALSE
<code>\$x = null;</code>	NULL	TRUE	FALSE	FALSE
<code>\$x</code> неопределена	NULL	TRUE	FALSE	FALSE
<code>\$x = array();</code>	array	TRUE	TRUE	FALSE
<code>\$x = false;</code>	boolean	TRUE	TRUE	FALSE
<code>\$x = true;</code>	boolean	FALSE	TRUE	TRUE
<code>\$x = 1;</code>	integer	FALSE	TRUE	TRUE
<code>\$x = 42;</code>	integer	FALSE	TRUE	TRUE
<code>\$x = 0;</code>	integer	TRUE	TRUE	FALSE
<code>\$x = -1;</code>	integer	FALSE	TRUE	TRUE
<code>\$x = "1";</code>	string	FALSE	TRUE	TRUE
<code>\$x = "0";</code>	string	TRUE	TRUE	FALSE
<code>\$x = "-1";</code>	string	FALSE	TRUE	TRUE
<code>\$x = "php";</code>	string	FALSE	TRUE	TRUE
<code>\$x = "true";</code>	string	FALSE	TRUE	TRUE
<code>\$x = "false";</code>	string	FALSE	TRUE	TRUE

Операторы сравнения

Пример	Название	Результат
<code>\$a == \$b</code>	Равно	TRUE если <code>\$a</code> равно <code>\$b</code> после преобразования типов.
<code>\$a === \$b</code>	Тождественно равно	TRUE если <code>\$a</code> равно <code>\$b</code> и имеет тот же тип.
<code>\$a != \$b</code>	Не равно	TRUE если <code>\$a</code> не равно <code>\$b</code> после преобразования типов.
<code>\$a !== \$b</code>	Тождественно не равно	TRUE если <code>\$a</code> не равно <code>\$b</code> или в случае, если они разных типов
<code>\$a < \$b</code>	Меньше	TRUE если <code>\$a</code> строго меньше <code>\$b</code> .
<code>\$a > \$b</code>	Больше	TRUE если <code>\$a</code> строго больше <code>\$b</code> .
<code>\$a <= \$b</code>	Меньше или равно	TRUE если <code>\$a</code> is меньше или равно <code>\$b</code> .
<code>\$a >= \$b</code>	Больше или равно	TRUE если <code>\$a</code> больше или равно <code>\$b</code> .

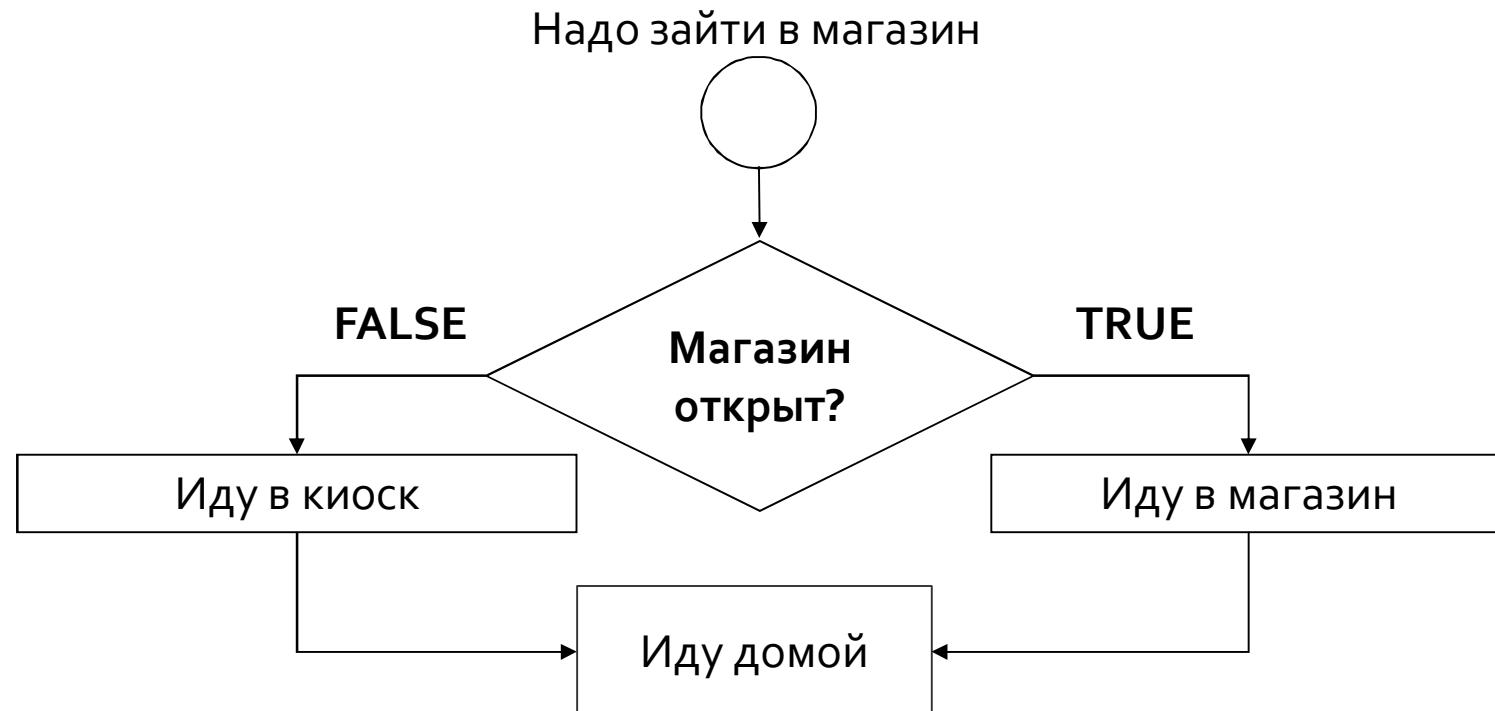
Сравнение различных типов

Тип операнда 1	Тип операнда 2	Результат
null или string	string	NULL преобразуется в "", числовое или лексическое сравнение
bool или null	что угодно	Преобразуется в bool, FALSE < TRUE
string, resource или number	string, resource или number	Строки и ресурсы переводятся в числа, обычная математика
array	array	Массивы с меньшим числом элементов считаются меньше, если ключ из первого операнда не найден во втором операнде - массивы не могут сравниваться, иначе идет сравнение соответствующих значений
array	что угодно	array всегда больше
object	что угодно	object всегда больше

Логические операторы

Пример	Название	Результат
<code>\$a and \$b</code>	И	TRUE если и <code>\$a</code> , и <code>\$b</code> TRUE.
<code>\$a or \$b</code>	Или	TRUE если или <code>\$a</code> , или <code>\$b</code> TRUE.
<code>\$a xor \$b</code>	Исключающее или	TRUE если <code>\$a</code> , или <code>\$b</code> TRUE, но не оба.
<code>!\$a</code>	Отрицание	TRUE если <code>\$a</code> не TRUE.
<code>\$a && \$b</code>	И	TRUE если и <code>\$a</code> , и <code>\$b</code> TRUE.
<code>\$a \$b</code>	Или	TRUE если или <code>\$a</code> , или <code>\$b</code> TRUE.

Управляющие конструкции: else



Управляющие конструкции: else

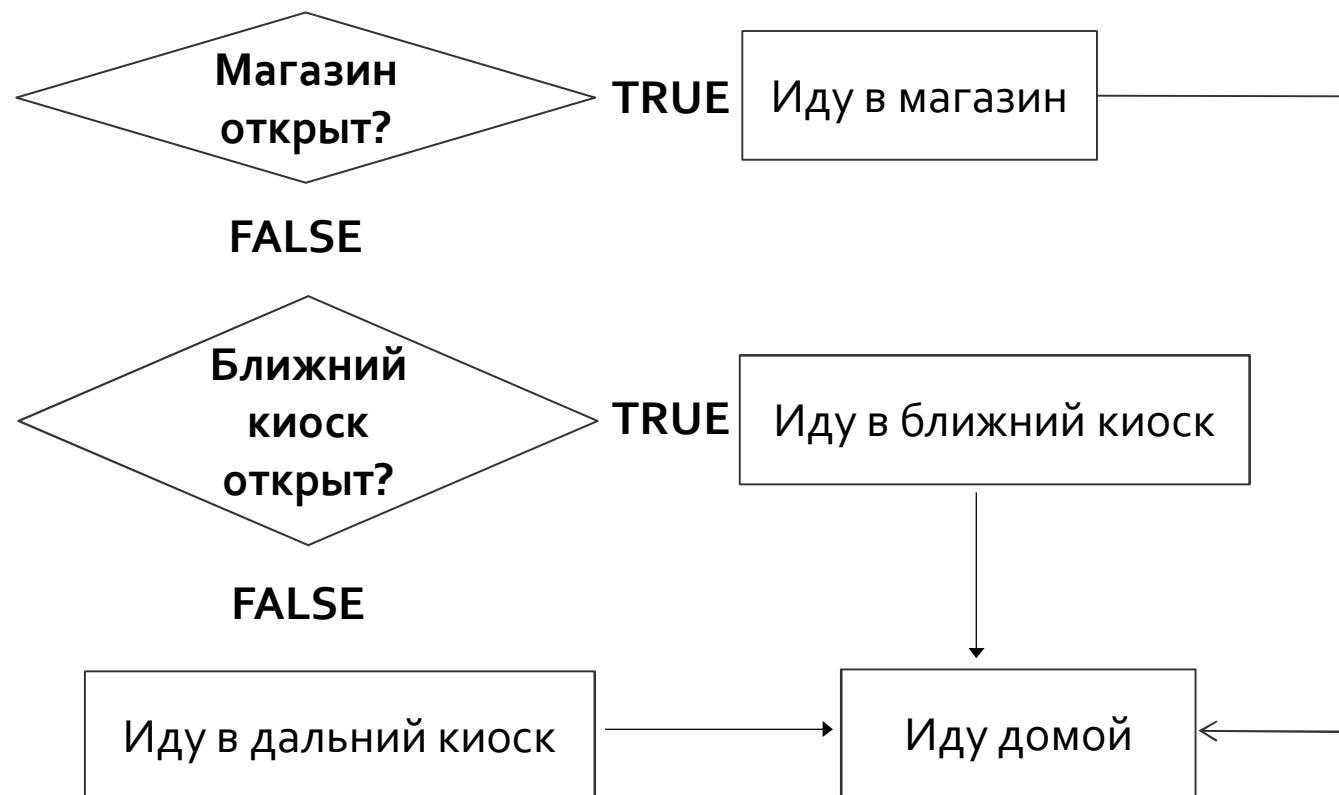
- ```
if ($shop == "open")
 echo "Иду в магазин";
else
 echo "Иду в киоск";
```
- ```
if ($shop and $money) {
    echo "Иду в магазин";
    echo "Покупаю хлеб";
} else {
    echo "Иду домой";
    echo "Туплю в телевизор";
}
```

Тернарный оператор

- ```
if ($shop)
 echo "Иду в магазин";
else
 echo "Иду в киоск";
```
- ```
echo ($shop) ? "Иду в магазин" : "Иду в киоск";
```
- Рекомендуется избегать "нагромождения" тернарных выражений.
- Поведение PHP неочевидно при использовании нескольких тернарных операторов в одном выражении:
 - ```
echo (true?'true':false?'t':'f');// 't'
```

# Управляющие конструкции: elseif

Надо зайти в магазин



# Управляющие конструкции: elseif

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

# Лабораторная работа 2.5

## Использование управляющей конструкции if - elseif - else

### Упражнение 1: Вывод приветствия в зависимости от времени суток

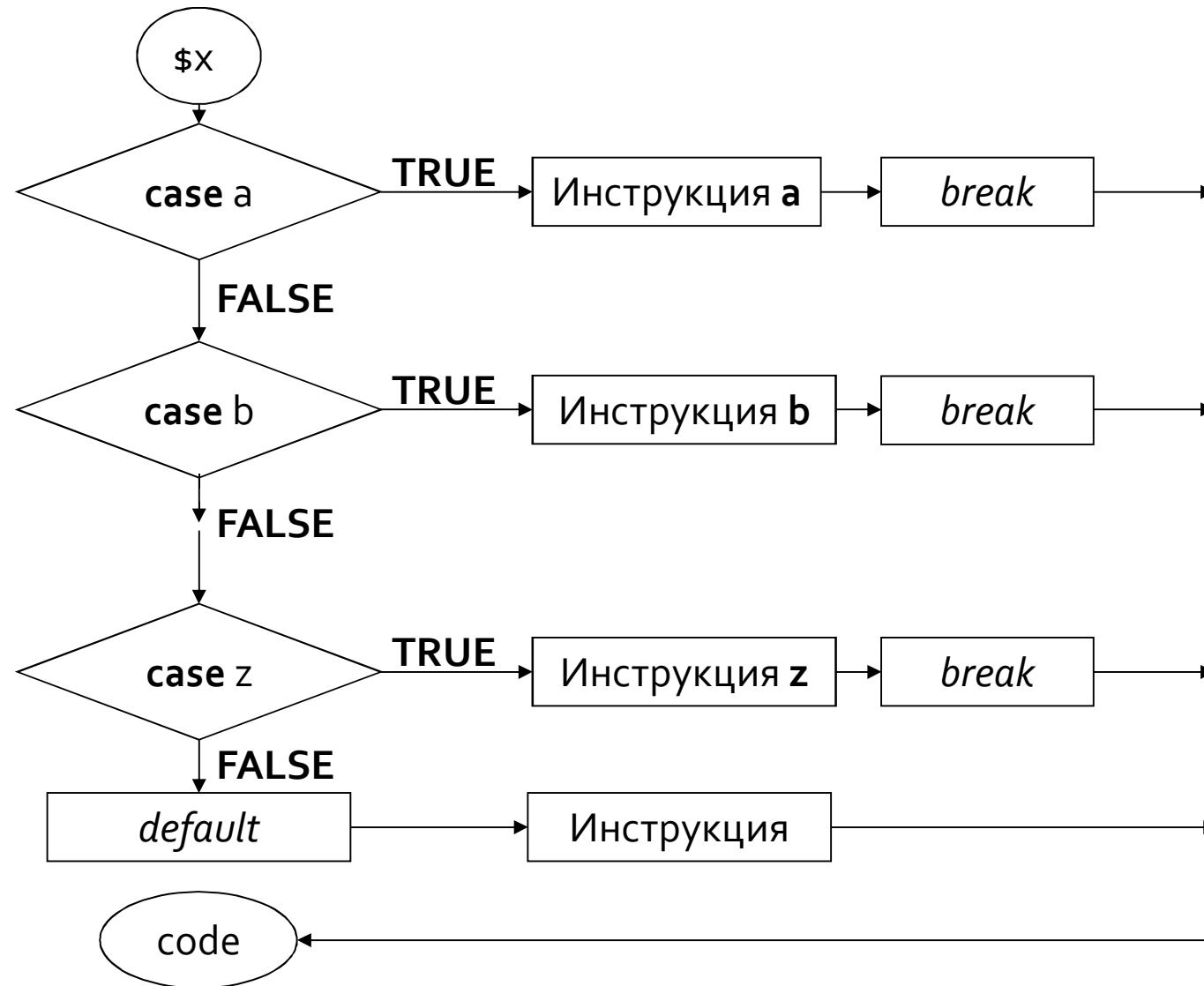
- Откройте в текстовом редакторе файл `index.php`
- В начале файла в php-блоке напишите:

```
/*
 * Получаем текущий час в виде строки от 00 до 23
 * и приводим строку к целому числу от 0 до 23
 */
$hour = (int)strftime('%H');
$welcome = '';// Инициализируем переменную для приветствия
```
- Используя управляющую конструкцию if – elseif - else присвойте переменной `$welcome` значение, изходя из следующих условий если число в переменной `$hour` попадает в диапазон:
  - от **0** до **6** - **Доброй ночи**
  - от **6** (включительно) до **12** - **Доброе утро**
  - от **12** (включительно) до **18** - **Добрый день**
  - от **18** (включительно) до **23** - **Добрый вечер**
- Если число в переменной `$hour` не попадает ни в один из вышеперечисленных диапазонов, то присвойте переменной `$welcome` значение **Доброй ночи**
- Между тэгами `<h1></h1>` вместо строки **Добро пожаловать на наш сайт!** напишите: `<?php echo $welcome?>`, **Гость!**
- Сохраните файл `index.php`
- Посмотрите результат в браузере

# Можно и так, но надо ли?

```
■ $day = 2;
■ if ($day == 1)
 echo "Понедельник";
elseif ($day == 2)
 echo "Вторник";
elseif ($day == 3)
 echo "Среда";
elseif ($day == 4)
 echo "Четверг";
elseif ($day == 5)
 echo "Пятница";
elseif ($day == 6)
 echo "Суббота";
elseif ($day == 7)
 echo "Воскресенье";
else
 echo "Неизвестный день";
```

# Управляющие конструкции: switch



# Управляющие конструкции: switch

```
■ $i = 1;
■ switch ($i) {
 case 0:
 echo "Результат: 0";
 case 1:
 echo "Результат: 1";
 case 2:
 echo "Результат: 2";
}
```

# Управляющие конструкции: switch

```
■ $i = 1;
■ switch ($i) {
 case 0:
 echo "Результат: 0"; break;
 case 1:
 echo "Результат: 1"; break;
 case 2:
 echo "Результат: 2"; break;
 default:
 echo "Результат: Много";
}
```

# Лабораторная работа 2.6

## Использование управляющей конструкции **switch**

### Упражнение 1: Вывод значения директивы PHP `post_max_size`

- Задача: вывести значение директивы `php.ini post_max_size` в байтах
- Откройте в текстовом редакторе файл **contact.php**
- В самом начале блока `<!-- Область основного контента -->` напишите:  
`<?php  
?>`
- Задайте два вопроса, необходимые для решения задачи
- Создайте переменную **\$size**, которая будет содержать текущее значение директивы `post_max_size`
- Получите данные, о величине в которой представленно значение (т.е. килобайты, мегабайты...)
- Используя управляющую конструкцию **switch** вычислите результат и сохраните его в переменную **\$size**
- После закрывающего тэга веб-формы `</form>` напишите:  
`<p>Максимальный размер отправляемых данных <?= $size?> байт.</p>`
- Сохраните файл **contact.php**
- Посмотрите результат в браузере

# Типы: array

- Создание пустого массива
  - `$arr = array();`
- Создание массива с элементами
  - `$arr = array("John", "root", "1234");`
- Обращение к элементу массива
  - `echo $arr[1]; // root`
- Добавление элементов массива
  - `$arr[] = 25;`
  - `$arr[] = true;`

# Полезные функции

- count
  - Считает количество элементов массива или количество свойств объекта
  - `$result = count($array);`
- print\_r
  - Выводит удобочитаемую информацию о переменной
  - `<pre><?php print_r($array);?></pre>`
- var\_dump
  - Выводит информацию о переменной
  - `<pre><?php var_dump($array);?></pre>`

# Индексация элементов

- Если массив еще не существует, он будет создан
  - `$arr[] = 1; // Это создает массив с одним элементом с ключом 0`
  - `$arr[5] = 2; // Это добавляет к массиву второй элемент с ключом 5`
  - `$arr[] = 3; // Это добавляет к массиву третий элемент с ключом 6`
  - `$arr = array(12 => 1, 5 => 2); // Это создает массив с двумя элементами с индексами 5 и 12`
  - `$arr[] = 3; // Это добавляет к массиву третий элемент с ключом 13`
- Удаление элемента массива
  - `unset($arr[12]); // Удаление элемента с ключом 12`

# Ассоциативный массив

- Ключ может быть либо integer, либо string
  - ```
$arr = array(  
    "name" => "John",  
    "login" => "root",  
    "pass" => "1234",  
    "age" => 25,  
    true);
```
 - Обращение к элементам массива
 - `echo $arr["name"];` // John
 - `echo $arr[0];` // 1

Лабораторная работа 2.7

Использование массива

Упражнение 1: Создание динамического меню

- Откройте в текстовом редакторе файл **index.php**

- В начале блока **<!-- Меню -->** напишите:

```
<?php  
$leftMenu = array(  
    'home'=>'index.php',  
    'about'=>'about.php',  
    'contacts'=>'contact.php',  
    'table'=>'table.php',  
    'calc'=>'calc.php'  
);  
?>
```

- В значениях атрибута **href** тэгов **<a>** вместо текущих значений выведите значения элементов массива **\$leftMenu** по следующему образцу:

```
<li><a href='<?= $leftMenu[ 'home' ]?>'>Домой</a></li>
```

- Сохраните файл **index.php**

- Посмотрите результат в браузере

Многомерный массив

- Значение может быть любого имеющегося в PHP типа
 - `$arr[0] = array(
 "login" => "vasya",
 "pass" => "1234");`
 - `$arr[1] = array(
 "login" => "petya",
 "pass" => "5678");`
 - `echo $arr[1]["login"]; // petya`

Лабораторная работа 2.8

Использование многомерного массива

Упражнение 1: Создание гибкого динамического меню

- Откройте в текстовом редакторе файл **index.php**

- В блоке **<!-- Меню -->** удалите массив **\$leftMenu**

- Вместо удаленного текста напишите:

```
<?php  
$leftMenu = array(  
    array('link'=>'Домой', 'href'=>'index.php'),  
    array('link'=>'О нас', 'href'=>'about.php'),  
    array('link'=>'Контакты', 'href'=>'contact.php'),  
    array('link'=>'Таблица умножения', 'href'=>'table.php'),  
    array('link'=>'Калькулятор', 'href'=>'calc.php')  
)  
?  
;
```

- Измените отрисовку ссылок на:

```
<li><a href='<?= $leftMenu[0]['href']?>'><?= $leftMenu[0]['link']?></a></li>  
<li><a href='<?= $leftMenu[1]['href']?>'><?= $leftMenu[1]['link']?></a></li>  
<li><a href='<?= $leftMenu[2]['href']?>'><?= $leftMenu[2]['link']?></a></li>  
<li><a href='<?= $leftMenu[3]['href']?>'><?= $leftMenu[3]['link']?></a></li>  
<li><a href='<?= $leftMenu[4]['href']?>'><?= $leftMenu[4]['link']?></a></li>
```

- Сохраните файл **index.php**

- Посмотрите результат в браузере

Короткий синтаксис массивов

- Начиная с версии PHP 5.4
 - `$arr1 = ["vasya", "1234"];`
 - `$arr2 = [
 "login" => "petya",
 "pass" => "5678"
];`

Полезные функции

- `$array = array('первый', 'второй', 'третий', 'четвертый');`
- По умолчанию внутренний указатель массива указывает на первый элемент
 - `echo current($array); // вернёт "первый"`
- Передвигает внутренний указатель массива на одну позицию вперёд
 - `next($array); // echo next($array) вернёт "второй"`
 - `echo current($array); // вернёт "второй"`
- Устанавливаем внутренний указатель массива на его последний элемент
 - `end($array); // echo end($array) вернёт "четвертый"`
- Передвигаем внутренний указатель массива на одну позицию назад
 - `prev($array); // echo prev($array) вернёт "третий"`
- Устанавливаем внутренний указатель массива на его первый элемент
 - `reset($array); // echo reset($array) вернёт "первый"`
- Получаем ключ текущего элемента
 - `echo key($array); // вернёт 0`

Выводы

- Синтаксис
- Операторы
- Переменные
- Константы
- Типы
- Управляющие конструкции
- Массивы

PHP Уровень 1

Циклы

Игорь Борисов
<http://igor-borisov.ru>

Темы модуля

- Операторы инкремента/декремента
- Цикл for
- Цикл while
- Цикл do-while
- Итерирование массива
- Цикл foreach

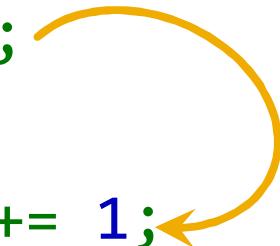
Операторы инкремента/декремента

- **++**
 - Увеличивает значение переменной на единицу
- **--**
 - Уменьшает значение переменной на единицу
- **PRE инкремент/декремент**
 - **+\$a** и **-\$a**
 - Увеличивает или уменьшает \$a на единицу и возвращает значение \$a.
- **POST инкремент/декремент**
 - **\$a++** и **\$a--**
 - Возвращает значение \$a, а затем увеличивает или уменьшает \$a на единицу.

Пример оператора инкремента

■ Постфиксный инкремент

- `$a = 1;`
- `echo "Должно быть 1: " . $a++;`
- `echo "Должно быть 2: " . $a;`
- `echo "Должно быть 1: $a"; $a += 1;`



■ Префиксный инкремент

- `$a = 1;`
- `echo "Должно быть 2: " . ++$a;`
- `echo "Должно быть 2: " . $a;`
- `$a += 1; echo "Должно быть 2: $a";`



Цикл for

- `for (Часть А; Часть В; Часть С) {
 // Тело цикла
}`
- `for ($i = 1; $i <= 10; $i++) {
 echo $i;
}`

Лабораторная работа 3.1

Использование цикла for

Упражнение 1: Вывод нечетных чисел из заданного диапазона

- Откройте проводник Windows
- Перейдите в папку **C:\Users\Public\OpenServer\domains\mysite.local\demo**
- Создайте в текущей папке файл **for.php**
- Откройте созданный файл **for.php** в текстовом редакторе
- Используя цикл **for** выведите в столбик **нечётные** числа от **1** до **50**
- Сохраните файл **for.php**
- Посмотрите результат в браузере

Цикл while

- `while (Условие) {
 // Тело цикла
}`
- `$i = 1;
while ($i <= 10) {
 echo $i++;
}`

Лабораторная работа 3.2

Использование цикла while

Упражнение 1: Вывод строки посимвольно

- Откройте проводник Windows
- Перейдите в папку **C:\Users\Public\OpenServer\domains\mysite.local\demo**
- Создайте в текущей папке файл **while.php**
- Откройте созданный файл **while.php** в текстовом редакторе
- Создайте переменную **\$var** и присвойте ей строковое значение **HELLO**
- Используя цикл **while** выведите значение переменной **\$var** в столбик так, чтобы на выходе в браузере получилось:

H
E
L
L
O

- Сохраните файл **while.php**
- Посмотрите результат в браузере

Цикл do-while

- ```
$i = 100;
do {
 echo $i++;
} while ($i <= 10);
```

# Управление циклами: break

- ```
$i = 1;
while ($i <= 10) {
    echo $i++;
    if($i == 5)
        break;
/* Тут можно было написать 'break 1;'. */
}
```
- Выведет: 1234

Управление циклами: continue

- ```
$i = 0;
while ($i < 9) {
 $i++;
 if($i == 5)
 continue;
 echo $i;
/* Тут можно было написать 'continue 1;'. */
}
```
- Выведет: 12346789

# Лабораторная работа 3.3

## Создание динамической таблицы умножения

### Упражнение 1: Создание HTML-таблицы

- В текстовом редакторе откройте файл **table.php**
- В начале файла создайте php-блоке, в котором создайте две целочисленные переменные **\$cols** и **\$rows**
- Присвойте созданным переменным произвольные значения в диапазоне от **1** до **10**
- В блоке **<!-- Таблица -->** удалите весь html-код и напишите:  
**<?php**  
**?>**
- В текущем php-блоке используя циклы **for** отрисуйте таблицу умножения в виде HTML -таблицы на следующих условиях:
  - Число **столбцов** должно быть равно значению переменной **\$cols**
  - Число **строк** должно быть равно значению переменной **\$rows**
  - Ячейки на пересечении столбцов и строк должны содержать значения, являющиеся **произведением** порядковых номеров столбца и строки
- Сохраните файл **table.php**
- Посмотрите результат в браузере

### Упражнение 2: Приводим таблицу к товарному виду

- Отрисуйте значения в ячейках первой строки и первого столбца полужирным шрифтом и выровняйте их по центру ячейки
- Сделайте фоновый цвет ячеек первой строки и первого столбца отличным от фонового цвета таблицы
- Сохраните файл **table.php**
- Посмотрите результат в браузере

# Цикл foreach

- `foreach (array as $value) {  
 // Тело цикла  
}`
- `foreach (array as $key => $value) {  
 // Тело цикла  
}`

# Цикл foreach: примеры

- ```
$arr = array('a'=>'one', 'b'=>'two', 'c'=>'three');
```
- ```
foreach ($arr as $val) {
 echo "$val\n";
}
```
- ```
foreach ($arr as $key => $val) {  
    echo "$key => $val\n";  
}
```
- ```
foreach ($arr as &$val) {
 $val = "__$val__";
}
```

# Лабораторная работа 3.4

## Создание динамического меню навигации по сайту

### Упражнение 1: Вывод меню с использованием цикла

- В текстовом редакторе откройте файл **index.php**
- Перенесите в php-блок в начале файла код инициализации массива из блока **<!-- Меню -->**, то есть:

```
$leftMenu = array(
 array('link'=>'домой', 'href'=>'index.php');
...

```

добавив комментарий **// Инициализация массива**
- В блоке **<!-- Меню -->** удалите всё **html-содержимое** (от **<ul>** до **</ul>** включительно)
- В php-блоке блока **<!-- Меню -->** отрисуйте вертикальное меню с помощью цикла **foreach**, передав ему в качестве аргумента массив **\$leftMenu**.  
Обратите **внимание**, что массив - многомерный.
- Сохраните файл **index.php**
- Посмотрите результат в браузере

# Выводы

- Операторы инкремента/декремента
- Цикл for
- Цикл while
- Цикл do-while
- Цикл foreach

# **PHP Уровень 1.**

# **Пользовательские**

# **функции**

Игорь Борисов  
<http://igor-borisov.ru>

# Темы модуля

- Описание функции
- Вызов функции
- Аргументы функции
- Аргументы функции по умолчанию
- Область видимости переменных
- Статические переменные
- Возврат значений
- Рекурсивная функция

# Описание и вызов функции

- Функция – программный блок, который может многократно выполняться в любом месте сценария
- Описание функции
  - `function sayHello(){  
 echo "<h1>Привет мир!</h1>";  
}`
- Вызов функции
  - `sayHello();`
- Проверка функции на существование
  - `if (function_exists("sayHello")){}`

# Правила использования

- Внутри функции можно использовать любой корректный PHP-код, в том числе другие функции
- Корректное имя функции начинается с буквы или знака подчеркивания, за которым следует любое количество букв, цифр или знаков подчеркивания
- Все функции PHP имеют глобальную область видимости - они могут быть вызваны вне функции, даже если были определены внутри и наоборот
- PHP не поддерживает перегрузку функции, также отсутствует возможность переопределить или удалить объявленную ранее функцию
- Функции не обязаны быть определены до их использования, исключая тот случай, когда функции определяются условно. В этом случае, обработка описания функции должна предшествовать ее вызову

# Функции, зависящие от условий

- `$makefoo = true;`  
    /\* Мы не можем вызвать функцию foo() в этом месте,  
     поскольку она еще не определена, но мы можем  
     обратиться к bar() \*/
- `bar();`
- `if ($makefoo) {`  
        `function foo() {`  
            `echo "Я не существую до тех пор, пока выполнение программы  
мы меня не достигнет.\n";`  
            `}`  
        `}`  
    /\* Теперь мы благополучно можем вызывать foo(),  
     поскольку \$makefoo была интерпретирована как true \*/
- `if ($makefoo) foo();`
- `function bar() {`  
        `echo "Я существую сразу с начала старта программы.\n";`  
    `}`

# Вложенные функции

- ```
function foo() {  
    function bar() {  
        echo "Я не существую пока не будет вызвана foo().\n";  
    }  
}  
/* Мы пока не можем обратиться к bar(),  
поскольку она еще не определена. */
```
- ```
foo();
/* Теперь мы можем вызвать функцию bar(),
обработка foo() сделала ее доступной. */
```
- ```
bar();
```

Аргументы функций

- ```
function sayHello($name){
 echo "<h1>Привет $name!</h1>";
}
```
- ```
sayHello("John");
```
- ```
$user = "Mike";
sayHello($user);
```
- ```
$func = "sayHello";  
$func("Guest");
```

Лабораторная работа 4.1

Создание функции отрисовки таблицы умножения

Упражнение 1: Отрисовка таблицы с помощью функции

- В текстовом редакторе откройте файл **table.php**
- В самом начале файла создайте php-блок
- В текущем php-блоке создайте функцию **drawTable()**
- Задайте для функции три аргумента: **\$cols, \$rows, \$color**
- Перенесите код (**[Ctrl] + [X]**), который отрисовывает таблицу умножения из блока **<!-- Таблица -->** в тело функции
- В блоке **<!-- Таблица -->** (там, где ранее отрисовывалась таблица) отрисуйте таблицу умножения вызывая функцию **drawTable()** с произвольными параметрами
- Сохраните файл **table.php**
- Посмотрите результат в браузере

Аргументы по умолчанию

- ```
function sayHello($name="Guest"){
 echo "<h1>Привет $name!</h1>";
}
```
- ```
sayHello("John"); // Привет John!
```
- ```
sayHello(); // Привет Guest!
```

# Области видимости переменных

- ```
function sayHello($name){  
    echo "<h1>Привет $name!</h1>";  
    $name = "Вася";  
}  
sayHello("John");
```
- ```
$name = "Mike";
sayHello($name);
```
- ```
echo $name; // ???
```

Обращение к глобальным переменным: Вариант 1

- ```
function sayHello($name){
 echo "<h1>Привет $name!</h1>";
 global $name;
 $name = "Вася";
}
```
- ```
$name = "Mike";  
sayHello($name);
```
- ```
echo $name; // Вася
```

## Обращение к глобальным переменным: Вариант 2

- ```
function sayHello($name){  
    echo "<h1>Привет $name!</h1>";  
    $GLOBALS['name'] = "Вася";  
}  
  
$name = "Mike";  
sayHello($name);  
  
echo $name; // Вася
```

Передача аргументов по ссылке

- ```
function sayHello($name){
 echo "<h1>Привет $name!</h1>";
 $name = "Вася";
}
sayHello("John");
$name = "Mike";
sayHello($name);
sayHello(&$name); // Передается
ссылка на переменную.
// С PHP 5.3 генерируется предупреждение
```

# Передача аргументов по ссылке

- ```
function sayHello(&$name){  
    echo "<h1>Привет $name!</h1>";  
    $name = "Вася";  
}
```
- `sayHello("John"); // ОШИБКА!`
- `$name = "Mike";
sayHello($name); // Передается
ссылка на переменную`
- `sayHello(&$name); // Передается
ссылка на переменную.
//С PHP 5.3 генерируется предупреждение`

Статические переменные

- ```
function test(){
 $a = 0;
 echo $a++;
}
```

  - `test(); // 0`
  - `test(); // 0`
  - `test(); // 0`
  
- ```
function test(){
    static $a = 0;
    echo $a++;
}
```

 - `test(); // 0`
 - `test(); // 1`
 - `test(); // 2`

Лабораторная работа 4.2

Создание функции отрисовки меню навигации по сайту

Упражнение 1: Отрисовка меню с помощью функции

- В текстовом редакторе откройте файл `index.php`
- В начале файла в php-блоке создайте функцию `drawMenu()`
- Задайте для функции первый аргумент `$menu` - в него будет передаваться массив, содержащий структуру меню
- Задайте для функции второй аргумент `$vertical` со значением по умолчанию равным `true`. Данный параметр указывает, каким образом будет отрисовано меню - вертикально или горизонтально
- Перенесите код (**[Ctrl] + [X]**), который отрисовывает меню навигации из блока `<!-- Меню -->` в тело функции
- Измените код таким образом, чтобы меню отрисовывалось в зависимости от входящего параметра `$vertical` - либо вертикально, либо горизонтально
- В блоке `<!-- Меню -->` (там, где ранее отрисовывалось меню) отрисуйте меню навигации вызвав функцию `drawMenu()` с необходимыми параметрами
- Сохраните файл `index.php`
- Посмотрите результат в браузере

Возврат значений

- ```
function square($num) {
 return $num * $num;
 // Этот код никогда не исполнится
 echo "Мертвый код";
}
echo square(4); // выводит 16
$result = square(4);
```

# Возвращение массивов

- ```
function numbers() {  
    return array (0, 1, 2);  
}  
list ($zero, $one, $two) = numbers();
```
- PHP 5.4 Разыменование массивов
- ```
function numbers() {
 return array (0, 1, 2);
}
$number = numbers()[1];
```

# Рекурсивная функция

```
■ function recursion($a) {
 if ($a < 20) {
 echo "$a\n";
 recursion($a + 1);
 }
}
```

# Использование аргументов переменной длины

- ```
function foo() {  
    $numargs = func_num_args();  
    echo "Всего аргументов: $numargs\n";  
    echo "Второй  
          аргумент: " . func_get_arg(1) . "\n";  
    $args = func_get_args();  
    for ($i = 0; $i < $numargs; $i++) {  
        echo "Аргумент $i: " . $args[$i] . "\n";  
    }  
}
```
- `foo(1, 2, 3);`

Уточнение типа (type-hint)

- `function foo(array $var) {`
 `// Ожидается только массив!`
}
- PHP 5.4
- `function bar(callable $var) {`
 `return $var();`
}
- `bar("function_name");`

Выводы

- Описание функции
- Вызов функции
- Аргументы функции
- Аргументы функции по умолчанию
- Область видимости переменных
- Статические переменные
- Возврат значений
- Рекурсивная функция

PHP Уровень 1. Что внутри PHP?

Игорь Борисов
<http://igor-borisov.ru>

Темы модуля

- Учимся читать документацию
- Встроенные функции
 - Математические функции
 - Функции для работы с переменными
 - Функции обработки строк
 - Функции для работы с массивами
 - Функции даты и времени
- Встроенные константы
- Суперглобальные переменные
- Функции эмуляции SSI

Документация PHP

The screenshot shows the official PHP documentation website. A red box highlights the 'documentation' link in the top navigation bar. A large red circle labeled '1' points to the 'What is PHP?' section, which includes a brief introduction and links to calling for papers and upcoming conferences. Another red circle labeled '2' points to the 'Formats' table, specifically the 'View Online' row, which lists various language versions. A third red circle labeled '3' points to the 'Russian' download link at the bottom.

What is PHP?

PHP is a widely-used open source scripting language that is especially well-suited for Web development.

1

Calling for papers: [PHP @ FrOSCon](#)

Upcoming conferences: [php|tek 2008: Chicago](#)

Stable Releases

Current PHP 5 Stable: **5.2.5**

Historical PHP 4 Stable: **4.4.8**

Formats	Destinations
View Online	English, Brazilian Portuguese, Chinese (Simplified), Chinese (Traditional), Cantonese, Czech, Danish, Dutch, Finnish, French, German, Greek, Hebrew, Hungarian, Italian, Japanese, Korean, Polish, Romanian, Russian , Slovak, Spanish, Swedish
Downloads	For downloadable formats, please visit our documentation downloads page.

2

3

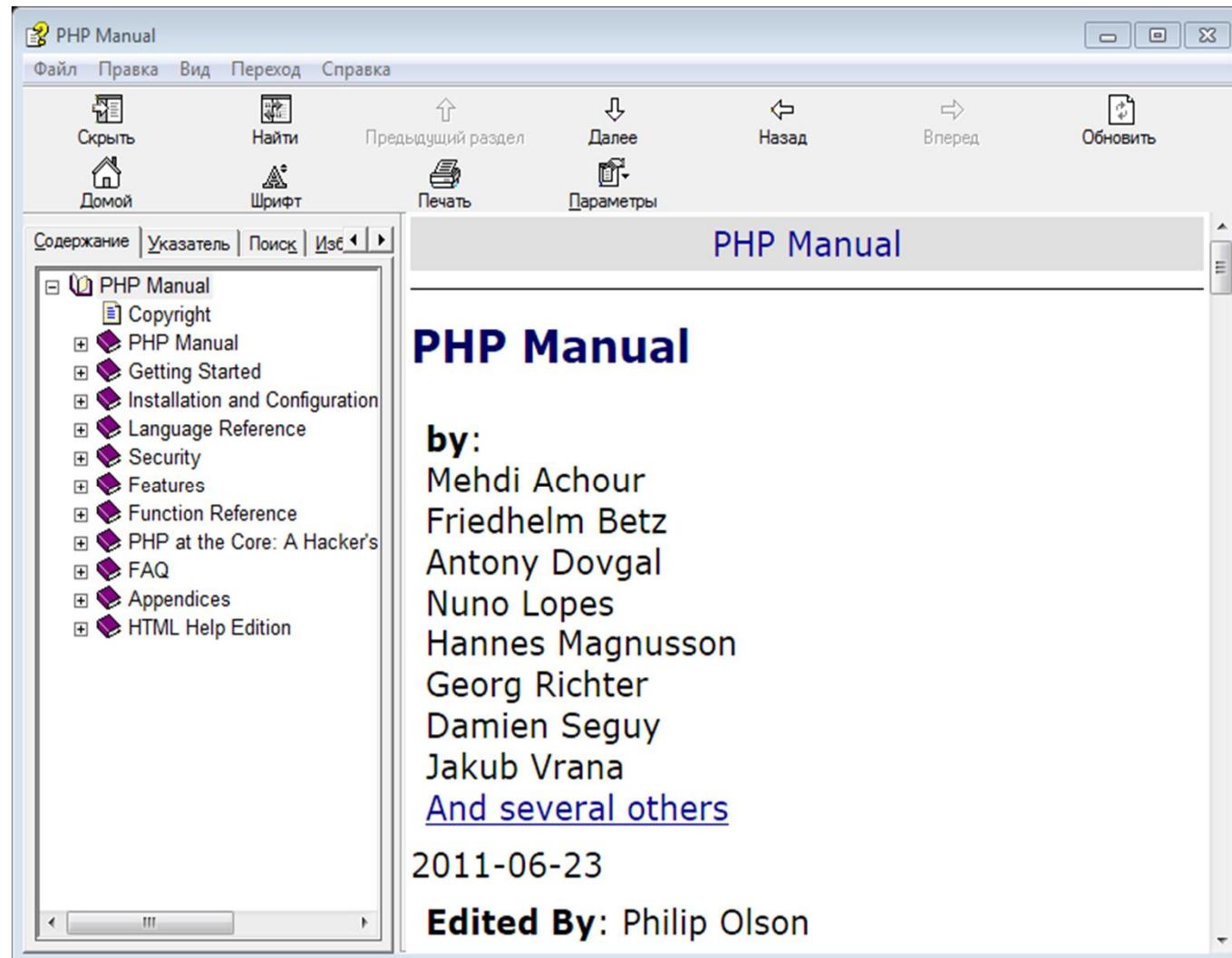
Russian

[html.gz](#)
Size: 2762Kb
Date: 25 Nov 2007

[tar.gz](#)
Size: 3437Kb
Date: 25 Nov 2007

[chm](#)
Size: 7240Kb
Date: 7 Dec 2007

Документация в формате chm



Как читать прототипы функций

strlen

(PHP 4, PHP 5)

`strlen` — Возвращает длину строки

[Report a bug](#)

Описание

```
int strlen ( string $string )
```

Возвращает длину строки *string*.

isset

(PHP 4, PHP 5)

`isset` — Устанавливает, определена ли переменная

[Report a bug](#)

Описание

```
bool isset ( mixed $var [, mixed $var [, $... ] ] )
```

Устанавливает, определена ли переменная.

Функции для работы с переменными

- `is_array`, `is_bool`, `is_float`, `is_int`, `is_null`, `is_string`, `is_numeric`
 - Определяет, является ли переменная тем или иным типом
- `isset`
- `empty`
- `settype`
- `gettype`
- `print_r`
- `var_dump`
- `unset`

- `intval`
 - Возвращает целое значение переменной
- `get_defined_vars`
 - Возвращает массив всех определенных переменных

Математические функции

- **abs**
 - Модуль числа
- **max**
 - Находит наибольшее значение
- **min**
 - Находит наименьшее значение
- **rand**
 - Генерирует случайное число
- **round**
 - Округляет число типа float
- **ceil**
 - Округляет дробь в большую сторону
- **floor**
 - Округляет дробь в меньшую сторону

Функции обработки строк

- **explode**
 - Разбивает строку на подстроки
- **substr_count**
 - Возвращает число вхождений подстроки
- **rtrim, ltrim, trim**
 - Удаление пробелов из начала и (или) конца строки
- **strtolower, strtoupper**
 - Преобразование строки в нижний и верхний регистр
- **lcfirst, ucfirst**
 - Преобразование первого символа строки в нижний и верхний регистр
- **printf, sprintf, vprintf, sscanf, fscanf**
 - Семейство print

Спецификаторы форматирования

- **%** - символ процента. Аргумент не используется.
- **b** - аргумент трактуется как целое и выводится в виде двоичного числа.
- **c** - аргумент трактуется как целое и выводится в виде символа с соответствующим кодом ASCII.
- **d** - аргумент трактуется как целое и выводится в виде десятичного числа со знаком.
- **e** - аргумент трактуется как float и выводится в научной нотации (например 1.2e+2).
- **u** - аргумент трактуется как целое и выводится в виде десятичного числа без знака.
- **f** - аргумент трактуется как float и выводится в виде десятичного числа с плавающей точкой.
- **o** - аргумент трактуется как целое и выводится в виде восьмеричного числа.
- **s** - аргумент трактуется как строка.
- **x** - аргумент трактуется как целое и выводится в виде шестнадцатиричного числа (в нижнем регистре букв).
- **X** - аргумент трактуется как целое и выводится в виде шестнадцатиричного числа (в верхнем регистре букв).

Функции семейства print

- Стандартное использование
 - `$num = 5; $location = "tree";`
 - `$format = "There are %d monkeys in the %s";`
 - `printf($format, $num, $location);`
 - `// There are 5 monkeys in the tree`
- Изменение порядка параметров
 - `$format = "The %s contains %d monkeys"; // Проблема`
 - `$format = "The %2\$s contains %1\$d monkeys";`
 - `printf($format, $num, $location);`
 - `// There tree contains 5 monkeys`
- Использование одного аргумента несколько раз
 - `$format = "The %2\$s contains %1\$d monkeys.`
`That's a nice %2\$s full of %1\$d monkeys.";`
 - `printf($format, $num, $location);`

Извлечение подстроки

- string **substr** (string \$string , int \$start [, int \$length])
 - Возвращает подстроку строки string длиной length, начинающегося с start символа по счету.
 - Если start неотрицателен, возвращаемая подстрока начинается в позиции start от начала строки, считая от нуля. Например, в строке 'abcdef', в позиции 0 находится символ 'a', в позиции 2 - символ 'c', и т.д.
 - `$rest = substr("abcdef", 1); //возвращает "bcdef"`
`$rest = substr("abcdef", 1, 3); //возвращает "bcd"`
`$rest = substr("abcdef", 0, 4); //возвращает "abcd"`
`$rest = substr("abcdef", 0, 8); //возвращает "abcdef"`
 - `$rest = substr("abcdef", -1); //возвращает "f"`
`$rest = substr("abcdef", -3, 1); //возвращает "d"`
 - `$rest = substr("abcdef", 2, -1); //возвращает "cde"`
`$rest = substr("abcdef", 4, -4); //возвращает ""`

Форматирование строки

- string **number_format** (float \$number [, int \$decimals])
string **number_format** (float \$number , int \$decimals , string \$dec_point , string \$thousands_sep)
 - Возвращает отформатированное число number
 - Функция принимает один, два или четыре аргумента (не три):
 - Если передан только один аргумент, number будет отформатирован без дробной части, но с запятой (",") между группами цифр по 3
 - Если переданы два аргумента, number будет отформатирован с decimals знаками после точки (".") и с запятой (",") между группами цифр по 3
 - Если переданы все четыре аргумента, number будет отформатирован с decimals знаками после точки и с разделителем между группами цифр по 3, при этом в качестве десятичной точки будет использован dec_point, а в качестве разделителя групп - thousands_sep
- ```
$number = 1234.56;
$english_format_number = number_format($number); // 1,234
$nombre_format_francais = number_format($number, 2, ',', ' ');
// 1 234,56
```
- ```
$number = 1234.5678;
$english_format_number = number_format($number, 2, '.', '');
// 1234.57
```

Замена подстроки

- mixed **str_replace** (mixed \$search , mixed \$replace , mixed \$subject [, int \$&count])
 - Эта функция возвращает строку или массив subject, в котором все вхождения search заменены на replace
 - Если subject - массив, поиск и замена производится в каждом элементе этого массива, и возвращается также массив.
 - Если и search, и replace - массивы, то используются все значения массива search и соответствующие значения массива replace для поиска и замены в subject.
 - Если в массиве replace меньше элементов, чем в search, в качестве строки замены для оставшихся значений будет использована пустая строка.
 - Если search - массив, а replace - строка, то replace будет использована как строка замены для каждого элемента массива search.
 - `$bodytag = str_replace("%body%", "black", "<body text='%body%'>"); // присваивает <body text='black'>`
 - `$str = str_replace("ll", "", "good golly miss molly!", $count); echo $count; // 2`

Разбивка строки на подстроки

- array **explode** (string \$separator , string \$string [, int \$limit])
 - Возвращает массив строк, полученных разбиением строки string с использованием separator в качестве разделителя
 - Если передан аргумент limit, массив будет содержать максимум limit элементов, при этом последний элемент будет содержать остаток строки string.
 - Если separator - пустая строка (""), возвращается FALSE. Если separator не содержится в string, то возвращается массив, содержащий один элемент string.
 - separator всегда должен содержать разделитель, а string - исходную строку
- ```
$pizza = "piece1 piece2 piece3 piece4";
$pieces = explode(" ", $pizza);
```
- ```
$data = "John:root:1234";
list($name, $login, $pass) = explode(":", $data);
```

Функции для работы с массивами

- `array_key_exists`
 - Проверяет, присутствует ли в массиве указанный ключ или индекс
- `in_array`
 - Проверяет, присутствует ли в массиве значение
- `array_keys`
 - Возвращает все или некоторое подмножество ключей массива
- `array_values`
 - Выбирает все значения массива
- `array_merge`
 - Сливает один или большее количество массивов
- `array_diff`
 - Вычисляет расхождение массивов
- `array_intersect`
 - Вычисляет схождение массивов

Стеки и очереди

- **int array_push (array &\$array , mixed \$var [, mixed \$...])**
 - Добавляет один или несколько элементов в конец массива
 - Использует array как стэк, и добавляет переданные значения в конец массива array.
 - Длина array увеличивается на количество переданных значений.
 - Имеет тот же эффект, что и выражение \$array[] = \$var; повторенное для каждой var.
 - Вместо использования array_push() для добавления одного элемента в массив, лучше использовать \$array[] = , потому что в этом случае не происходит затрат на вызов функции.
 - `$stack = array("orange", "banana");
array_push($stack, "apple", "raspberry");`
- **mixed array_pop (array &\$array)**
 - Извлекает и возвращает последнее значение параметра array, уменьшая размер array на один элемент.
 - Если array пуст (или не является массивом), будет возвращён NULL.
 - Эта функция сбрасывает указатель массива после использования.
 - `$stack = array("orange", "banana", "apple", "raspberry");
$fruit = array_pop($stack);`

Стеки и очереди

- **int array_unshift (array &\$array , mixed \$var [, mixed \$...])**
 - Добавляет переданные в качестве аргументов элементы в начало массива array
 - Обратите внимание, что список элементов добавляется целиком, то есть порядок элементов сохраняется
 - Все числовые ключи будут изменены таким образом, что нумерация массива будет начинаться с нуля, в то время как строковые ключи останутся прежними
 - `$queue = array("orange", "banana");
array_unshift($queue, "apple", "raspberry");`
- **mixed array_shift (array &\$array)**
 - Извлекает первое значение массива array и возвращает его, сокращая размер array на один элемент
 - Все числовые ключи будут изменены таким образом, что нумерация массива начнётся с нуля, в то время как строковые ключи останутся прежними
 - Эта функция сбрасывает указатель массива после использования
 - `$stack = array("orange", "banana", "apple", "raspberry");
$fruit = array_shift($stack);`

Сортировка массива

- **bool sort (array &\$array [, int \$sort_flags = SORT_REGULAR])**
 - Эта функция сортирует массив
 - После завершения работы функции элементы массива будут расположены в порядке возрастания
- **bool ksort (array &\$array [, int \$sort_flags = SORT_REGULAR])**
 - Сортирует массив по ключам, сохраняя отношения между ключами и значениями
 - Эта функция полезна, в основном, для работы с ассоциативными массивами
- **bool asort (array &\$array [, int \$sort_flags = SORT_REGULAR])**
 - Эта функция сортирует массив таким образом, что сохраняются отношения между ключами и значениями
 - Она полезна, в основном, при сортировке ассоциативных массивов, когда важно сохранить отношение ключ => значение

Сортировка массива

- rsort — Сортирует массив в обратном порядке
- arsort — Сортирует массив в обратном порядке, сохраняя ключи
- krsort — Сортирует массив по ключам в обратном порядке
- natsort — Сортирует массив, используя алгоритм "natural order"
- natcasesort — Сортирует массив, используя алгоритм "natural order" без учета регистра символов
- usort — Сортирует массив по значениям используя пользовательскую функцию для сравнения элементов
- uksort — Сортирует массив по ключам, используя пользовательскую функцию для сравнения ключей
- uasort — Сортирует массив, используя пользовательскую функцию для сравнения элементов с сохранением ключей

Функции даты и времени

- array **getdate** ([int \$timestamp = time()])
 - Возвращает ассоциативный массив (array), содержащий информацию о дате, представленной меткой времени timestamp, или текущем системном времени, если timestamp не был передан
- **\$today = getdate();**
print_r(\$today);
 - [seconds] => 40
 - [minutes] => 58
 - [hours] => 21
 - [mday] => 17
 - [wday] => 2
 - [mon] => 6
 - [year] => 2003
 - [yday] => 167
 - [weekday] => Tuesday
 - [month] => June
 - [0] => 1055901520

Метка времени timestamp

- **int time (void)**
 - Возвращает количество секунд, прошедших с начала Эпохи Unix (The Unix Epoch, 1 января 1970 00:00:00 GMT) до текущего времени
 - // Что-то типа 1234567890
`echo time();`
 - // 7 дней; 24 часа; 60 минут; 60 секунд
`$nextWeek = time() + (7 * 24 * 60 * 60);`
- **int mktime ([int \$hour [, int \$minute [, int \$second [, int \$month [, int \$day [, int \$year [, int \$is_dst]]]]]]])**
 - Функция возвращает метку времени Unix, соответствующую дате и времени, заданным аргументами
 - `echo mktime(0, 0, 0, 1, 1, 2011);`
`echo mktime(0, 0, 0, 13, 1, 2010);`
`echo mktime(0, 0, 0, 12, 32, 2010);`

Форматирование даты и времени

- **string date (string \$format [, int \$timestamp])**
 - Возвращает строку со временем, отформатированную в соответствии с указанным форматом, используя метку времени, заданную аргументом timestamp или текущее системное время, если timestamp не задан
- // установим часовой пояс по умолчанию
`date_default_timezone_set('UTC');`
- `echo date("l");` // выведет что-то вроде: Monday
- // выведет что-то вроде: 25-11-2011 12:46:33
`echo date('d-m-Y H:i:s');`
- // выведет что-то вроде: Mon, 15 Aug 2005
15:12:46 UTC
`echo date('r', mktime(15, 12, 46, 8, 15, 2005));`

Преобразование строки

- **int strtotime (string \$time [, int \$now])**
 - Преобразует текстовое представление даты на английском языке в метку времени Unix
 - Первым параметром функции должна быть строка с датой на английском языке, которая будет преобразована в метку времени относительно метки времени, переданной в now, или текущего времени, если аргумент now опущен
- `$dt = strtotime("now");`
- `$dt = strtotime("10 September 2000");`
- `$dt = strtotime("+1 day");`
- `$dt = strtotime("+1 week");`
- `$dt = strtotime("+1 week 2 days 4 hours");`
- `$dt = strtotime("next Thursday");`
- `$dt = strtotime("last Monday");`

Языковые конструкции

- die и exit
- echo и print
- isset и unset
- include и include_once
- require и require_once
- empty
- eval
- list
- return

Константы и псевдоконстанты

- `__LINE__`
- `__FILE__`
- `__FUNCTION__`
- `__DIR__` (PHP 5.3)

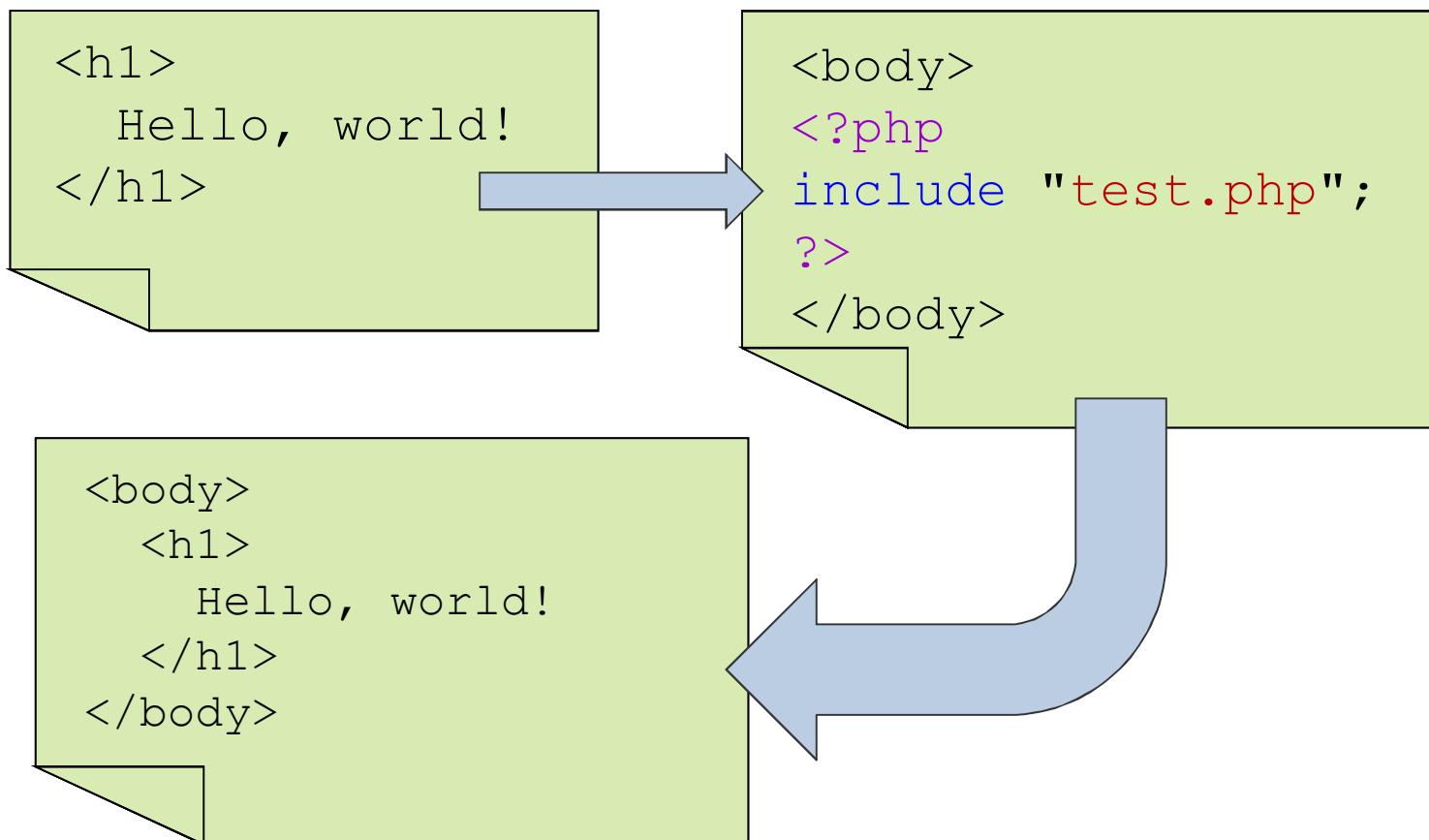
- `E_ALL`
- `PHP_VERSION`
- `PHP_OS`
- `M_PI`

- `get_defined_constants(true)`

Суперглобальные переменные

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

Принцип подключения файлов



Функции эмуляции SSI

- Подключение файла
 - `include`
 - В случае ошибки генерирует предупреждение E_WARNING и продолжает исполнение кода
 - `require`
 - В случае ошибки генерирует ошибку E_COMPILE_ERROR и останавливает исполнение кода
- Однократное подключение файла
 - `include_once`
 - В случае ошибки, действия аналогичны `include`
 - `required_once`
 - В случае ошибки, действия аналогичны `require`

Стандартный пример include

- vars.php

- <?php
 \$color = 'green';
 \$fruit = 'apple';
 ?>

- test.php

- <?php
 echo "A \$color \$fruit"; // A
 include 'vars.php';
 echo "A \$color \$fruit"; // A green apple
 ?>

include внутри функции

- vars.php
 - <?php
 - \$color = 'green';
 - \$fruit = 'apple';
 - ?>
 - test.php
 - function foo(){
 - global \$color;
 - include 'vars.php';
 - echo "A \$color \$fruit";
 - }
 - foo(); // A green apple
 - echo "A \$color \$fruit"; // A green

include с возвратом значения

- return.php

- <?php
 \$var = 'PHP';
 return \$var;
 ?>

- noreturn.php

- <?php
 \$var = 'PHP';
 ?>

- testreturns.php

- <?php
 \$foo = include 'return.php';
 echo \$foo; // Выводит 'PHP'
 \$bar = include 'noreturn.php';
 echo \$bar; // Выводит 1
 ?>

Лабораторная работа 5

Конструирование сайта

Упражнение 1: Создание подключаемых файлов

- В текстовом редакторе создайте новый файл
- Перенесите ([**Ctrl**] + [X]) в файл php-блок с функцией **drawTable()** из файла **table.php**
- Перенесите ([**Ctrl**] + [X]) в файл функцию **drawMenu()** из файла **index.php**
- Сохраните файл под именем **lib.inc.php** и сохраните файлы **index.php** и **table.php**
- В текстовом редакторе создайте новый файл и создайте в нем php-блок
- Перенесите ([**Ctrl**] + [X]) в файл весь php-код из самого верхнего php-блока файла **index.php**
- Сохраните файл под именем **data.inc.php** и сохраните файл **index.php**
- В текстовом редакторе создайте новый файл
- Перенесите ([**Ctrl**] + [X]) в файл всё, что находится внутри блока **<!-- Верхняя часть страницы -->** из файла **index.php**
- Сохраните файл под именем **top.inc.php** и сохраните файл **index.php**
- В текстовом редакторе создайте новый файл
- Перенесите ([**Ctrl**] + [X]) в файл всё, что находится внутри блока **<!-- Навигация -->** из файла **index.php**
- Сохраните файл под именем **menu.inc.php** и сохраните файл **index.php**
- В текстовом редакторе создайте новый файл
- Перенесите ([**Ctrl**] + [X]) в файл всё, что находится внутри блока **<!-- Нижняя часть страницы -->** из файла **index.php**
- Сохраните файл под именем **bottom.inc.php** и сохраните файл **index.php**
- В текстовом редакторе создайте новый файл
- Перенесите ([**Ctrl**] + [X]) в файл всё, что находится внутри блока **<!-- Область основного контента -->** из файла **index.php**
- Сохраните файл под именем **index.inc.php** и сохраните файл **index.php**

Упражнение 2: Подключение файлов

- В текстовом редакторе откройте (если не открыт) файл **index.php**

- В самом верху файла подключите файлы **lib.inc.php** и **data.inc.php**
- В блоке **<!-- Верхняя часть страницы -->** подключите файл **top.inc.php**
- В блоке **<!-- Навигация -->** подключите файл **menu.inc.php**
- В блоке **<!-- Область основного контента -->** подключите файл **index.inc.php**
- В блоке **<!-- Нижняя часть страницы -->** подключите файл **bottom.inc.php**
- Сохраните файл **index.php**
- Посмотрите результат в браузере

Выводы

- Документация PHP
- Встроенные функции
 - Математические функции
 - Функции для работы с переменными
 - Функции обработки строк
 - Функции для работы с массивами
 - Функции даты и времени
- Встроенные константы
- Суперглобальные переменные
- Функции эмуляции SSI

PHP Уровень 1. Работа с HTTP: формы

Игорь Борисов
<http://igor-borisov.ru>

Темы модуля

- HTTP/1.1
 - Заголовки запроса и ответа
 - Статус сервера
- Доступ к заголовкам запроса –
переменные окружения сервера
- Работа с веб-формами
 - Методы GET и POST
 - Различие методов
- Проверка передаваемых значений

HTTP – HyperText Transfer Protocol

- Запрос клиента

- Метод доступа
- URI
- Версия протокола
- Мета информация (заголовки запроса)

- Ответ сервера

- Версия протокола
- Статус ответа
- Описание статуса ответа
- Мета информация (заголовки ответа)

HTTP: пример

```
GET /folder/index.html HTTP/1.1 ↵
Host: www.specialist.ru ↵
Accept: */* ↵
Accept-Language: ru ↵
Referer: http://yandex.ru/yandsearch?text=Rehc ↵
User-Agent: Mozilla 4.0 (compatible; MSIE 6.1,...) ↵
↵
```

```
HTTP/1.1 200 OK ↵
Server: Microsoft IIS 6 ↵
Content-Type: text/html ↵
Content-Length: 16345 ↵
Last-Modified: Sun, 03 Jul 2005 18:00:00 GMT ↵
↵
<html>...
</html>
```

Передача форм

```
<form action="..." method="...">  
    Логин:  
    <input name="login"  
           type="text">  
    Пароль:  
    <input name="pwd"  
           type="password">  
    <input type="submit">  
</form>
```

- Методы
 - GET
 - POST

пасспорт

[Зарегистрироваться](#)

Логин:

Пароль:

запомнить на две недели

[Войти](#) [Закрыть](#)

[Забыли пароль?](#)

Метод GET

GET /action.php?login=Vasya&pwd=Parol HTTP/1.1

↙

Host: www.specialist.ru ↙

Accept: */* ↙

Accept-Language: ru ↙

Referer: http://yandex.ru/yandsearch?text=Rehc

↙

User-Agent: Mozilla 4.0 (compatible; MSIE

6.1,...) ↙

↙

Метод POST

```
POST /action.php HTTP/1.1 ↵
Host: www.specialist.ru ↵
Accept: */* ↵
Accept-Language: ru ↵
Referer: http://yandex.ru/yandsearch?text=Rehc
↵
User-Agent: Mozilla 4.0 (compatible; MSIE
6.1,...) ↵
Content-Type: application/x-www-form-
urlencoded ↵
Content-Length: 20 ↵
↵
login=Vasya&pwd=Parol ↵
↵
```

Приём данных

- Для данных, переданных в строке запроса
 - `$_GET['login'];`
 - `$_GET['pwd'];`
- Для данных, переданных в теле запроса
 - `$_POST['login'];`
 - `$_POST['pwd'];`
- Вариант
 - `$_REQUEST['login'];`
 - `$_REQUEST['pwd'];`
 - См. директиву `php.ini variables_order`

Обработка данных

- Никогда не включайте директиву php.ini **register_globals**
- Никогда не используйте массивы **\$_GET**, **\$_POST** и **\$_REQUEST** без предварительной обработки!
- Обработка данных
 - `$name = trim(strip_tags($_POST['name']));`
 - `$age = (int)$_POST['age'];`
 - `$message = htmlspecialchars($_POST['message']);`
- Использование данных
 - Здравствуйте, `<?php echo $name; ?>`.
 - Вам `<?php echo $age; ?>` лет.
 - Вы написали: `<?php echo $message; ?>`.

Лабораторная работа 6

Передача параметров на сервер

Упражнение 1: Создание подключаемых файлов

- В текстовом редакторе откройте файл **about.php**
- Удалите всё, кроме содержимого блока **<!-- Область основного контента -->**
- Сохраните файл **about.php**
- В текстовом редакторе откройте файл **contact.php**
- Удалите всё, кроме содержимого блока **<!-- Область основного контента -->**
- Сохраните файл **contact.php**
- В текстовом редакторе откройте файл **calc.php**
- Удалите всё, кроме содержимого блока **<!-- Область основного контента -->**
- Сохраните файл **calc.php**

Упражнение 2: Изменение меню

- В текстовом редакторе откройте файл **data.inc.php**
- Необходимо изменить значения во всех элементах **href** массива **\$leftMenu** на **index.php** и добавить параметр (например **id**), который будет передавать методом **GET** уникальные значения
 - Измените значения, которые в результате могут выглядеть так:
index.php
index.php?id=about
index.php?id=contact
index.php?id=table
index.php?id=calc
- Сохраните файл **data.inc.php**

Упражнение 3: Приём данных от пользователя

- В текстовом редакторе откройте файл **index.php**
- В верхней части файла перед закрывающим тэгом **?>** напишите:
// Инициализация заголовков страницы
\$title = 'Сайт нашей школы';
\$header = "\$welcome, Гость!";
\$id = strtolower(strip_tags(trim(\$_GET['id'])));

```

switch($id){
    case 'about':
        $title = 'О сайте';
        $header = 'О нашем сайте';
        break;
    case 'contact':
        $title = 'Контакты';
        $header = 'Обратная связь';
        break;
    case 'table':
        $title = 'Таблица умножения';
        $header = 'Таблица умножения';
        break;
    case 'calc':
        $title = 'Он-лайн калькулятор';
        $header = 'Калькулятор';
        break;
}

```

- Между тэгами <title></title> напишите:
`<?php echo $title?>`
- В блоке <!-- Заголовок --> между тэгами <h1></h1> напишите:
`<?php echo $header?>`
- В блоке <!-- Область основного контента --> удалите всё содержимое и напишите:
`<?php
switch($id){
 case 'about': include 'about.php'; break;
 case 'contact': include 'contact.php'; break;
 case 'table': include 'table.php'; break;
 case 'calc': include 'calc.php'; break;
 default: include 'index.inc.php';
}
?>`
- Сохраните файл index.php
- Посмотрите результат в браузере. Обратите внимание, что файл table.php пока не готов к использованию

Упражнение 4: Допиливаем таблицу умножения

- В текстовом редакторе откройте файл table.php
- Удалите всё, кроме блока <!-- Область основного контента -->
- В верхней части файла напишите:
`<?php
if($_SERVER['REQUEST_METHOD'] == 'POST'){
 $cols = abs((int) $_POST['cols']);`

```
$rows = abs((int) $_POST['rows']);
$color = trim(strip_tags($_POST['color']));
}
$cols = ($cols) ? $cols : 10;
$rows = ($rows) ? $rows : 10;
$color = ($color) ? $color : 'yellow';
?>
```

- В блоке <!-- Таблица --> исправьте вызов функции на:
`drawTable($cols, $rows, $color);`
- В значении атрибута **action** тэга <form> напишите:
`<?= $_SERVER['REQUEST_URI']?>`
- Добавьте в тэг <form> атрибут **method** со значением **POST**
- Сделайте так, чтобы введенные значения оставались в текстовых полях формы после перезагрузки страницы
- Сохраните файл **table.php**
- Посмотрите результат в браузере

Обработка ошибок

- Функция перехвата ошибок
 - `function myError($errno, $errstr, $errfile, $errline){}`
- Установка перехватчика
 - `set_error_handler("myError");`
- Отлавливаем ошибки
 - `if ($error)
trigger_error("Что-то случилось", E_USER_ERROR);`
- Пользовательские ошибки
 - `E_USER_WARNING`
 - `E_USER_ERROR`
 - `E_USER_NOTICE`
- `error_log("Ошибка!\n", 3, "error.log");`

Выводы

- HTTP/1.1
 - Заголовки запроса и ответа
 - Статус сервера
- Доступ к заголовкам запроса – переменные окружения сервера
- Работа с веб-формами
 - Методы GET и POST
 - Различие методов
- Проверка передаваемых значений
- Обработка ошибок

Практическая работа

Создание он-лайн калькулятора

Упражнение 1: Уяснение задачи

- В текстовом редакторе откройте файл **calc.php**
- Калькулятор принимает два числа и производит над ними математические действия
- Математические действия зависят от передаваемого оператора. При том что:
 - Калькулятор оперирует только целыми числами (тип `integer`)
 - Калькулятор понимает 4 действия: сложение, вычитание, умножение и деление
- Результат выводится в виде строки. Например, при переданных данных **2, 3** и **+**, можно вывести:
Результат 2 + 3 = 5

Упражнение 1: Прием данных из формы

- Убедитесь, что передача данных передаваемых веб-формой осуществлена методом **POST**
- Примите данные переданные веб-формой
- Убедитесь, что все данные пришли со значениями
- Отфильтруйте пришедшие данные

Упражнение 2: Получение необходимого результата

- Используя управляющую конструкцию **switch**, производите различные математические действия в зависимости от оператора
- В случае деления, проверьте делитель на равенство с нулем (на ноль делить нельзя)
- Помните, что калькулятор выполняет только 4 действия, а пользователь может передать неверный оператор - это надо отследить и уведомить пользователя об ошибке

Упражнение 3: Вывод результата и проверка работы

- Выведите результат вычислений перед html-кодом отрисовки веб-формы
- Сохраните файл **calc.php**
- Запустите калькулятор в браузере и проверьте его работу