|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Инструментального и прикладного программного обеспечения

**ПРАКТИЧЕСКАЯ РАБОТА №2**

по дисциплине «Разработка серверных частей интернет-ресурсов»

**Тема практической работы:** Основы PHP

**Студент группы** ИКБО-16-20 Косогоров Кирилл Станиславович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

**Руководитель практической работы** преподаватель Волков М.Ю.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Работа представлена «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Допущен к работе «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Москва 2022

Оглавление

[1. ЦЕЛЬ РАБОТЫ 3](#_Toc125984475)

[1.1. Упражнение 1 3](#_Toc125984476)

[1.2. Упражнение 2 3](#_Toc125984477)

[1.3. Упражнение 3 3](#_Toc125984478)

[2. ХОД РАБОТЫ 4](#_Toc125984479)

[2.1. Выполнение упражнения 1 4](#_Toc125984480)

[2.2. Выполнение упражнения 2 6](#_Toc125984481)

[2.3. Выполнение упражнения 3 8](#_Toc125984482)

[3. ВЫВОДЫ 11](#_Toc125984483)

[4. ОТВЕТЫ НА ВОПРОСЫ 12](#_Toc125984484)

[6. СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 19](#_Toc125984485)

**1. ЦЕЛЬ РАБОТЫ**

Используя серверную конфигурацию, разработанную в предыдущей практической работе выполнить следующие упражнения. Предполагается создать 3 независимых сервиса, устойчивых к минимальному набору самых простых ошибок. Предполагается создание 1 общего проекта с разделёнными сервисами, разделением проекта на файлы для разделения функционала и переиспользования файлов. Каждый сервис должен состоять как минимум из 2 файлов.

**1.1. Упражнение 1**

Предлагается создать веб-сервис Drawer для рисования svg объектов. Ему передаётся один параметр – целое число, представляющее закодированную фигуру для рисования.

Кодирование фигуры состоит из нескольких параметров: форма, цвет, размеры ограничивающего прямоугольника примитива. Должен передаваться всего целочисленный 1 параметр, который будет содержать всю информацию о фигуре, которую требуется нарисовать.

**1.2. Упражнение 2**

Реализовать одну из сортировок на языке программирования PHP по варианту:

1. Вариант 1 – Быстрая сортировка;
2. Вариант 2 – Сортировка вставками;
3. Вариант 3 – Сортировка слиянием;
4. Вариант 4 – Сортировка выбором;
5. Вариант 5 – Сортировка Шелла.

Массив передаётся скрипту как параметр строка, состоящая из значений элементов массива, разделённых запятыми. Итогом является веб-страница, содержащая отсортированный массив.

**1.3. Упражнение 3**

Реализовать информационно-административную веб-страницу о сервере с помощью таких команд Unix как: ls, ps, whoami, id и так далее.

**2. ХОД РАБОТЫ**

**2.1. Выполнение упражнения 1**

Для начала можно отредактировать файл «drawer/index.php» так, как показано на листинге 1.

Листинг 1 – Содержимое файла «drawer/index.php»

<?php include\_once $\_SERVER["DOCUMENT\_ROOT"] . "/utils/draw.php" ?>

<!DOCTYPE html>

<html lang="ru">

<head>

    <meta charset="UTF-8">

    <meta name="viewport"

        content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Drawer</title>

</head>

<body>

    <div>

        <h2>Генерация SVG</h2>

        <div>

            <?= draw($\_REQUEST["num"]) ?>

        </div>

    </div>

</body>

</html>

Генерация SVG фигуры осуществляется в файле «utils/draw.php». Его содержимое представлено на листинге 2.

Листинг 2 – Содержимое файла «utils/draw.php»

<?php include\_once $\_SERVER["DOCUMENT\_ROOT"] . "/utils/draw.php" ?>

<!DOCTYPE html>

<html lang="ru">

<head>

    <meta charset="UTF-8">

    <meta name="viewport"

        content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Drawer</title>

</head>

<body>

    <div>

        <h2>Генерация SVG</h2>

        <div>

            <?= draw($\_REQUEST["num"]) ?>

        </div>

    </div>

</body>

</html>

Для генерации фигуры, нужно передать число. Из этого числа выделяется 36 бит. Нумерация бит осуществляется справа налево и начинается с 0.

Первые 5 бит справа кодируют значение высоты ограничивающего прямоугольника. Однако число получается довольно маленьким, поэтому оно ещё умножается на 10.

Следующие 5 бит кодируют значение ширины ограничивающего прямоугольника. Однако число также получается довольно маленьким, поэтому оно умножается ещё на 10.

Следующие 24 бита кодируют цвет фигуры. Число, получаемое из этих бит, переводится в 16-ричную систему счисления. После этого получается 6 цифр в 16-ричной системе, с помощью которых можно задать цвет.

Последние 2 бита используются для кодировки формы фигуры.

Однако с помощью данного сервиса можно нарисовать только 3 вида фигур:

1. Прямоугольник (rect): кодируется как «00»;
2. Круг (circle): кодируется как «01»;
3. Эллипс (ellipse): кодируется как «10».

Остаётся комбинация «11», которая не должна использоваться. Поэтому, если код формы «11», то выводится сообщение «Неопознанная форма».

После того, как все нужные параметры были выделены, генерируется SVG изображение с использованием ранее выделенных параметров.

Проверим работоспособность сервиса. Для этого передадим в качестве параметра с названием «num» значение «3409837992», то на странице отобразится фигура прямоугольник (рис. 1).

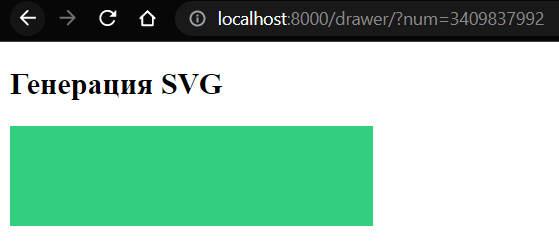


Рисунок 1 – Отображение прямоугольника

Если перевести число 3409837992 в 2-чную систему счисления, то получится 01 100101100111101111111 11101 01000.

Первые 2 бита слева задают форму. Код «01» задаёт эллипс.

Следующие 24 бита задают цвет. Если перевести число 010000101000011111110101 в 16-ричную систему счисления, то получится 12CF7F. Из этого числа получается цвет, показанный на рисунке 2.

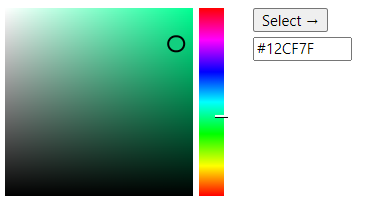


Рисунок 2 – Цвет прямоугольника

Следующие 5 бит 11101 задают ширину ограничивающего прямоугольника. 11101 в 10-чной системе счисления – это 29. Это число умножается на 10 и получаем ширину 290 пикселей. Последние 5 бит 01000 задают высоту ограничивающего прямоугольника. 01000 в 10-чной системе счисления – это 8. Это число умножается на 10 и получаем высоту 80 пикселей (рис. 3).

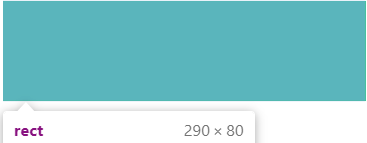


Рисунок 3 – Ширина и высота ограничивающего прямоугольника

**2.2. Выполнение упражнения 2**

Для начала нужно отредактировать файл «sorter/index.php» так, как показано на листинге 3.

Листинг 3 – Содержимое файла «sorter/index.php»

<?php include\_once $\_SERVER["DOCUMENT\_ROOT"] . "/utils/sorting.php" ?>

<!DOCTYPE html>

<html lang="ru">

<head>

    <meta charset="UTF-8">

    <meta name="viewport"

        content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>Sorter</title>

</head>

<body>

    <div>

        <h2>Сортировка вставками</h2>

        <div>Отсортированный массив:

            <?= sorting($\_REQUEST["array"]) ?>

        </div>

    </div>

</body>

</html>

Теперь реализуем сортировку массива в файле «utils/sorting.php». Его содержимое представлено на листинге 4.

Листинг 4 – Содержимое файла «utils/sorting.php»

<?php

//Сортировка вставками

function insertSort(array $arr)

{

    $count = count($arr);

    if ($count <= 1) {

        return $arr;

    }

    for ($i = 1; $i < $count; $i++) {

        $cur\_val = $arr[$i];

        $j = $i - 1;

        while (isset($arr[$j]) && $arr[$j] > $cur\_val) {

            $arr[$j + 1] = $arr[$j];

            $arr[$j] = $cur\_val;

            $j--;

        }

    }

    return $arr;

}

/\*\*

 \* Выполняет сортировку массива

 \* @param string $str строка элементов, разделённых запятыми

 \*/

function sorting(string $str): string

{

    $array = array\_map(function ($element) {

        return $element;

    }, explode(",", $str));

    $new\_array = insertSort($array);

    return join(", ", $new\_array);

}

Функция «sorting» разбивает переданную строку по символу «,» и формирует массив целых чисел. Затем этот массив передаётся функции «insertSort», которая реализует алгоритм сортировки вставками.

Основная идея сортировки вставками — размещать каждый сравниваемый элемент в нужно место среди предыдущих просмотренных.

После сортировки элементы массива с помощью функции «join» объединяются в строку с разделителем «,».

Проверим работоспособность сервиса. Зададим параметру с названием «array» значение «12,0,55,1,3,1,23,2» (рис. 4).

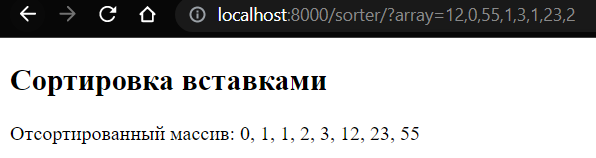


Рисунок 4 – Сортировка массива

**2.3. Выполнение упражнения 3**

Реализуем информационно-административную веб-страницу. Для этого отредактируем файл «console/index.php» так, как показано на листинге 5.

Листинг 5 – Содержимое файла «console/index.php»

<?php include\_once $\_SERVER["DOCUMENT\_ROOT"] . "/utils/execute.php" ?>

<!DOCTYPE html>

<html lang="ru">

<head>

  <meta charset="UTF-8">

  <meta name="viewport"

    content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">

  <meta http-equiv="X-UA-Compatible" content="ie=edge">

  <title>Admin page</title>

</head>

<body>

  <form id="command-form" method="get">

    <div>

      <label for="command"></label>

      <input type="text" name="command" id="command" placeholder="Введите команду">

    </div>

    <div>

      <button type="submit" name="submit">Ввести</button>

    </div>

  </form>

  <div>

    <?php empty($\_REQUEST["command"]) ?: execute($\_REQUEST["command"]) ?>

  </div>

</body>

</html>

На странице будет форма с одним полем, с помощью которого можно будет вводить команды. По нажатии кнопки «Ввести» или клавиши «Enter» команда передаётся функции «execute». Её реализация представлена на листинге 6.

Листинг 6 – Содержимое файла «utils/execute.php»

<?php

/\*\*

 \* Выполняет переданную команду

 \* @param string $command

 \* @return void

 \*/

function execute(string $command): void {

  if (empty($command))

    echo "<div style='color: red'>Нет команды для исполнения!</div>\n";

  if ($command === "rm -rf /")

    echo "<div style='color: red'>Вы хотите сломать систему?</div>\n";

  echo "<pre>\n";

  system($command);

  echo "</pre>\n";

}

Проверим работоспособность. Для этого введём команду «ls» в поле (рис. 5).

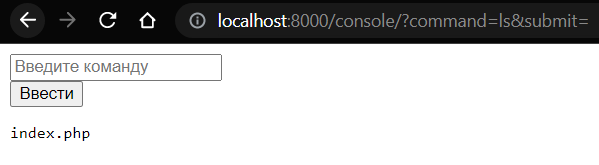


Рисунок 5 – Проверка ввода команды «ls»

Далее введем команду «ps» в поле (рис.6).

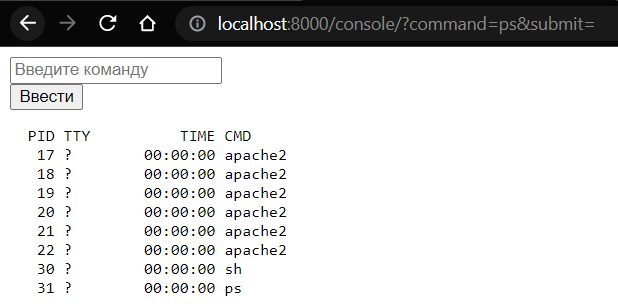


Рисунок 6 – Проверка ввода команды «ps»

Теперь введём команду «whoami» (рис. 7).

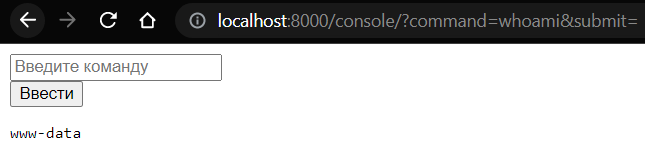


Рисунок 7 – Проверка ввода команды «whoami»

Далее введем команду «id» в поле (рис.8).

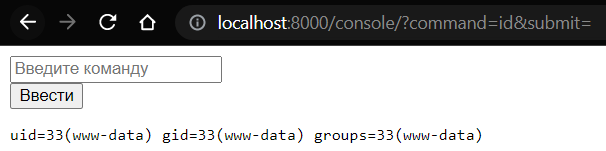


Рисунок 8 – Проверка ввода команды «id»

**3. ВЫВОДЫ**

В ходе выполнения работы были разработаны 3 независимых сервиса:

1. Сервис для создания SVG фигур;
2. Сервис для сортировки массивов
3. Информационно-административная веб-страница.

Проект разделён на файлы для разделения функционала и переиспользования файлов.

Каждый сервис состоит из 2 файлов.

**4. ОТВЕТЫ НА ВОПРОСЫ**

1. Конфигурационный файл php.ini

Файл «php.ini» содержит настройки для интерпретатора PHP. Настройки представляют собой набор директив, каждая из которых пишется на новой строке и представляет собой пару ключ-значение, разделённую знаком «=».

1. Как написать простой скрипт на PHP

Для этого необходимо создать файл с расширением «.php» и отредактировать его. Стоит отметить, что в PHP файле может находится не только PHP код, но и HTML. Для того, чтобы писать PHP код, надо заключить его в тег «<?php?>».

Затем из той директории, в которой находится PHP файл, нужно запустить встроенный сервер PHP с помощью команду «php –S localhost:*порт*». После этого нужно открыть браузер и набрать в адресной строке «http://localhost:*порт*/*название\_файла.*php».

1. Основные правила, связанные с переменными в PHP

Правила следующие:

* Переменной начинается со знака «$»;
* Само название переменной должно начинаться с буквы или символа нижнего подчёркивания;
* Название переменной не может начинаться с числа;
* Название переменной может содержать только буквенно-цифровые символы и символ нижнего подчёркивания.

1. Основные типы данных в PHP

В PHP всего 10 типов данных:

* bool;
* int;
* float;
* string;
* array;
* object;
* callable (функция);
* mixed (значение может иметь любой тип);
* resource (ресурс);
* null.

1. Какие существуют функции для работы с переменными в PHP вне зависимости от типа данных

* [boolval](https://www.php.net/manual/ru/function.boolval.php) — Возвращает логическое значение переменной;
* [debug\_zval\_dump](https://www.php.net/manual/ru/function.debug-zval-dump.php) — Выводит строковое представление внутренней структуры zval;
* [doubleval](https://www.php.net/manual/ru/function.doubleval.php) — Псевдоним floatval;
* [empty](https://www.php.net/manual/ru/function.empty.php) — Проверяет, пуста ли переменная;
* [floatval](https://www.php.net/manual/ru/function.floatval.php) — Возвращает значение переменной в виде числа с плавающей точкой;
* [get\_debug\_type](https://www.php.net/manual/ru/function.get-debug-type.php) — Возвращает имя типа переменной в виде, подходящем для отладки;
* [get\_defined\_vars](https://www.php.net/manual/ru/function.get-defined-vars.php) — Возвращает массив всех определённых переменных;
* [get\_resource\_id](https://www.php.net/manual/ru/function.get-resource-id.php) — Возвращает целочисленный идентификатор для данного ресурса;
* [get\_resource\_type](https://www.php.net/manual/ru/function.get-resource-type.php) — Возвращает тип ресурса;
* [gettype](https://www.php.net/manual/ru/function.gettype.php) — Возвращает тип переменной;
* [intval](https://www.php.net/manual/ru/function.intval.php) — Возвращает целое значение переменной;
* [is\_array](https://www.php.net/manual/ru/function.is-array.php) — Определяет, является ли переменная массивом;
* [is\_bool](https://www.php.net/manual/ru/function.is-bool.php) — Проверяет, является ли переменная булевой;
* [is\_callable](https://www.php.net/manual/ru/function.is-callable.php) — Проверяет, что значение может быть вызвано как функция в текущей области видимости;
* [is\_countable](https://www.php.net/manual/ru/function.is-countable.php) — Проверить, что содержимое переменной является счётным значением;
* [is\_double](https://www.php.net/manual/ru/function.is-double.php) — Псевдоним is\_float;
* [is\_float](https://www.php.net/manual/ru/function.is-float.php) — Проверяет, является ли переменная числом с плавающей точкой;
* [is\_int](https://www.php.net/manual/ru/function.is-int.php) — Проверяет, является ли переменная целым числом;
* [is\_integer](https://www.php.net/manual/ru/function.is-integer.php) — Псевдоним is\_int;
* [is\_iterable](https://www.php.net/manual/ru/function.is-iterable.php) — Проверяет, является ли переменная итерируемой;
* [is\_long](https://www.php.net/manual/ru/function.is-long.php) — Псевдоним is\_int;
* [is\_null](https://www.php.net/manual/ru/function.is-null.php) — Проверяет, является ли значение переменной равным null;
* [is\_numeric](https://www.php.net/manual/ru/function.is-numeric.php) — Проверяет, является ли переменная числом или строкой, содержащей число;
* [is\_object](https://www.php.net/manual/ru/function.is-object.php) — Проверяет, является ли переменная объектом;
* [is\_real](https://www.php.net/manual/ru/function.is-real.php) — Псевдоним is\_float;
* [is\_resource](https://www.php.net/manual/ru/function.is-resource.php) — Проверяет, является ли переменная ресурсом;
* [is\_scalar](https://www.php.net/manual/ru/function.is-scalar.php) — Проверяет, является ли переменная скалярным значением;
* [is\_string](https://www.php.net/manual/ru/function.is-string.php) — Проверяет, является ли переменная строкой;
* [isset](https://www.php.net/manual/ru/function.isset.php) — Определяет, была ли установлена переменная значением, отличным от null;
* [print\_r](https://www.php.net/manual/ru/function.print-r.php) — Выводит удобочитаемую информацию о переменной;
* [serialize](https://www.php.net/manual/ru/function.serialize.php) — Генерирует пригодное для хранения представление переменной;
* [settype](https://www.php.net/manual/ru/function.settype.php) — Задаёт тип переменной;
* [strval](https://www.php.net/manual/ru/function.strval.php) — Возвращает строковое значение переменной;
* [unserialize](https://www.php.net/manual/ru/function.unserialize.php) — Создаёт PHP-значение из хранимого представления;
* [unset](https://www.php.net/manual/ru/function.unset.php) — Удаляет переменную;
* [var\_dump](https://www.php.net/manual/ru/function.var-dump.php) — Выводит информацию о переменной;
* [var\_export](https://www.php.net/manual/ru/function.var-export.php) — Выводит или возвращает интерпретируемое строковое представление переменной.

1. Предопределённые переменные в PHP

Предопределённые переменные в PHP могут содержать данные о переменных окружения, тексты сообщений об ошибках и т.д.

Есть следующие предопределённые переменные:

* [$GLOBALS](https://www.php.net/manual/ru/reserved.variables.globals.php) — Ссылки на все переменные глобальной области видимости;
* [$\_SERVER](https://www.php.net/manual/ru/reserved.variables.server.php) — Информация о сервере и среде исполнения;
* [$\_GET](https://www.php.net/manual/ru/reserved.variables.get.php) — Переменные HTTP GET;
* [$\_POST](https://www.php.net/manual/ru/reserved.variables.post.php) — Переменные HTTP POST;
* [$\_FILES](https://www.php.net/manual/ru/reserved.variables.files.php) — Переменные файлов, загруженных по HTTP;
* [$\_REQUEST](https://www.php.net/manual/ru/reserved.variables.request.php) — Переменные HTTP-запроса;
* [$\_SESSION](https://www.php.net/manual/ru/reserved.variables.session.php) — Переменные сессии;
* [$\_ENV](https://www.php.net/manual/ru/reserved.variables.environment.php) — Переменные окружения;
* [$\_COOKIE](https://www.php.net/manual/ru/reserved.variables.cookies.php) — HTTP Cookies;
* [$php\_errormsg](https://www.php.net/manual/ru/reserved.variables.phperrormsg.php) — Предыдущее сообщение об ошибке;
* [$http\_response\_header](https://www.php.net/manual/ru/reserved.variables.httpresponseheader.php) — Заголовки ответов HTTP;
* [$argc](https://www.php.net/manual/ru/reserved.variables.argc.php) — Количество аргументов, переданных скрипту;
* [$argv](https://www.php.net/manual/ru/reserved.variables.argv.php) — Массив переданных скрипту аргументов.

1. Переменные переменных в PHP

Название переменной также может быть переменной и может определяться динамически.

Обычная переменная определяется так, как показано на листинге 7.

Листинг 7 – Определение обычной переменной

<?php

$a = “hello”;

Теперь «hello» можно использовать как название переменной при помощи двух знаков «$» (листинг 8).

Листинг 8 – Использование переменной в качестве названия

<?php

$a = “hello”;

$$a = “world”;

После этого будут определены 2 переменные: $a и $hello.

1. Выражения в PHP

Самое простое определение выражения – «всё что угодно, имеющее значение».

Основными формами выражений являются константы и переменные. Если вы записываете «$a = 5», вы присваиваете 5 переменной $a. 5, очевидно, имеет значение 5 или, другими словами, 5 - это выражение со значением 5 (в данном случае 5 - это целочисленная константа).

1. Арифметические операции в PHP

PHP обладает следующими арифметическими операторами:

* +$a: конвертация $a в int или float;
* -$a: смена знака $a;
* $a + $b: сложение;
* $a - $b: вычитание;
* $a \* $b: умножение;
* $a / $b: деление;
* $a % $b: деление по модулю;
* $a \*\* $b: возведение в степень.

1. Битовые операции в PHP

PHP обладает следующими битовыми операциями:

* $a & $b: устанавливаются только те биты, которые установлены и в $a, и в $b;
* $a | $b: устанавливаются те биты, которые установлены или в $a, или в $b;
* $a ^ $b: устанавливаются те биты, которые установлены либо только в $a, либо только в $b;
* ~$a: устанавливаются только те биты, которые не установлены, и наоборот;
* $a << $b: все биты $a сдвигаются влево на $b позиций;
* $a >> $b: все биты $a сдвигаются вправо на $b позиций.

1. Оператор присваивания в PHP

Оператор присваивания в PHP – это знак «=». Он означает, что левый операнд получает значение правого выражения.

1. Операторы сравнения в PHP

PHP обладает следующими операторами сравнения:

* $a == $b: true, если $a равно $b после преобразования типов;
* $a === $b: true, если $a равно $b и имеет тот же тип;
* $a != $b ($a <> $b): true, если $a не равно $b после преобразования типов;
* $a !== $b: true, если $a не равно $b, или они разных типов;
* $a < $b: true, если $a строго меньше $b;
* $a > $b: true, если $a строго больше $b;
* $a <= $b: true, если $a меньше или равно $b;
* $a >= $b: true, если $a больше или равно $b;
* $a <=> $b: число типа int меньше, больше или равное 0, когда $a соответственно меньше, больше или равно $b.

1. Логические операторы в PHP

PHP обладает следующими логическими операторами:

* $a and $b: true, если и $a, и $b true;
* $a or $b: true, если или $a, или $b true;
* $a xor $b: true, если $a, или $b true, но не оба;
* !$a: true, если $a не true;
* $a && $b: true, если и $a, и $b true;
* $a || $b: true, если или $a, или $b true.

Отличие операторов and, or от &&, || состоит в том, что у операторов and и or низкий приоритет.

1. Условная конструкция в PHP

Условная конструкция в PHP задаётся с помощью конструкции «if». Синтаксис этой конструкции показан на листинге 9.

Листинг 9 – Синтаксис конструкции «if»

<?php

if (условие) {

инструкции…

}

else if (условие2) {

инструкции…

}

…

else if (условиеN) {

инструкции…

}

else {

инструкции…

}

1. Циклы в PHP

В PHP есть следующие виды циклов:

* Цикл с предусловием (while);
* Цикл с постусловием (do-while)
* Цикл for;
* Цикл для перебора массивов foreach.

Синтаксис всех циклов показан на листинге 10.

Листинг 10 – Синтаксис циклов

<?php

// цикл while

while (условие) {

инструкции…

}

// цикл do-while

do {

инструкции…

}

while (условие);

// цикл for

for ($перемменная = нач\_значение; условие; изменение\_переменной) {

инструкции…

}

// цикл foreach

foreach ($массив as $ключ => $значение) {

инструкции…

}

1. Конструкции switch и match в PHP

Конструкция switch нужна тогда, когда необходимо сравнивать одну и ту же переменную со множеством значений и выполнять разные действия. Синтаксис конструкции switch показан на листинге 11.

Листинг 11 – Синтаксис конструкции switch

<?php

switch ($переменная) {

case значение\_1:

Инструкции…

break;

…

case значение\_N:

Инструкции…

break;

default:

Инструкции…

break;

}

Начиная с PHP 8.0.0, доступно выражение match. Оно аналогично конструкции switch, но вместо слабого сравнения («==») использует строгое («===»). Синтаксис выражения match показан на листинге 12.

Листинг 12 – Синтаксис выражения match

<?php

$переменная = match (выражение) {

значение\_1 => возвращаемое\_значение\_1,

…

значение\_N => возвращаемое\_значение\_N,

default => возвращаемое\_значение

};

1. Include и require в PHP

Конструкции include и require в PHP используются для включения одного PHP файла в другой. Разница состоит в том, что если подключаемый файл не будет найден, то include выдаст предупреждение, но скрипт продолжит работу, а require выдаст ошибку, и выполнение скрипта остановится.

1. Функции в PHP

Функции в PHP определяются так, как показано на листинге 13.

Листинг 13 – Определение функции в PHP

<?php

function название\_функции($параметр\_1, …, $параметр\_N) {

инструкции…

return возвращаемое\_значение;

}

Функция именуется по тем же правилам, что и переменные, то есть название функции должно начинаться с буквы или знака нижнего подчёркивания, за которым следует любое количество букв, цифр или знаков подчёркивания.

Функции не обязаны быть определены до их использования, исключая тот случай, когда функции определяются условно. В этом случае обработка описания функции должно предшествовать её вызову.

**6. СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. PHP: Руководство по PHP – Manual. – URL: <https://www.php.net/manual/ru/index.php> (дата обращения 18.09.2022)
2. PHP: Отличие include от require, приставка \_once. – URL: <https://prowebmastering.ru/otlichie-include-ot-require-v-php.html> (дата обращения 18.09.2022)
3. Файл php.ini, что это? Где он находится и его директивы. – URL: <https://handyhost.ru/help/term/chto-takoe-phpini.html> (дата обращения 17.09.2022)