|  |  |  |
| --- | --- | --- |
|  |  |  |
| 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-30-20 Патина В.С.

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

(подпись студента)

**Руководитель практической работы** преподаватель Благирев М.М.

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

(подпись руководителя)

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

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

Москва 2022

**СОДЕРЖАНИЕ**

[Постановка задачи 1](#_Toc113622084)

[Выполнение работы 1](#_Toc113622085)

[Ссылка на удаленный репозиторий 8](#_Toc113622086)

[Ответы на вопросы к практической: 9](#_Toc113622087)

[ВЫВОДЫ 9](#_Toc113622088)

[СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ 10](#_Toc113622089)

Постановка задачи

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

Выполнение работы

Создадим рабочую директорию со структурой папок (Рисунок 1).

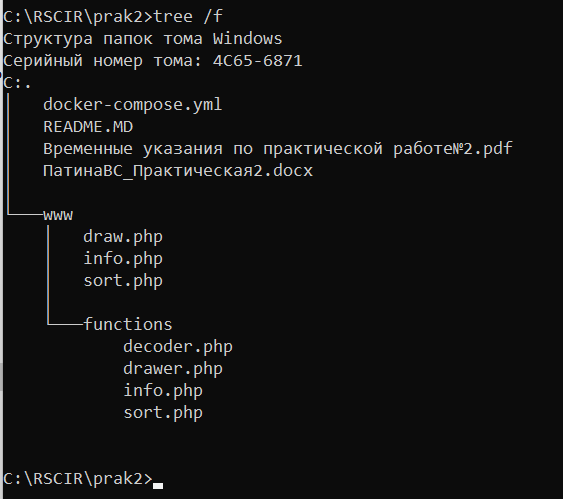


Рисунок 1 - Рабочая директория со структурой папок и файлов

Содержимое файла docker-compose.yml представлено в листинге 1.

Листинг 1 – Содержимое файла docker-compose.yml

version: "3"

services:

  apache:

    image: php:7.4-apache

    ports:

      - 8081:80

    volumes:

      - ./www:/var/www/html

Считываем код фигуры из запроса, расшифровываем его и выводим полученную фигуру на страницу (Листинг 2).

Листинг 2 – Содержимое файла draw.php

<html lang="ru">

<head>

    <title>Рисование фигур</title>

    <style>

        body {

        background-color: #fa8e47;

        }

    </style>

</head>

<body>

    <?php

    $n = htmlspecialchars($\_GET['code']);

    include \_\_DIR\_\_ . '/functions/decoder.php';

    include \_\_DIR\_\_ . '/functions/drawer.php';

    $a = decode($n);

    echo draw($a["shape"], $a["color"], $a["width"], $a["height"]);

    ?>

</body>

</html>

Логика кодировки информации о фигуре описана в листинге 3.

Листинг 3 – Содержимое файла decoder.php

<?php

define('SHAPE', 0b1100000000000000);

define('RED', 0b0011000000000000);

define('GREEN', 0b0000110000000000);

define('COLOR', 85);

define('SIZE', 20);

define('BLUE', 0b0000001100000000);

define('WIDTH', 0b0000000011110000);

define('HEIGHT', 0b0000000000001111);

//Код из 16 бит

//2 бита - фигура

//6 бит - цвет:

//красный, зеленый синий (каждый цвет по два бита)

// 4 бита - ширина

// 4 бита - высота

function getParam($el, $mask, $shift) {

        return $el & $mask >> $shift;

    }

function getShape($code) {

    switch ($code) {

        case 0b00:

           return 'Circle';

        case 0b01:

            return 'Rectangle';

        case 0b10:

            return 'Romb';

        case 0b11:

            return 'Ellipse';

    }

}

function decode($code) {

    $shift = 16;

    $shape = getShape(getParam($code, SHAPE, $shift - 2));

    $red = getParam($code, RED, $shift - 4) \* COLOR;

    $green = getParam($code, GREEN, $shift - 6) \* COLOR;

    $blue = getParam($code, BLUE, $shift - 8) \* COLOR;

    $width = getParam($code, WIDTH, $shift - 12) \* SIZE;

    $height = getParam($code, HEIGHT, $shift - 16) \* SIZE;

    $color = "rgb({$red}, {$green}, {$blue})";

    return ["shape" => $shape, "color" => $color, "width" => $width, "height" => $height];

}

?>

Отрисовка фигуры производится с помощью функции draw() (Листинг 4).

Листинг 4 – Содержимое файла drawer.php

<?php

function draw($shape, $color, $w, $h){

    $openTag = "<svg xmlns='http://www.w3.org/2000/svg' version='1.1' width='500' height='300'>";

    $closeTag = "</svg>";

    $colorTag = "style='fill: {$color}'/>";

    $param = min($h, $w) / 2;

    $r = $param / 2;

    switch ($shape) {

        case "Circle":

            $f = "<circle cx='{$param}' cy='{$param}' r = '{$r}' ";

            break;

        case "Rectangle":

            $f = "<rect x='23' y='23' width='{$w}' height='{$h}' ";

            break;

        case "Romb":

            $hSize2 = round($h / 2);

            $hSize1 = round($w / 2);

            $f = "<polygon points='0,{$hSize2} {$hSize1},{$h} {$w},{$hSize2} {$hSize1},0' ";

            break;

        case "Ellipse":

            $f = "<ellipse cx='{$w}' cy='{$h}' rx='{$w}' ry='{$h}'";

            break;

        default:

             break;

    }

    return $openTag.$f.$colorTag.$closeTag;

}

?>

Работа веб-сервиса по рисованию svg-объектов представлена на рисунках 2, 3, 4.

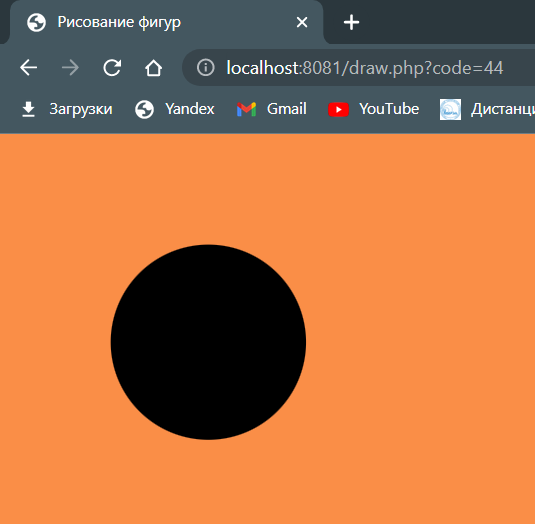


Рисунок 2 – Круг

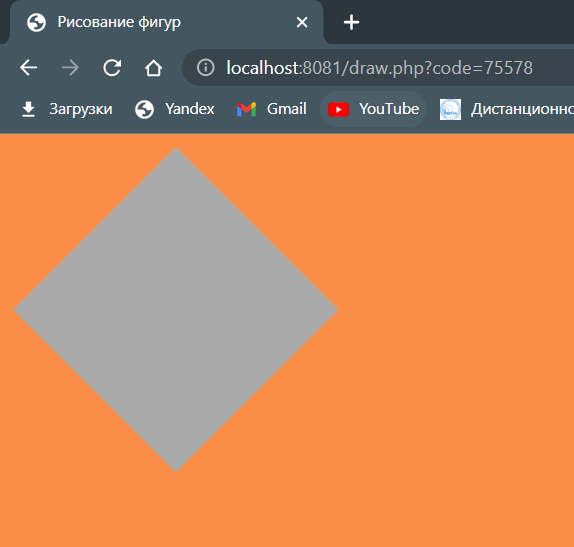


Рисунок 3 - Ромб

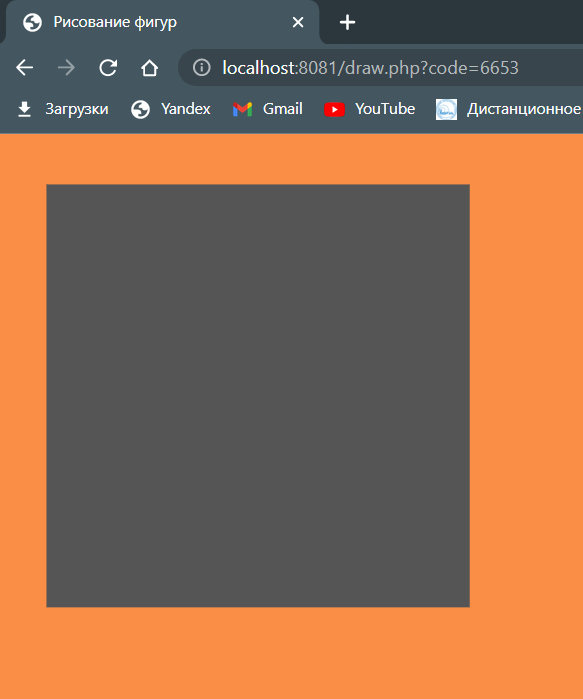


Рисунок 4 - Квадрат

Считываем массив (как строку) из запроса, преобразовываем эту строку в массив и выводим отсортированный массив (Листинг 5).

Листинг 5 - Содержимое файла sort.php

<html lang="ru">

<head>

    <title>Сортировка выбором</title>

</head>

<body>

    <?php

    $arrayString = htmlspecialchars($\_GET["array"]);

    $array = array\_map(null, explode(',', $arrayString));

    include \_\_DIR\_\_ . '/functions/sort.php';

    print\_r(selectionSort($array));

    ?>

</body>

</html>

Реализация сортировки выбором содержится в файле sort.php (Листинг 6).

Листинг 6 - Содержимое файла sort.php в папке functions

<?php

function selectionSort($array){

    $n = count($array);

    for ($i = 0; $i < $n; $i++){

        $min = $i;

        for ($j = $i; $j < $n; $j++) {

            if ($array[$j] < $array[$min]) {

                $min =  $j;

            }

        }

        $tmp = $array[$i];

        $array[$i] = $array[$min];

        $array[$min] = $tmp;

    }

    return $array;

}

?>

В итоге, получаем веб-страницу, содержащую отсортированный массив (Рисунок 5)

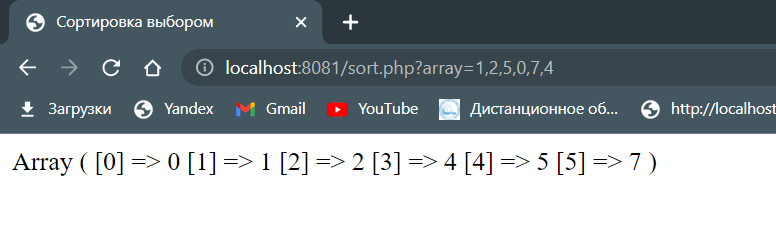


Рисунок 5 – Веб-страница, содержащая отсортированный массив

Считываем команды строкой, преобразовываем строку в массив из команд, выполняем их, печатаем результат выполнения команд на страницу (Листинг 7).

Листинг 7 – Содержимое файла info.php

<html lang="ru">

<head>

    <title>Команды</title>

</head>

<body>

    <?php

    $com = htmlspecialchars($\_GET["commands"]);

    $arrayCommands = explode(',', $com);

    require \_\_DIR\_\_ . '/functions/info.php';

    print\_res($arrayCommands);

    ?>

</body>

</html>

Выполнение и печать результата выполнения команд осуществляется с помощью функции print\_res (Листинг 8).

Листинг 8 – Содержимое файла info.php в папке functions

<?php

    function print\_res($commands) {

        foreach ($commands as $command) {

            print\_r('Результат выполнения команды ' . $command . '<br>');

            exec($command, $result);

            foreach ($result as $i) {

                print\_r($i . '<br>');

            }

        }

    }

    ?>

Информационно-административная веб-страница о сервере (Рисунок 6).

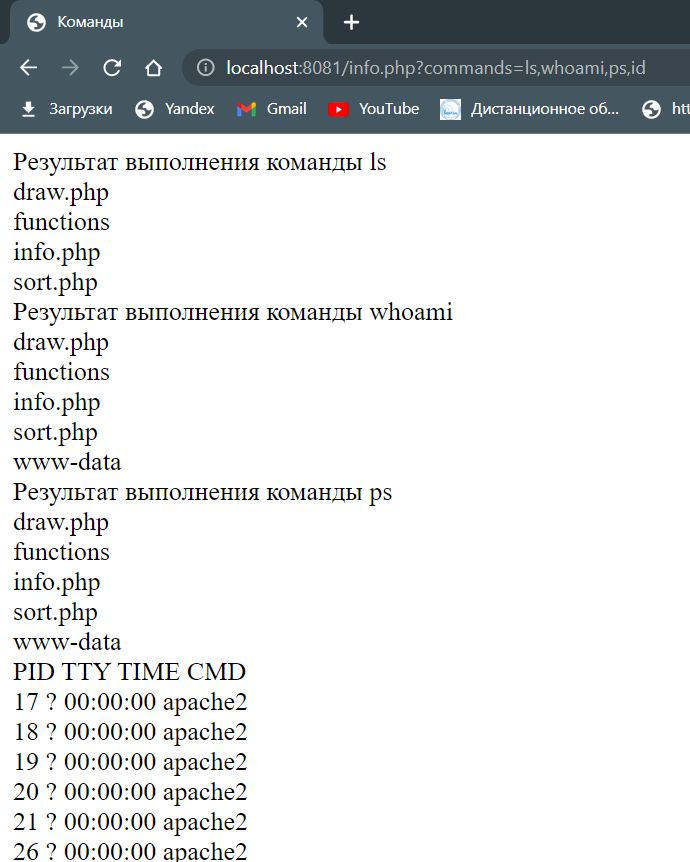


Рисунок 6 – Веб-страница, содержащая результат выполнения команд ls. whoami, id, ps

Ссылка на удаленный репозиторий

https://github.com/devilishgrapefruit/RSCIR/tree/main/prak2

Ответы на вопросы к практической:

**1. Конфигурационный файл php.ini.**

**2. Как написать простой скрипт на php.**

**3. Основные правила, связанные с переменными в php.**

**4. Основные типы данных в php.**

**5. Какие существуют функции для работы с переменными в php вне**

**зависимости от типа данных.**

**6. Предопределенные переменные в php.**

**7. Переменные переменных в php.**

**8. Выражения в php.**

**9. Арифметические операторы в php.**

**10. Битовые операции в php.**

**11.Оператор присваивания в php.**

**12.Операторы сравнения в php.**

**13.Логические операторы в php.**

**14.Условная конструкция в php.**

**15.Циклы в php.**

**16. Конструкции switch и match в php.**

**17. Include и require в php.**

**18. Функции в php.**

ВЫВОДЫ

В процессе выполнения практической работы мы создали свою конфигурацию серверного программного обеспечения, в которой присутствуют веб-сервер, операционная система, язык программирования и база данных. Для проверки работоспособности конфигурации мы сгенерировали тестовую страничку с необходимыми данными.

СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ

1. Установка и настройка PHP URL: <https://www.php.net/manual/ru/install.php> (Дата обращения 06.09.2022)
2. Официальная документация докера: URL: https://docs.docker.com/ (Дата обращения 06.09.2022)
3. Статья о назначении докера простыми словами. URL: https://habr.com/ru/post/309556/ (Дата обращения 06.09.2022)
4. Настройка связки Apache + PHP + MySQL + phpMyAdmin URL: <https://puzzleweb.ru/other/apache.php> (Дата обращения 06.09.2022)
5. Создание базы данных в MySQL URL: <https://selectel.ru/blog/tutorials/how-to-create-databases-in-mysql/> (Дата обращения 06.09.2022)