|  |  |  |
| --- | --- | --- |
|  |  |  |
| 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:**

**Студент группы** ИКБО-20-19 Московка А.А.

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

(подпись студента)

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

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

(подпись руководителя)

Работа представлена «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Допущен к работе «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Москва 2021

***Цель работы***

Используя серверную конфигурацию, разработанную в прошлой

практической работе выполнить следующие упражнения. Предполагается

создать 3 независимых сервиса, устойчивых к минимальному набору

самых простых ошибок.

***Задание 1:***

Предлагается создать веб-сервис Drawer для рисования svg объектов.

Ему передается один параметр - целое число, представляющее

закодированную фигуру для рисования.

***Код программы:***

*docker-compose.yml – общий для всех трех заданий*

**

Рисунок 1 – Листинг docker-compose.yml.

*Dockerfile*

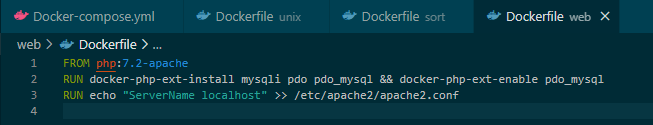


Рисунок 2 – Листинг Dockerfile (web)

*index.php*

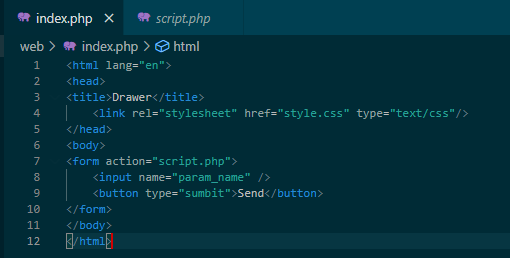
**

Рисунок 3 – Листинг index.php (Drawer)

*script.php*

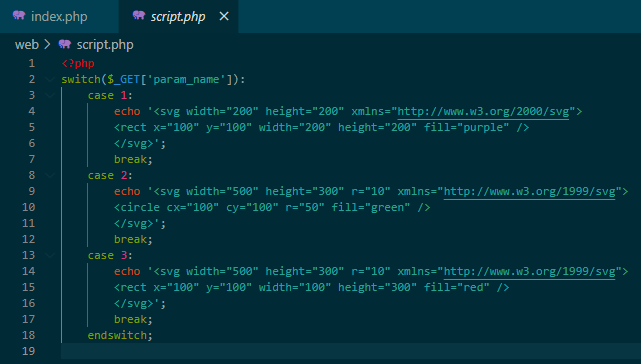
**

Рисунок 4 – Листинг script.php (Drawer)

***Результат:***

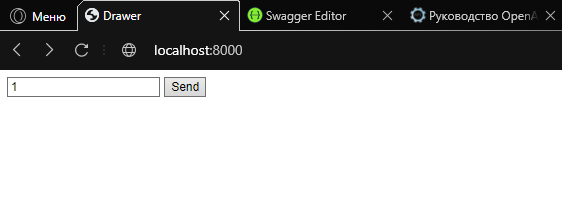


Рисунок 5 – Отправка запроса

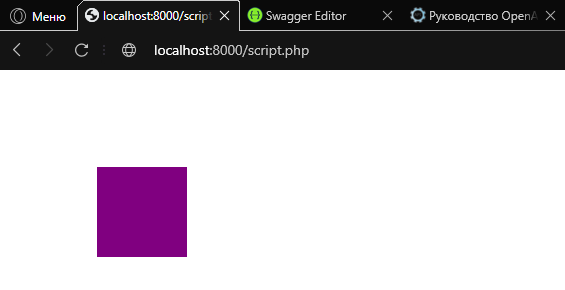


Рисунок 6 – Результат отрисовки квадрата

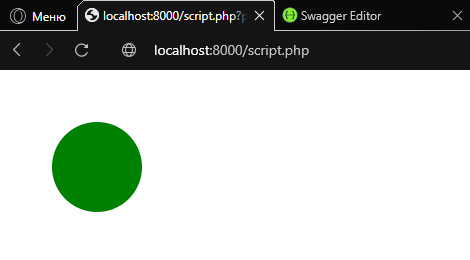


Рисунок 7 – Результат отрисовки круга

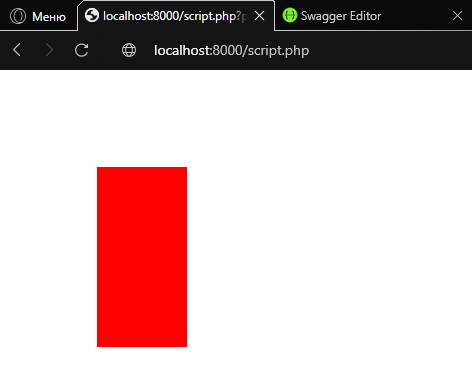


Рисунок 8 – Результат отрисовки прямоугольника

***Задание 2:***

Реализовать одну из сортировок на языке программирования PHP по

варианту: Вариант 1 – Быстрая сортировка.

***Код программы:***

*docker-compose.yml – общий для всех трех заданий*

**

Рисунок 9 – Листинг docker-compose.yml

*Dockerfile*

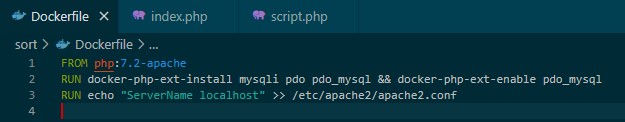


Рисунок 10 – Листинг Dockerfile (Sort)

*index.php*

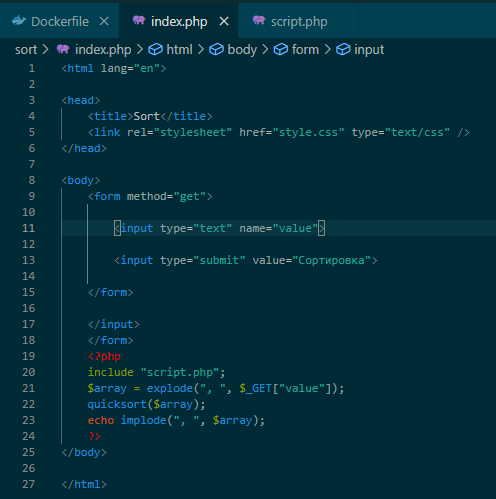


Рисунок 11 – Листинг index.php (Sort)

*script.php*

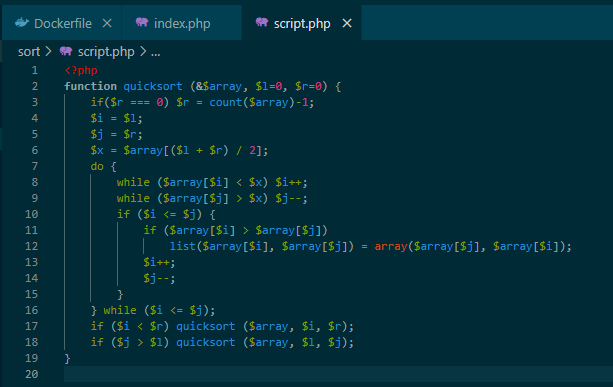


Рисунок 12 – Листинг script.php (Sort)

***Результат:***

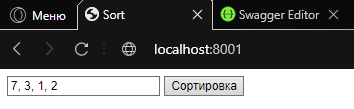


Рисунок 13 – Ввод неотсортированного массива

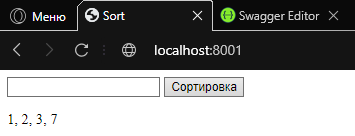


Рисунок 14 – Отсортированный массив

***Задание 3:***

Реализовать информационно-административную веб-страницу о

сервере с помощью таких команд Unix как: ls, ps, whoami, id и так далее.

***Код программы:***

*docker-compose.yml – общий для всех трех заданий*

**

Рисунок 15 – Листинг docker-compose.yml

*Dockerfile*

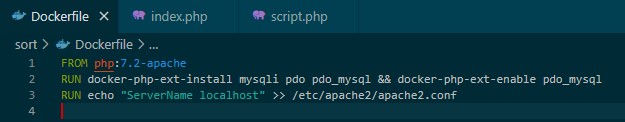


Рисунок 16 – Листинг Dockerfile (Unix)

*Index.php*

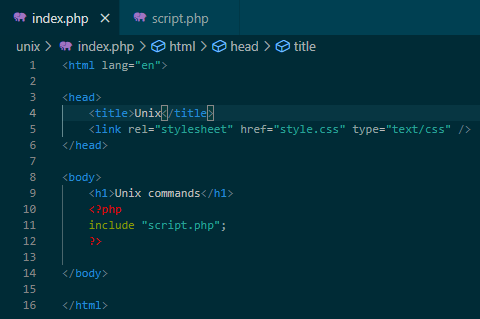
**

Рисунок 17 – Листинг index.php (Unix)

*Script.php*

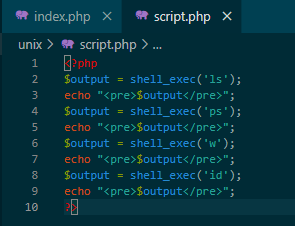
**

Рисунок 18 – Листинг script.php (Unix)

***Результат:***



Рисунок 19 – Результат вывода

Вопросы:

1. Конфигурационный файл php.ini.

Файл конфигурации (php.ini) считывается при запуске PHP. Для версий серверных модулей PHP это происходит только один раз при запуске веб-сервера.

1. Как написать простой скрипт на php.

Создать php файл и внутри <?php ?> писать скрипт.

1. Основные правила, связанные с переменными в php.
   1. Имя переменной чувствительно к регистру.
   2. Правильное имя переменной должно начинаться с буквы или символа подчёркивания и состоять из букв, цифр и символов подчёркивания в любом количестве.
   3. По умолчанию переменные всегда присваиваются по значению.
2. Основные типы данных в php.
   1. bool (логический тип)
   2. int (целые числа)
   3. float (дробные числа)
   4. string (строки)
   5. array (массивы)
   6. object (объекты)
   7. callable (функции)
   8. mixed (любой тип)
   9. resource (ресурсы)
   10. null (отсутствие значения)
3. Какие существуют функции для работы с переменными в php вне

зависимости от типа данных.

* 1. boolval — Возвращает логическое значение переменной
  2. debug\_zval\_dump — Выводит строковое представление внутренней структуры zval
  3. doubleval — Псевдоним floatval
  4. empty — Проверяет, пуста ли переменная
  5. floatval — Возвращает значение переменной в виде числа с плавающей точкой
  6. get\_debug\_type — Возвращает имя типа переменной в виде, подходящем для отладки
  7. get\_defined\_vars — Возвращает массив всех определённых переменных
  8. get\_resource\_id — Возвращает целочисленный идентификатор для данного ресурса
  9. get\_resource\_type — Возвращает тип ресурса
  10. gettype — Возвращает тип переменной
  11. intval — Возвращает целое значение переменной
  12. is\_array — Определяет, является ли переменная массивом
  13. is\_bool — Проверяет, является ли переменная булевой
  14. is\_callable — Проверяет, что значение может быть вызвано как функция в текущей области видимости
  15. is\_countable — Проверить, что содержимое переменной является счётным значением
  16. is\_double — Псевдоним is\_float
  17. is\_float — Проверяет, является ли переменная числом с плавающей точкой
  18. is\_int — Проверяет, является ли переменная целым числом
  19. is\_integer — Псевдоним is\_int
  20. is\_iterable — Проверяет, является ли переменная итерируемой
  21. is\_long — Псевдоним is\_int
  22. is\_null — Проверяет, является ли значение переменной равным null
  23. is\_numeric — Проверяет, является ли переменная числом или строкой, содержащей число
  24. is\_object — Проверяет, является ли переменная объектом
  25. is\_real — Псевдоним is\_float
  26. is\_resource — Проверяет, является ли переменная ресурсом
  27. is\_scalar — Проверяет, является ли переменная скалярным значением
  28. is\_string — Проверяет, является ли переменная строкой
  29. isset — Определяет, была ли установлена переменная значением, отличным от null
  30. print\_r — Выводит удобочитаемую информацию о переменной
  31. serialize — Генерирует пригодное для хранения представление переменной
  32. settype — Задаёт тип переменной
  33. strval — Возвращает строковое значение переменной
  34. unserialize — Создаёт PHP-значение из хранимого представления
  35. unset — Удаляет переменную
  36. var\_dump — Выводит информацию о переменной
  37. var\_export — Выводит или возвращает интерпретируемое строковое представление переменной

1. Предопределенные переменные в php.

PHP предоставляет всем скриптам большое количество предопределённых переменных. Эти переменные содержат всё, от внешних данных до переменных среды окружения, от текста сообщений об ошибках до последних полученных заголовков.

Например - $\_GET — Переменные HTTP GET

$\_POST — Переменные HTTP POST

1. Переменные переменных в php.

Переменная переменной - имя переменной, которое может быть определено и изменено динамически. Обычная переменная c таким выражением:

<?php  
$a = 'hello';  
?>

Переменная переменной берет значение переменной и рассматривает его как имя переменной. В вышеприведённом примере hello может быть использовано как имя переменной при помощи двух знаков доллара. То есть:

<?php  
$$a = 'world';  
?>

Теперь в дереве символов PHP определены и содержатся две переменные: $a, содержащая "hello" и $hello, содержащая "world".

1. Выражения в php.

Выражение – это "все что угодно, имеющее значение". Основными формами выражений являются константы и переменные. Немного более сложными примерами выражений являются функции.

1. Арифметические операторы в php.

* +$a Идентичность
* -$a Отрицание
* $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.

* $a & $b И
* $a | $b Или
* $a ^ $b Исключающее или
* ~ $a Отрицание
* $a << $b Сдвиг влево - Все биты переменной $a сдвигаются на $b позиций влево (каждая позиция подразумевает "умножение на 2")
* $a >> $b Сдвиг вправо - Все биты переменной $a сдвигаются на $b позиций вправо (каждая позиция подразумевает "деление на 2")

11.Оператор присваивания в php.

Базовый оператор присваивания обозначается как "=".

<?php  
$a = ($b = 4) + 5; // $a теперь равно 9, а $b было присвоено 4.  
?>

В дополнение к базовому оператору присваивания имеются "комбинированные операторы" для всех бинарных арифметических операций, операций объединения массивов и строковых операций, которые позволяют использовать некоторое значение в выражении, а затем установить его как результат данного выражения.

12.Операторы сравнения в php.

* $a == $b - Равно true если $a равно $b после преобразования типов.
* $a === $b - Тождественно равно true если $a равно $b и имеет тот же тип.
* $a != $b - Не равно true если $a не равно $b после преобразования типов.
* $a <> $b - Не равно true если $a не равно $b после преобразования типов.
* $a !== $b - Тождественно не равно true если $a не равно $b, или они разных типов.
* $a < $b - Меньше true если $a строго меньше $b.
* $a > $b - Больше true если $a строго больше $b.
* $a <= $b - Меньше или равно true если $a меньше или равно $b.
* $a >= $b - Больше или равно true если $a больше или равно $b.
* $a <=> $b - Космический корабль (spaceship) Число типа int меньше, больше или равное нулю, когда $a соответственно меньше, больше или равно $b.

13.Логические операторы в php.

* $a and $b - И
* $a or $b - Или
* $a xor $b - Исключающее или
* ! $a - Отрицание.
* $a && $b - И
* $a || $b – Или

14.Условная конструкция в php.

Конструкция if (условие) проверяет истинность некоторого условия, и если оно окажется истинным, то выполняется блок выражений, стоящих после if. Если же условие ложно, то есть равно false, тогда блок if не выполняется. Например:

<?php

$a = 4;

if($a>0){

echo "Переменная a больше нуля";

}

echo "<br>конец выполнения программы";

?>

Блок выражений ограничивается фигурными скобками. И так как в данном случае условие истинно (то есть равно true): значение переменной $a больше 0, то блок инструкций в фигурных скобках также будет выполняться. Если бы значение $a было бы меньше 0, то блок if не выполнялся.

Если блок if содержит всего одну инструкцию, то можно опустить фигурные скобки:

<?php

$a = 4;

if($a>0)

echo "Переменная a больше нуля";

echo "<br>конец выполнения программы";

?>

Можно в одной строке поместить всю конструкцию:

if($a>0) echo "Переменная a больше нуля";

15.Циклы в php.

Циклы позволяют повторять определенное (и даже неопределенное - когда работа цикла зависит от условия) колличество раз различные операторы. Данные операторы называются телом цикла. Проход цикла называется итерацией.

PHP поддерживает 4 вида циклов:

* Цикл с предусловием (while);
* Цикл с постусловием (do-while);
* Цикл со счетчиком (for);
* Специальный цикл перебора массивов (foreach).

При использовании циклов есть возможность использования операторов break и continue. Первый из них прерывает работу всего цикла, а второй - только текущей итерации.

16. Конструкции switch и match в php.

Конструкция switch..case является альтернативой использованию конструкции if..elseif..else. Оператор switch получает некоторое выражение и сравнивает его с набором значений.

После ключевого слова switch в скобках идет сравниваемое выражение. Значение этого выражения последовательно сравнивается со значениями, помещенными после операторов сase. И если совпадение будет найдено, то будет выполняться определенный блок сase.

Начиная с версии 8.0 в PHP была добавлена поддержка другой, похожей конструкции - match. Она позволяет оптимизировать конструкцию switch. Конструкция match также принимает некоторое выражение и сравнивает его с набором значений.

17. Include и require в php.

В PHP есть две функции, которые используются для помещения содержимого файла, содержащего исходный код PHP, в другой файл PHP. Это функции Include() и Require(). Обе функции одинаковы, но они имеют одно различие. Разница в том, что функция include() выдает предупреждение, но скрипт продолжит выполнение, а функция require() выдает предупреждение и фатальную ошибку, т.е. скрипт не будет продолжать выполнение. Эти две функции используются для помещения данных файла в другой файл PHP перед его выполнением сервером.

18. Функции в php.

Внутри функции можно использовать любой корректный PHP-код, в том числе другие функции и даже объявления классов.

Имена функций следуют тем же правилам, что и другие метки в PHP. Корректное имя функции начинается с буквы или знака подчёркивания, за которым следует любое количество букв, цифр или знаков подчёркивания. В качестве регулярного выражения оно может быть выражено так: ^[a-zA-Z\_\x80-\xff][a-zA-Z0-9\_\x80-\xff]\*$. Все функции и классы PHP имеют глобальную область видимости - они могут быть вызваны вне функции, даже если были определены внутри и наоборот.

Вывод

При выполнении текущей практической работы были получены навыки работы с функциями, алгоритмами сортировки и использования различных методов.

Список использованных источников

1. Моуэт, Э. Использование Docker / Э. Моуэт ; научный редактор А. А. Маркелов ; перевод с английского А. В. Снастина. — Москва : ДМК Пресс, 2017. — 354 с. — ISBN 978-5-97060-426-7. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/93576 (дата обращения: 16.09.2021). — Режим доступа: для авториз. пользователей.
2. Алибеков, Б. И. Лабораторный практикум по Web-программированию на PHP : учебное пособие / Б. И. Алибеков. — Махачкала : ДГУ, 2018. — 273 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/158357 (дата обращения: 16.09.2021). — Режим доступа: для авториз. пользователей.
3. Джош, Л. Современный PHP. Новые возможности и передовой опыт / Л. Джош ; перевод с английского Р. Н. Рагимов. — Москва : ДМК Пресс, 2016. — 304 с. — ISBN 978-5-97060-184-6. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/93269 (дата обращения: 16.09.2021). — Режим доступа: для авториз. пользователей.