**Министерство науки и высшего образования Российской Федерации**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**ITMO University**

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3**

**По дисциплине** Web-программирование

**Тема работы** Разработка веб-приложений с использованием Gulp и PHP

**Обучающийся** Денисов Илья Алексеевич

**Факультет** прикладной информатики

**Группа** К3323

**Направление подготовки** 11.03.02 Инфокоммуникационные технологии и системы связи

**Образовательная программа** Программирование в инфокоммуникационных системах

|  |  |  |  |
| --- | --- | --- | --- |
| **Обучающийся** | (дата) | (подпись) | Денисов И.А.  (Ф.И.О.) |
| **Руководитель** | (дата) | (подпись) | Марченко Е.В.  (Ф.И.О.) |

Оглавление

[Часть 1. Введение 3](#_Toc185363322)

[1. Цель работы 3](#_Toc185363323)

[2. Задачи работы 3](#_Toc185363324)

[Часть 2. Основная часть 3](#_Toc185363325)

[1. Настройка gulp 3](#_Toc185363326)

[3. Методы GET и POST 5](#_Toc185363327)

[4. Форма обратной связи и php-handler 5](#_Toc185363328)

[5. Установка инструментария для отладки проектов 6](#_Toc185363329)

[Часть 3. Вывод 8](#_Toc185363330)

# Часть 1. Введение

## Цель работы

Изучить gulp для запуска нескольких задач одновременно, изучить основы работы с PHP, и то, как можно использовать WordPress для Web разработки.

## 2. Задачи работы

1. Настроить gulp: а) создать два таска – настроить на последовательное и параллельное выполнение; б) настроить отображение файлов проекта в браузере и автоматическую перезарузку при изменении одного из контролируемых файлов проекта.
2. Создать форму для отправки информации по обратной связи от пользователя. Разработать файл с формой и php скрипт по образцу. В отчете привести информацию о использовании методов get и метод post.
3. Установить инструментарий для отладки проектов. Настроить портал <http://test.site> с помощью движка с портала wordpress.

# Часть 2. Основная часть

## Настройка gulp

В данном задании требуется настроить gulp: создать два таска – настроить на последовательное и параллельное выполнение, а также настроить отображение файлов проекта в браузере и автоматическую перезагрузку при изменении одного из контролируемых файлов проекта.

Был создан файл gulpfile.js, в котором реализованы 3 таски: styles – для компиляции SASS в CSS, serve – для автоматической перезагрузки при изменениях в контролируемых файлов проекта (html или scss), add-text – для добавления текста в конец html страницы.

Вид исходной страницы index.html представлен на рисунке 1.

Изображение выглядит как текст, снимок экрана, Шрифт, зеленый

Автоматически созданное описание

Рисунок 1. Исходная страница index.html

В gulpfile.js было реализовано параллельное и последовательное выполнение задач с помощью методов parallel и series соответственно. В тасках styles и serve была реализована пауза в 3 секунды с помощью написанной функции sleep. Это позволило более явно увидеть отличие параллельного и последовательного выполнения задач. При последовательном запуске все задачи выполнялись за 6.17 секунд (логи последовательного запуска представлены на рисунке 2). При параллельном – за 3.14, что говорит о том, что задачи styles и serve были запущены параллельно (логи параллельного запуска представлены на рисунке 3). Также было проверено, что задача add-text действительно добавляет текст в html, и то, что при изменении файла scss меняется отображение страницы в браузере (рисунок 4).

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2. Логи последовательного запуска

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

Рисунок 3. Логи параллельного запуска

Изображение выглядит как текст, снимок экрана, Шрифт, Цвет электрик

Автоматически созданное описание

Рисунок 4. index.html после добавления текста и обновления стилей

## Методы GET и POST

**Метод GET**

* Данные передаются через URL, и они могут быть видны в адресной строке браузера. Примером использования метода GET будет форма поиска, где вводимые данные не являются конфиденциальными.
* Обычно используется для получения данных с сервера.

**Метод POST**

* Данные отправляются через HTTP-заголовок, и они не отображаются в адресной строке. Это делает его более безопасным для передачи конфиденциальной информации (например, паролей, личных данных).
* Используется для создания или обновления данных на сервере.

## Форма обратной связи и php-handler

В данном задании было необходимо разработать файл html формы, запрашивающей обратную связь у пользователя, а также скрипт-обработчик данной формы на PHP.

Для выполнения этого задания был установлен дистрибутив для сборки локального веб-сервера XAMPP. Далее был запущен Apache сервер. В папке C:\xampp\htdocs была создана папка проекта feedback-form, в которой были реализованы форма form.html и обработчик form-handler.php.

В файле обработчика были прописаны запросы на получение данных из массива $\_POST, который хранит в себе данные, отправленные методом POST. Проверка на то, отправлен ли POST запрос на сервер, является условием получения данных. Также используется функция htmlspecialcharts() для преобразования специальных символов в HTML-сущности.

Страница заполненной формы и результат её обработки после нажатия кнопки «Отправить» представлена на рисунке 5.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описаниеИзображение выглядит как текст, снимок экрана, Шрифт, алгебра

Автоматически созданное описание

Рисунок 5. Форма обратной связи и результат её обработки

## Установка инструментария для отладки проектов WordPress

На компьютер был установлен WordPress с официального сайта. Далее с помощью phpMyServer была создана база данных wordpress.

Изображение выглядит как текст, программное обеспечение, Значок на компьютере, Шрифт

Автоматически созданное описание

Рисунок 6. Создание БД для WordPress

Далее по адресу <http://localhost/wordpress> созданная БД была добавлена в WordPress. После этого была заполнена информация о сайте - название, имя пользователя, пароль и адрес электронной почты, с указанием имени хоста test.site.

Далее необходимо было создать виртуальный хост для test.site. Для этого в файле C:\Windows\System32\drivers\etc\hosts для локалхоста 172.0.0.1 был добавлен виртуальный адрес test.site. Также, чтобы форма обратной связи продолжала работать, для неё также был реализован виртуальный хост.

Изображение выглядит как текст, Шрифт, снимок экрана, Графика

Автоматически созданное описание

Рисунок 7. Прописывание виртуальных хостов в hosts

Добавление виртуальных адресов в XAMPP

Далее виртуальные хосты были добавлены в конфигурацию XAMPP(в файл C:\xampp\apache\conf\extra\httpd-vhosts.conf).

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 8. Добавление виртуальных хостов в XAMPP

Как итог – по адресу test.site/wordpress работает тестовый блог (рисунок 9). Форма обратной связи также работает по адресу feetback.local (рисунок 10).

Изображение выглядит как текст, снимок экрана, программное обеспечение, Значок на компьютере

Автоматически созданное описание

Рисунок 9. Сайт по адресу test.site/wordpress

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Автоматически созданное описание

Рисунок 10. Форма обратной связи по новому адресу

# Часть 3. Вывод

В данной работе были изучены два разных способа запуска выполнения задач в Gulp: последовательно и параллельно. Также были закреплены основы работы с gulp и browsersync для подгрузки изменений в файлах на страницу в браузере. Были изучены основы работы с XAMPP, реализован php скрипт для обработки данных из формы. Был создан первый сайт с использованием WordPress, изучено создание виртуальных хостов и протестирован запуск сайта на виртуальном хосте.