

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**  
**ITMO University**

Факультет «Факультет инфокоммуникационных технологий»

Направление подготовки «Инфокоммуникационные технологии и системы  
связи»

Практическая работа №2

Выполнила:  
Колтунова Полина Владимировна  
Группа №K3320  
Проверила:  
Марченко Елена Вадимовна

Санкт-Петербург  
2024

## ВВЕДЕНИЕ

Цель работы: изучить основы работы с Git, gulp и написать программу клиент.

Задания:

Задание 1. Установить Git на компьютер, настроить на работу с проектом, выполнить изменения в файлах проекта. Для выполняемых изменений сделать коммиты (не менее трех). Проверить, что коммиты создаются. Локальный репозиторий синхронизировать с удаленным.

Задание 2. Установить Gulp. Проверить процесс установки, отметить основные этапы. Создать task.

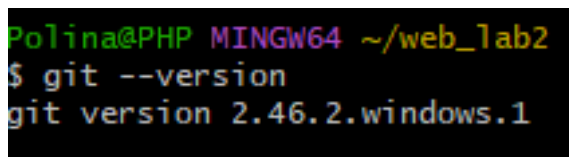
Задание 3. Написать программу клиент, которая показывает web-страницы одна за другой из списка (в программе можно задавать адреса страниц и интервал показа).

## ГЛАВА 1

### Задание 1. Работа с Git

Ссылка на проект: [https://github.com/polinakoltunova/web\\_lab2\\_task1\\_git](https://github.com/polinakoltunova/web_lab2_task1_git)

На компьютере уже была установлена программа Git. Проверка наличия Git на системе была выполнена с помощью команды `git --version`. Результатом выполнения команды подтвердилось, что Git установлен и готов к использованию (рис.1).



```
Polina@PHP MINGW64 ~/web_lab2
$ git --version
git version 2.46.2.windows.1
```

*Рисунок 1 – Проверка наличия Git на системе*

Далее был создан локальный репозиторий.

Для этого была создана папка `task1_git`, в которую были добавлены 2 файла с расширением `.html` из практической работы №1. Затем был инициализирован локальный репозиторий с помощью команды `git init`. Затем добавлены файлы в репозиторий с помощью команды (`git add .`). После чего были поочередно добавлены изменения в каждый файл и эти изменения закоммичены с помощью команды `git commit -m "описание коммита"`.

Проверка истории коммитов была сделана при помощи команды `git log` (рис.2).

```
$ git log
commit e8c753a1a7407a53963888a19f0d526eb3890070 (HEAD -> master)
Author: polina.koltunova <polinakoltunovapol@mail.ru>
Date: Wed Oct 2 11:55:18 2024 +0300

    Третий коммит - внесены изменения в файл findings.html

commit e3f17678f35aa03b7eb1f34bca4d7055b30006f1
Author: polina.koltunova <polinakoltunovapol@mail.ru>
Date: Wed Oct 2 11:54:18 2024 +0300

    Второй коммит - внесены изменения в файл code.html

commit 7247e820ae1bd6b419e4f9976acf2ed819764542
Author: polina.koltunova <polinakoltunovapol@mail.ru>
Date: Wed Oct 2 11:53:05 2024 +0300

    Первый коммит - добавлены файлы
```

*Рисунок 2 - Проверка истории коммитов*

Для настройки удаленного репозитория был создан новый репозиторий web\_lab2\_task1\_git на GitHub. Затем локальный репозиторий был связан с удаленным при помощи команды `git remote add origin https://github.com/polinakoltunova/web_lab2_task1_git.git`. Затем с ветки master переключились на ветку main с командой `git checkout main`. Далее были объединены локальные изменения из ветки master (в которой были проведены изменения с файла и коммиты) в ветку main при помощи команды `git merge master --allow-unrelated-histories`.

В итоге локальный и удаленный репозитории синхронизовались, коммиты и изменения успешно отправились, что показано на рис.3.

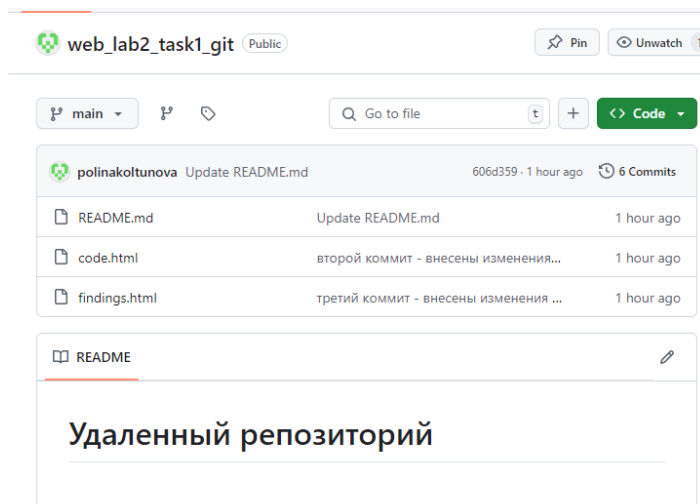


Рисунок 3 – Проверка успешной синхронизации

## Задание 2. Работа с Gulp

Перед установкой Gulp был установлен сервер Node JS и вместе с ним пакетный менеджер NPM с официального сайта <https://nodejs.org/en/> (рис.4).

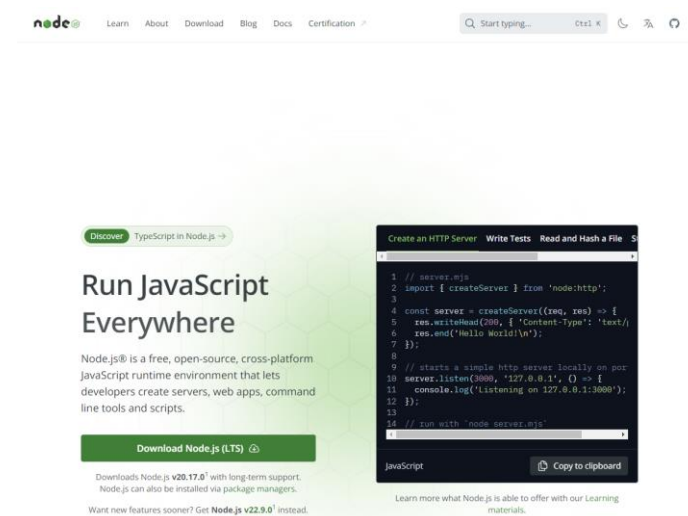
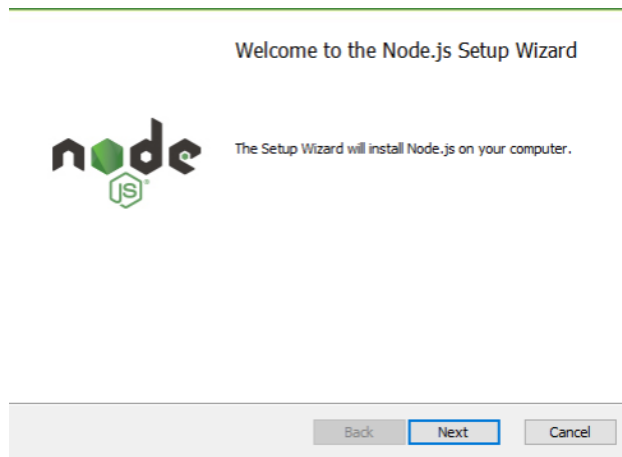
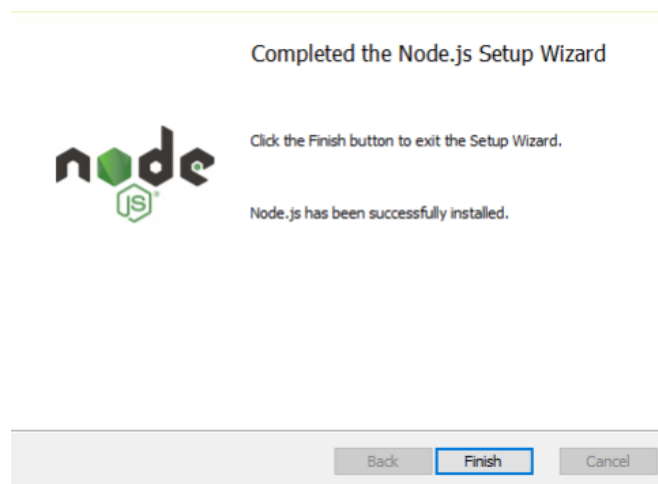


Рисунок 4 - Сайт <https://nodejs.org/en/>

Процесс установки Node JS и NPM показан на рис.5-6.



*Рисунок 5 - Процесс установки Node JS и NPN*



*Рисунок 6 - Процесс установки Node JS и NPN*

Проверка установки показана на рис.7. Было установлено все корректно, и видна информация о версиях.

```
MINGW64:/c:/Users/Polina

Polina@PHP MINGW64 ~
$ node -v
v20.17.0

Polina@PHP MINGW64 ~
$ npm -v
10.8.2
```

*Рисунок 7 - Проверка установки*

Для установки Gulp были введены следующие команды, перечисленные на официальном сайте <https://gulpjs.com/docs/en/getting-started/quick-start>:

`npm install --global gulp-cli` //установлена утилита командной строки

`npx mkdirp web_lab2` //создан каталог проекта

`npm init` //создан файл `package.json` в каталоге проекта

`npm install --save-dev gulp` //установлен пакет `gulp` в секцию `devDependencies` файла `package.json` для использования `gulp` только для разработки.

В итоге код файла `package.json` представлен на рис.8.

```
{
  "name": "ex2",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "description": "",
  "devDependencies": {
    "gulp": "^5.0.0"
  }
}
```

*Рисунок 8 - Код файла `package.json`*

Затем была реализована задача `Gulp` с именем `hello`, которая выводит сообщение в консоль.

Для этого была импортирована библиотека `Gulp`, которая является инструментом для автоматизации задач в разработке. Затем была определена задача с именем `hello`. Внутри задачи определена функция, которая выполняется при вызове задачи. Эта функция содержит команду `console.log`, которая выводит строку в консоль. Функция обратного вызова `callback` вызывается для того, чтобы сообщить `Gulp`, что задача завершена.

Код файла `gulpfile.js` представлен на рис.9.

```
const gulp = require('gulp');
2 references
gulp.task("hello", function (callback) {
  console.log("Hello, people!");
  callback();
});
```

*Рисунок 9 - Итоговый код файла gulpfile.js*

Для выполнения задачи была выполнена команда `gulp hello` (рис.10).

```
Polina@PHP MINGW64 ~/web_lab2
$ gulp hello
[13:13:18] Using gulpfile ~\web_lab2\gulpfile.js
[13:13:18] Starting 'hello'...
Hello, people!
[13:13:18] Finished 'hello' after 1.42 ms
```

*Рисунок 10 - Запуск задачи*

### **Задание 3.**

Была написана программа клиент, которая показывает web-страницы одна за другой из списка. Функционал программы подразумевает добавление адресов страниц, задания интервала показа, просмотр введенных адресов, остановка показа и очистка списка адресов.

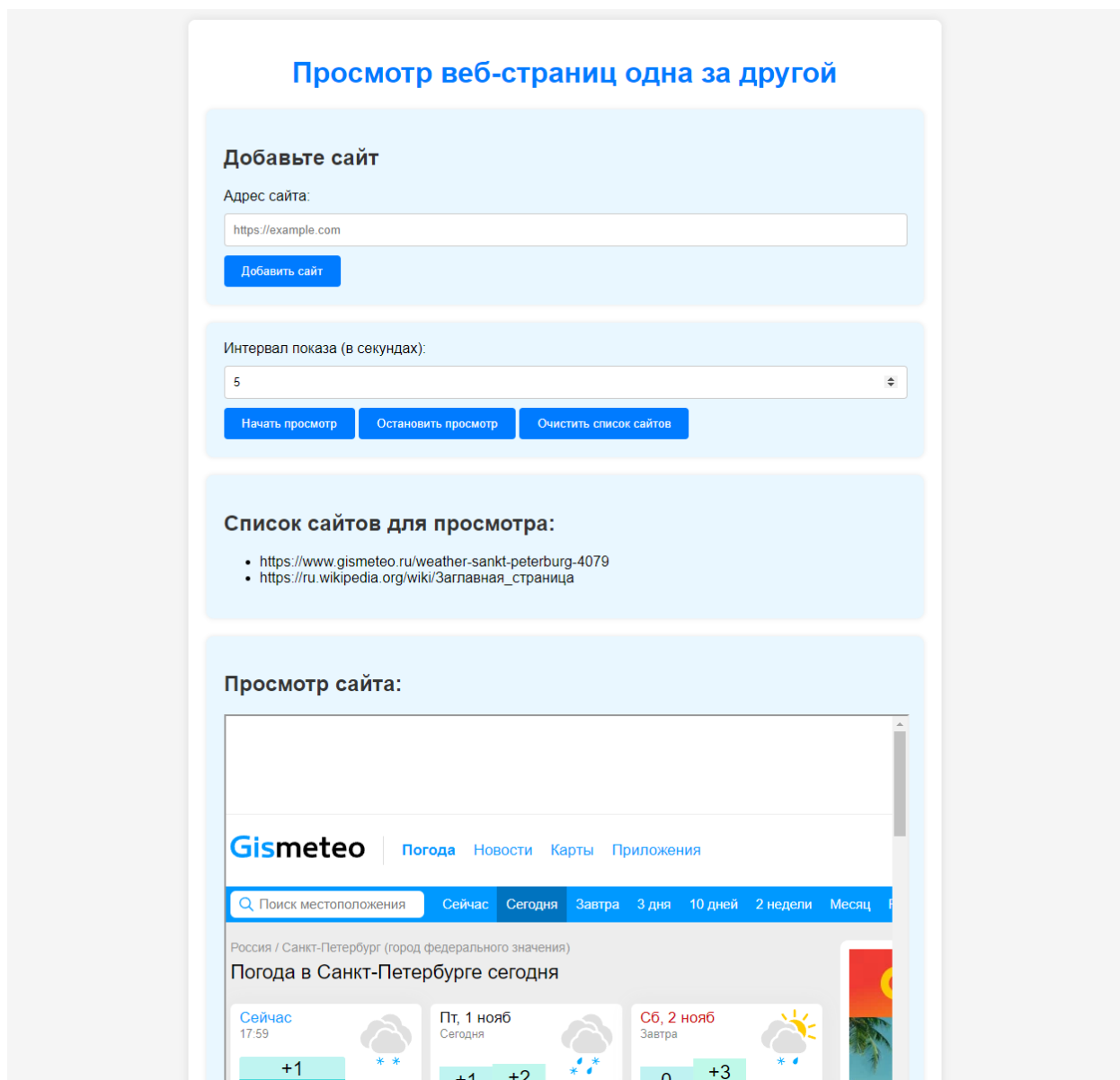
Файл `index.html` отвечает за структуру и элементы страницы: текстовые поля для ввода адресов и интервала показа, кнопки управления и область просмотра в `iframe`. К нему подключены `style.css` и `main.js`, которые дополняют его функциональность. Файл `style.css` придаёт стилевое оформление всем элементам, улучшая их внешний вид и создавая удобный интерфейс. Файл `main.js` содержит JavaScript-код, управляющий поведением элементов: добавлением адресов в список, циклическим отображением сайтов с заданным интервалом, остановкой и очисткой списка.

В файле `main.js` сначала инициализируются переменные для работы с элементами интерфейса, такими как поля ввода адреса и интервала, кнопки управления и область просмотра в `iframe`. Введённые адреса хранятся в



массиве `urls`, а текущий индекс сайта отслеживается переменной `currentIndex`. При нажатии на кнопку добавления сайта `addButton` проверяется валидность адреса, и если он новый, он добавляется в список. Кнопка запуска просмотра `startButton` начинает последовательное отображение сайтов в `iframe` с заданным интервалом; при этом каждый сайт загружается на основе `currentIndex`, который увеличивается циклически, чтобы показывать сайты по очереди. При нажатии на кнопку остановки просмотра `stopButton` текущий интервал очищается, а область просмотра скрывается. Кнопка очистки списка `clearButton` удаляет все адреса из массива и визуального списка, и при активном просмотре автоматически его завершает.

В итоге страница с заполненными полями выглядит следующим образом (рисунок 11).



*Рисунок 11 – Вид страницы*

## ВЫВОД

Цель работы достигнута. Было проверено, что на компьютере установлен Git, затем был создан локальный репозиторий, проделаны изменения с файлами, для которых сделаны коммиты. После чего локальный репозиторий был синхронизован с удаленным. Был установлен Gulp и создана задача task. Написана программа клиент, которая показывает web-страницы с интервалом одна за другой из списка сайтов, которые вводит пользователь.