

Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ЛАБОРАТОРНАЯ РАБОТА №3

по Технологии разработки ПО

Изучение Git и GitHub

Студент

Лобов М.Ю.

Группа АИ-18

Руководитель

Назаркин О.А.

Доцент

Липецк 2021 г.

Цель работы

Изучить на практике средства контроля версий Git и веб-сервис для хостинга IT-проектов GitHub.

Задание кафедры

Попробовать в использовании некоторые команды Git и изучить возможности использования сервиса GitHub.

Ход работы

Git — распределённая система контроля версий, которая даёт возможность разработчикам отслеживать изменения в файлах и работать над одним проектом совместно с коллегами.

Подход Git к хранению данных похож на набор снимков миниатюрной файловой системы. Каждый раз, когда вы сохраняете состояние своего проекта в Git, система запоминает, как выглядит каждый файл в этот момент, и сохраняет ссылку на этот снимок.

Перечислим и опишем основные команды:

1) `git config` – применяется для указания пользовательских настроек. К примеру, данная команда используется для установки адреса электронной почты:

```
git config --global user.email адрес@gmail.com.
```

2) `git init` – используется для создания GIT репозитория.

3) `git add` – может быть использована для добавления файлов в индекс.

4) `git clone` – используется для клонирования репозитория.

5) `git commit` – используется для коммита изменений в файлах проекта. Обратите внимание, что коммиты не сразу попадают на удаленный репозиторий.

6) `git push` – еще одна из часто используемых `git` команд. Позволяет поместить изменения в главную ветку удаленного хранилища, связанного с рабочим каталогом.

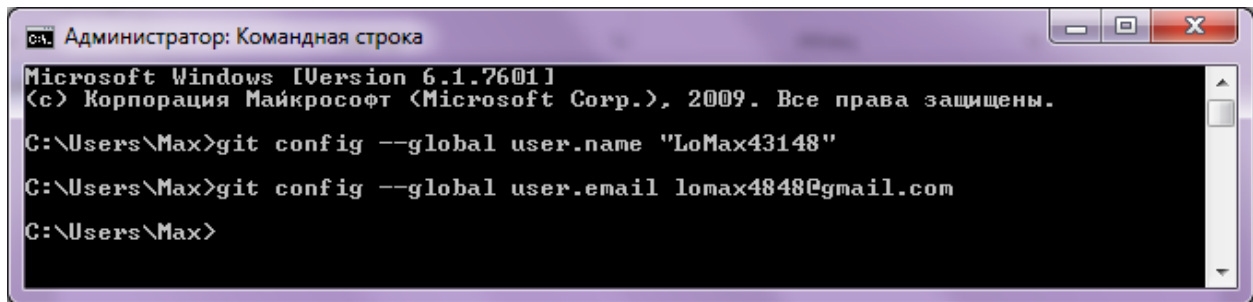
7) `git checkout` – может быть использована для создания веток или переключения между ними.

8) `git branch` – может быть использована для отображения, создания или удаления веток.

9) `git merge` – используется для объединения ветки в активную ветвь.

10) `git pull` – используется для объединения изменений, присутствующих в удаленном репозитории, в локальный рабочий каталог.

Для начала работы с git необходимо авторизоваться, выполнив в командной строке следующие команды:

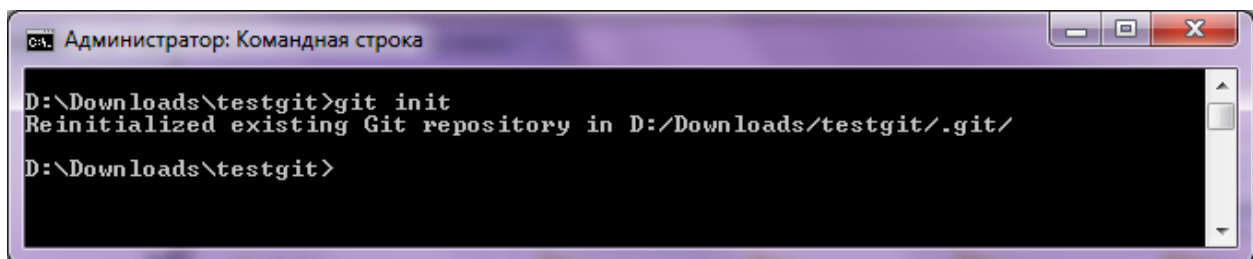


```
Microsoft Windows [Version 6.1.7601]
(c) Корпорация Майкрософт (Microsoft Corp.), 2009. Все права защищены.

C:\Users\Max>git config --global user.name "LoMax43148"
C:\Users\Max>git config --global user.email lomax4848@gmail.com
C:\Users\Max>
```

Рисунок 1 – Настройка имени пользователя

Далее инициализируем репозиторий:




```
D:\Downloads\testgit>git init
Reinitialized existing Git repository in D:/Downloads/testgit/.git/

D:\Downloads\testgit>
```

Рисунок 2 – Инициализация репозитория

Теперь добавим все файлы данной директории в очередь на загрузку в локальный репозиторий командой «**git add .**». И проверим, добавлены ли файлы:



```
D:\Downloads\testgit>git status
On branch master

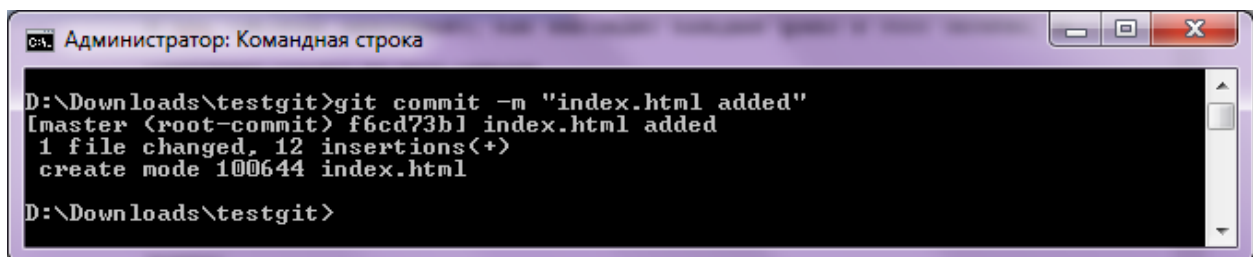
No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   index.html

D:\Downloads\testgit>
```

Рисунок 3 – Добавлены новые файлы в репозиторий

Теперь подтвердим данные изменения, выполнив команду **git commit**:



```
D:\Downloads\testgit>git commit -m "index.html added"
[master (root-commit) f6cd73b1] index.html added
 1 file changed, 12 insertions(+)
 create mode 100644 index.html

D:\Downloads\testgit>
```

Рисунок 4 – Первый коммит

Теперь создадим новую ветку в данной репозитории, выполнив команду **git branch newbranch**, где newbranch – название ветки. Теперь перейдём на эту ветку, выполнив команду **git checkout newbranch**.

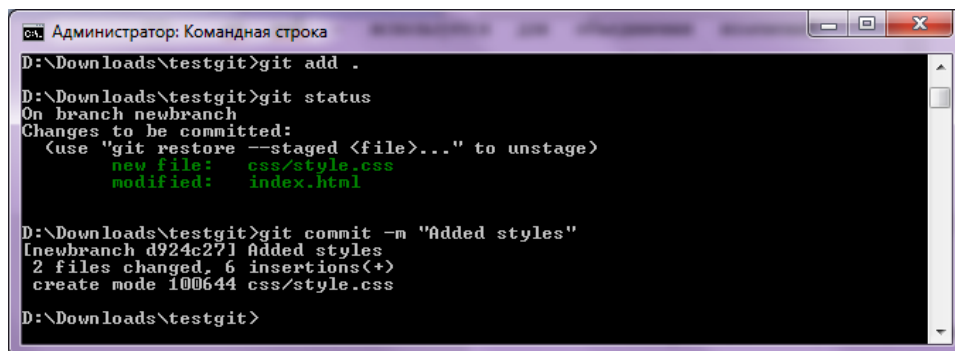
Поработаем над файлами в нашей директории. Например, добавим папку css, в которой будет храниться файл со стилями style.css. Заполним его:

```
h1 {  
    color: red;  
    font-family: 'Times New Roman', Times, serif;  
    font-style: italic;  
}
```

И добавим ссылку на этот файл в index.html:

```
<link rel="stylesheet" href="css/style.css">
```

Далее добавляем изменения с помощью команды **git add** и подтверждаем изменения с помощью команды **git commit**:

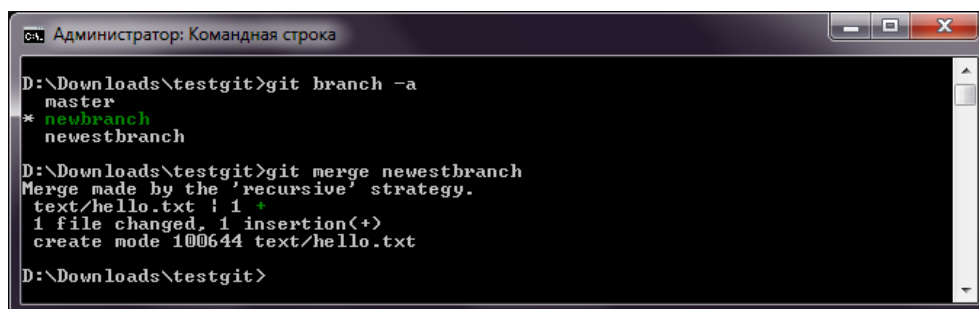


```
Администратор: Командная строка  
D:\Downloads\testgit>git add .  
D:\Downloads\testgit>git status  
On branch newbranch  
Changes to be committed:  
  (use "git restore --staged <file>..." to unstage)  
    new file:   css/style.css  
    modified:   index.html  
  
D:\Downloads\testgit>git commit -m "Added styles"  
[newbranch d924c27] Added styles  
 2 files changed, 6 insertions(+)  
 create mode 100644 css/style.css  
D:\Downloads\testgit>
```

Рисунок 5 – Коммит в новой ветке

Создадим ещё одну ветку командой **git checkout -b newestbranch** и добавим папку text, в которую перенесём текстовый файл. Выполним команды **git add** и **git commit** для сохранения изменений в данной ветке.

Попробуем теперь объединить ветки newbranch и newestbranch следующим образом:



```
Администратор: Командная строка  
D:\Downloads\testgit>git branch -a  
master  
* newbranch  
newestbranch  
  
D:\Downloads\testgit>git merge newestbranch  
Merge made by the 'recursive' strategy.  
 text/hello.txt | 1 +  
 1 file changed, 1 insertion(+)  
 create mode 100644 text/hello.txt  
D:\Downloads\testgit>
```

Рисунок 6 – Объединение веток

Теперь нам нужно «запустить» данный репозиторий на удалённый сервер с помощью сервиса GitHub. Для этого создадим новый удалённый репозиторий:

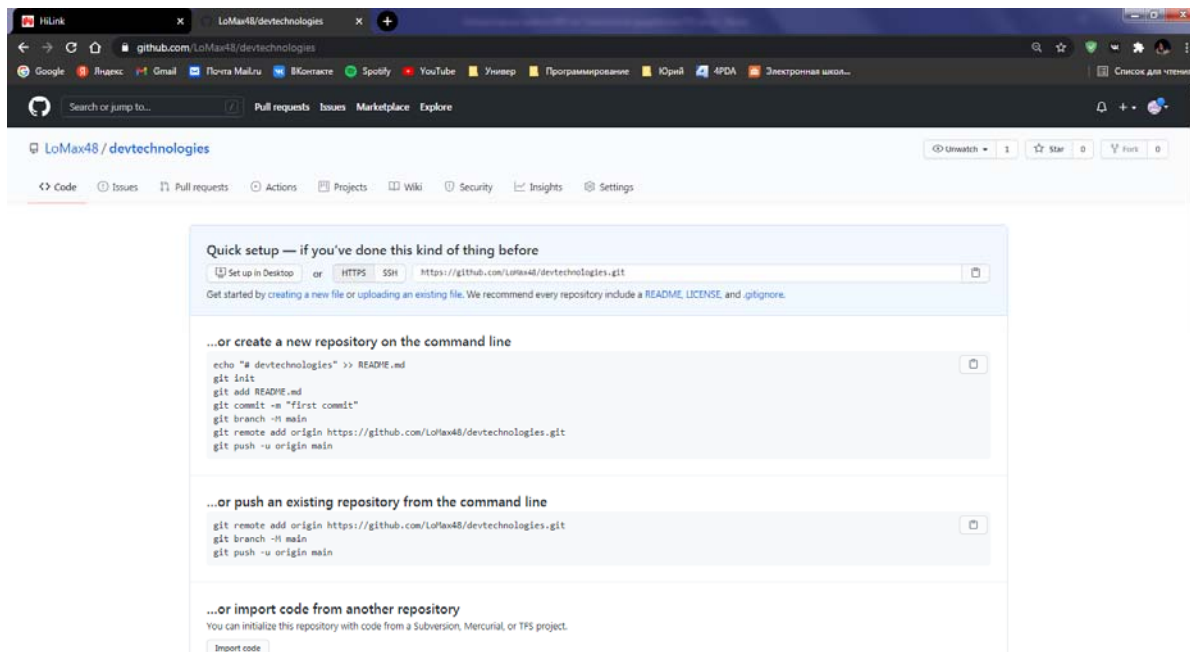


Рисунок 7 – Создание репозитория

И внесём изменения, перенеся данные из локального репозитория. Для этого в командной строке выполним команды **git remote add origin https://github.com/LoMax48/devtechnologies.git**, **git push -u origin master** и **git push -u origin newbranch**. Посмотрим, что получилось:

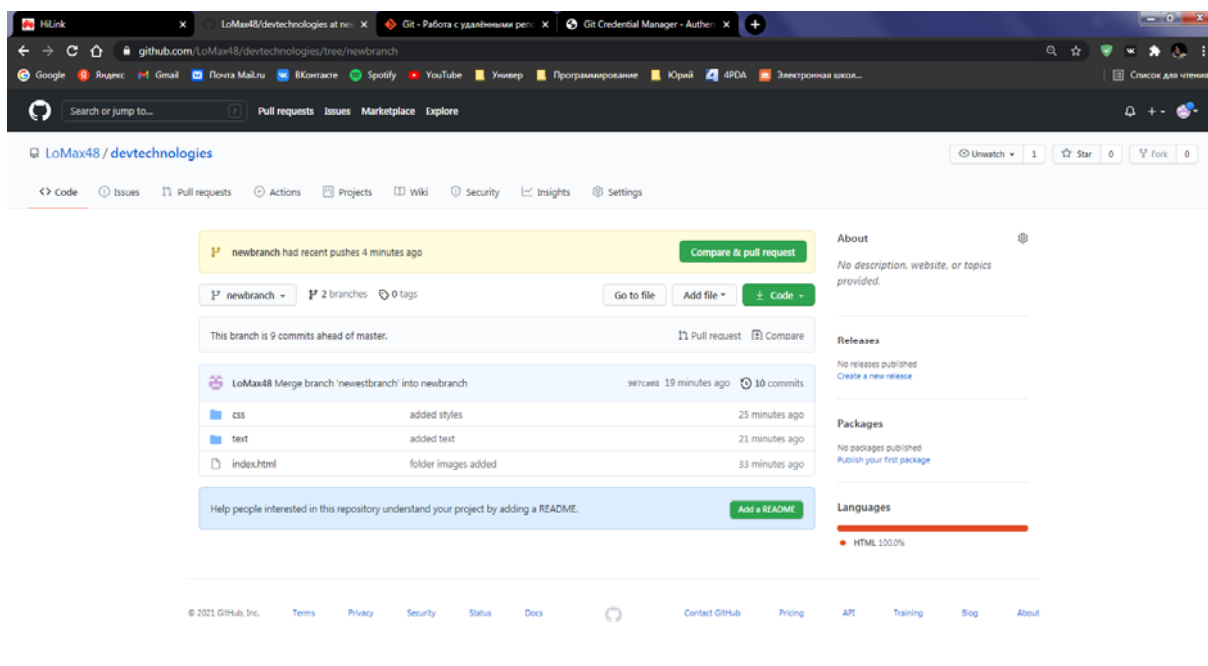


Рисунок 8 – Содержимое ветки «newbranch» в удалённом репозитории

Вывод

В ходе выполнения лабораторной работы были изучены основные команды и принципы работы с ветками в Git. Также изучен механизм загрузки локального репозитория на удалённый сервер с помощью сервиса GitHub.