



**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**  
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**  
**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ЛИПЕЦКИЙ**  
**ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт      компьютерных наук  
Кафедра      автоматизированных систем управления

**ЛАБОРАТОРНАЯ РАБОТА №6**  
**по дисциплине**  
**“Технологии разработки программного обеспечения”**

Студент      АС-21-1      \_\_\_\_\_      Станиславчук С. М.  
(подпись, дата)

Руководитель  
Доцент, к.т.н.      \_\_\_\_\_      Назаркин О. А.  
(подпись, дата)

Липецк 2024

## Содержание

1. Зачем нужен git?
2. Пошаговая загрузка изменений в локальный репозиторий (commit) и удаленный репозиторий (push) github.com

## 1. Зачем нужен гит?

Git — абсолютный лидер по популярности среди современных систем управления версиями. Это развитый проект с активной поддержкой и открытым исходным кодом. Система Git была изначально разработана в 2005 году Линусом Торвальдсом — создателем ядра операционной системы Linux. Git применяется для управления версиями в рамках колоссального количества проектов по разработке ПО, как коммерческих, так и с открытым исходным кодом. Система используется множеством профессиональных разработчиков программного обеспечения. Она превосходно работает под управлением различных операционных систем и может применяться со множеством интегрированных сред разработки (IDE).

С помощью Git-а вы можете откатить свой проект до более старой версии, сравнивать, анализировать или сливать свои изменения в репозиторий.

Репозиторием называют хранилище вашего кода и историю его изменений. Git работает локально и все ваши репозитории хранятся в определенных папках на жестком диске.

Репозитории можно хранить и в интернете. Обычно для этого используют два сервиса:

GitHub

GitLab

Также, недавно было запущено 2 отечественных проекта GitFlic и GitVerse.

Каждая точка сохранения вашего проекта носит название коммит (commit). У каждого commit-а есть hash (уникальный id) и комментарий. Из таких commit-ов собирается ветка. Ветка - это история изменений. У каждой ветки есть свое название. Репозиторий может содержать в себе несколько веток, которые создаются из других веток или вливаются в них.

Таким образом, создание точек сохранений кода - это неотъемлемая часть любого более-менее большого проекта сегодня. Устранение ошибок, путем отката на предыдущую версию проекта; многопользовательский режим работы над проектом, путем слияния веток разных пользователей и многое другое; контроль изменений между версиями (как команда diff в терминале).

## 2. Демонстрация функционала

Для начала установим git:

```
# Arch linux
sudo pacman -S git
```

Теперь укажем имя и почту в конфиге git

```
git config --global user.name "<ваше_имя>"
git config --global user.email "<адрес_почты@email.com>"
```

Теперь всё готово, чтобы создать репозиторий

```
# Инициализация/создание репозитория
git init # инициализировать в директории преокта

git add . # добавить все файлы директории в коммит

git reset unityhub # убрать ненужные директории

git rm -r --cached '*Cache*' # удалить из коммита все подпапки, содержащие Cache
git rm -r --cached '*cache*'
git rm -r --cached '*logs*'

# Чтобы не заниматься этим каждый коммит, можно создать .gitignore файл, указать в нем
# ненужные файлы и при дальнейших коммитах, выбранные файлы будут игнорироваться

git status # проверить состояние коммита

# если все в порядке, можно делать коммит
git commit -m "Init"
```

Результат коммита:

```
[stanik@archlinux ~/.config]$ git commit -m "Init"
Auto packing the repository in background for optimum performance.
See "git help gc" for manual housekeeping.
[master 7ee9be1] Init
648 files changed, 88965 insertions(+), 115 deletions(-)
create mode 100644 .bashrc
create mode 100644 .gitconfig
create mode 100644 .vimrc
create mode 100644 GIMP/2.10/action-history
```

Теперь, были бы мы в 2021 году, осталось только сделать `git push`, заранее залогинившись в учетную запись и указав ссылку на обновляемый репозиторий. Но теперь эта процедура, я так понимаю в целях безопасности, делается только при помощи токена. Токен — это ключ от вашего репозитория. Он создается на определенный срок, выбираемый

пользователем, и должен обновляться вручную каждые выбранные n дней. Сгенерировать его можно в настройках.

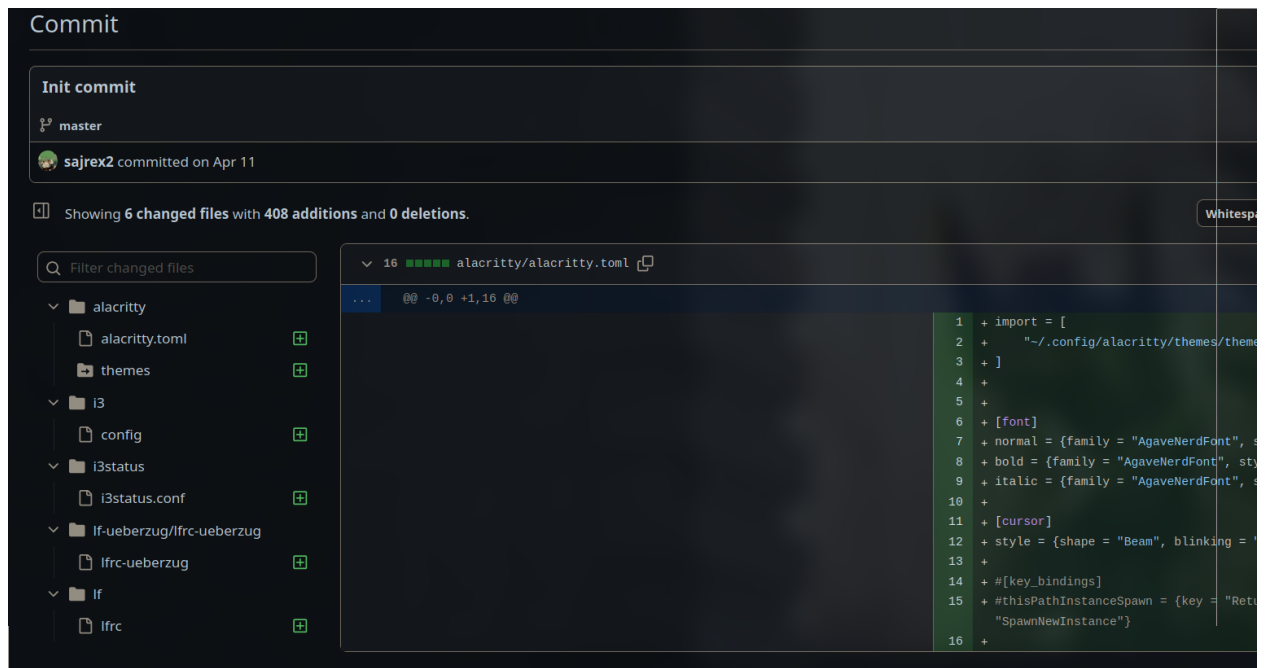
И финальная команда для загрузки в репозиторий будет выглядеть следующим образом:

```
git push https://TOKEN@github.com/username/reponame.git
```

в моем случае

```
git push https://ghp\_gC0jtSI13sHDbHtG#####@github.com/lxvevery1/dotfiles.git
```

## Изменения в ветке «master» моего репозитория dotfiles



Теперь мои конфиги лежат на надежных серверах GitHub! Если я что-нибудь сломаю, всегда смогу вернуться на свою рабочую версию.

**Вывод:** в ходе лабораторной работы освоил работу с Git CLI