Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

Факультет информационных технологий и программирования

Дисциплина «Введение в цифровую культуру и программирование»

Лабораторная работа №2

Изучение системы управления версиями Git.

Выполнил студент группы № M3109

Хлучин Георгий Владимирович.

Подпись:

Проверил:

Повышев Владислав Вячеславович

Санкт-Петербург

2021

Знакомство с Git.

1. Установка.
2. Нажимаем Win + r, в появившемся окне вводим cmd, чтобы вызвать командную строку.
3. В появившемся окне вводим git –version (это программа, используя которую можно управлять версиями проекта, сами версии можно хранить, включая в онлайн-хранилище GitHub.)
4. Создаем папку на рабочем столе(gitTest)
5. Заходим в Visual Studio Code и создаем какой-то проект.
6. Инициализируем репозиторий.
7. Или вместо этого заходим в командную строку 1) cd C:\Users\user\Desktop\GitTest (переходим в папку) 2) dir (открываем каталоги) 3) git init (инициализируем репозиторий) * с помощью команды cd мы можем перемещаться между каталогами.
8. Создаем хранилище на GitHub.
9. С помощь команд git config –global user.name и git config –global user.email вводим свои данные
10. Переходим в Visual Studio Code
11. Git init
12. Git commit -m “first commit”
13. Git remote add origin
14. Git push -u origin master
15. Поздравляю!!! Ваш код лежит в репозитории.
16. Как загрузить чужой репозиторий: 1) git clone (клонирование, после команды необходимо написать ссылку)
17. Добавим изменения в проект: 1) git status (узнаем статус); 2) git add . (можно в кавычках указать конкретный файл); 3) git status (проверяем статус); 4) git commit -m; 5) git status; 6) git push.
18. Разберемся с дополнительными командами в редакторе Visual Studio Code: 1) git pull (подгружает все изменения с облака); 2) git log (выводит коммиты)
19. Пробуем создавать собственные команды: 1) находим .gitconfig; 2)создаем [alias]
20. Рассмотрим, то как делаются отмены изменений и возврат к другому коммиту: 1) git checkout – название файла (отменяет все изменения); 2) git checkout . (отменяет все изменения во всех файлах); 3) git reset название файла (если была введена программа git add . , используется чтобы вытащить файл); 4) git reset –hard HEAD^1 (если был произведен коммит); 5)git reset –soft HEAD^1(чтобы вытащить изменения, но не удалять);
21. Создание и работа с ветками: 1)git branch (показывает ветки, а также ветку на которой находимся); 2) git branch -v (показывает ветку а также последний комит); 3) git branch название ветки которую хотим создать; 4) git checkout название ветки на которую хотим перейти; 5) git checkout -b название ветки которую хотим создать и сразу на нее перейти; 6) git branch -m новое название ветки (меняет название ветки); 7) git push –set-upstream origin название ветки (заливает ветку в хранилище); 8) git branch -d название ветки (удаляет ветку)
22. Создаем свои фрагменты кода

Команды Git.

1) git –stat
2) git log –stat

3) git log –pretty=format:”%an”
4) git log –pretty=format:”%an - %cn”
5) git log –pretty=format:”%an - %cn - %H”
6) git log –pretty=format:”%an - %cn - %ce”
7) git log –pretty=format:”%an - %s”
8) git tag – показывает версии тегов
9) git tag -a v0.1(версию можно указывать любую) -m (комментарий в кавычках) → это создание тега
10) git show – показывает теги
11) git tag -d название тега который я хочу убрать
12) git show название тега (показывает подробную информацию)
13) git sqash (это команда которая помогает уплотнить коммиты)
14) git submodule add(добавляет новый подмодуль)

**тег - это просто метка, в отличие от ветки