Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Дисциплина: Пользовательские интерфейсы информационных систем

Отчет по практическим занятиям

Гит

Студент гр. 210902Г.В. Ульянова

Преподаватель: К.И. Давыдович

Минск 2024

**Git**

Репозиторий Git— это хранилище, в котором расположен ваш проект и его история.

Вы можете думать о коммите как о снимке вашего проекта в определенный момент времени. Правда, коммит содержит только информацию об изменениях, которые были внесены в репозиторий с момента последнего коммита.

Ветка — это параллельная версия репозитория.

**Для начала работы необходимо ввести свое имя и электронную почту:**

1.git config –global user.name “my name”

2.git config –global user.email “email@com”

**Для создания нового репозитория необходимо его инициализировать с помощью:**

3.git init

**После этого добавляем файлы и коммитим изменения:**

*4.*git add my.js

5.git commit -m “new file*”*

**Для внесения изменений, добавленных в файлы**

*git add.*

**Для отмены данной команды:**

*git reset*.

**Но если изменения еще не проиндексированы, то отменить можно с помощью:**

*git checkout.*

**Для просмотра текущего состояния репозитория (к примеру, файлы, в которые были внесены изменения) используем:**

*git status*.

**Если мы хотим внести изменения нескольких файлов в один коммит, то либо используем git add по очереди:**

*git add 1.js*

*git add 2.js*

*git commit -m “2 files changed”*

**Либо используем, которая добавляет изменения из всех файлов репозитория**

*git add*

**Для получения списка внесенных изменений на репозитории используем:**

*git log*

**Если нам необходимо откатиться назад, то можем использовать:**

*git checkout <уникальный номер коммита>.*

**Для переключения между ветками и возвращения на последнюю версию можем использовать:**

*git switch <название ветки>.*

**Для быстрого переключения между версиями используются теги. Тег присваивается с помощью:**

*git tag <название тега>.*

**Просмотреть теги можно с помощью:**

*git tag.*

**Для отмены уже созданного коммита необходимо использовать команду:**

*git revert <номер коммита>*

**либо для самого последнего**

*git revert HEAD.*

**Для сброса историй изменений к определённому коммиту используем:**

*git reset –hard <номер коммита>*

***.* Стоит отметить, что не рекомендуется использовать такую команду на удаленных репозиториях.**

**Если мы хотим удалить тег с коммита:**

git tag -d <название тега>.

**Для внесения изменений в коммит:**

*git add my.js*

*git commit –amend -m “redacted commit”*

**Для создания новой ветки используем команду:**

*git switch -c <название>.* для переключения между ветками алгоритм такой же.

**Для просмотра изменений в конкретном коммите используется:**

git show.

Опция --all гарантирует, что мы видим все ветки, так как по умолчанию в логе показывается только текущая ветка

Опция --graph добавляет простое дерево коммитов, представленное в виде простых текстовых линий.

Если нам необходимо переместить в какую-либо директорию:

git mv my.js jses/my.js.

**Используется для объединения изменений из одной ветки в другую. Основная функция этой команды заключается в том, чтобы взять изменения из одной ветки и интегрировать их в текущую ветку**

*git merge main.*

**команда в Git, которая позволяет переносить или "переписывать" последовательность коммитов с одной ветки на другую. Она используется для упрощения истории проекта и интеграции изменений.Команда используется для переноса изменений на ветку.**

*git rebase*

**Для клонирования репозитория используем команду:**

*git clone <название репозитория>.*

**Для просмотра доступных удаленных репозиториев используем:**

git remote.

**Для обозначения удаленных репозиториев используем**

origin.

**Для просмотра веток, доступных на локальном репозитории:**

*git branch*,

**а для удаленных веток**

*git branch -a*.

**используется для загрузки изменений из удалённого репозитория в ваш локальный репозиторий, не изменяя при этом вашу текущую рабочую ветку.**

*git fetch*.

**используется для отправки ваших локальных изменений в удалённый репозиторий.**

git push

**Загружает последние изменения из удалённого репозитория и автоматически объединяя их с локальной веткой**

*git pull*