Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

ОТЧЕТ

по выполнению практической занятия

на тему

**GIT How To**

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Студент гр. 210902 Писаренко Я.Д. |
| Проверил: |  | Давыдович К.И. |

Минск

2024

**Терминология.**

**Git** — это **система контроля версий**, которая позволяет отслеживать изменения в вашем коде с течением времени.

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

**Коммит** — это небольшой кусочек истории репозитория, основанный на предыдущем коммите. Все они связаны между собой в цепочку, формируя историю изменений вашего проекта.

**Ветка** — это параллельная версия репозитория. Ветки позволяют вам работать над отдельными функциями вашего проекта, не влияя на основную версию.

**Чистый репозиторий** — это репозиторий, не имеющий рабочей директории. Он содержит только директорию .git, в которой Git хранит все свои внутренние данные. Основное предназначение таких репозиториев — быть центральным хранилищем, в которое разработчики могут отправлять и из которого могут получать данные.

**Общий репозиторий** – это репозиторий, получающий отправленные нами изменения.

***Функции git:***

git init – команда, которая позволяет создать git-репозиторий из указанной директории.

git status – команда, позволяющая проверить текущее состояние репозитория.

git branch – команда, позволяющая переименовать ветку.

git add – команда, позволяющая добавить изменения в репозиторий.

git checkout – команда, позволяющая отменить изменения в репозитории.

git reset – команда, позволяющая снять индексацию изменений, сбрасывать область подготовки к HEAD.

HEAD — это указатель на текущую ветку, которая, в свою очередь, является указателем на последний коммит, сделанный в этой ветке.

git add file – команда, которая говорит git, что надо отметить текущее состояние файла, коммит которого будет произведен позже.

git log – команда, позволяющая получить список произведенных изменений. Его формат зависит от нескольких деталей:

* --pretty="..." — определяет формат вывода.
* %h — укороченный хеш коммита.
* %ad — дата коммита.
* | — просто визуальный разделитель.
* %s — комментарий.
* %d — дополнения коммита («головы» веток или теги).
* %an — имя автора.
* --date=short — сохраняет формат даты коротким и симпатичным.

git switch – команда, необходимая для переключения между ветками.

git tag – команда, позволяющая создать тег. Теги нужны для того, чтобы переходить к определенной версии проекта по названию. С помощью этой команда также можно просмотреть, какие теги доступны. Просмотр тегов доступен с помощью git log main –all, где main – название ветки по умолчанию. Метка HEAD показывает коммит, на который вы переключились.

git checkout – команда, позволяющая вернуться к версии файла, которая уже была закоммичена.

git revert – коммит, удаляющий изменения, сохраненные нежелательным коммитом.

git show – команда, необходимая для просмотра изменений в конкретном коммите.

git mv – команда, которая гарантирует, что перемещение будет записано в истории Git как перемещение.

git merge – команда, которая дает возможность слияния двух веток.

git merge –abort – команда, которая отменяет слияние двух веток.

git rebase – команда, позволяющая перебазированиею.

Лучше использовать команду rebase для краткосрочных, локальных веток и команду merge для веток в публичном репозитории.

git clone – команда, позволяющая создавать клон репозитория.

Если изменения происходят в оригинальном репозитории, необходимо подтягивать и сливать эти изменения в клонированный репозиторий.

git fetch – команда, которая будет подтягивать новые коммиты из удаленного репозитория, но не будет сливать их с наработками в локальных ветках.

git pull – команда, которая подтягивает и сливает изменения из удаленной ветки в текущую одним вызовом.

**Удаление тега:**

git tag -d oops

git log –all

**Внесение изменений в коммиты:**

Добавление коммита:

git add hello.html

git commit -m "Added copyright statement"

git log

Изменение коммита:

git add hello.html

git commit --amend -m "Added copyright statement with email"

**Создание ветки:**

touch style.css – добавление файла стилей

git add style.css

git commit -m "Added css stylesheet"

Далее необходимо добавить CSS в html-файл и добавить коммит:

git add hello.html

git commit -m "Included stylesheet into hello.html"

**Просмотр текущих веток**

git log --all –graph

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.

**Слияние веток**

git switch style

git merge main

git log --all –graph

При слиянии двух веток Git пытается перенести изменения из одной ветки в другую. Если в обеих ветках была изменена одна и та же часть файла, Git может не справиться с автоматическим слиянием изменений. В этом случае Git сообщит о конфликте и попросит разрешить его вручную.

Чтобы разрешить конфликт, нужно отредактировать файл до состояния, которое нас устраивает, и затем закоммитить его как обычно.

**Добавление локальной ветки, которая отслеживает удаленную ветку**

git branch --track style origin/style

git branch -a

git log --max-count=2

**Создание чистого репозитория**

cd ..

git clone --bare work work.git

ls work.git

**Добавление удаленного репозитория**

cd work

git remote add shared ../work.git

**Отправка изменений**

Закоммиченное изменение в файле README:

git switch main

git add README

git commit -m "Added shared comment to readme"

Отправка изменений в общий репозиторий:

git push shared main

**Подтягивание общих изменений**

git remote add shared ../work.git

git branch --track shared main

git pull shared main

cat README

**Размещение Git-репозиториев**

1. *Запуск Git-сервера*

# (From the "repositories" directory)

git daemon --verbose --export-all --base-path=.

В отдельном окне терминала перейдите в вашу директорию repositories:

# (From the "repositories" directory)

git clone git://localhost/work.git network\_work

cd network\_work

ls

1. *Отправка изменений в Git Daemon*

Если вы хотите разрешить отправку изменений (push) в репозиторий Git Daemon, добавьте метку --enable=receive-pack к команде git daemon