GIT

About

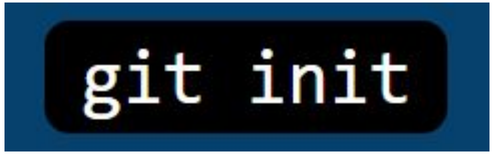
- ❏ Version control - is a **system that records changes to a file** or set of files over time so that **you can recall specific versions later**.
- ❏ It allows you to **revert files back** to a previous state, **revert the entire project** back to a previous state, **compare changes over time**, see who last modified something that might be causing a problem, who introduced an issue and when, and more.

Create a new repository

create new directory

open it

perform



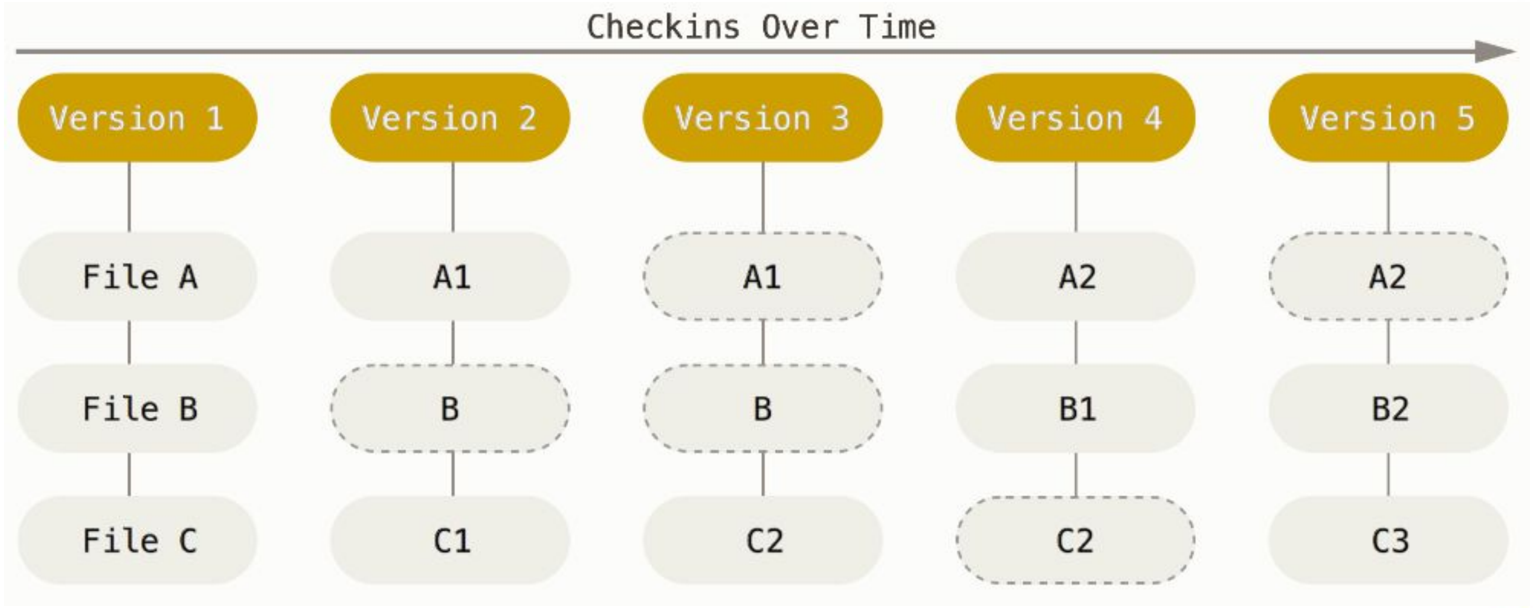
```
git init
```

Checkout the repository

create a **working copy of a local repository** by running the command

```
git clone /path/to/repository
```

In order to understand how files are stored

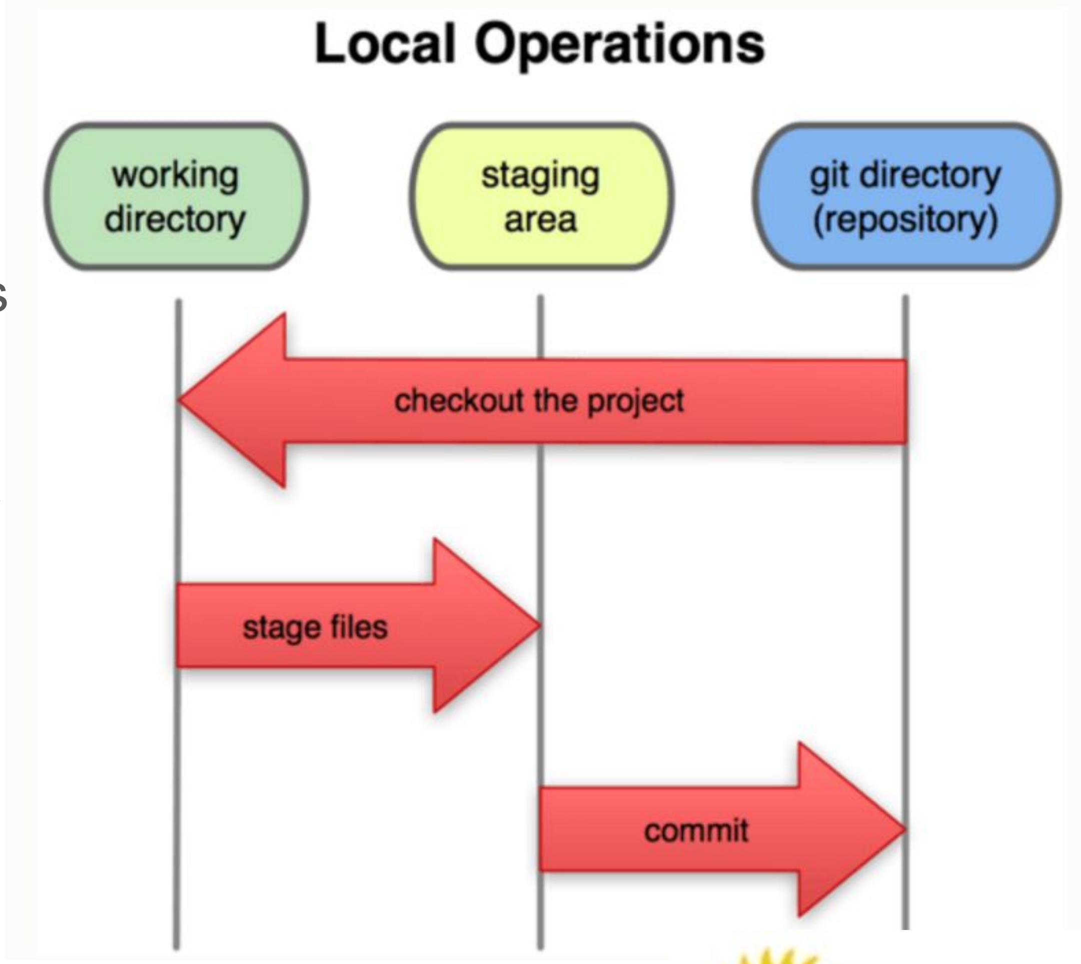


File states

git directory - commit changes

staging area - changed files included in next commit, but still uncommited

working directory - changed uncommited files



Standard GIT workflow

change files in the **working directory**

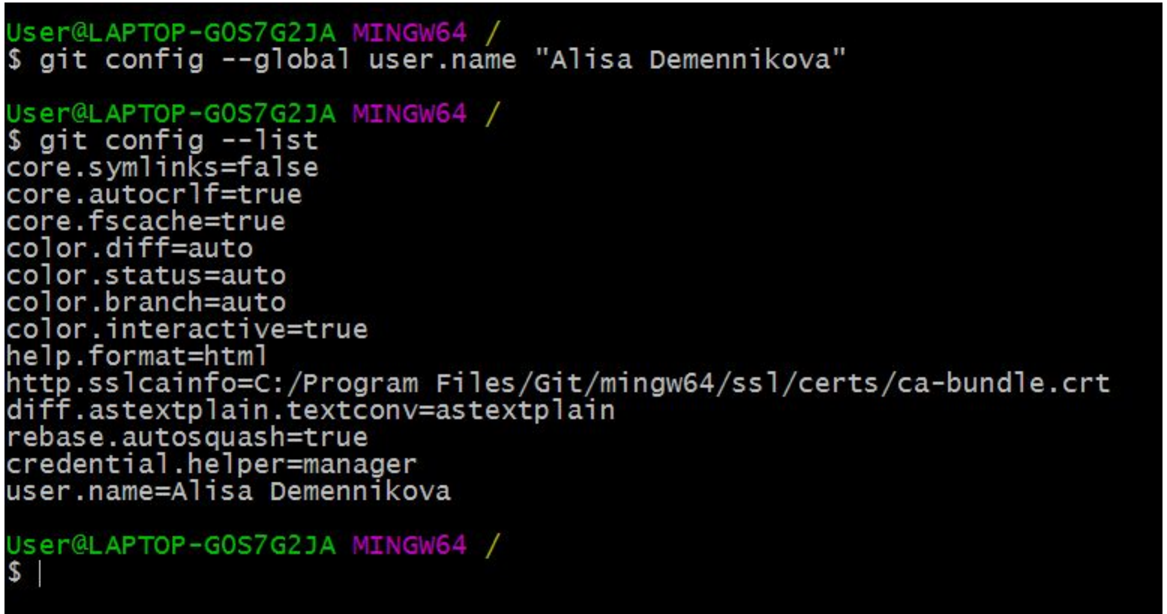
prepares the files, adding snapshots in their **staging area**

make a commit, which takes the prepared files from the index, and puts them in a **Git directory** for permanent storage

GIT configure

***git config --global user.name** "USER_NAME"*

git config --global user.email email@example.com



```
User@LAPTOP-G0S7G2JA MINGW64 /
$ git config --global user.name "Alisa Demennikova"

User@LAPTOP-G0S7G2JA MINGW64 /
$ git config --list
core.symlinks=false
core.autocrlf=true
core.fscache=true
color.diff=auto
color.status=auto
color.branch=auto
color.interactive=true
help.format=html
http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt
diff.astextplain.textconv=astextplain
rebase.autosquash=true
credential.helper=manager
user.name=Alisa Demennikova

User@LAPTOP-G0S7G2JA MINGW64 /
$ |
```

verify all settings:
git config --list
verify settings by key value:
git config {KEY}
git config user.name

Add & Commit

You can **propose changes** (add it to the Index - Staging area) using

```
git add <filename>
```

```
git add *
```

To actually **commit these changes** use

```
git commit -m "Commit message"
```

Now the file is committed to the HEAD, but not in your remote repository yet.

Pushing changes

Your changes are now in the **HEAD of your local working copy**. To send those changes to your remote repository, execute

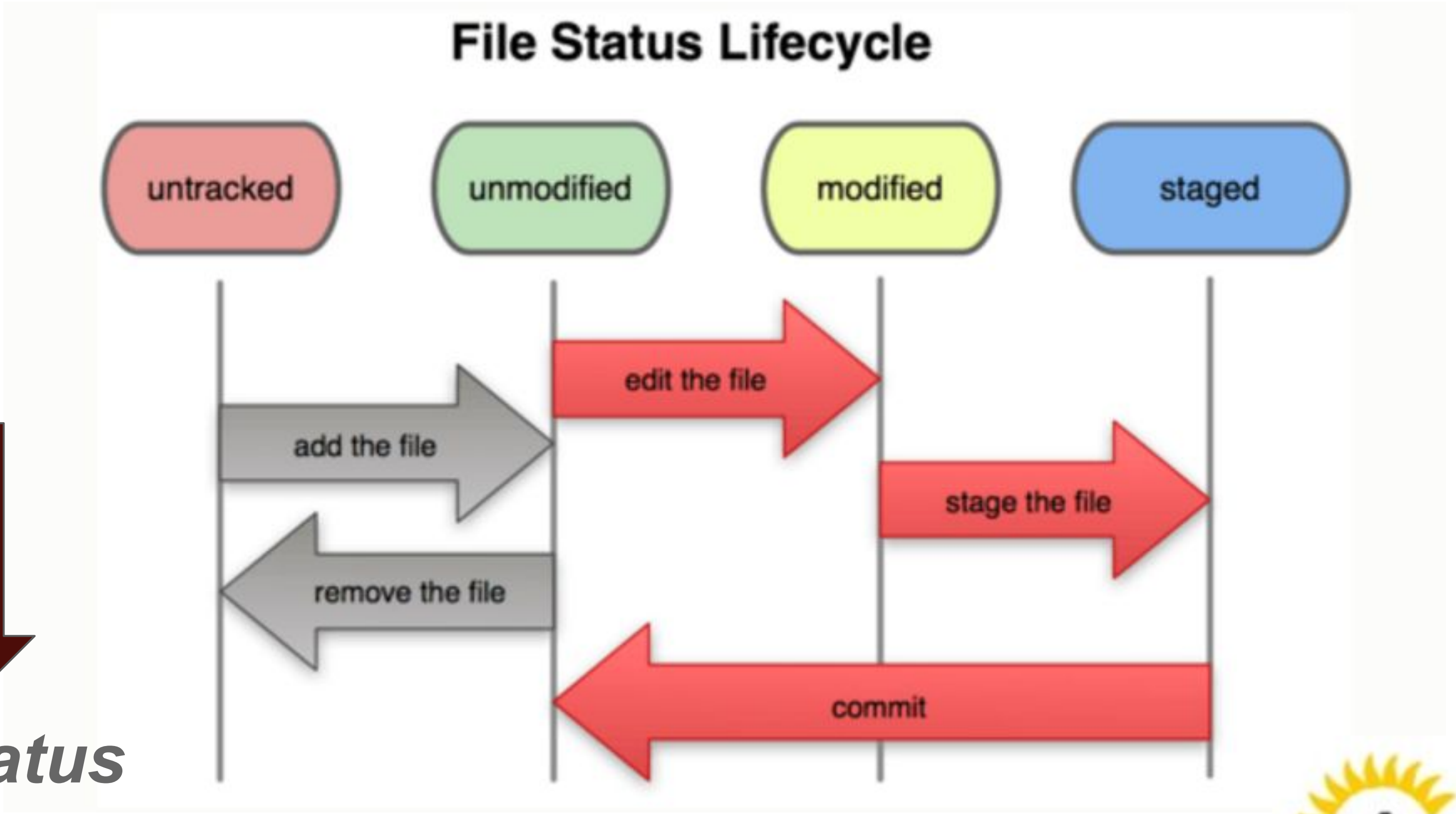
```
git push origin master
```

If you have **not cloned an existing repository** and **want to connect your repository** to a remote server, you need to add it with

```
git remote add origin <server>
```

Now you are able to push your changes to the selected remote server

Record changes to the repository



git status

Delete files

To remove a file from Git, you have to **remove it from your tracked files** (staging area)

```
git rm
```

you may want to **keep the file on your hard drive but not have Git track** it anymore

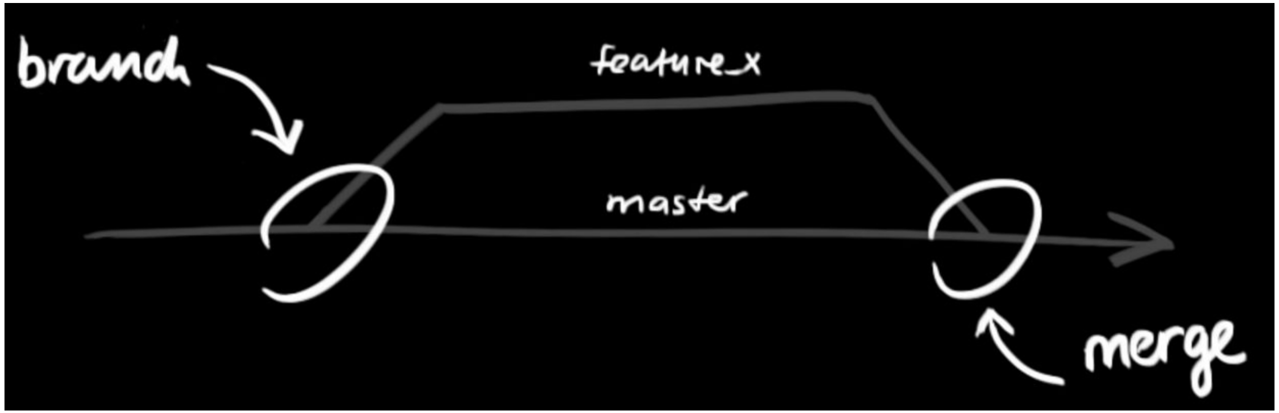
```
git rm --cached FILE_NAME
```

Branching

Branches are used to **develop features isolated from each other**.

The ***master*** branch is the **"default" branch** when you create a repository.

Use other branches for development and merge them back to the master branch upon completion.



Branching

create a new branch named "feature_x" and switch to it using

```
git checkout -b feature_x
```

switch back to master

```
git checkout master
```

and delete the branch again

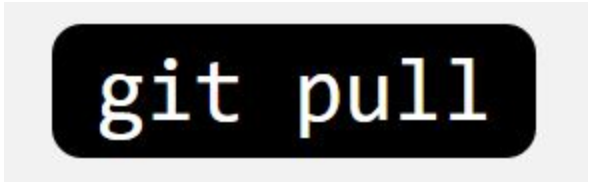
```
git branch -d feature_x
```

a branch is **not available to others** unless you **push the branch** to your remote repository

```
git push origin <branch>
```

Update changes from repository

to **update** your local repository **to the newest commit**, execute



```
git pull
```

Log

you can study repository history using

```
git log
```

to see only the commits of a certain author

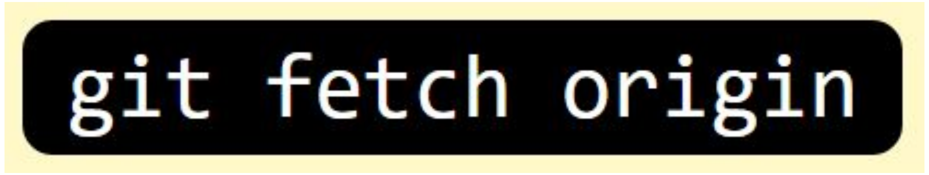
```
git log --author=bob
```

To see a very compressed log where each commit is one line

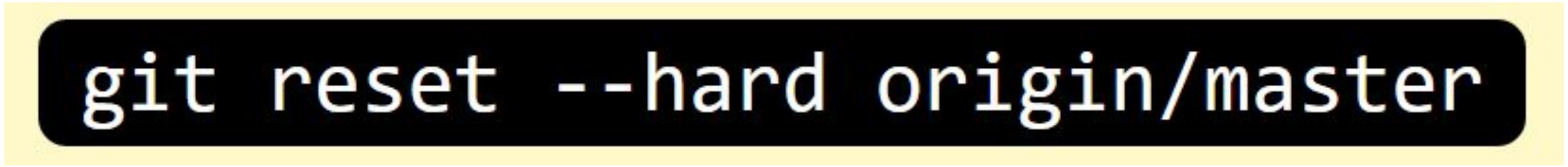
```
git log --pretty=oneline
```

Drop all local changes

If you instead want to **drop all your local changes and commits**, fetch the latest history from the server and point your local master branch at it like this



```
git fetch origin
```



```
git reset --hard origin/master
```

Task

1. Create new repository in any project on your computer. Pull this project to you GitHub.

2. Checkout the repository https://github.com/AliceIgorevna/TestProj.git, add changes to project (any you want), push this changes, do any changes again, commit them and then cancel your commit.

3. Read tutorial https://git-scm.com/book/ru/v1/%D0%92%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5-%D0%9E-%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D0%B5-%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D0%B9