Programmable Web Project

Exercise 0

Introduction to GIT

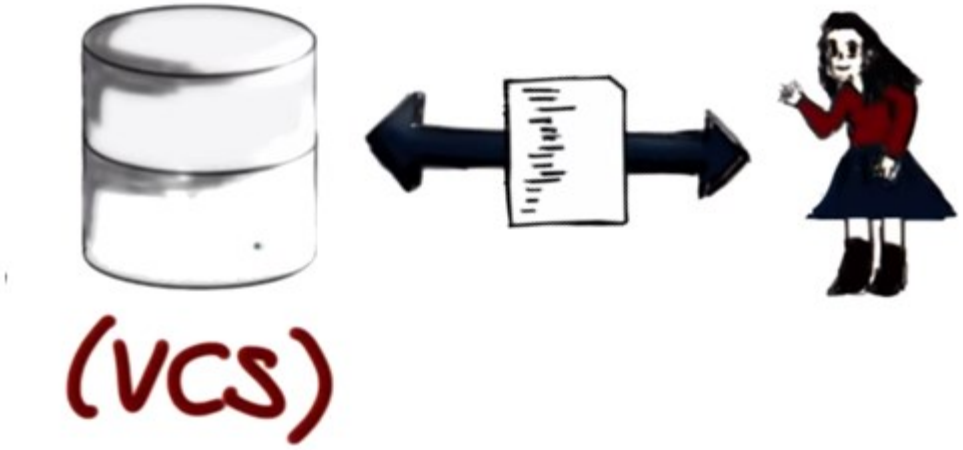
Iván Sánchez Milara

Version Control System. Definition.

Software that:

- **Records changes to a file or set of files**
- **Permits recalling older revisions of them**

- **Mainly source code and documents.**
- NOT BINARY FILES in general.



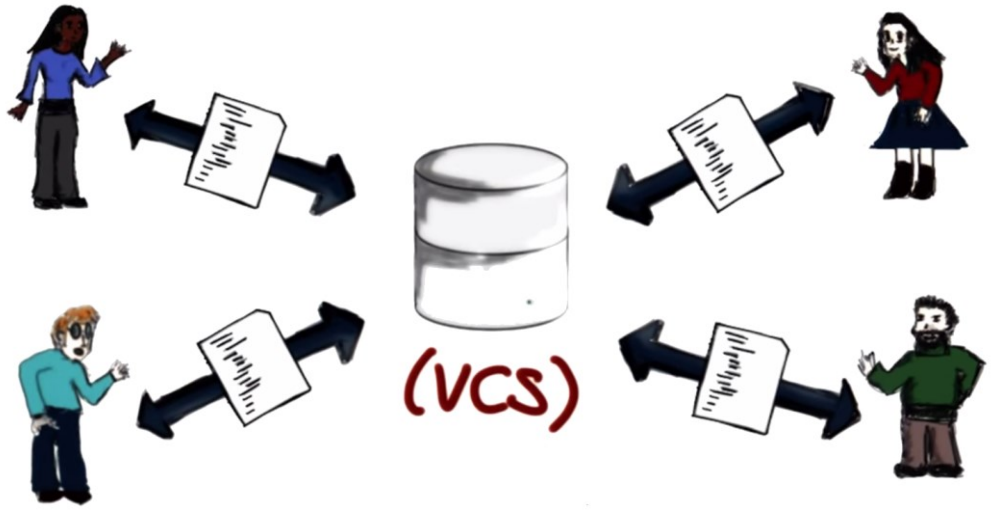
Version Control System. Definition.

Enables the cooperation among multiple

- Share files
- Share old version of the files

Other names:

- Revision Control
- Source Control

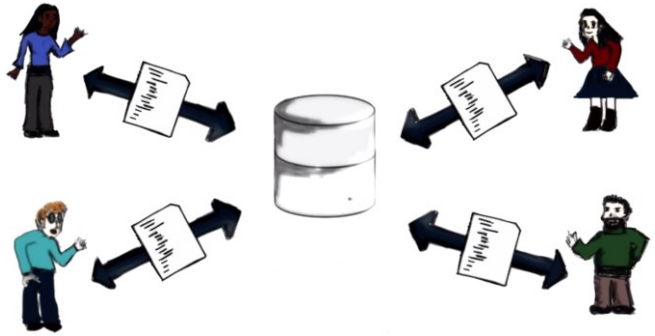


Why version control system?

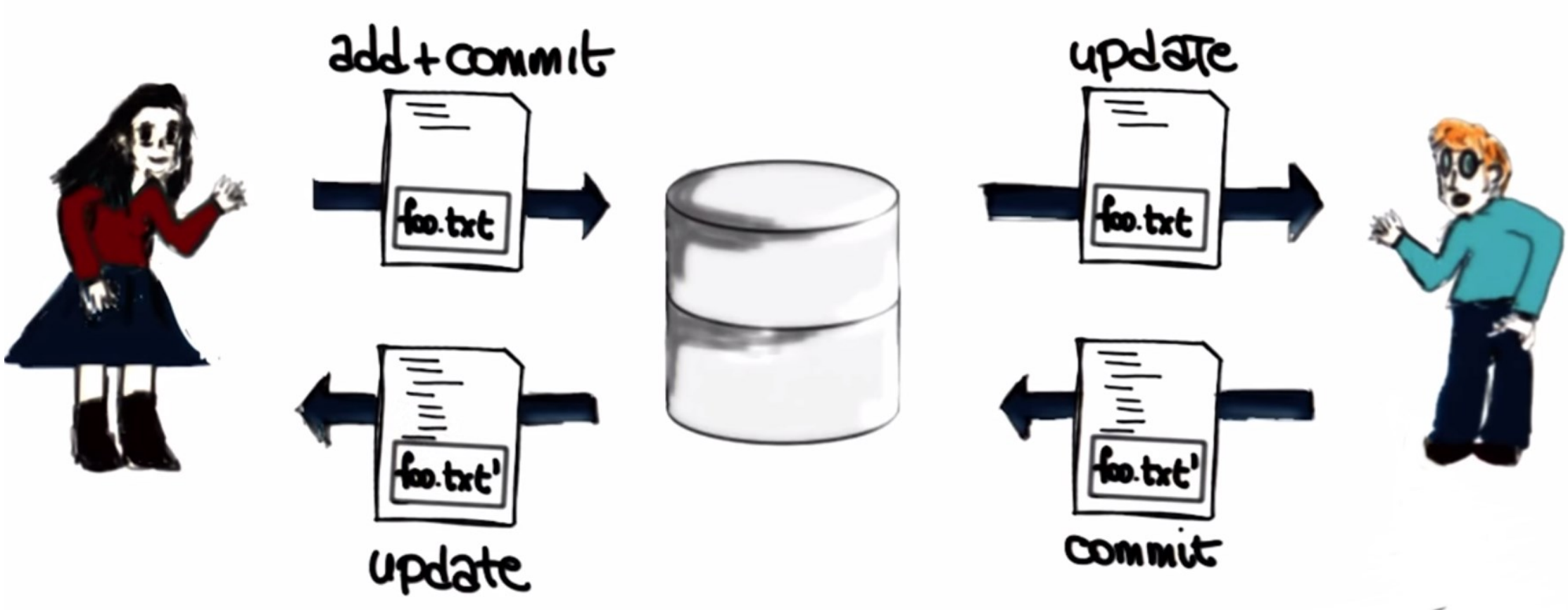
- Archive different versions of the same file
- Mantain historical information of a file
- Recover from accidental deletions or edits

- Conserve disk space

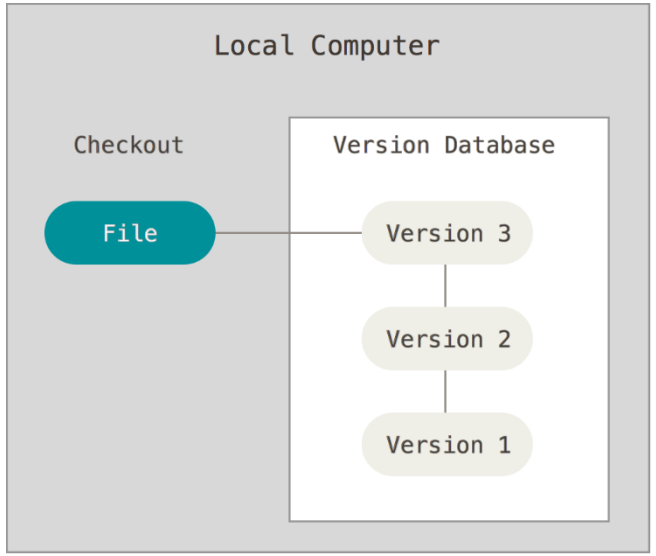
- Enforce discipline
- Enable collaboration



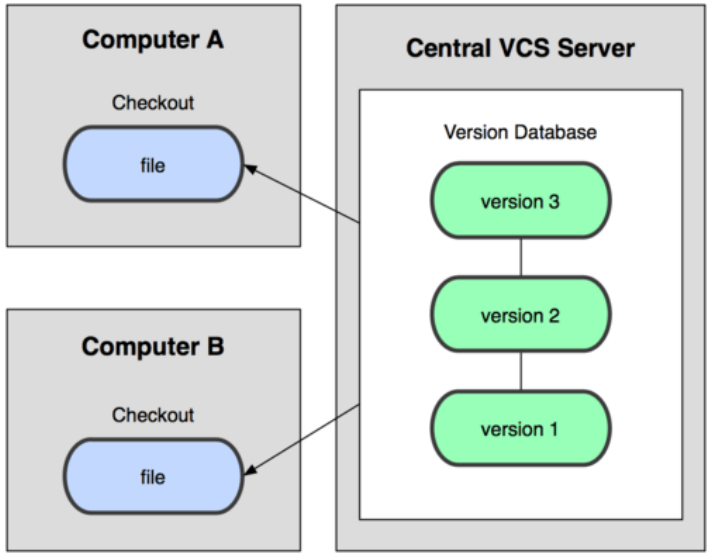
Basic workflow in a VCS.



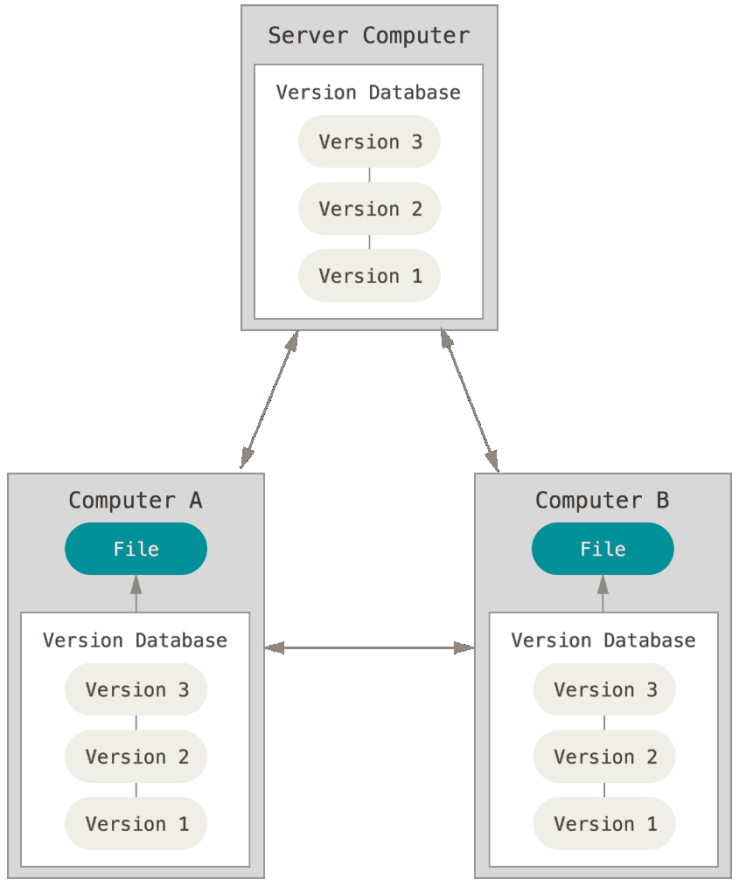
Types of VCS



Local VCS



Centralized VCS



Distributed VCS

https://git-scm.com/book/en/v2/

GIT

Free, open source, distributed and multiple platform Version Control system

Characteristics:

- Fast
- Simple design
- Paralell work with multiple branches
- Able to handle efficiently large projects

Flexible, permits multiple workflows

https://git-scm.com/

GIT. BASIC TERMINOLOGY (I)

- **Repository:**

- Contains the complete history of the project from its inception.
- Consists on multiple ”commits”

- **Commit:**

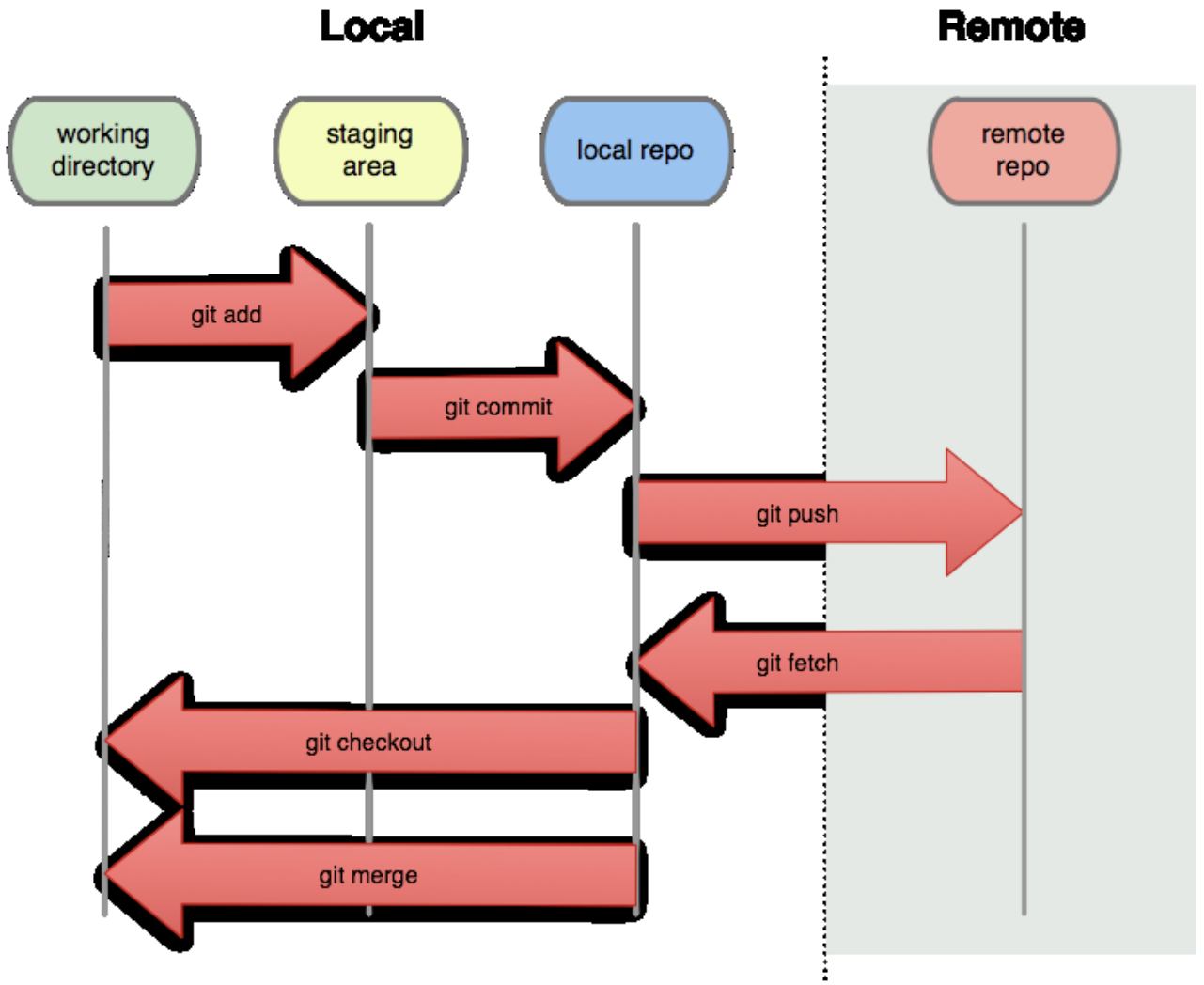
- Snapshot of the project contain (files and directories) at certain time.
- In addition it contains following metadata:
  - Author identification
  - Commiter identification
  - Commit message
  - List of zero or more ”parent – commits”(preceding state of the project)

GIT workflow and lifecycle(I)

Files can be in one of the following states:

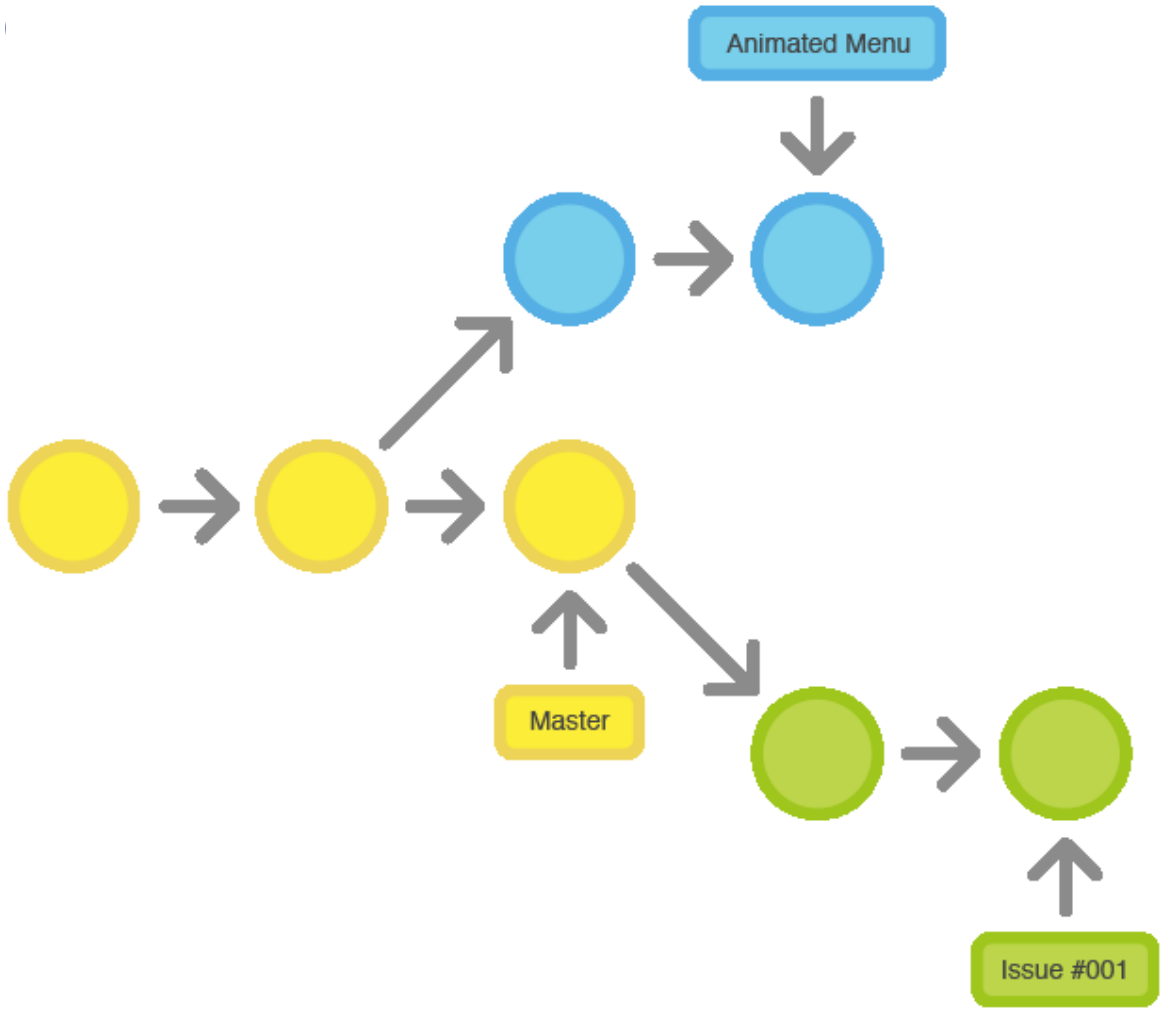
- **Commited**
- **Modified**
- **Staged** (ready to be commited)

GIT workflow and lifecycle(II)



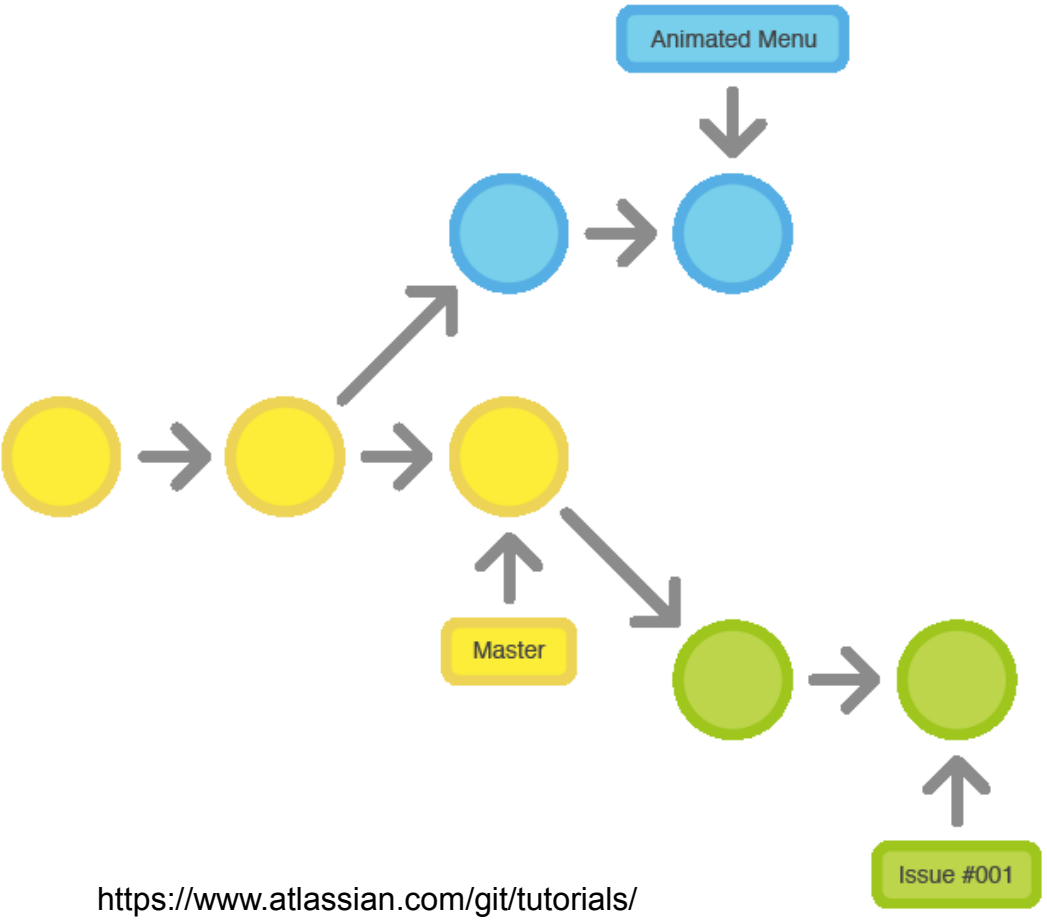
https://git-scm.com/book/en/v2/

GIT. BASIC termin



https://www.atlassian.com/git/tutorials/

GIT. BASIC terminology (III)



https://www.atlassian.com/git/tutorials/

- **Branches: Alternative path of software development that allows developing features isolated from each other**

- When a branch is created a commit has two different children which follow alternative paths
- Branch name is the last commit of the branch
  - **Master:** Main branch of development

Git. Commands (I)

CONFIGURE TOOLING

Configure user information for all local repositories

`$ git config --global user.name "[name]"`
Sets the name you want attached to your commit transactions
`$ git config --global user.email "[email address]"`
Sets the email you want attached to your commit transactions
`$ git config --global color.ui auto`
Enables helpful colorization of command line output

CREATE REPOSITORIES

Start a new repository or obtain one from an existing URL

`$ git init [project-name]`
Creates a new local repository with the specified name
`$ git clone [url]`
Downloads a project and its entire version history

MAKE CHANGES

Review edits and craft a commit transaction

`$ git status`
Lists all new or modified files to be committed
`$ git diff`
Shows file differences not yet staged
`$ git add [file]`
Snapshots the file in preparation for versioning
`$ git diff --staged`
Shows file differences between staging and the last file version
`$ git reset [file]`
Unstages the file, but preserve its contents
`$ git commit -m "[descriptive message]"`
Records file snapshots permanently in version history

https://training.github.com/kit/downloads/github-git-cheat-sheet.pdf

Git. Commands (II)

GROUP CHANGES

Name a series of commits and combine completed efforts

$ git branch
Lists all local branches in the current repository
$ git branch [branch-name]
Creates a new branch
$ git checkout [branch-name]
Switches to the specified branch and updates the working directory
$ git merge [branch]
Combines the specified branch's history into the current branch
$ git branch -d [branch-name]
Deletes the specified branch

REVIEW HISTORY

Browse and inspect the evolution of project files

$ git log
Lists version history for the current branch
$ git log --follow [file]
Lists version history for a file, including renames
$ git diff [first-branch]...[second-branch]
Shows content differences between two branches
$ git show [commit]
Outputs metadata and content changes of the specified commit

https://training.github.com/kit/downloads/github-git-cheat-sheet.pdf

Git. Commands (III)

SYNCHRONIZE CHANGES

Register a repository bookmark and exchange version history

$ git fetch [bookmark]
Downloads all history from the repository bookmark
$ git merge [bookmark]/[branch]
Combines bookmark's branch into current local branch
$ git push [alias] [branch]
Uploads all local branch commits to GitHub
$ git pull
Downloads bookmark history and incorporates changes

https://training.github.com/kit/downloads/github-git-cheat-sheet.pdf

Conflicts resolution

```
the number of planets are
<<<<<<< HEAD
nine
=======
eight
>>>>>>> branch-a
```

- In branch-a, you wrote the word "nine," but your colleague wrote "eight."
- Git automatically adds *conflict markers* to the affected areas.

GIT. COMMANDS (IV)

If at some point you do not know how to use a command, type:

git help <command>

Code hosting sites

GitHub

https://github.com/

Bitbucket

https://bitbucket.org/

GitLab

https://about.gitlab.com/

Material

- **Git webpage:**
  - https://git-scm.com/
- **Courses in Udacity:**
  - https://www.udacity.com/course/viewer#!/c-ud805/l-3666138591/m-575808545
    - Majority of pictures in the presentation taken from here.
  - https://www.udacity.com/course/how-to-use-git-and-github--ud775
- **Tutorials:**
  - https://www.atlassian.com/git/tutorials/
  - https://www.atlassian.com/git/tutorials/comparing-workflows/
  - http://rogerdudler.github.io/git-guide/
- **Books:**
  - http://chimera.labs.oreilly.com/books/1230000000561
  - https://git-scm.com/book/en/v2

Guided exercise

https://atlassian.cs-aware.eu/confluence/display/5OTCSE/Introduction+to+Git

Instructions to setup your Gitlab/Github environment

https://lovelace.oulu.fi/ohjelmoitava-web/programmable-web-project-spring-2020/pwp-tutorial-setting-up-the-project-work-environment-in-git/