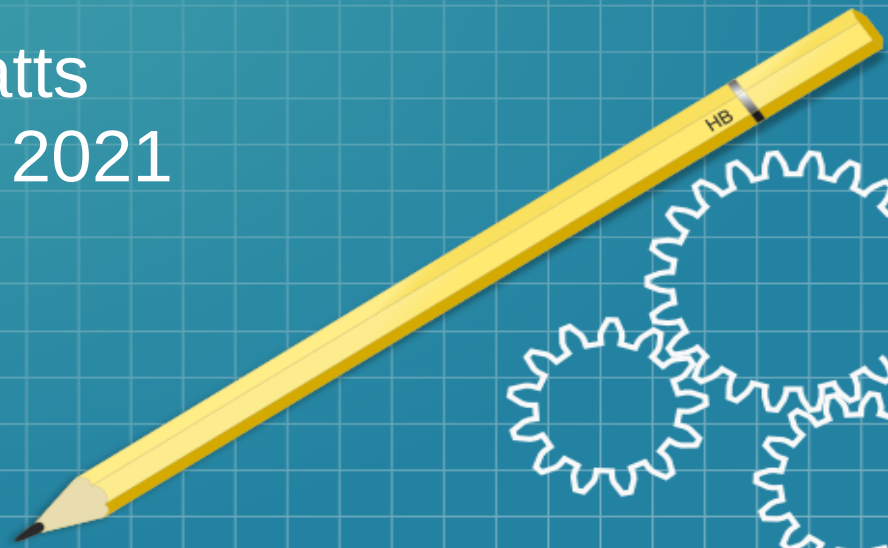
What is version control and Git?

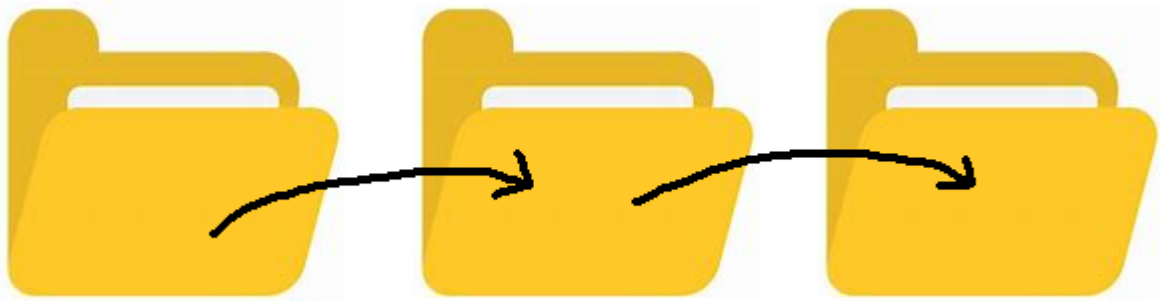
Joanna Watts

24th February 2021



Version control?

- The name of any system adopted aiming to keep a record of changes for a set of files (and occasionally a single file) over time.



Why?

- These records can be used later for reverting back to a previous working version if something should go wrong.
- Files corrupted
- Bug introduced
- Piece of work missing
- See who made changes
- Dates changes were made

I need special software?

- Copying the files manually into dated folders (weekly or after each significant change).
- Error-prone.
- Accidentally copy files into the wrong folder.
- Overwrite important files.
- Copy the wrong files or simply forgetting to do it in the first place.

Three types of VCSs

- VCSs (version control systems).

1. Local
2. Centralised
3. Distributed

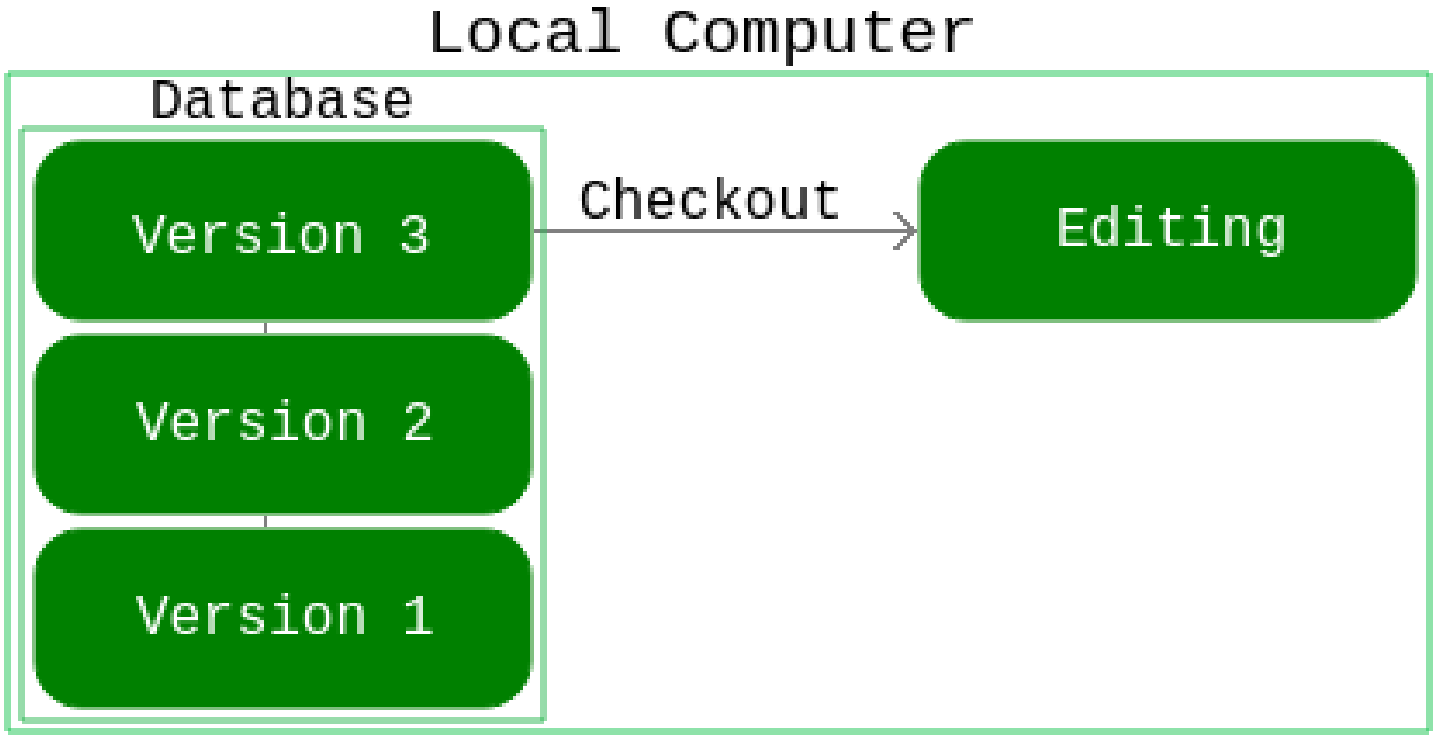
- Advantages and disadvantages of each.

Local VCSs

- Record of the file changes kept on local system.
- Database of patch sets (differences between files).
- Recreate any file for any date by adding up the patches.
- Only one person has access.
- Backups extremely important.

Local VCSs

- Example – RCS (Revision Control System)

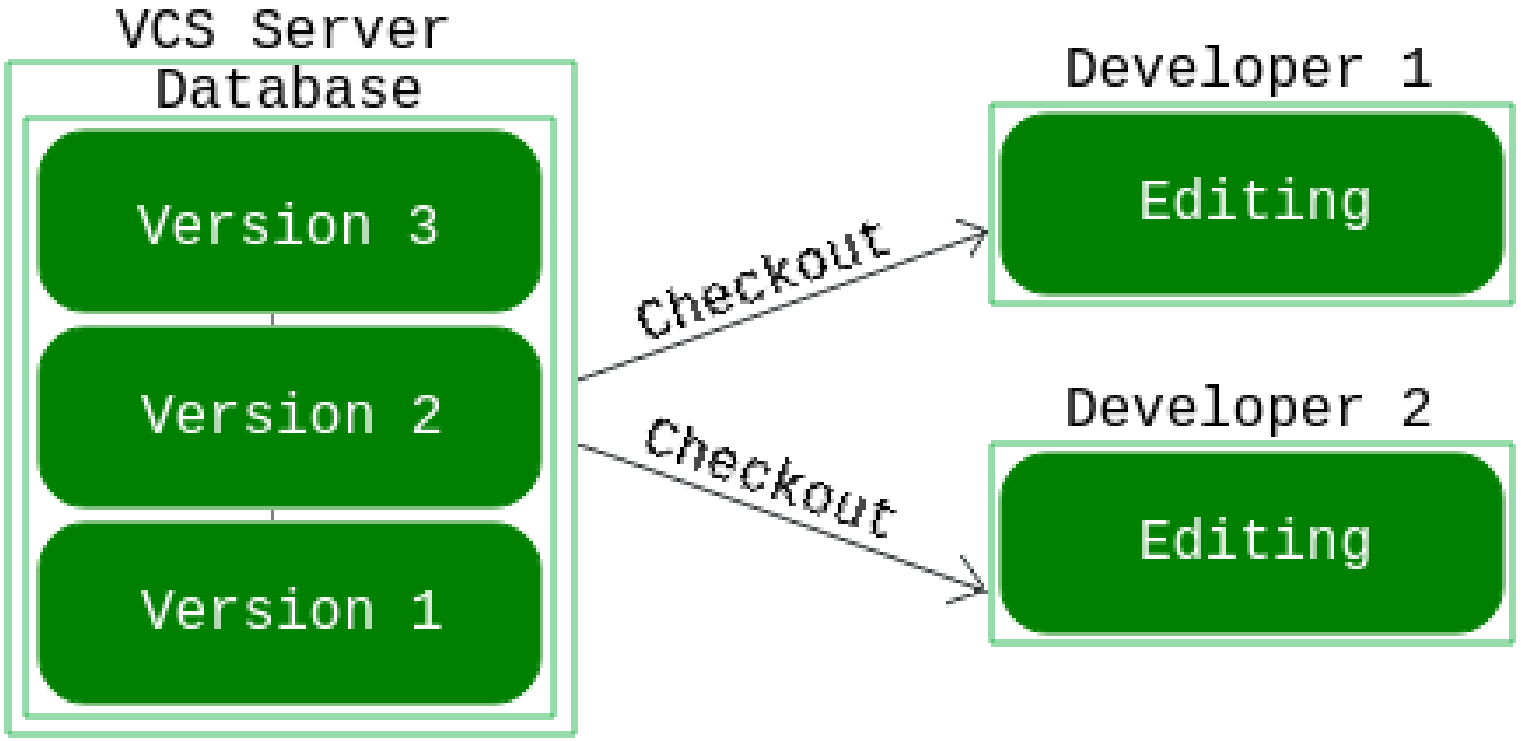


Centralised VCSs (CVCSs)

- Store files and their history on separate server.
- Multiple developers have access to the files.
- Admin modification rules.
- Records who did what & when.
- Know what people are working on.
- Rely on one server.
- Backups necessary.
- File conflicts → resolving and/or file locking required.

Centralised VCSs (CVCSs)

- Examples – Subversion, Perforce & CVS.

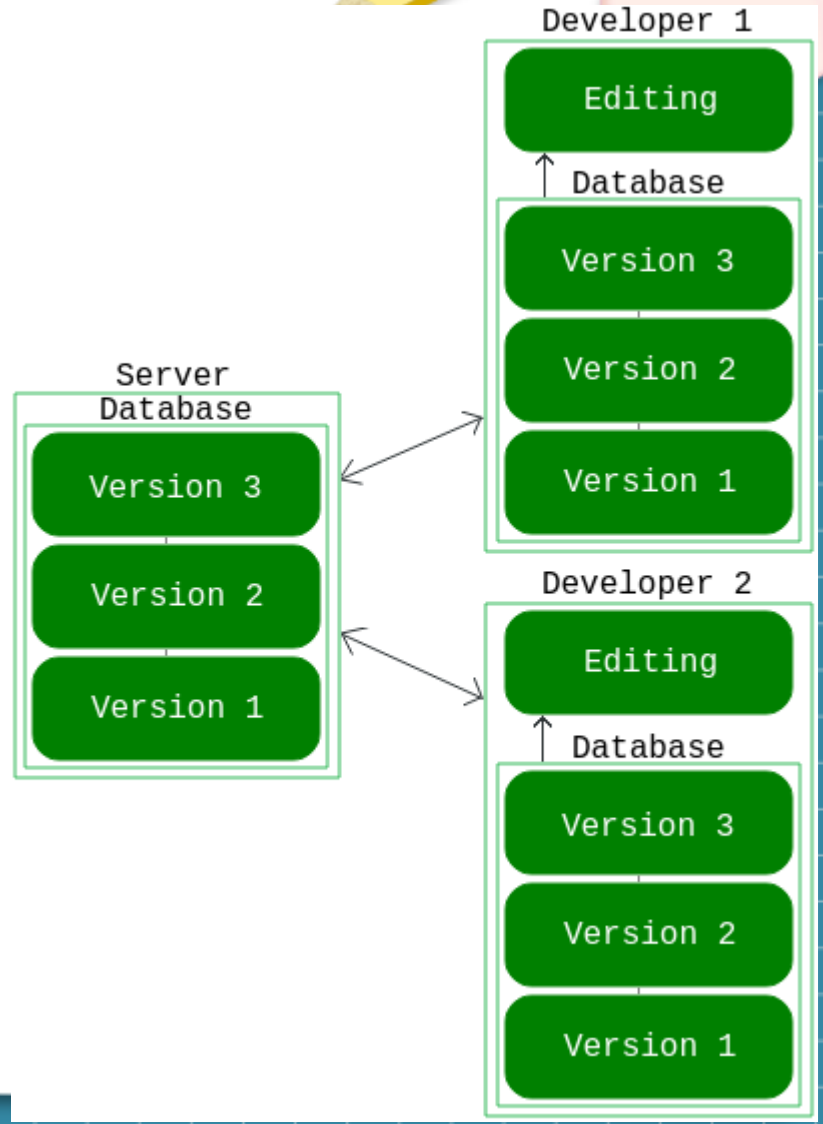


Distributed VCSs (DVCSs)

- Developers checkout latest version as well as history.
- Secure → backups kept on local machines.
- Standard practice.
- Fast → everything local.
- Offline work.
- Different workflows possible.

Distributed VCSs (DVCSs)

- Examples – Mercurial, Bazaar, Darcs & Git.



Git

- Since 2005.
- Free, open-source VCS.
- Efficient.
- Easy branching.
- “Snapshots” taken on each commit.
- Checksum for each file → SHA-1 hash.
- Mostly used via command line.
- Difficult to delete data → secure.

Git database

- Underlined = modified file.

Version 1	Version 2	Version 3	Version 4	Version 5	Version 6
File x	x 1	x 1	x 2	x 2	x 2
File y	y	y	y	y	y 1
File z	z	z 1	z 2	z 3	z 3

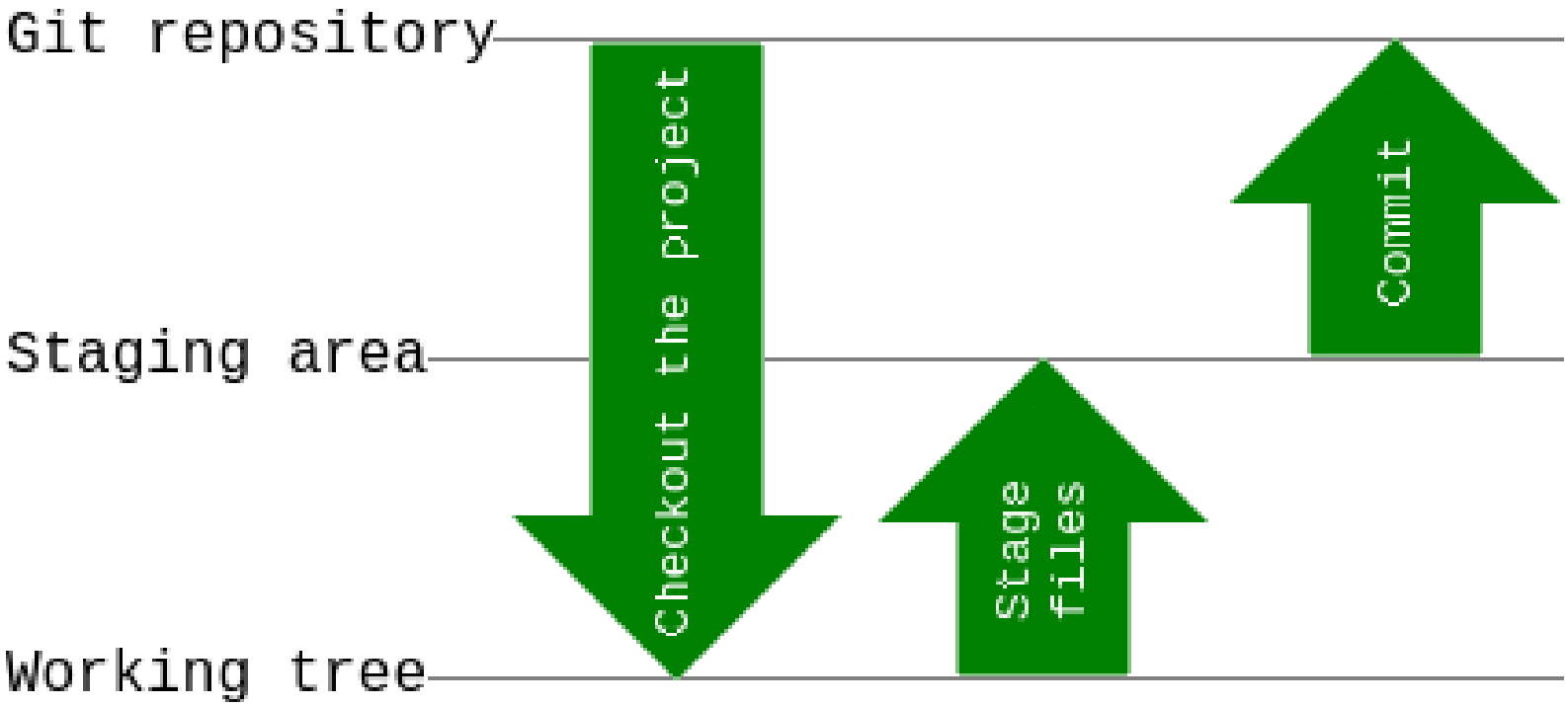
How Git works

Three states:

1. Modified – Local file changed
2. Staged – File changed has been marked to go into the next commit (which will create a new version)
3. Committed – Data stored in the Git repository

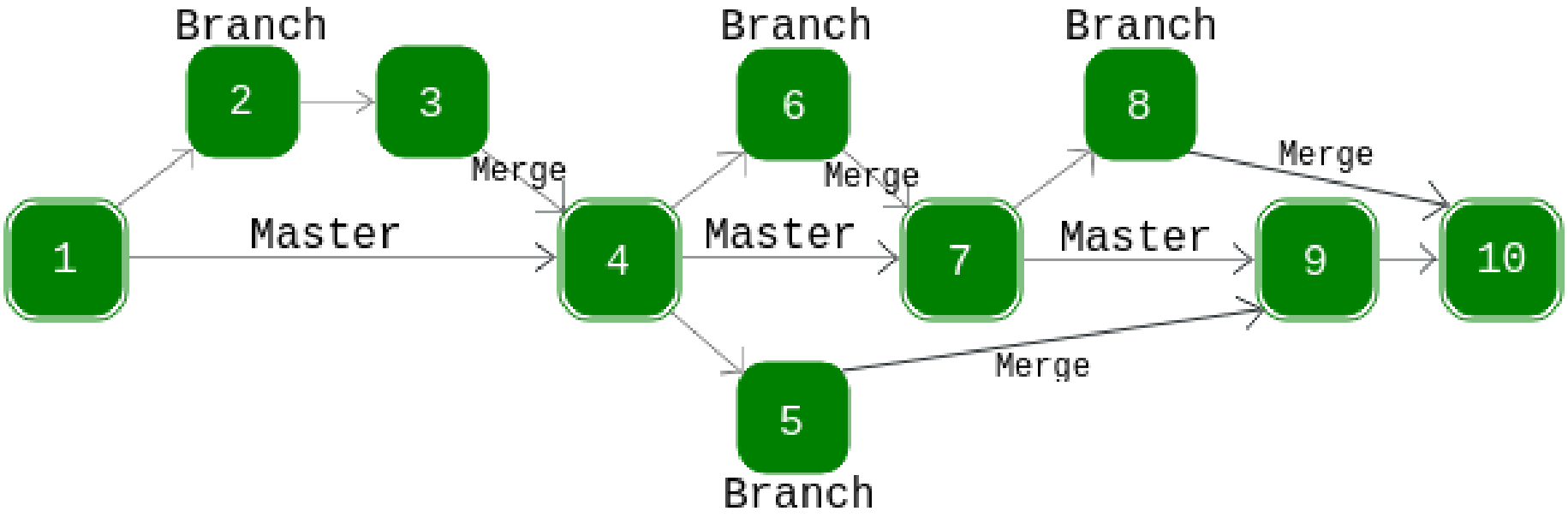
Git workflow

- Three sections of a Git project: working tree, staging area and Git repository.



Branching

- New branch → new copy of latest version created.
- Name them e.g. BUGFIX806, HOTFIX023 → keep track should you need to switch between multiple tasks.



What do commands look like?

`git init <directory>` Creates a Git repository in a specified directory

`git add <directory/file>` Stage all changes in the stated file or directory ready for the next commit

`git status` List which files are modified, staged or untracked (files inside a project that have been configured with Git, but are not part of the Git repository. Use “git add” to add them to the Git repository.)

`git diff` Show the unstaged differences between your modified files and your working directory

`git branch <branchName>` Create a new branch

`git checkout <branchName>` Checkout an existing branch

`git merge <branchName>` Merge the specified branch into the current branch

`git pull <remote>` Get the remote’s copy of the branch you are in and merge it with your local one

`git push` Pushes your current branch to the remote repository. If it does not exist on the remote yet, one will be created

What is GitHub?

- Sign up and store your Git repository online.
- Extra security.
- Access anywhere.
- Paid-for service → price increasing on amount of storage needed, actions needed, support and other features e.g. wikis
- Free versions are available → limited storage & actions → not be suitable for large projects.

Thank you for listening