To Git or Not to Git

Sergiu Ivanov

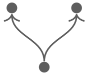
sivanov@colimite.fr

`http://lacl.fr/~sivanov/doku.php?id=en:togitornottogit`

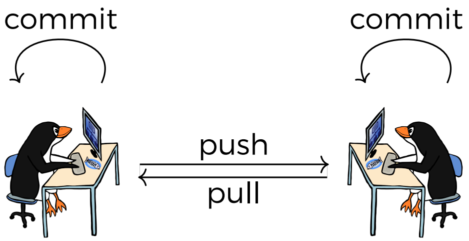
What is Git?

What is Git?

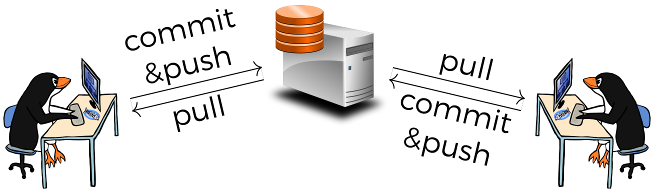
- a system for managing different versions of text files
- non-linear (branching) histories are allowed



Git: decentralised



SVN/CVS/etc.: centralised



https://openclipart.org/

Why Git?

Why Git?

- decentralised (flexible)
- fast
- clean interface (+ graphical tools)

What do we track with Git?

https://openclipart.org/

What do we track with Git?

✓ text files (program code, LaTeX code, etc.)

✗ images, PDFs, executables

- Git workflow is text-oriented
- no point in storing files generated from tracked source

`https://openclipart.org/`

One Unpleasant Effect of Tracking Binaries

1. Alice commits **program.c** and the executable **program**
2. Bob clones Alice's repository
3. Alice changes **program.c** and recompiles **program**
4. Bob wants to follow Alice's update
   - Bob does no changes

https://openclipart.org/

One Unpleasant Effect of Tracking Binaries

1. Alice commits **program.c** and the executable **program**
2. Bob clones Alice's repository
3. Alice changes **program.c** and recompiles **program**
4. Bob wants to follow Alice's update
    - Bob does no changes

Merge conflict!

- changes in **program** are not localised
- properly diffing binary files is tricky

`https://openclipart.org/`

What is a commit?

What is a commit?

A structure containing the following elements:

- a commit message
- an author
- a description of changes: additions/deletions
- a reference to the parent commit

What is good size for a commit?

What is good size for a commit?

- A commit is a set of changes bringing the program from one working state to another. (almost always)
- A commit is a set of changes which can be “naturally” described in one sentence.

Rule of thumb (C++/LaTeX): $\leq$ 100 lines/commit

- varies depending on the context/language
- 1-line commits are fine
- 100000-line commits are almost never fine

What’s in the commit message?

Line 1	the sentence describing the commit
	[empty line]
Rest	More detailed description

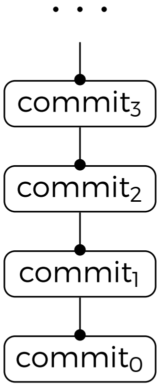
- justification of the introduced changes
- optional

$$\begin{tikzcd}
 & \text{Git} \arrow[dl] \arrow[dr] & \\
\text{Tools} & & \text{Model}
\end{tikzcd}$$

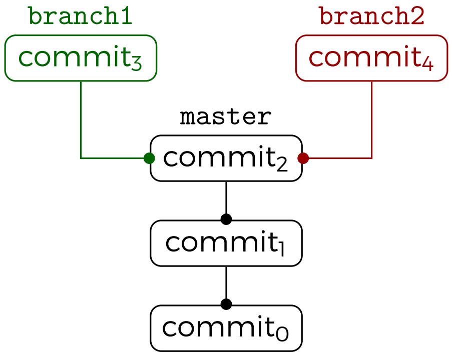
git add, git commit, ...

What do we manipulate?

Git Model: Commit Stacks



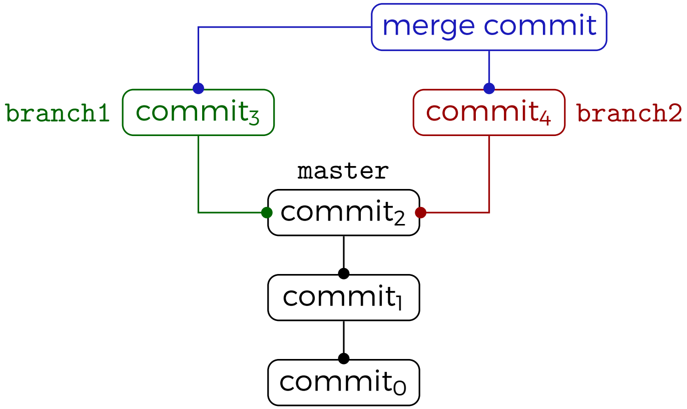
Branches: Multi-headed Commit Stacks



A branch is a name referring to a commit and to all its parent commits.

Branch names have no special meaning (not even `master`).

Merging Collaborative Work

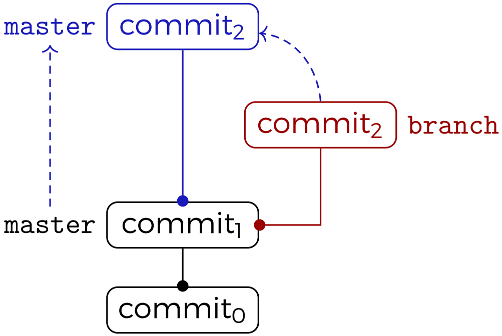


```
git checkout branch2
git merge branch1
```

Replay `branch`$_1$ on top of `branch`$_2$

- stop and let the user resolve the conflicts

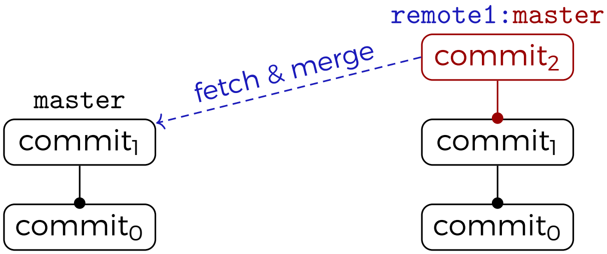
Fast-forward Merges



No divergence between `master` and `branch`.

Frequent case in practice.

Remotes: Pushing'n'Pulling



A remote is a name referring to a remote repository.

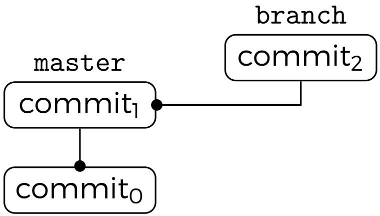
To pull changes from a remote is to:

1. make a local copy of the remote branch
2. merge the local copy into the local branch

Pushing is reverse pulling.

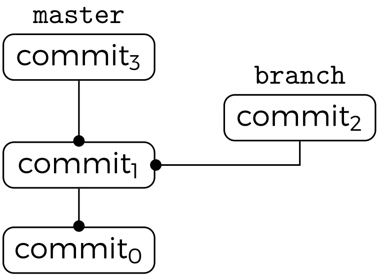
- merging happens in the remote repository

Rebase: Transplanting© Branches



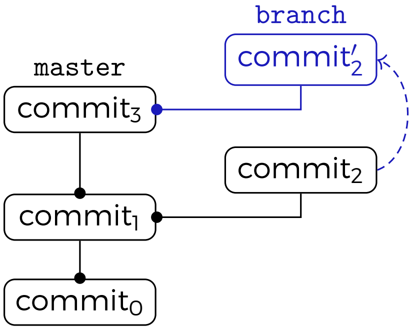
https://openclipart.org/

Rebase: Transplanting© Branches



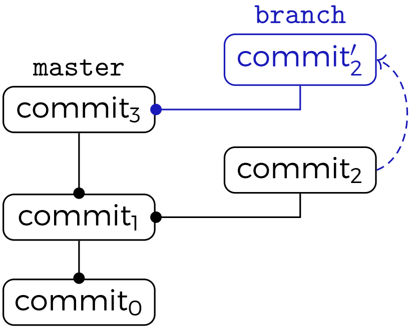
https://openclipart.org/

Rebase: Transplanting© Branches



https://openclipart.org/

Rebase: Transplanting© Branches

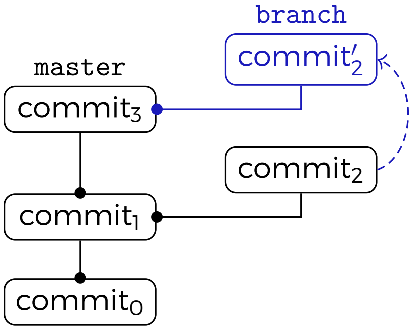


Commits are replayed and modified.

- at least the references to parents change

https://openclipart.org/

Rebase: Transplanting© Branches



Commits are replayed and modified.

- at least the references to parents change

Commits can be lost!!

https://openclipart.org/

Conclusion

Proper organisation of history and branches

- is a documentation effort;
- « requires a certain discipline ».

père Ibrahim

https://openclipart.org/