Collaborative LaTeX: An Introduction

Abel Souza

LaTeX

- LaTeX is a well known and utilised word processor for creating documents;
  - Although you have other options, like MS Word (WYSIWYG), with LaTeX one does not care about formatting, as everything is set in a specific template file;
- Many environments for LaTeX:
  - TeXStudio, texmaker, Kate, …
    - More in https://en.wikipedia.org/wiki/Comparison_of_TeX_editors

How do people collaborate?

- Email;
- Dropbox;
- Overleaf, ShareLaTeX, Authorea,…
- Use a Version Control System (VCS);

Version Control System

- Have you ever:
  - Made a change, realised it was a mistake and wanted to revert back?
  - Lost documents and had an old backup?
  - Wanted to see differences between two versions of a document?
  - Wanted to review the history of changes?
  - Wanted to share your doc, or let other people work on it?

Version Control System

- If you answered ‘Yes' to some of these questions, a VCS should make your life easier.

  - Quote: “A civilised tool for a civilised age”

Version Control System

- Centralised
  - Designed with the intent that there is One True Source that is Blessed, and therefore Good.
    - CVS, SVN,…
- Distributed
  - Designed with the intent that one repository is as good as any other, and that merges from one repository to another are just another form of communication.
  - Allows many collaborators to work on a given project without requiring them to share a common network;
    - Git

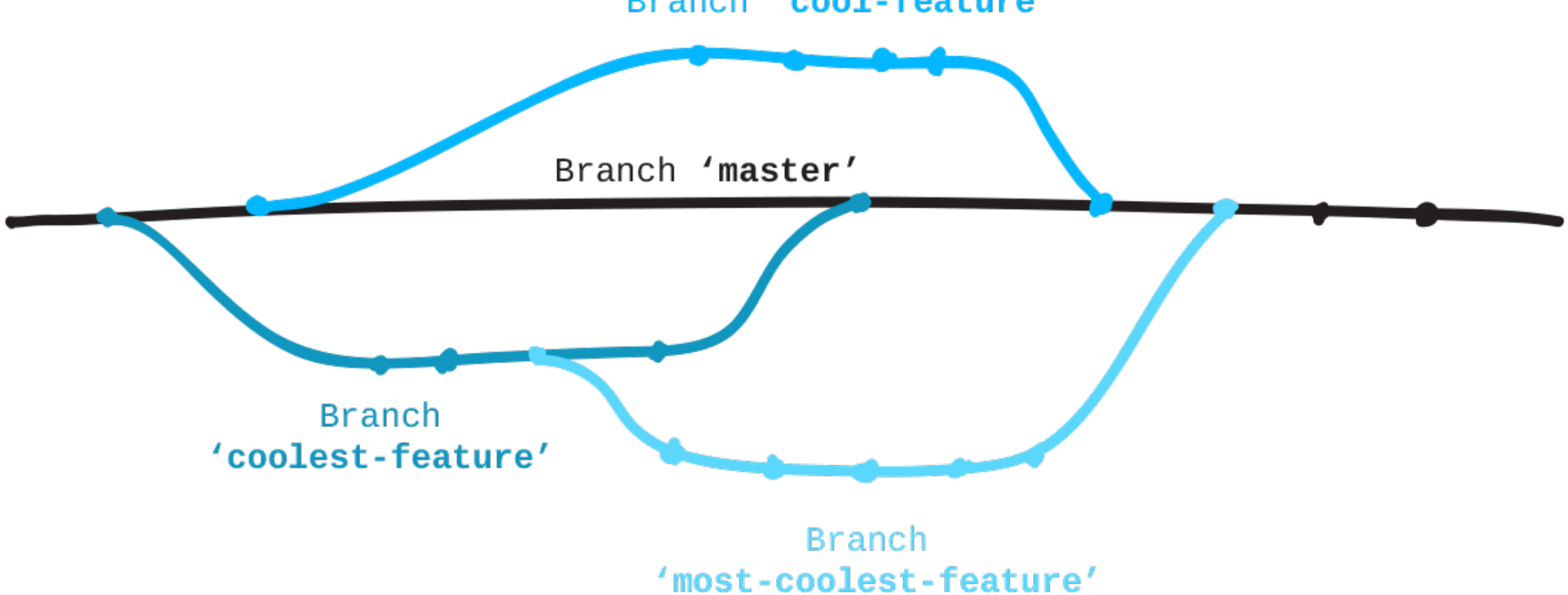
Repositories

- Each VCS project must have a repository where the owner can keep all files and changes, which are later distributed to other people/collaborators;

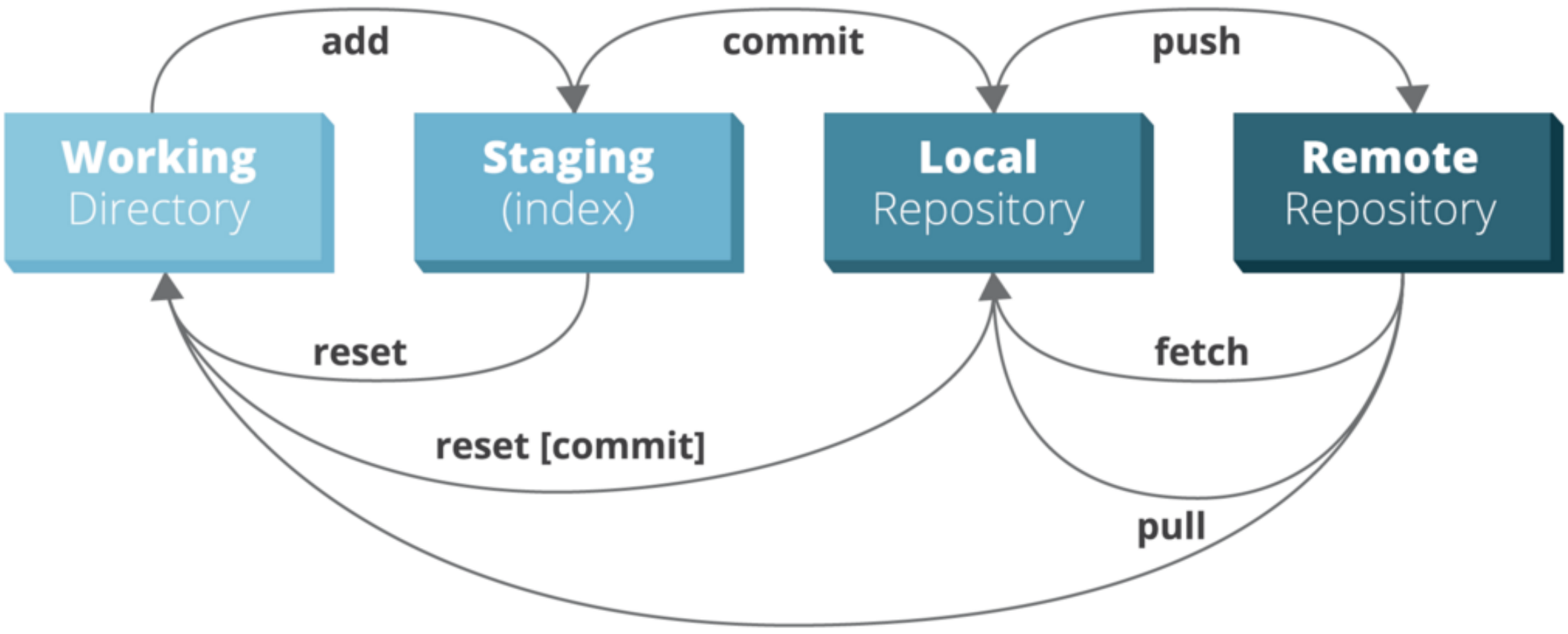
- GitHub, GitLab, BitBucket,…

GIT

- Works with the concept of Branches and Merges:
  - Development line with a given purpose;



GIT Commands



Git Cheat Sheet

REBELLABS by ZEROTURNAROUND

Create a Repository

From scratch -- Create a new local repository
```
$ git init [project name]
```

Download from an existing repository
```
$ git clone my_url
```

Observe your Repository

List new or modified files not yet committed
```
$ git status
```

Show the changes to files not yet staged
```
$ git diff
```

Show the changes to staged files
```
$ git diff --cached
```

Show all staged and unstaged file changes
```
$ git diff HEAD
```

Show the changes between two commit ids
```
$ git diff commit1 commit2
```

List the change dates and authors for a file
```
$ git blame [file]
```

Show the file changes for a commit id and/or file
```
$ git show [commit]:[file]
```

Show full change history
```
$ git log
```

Show change history for file/directory including diffs
```
$ git log -p [file/directory]
```

Working with Branches

List all local branches
```
$ git branch
```

List all branches, local and remote
```
$ git branch -av
```

Switch to a branch, my_branch, and update working directory
```
$ git checkout my_branch
```

Create a new branch called new_branch
```
$ git branch new_branch
```

Delete the branch called my_branch
```
$ git branch -d my_branch
```

Merge branch_a into branch_b
```
$ git checkout branch_b
$ git merge branch_a
```

Tag the current commit
```
$ git tag my_tag
```

Make a change

Stages the file, ready for commit
```
$ git add [file]
```

Stage all changed files, ready for commit
```
$ git add .
```

Commit all staged files to versioned history
```
$ git commit -m "commit message"
```

Commit all your tracked files to versioned history
```
$git commit -am "commit message"
```

Unstages file, keeping the file changes
```
$ git reset [file]
```

Revert everything to the last commit
```
$ git reset --hard
```

Synchronize

Get the latest changes from origin (no merge)
```
$ git fetch
```

Fetch the latest changes from origin and merge
```
$ git pull
```

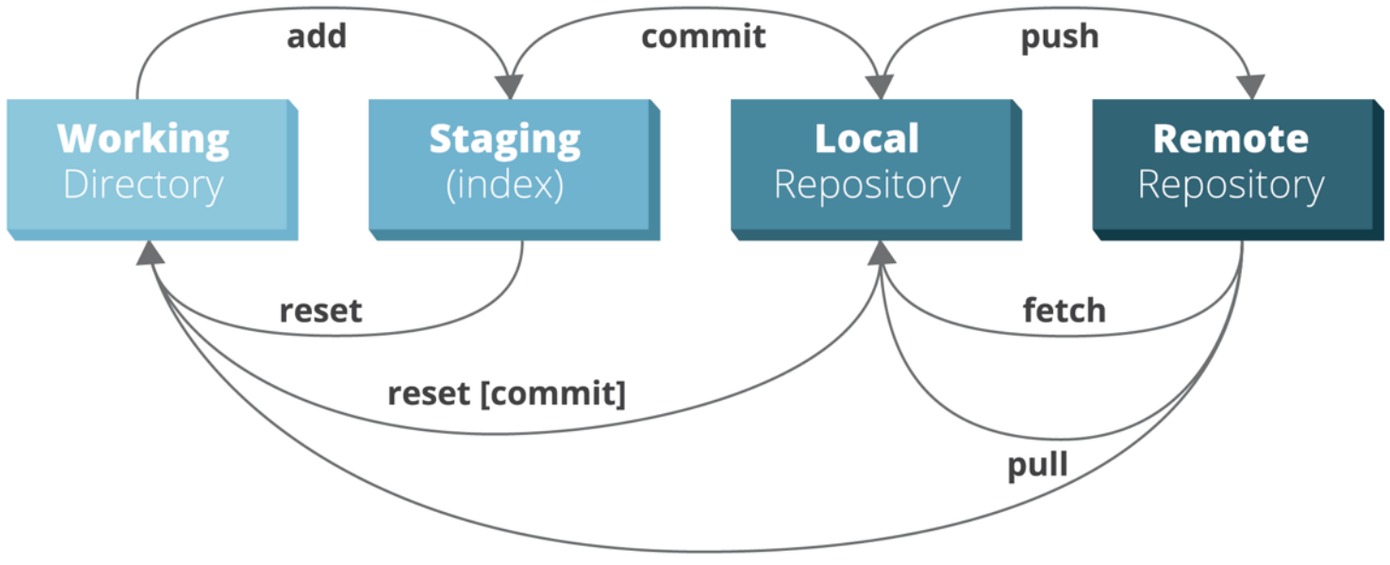
Fetch the latest changes from origin and rebase
```
$ git pull --rebase
```

Push local changes to the origin
```
$ git push
```

Finally!

When in doubt, use git help
```
$ git command --help
```

Or visit **https://training.github.com/** for official GitHub training.



Tools

- Useful for those who do not want to memorize all these commands, but still want to use the advanced features and commands;

- SmartGIT (All), SourceTree (Mac),…

SmartGIT

- SmartGit is a Git client with support for GitHub (Pull Requests+Comments), SVN and Mercurial;

- It runs on Mac OS X, Windows and Linux;

- Free for academic purposes;

Small Demo