Git Cheat Sheet

Version control is like an unlimited 'undo'. It allows many people to work in parallel, collaborate and share their work.

Setting Up Git - Once per computer

Configure user information for all local repositories:

```
$ git config --global user.name "[name]"
$ git config --global user.email "[email address]"
$ git config --global init.defaultBranch main
```

Creating a Repository

Git stores all of its repository data in the `.git` directory.

Turn an existing directory into a git repository:

```
$ git init
```

Tracking Changes

Files can be stored in a project's working directory (which users see), the staging area (where the next commit is being built up, after `git add`) and the local repository (where commits are permanently recorded, after `git commit`).

Show the status of a repository:

```
$ git status
```

Put files in the staging area:

```
$ git add [file]
```

Save staged content as a new commit in the local repository:

```
$ git commit -m "[descriptive message]"
```

Exploring History

Display differences in file compared to a previous commit:

```
$ git diff [commit] [file]
```

Restore version of a file from last commit:

```
$ git restore [file]
```

Restore version of a file from a previous commit:

```
$ git restore --source [commit] [file]
```

Ignoring Things

The `.gitignore` file is a text file that tells Git which files or folders to track and which to ignore in the repository.

Remotes in GitHub / Collaborating

First, you'll need to set up SSH to authenticate with GitHub.

A local Git repository can be connected to one or more remote repositories in GitHub (or other hosting platforms).

Clone (download) a GitHub repository:

```
$ git clone [url]
```

Copy changes from a local repository to a remote repository:

```
$ git push
```

Copy changes from a remote repository to a local repository:

```
$ git pull
```

Branches

Branches allow parallel work without affecting the main codebase. Each branch is a parallel snapshot until merged.

See current branches:

```
$ git branch
```

Switch to a different branch

```
$ git switch [branch-name]
```

Create a new branch

```
$ git switch -c [branch-name]
```

Merge a specified branch's history into current branch

```
$ git merge [branch]
```

Delete the specified branch

```
$ git branch -d [branch-name]
```

Pull requests (PR) on GitHub enable review and discussion before merging.

Forks

A fork is your own copy of another GitHub repository (the "upstream" repo). You can make changes freely even if you don't have write access to the upstream repo.

To see current remote repositories being tracked:

```
$ git remote -v
```

Add a new remote being tracked, maybe the upstream repo:

```
$ git remote add [remote-name] [url]
```

You can contribute by creating a pull request from your fork to the upstream repository.

Conflicts

Conflicts occur when two or more people change the same lines of the same file. Git doesn't allow people to overwrite changes blindly, but highlights conflicts so that they can be resolved.

Defining a workflow with your team and communicating is the first step to avoid merge conflicts.

Open Science / Licensing / Citation / Hosting

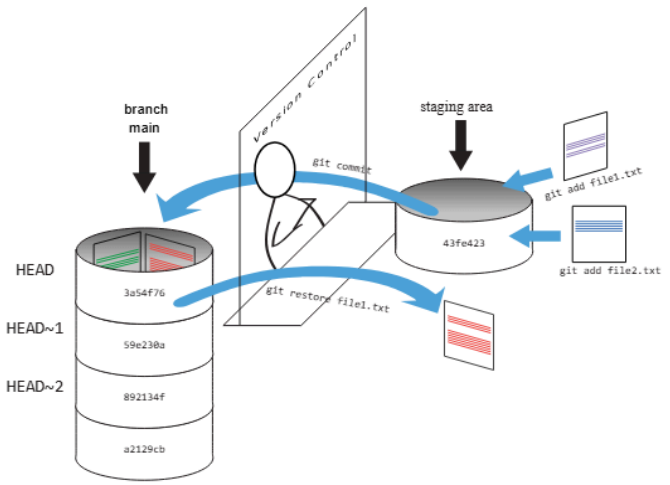
Open scientific work is more useful and accelerates discovery.

The LICENSE.md or LICENSE.txt file is often used to indicate how the contents of the repo may be used by others.

Add a CITATION file to explain how you want your work cited.

Projects can be hosted on university servers, on personal domains, or on a public hosting service (like GitHub or GitLab).

Rules regarding intellectual property and storage of sensitive information apply no matter where code and data are hosted.



Huang, Daisie (2015). How Git works: a cartoon.