Digital Lab Notebook with Git/GitHub and Markdown

Instructor's Notes

Table of Contents

Introduction (∼40min)

What is git?

Duration: 15min

- Before you begin, make sure everyone has git installed in their local machine (for later use)
- Start with examples of keeping track of changes of a document (e.g. thesis/paper)
  - MS Word keeps record of changes of the **current** document – when you close and re-open the document, you lose all of the changes you made previously
  - To avoid the situation as above, people usually make a copy before they start editing, and then (eventually) compare the new version with the old one (which is a hassle)
- Example FAQ: "Can you reproduce your results?" (were your methods/protocols well documented?)
- Example FAQ: "What protocol(s) did you use for that experiment?" (when you have multiple trial/error attempts)

- Make absolutely clear the objective(s) of using a version control system
  - To keep track of and maintain the most current version of your document
  - Keep a long-living history of your document/protocols
  - Share documents with others for collaboration
  - ‘Back-up’ of your codes

(Make a diagram somewhere)

Creating GitHub

Duration: 10min

- Create a GitHub account
- Explain what GitHub is
  - A “remote” server for your git repository
  - Essentially an online storage space for the documents that you want to version control
- Create a DLN repository to work with during the course
- Explain what a repository is

Markdown files

Duration: 15min

- Edit README.md
- Explain and show what a Markdown document is

- Introduce some Markdown syntax
    - Headers
    - Text decorations (bold, italics, underline, etc.)
    - Web Links
    - Adding images
    - Footnotes
    - Tables

Play around with Markdown syntax using the README file (saving changes and committing comes in the next chapter/section).

Making Digital Lab Notebook (∼50min)

Change, add, and commit

Duration: 20min

- Save changes and commit the README.md
- Explain what committing means
- Show what the logs/commit history look like
- Explain what the git commit ID/hash are
- Add/remove more stuff to the README file and commit changes
- Note the diff of the documents when making changes
- Note that, althought you can "checkout" previous commits, you can't "revert" the repo to that particular commit on GitHub
  - Reverting to different commit will be covered in command line git
- Keep adding changes to the README and demonstrate how git/GitHub is keeping track of all the file changes

Creating new entries

Duration: 15min

- Start creating new entries to the repo
  - Make a new directory in the repo for your entries (e.g. protocols, notes, results)
- Add an entry to the directories as before and commit
- Keep showing what the commits look like through the commit history

Linking entries

Duration: 15min

- Start creating links to your new entries in README
  - Explain directory structure when creating links to new entries
- Point out the power of doing this
  - README as the content viewer/navigator
  - Individual entries organised in directories
  - Everything version controlled with git

Working Collaboratively and Locally (∼50min)

Sharing your repository

Duration: 15min

- Just send your collaborator the URL to your repository
- Demonstrate how to give editing access to others for collaboration
- Let everyone play around with editing other people's repository and sending pull requests
- Introduce the idea of merge conflicts and let them try it out

Why use git on local machine?

Duration: 15min

- Able to checkout, but can't revert to a particular commit on GitHub
- For bulk adds/commits (edit locally, and then push commits)
- You can't have a "working draft" when you are editing it on GitHub – whatever change you make to the document, you have to commit it after each edit

    - You can, but you will have to create a new branch for that (which could be a hassle to keep track of)

Setting up git on your local machine

Duration: 20min

- Follow first half of the SWC git lesson for git setup
- Clone the already created repository from GitHub
- Repeat similar process as what we did on GitHub (add changes, commit, log, revert, push)
- Explain new ideas (i.e. add, revert, pull, and push)