Git/Hub for Collaborative DataSci & DataEng

With the AI Club for Gender Minorities, London

By Alessia Visconti, Ester Ramos & Fei Phoon

26/02/2019

For Alessia, if you need a slide or two?

Actually, you're here to learn:

~~**Git/Hub for Collaborative DataSci & DataEng**~~

Good Collaboration in Nearly Anything File-Based, using GitHub

Good collaboration...

- makes a safe space for constructive and honest discussion.
- makes room and time for teaching and learning.
- accepts mistakes & mishaps - and that we'll help each other fix and learn from them.
- does not blame and shame.
- gives credit and recognition where it is due.
- is first and foremost about working with and respecting other people.

Good collaboration...

- is NOT an optional "soft" skill for engineers - it's essential!

Git & GitHub are only tools to help us focus on tracking contribution and discussion.

Git vs GitHub

- **Git != GitHub**
- **GitHub:** a web application that allows people to store and collaborate on a project. Today we'll learn to use this!
- **Git:** a version control system; the underlying magic/the engine under the hood, that manages contribution history and order.

Key Git Concepts

1. **Repository:** a project.
2. **Branch:** a collection of changes.
3. **Master:** the "default" branch, also the most recent, fully-approved version of the project.
4. **Commit:** a saved change on a branch. So branches are groups of changes with a common objective.
5. **Pull request (PR)**: a request to add your branch of changes to `master`.
6. **Merge**: the action that accepts your branch into the master branch.

Today's learning objectives

- Learn a universally-accepted collaboration workflow
- Use key Git concepts on a project, using the GitHub web UI.

Workshop Time

See: https://github.com/feiphoon/github-workshop/

Reminders for collaboration:

- Be kind (you can still be firm & opinionated!)
- Leave constructive comments
- If you feel an online conversation isn't working out, retry the discussion in person.
- If you're lost, ask for help! There are no silly questions, only lost people feeling sad that they didn't ask for help sooner.

Workshop Recap

- created a repository
- learned to make commits
- logged an issue
- responded to an issue
- asked for a review
- reviewed changes
- accepted a review
- merged or confirmed contributing your changes
- tidied up!

Git in the Wild

- Today's exercise: a pared-down version of what many code contributors use, from individuals & small teams on hobby projects, to developers & engineers on large repositories at work.
- Other variations for Git collaboration:
  - **git-flow** - fancier version of what we did today, designed especially to manage releases e.g. `release v1.7`
  - **forking** - especially for open source or community-supported projects

Git the Party Started

- Examples of non-code projects on GitHub:
  - A crowdsourced travel itinerary: https://github.com/dylanegan/travel
  - A novel: https://github.com/gregorygershwin/Benjamin-Buckingham-And-The-Nightmares-Nightmare-Novel
- Open source libraries with collaborators all over the world, e.g. https://python-sprints.github.io/

Other Interfaces for Git

This is down to your workflow preferences:

- Bitbucket
- Gitlab
- GitHub Desktop
- Git command line
- Git integration in code editors & IDEs (Atom/Sublime/PyCharm)

Your Future Lies In:

- finishing these exercises with your teammates
- starring the workshop repo for reference - check back for some light recap reading
- looking at the history of the workshop repository - we collaborated remotely to create it!
- exploring GitHub further by using it for a project
- coming to our next session to learn to use GitHub Desktop and Git from command line
- eventually learning to contribute to open source e.g. Pandas

Question Time

Thank you!