Getting Started with Git and GitHub pt. 2

April 28th, 2025

Worked Through Example and Accessing the Code

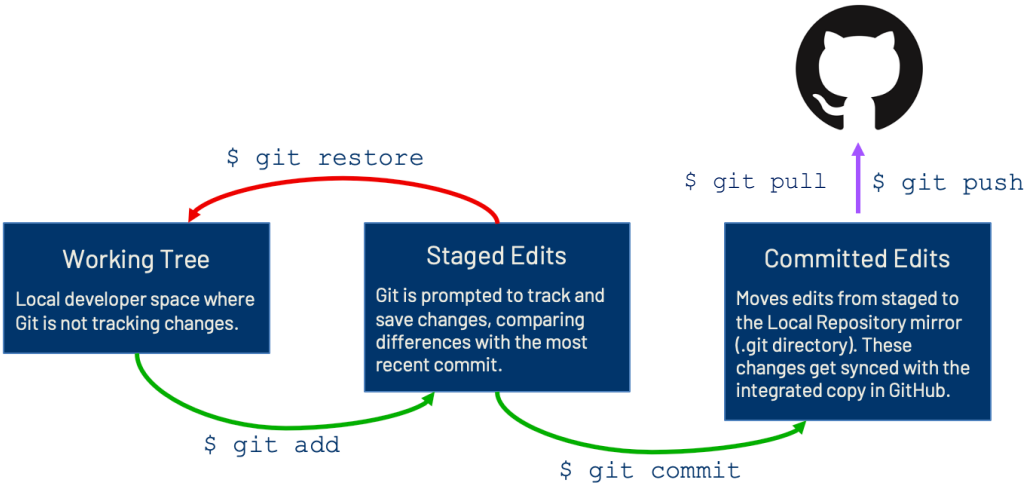
Open the workshop webpage: https://ysph-dsde.github.io/Book-of-Workshops/Git-and-GitHub/

On the introduction page for today's workshop, you will find directions on how to import or create a "clean-break" copy of the two GitHub repositories that will be used today. We will organize in-person attendees into small groups so that you can take full advantage of the collaboration section.

- Everyone will need to import the ysph-dsde/JHU-CRC-Vaccinations repository prior to cloning it onto their local device.
- Have one person in your group import the ysph-dsde/JHU-CRC-Cases-and-Deaths repository. This person will then add the other group members as collaborators.

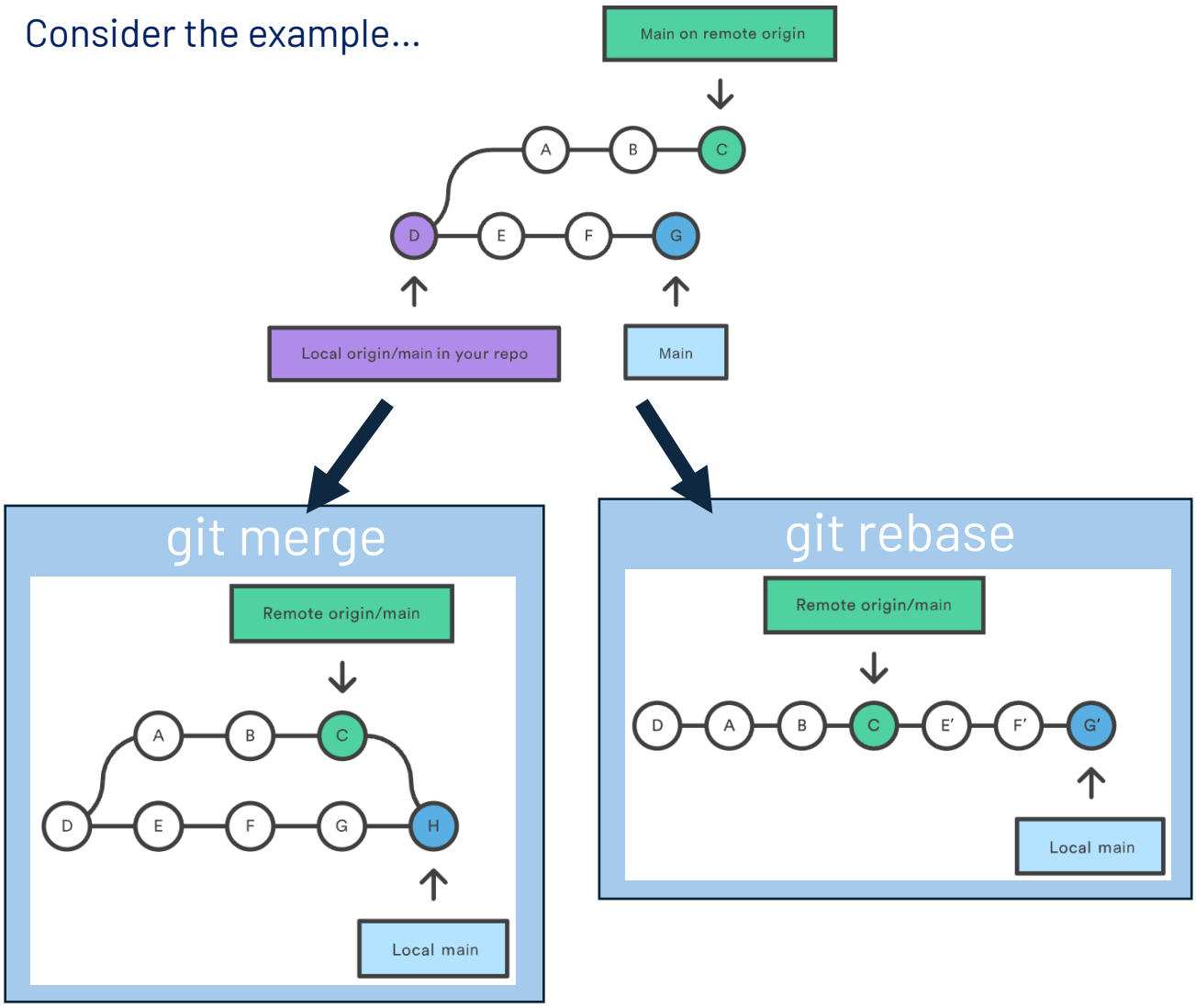
Please keep in mind that our "Book of Workshops" is still a work in progress. We plan to add quite a lot of the materials to this webpage for this and other workshops for students to access for asynchronous learning opportunities.

Figures



Standard version control overview reflecting the Git bash commands executed through the command-line interface in local project directories.

Consider the example...



Two methods to reconcile divergent branches during a pull. Merging is the default setting. Images come from Atlassian's Git Tutorial, downloaded April 24th, 2025.

Helpful Cheat Sheets:

1. Git Cheat Sheet by Atlassian: https://www.atlassian.com/git/tutorials/atlassian-git-cheatsheet
2. Vim Cheat Sheet by Richard Torruellas: https://vim.rtorr.com/
3. Bash Shortcuts by Mohan Balasundaram: https://gist.github.com/tuxfight3r/60051ac67c5f0445efee
4. Command Line Cheat Sheet by Tobias Günther: https://www.git-tower.com/blog/command-line-cheat-sheet

Glossary

Version Control	Manage, organize, and track different versions of files. These systems identify differences between versions and allows reverting to older versions. Example: Google docs.
Distributive Version Control System	The project codebase is copied as a mirror to each contributor's local computer. Local changes get synched via patches sent peer-to-peer through the server.
Command-Line Interface (CLI)	A texted-base application that directly interacts with the computer's operating system, manages files, and can run programs. It typically lacks a graphical user interface (GUI).
Shell	A program used by the CLI to mediate communication between the user and computer by interpreting commands and outputs. Examples: Bash, Power Shell, etc.
Mirror	An exact copy of a project from a server, including a full-change history.
Server	Computer or system that provides resources (i.e. data or programs) to other computers, known as clients, over a network.
Patch	Snippets of code or data used to update existing software.
Peer-to-peer	Participants in a network act as both client and server by trading resources and services with one another.
`git add`	Prompt git to track changes that have been made to specific files and compare those differences to previously saved version in the .git directory.
`git commit`	Save your changes as a snapshot in your project's history, present in the .git directory.
`git push`	Upload your committed work to a shared online location, so others can see it.
`git merge`	One method to reconcile different committing histories in divergent branches. Creates a new version integrating the head of the two branches in a three-way commit.
`git rebase`	An alternative to merge. The branch commit histories are realigned so that the leading one defines the commit parent history of the following branch, thus rebasing its commits.
`git clone`	Make a copy of a project from the internet to your own computer.

`git branch`	Create or list different "versions" or "paths" of a project that you can work on separately.
`git checkout`	Command to switch between different the branches.