Assignment 4. Basic change management

[course home > assignments]

Useful pointers

- Scott Chacon, Pro Git. Read the following: §§1–3 (basics and branching), §5 (distributed version control), §7.1 (revision selection), §7.5 (searching), and §7.10 (debugging).
- You may also want to refer to the Git web site. Git is written by Linus Torvalds, Jun Hamano *et al.*

Laboratory A: Exploring a linear development history

This lab uses the development repository for the Time Zone Database (tzdb).

1. Use GitHub from a browser to compute the difference between the previous and current commit to this repository. Save the resulting web page as a file `prevcur.html`.
2. Use GitHub from a browser to compute the difference between tzdb releases 2022f and 2022g. Save the resulting web page as a file `2022f-2022g.html`.
3. Clone the tzdb development repository, in Git format.
4. Write a shell or Python script `justone` that displays the difference from the previous and current commit, assuming the repository is what an ordinary Git command would use. Use your command on the just-cloned repository, and put the output of your command into a file `justone.out`.
5. Write a shell or Python script `compare-releases` that displays the difference between two tzdb releases given as arguments to the command. For example, `compare-releases 2022f 2022g` should output the difference between tzdb release 2022f and tzdb release 2022g.

Put the output of this particular invocation into a file `2022f-2022g.diff`.

6. Suppose we're interested in the number of commits from each time zone. Write a shell or Python script `tzcount` that postprocesses the output of `git log` and outputs a simple report of time zones and number of commits from that time zone. Each line of output should look something like "`-0500 1802`", meaning there were 1802 commits from the −0500 time zone. Use the commit date, not the author date, to determine the time zone of the commit. Sort the output numerically by its first (numeric timezone) column. Run the command `git log 2012j..2022g | ./tzcount` using the tzdb repository, and put its output into a file `tzdb-2012j-2022g.tzcount`.
7. Suppose the maintainer of tzdb is being sued for copyright infringement because one of the source files contains the following sentence: "Even newspaper reports present contradictory information." Also suppose the plaintiff claims that this statement was improperly copied from the plaintiff's book. Use Git and other commands to find out how this statement was originally introduced to the tzdb files. (This is not the same thing as merely finding the last change to the lines containing the statement in question.) Create a text file `who-contributed.txt` that describes what commands and/or scripts that you used, and what the result of your investigation was.

Laboratory B: Exploring nonlinear development histories

Examine the copy of Git's own git repository on SEASnet in the directory `~eggert/src/github/git`.

1. Find the mergepoint M at c03bee6e9f5c05259f5f501e1f47cd8adb63af38 (committed 2022-10-02), and draw the directed graph of all paths to M from the commit 2a7d63a2453e2c30353342a2c9385fa22a846987 (committed 2022-09-26). Label each node in your graph with the commit ID, author, and committer if different. Your graph should contain all the abovementioned commits, along with any and all intervening commits; the arcs in your path should be from child to parent. You need not graph descendants of M, or ancestors of the other commits.

2. Clone Git's git repository yourself from GitHub, and briefly describe the differences between your repository and the one in `~eggert/src/github/git`. (Hint: look at the output of `git branch`.)

Put your descriptions into a text file `git.txt`. Put your diagram into a PDF file `git-graph.pdf`.

And now for a more open-ended search. The Git v2.39 Release Notes (2022-12-12) say "In read-only repositories, "git merge-tree" tried to come up with a merge result tree object, which it failed (which is not wrong) and led to a segfault (which is bad), which has been corrected." Which code change or changes actually made this happen, and who authored the changes? Do not worry about subsequent administrative changes such as merge commits; look for the original changes that actually fixed the problem in question. Give the commit ID or IDs for the relevant change or changes, and explain how you discovered them, all in a text file `git-detective.txt`. Also generate patch files for the relevant change or changes, in `git format-patch` format.

Submit

Submit the following files at the top level of a gzip-compressed tarball `gitlabs.tgz`. Your tarball may contain other files if you think it necessary.

- `prevcur.html` web page
- `2022f-2022g.html` web page
- `justone` script
- `justone.out` output file
- `compare-releases` script
- `2022f-2022g.diff` output file
- `tzcount` script
- `tzdb-2012j-2022g.tzcount` output file
- `who-contributed.txt` infringement report
- `git.txt` descriptions
- `git-graph.pdf` descriptions
- `git-detective.txt` descriptions
- The output files generated by `git format-patch` in your detective work.

$Id: assign4.html,v 1.53 2023/02/09 04:50:00 eggert Exp $