Git Gud!

Merge and Rebase

By Sander Beekhuis

Merge

```
→ git log --oneline --graph --all --date-order
*   ce33f67 (HEAD -> main) Merge branch 'my-feature'
|\
| * fea9a04 (my-feature) Final bit of work
* | 7267c51 Something happens in the meantime
| * 0fefa54 Some more work
| * 964f71f Some work
|/
* b662be3 Initial commit
```

- `merge --no-commit`
- `merge --abort`

Rebase

Conceptually: "Moves" a branch to a new location

Actually:

- Does not actually move anything
- Makes new commits based on the old ones
- Asks you to intervene when automatic resolution is not possible

Interactive Rebase

`git rebase -i` - Clean up your commit's while rebasing.

```
pick 121bce7 Add basic logging

# Rebase 28a42e2..121bce7 onto 28a42e2 (1 command)
#
# Commands:
# p, pick <commit> = use commit
# r, reword <commit> = use commit, but edit the commit message
# e, edit <commit> = use commit, but stop for amending
# s, squash <commit> = use commit, but meld into previous commit
# f, fixup [-C | -c] <commit> = like "squash" but keep only the previous
#                    commit's log message, unless -C is used, in which case
#                    keep only this commit's message; -c is same as -C but
#                    opens the editor
```

History alteration

```
workshop-git on ᚹmain ↑1 ↓1
→git status
On branch main
Your branch and 'origin/main' have diverged,
and have 1 and 1 different commits each, respectively.
  (use "git pull" to merge the remote branch into yours)

nothing to commit, working tree clean
```

Your local changes can not be pushed in the normal way since they differ from the remote. They can even possibly miss meaningful changes.

- `push --force` to the rescue
- `push --force-with-lease`

Merge conflict

Git merge logic:

- Find "merge base" commit
- Create diffs for both branches
- Play both diffs
- Ask user when both diffs edit the same chunk differently

Time for a lab