presentation.md

marp: true theme: ing class: invert

ING Training GIT

Why GIT?

- Wide-spread usage. Almost all software projects are developed using GIT
  - Linux
  - Python
  - Pandas
- Full history & backup of all changes
- Useful even while working alone.

👉 Essential skill as data scientist.

Homework

GIT for humans

Git has a very steep learning curve.

History: Created by Linus Torvalds in 2005 (wiki)

Learning goals today

- Basic git workflow
- Working with multiple people via a remote (github or gitlab)
- Fixing merge conflicts
- Branching
- Merge requests
- Using github's features

Windows?

`cmd` or `powershell` wont't cut it. Either:

- Install Git Bash for windows
- Use Windows Subsystem for Linux (windows 10)

Show & tell: The basics

- `mkdir projectname && cd projectname`
- `git init` Make the current folder a git repository
- `git status` See the current status of git. Use it A LOT!
- `git add <file>` Add a file to the staging area
- `git commit` Add the changes in the staging area to a commit 📸
- `git log` View the list of commits

👉 Fun commit messages? gitmoji

Exercise 🔨

1. Create a local git repo
2. Add a new file
3. Commit it to git

Solution 📸

```
git init
touch README.md
git add README.md
git commit -m "Add README"
```

Using the history

- `git diff <commit>` or `git diff <file>` Compare changes
- `git checkout <commit>` Time travel to a specific commit
- `git checkout <file>` (⚠️) Removes all edits to a file since the last commit
- `git checkout master` Go back to most current version of the `master` branch
- `git revert <commit>` Undo a commit (creates a new commit)
- `git blame <file>` For each line who wrote it

Working together

We need to send our work to a central location, a single *origin*. This can be gitlab or github or another hosting service.

Show and tell: projects on github

On github.com:

- Navigate to this training repo
- Add SSH keys

Using github

- `git clone` Downloads the project.
- `git remote -v` shows the URL of the *origin*.
- `git push origin master` Pushes new commits to *origin* from the *master* branch

Exercise 🔨

1. Clone our example repo
2. Create a new file called `<yourname>.md`
3. Push it to the origin

Viewing our collaboration

- Github's visual commit history (instead of `git log`)
- Github's visual blame (instead of `git blame`)

Show and tell: Merge conflicts

What if you edit the same line? Let's try:

- (trainer) change, add and commit a line in a file
- (volunteer) change, add and commit same line
- (volunteer) push code to origin
- (trainer) push code to origin ❗ conflict ❗

To solve, edit the files manually.

Exercise 🔨

1. Make pairs
2. Agree on editing the same line in one of your `<yourname>.md` files
3. Let one person push first, and the second shares his/her screen
4. Solve the merge conflict together

Branches

What if your changes break work? Work on a copy of the code using *branches*.

- `git checkout -b somethingnew` Create a new branch
- `git branch` Overview of the branches in the repository
- `git checkout master` Go to the *master* branch
- `git checkout <branch name>` Go to the branch
- `git merge <branch name>` Merge commits from into the current branch
- `git branch -d <branch name>` Delete the branch

Exercise 🔨

1. Make a new local branch and create a first commit
2. Switch to the master branch and create a second commit
3. Merge the two branches so everything is in the master branch

Show & tell: Github UI

- Navigation
- Starring projects
- Files, commits
- File history
- Git blame
- Statistics
- Github issues & closing through a commit message

🚚 Note: ING is moving to Azure DevOps and decommissioning gitlab this year

Show & tell: Using remote branches

- Create a branch using the github UI or,
- Push a branch using `git checkout -b branchname` and `git push origin branchname`

💪 Pull Requests !

- Pull requests -> New pull request
  - With commit message, when merged, closes linked issue
- Very powerful, especially when using web IDE for quick edits

Exercise 🔨

1. Create a specific issue for yourself
2. Create a branch
3. Push to this branch
4. Create a pull request (PR) for it
5. Checkout the branch for your PR locally
6. Update your `<yourname>.md` file and push
7. Merge your PR

Final exercise 🔨

In pairs of two. Instructions per person:

- Create your own individual github issue
- Create a branch
- Create a github pull request (PR) for your issue
- Checkout the branch for your PR locally
- Edit a file and push a commit to your branch
  - You can reference your issue in the commit message ("did a thing see #1")
  - Make sure to fix any merge conflicts if they arise
- Next, checkout and push a commit to your teammate's branch
- Finally, have your teammate review (leave some comments?) & approve your PR
- Merge your PR! 🎉

Finally: Working more efficiently

Gitlab/Github aliases:

```
alias gst='git status'
alias gp='git pull'
alias gc='git commit -m'
alias gaa='git add --all'
alias gl='git log --graph'
alias gpo='git push origin'
alias gpom='git push origin master'
```

Advanced git

- `git commit --amend` Adding to last commit
- `git commit -am "message"` Short for `git add --all` and `git commit -m` . Or with aliases above: `gaa && gc "message"` .
- `git commit -m "message" -m "detailed message"` . Add description.

👉 Oh Shit! Made a mistake or got stuck? See ohshitgit.com

Learn more

- `git <command> -h` for a quick help
- See timvink/dotfiles for a good terminal setup

Thanks! 👋

Let's get in touch! Contact me at daniel.timbrell@ing.com

Follow me: https://www.linkedin.com/in/dantimbrell/