onsdag 24 mars 2021

Git

Working alone

- There are three areas or stages in git
  - Working directory
  - Staging area
    - Move files into staging area, the middle ground, we can cherry pick the files that we want to commit to the history or repository
  - History or repository
    - Snapshots or commits of the state of directories in time
    - These commits are saved here in a timeline
- Commands
  - Saving changes
    - git add foo.txt
      - The new file foo.txt have never been saved as a snapshot before, we want to move this file from working directory to the staging area using add
      - Using git add . Adds all the files
      - Git add -a adds all files that have been changed
    - git commit -> moves everything in the staging area to the git repository
    - Use -m ”text” to add a comment
  - Snapshots
    - Git takes snapshots of the current state, if we want to jump back to a snapshot, the changes that have been made after this snapshot will disappear
  - Inspect content of repository
    - git status
      - Will tell me how the working directory and staging area
    - git log
      - Will present me with a whole timeline of all the changes that I have made

  - Commits have an id, that seem random and long, but captures changes of commit. If you want to make a commit specific command this id will be used

- Tutorial
  - mkdir basic-git-seminar
  - cd basic-git-seminar
  - touch foo.txt
  - git init
    - Initialize as git repository
  - get a packet that shows in terminal which branch you are working in
  - git status
    - Shows untracked files, files that we want to add using command add
  - git add foo.txt
    - File is in staging area now
  - git commit -m ”Add foo.txt file”
    - 60 character is recommended limit of comment
  - git status
    - Nothing in staging area
  - echo ”Text” > bar.txt
  - Git add -A
    - Adds all changed files to staging area
  - git commit -m ”add new file…”
    - Move from staging area to repository
  - mkdir inner.dir
  - Cd inner-dir
  - echo ”inner file” > inner.txt
  - cd ..
  - tree
    - We now have new file in the inner directory in the repository tree

  - Cd inner-dir
  - git add -A
    - Will now only add the file in inner-dir
  - Read up on git aliases to create shorter commands
    - Can for eg change commit to ”ci” or status to ”st”
- Undoing changes
  - Up until now we have added new changes, but how about undoing?
  - git checkout a1e8fb5
    - Use checkout on a specific commit id
    - Brings us to a detached head state
      - Read only, can’t make changes here
  - git reset foo.txt
    - Takes files that have been added to the staging area and moves it to the working directory
  - git reset a1e8fb5
    - removes a commit from the history, but still keeps the changes
    - Removes the commit, but not the files
  - git reset —hard a1e8fb5
    - Removes commit and changes
    - ie removes the commit and the files
  - Warning don’t reset public history!
    - Say everybody is working on a shared directory
    - Everyone then has the same history
    - But if you use reset, then everybody doesn’t share the same history any longer
    - So do not use reset when working with public repository
  - .gitignore
    - In this file we include files that we don’t want to commit

    - For example, we don’t want to add bin files or class files

- Tutorial
    - touch phil-file.txt
    - git checkout [id]
        - Warning: you are in a detached HEAD state
        - This is read-only
    - git add .
        - adds the new file
    - touch another-file.txt
    - git reset phil-file.txt
        - Removes file from staging area
    - change the file in some way
    - Git add phil-file.txt
    - git commit -m ”Add Phil’s file”
    - git reset —hard [id]
        - removes the latest commit and files and moves us to the specific commit that we choose using [id]
- The Social Picture
    - If you are working on your project from home, and your friends are working on the same project from home, you want to make sure that you stay up to date with the changes
    - You have your local system and the Github server ’Origin’
    - git remote add origin <url>
        - adds the remote repositories to your local system
        - Creates a link between local and remote repository
    - git push origin master
        - pushes to the server origin and the master branch
        - makes sure that the origin server’s master branch points to the latest commit that you have done on your local system

- git pull origin
  - makes sure that the local master branch points to the latest commit in the origin server master branch
- git clone <url>
  - If you have created a new repository on git, and someone else wants to work on this repository on their local system
- git log
  - origin/master is the remote system
  - HEAD is the local system