Universität Göttingen
Institute for Computer Science / GWDG
Jonathan Decker

Exercise Introduction

Before attempting the exercises in this document please ensure that you have read and understood the key topics covered in Tutorial.

Contents

Task 1: Basic Git Setup (5 min) **1**

Task 2: Commits and branches (10 min) **1**

Task 3: Remote repositories (5 min) **2**

Task 4: Working with .gitignore (2 min) **3**

Task 1: Basic Git Setup (5 min)

Run the following commands and observe what they do. Feel free to test around.
You can find help for any git command using `git <COMMAND> --help` or `man git-<COMMAND>`.
Feel free to replace `nano` with your favorite text editor command.

Setup

`mkdir -p $HOME/git-ex && cd $HOME/git-exercise`	
`git --version`	
`git config --global user.name "NAME"`	Set your name.
`git config --global user.email "EMAIL"`	Set your email.
`git config --global core.editor "nano"`	Set nano as editor for commit messages.
`git init --initial-branch=main`	
`git status`	

Task 2: Commits and branches (10 min)

Committing

`touch README`	
`git status`	
`git add .`	
`git status`	
`git commit -m "Initial Commit"`	
`nano README`	Write a few words into the file and close nano.
`git status`	
`git diff README`	See your changes, close with `q`.

`git add README`	
`git commit`	Write a commit message using nano, save and close.

Reverting changes

`rm README`	
`git status`	
`git reset --hard HEAD`	Undo the delete by reverting to the last commit this also undoes any other changes you made.
`git status`	
`ls`	See that the README file is back.
`rm README`	
`git commit -a -m "Deleted README"`	Use **-a** flag to commit a staged changes.
`git status`	
`ls`	Confirm that README was deleted and the change was committed.
`git revert` TAB+TAB	This shows a list of your recent commits. Type the first two characters of the id of your last commit and press TAB and ENTER. Write a commit message for your reverted commit, save and close nano.

Creating branches

`git switch -c feature`	Create a new branch.
`echo "This is a new file." > new_file`	
`git add new_file`	Stage the new file.
`git commit -m"add a new file"`	Create a commit on **feature**.
`git log --graph --oneline --decorate`	Visualize the commit history.
`git checkout main`	Return to the initial branch.
`ls`	What happened to the working tree?

Task 3: Remote repositories (5 min)

Remote repository

	Make sure you can login to `https://gitlab.gwdg.de`, `https://gitlab-ce.gwdg.de` or `https://github.com`. Replace the domain and username accordingly.
`git remote add origin "https://gitlab-ce.gwdg.de/USERNAME/git-exercise.git`	
`git push --set-upstream origin main`	This will query your credentials if you do not have them stored already and create a remote repository. The visibility of the repository is private by default so only you and the teammates you have explicitly invited have access. Visit `https://gitlab-ce.gwdg.de/USERNAME/git-exercise` to view your new project.
`rm -rf ~/git-exercise`	Delete the local copy of the repository.
`cd`	switch back to your home directory.
`git clone https://gitlab-ce.gwdg.de/USERNAME/git-exercise.git`	
	Download the repository from remote.

`cd git-exercise`	
	Make a change to README on the web and commit it.
`git pull`	
`git log`	See the change you made on the web.

Task 4: Working with .gitignore (2 min)

gitignore

`touch credentials`	
`git status`	See that the credentials file can be staged.
`nano .gitignore`	Write **credentials** into the file and save your changes.
`git status`	See that only **.gitignore** can be staged and credentials is ignored.
`git add credentials`	

Git has many more features, one of them, which is commonly used, is branching.

Further Reading

- Missing Semester; Version Control (Git): `https://missing.csail.mit.edu/2020/version-control/`
- Learn Git branching: `https://learngitbranching.js.org/`