

WORKSHOP 0 – GIT

Adrian.Pop@liu.se

Agenda

- What is git?
- Configuring
- Creating a repository
- Cloning
- Status & Log
- Checkout & Branching
- Adding files & Committing
- Merging & Conflict handling
- Push & Multiple remotes
- Updating the repository
- Extra information
- Exercise

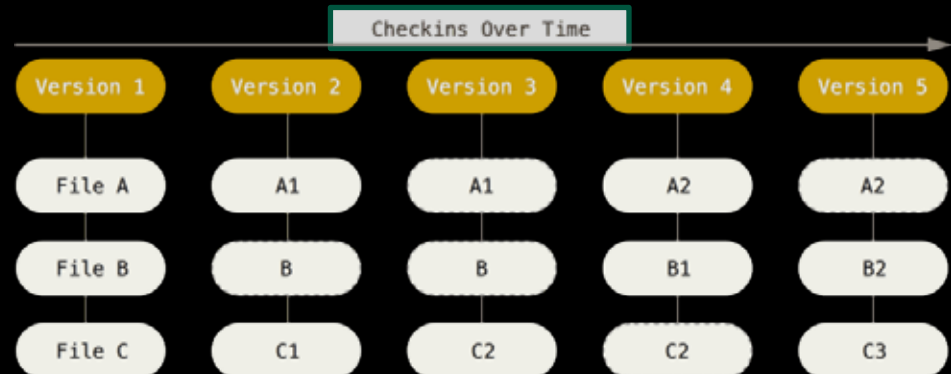
What is git?

- Source versioning system based on snapshots
- Snapshots can be named (branches)
- Most operations are local, except clone and push



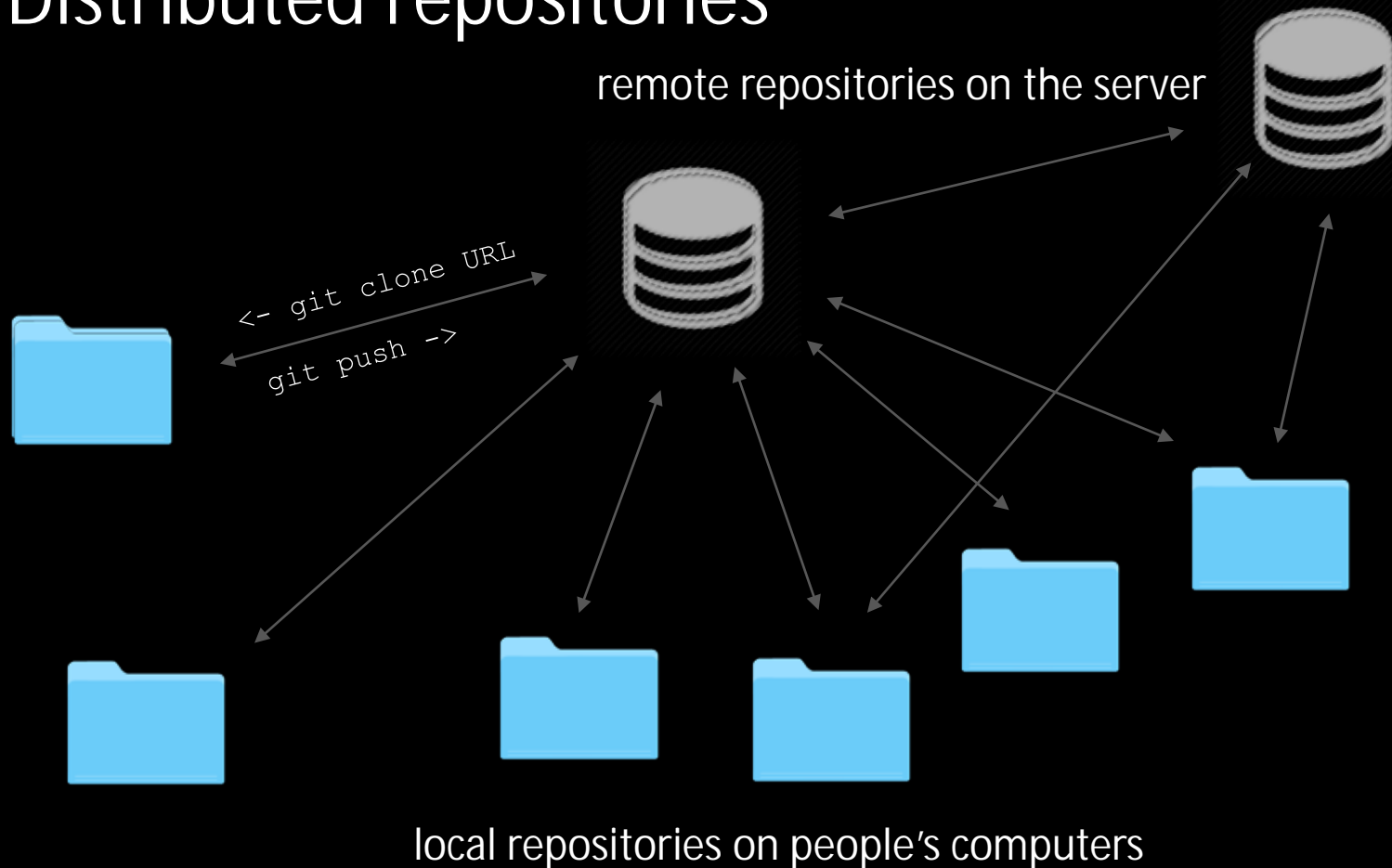
Traditional delta storage – SVN, CVS

Snapshot with COW – GIT



What is git?

- Distributed repositories

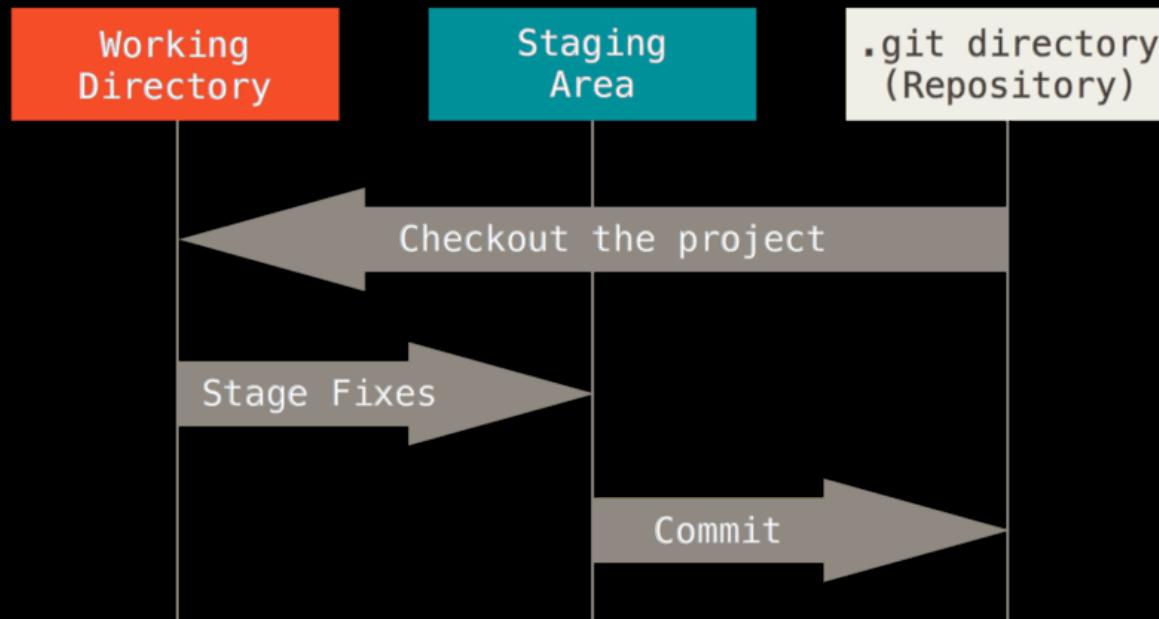


The three states

- three main states that your files can reside in:
 - `modified`
 - `staged`
 - `committed`
- `modified` means that you have changed the file but have not committed it to your database yet.
- `staged` means that you have marked a modified file in its current version to go into your next commit snapshot.
- `committed` means that the data is safely stored in your local database.

The three sections

- Three main sections of a Git project:
 - the working tree
 - the staging area
 - and the git directory



The usual workflow

- `.git` directory stores the metadata and object database for your project
 - most important part of Git, copied when you clone a repository
- The basic git workflow:
 1. modify files in your working tree
 2. check the status (`git status`)
 3. selectively stage the changes (files) you want as part of your next commit (`git add file`) which adds them to the staging area
 4. do the commit (`git commit -m 'message'`) which permanently stores the files from the staging area in a snapshot inside your `.git` directory
- when a file is in the `.git` directory is considered **committed**.
- if the file is modified and was added to the staging area, it is **staged**.
- if a file was changed since it was checked out but was not staged, it is **modified**.

Configuring

- Configure – tell git who you are and other things

```
git config --list --show-origin
```

```
git config --global user.name "John Doe"
```

```
git config --global user.email john@doe.com
```

```
git config --global core.editor emacs
```

```
git config --global core.editor
```

```
    "' C: /Program Files/Notepad++/notepad++.exe'  
    -multiInst -notabbar -nosession -noPlugin"
```

```
git config --global init.defaultBranch master
```

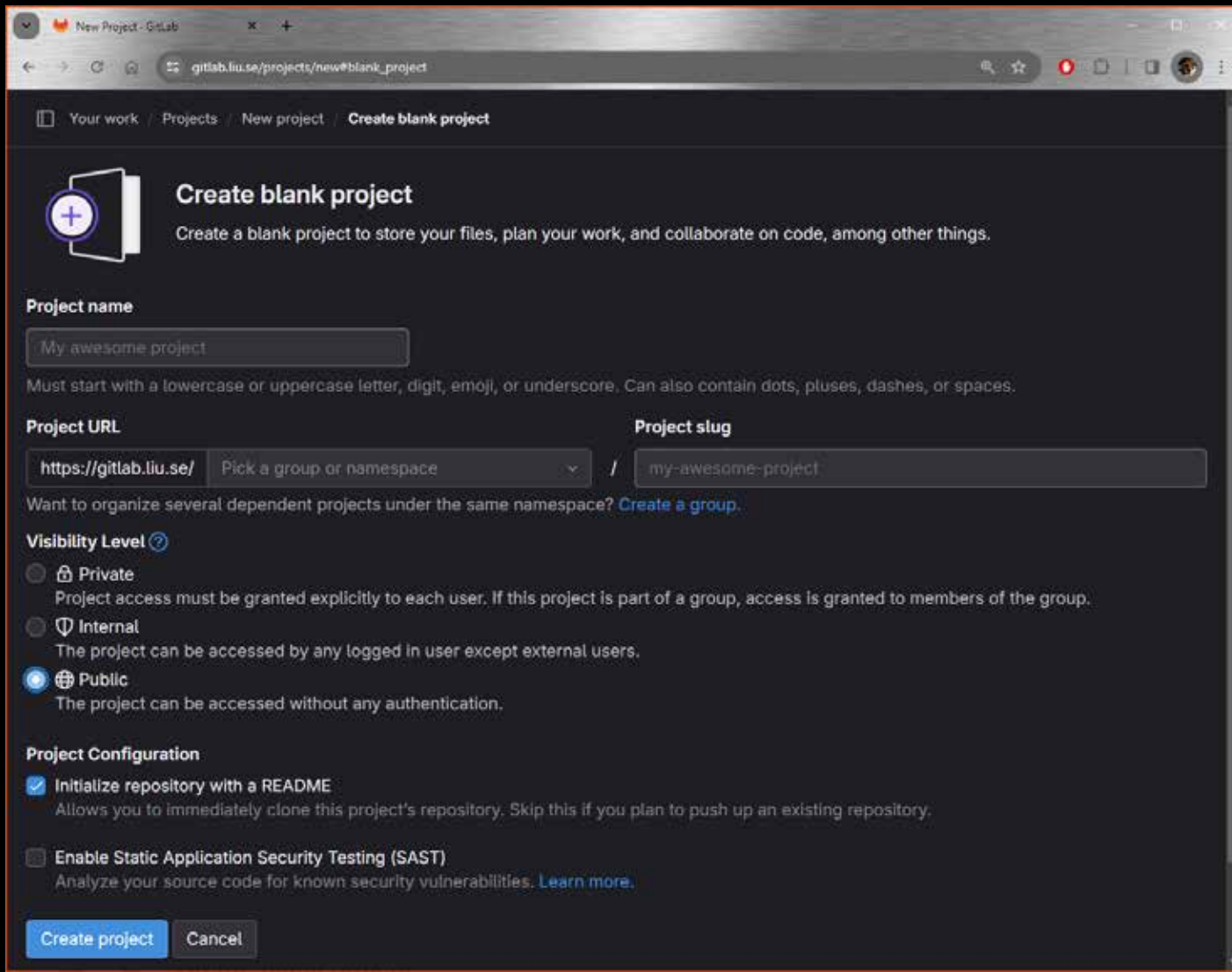
```
git config --global init.defaultBranch main
```


Configuring

- `git config --list --show-origin`

```
MINGW64/c:/home/adrpo33/tddd83-git-tutorial/tddd83-git
adrpo33@ida-0030 MINGW64 /c:/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git config --list --show-origin
file:C:/Program Files/Git/etc/gitconfig http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-bundle.crt
file:C:/Program Files/Git/etc/gitconfig http.sslbackend=openssl
file:C:/Program Files/Git/etc/gitconfig diff.astextplain.textconv=astextplain
file:C:/Program Files/Git/etc/gitconfig filter.lfs.clean=git-lfs clean -- %f
file:C:/Program Files/Git/etc/gitconfig filter.lfs.smudge=git-lfs smudge -- %f
file:C:/Program Files/Git/etc/gitconfig filter.lfs.process=git-lfs filter-process
file:C:/Program Files/Git/etc/gitconfig filter.lfs.required=true
file:C:/Program Files/Git/etc/gitconfig credential.helper=manager
file:C:/Program Files/Git/etc/gitconfig core.autocrlf=true
file:C:/Program Files/Git/etc/gitconfig core.fscache=true
file:C:/Program Files/Git/etc/gitconfig core.symlinks=false
file:C:/Program Files/Git/etc/gitconfig core.longpaths=true
file:C:/Program Files/Git/etc/gitconfig pull.rebase=false
file:C:/Program Files/Git/etc/gitconfig credential.https://dev.azure.com.usehttppath=true
file:C:/Program Files/Git/etc/gitconfig init.defaultbranch=master
file:C:/home/adrpo33/.gitconfig user.email=adrian.pop@liu.se
file:C:/home/adrpo33/.gitconfig user.name=Adrian Pop
file:C:/home/adrpo33/.gitconfig user.signingkey=34598CD8A9EFC1BE
file:C:/home/adrpo33/.gitconfig core.editor=C:/OHDDev/tools/msys/usr/bin/e-git.sh
file:C:/home/adrpo33/.gitconfig core.eol=lf
file:C:/home/adrpo33/.gitconfig core.autocrlf=input
file:C:/home/adrpo33/.gitconfig filesystem.AdoptOpenJDK|12.0.1|206402386.timestampresolution=11992 microseconds
file:C:/home/adrpo33/.gitconfig filesystem.AdoptOpenJDK|12.0.1|206402386.minracynanoseconds=0
file:C:/home/adrpo33/.gitconfig filesystem.AdoptOpenJDK|13|206402386.timestampresolution=29920 microseconds
file:C:/home/adrpo33/.gitconfig filesystem.AdoptOpenJDK|13|206402386.minracynanoseconds=0
file:C:/home/adrpo33/.gitconfig filesystem.Eclipse OpenJ9|11.0.5|206402386.timestampresolution=1018 microseconds
file:C:/home/adrpo33/.gitconfig filesystem.Eclipse OpenJ9|11.0.5|206402386.minracynanoseconds=0
file:C:/home/adrpo33/.gitconfig filesystem.Eclipse OpenJ9|11.0.6-121158955.timestampresolution=1001 microseconds
file:C:/home/adrpo33/.gitconfig filesystem.Eclipse OpenJ9|11.0.6-121158955.minracynanoseconds=0
file:C:/home/adrpo33/.gitconfig filesystem.Eclipse OpenJ9|11.0.6|206402386.timestampresolution=1033 microseconds
file:C:/home/adrpo33/.gitconfig filesystem.Eclipse OpenJ9|11.0.6|206402386.minracynanoseconds=0
file:C:/home/adrpo33/.gitconfig filesystem.AdoptOpenJDK|11.0.8|206402386.timestampresolution=7978 microseconds
file:C:/home/adrpo33/.gitconfig filesystem.AdoptOpenJDK|11.0.8|206402386.minracynanoseconds=0
file:C:/home/adrpo33/.gitconfig filesystem.AdoptOpenJDK|14.0.2|206402386.timestampresolution=1002 microseconds
file:C:/home/adrpo33/.gitconfig filesystem.AdoptOpenJDK|14.0.2|206402386.minracynanoseconds=0
file:C:/home/adrpo33/.gitconfig credential.helper=manager-core
file:C:/home/adrpo33/.gitconfig filesystem.AdoptOpenJDK|16.0.1|206402386.timestampresolution=1007 microseconds
file:C:/home/adrpo33/.gitconfig filesystem.AdoptOpenJDK|16.0.1|206402386.minracynanoseconds=0
file:C:/home/adrpo33/.gitconfig credential.https://gitlab.liu.se.provider=generic
file:C:/home/adrpo33/.gitconfig filesystem.AdoptOpenJDK|11.0.11|206402386.timestampresolution=10003 microseconds
file:C:/home/adrpo33/.gitconfig filesystem.AdoptOpenJDK|11.0.11|206402386.minracynanoseconds=0
file:C:/home/adrpo33/.gitconfig credential.https://collaborating.tuhh.de.provider=generic
file:C:/home/adrpo33/.gitconfig credential.https://openmodelica.org.provider=generic
file:.git/config core.repositoryformatversion=0
file:.git/config core.filemode=false
file:.git/config core.bare=false
file:.git/config core.logallrefupdates=true
file:.git/config core.symlinks=false
file:.git/config core.ignorecase=true
adrpo33@ida-0030 MINGW64 /c:/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$
```

Creating a repository – gitlab



The screenshot shows the 'Create blank project' page in the GitLab web interface. The browser address bar shows 'gitlab.liu.se/projects/new@blank_project'. The breadcrumb trail is 'Your work / Projects / New project / Create blank project'. The main heading is 'Create blank project' with a sub-heading 'Create a blank project to store your files, plan your work, and collaborate on code, among other things.' The form includes a 'Project name' field with the value 'My awesome project'. Below it is a note: 'Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.' The 'Project URL' section shows 'https://gitlab.liu.se/' followed by a dropdown menu 'Pick a group or namespace' and a 'Project slug' field with the value 'my-awesome-project'. A link 'Want to organize several dependent projects under the same namespace? Create a group.' is present. The 'Visibility Level' section has three radio buttons: 'Private' (unselected), 'Internal' (unselected), and 'Public' (selected). The 'Project Configuration' section has two checkboxes: 'Initialize repository with a README' (checked) and 'Enable Static Application Security Testing (SAST)' (unchecked). At the bottom are 'Create project' and 'Cancel' buttons.

Create blank project
Create a blank project to store your files, plan your work, and collaborate on code, among other things.

Project name
My awesome project
Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.

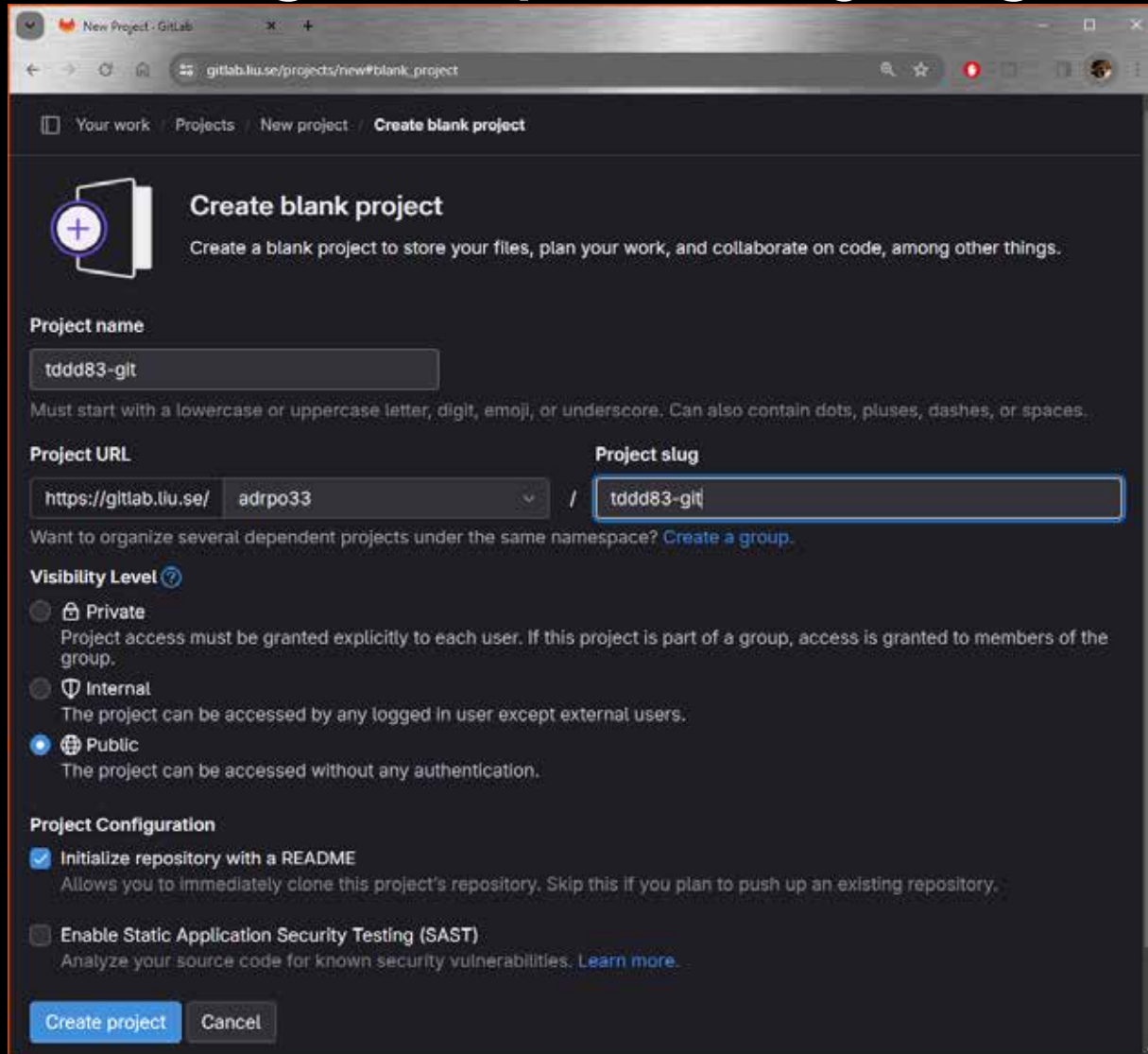
Project URL **Project slug**
https://gitlab.liu.se/ Pick a group or namespace / my-awesome-project
Want to organize several dependent projects under the same namespace? [Create a group.](#)

Visibility Level [?](#)
 Private
Project access must be granted explicitly to each user. If this project is part of a group, access is granted to members of the group.
 Internal
The project can be accessed by any logged in user except external users.
 Public
The project can be accessed without any authentication.

Project Configuration
 Initialize repository with a README
Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.
 Enable Static Application Security Testing (SAST)
Analyze your source code for known security vulnerabilities. [Learn more.](#)

[Create project](#) [Cancel](#)

Creating a repository – gitlab



The screenshot shows the 'Create blank project' page in GitLab. The browser address bar shows 'gitlab.ltu.se/projects/new#blank_project'. The page title is 'Create blank project' with a sub-header 'Create a blank project to store your files, plan your work, and collaborate on code, among other things.' The form includes a 'Project name' field with 'tddd83-git', a 'Project URL' dropdown set to 'https://gitlab.ltu.se/ adrho33', and a 'Project slug' field with 'tddd83-git'. The 'Visibility Level' section has 'Public' selected. The 'Project Configuration' section has 'Initialize repository with a README' checked and 'Enable Static Application Security Testing (SAST)' unchecked. At the bottom are 'Create project' and 'Cancel' buttons.

Create blank project
Create a blank project to store your files, plan your work, and collaborate on code, among other things.

Project name
tddd83-git
Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.

Project URL **Project slug**
https://gitlab.ltu.se/ adrho33 / tddd83-git

Want to organize several dependent projects under the same namespace? [Create a group.](#)

Visibility Level ?

- Private
Project access must be granted explicitly to each user. If this project is part of a group, access is granted to members of the group.
- Internal
The project can be accessed by any logged in user except external users.
- Public
The project can be accessed without any authentication.

Project Configuration

- Initialize repository with a README
Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.
- Enable Static Application Security Testing (SAST)
Analyze your source code for known security vulnerabilities. [Learn more.](#)

[Create project](#) [Cancel](#)

```
git clone git@gitlab.liu.se:adrpo33/tddd83-git.git
```

Creating a repository – gitlab

The screenshot shows the GitLab interface for a newly created repository named 'tddd83-git'. A blue notification banner at the top states 'Project 'tddd83-git' was successfully created.' Below this, the repository name 'tddd83-git' is displayed with icons for notifications, stars (0), and forks (0). The repository statistics show '1 Commit', '1 Branch', '0 Tags', and '3 KIB Project Storage'. An 'Initial commit' by 'Adrian Pop' is shown with a commit hash of '26671c26'. The interface includes a navigation bar with 'main' and 'tddd83-git' branches, and a 'Code' dropdown menu. Below the navigation bar are buttons for 'README', 'Add LICENSE', 'Add CHANGELOG', 'Add CONTRIBUTING', and 'Enable Auto DevOps'. There are also buttons for 'Add Kubernetes cluster', 'Set up CI/CD', and 'Configure Integrations'. A table lists the repository's files:

Name	Last commit	Last update
README.md	Initial commit	just now

Below the table, the repository name 'tddd83-git' and the section 'Getting started' are visible.

This screenshot shows the cloning options for the repository. The 'Clone with SSH' option is selected and circled in red, with the URL 'git@gitlab.liu.se:adrpo33/tddd83-git.git' also circled. The 'Clone with HTTPS' option is also visible with its URL. Below these are options to 'Open in your IDE' (Visual Studio Code and IntelliJ IDEA) and 'Download source code' (zip, tar.gz, tar.bz2, tar).

Clone with SSH
git@gitlab.liu.se:adrpo33/tddd83-git.git

Clone with HTTPS
https://gitlab.liu.se/adrpo33/tddd83-git.git

Open in your IDE

- Visual Studio Code (SSH)
- Visual Studio Code (HTTPS)
- IntelliJ IDEA (SSH)
- IntelliJ IDEA (HTTPS)

Download source code

- zip
- tar.gz
- tar.bz2
- tar

Creating a repository – gitlab

The screenshot shows a web browser window displaying the GitLab interface for a newly created repository named 'tddd83-git'. The browser's address bar shows the URL 'gitlab.ltu.se/adrho33/tddd83-git'. A blue notification banner at the top states 'Project 'tddd83-git' was successfully created.' Below this, the repository name 'tddd83-git' is displayed with a globe icon, and statistics for 'Star 0' and 'Fork 0'. A red arrow points from the 'Fork 0' button to a callout box. The repository's initial commit is shown as 'Initial commit' by 'Adrian Pop' with a commit hash of '26671c26'. The interface includes navigation tabs for 'main', 'tddd83-git', and '+', along with buttons for 'History', 'Find file', 'Edit', and 'Code'. Below these are options to 'Add LICENSE', 'Add CHANGELOG', 'Add CONTRIBUTING', and 'Enable Auto DevOps'. A table lists the repository's files, showing 'README.md' with its last commit and update. The 'README.md' content is partially visible, starting with 'tddd83-git' and 'Getting started'.

Project 'tddd83-git' was successfully created.

tddd83-git 🔔 ☆ Star 0 🍴 Fork 0 ⋮

1 Commit 1 Branch 0 Tags 3 KIB Project Storage

Initial commit
Adrian Pop authored just now 26671c26

main tddd83-git / + History Find file Edit Code

README Add LICENSE Add CHANGELOG Add CONTRIBUTING + Enable Auto DevOps

Add Kubernetes cluster + Set up CI/CD Configure Integrations

Name	Last commit	Last update
README.md	Initial commit	just now

README.md

tddd83-git

Getting started

To make it easy for you to get started with GitLab, here's a list of recommended next steps:

Create new fork

Fork 0

Creating a repository via fork will copy the repository to your account or your project and then you can clone it locally from there

Creating a repository – github

New repository

github.com/new

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Repository template

No template -

Start your repository with a template repository's contents.

Owner * Repository name *

adrpo /

Great repository names are short and memorable. Need inspiration? How about [literate-pancake](#) ?

Description (optional)

Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None ▾

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None -

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

ⓘ You are creating a public repository in your personal account.

Create repository

Creating a repository – github

New repository

github.com/new

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Required fields are marked with an asterisk (*).

Repository template

No template -

Start your repository with a template repository's contents.

Owner * adrpo - / Repository name * tddd83-git

✔ tddd83-git is available.

Great repository names are short and memorable. Need inspiration? How about [literate-pancake](#) ?

Description (optional)

A repository to learn git

Public
Anyone on the internet can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

gitignore template: None -

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

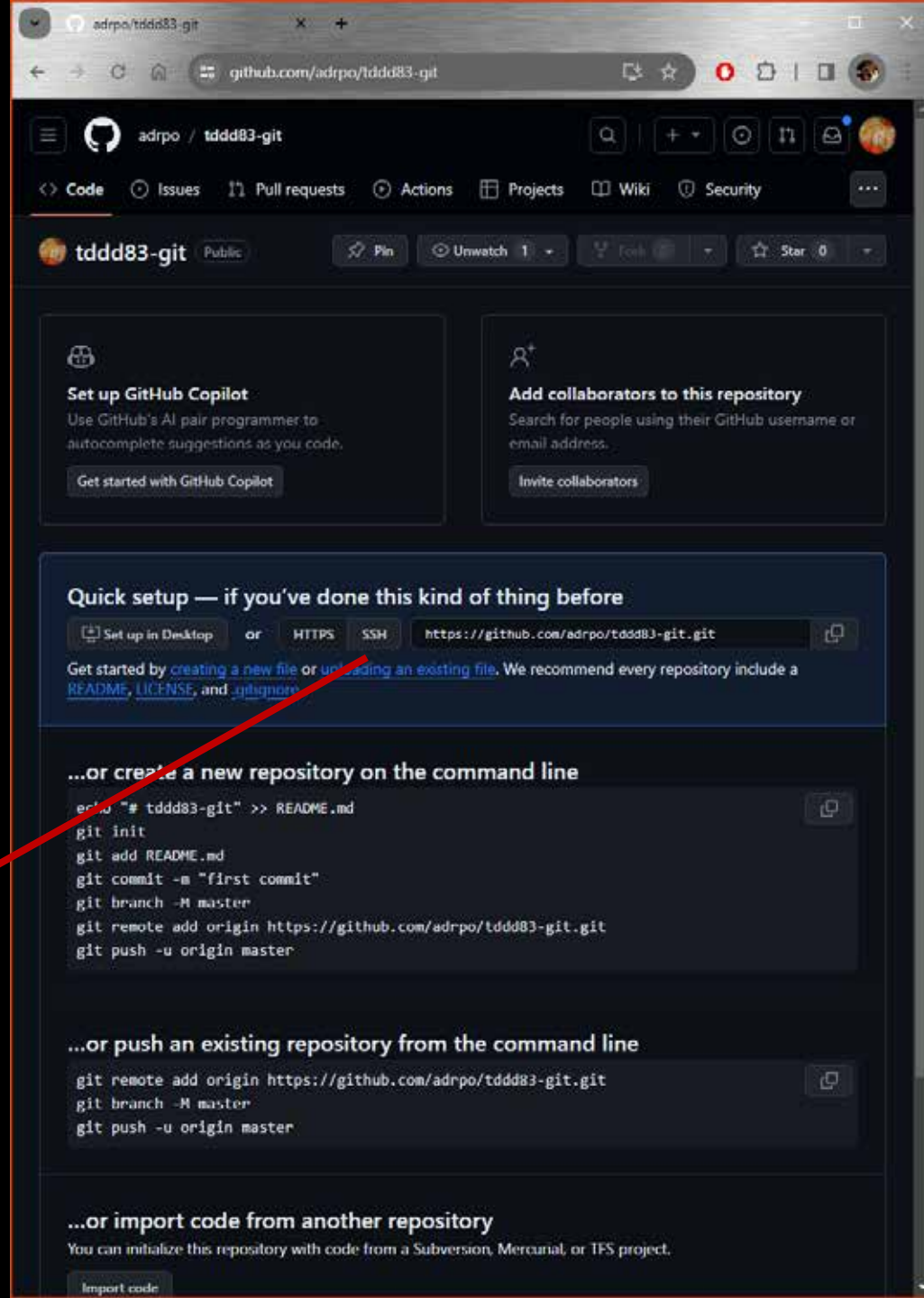
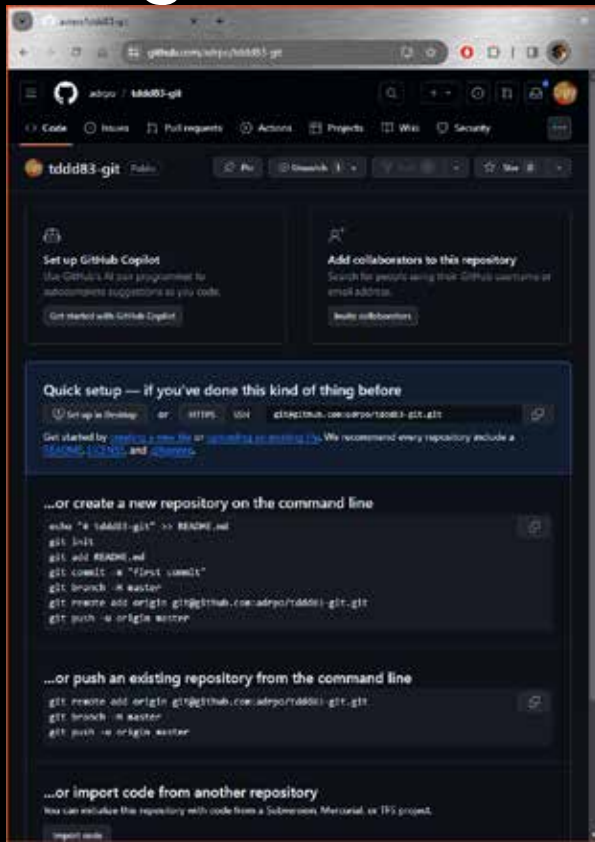
License: None -

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

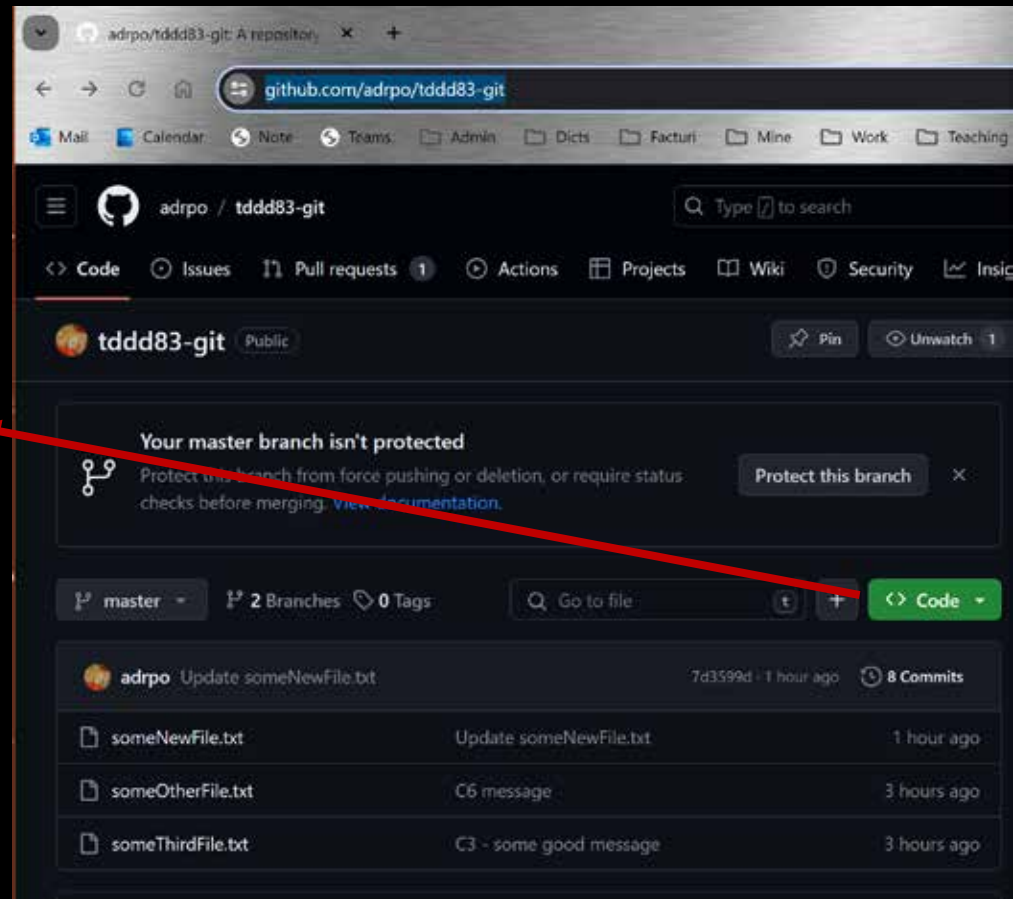
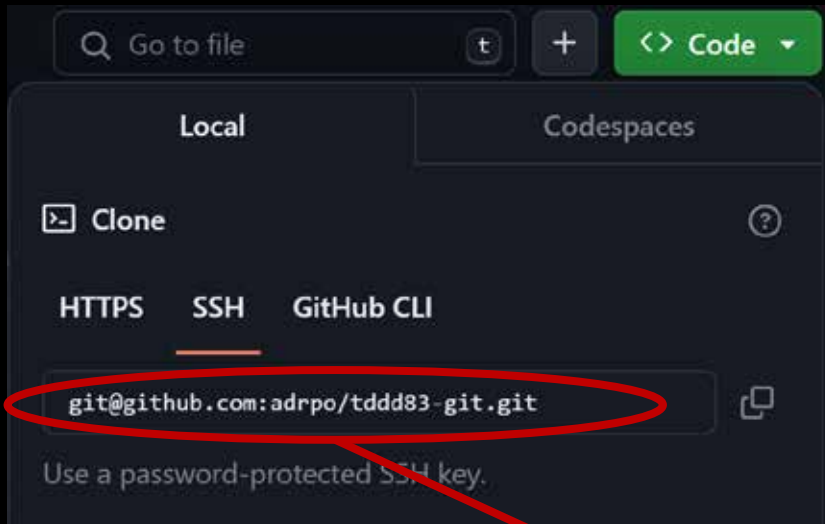
ⓘ You are creating a public repository in your personal account.

[Create repository](#)

Creating a repository – github



Creating a repository – github



```
git clone git@github.com:adrpo/tddd83-git.git
```

Creating a repository - local

- Init – initialize a new repository from a folder

```
mkdir tddd83-git
```

```
cd tddd83-git
```

```
git init
```

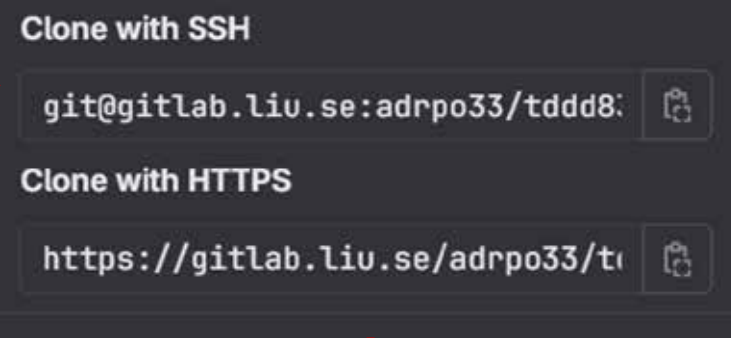


```
M /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
adrpo33@ida-0030 UCRT64 /c/home/adrpo33
# mkdir tddd83-git-tutorial
adrpo33@ida-0030 UCRT64 /c/home/adrpo33
# cd tddd83-git-tutorial/
adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial
# mkdir tddd83-git
  cd tddd83-git
  git init
Initialized empty Git repository in C:/home/adrpo33/tddd83-git-tutorial/tddd83-git/.git/
adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# |
```

on gitlab for your project, click
github



Cloning



- Clone – copy a remote repository locally

```
git clone URL [LOCAL_REPO_NAME]
```

URL can be git+ssh, https, path

```
git clone git@gitlab.liu.se:adrpo33/tddd83-git.git
```

after clone

git remote -v

will list the URL as origin

```
git clone https://gitlab.liu.se/adrpo33/tddd83-git.git
```

```
git clone /local/path/to/tddd83-git.git
```

git clone with a new name

```
git clone git@gitlab.liu.se:adrpo33/tddd83-git.git newName
```

```
git clone https://gitlab.liu.se/adrpo33/tddd83-git.git newName
```

```
git clone /local/path/to/tddd83-git.git newName
```

Status & Log

- Status – get the status of the working copy
`git status`

```
M /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
cd tddd83-git
git init
Initialized empty Git repository in C:/home/adrpo33/tddd83-git-tutorial/tddd83-git/.git/

adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)

adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# x|
```

```
M /c/home/adrpo33/tddd83-git-tutorial/tddd83-git

adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# cat > someNewFile.txt
Have some text

adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# git status
On branch master

No commits yet

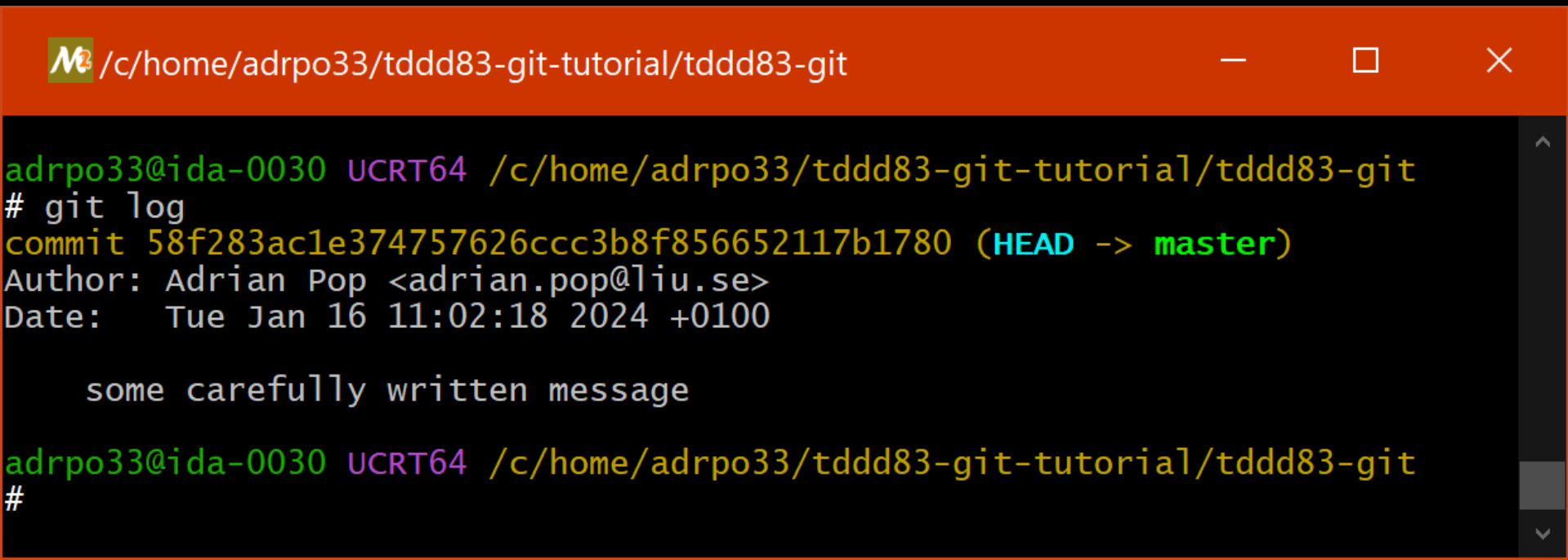
Untracked files:
  (use "git add <file>..." to include in what will be committed)
  someNewFile.txt

nothing added to commit but untracked files present (use "git add" to track)

adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# |
```

Status & Log

- Log – see what commits are in your checkout
`git log`

A terminal window with a dark background and a red title bar. The title bar contains a small icon, the path `/c/home/adrpo33/tddd83-git-tutorial/tddd83-git`, and window control buttons (minimize, maximize, close). The terminal shows the command `git log` and its output, including commit hash, author, date, and message.

```
adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# git log
commit 58f283ac1e374757626ccc3b8f856652117b1780 (HEAD -> master)
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 11:02:18 2024 +0100

    some carefully written message

adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
#
```

Checkout & Branching

- Checkout – get a specific version (snapshot) from `.git` in the working tree

```
git checkout master
```

```
git checkout SomeNiceFeature
```



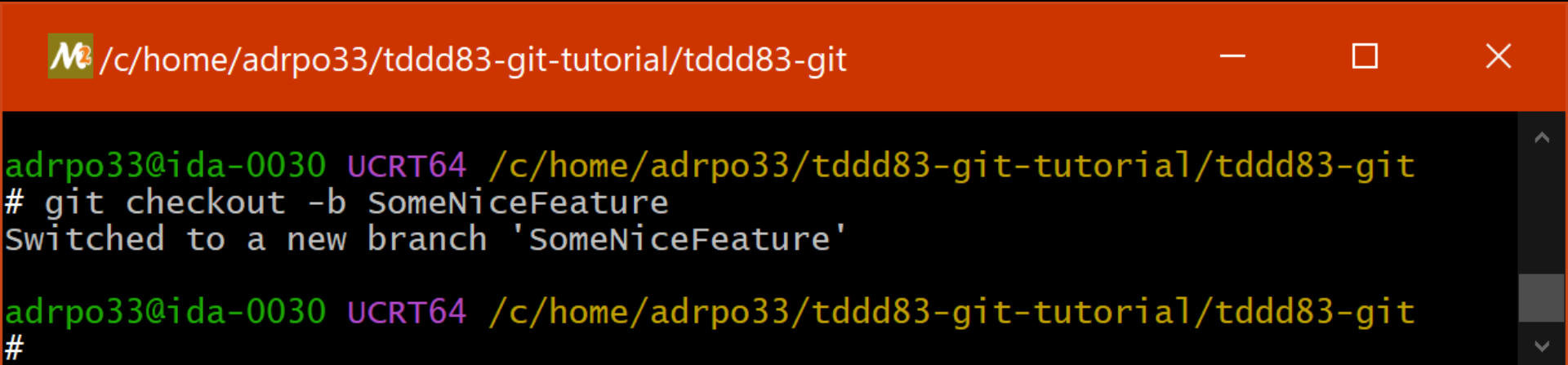
```
Me /c/home/adrpo33/tddd83-git-tutorial/tddd83-git - □ ×  
adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git  
# git checkout SomeNiceFeature  
Switched to branch 'SomeNiceFeature'  
adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git  
# git checkout master  
Switched to branch 'master'  
adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git  
# git checkout SomeNiceFeature  
Switched to branch 'SomeNiceFeature'  
adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git  
#
```


Checkout & Branching

- Branching – create a new branch from the current checkout branch

```
git checkout -b NEW_BRANCH_NAME
```

```
git checkout -b SomeNewNiceFeature
```

A terminal window with a dark background and a red title bar. The title bar contains a small icon and the path "/c/home/adrpo33/tddd83-git-tutorial/tddd83-git". The terminal text shows the execution of the "git checkout -b SomeNiceFeature" command, resulting in a message "Switched to a new branch 'SomeNiceFeature'".

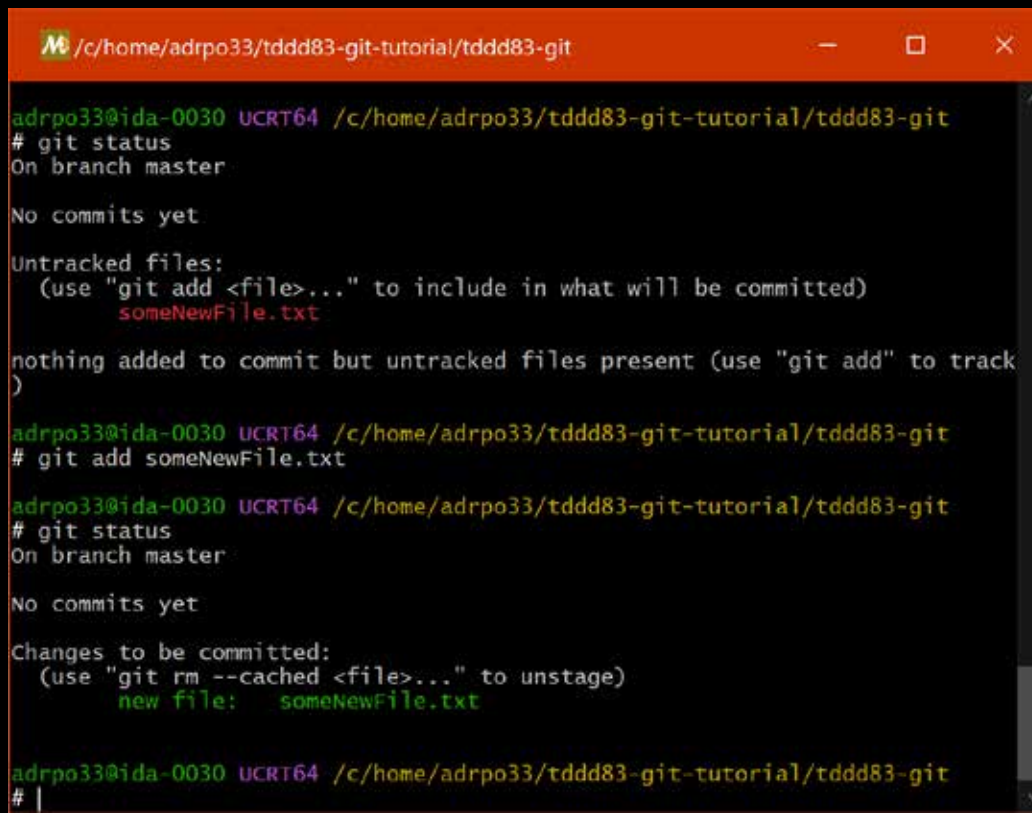
 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git

```
adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# git checkout -b SomeNiceFeature
Switched to a new branch 'SomeNiceFeature'
```

```
adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
#
```

Adding files & Committing

- Add – add changes you want to commit -> stage
`git add FILE_NAME`



```
adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  someNewFile.txt

nothing added to commit but untracked files present (use "git add" to track
)

adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# git add someNewFile.txt

adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# git status
On branch master

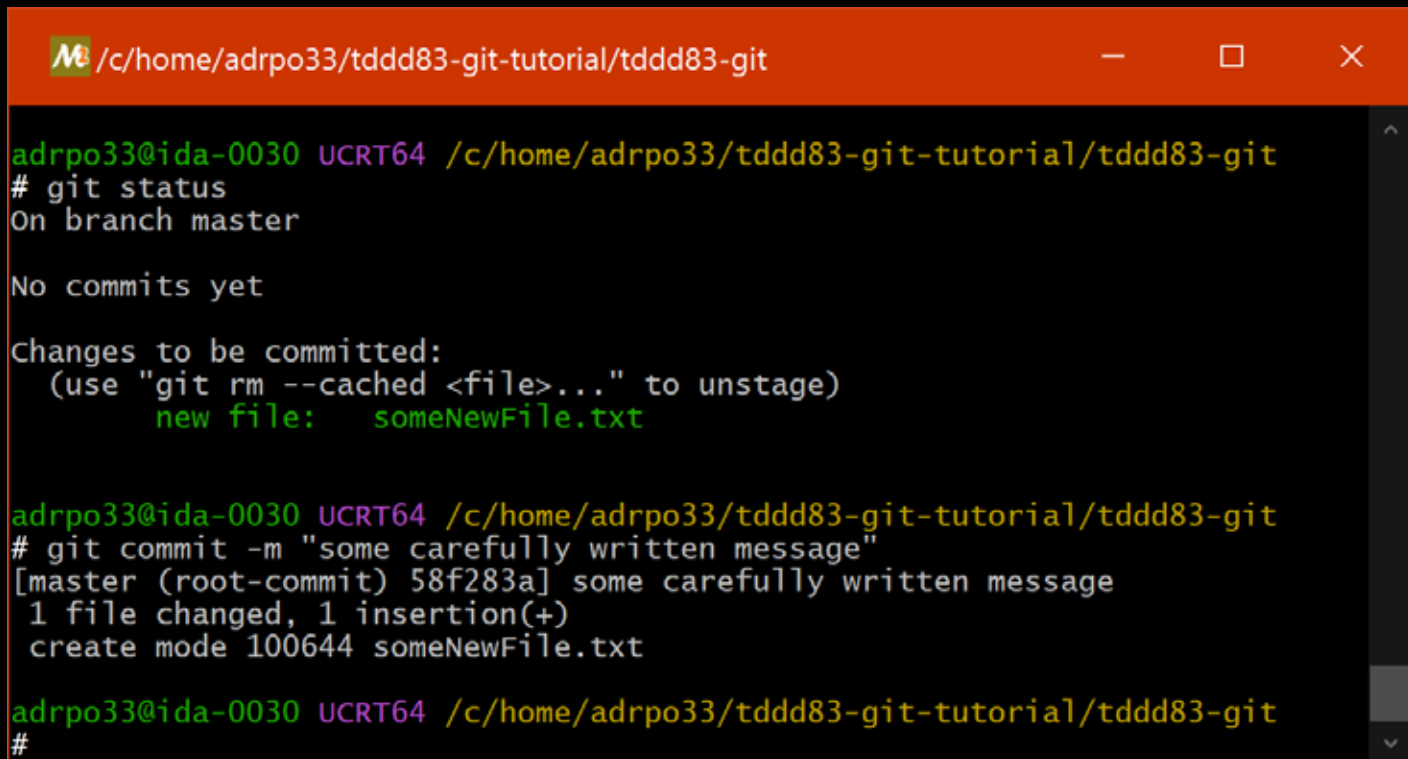
No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
  new file:   someNewFile.txt

adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# |
```


Adding files & Committing

- Commit – commit your staged files -> .git directory
`git commit -m "some carefully written message"`



```
adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# git status
On branch master

No commits yet

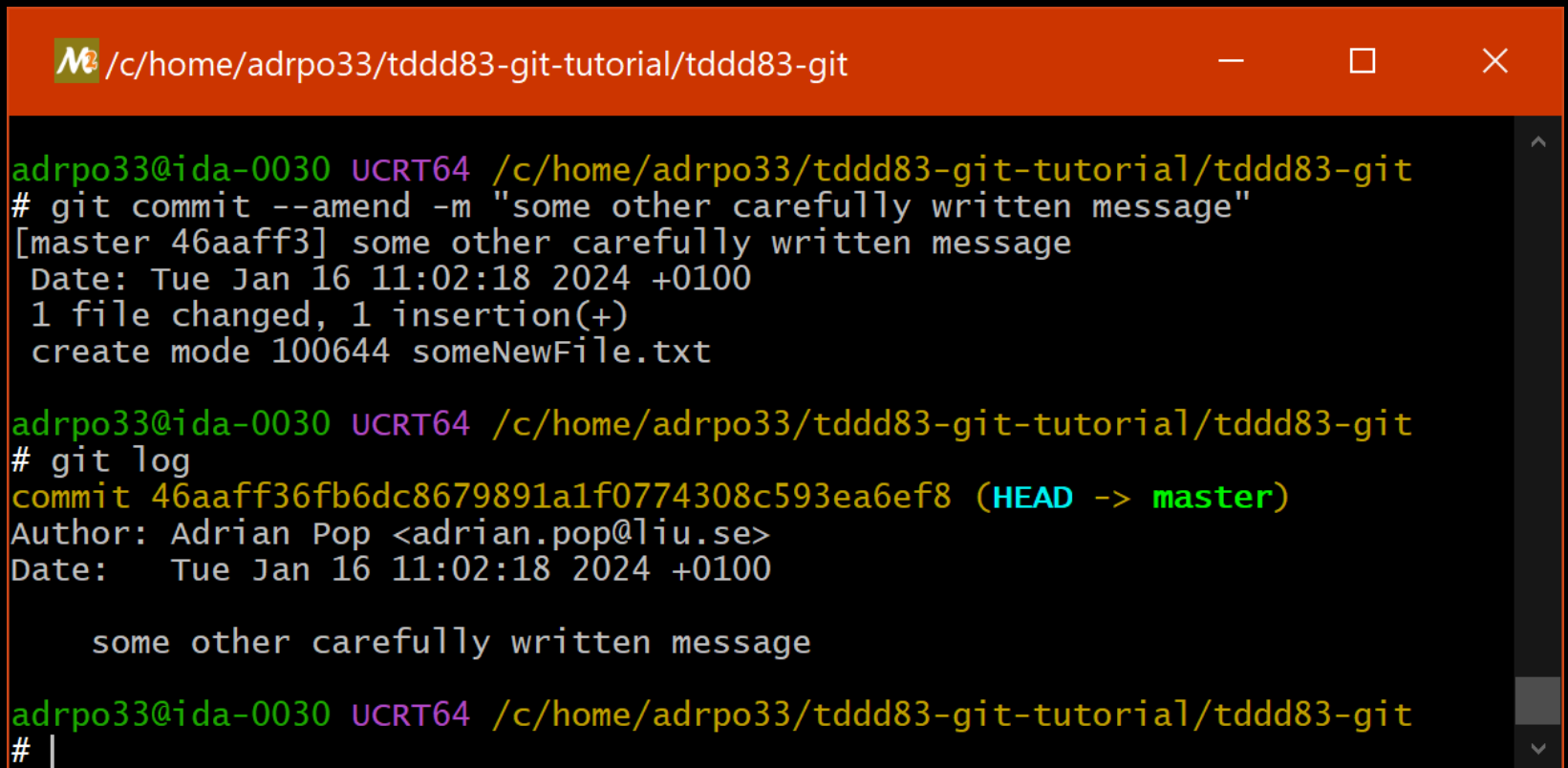
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   someNewFile.txt

adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# git commit -m "some carefully written message"
[master (root-commit) 58f283a] some carefully written message
 1 file changed, 1 insertion(+)
 create mode 100644 someNewFile.txt

adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
#
```

Adding files & Committing

- Commit – amend the commit if you did a mistake
`git commit -amend`



```
adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# git commit --amend -m "some other carefully written message"
[master 46aaff3] some other carefully written message
Date: Tue Jan 16 11:02:18 2024 +0100
1 file changed, 1 insertion(+)
create mode 100644 someNewFile.txt

adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# git log
commit 46aaff36fb6dc8679891a1f0774308c593ea6ef8 (HEAD -> master)
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 11:02:18 2024 +0100

    some other carefully written message

adrpo33@ida-0030 UCRT64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git
# |
```

Merging & Conflict handling

1. Init + commit C1 (someNewFile.txt) -> master
2. We branched SomeNiceFeature from master (C1 commit) via git checkout -b SomeNiceFeature

SomeNiceFeature

/

Init --- C1 --- master

```
MINGW64:/c/home/adrpo33/tddd83-git-tutorial/tddd83-git
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git status
On branch master
nothing to commit, working tree clean

adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git log
commit 46aaff36fb6dc8679891a1f0774308c593ea6ef8 (HEAD -> master, SomeNiceFeature)
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 11:02:18 2024 +0100

    some other carefully written message

adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git checkout SomeNiceFeature
Switched to branch 'SomeNiceFeature'

adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeNiceFeature)
$ git status
On branch SomeNiceFeature
nothing to commit, working tree clean

adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeNiceFeature)
$ git log
commit 46aaff36fb6dc8679891a1f0774308c593ea6ef8 (HEAD -> SomeNiceFeature, master)
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 11:02:18 2024 +0100

    some other carefully written message

adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeNiceFeature)
$ |
```

Merging & Conflict handling

- Let's diverge the branches `master` and `SomeNiceFeature`!

```
git checkout master
```

```
echo "Some Text Here" > someOtherFile.txt
```

```
git add someOtherFile.txt
```

```
git commit -m "C2 - some nice message"
```

MINGW64:/c:/home/adrpo33/tddd83-git-tutorial/tddd83-git

```
adrpo33@ida-0030 MINGW64 /c:/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ echo "Some Text Here" > someOtherFile.txt
```

```
adrpo33@ida-0030 MINGW64 /c:/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
  someOtherFile.txt
```

`SomeNiceFeature`

`Init --- C1 --- C2 --- master`

```
nothing added to commit but untracked files present (use "git add" to track)
```

```
adrpo33@ida-0030 MINGW64 /c:/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git add someOtherFile.txt
```

```
adrpo33@ida-0030 MINGW64 /c:/home/adrpo33/tddd83-git-tutorial/tddd83-git
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
  new file:   someOtherFile.txt
adrpo33@ida-0030 MINGW64 /c:/home/adrpo33/tddd83-git-tutorial/tddd83-git
$ git commit -m "C2 - some nice message"
[master e533b30] C2 - some nice message
 1 file changed, 1 insertion(+)
 create mode 100644 someOtherFile.txt
```

Merging & Conflict handling

- Let's diverge the branches `master` and `SomeNiceFeature`!

```
git checkout SomeNiceFeature
```

```
echo "Some New Text Here" > someThirdFile.txt
```

```
git add someThirdFile.txt
```

```
git commit -m "C3 - some good message"
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git checkout SomeNiceFeature
Switched to branch 'SomeNiceFeature'
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeNiceFeature)
$ echo "Some New Text Here" > someThirdFile.txt
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeNiceFeature)
$ git status
On branch SomeNiceFeature
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    someThirdFile.txt
```

```
          C3 --- SomeNiceFeature
          /
Init --- C1 --- C2 --- master
```

```
nothing added to commit but untracked files present (use "git add" to track)
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeNiceFeature)
$ git add someThirdFile.txt
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeNiceFeature)
$ git status
On branch SomeNiceFeature
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   someThirdFile.txt
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeNiceFeature)
$ git commit -m "C3 - some good message"
[SomeNiceFeature 59d8911] C3 - some good message
1 file changed, 1 insertion(+)
create mode 100644 someThirdFile.txt
```

Merging & Conflict handling

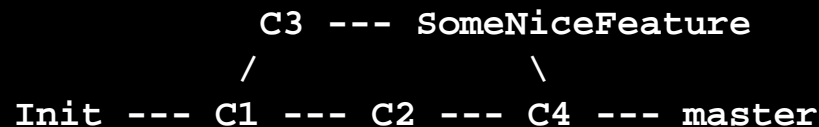
- Merge – Incorporates changes from the named commits (since the time their histories diverged from the current branch) into the current branch

```
git merge COMMIT
```

- Merging a branch into master/main

```
git checkout master
```

```
git merge SomeNiceFeature
```



Merging & Conflict handling

- Merging a branch into master/main
`git checkout master`
`git merge SomeNiceFeature`

```
      C3 --- SomeNiceFeature
     /           \
Init --- C1 --- C2 --- C4 --- master
```

```
MINGW64/c/home/adrpo33/tddd83-git-tutorial/tddd83-git
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git checkout SomeNiceFeature
Switched to branch 'SomeNiceFeature'

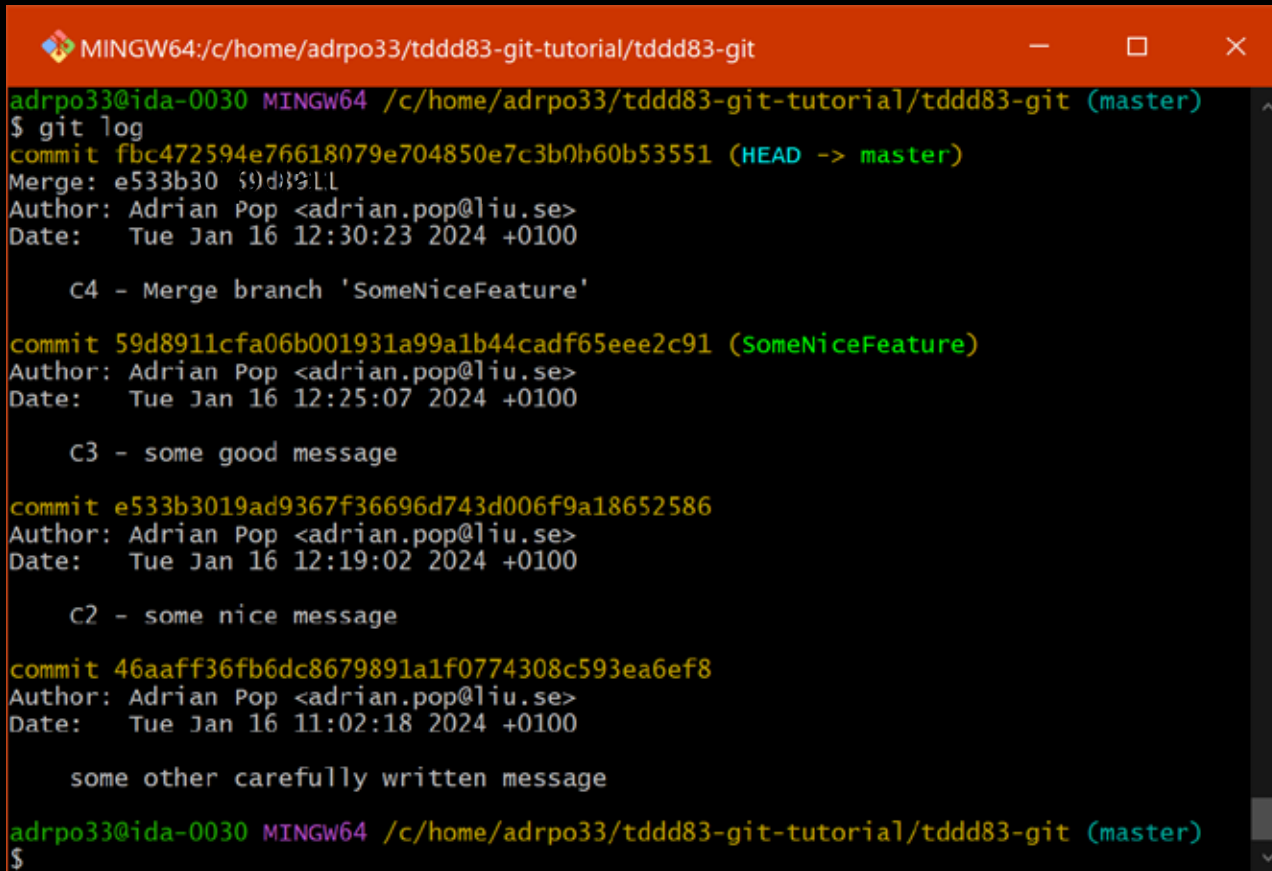
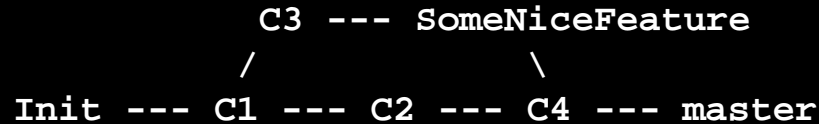
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeNiceFeature)
$ git checkout master
Switched to branch 'master'

adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git merge SomeNiceFeature
hint: waiting for your editor to close the file...
```

```
C:\home\adrpo33\tddd83-git-tutorial\tddd83-git\MERGE_MSG - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
MERGE_MSG
1 | C4 - Merge branch 'SomeNiceFeature'
2 | # Please enter a commit message to explain why this merge is necessary,
3 | # especially if it merges an updated upstream into a topic branch.
4 | #
5 | # Lines starting with '#' will be ignored, and an empty message aborts
6 | # the commit.
7 |
Normal text file length: 262 lines: 7 Ln: 1 Col: 6 Pos: 6 Unix (LF) UTF-8 IN
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git merge SomeNiceFeature
Merge made by the 'ort' strategy.
someThirdFile.txt | 1 +
1 file changed, 1 insertion(+)
create mode 100644 someThirdFile.txt
```

Merging & Conflict handling



```
MINGW64:/c/home/adrpo33/tddd83-git-tutorial/tddd83-git
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git log
commit fbc472594e76618079e704850e7c3b0b60b53551 (HEAD -> master)
Merge: e533b30 59d8911
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 12:30:23 2024 +0100

    C4 - Merge branch 'SomeNiceFeature'

commit 59d8911cfa06b001931a99a1b44cadf65eee2c91 (SomeNiceFeature)
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 12:25:07 2024 +0100

    C3 - some good message

commit e533b3019ad9367f36696d743d006f9a18652586
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 12:19:02 2024 +0100

    C2 - some nice message

commit 46aaff36fb6dc8679891a1f0774308c593ea6ef8
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 11:02:18 2024 +0100

    some other carefully written message

adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$
```


Merging & Conflict handling

- Let's diverge the branches `master` and `SomeConflict`!

```
git checkout master
```

```
git checkout -b SomeConflict
```

```
          C3 --- SomeNiceFeature          SomeConflict
         /                               \ /
Init --- C1 --- C2 ----- C4 --- master
```

```
echo "Some Newer Text Here" > someOtherFile.txt
```

```
git add someOtherFile.txt
```

```
git commit -m "C5 message"
```

```
          C3 --- SomeNiceFeature          C5 --- SomeConflict
         /                               \ /
Init --- C1 --- C2 ----- C4 --- master
```

Merging & Conflict handling

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git checkout -b SomeConflict
Switched to a new branch 'SomeConflict'
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeConflict)
$ echo "Some Newer Text Here" > someOtherFile.txt
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeConflict)
$ git commit -m "C5 message"
On branch SomeConflict
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
       modified:   someOtherFile.txt
```

```
no changes added to commit (use "git add" and/or "git commit -a")
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeConflict)
$ git add someOtherFile.txt
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeConflict)
$ git commit -m "C5 message"
[SomeConflict 568e915] C5 message
1 file changed, 1 insertion(+), 1 deletion(-)
```

Merging & Conflict handling

- Let's diverge the branches `master` and `SomeConflict`!

```
git checkout master
```



```
echo "Some Extra Newer Text Here" > someOtherFile.txt
```

```
git add someOtherFile.txt
```

```
git commit -m "C6 message"
```



Merging & Conflict handling

MINGW64:/c/home/adrpo33/tddd83-git-tutorial/tddd83-git

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (SomeConflict)
```

```
$ git checkout master  
Switched to branch 'master'
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)  
$ echo "Some Extra Newer Text Here" > someOtherFile.txt
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)  
$ git add someOtherFile.txt
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)  
$ git commit -m "C6 message"  
[master b4174e8] C6 message  
1 file changed, 1 insertion(+), 1 deletion(-)
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)  
$ |
```

Merging & Conflict handling

- Merging a branch into master/main with conflict
`git checkout master`
`git merge SomeConflict`



```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
```

```
$ git merge SomeConflict
```

```
Auto-merging someOtherFile.txt
```

```
CONFLICT (content): Merge conflict in someOtherFile.txt
```

```
Automatic merge failed; fix conflicts and then commit the result.
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master|MERGING)
```

```
$ git status
```

```
On branch master
```

```
You have unmerged paths.
```

```
(fix conflicts and run "git commit")
```

```
(use "git merge --abort" to abort the merge)
```

```
Unmerged paths:
```

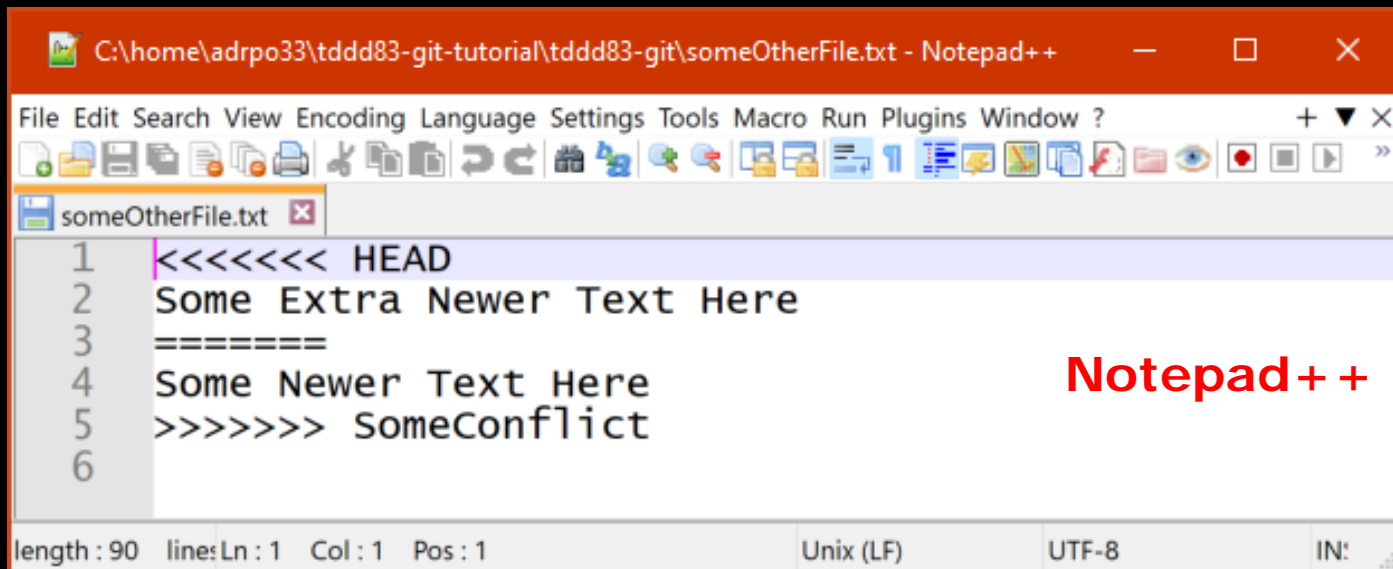
```
(use "git add <file>..." to mark resolution)
```

```
both modified: someOtherFile.txt
```

```
no changes added to commit (use "git add" and/or "git commit -a")
```

Merging & Conflict handling

- Merging a branch into master/main with conflict
`git merge SomeConflict`



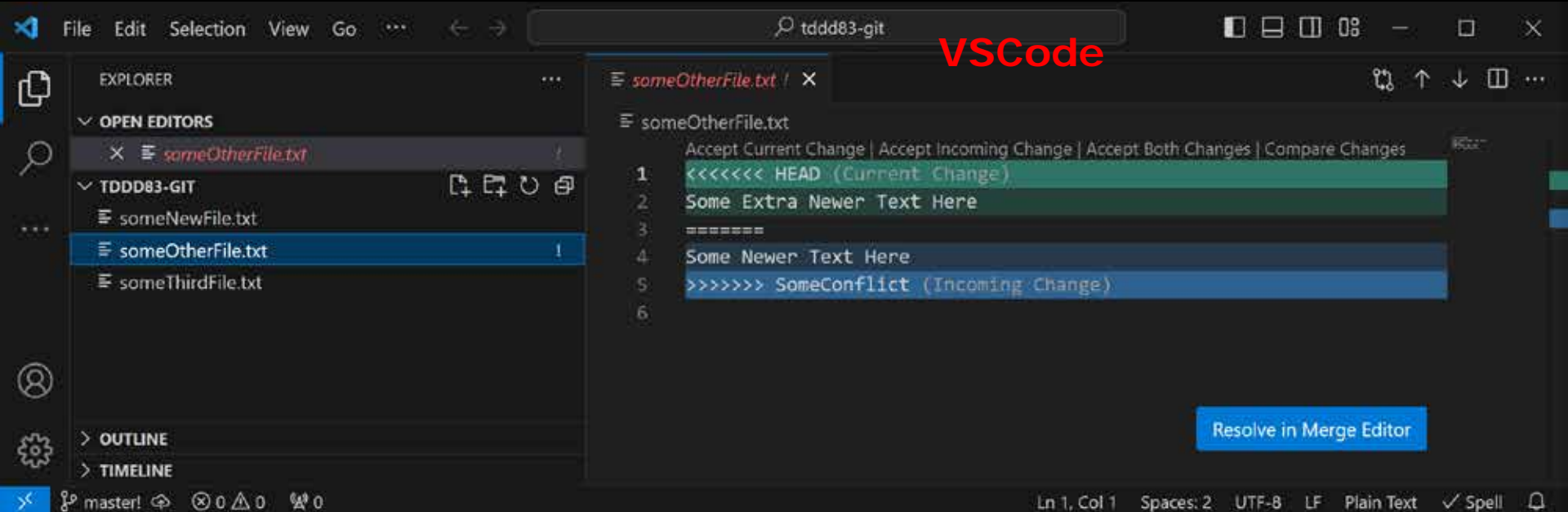
The screenshot shows a Notepad++ window with the file `someOtherFile.txt` open. The text content is as follows:

```
1 <<<<<<< HEAD
2 Some Extra Newer Text Here
3 =====
4 Some Newer Text Here
5 >>>>>> SomeConflict
6
```

The status bar at the bottom indicates: length: 90 line: Ln: 1 Col: 1 Pos: 1 Unix (LF) UTF-8 IN: .

Merging & Conflict handling

- Merging a branch into master/main with conflict
`git merge SomeConflict`



Merging & Conflict handling

- Merging a branch into master/main with conflict
`git merge SomeConflict`



- 4 ways to resolve the conflict using your preferred editor
 - Keep the text in between <<<< T1 =====
 - T1
 - Keep the text in between ===== T2 >>>>
 - T2
 - Keep both
 - T1
 - T2
 - Replace with anything else but remove all the lines containing markers
 - <<<<< ===== >>>>>

Merging & Conflict handling

- Merging a branch into master/main with conflict

```
git merge SomeConflict
```

```
      C3 --- SomeNiceFeature      C5 --- SomeConflict
      /                          \ /
Init --- C1 --- C2 ----- C4 --- C6 ---- master
```



```
git add someOtherFile.txt
```

```
git commit -m "C7 - resolved conflict"
```

```
      C3 --- SomeNiceFeature      C5 --- SomeConflict
      /                          \ /
Init --- C1 --- C2 ----- C4 --- C6 ---- C7 ---- master
```



Merging & Conflict handling

```
MINGW64:/c/home/adrpo33/tddd83-git-tutorial/tddd83-git
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master|MERGING)
$ git add someOtherFile.txt

adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master|MERGING)
$ git commit -m "C7 - resolved conflict"
[master 516c4d4] C7 - resolved conflict

adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git log
commit 516c4d4e6ea8272062fe04c9f0db5fba6bf5ce1 (HEAD -> master)
Merge: b4174e8 568e915
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 13:01:07 2024 +0100

    C7 - resolved conflict

commit b4174e83abf1528d44c2fdd8df00bd2847bb9533
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 12:52:23 2024 +0100

    C6 message

commit 568e915964cf72ffe6ed56b9e67df353b9459e38 (SomeConflict)
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 12:48:00 2024 +0100

    C5 message

commit fbc472594e76618079e704850e7c3b0b60b53551
Merge: e533b30 59d8911
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 12:30:23 2024 +0100

    C4 - Merge branch 'SomeNiceFeature'

commit 59d8911cfa06b001931a99a1b44cadf65eee2c91 (SomeNiceFeature)
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 12:25:07 2024 +0100

    C3 - some good message

commit e533b3019ad9367f36696d743d006f9a18652586
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 12:19:02 2024 +0100

    C2 - some nice message

commit 46aaff36fb0dc8679891a1f0774308c593ea6ef8
Author: Adrian Pop <adrian.pop@liu.se>
Date: Tue Jan 16 11:02:18 2024 +0100

    some other carefully written message

adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$
```

Push & Multiple remotes

- Get the available remotes (servers from where we cloned or servers where we want to push the changes)

```
git remote -v
```

- Default remotes depends on how you created the repo

```
git clone URL
```

```
# the URL is added as a remote called origin
```

```
git init
```

```
# No remotes for locally created repositories
```

- Add new remotes

```
git remote add origin git@gitlab.liu.se:adrpo33/tddd83-git.git
```

```
git remote add gh https://github.com/adrpo/tddd83-git.git
```

```
git remote -v
```

Push & Multiple remotes

```
git remote add origin git@gitlab.liu.se:adrpo33/tddd83-git.git  
git remote add gh https://github.com/adrpo/tddd83-git.git  
git remote -v
```

```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)  
$ git remote -v  
gh      https://github.com/adrpo/tddd83-git.git (fetch)  
gh      https://github.com/adrpo/tddd83-git.git (push)  
origin  git@gitlab.liu.se:adrpo33/tddd83-git.git (fetch)  
origin  git@gitlab.liu.se:adrpo33/tddd83-git.git (push)
```

Push & Multiple remotes

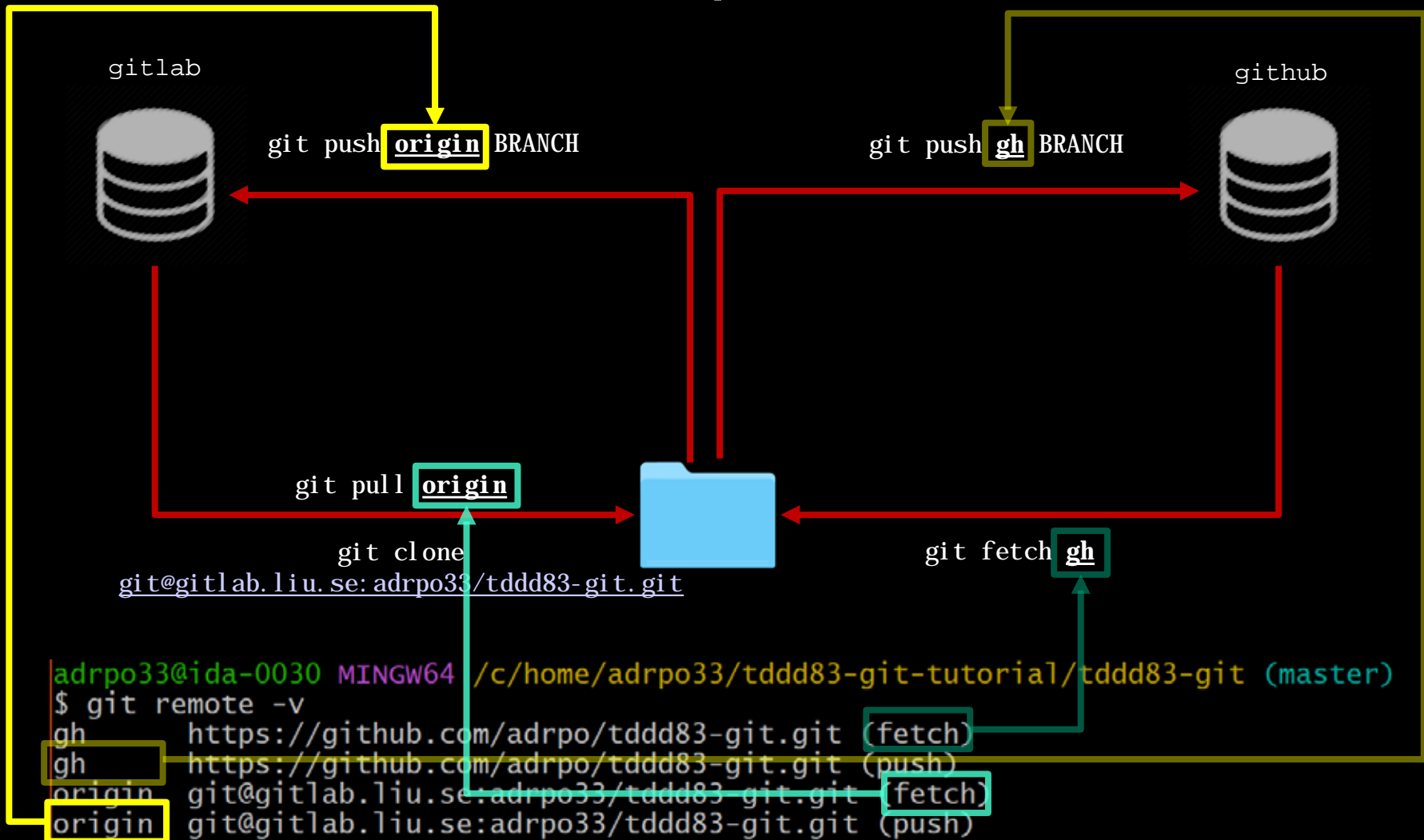
```
adrpo33@ida-0030 MINGW64 /c/home/adrpo33/tddd83-git-tutorial/tddd83-git (master)
$ git remote -v
gh          https://github.com/adrpo/tddd83-git.git (fetch)
gh          https://github.com/adrpo/tddd83-git.git (push)
origin     git@gitlab.liu.se:adrpo33/tddd83-git.git (fetch)
origin     git@gitlab.liu.se:adrpo33/tddd83-git.git (push)
```

- Push the local changes to the servers

```
git push origin master
```

```
git push gh SomeConflict
```

Push & Multiple remotes



Updating the repository

- If you created the repository locally (not via clone or fork) you need to select an upstream branch before you can do `git pull`

```
git checkout master
git branch --set-upstream-to=origin/master
git checkout SomeBranch
git branch --set-upstream-to=gh/SomeBranch
```

- Update with the latest from the server

```
git checkout master
git pull
```

- Fetch and merge from different server

```
git fetch gh
git checkout master
git merge gh/master
```

Updating the repository

- Fetch, checkout and merge from different server

```
# get the changes from the "gh" remote
```

```
git fetch gh
```

```
# checkout the branch SomeOtherBranch form "gh"
```

```
git checkout gh/SomeOtherBranch
```

```
# checkout the origin master
```

```
git checkout master
```

```
# merge the branch SomeOtherBranch from "gh" remote
```

```
git merge SomeOtherBranch
```


Extra information

- Branch command

```
# get all local branches
```

```
git branch
```

```
# get all local and remote branches
```

```
git branch -a
```

- .gitignore file

– add here all files/directories that git should ignore and commit it

```
git add .gitignore
```

```
git commit -m "ignore these files"
```

```
*.exe          .gitignore
*.obj
*.o
*.pyc
some/generated/directory
```

```
git status
```

will ignore these files and
not show them for staging
or modification

Extra information

- Diff command – display the changes in the files

show all changes

```
git diff
```

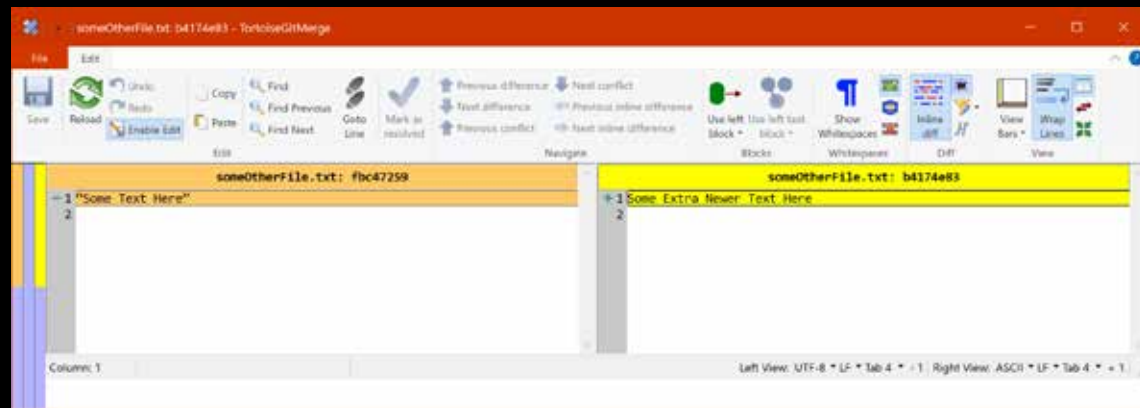
show the changes for a specific file

```
git diff FILE
```

```
git diff PATH/TO/FILE
```

- Visual Diffs

- TortoiseGit (Windows)
- VSCode (Windows, Linux, Mac)
- gitlab & github (when making a pull request or merge request)



Exercise

- Follow the commands in this presentation

The end!

- Questions?