



Universität Hamburg

DER FORSCHUNG | DER LEHRE | DER BILDUNG



CHAI

Humanities-Centered AI

Understanding Data vs. Machine Training

**Malte Luttermann – Institute for Humanities-Centered
Artificial Intelligence (CHAI)**

November 28, 2025

Overview of Contents

- (1) Programming language Python
 - (a) Introduction and first steps
 - (b) Basics
 - (c) Advanced
- (2) Markup languages
 - (a) \LaTeX
 - (b) Markdown
- (3) Development environments
 - (a) Jupyter notebooks
- (4) Version control
 - (a) Git and GitHub
- (5) Scientific computing
 - (a) NumPy and SciPy
- (6) Data processing and visualisation
 - (a) Pandas, matplotlib, and NLTK
- (7) Machine learning (scikit-learn)
 - (a) Basics (datasets, analysis)
 - (b) Simple methods (clustering, ...)
- (8) Deep learning
 - (a) PyTorch

Topics for Today

- (1) Version control
- (2) Git
 - Idea, configuration
 - Local: commit, stash, branch, merge
 - Remote: push, pull, merge
- (3) GitHub



Acknowledgement

- The upcoming slides are taken from the following lecture and have been translated and partially modified:
 - Dr. Magnus Bender: »[Python für Machine Learning und Data Science](#)« as part of the German course »Werkzeuge für das wissenschaftliche Arbeiten«

Versions and History

- Different versions of files
 - Various features/problems are being worked on (simultaneously)
- Keep track of changes (save history)
- `import ...` refers to the file name



data.py



data_fixed.py



20251117_data.py



plot.py

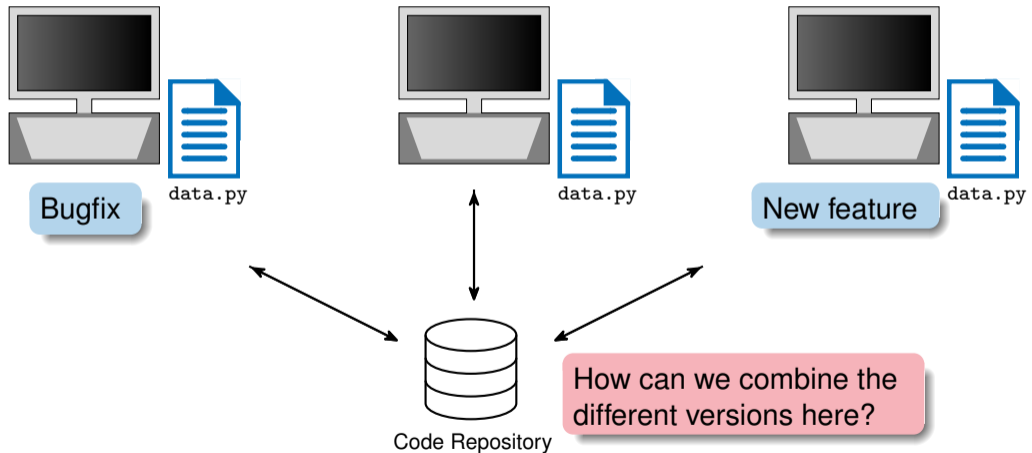


plot_v2.py



20251115_plot.py

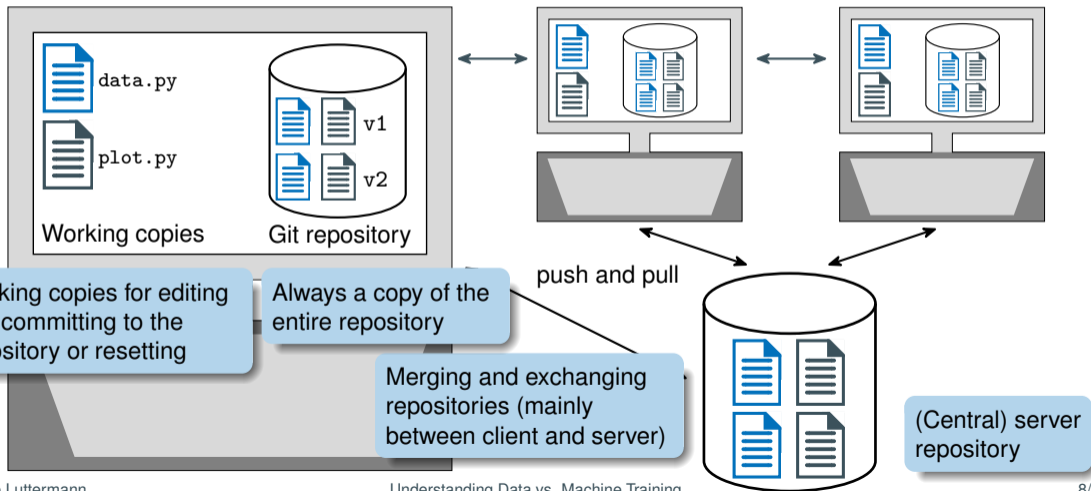
Various Development Locations



Solution: Version Control

- History of changes (text-based files)
 - In particular, it is also possible to reset changes!
- Multiple development branches at the same time
 - Various (new) features and bug fixes
 - Different locations
- Merging of development branches
- One central repository

Distributed Repositories



Git



- Standard tool for version control (especially for open source projects)
 - Developed by Linus Torvalds for Linux kernel
- SHA-1 hash for each committed change
- Full local usage possible, no (central) server required
- Data lost on a repository server can be restored directly from the local repository

Git – Installation



- Installed on macOS
- Package sources under Linux
- Download and installation guide for Windows at <https://git-scm.com/install/>

Push and Pull

- SSH authentication recommended ([guide from GitHub](#))
- Alternatively via HTTP(S) using username and password ([guide from GitHub](#))

Git – Configuration

- Username and email address
 - `git config --global user.name "My Name"`
 - `git config --global user.email "me@example.com"`
 - `--global` for global configuration (otherwise per repository)
- Commits (changes to the repository) are assigned this information
 - Must neither be a real mail address nor your real name
 - However, with a fake mail address, commits cannot be assigned
- If you exchange commits or push them to another repository, your name and email address are shared
- Additional configuration is not required

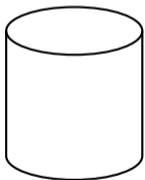
The First Git Repository

```
$> git init
```

```
Initialized empty Git repository in ./.
```

The First Git Repository

```
$> tree -a .  
.  
|-- .git  
|   |-- HEAD  
|   |-- config  
|   |-- description  
|   |-- hooks  
|       |-- ...  
|   |-- info  
|       |-- exclude  
|-- objects  
|   |-- info  
|   |-- pack  
|-- refs  
|   |-- heads  
|   |-- tags
```



Git repository



data.py



plot.py

Working copies

- New empty repository at `./ .git/`
- Working copies at `./`

Git Repository Overview

Git Repository

Stash

Working Tree

Index

Repository

Remote

Changes that
you want to
quickly save
temporarily

Working
directory

Files known
to Git

Versioned
files

Adding Files



```
$> git status
On branch main
No commits yet

$> touch a.txt
```

```
$> git status
On branch main
No commits yet
Untracked files:
  a.txt

$> git add .
```

```
$> git status
On branch main
No commits yet
Changes to be committed:
  new file:   a.txt
```

Commit

```
git commit [-m "My change"]
```

Git Repository

Stash

Working Tree

Index

Repository

Remote

```
$> git status
On branch main
No commits yet
Changes to be committed:
  new file:   a.txt
  new file:   b.txt
```

```
$> git commit -m "Add files a.txt and b.txt"
[main (root-commit) b7d6301] Add files a.txt and b.txt
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 a.txt
 create mode 100644 b.txt
```

```
$> git status
On branch main
nothing to commit, working tree clean
```

Commit

- `git commit [-m "My change"]`
 - Without `-m`, an editor opens to enter the commit message
 - The commit message should briefly describe the changes made
- Commits should not be deleted because others may already have fetched them and built upon them
 - Better make a new commit with changes

Further Changes

Git Repository

Stash

Working Tree

Index

Repository

Remote

`git checkout -- FILE[S]`

```
$> git status
On branch main
Changes not staged for commit:
  modified:  a.txt

$> git add .
$> vim b.txt
```

```
$> git status
On branch main
Changes to be committed:
  modified:  a.txt

Changes not staged for commit:
  modified:  b.txt
```

Further Changes

Git Repository

Stash

Working Tree

Index

Repository

Remote

`git restore -- FILE[S]`

```
$> git status
On branch main
Changes not staged for commit:
  modified:  a.txt

$> git add .
$> vim b.txt
```

```
$> git status
On branch main
Changes to be committed:
  modified:  a.txt

Changes not staged for commit:
  modified:  b.txt
```

Further Changes

- `git checkout -- FILE[S]`
 - Reset file(s) to their status in index
- `git restore -- FILE[S]`
 - Reset file(s) to their status in repository
 - With option `--staged`, reset also file(s) in index to their status in repository

Temporary Storage



```
$> git status
On branch main
Changes to be committed:
  modified:   a.txt
```

```
$> git stash
Saved working directory ...
```

```
$> git status
On branch main
nothing to commit, working tree clean
```

```
$> git stash pop
On branch main
Changes not staged for commit:
  modified:   a.txt
no changes added to commit
Dropped refs/stash@{0}
(c4249e430862620266856a44467e60fe79bc7c04)
```

History of Commits

```
$> git log --oneline
```

```
Lectures -- less - git log --oneline -- 120x36
4fee3c0 (HEAD -> master, origin/master, origin/HEAD) Update UDMT exam template
4890cf9 Small fix UDMT exam template
ec23a55 Continue UDMT exam template
42ca1a7 Continue UDMT exam template
d4aa449 Fixes UDMT slides 5
074fa9c Continue slides UDMT 6
39bf114 Continue slides UDMT 6
11c1d5e Continue slides UDMT 6
16dee4f Continue slides UDMT 6
89285f0 Slides db 6 done
1536a68 Continue slides db 6
875ee66 Continue slides db 6
89c0387 Begin db slides 6
573dbca Continue slides UDMT 6
c2f5bc0 Continue slides UDMT 6
18f469f Continue slides UDMT 6
414d5ce Continue slides UDMT 6
5901615 Continue slides UDMT 6
b561cb5 Continue slides UDMT 6
f3bd9b8 Continue slides UDMT 6
b35f038 Begin slides UDMT 6
3d19426 Finish slides UDMT seminar 5
cbc2113 Continue slides UDMT 5
c9d692e Fixes UDMT 3 + 4
4cbb478 Continue slides UDMT 5
e0212d4 Continue slides UDMT 5
114e76b Continue slides UDMT 5
0f3307f Fix typo db 5
d22f3a1 Continue slides UDMT 5
f260f4d Continue slides UDMT 5
8cf7c25 Continue slides UDMT 5
f4fa654 Continue slides UDMT 5
fec2304 Continue slides UDMT 5
4cc1b3d Add note db slides 5
de4cb78 First draft slides db 5
:
```

History of Commits

- git checkout <hash>
changes working tree to
the state of the commit
 - No changes possible
- Back to current state via
git checkout HEAD

```
Lectures -- less - git log --oneline -- 120x36
4fee3c0 (HEAD -> master, origin/master, origin/HEAD) Update UDMT exam template
4890cf9 Small fix UDMT exam template
ec23a55 Continue UDMT exam template
42ca1a7 Continue UDMT exam template
d4aa449 Fixes UDMT slides 5
074fa9c Continue slides UDMT 6
39bf114 Continue slides UDMT 6
11c1d5e Continue slides UDMT 6
16dee4f Continue slides UDMT 6
89285f0 Slides db 6 done
1536a68 Continue slides db 6
875ee66 Continue slides db 6
89c0387 Begin db slides 6
573dbca Continue slides UDMT 6
c2f5bc0 Continue slides UDMT 6
18f469f Continue slides UDMT 6
414d5ce Continue slides UDMT 6
5901615 Continue slides UDMT 6
b561cb5 Continue slides UDMT 6
f3bd9b8 Continue slides UDMT 6
b35f038 Begin slides UDMT 6
3d19426 Finish slides UDMT seminar 5
cbc2113 Continue slides UDMT 5
c9d692e Fixes UDMT 3 + 4
4cbb478 Continue slides UDMT 5
e0212d4 Continue slides UDMT 5
114e76b Continue slides UDMT 5
0f3307f Fix typo db 5
d22f3a1 Continue slides UDMT 5
f260f4d Continue slides UDMT 5
8cf7c25 Continue slides UDMT 5
f4fa654 Continue slides UDMT 5
fec2304 Continue slides UDMT 5
4cc1b3d Add note db slides 5
de4cb78 First draft slides db 5
:
```

Requirements for Version Control

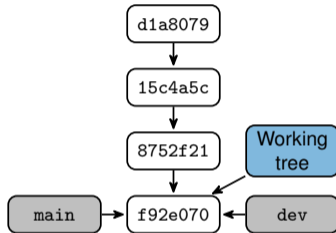
- History of changes (text-based files) ✓
- Multiple development branches at the same time
 - Various (new) features and bug fixes
 - Different locations
- Merging of development branches
- One central repository ✓

Branches

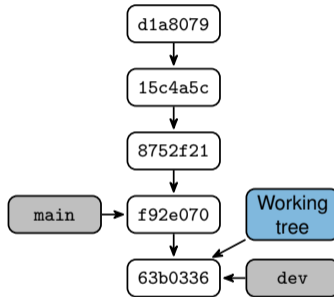
- So far on branch `main` (previously `master`)
- Branching out of development and then merging

```
Lectures — less · git log --oneline — 120×36
4fee3c0 (HEAD -> master, origin/master, origin/HEAD) Update UDMT exam template
4890cf9 Small fix UDMT exam template
ec23a55 Continue UDMT exam template
42ca1a7 Continue UDMT exam template
d4aa449 Fixes UDMT slides 5
074fa9c Continue slides UDMT 6
39bf114 Continue slides UDMT 6
11c1d5e Continue slides UDMT 6
16dee4f Continue slides UDMT 6
89285f0 Slides db 6 done
1536a68 Continue slides db 6
875ee66 Continue slides db 6
89c0387 Begin db slides 6
573dbca Continue slides UDMT 6
c2f5bc0 Continue slides UDMT 6
18f469f Continue slides UDMT 6
414d5ce Continue slides UDMT 6
5901615 Continue slides UDMT 6
b561cb5 Continue slides UDMT 6
f3bd9b8 Continue slides UDMT 6
b35f038 Begin slides UDMT 6
3d19426 Finish slides UDMT seminar 5
cbc2113 Continue slides UDMT 5
c9d692e Fixes UDMT 3 + 4
4cbb478 Continue slides UDMT 5
e0212d4 Continue slides UDMT 5
114e76b Continue slides UDMT 5
0f3307f Fix typo db 5
d22f3a1 Continue slides UDMT 5
f260f4d Continue slides UDMT 5
8cf7c25 Continue slides UDMT 5
f4fa654 Continue slides UDMT 5
fec2304 Continue slides UDMT 5
4cc1b3d Add note db slides 5
de4cb78 First draft slides db 5
:
```

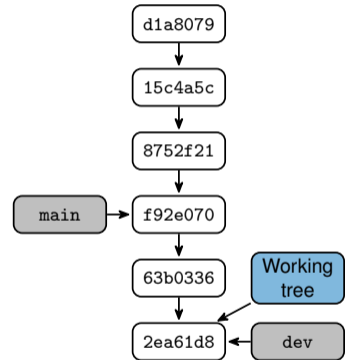
Branches – Idea



```
$> git branch dev
```

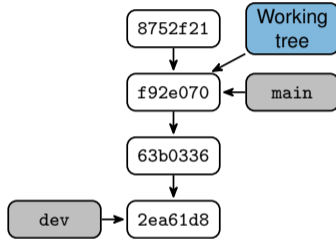


```
$> git checkout dev  
$> git commit
```

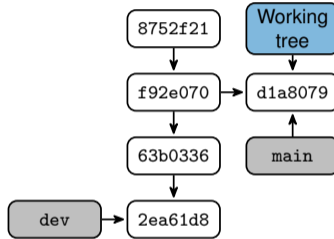


```
$> git commit
```

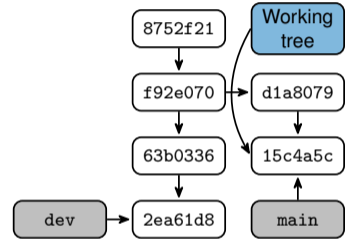
Branches – Branching



\$> git checkout main



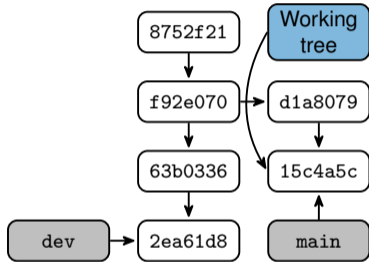
\$> git commit



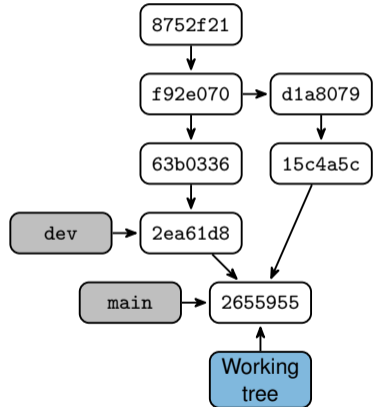
\$> git commit

What happens now?

Branches – Merging

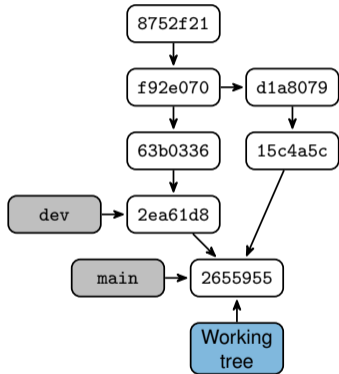


`$> git merge dev`

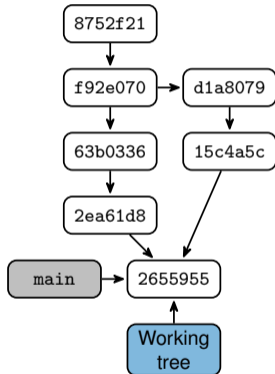


A commit merges
the two branches

Branches – Merging

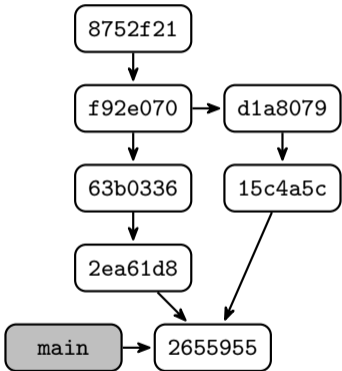


```
$> git branch -d dev
```



Possible problems?

Merge Conflicts



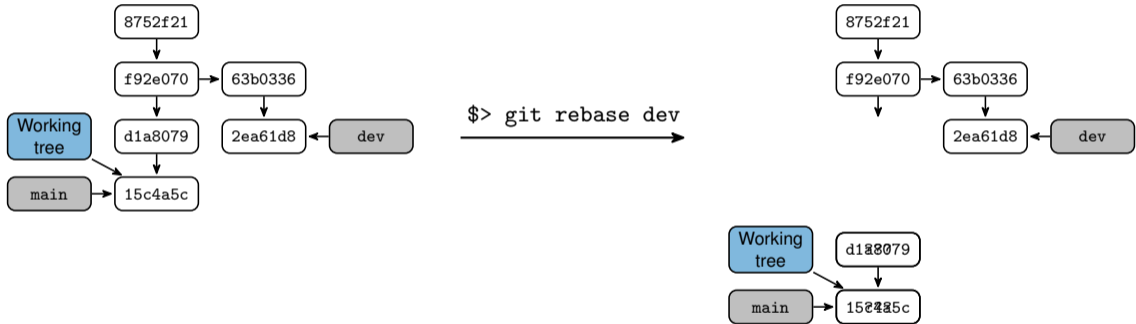
```
$> git merge dev
# Error message
$> git status
# Problematic files are displayed
$> vim fileWithConflict.txt
# Resolve conflict
$> ...
$> git add fileWithConflict.txt
$> git commit
```

Merge conflicts occur if the same line of the same file has been edited in both branches

Further Merge Strategies – Fast Forward



Further Merge Strategies – Rebase



Further Merge Strategies – Rebase



Gitignore

- Git is not very good at managing binary and other non-text files
 - Works as well, but merge conflicts are difficult to resolve
- Compiled programs and \LaTeX PDFs should therefore not be included in the repository
 - `.gitignore` file specifies files and directories to be ignored by Git

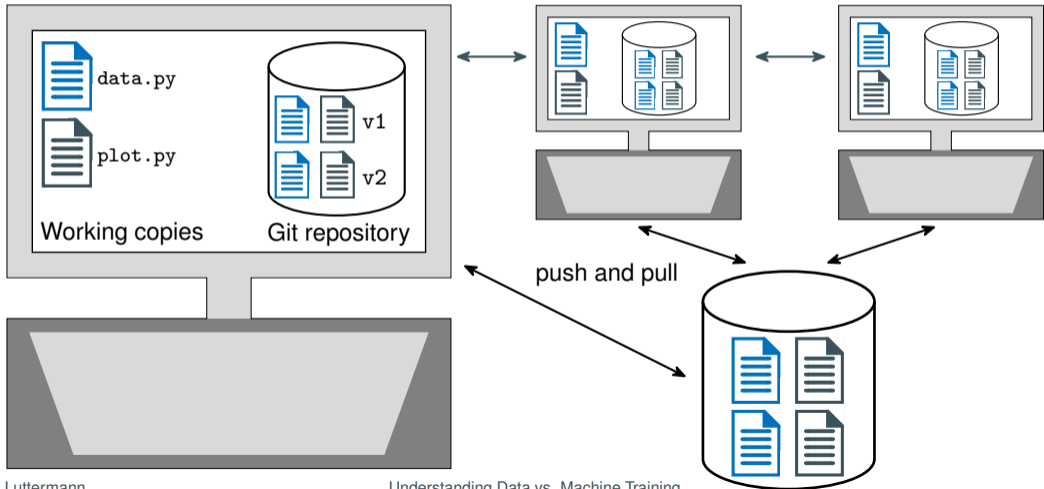
```
1 .DS_Store
2 __pycache__
3 *.aux
4 *.fdb_latexmk
5 *.log
6 *.pdf
7 /data/*
8 /bin/*
```

Requirements for Version Control

- History of changes (text-based files) ✓
- Multiple development branches at the same time ✓
 - Various (new) features and bug fixes ✓
 - Different locations
- Merging of development branches ✓
- One central repository ✓

Different locations? So far, we have only worked with local repositories!

Distributed Repositories



Remote Repository

- Access via a URL
 - Accessible via network or locally
 - Read and/or write permissions
- Different protocols:
 - SSH (requires authentication)
 - HTTP(S) (requires authentication only for uploads)

Add Remote Repositories

```
$> git remote add <name> <url>
```

```
$> git remote add origin https://github.com/torvalds/linux.git
```

```
$> git remote add origin https://Servergithub.com/User/Ownertorvalds/Repositorylinux.git
```

- Source, usually named origin, writing in general not allowed

```
$> git remote add MyCopy git@github.com:myuser/linux.git $> git remote  
add MyCopy git@github.com: Servermyuser / Repositorylinux.git
```

- Own copy, named as MyCopy here, writing in general allowed

Downloading from a Remote Repository

Git Repository


Stash

Working Tree

Index

Repository

Remote



```
git fetch [-- all | REMOTE BRANCH]
git pull REMOTE BRANCH
```

```
$> git fetch -- all
remote: Enumerating objects: 513, done.
remote: Counting objects: 100% (116/116), done.
remote: Compressing objects: 100% (107/107), done.
remote: Total 513 (delta 48), reused 0 (delta 0), pack-reused 397
Receiving objects: 100% (513/513), 271.85 KiB | 1.93 MiB/s, done.
Resolving deltas: 100% (289/289), done.
From https://server/user/repo
* [new branch] main -> origin/main
$> git merge origin/main
```

Merge conflicts may occur!

Downloading from a Remote Repository

- `$> git fetch -- all`
 - Download all changes (commits, branches) from all remotes to the local repository (no changes to the working tree)
- `$> git merge origin/main`
 - Merge the current local branch with a remote branch
- `$> git pull origin main`: fetch and merge in one step, i.e., equivalent to:
 - `$> git fetch -- all`
 - `$> git merge origin/main`

Initial Download from a Remote Repository

Git Repository

Stash

Working Tree

Index

Repository

Remote

`git fetch [-- all | REMOTE BRANCH]`

`git pull REMOTE BRANCH`

`git clone URL`

```
$> mkdir linux
```

```
$> cd ./linux/
```

```
$> git init
```

```
$> git remote add origin https://github.com/torvalds/linux.git
```

```
$> git pull origin master
```

equivalent to:

```
$> git clone https://github.com/torvalds/linux.git
```

Uploading to a Remote Repository

Git Repository

Stash

Working Tree

Index

Repository

Remote

`git push REMOTE BRANCH`

`git fetch [-- all | REMOTE BRANCH]`

`git pull REMOTE BRANCH`

`git clone URL`

```
$> git commit -m "My changes"  
$> git push MyCopy main  
Enumerating objects: 513, done.  
Counting objects: 100% (513/513), done.  
Delta compression using up to 12 threads  
Compressing objects: 100% (200/200), done.  
Writing objects: 100% (513/513), 271.84 KiB | 2.75 MiB/s, done.  
Total 513 (delta 289), reused 513 (delta 289), pack-reused 0  
remote: Resolving deltas: 100% (289/289), done.  
To server:user/repo.git  
* [new branch] main -> main
```

Git Commands I

Configuration

```
git config [--global] user.name "NAME"
```

Declare name that is used to sign commits

```
git config [--global] user.email "E-MAIL"
```

Declare mail that is used in commits

Git Commands II

Repository, history	<code>git init</code>	Create empty repository
	<code>git log --oneline --graph</code>	Display history of commits
	<code>git checkout HASH</code>	Open commit in working tree
	<code>git checkout HEAD</code>	Back to head of repository

Git Commands III

Working tree, index, commit	<code>git add FILE[S]</code>	Mark a file/path for commit (staged)
	<code>git status</code>	Display file status in repository
	<code>git commit [-m "MESSAGE"]</code>	Create commit
	<code>git rm FILE[S]</code>	Delete files from the index (not including old commits)
	<code>git checkout -- FILE</code>	Reset file in working tree to staged version
	<code>git restore [--staged] FILE</code>	Reset file in working tree to last commit

Git Commands IV

Stash	<code>git stash</code>	Save working tree to temporary storage
	<code>git stash pop</code>	Apply temporary storage to working tree

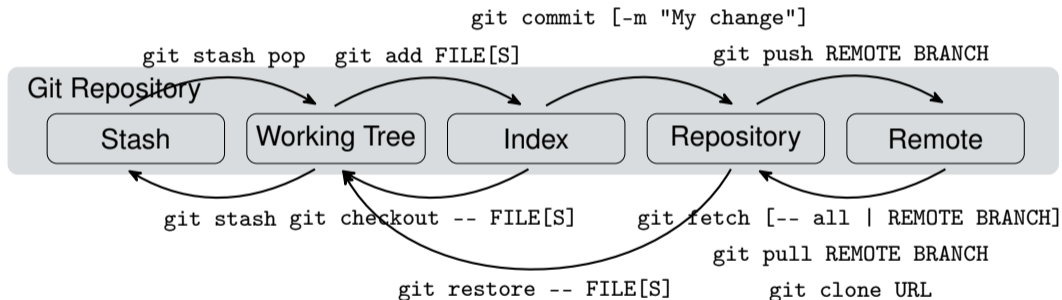
Git Commands V

Branches, merge	<code>git branch NAME</code>	Create new branch
	<code>git checkout NAME</code>	Change to branch in working tree
	<code>git merge NAME</code>	Merge branch into current branch

Git Commands VI

Remote	<code>git remote add NAME URL</code>	Connect a remote repository
	<code>git push REMOTE BRANCH</code>	Upload branch to remote repository
	<code>git fetch [--all REMOTE BRANCH]</code>	Synchronise branch or everything with remote repository
	<code>git pull REMOTE BRANCH</code>	Download branch from remote repository to working tree
	<code>git clone URL</code>	Download repository from URL and create it locally

Git Commands Overview



Git Repository Hosting

- GitHub, GitLab, Bitbucket, Gitea (self hosting), ...
- Web interface for displaying the files and the commit history
 - Commits, branches, ...
 - Forks, pull requests, ...
- Project management
 - Issues, milestones, wikis, ...

GitHub

matplotlib / matplotlib

Code Issues 1.2k Pull requests 405 Actions Projects 2 Wiki Security Insights

matplotlib Public Sponsor Watch 587 Fork 8.1k Star 22k

main 32 Branches 146 Tags

Go to file Add file Code

rcomer Merge pull request #30788 from raphaelquast/qt_backend_f11 9a2f807 · 7 hours ago 53,404 Commits

.circleci	FIX: Fix unit example so that we can unpin numpy>2.1	9 months ago
.devcontainer	swap xkcd script for humor sans	2 years ago
.github	github: added explicit do not merge label to label check (...)	yesterday
LICENSE	Add a last resort font for missing glyphs	7 months ago
ci	MNT: Define Protocol for Animation.event_source	2 months ago
doc	Merge pull request #30733 from rcomer/pie_label	yesterday
extern	Fix typos in some files (#30759)	last week
galleries	Merge pull request #30733 from rcomer/pie_label	yesterday
lib	Fix typo in key-mapping for "f11"	9 hours ago
requirements	Merge pull request #30610 from dstansby/mpl-sphinx-th...	2 months ago
src	DOC: Correct grammatical issues especially on a/an usag...	2 weeks ago
subprojects	Fix building freetype 2.6.1 on macOS clang 18	last year
tools	ENH: introduce PieContainer and pie_label method	4 days ago

About matplotlib: plotting with Python

matplotlib.org/stable/

python gtk data-science qt data-visualization tk matplotlib plotting wx

Readme Code of conduct Contributing Security policy Cite this repository Activity Custom properties 22k stars 587 watching 8.1k forks Report repository

Releases 97

REL: v3.10.8 (Latest) 2 weeks ago

GitHub

The screenshot shows the GitHub repository page for `matplotlib/matplotlib`. The repository is public and has 587 watchers, 8.1k forks, and 22k stars. The repository structure is listed on the left, and a 'Clone' dropdown menu is open, showing options for cloning via HTTPS, SSH, or GitHub CLI. The 'About' section on the right provides information about the repository, including its description, tags, and release information.

Repository Structure:

File/Folder	Description	Last Update
<code>rcomer</code>	Merge pull request #30788 from raphaelquast/qt_backend_ft	
<code>.circleci</code>	FIX: Fix unit exampl	
<code>.devcontainer</code>	swap xkcd script f	
<code>.github</code>	github: added expl	
<code>LICENSE</code>	Add a last resort fo	
<code>ci</code>	MNT: Define Proto	
<code>doc</code>	Merge pull request	
<code>extern</code>	Fix typos in some files (#30759)	last week
<code>galleries</code>	Merge pull request #30733 from rcomer/pie_label	yesterday
<code>lib</code>	Fix typo in key-mapping for "f11"	9 hours ago
<code>requirements</code>	Merge pull request #30610 from dstansby/mpl-sphinx-th...	2 months ago
<code>src</code>	DOC: Correct grammatical issues especially on a/an usag...	2 weeks ago
<code>subprojects</code>	Fix building freetype 2.6.1 on macOS clang 18	last year
<code>tools</code>	ENH: introduce PieContainer and pie_label method	4 days ago

Clone Options:

- Local
- Codespaces
- Clone
- HTTPS
- SSH
- GitHub CLI
- git@github.com:matplotlib/matplotlib.git
- Use a password-protected SSH key.
- Open with GitHub Desktop
- Download ZIP

About: matplotlib: plotting with Python

matplotlib.org/stable/

python gtk data-science qt data-visualization tk matplotlib plotting wx

Readme Code of conduct Contributing Security policy Cite this repository Activity Custom properties 22k stars 587 watching 8.1k forks Report repository

Releases 97

REL: v3.10.8 (Latest) 2 weeks ago

GitHub

The screenshot displays the GitHub interface for the `matplotlib` repository. At the top, a list of recent commits is shown with columns for the file name, commit message, and time since the commit. Below this, the repository's README is visible, featuring a header with various badges for package managers, download statistics, and health scores. The main content of the README is the `matplotlib` logo and a brief description of the library as a tool for creating static, animated, and interactive visualizations in Python.

File	Commit Message	Time
README.md	Update README links to static images	last month
SECURITY.md	Update release notes for 3.10.0	11 months ago
azure-pipelines.yml	CI: keep other branch of MSFT setup	7 months ago
environment.yml	Update mpl-sphinx-theme in environment.yml	last week
meson.build	BLD: Ensure meson.build has the right version of Python	7 months ago
meson.options	Use new name for Meson options file	last year
pyproject.toml	BLD: update trove metadata to support py3.14	last month
tox.ini	API: bump minimum supported version of Python	7 months ago

Deployments 17
release last month
[+ 16 deployments](#)

Languages

- Python 92.9%
- C++ 4.2%
- Jupyter Notebook 0.9%
- C 0.8%
- Objective-C 0.6%
- JavaScript 0.3%
- Other 0.3%

matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.
Check out our [home page](#) for more information.

Summary

- Version control
- Git
 - Idea, configuration
 - Local: commit, stash, branch, merge
 - Remote: push, pull, merge
- GitHub

