

Git and GitHub

Anastazie Sedláková
Libor Mořkovský
Václav Janoušek

Versioning

"FINAL".doc



↪ FINAL.doc!



↪ FINAL_rev.2.doc

Versioning



FINAL_rev.6.COMMENTS.doc



FINAL_rev.8.comments5.
CORRECTIONS.doc

Versioning

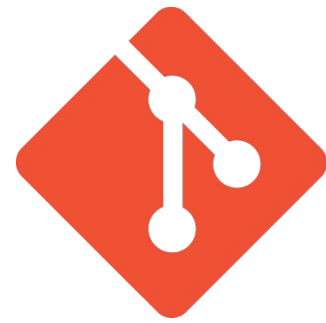
JORGE CHAM © 2012



FINAL_rev.18.comments7.
corrections9.MORE.30.doc

FINAL_rev.22.comments49.
corrections.10.#@\$%WHYDID
ICOMETOGRADSCHOOL?????.doc

Git



- System to keep track of changes that happen across the set of files
- A.K.A. version control tool



Github

- Online service to run Git in the cloud
- You version your files with Git and then share and store using github
- Versioned files are easily shared across multiple computers
- **Do not share data in Github.** It is not optimized to share data
 - Put data directory into .gitignore file
 - Create .gitignore file using nano in your project directory

Git vs Google Docs

	<i>saves current state ...</i>	<i>keeps track of ...</i>
Git	on request	directory tree (multiple files and their position)
Google Docs	automatically	single file

The screenshot displays the Google Docs interface. On the left, a document is open with a dark header bar showing a back arrow and the time 'Today, 1:45 PM'. Below the header, there is a print icon and a zoom level of '100%'. The main content area shows a document with a highlighted sentence: '-Here I have added some text.' Below this, there is a section titled 'What Is Google Docs' with a paragraph of text: 'Google Docs is Google's online word processor. What's its competitor, Microsoft Word, are its collaborative features. documents across platforms and work on them together in. Your collaborators don't even need a Google account to view and share with them.'

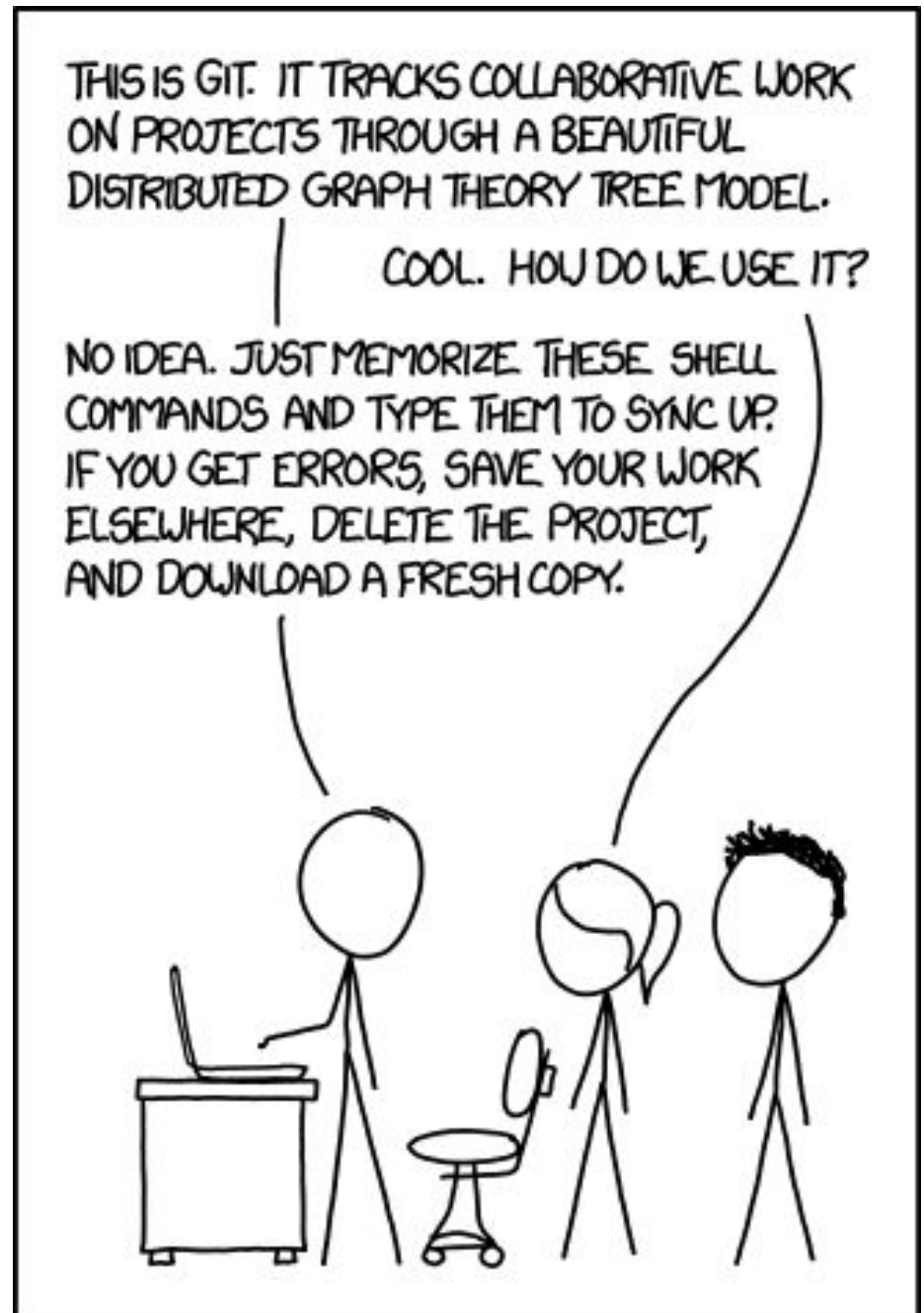
On the right side, a 'Version history' sidebar is open. It has a blue header with the title 'Version history' and a toggle switch for 'Only show named versions' which is currently turned on. The sidebar lists three versions under the heading 'Today':

- March 9, 1:45 PM** (Current version) by Tina Sieber
- First Draft** (March 9, 1:43 PM) by Tina Sieber
- March 9, 12:57 PM** by Tina Sieber

Git and Github: upsides

- collaboration
 - sharing
 - peer review
- backup
 - can go back to any version
 - simple and secure way to distribute to other machines

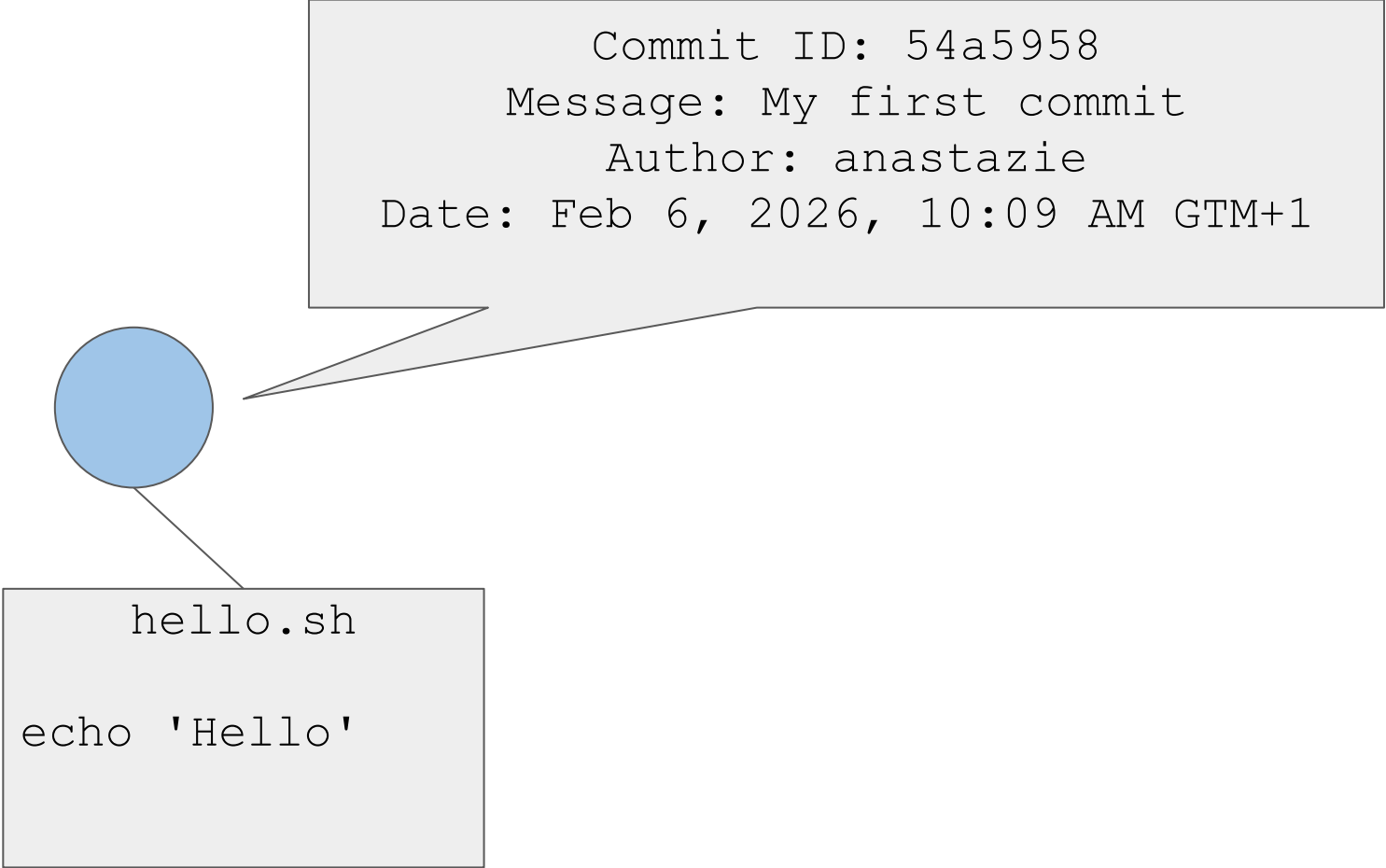
Git: downsides



Vocabulary

- Repository
 - Project's folder
 - A repository contains all of the project files (including documentation), and stores each file's revision history
 - Repositories can have multiple collaborators and can be either public or private
- Commit
 - Records changes (snapshots) to one or more files
 - Core building block units of a Git project timeline
- Branch
 - A branch is a parallel version of a repository. It is contained within the repository, but does not affect the primary or main branch allowing you to work freely without disrupting the "live" version.
- More terms [here](#)

Commits



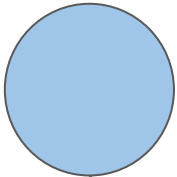
```
Commit ID: 54a5958  
Message: My first commit  
Author: anastazie  
Date: Feb 6, 2026, 10:09 AM GMT+1
```

```
hello.sh  
  
echo 'Hello'
```

Commits

Commit identification

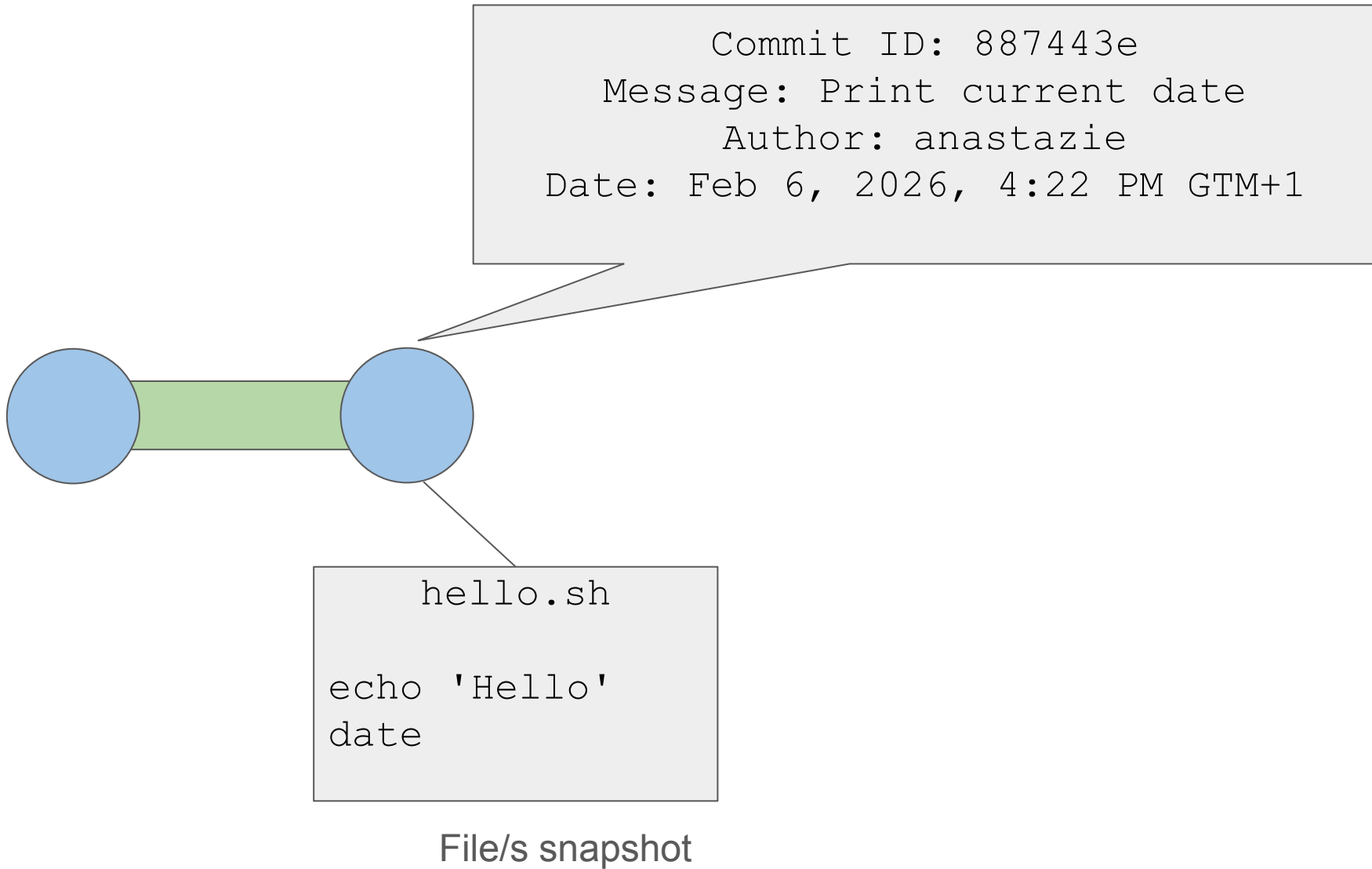
```
Commit ID: 54a5958  
Message: My first commit  
Author: anastazie  
Date: Feb 6, 2026, 10:09 AM GMT+1
```



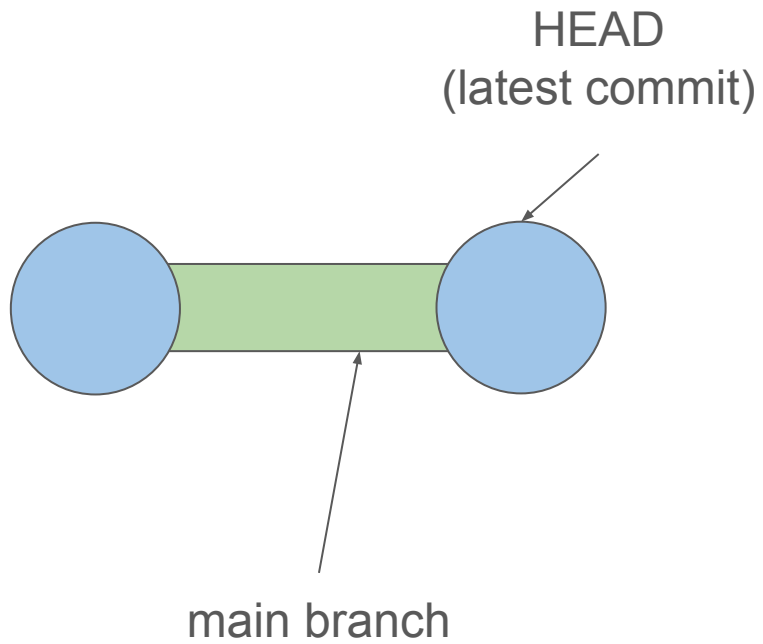
```
hello.sh  
  
echo 'Hello'
```

File/s snapshot

Commits

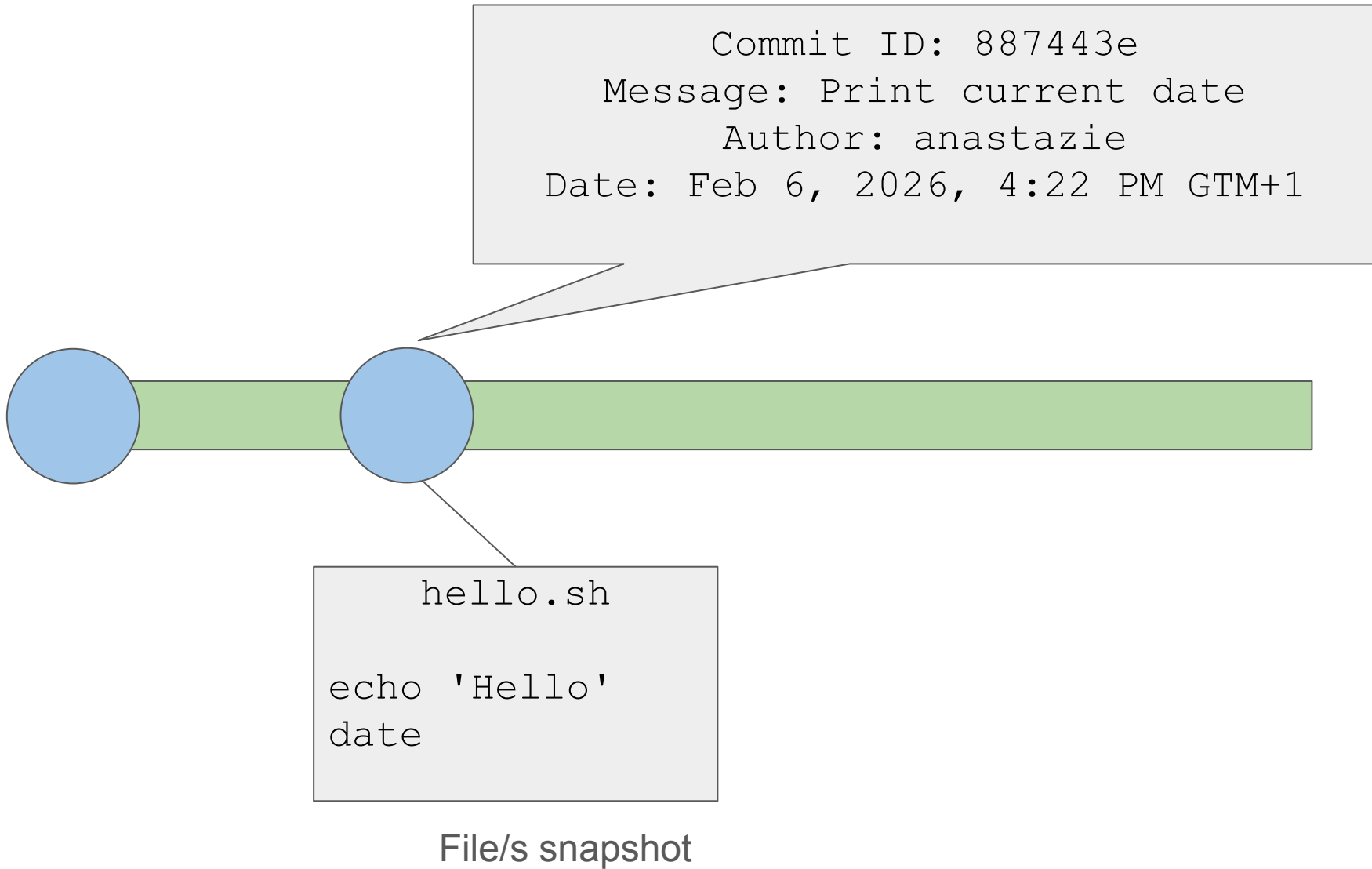


Commits



File/s snapshot

Commits



GitHub

“Social part of programming”



Exercise

- Register at <https://github.com/>
- Create repository (use + sign)
 - Visibility: Public
 - Add README On
 - Add license: Apache License 2.0. (or any other preferred)

Exercise

Create a new repository

Repositories contain a project's files and version history. Have a project elsewhere? [Import a repository](#).
Required fields are marked with an asterisk (*).

1 General

Owner * **Repository name ***

Choose an owner ▾ /

Great repository names are short and memorable. How about [super-tribble?](#)

Description

0 / 350 characters

2 Configuration

Choose visibility * Public ▾

Choose who can see and commit to this repository

Start with a template No template ▾

Templates pre-configure your repository with files.

Add README On

READMEs can be used as longer descriptions. [About READMEs](#)

Add .gitignore No .gitignore ▾

.gitignore tells git which files not to track. [About ignoring files](#)

Add license Apache License 2.0 ▾

Licenses explain how others can use your code. [About licenses](#)

Create repository

Configuring git

- Specifying information about you to identify your commits

```
$ git config --global user.email  
"ferda@mraveniste.cz"
```

```
$ git config --global user.name "Ferda  
Mravenec"
```

Adding SSH keys to Github

- Github need to know who is making commits in repository
- You generate keys locally or on server. Your SSH key has private and public parts
- Keys are stored in ~/.ssh directory
- You then save public part in Github

Adding SSH keys to Github

```
$ ssh-keygen -t ed25519
```

```
Generating public/private ed25519 key pair.
```

```
Enter file in which to save the key (/home/nasta/.ssh/id_ed25519):
```

```
Enter passphrase for "/home/nasta/.ssh/id_ed25519" (empty for no  
passphrase):
```

```
Enter same passphrase again:
```

```
Your identification has been saved in /home/nasta/.ssh/id_ed25519
```

```
Your public key has been saved in /home/nasta/.ssh/id_ed25519.pub
```

```
The key fingerprint is:
```

```
SHA256:rx+tafZatRarelwb/wlQK01jim6K7SwyhiA4nPiac5M nasta@ngs-course-2026
```

```
The key's randomart image is:
```

```
...
```

```
$ cat ~/.ssh/id_ed25519.pub
```

```
ssh-ed25519
```

```
XXXXC3NzaC1lZDI1NTE5XXXXXXXXIGIxZEsut6N5WYdOu/80Qb1ykJ1T5khRWJSev9Igj1uc
```

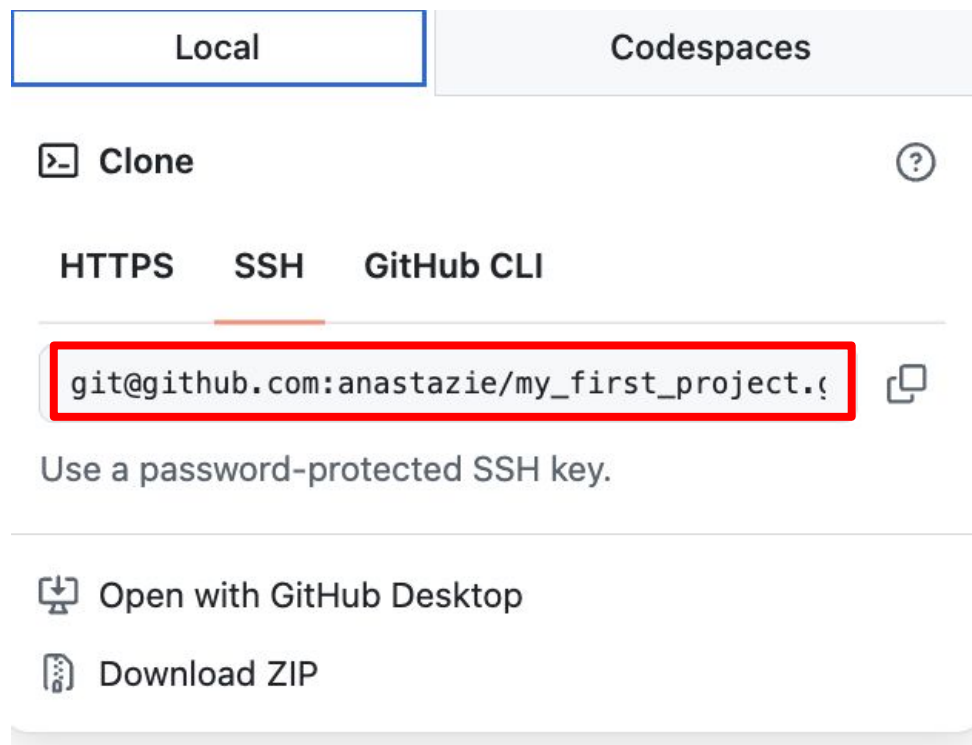
```
nasta@ngs-course-2026
```

Adding SSH key on Github

- In the upper-right corner of any page on GitHub, click your profile picture, then click Settings.
- In the "Access" section of the sidebar, click SSH and GPG keys.
- Click New SSH key or Add SSH key.
- In the "Title" field, add a descriptive label for the new key. For example, if you're using a personal laptop, you might call this key "Personal laptop".
- Select the type of key, either authentication or signing. For more information about commit signing, see *About commit signature verification*.
- In the "Key" field, paste your public key.
- Click Add SSH key.
- [Instructions on Github](#)

Cloning repository from remote

- Once you created repository on Github, you want to download it on your local machine or server
 - `git clone git@github.com:anastazie/my_first_project.git`
- Repository folder will appear in you working directory



The screenshot shows the GitHub interface for cloning a repository. At the top, there are two tabs: 'Local' (selected) and 'Codespaces'. Below the tabs, there is a 'Clone' button with a terminal icon and a help icon. Underneath, there are three options: 'HTTPS', 'SSH' (selected), and 'GitHub CLI'. A red box highlights the repository URL: `git@github.com:anastazie/my_first_project.git`. Below the URL, there is a note: 'Use a password-protected SSH key.' At the bottom, there are two options: 'Open with GitHub Desktop' and 'Download ZIP'.

Cloning repository from remote

The screenshot shows the GitHub interface for a repository named 'my_first_project' by user 'anastazie'. The repository is public and has 1 branch (main) and 0 tags. The 'Code' dropdown menu is open, displaying options for cloning the repository. The 'Local' tab is selected, showing the 'Clone' button and three methods: HTTPS, SSH, and GitHub CLI. The SSH URL is `git@github.com:anastazie/my_first_project.git`. Below the cloning options, there are links to 'Open with GitHub Desktop' and 'Download ZIP'. The repository content shows an initial commit with files 'LICENSE' and 'README.md'. The README file is open, displaying the repository name 'my_first_project' and the description 'Test project' under an Apache-2.0 license. The right sidebar contains sections for 'About', 'Releases', and 'Packages', all showing zero counts and links to create new content.

anastazie / my_first_project

Code Issues Pull requests Agents Actions Projects Wiki Security Insights Settings

my_first_project Public

main 1 Branch 0 Tags

Go to file Add file Code

Local Codespaces

Clone

HTTPS SSH GitHub CLI

git@github.com:anastazie/my_first_project.git

Use a password-protected SSH key.

Open with GitHub Desktop

Download ZIP

anastazie Initial commit

LICENSE Initial commit

README.md Initial commit

README Apache-2.0 license

my_first_project

Test project

About

Test project

Readme

Activity

0 stars

0 watching

0 forks

Releases

No releases published

[Create a new release](#)

Packages


No packages published

[Publish your first package](#)

Commit changes

- Check what files are modified, added or deleted
 - `git status`
- add files to staging area (where you prepare the next commit)
 - `git add hello.sh`

Working copy	Staging area	Commits
hello.sh .git LICENSE README.md	hello.sh	



```
$ git status
```

```
On branch main
```

```
Your branch is up to date with 'origin/main'.
```

```
Untracked files:
```

```
(use "git add <file>..." to include in what will be committed)
```

```
hello.sh
```

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

```
$ git add hello.sh
```

```
$ git status
```

```
On branch main
```

```
Your branch is up to date with 'origin/main'.
```

```
Changes to be committed:
```

```
(use "git restore --staged <file>..." to unstage)
```

```
new file: hello.sh
```

```
$ git commit -m 'print Hello'
```


```
[main 85074ac] print Hello
```

```
1 file changed, 1 insertion(+)
```

```
create mode 100644 hello.sh
```

Commit changes

- initialize a repository
 - `git init`
- add files to staging area (where you prepare the next commit)
 - `git add hello.sh`
- create a commit
 - `git commit -m 'print Hello'`

Working copy	Staging area	Commits
hello.sh .git LICENSE README.md		hello.sh 85074...

Navigating the commit history of a Git repository

```
$ git log
commit 85074acd9a3f92371d2899d4e08ed1525d0bc170 (HEAD -> main)
Author: Anastazie Sedlakova <nasta@sedlakovi.org>
Date:   Sat Feb 7 14:55:44 2026 +0100

    print Hello

commit 2de4fcae4f6b3c15832bb83587b60b2a2dfffd8f5 (origin/main,
origin/HEAD)
Author: Anastazie Sedlakova <anastazie@upheal.io>
Date:   Sat Feb 7 14:44:21 2026 +0100






    Initial commit
```




Transferring your changes to Github

- Use git push
- First time git push -u origin main
 - Establish a Connection: It links your local main branch to the main branch on the remote server (origin).
 - Set Up "Tracking": The -u flag sets the remote branch as the upstream for your local branch.
 - Convenience for Later: Once this link is created, you can just type git push or git pull in the future without having to specify origin main every single time.

Pushing to remote (Github)

```
$ git push -u origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 10 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 327 bytes | 327.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:anastazie/my_first_project.git
 2de4fca..85074ac  main -> main
```

 anastazie print Hello	85074ac · 6 minutes ago	 2 Commits
 LICENSE	Initial commit	18 minutes ago
 README.md	Initial commit	18 minutes ago
 hello.sh	print Hello	6 minutes ago







 **README**  Apache-2.0 license 

my_first_project

Test project

About

Test project

-  Readme
-  Apache-2.0 license
-  Activity
-  0 stars
-  0 watching
-  0 forks

Releases

No releases published
[Create a new release](#)

Packages

No packages published
[Publish your first package](#)

Languages



Initialize git repository locally

- In your project, initialize git
 - `git init`
- Commit changes
- Create a remote repository: Log in to GitHub and create a new, empty repository. Do not initialize it with a README or license
- Link the local and remote repositories: Copy the remote repository URL SSH
 - `git remote add origin <REMOTE_URL>`
- Push to the remote
 - The `-u` flag sets the "upstream" tracking, allowing you to simply use `git push` for future updates to this branch
 - `git push -u origin main`

Diff

A shorthand for “differences”. Usually between working copy and the last commit.

‘-’ means line removed

‘+’ means line added

A sample from course materials repo:

```
``pull`` will do a merge for you
```

```
- ``git checkout --theirs``
```

```
+ ``git checkout --theirs {file path}``
```

```
in conflict, choose the version from the remote branch
```

Pulling from a remote

- download commits and merge into working copy
 - `git pull`

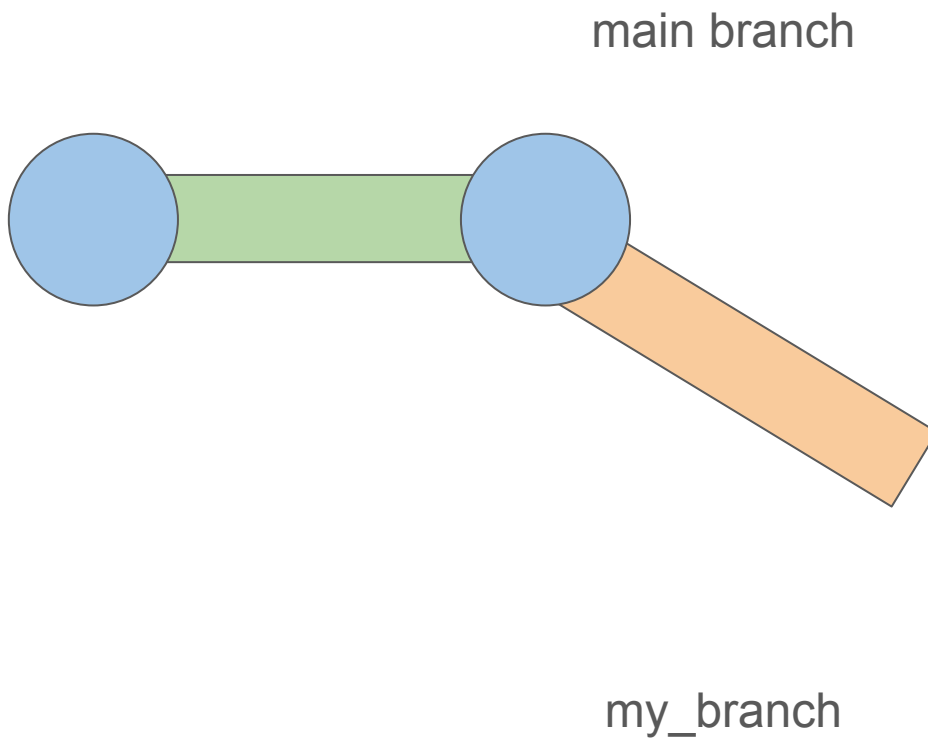
Exercise

- In your repository, create `hello.sh` file
- Commit changes
- Push changes to Github

Branches

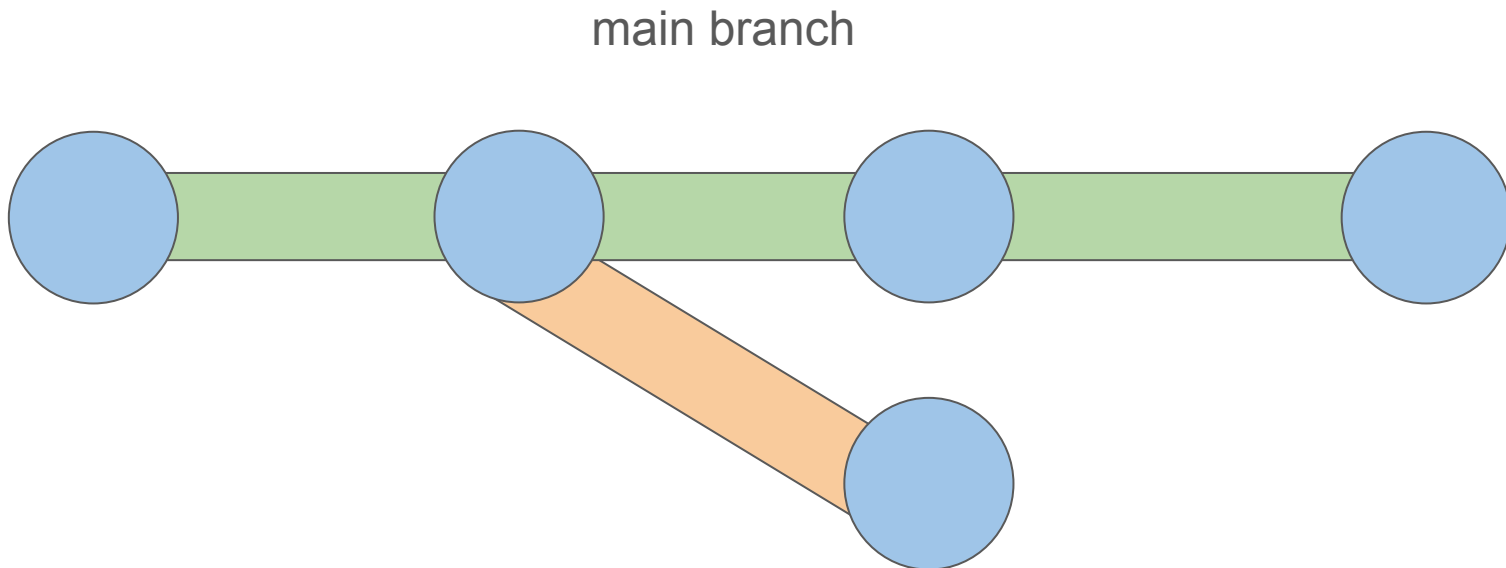
- branches help you manage multiple versions of your code
- important for programmers, but we can't ignore it completely
- we'll be using one local branch called `main`
- on a remote, there is a twin branch, `origin/main`

Commits



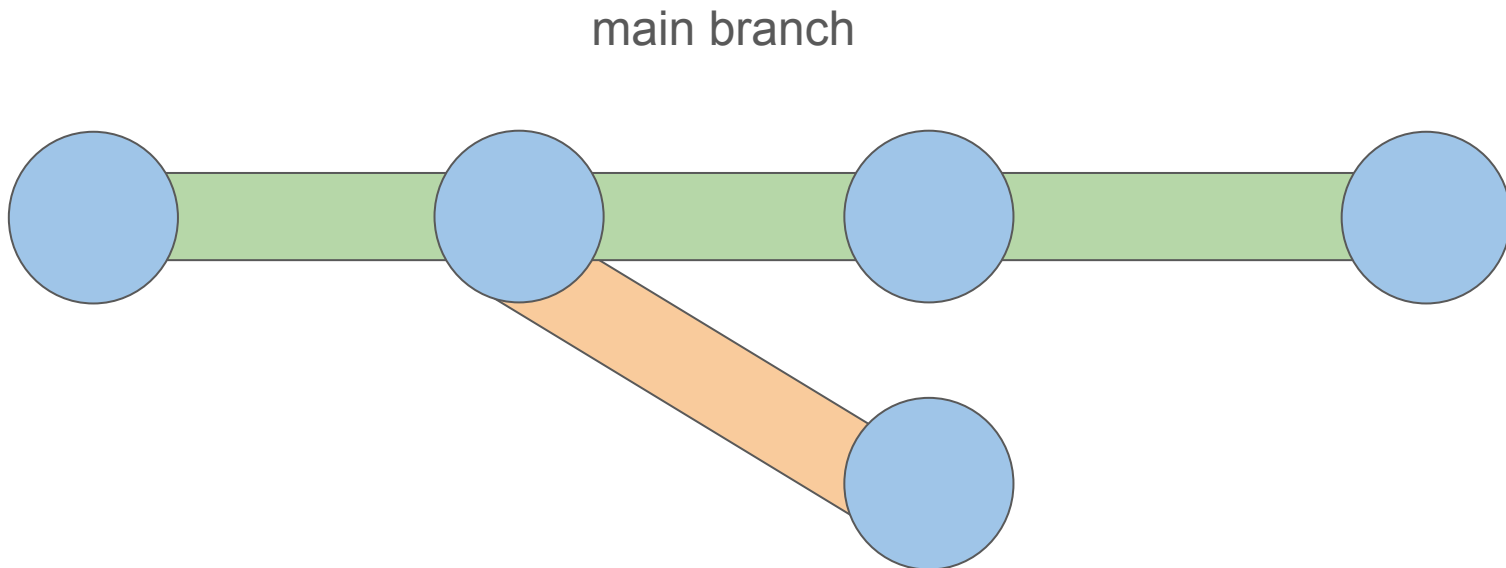
Commits

Working on your branch does not affect the main branch



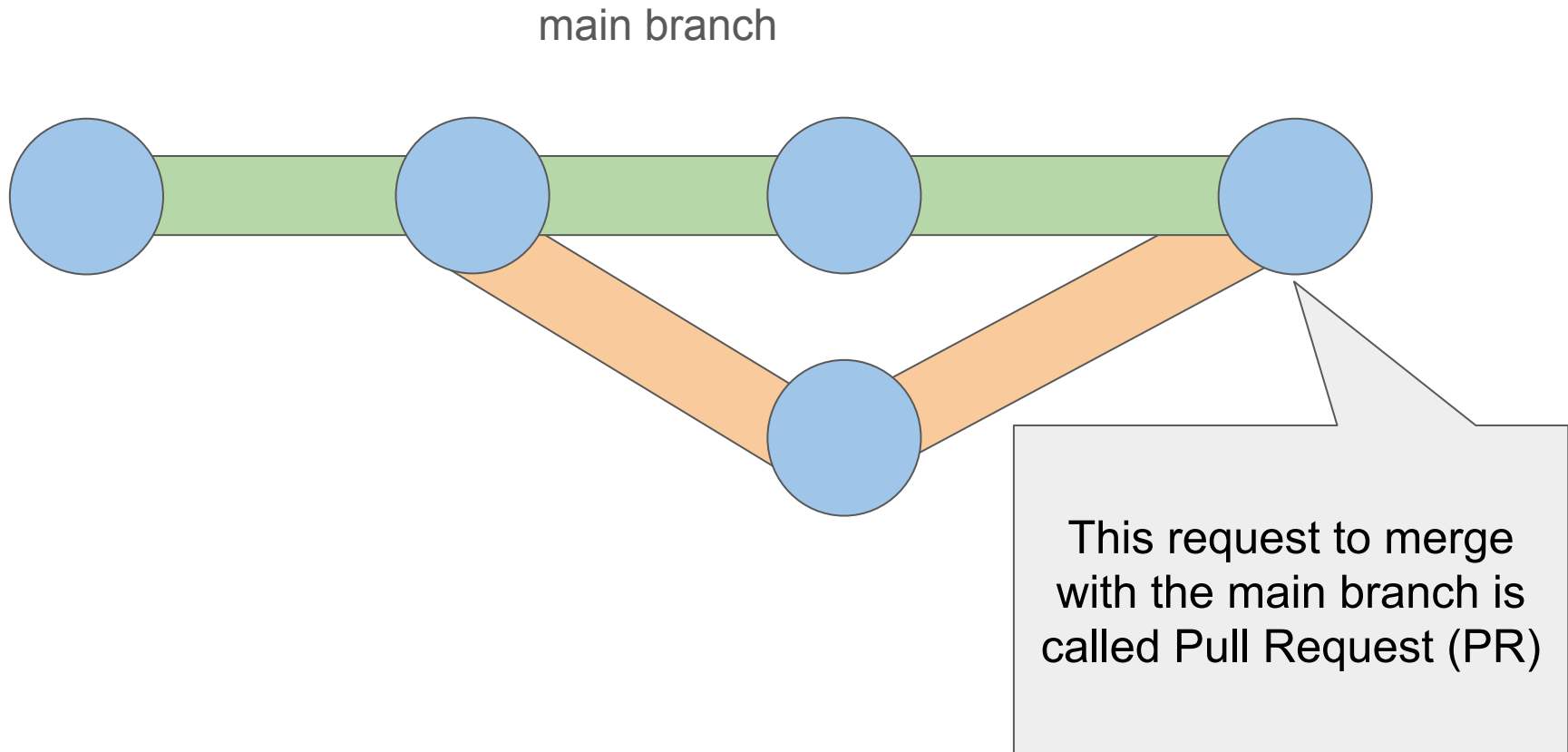
Commits

Working on your branch does not affect the main branch



Commits

Once you are done working on your branch you can merge it to the main branch



Working with branches

- Create branch
 - `git checkout -b add-date`
 - Alternatively use `git switch -c add-date`
 - You will be automatically moved to this branch
- Move to another branch
 - `git checkout <branch name>`
 - Alternatively `git switch <branch name>`

Working with branches

```
$ git checkout -b add-date
```

```
Switched to a new branch 'add-date'
```

```
$ git switch main
```

```
Switched to branch 'main'
```

```
Your branch is up to date with 'origin/main'.
```

Exercise

- Create new branch
- In that branch modify `hello.sh`
- Push your changes

Working with branches

```
$ git add hello.sh
$ git commit -m 'add date'
[add-date e532482] add date
 1 file changed, 1 insertion(+)
$ git push -u origin add-date
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 10 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 270 bytes | 270.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote: Create a pull request for 'add-date' on GitHub by visiting:
       https://github.com/anastazie/my\_first\_project/pull/new/add-date
remote:
To github.com:anastazie/my_first_project.git
 * [new branch]      add-date -> add-date
branch 'add-date' set up to track 'origin/add-date'.
```

Creating pull request

When I want changes in my branch to be in the main branch. Pull request can be then merged into main branch.

The screenshot shows the GitHub interface for a repository named 'my_first_project'. At the top, there are navigation buttons for 'Pin', 'Watch' (0), 'Fork' (0), and 'Star' (0). Below this, a yellow banner highlights a commit by 'add-date' with a 'Compare & pull request' button. The repository details show 'main' branch, '1 Branch', and '0 Tags'. A commit by 'anastazie' is shown with files 'LICENSE', 'README.md', and 'hello.sh'. The README section is visible, showing the project name 'my_first_project' and the description 'Test project'. On the right sidebar, there are sections for 'About', 'Releases', 'Packages', and 'Languages'.

my_first_project Public

Pin Watch 0 Fork 0 Star 0

add-date had recent pushes 3 seconds ago [Compare & pull request](#)

main 1 Branch 0 Tags

Go to file Add file Code

anastazie print Hello 85074ac · 30 minutes ago 2 Commits

LICENSE	Initial commit	41 minutes ago
README.md	Initial commit	41 minutes ago
hello.sh	print Hello	30 minutes ago

Feb 7, 2026, 2:55 PM GMT+1

my_first_project

Test project

About

Test project

- Readme
- Apache-2.0 license
- Activity
- 0 stars
- 0 watching
- 0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Languages

- Shell 100.0%

Creating pull request

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#) or [learn more about diff comparisons](#).

base: main ← compare: add-date ✓ **Able to merge.** These branches can be automatically merged.

Add a title

Add date to hello.sh

Add a description

Write Preview

I want this script to print also current date

Markdown is supported Paste, drop, or click to add files

Reviewers Suggestions

Copilot Request

Copilot can review pull requests

Request a review from Copilot to get fast, actionable feedback on your code, so you can start iterating before you receive a human review.

Learn more OK, dismiss

None yet

Milestone

No milestone

Development

Use [Closing keywords](#) in the description to automatically close issues

Helpful resources

[GitHub Community Guidelines](#)

1 commit 1 file changed 1 contributor

Commits on Feb 7, 2026

add date

anastazie committed 6 minutes ago

e532482 <>

Creating pull request

Add date to hello.sh #1

Edit <> Code

Open [anastazie](#) wants to merge 1 commit into [main](#) from [add-date](#)

Conversation 0 Commits 1 Checks 0 Files changed 1 +1



anastazie commented [now](#)

Owner ...

I want this script to print also current date



Mention [@copilot](#) in a comment to make changes to this pull request.

[add_date](#)

e532482



No conflicts with base branch
Merging can be performed automatically.

Merge pull request You can also merge this with the command line. [View command line instructions.](#)

Still in progress? [Convert to draft](#)



Add a comment

Write

Preview



Add your comment here...

Markdown is supported

Paste, drop, or click to add files

Reviewers

Suggestions

Copilot

Request

Still in progress? [Convert to draft](#)

Assignees

No one—[assign yourself](#)

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

None yet

Notifications

Customize

[Unsubscribe](#)

You're receiving notifications because you authored the thread.

1 participant

Exercise

- Create pull request

Pulling from a remote

- download commits and merge into working copy
 - `git pull`
- simple as that, unless there is a **merge conflict**



Merge conflict

- Changes were made in the same place in the same file in your branch and main branch
- Git cannot decide on its own which version to keep
- You need to decide what to do
 - Use main branch version
 - Use your branch version
 - Combine both

Merge conflict



hello.sh on my_branch

```
1 echo 'Hello'  
2 date  
3 ls -l ~ | wc -l
```

hello.sh on main

```
1 echo 'Hello'  
2 cd ~  
3 mkdir data
```

Resolving conflicts - text files

- check what's conflicted
 - `git status`
- text files need to be edited
 - `nano my-file.txt`
 - `git add my-file.txt`
- commit the resolution, without any message
 - `git commit`
 - recently available: `git merge --continue`

Resolving conflicts - binary files

- check what's conflicted
 - `git status`
- binary files are simple, choose either the incoming (theirs) or ours
 - `git checkout --ours binary-file.jpg`
 - `git add binary-file.jpg`
- commit the resolution, without any message
 - `git commit`
 - recently available: `git merge --continue`

Resolving conflicts

In text files, if both branches modify the same line, a conflict has to be resolved by deleting all the unwanted text (also the marks).

Here are lines that are either unchanged from the common ancestor, or cleanly resolved because only one side changed, or cleanly resolved because both sides changed the same way.

```
<<<<<< yours:sample.txt
```

```
Conflict resolution is hard;  
let's go shopping.
```

```
=====
```

```
Git makes conflict resolution easy.
```

```
>>>>>> theirs:sample.txt
```

```
And here is another line that is cleanly resolved or unmodified.
```

What have we learned?

- We are using Git to have explicit versioning of directory tree
- Git makes snapshot of one or multiple files and saves them as a commit
- Repository is a project inside which git tracks changes
- You can use Github to share and store your repository. To do that you push your local branch to remote
- Branches allows multiple collaborators to work in the same repository without disrupting the main branch
- When you want your code to be in the main branch, you create pull request