

Software Engineering Processes

Course Code: XB_0089

Fernanda Madeiral & Claudia Raibulet

Lecture: Version control systems &
Git & GitHub & Open Source



Bachelor in Computer Science, 2023/2024

Version control systems



Version control systems

Version control systems are software tools that help software teams keep track and manage changes to source code over time.

Version control systems



Change history recording

Independent development

Storage management

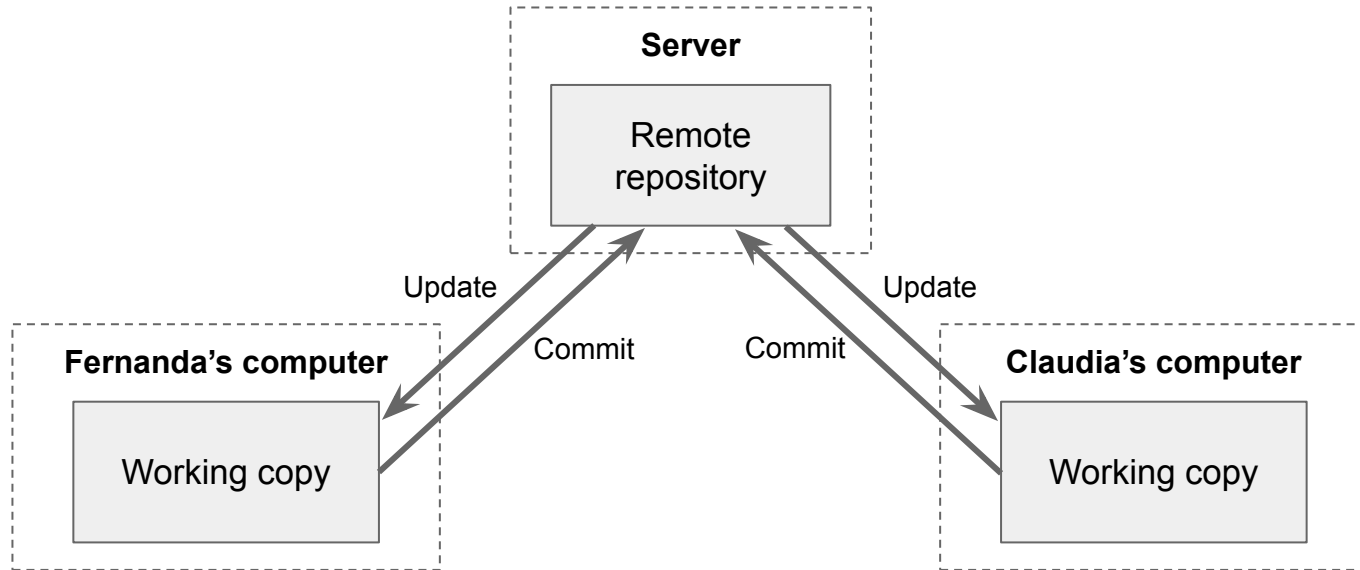
Repositories

Repository: files/folders that contain the project's files and each file's version history.

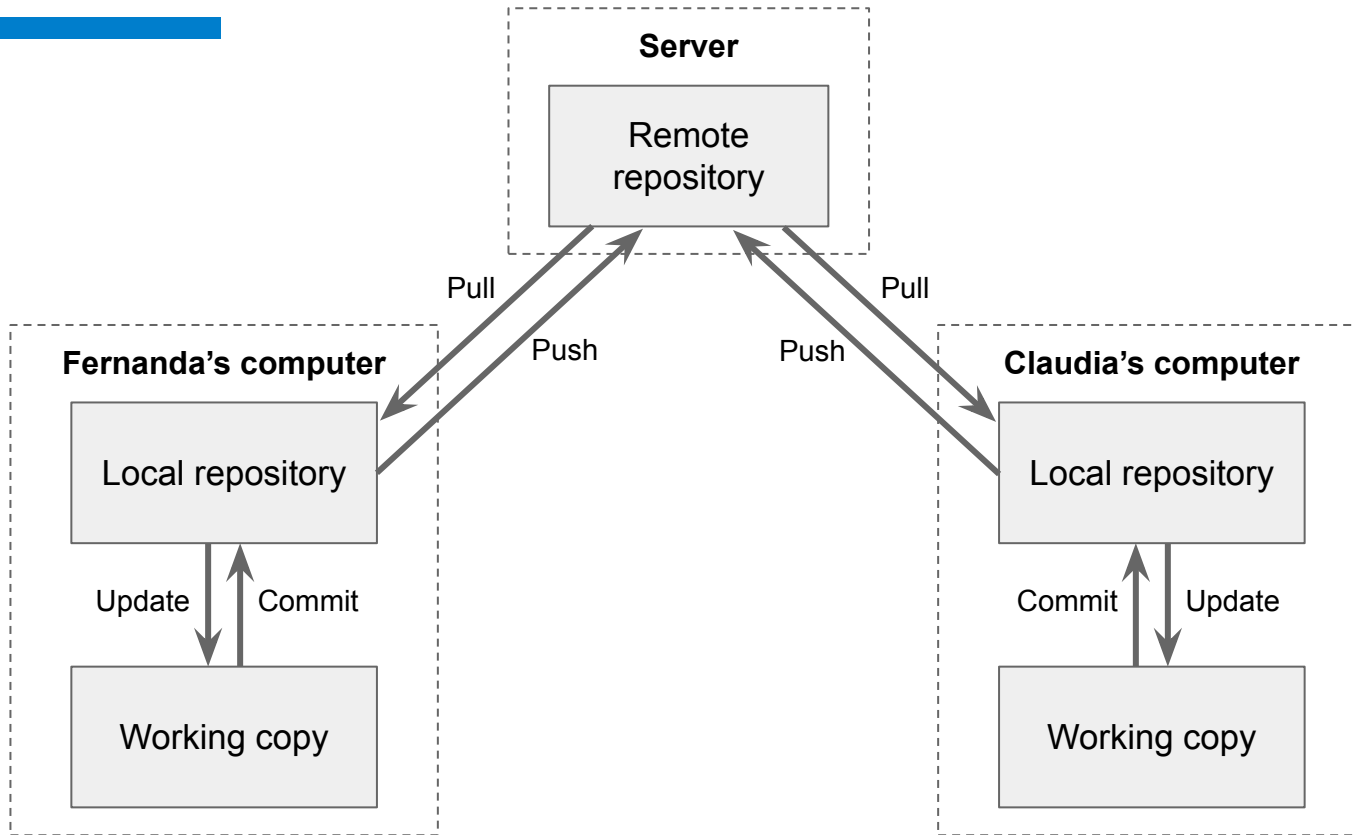
Local repository: a repository stored on a developer computer, only the developer has access to it.

Remote repository: a repository stored on some remote computer, all team members have access to it.

Centralized version control systems



Distributed version control systems



Centralized vs. distributed version control systems

Advantages of distributed version control systems:

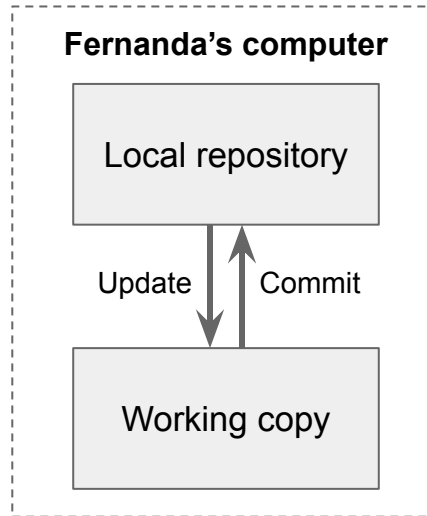
- Flexibility for local work without network connectivity
- Speed to perform commands on the repository
- Resilience to restore a damaged remote repository

Centralized vs. distributed version control systems

Disadvantages of distributed version control systems:

- Storage: if the project has a long history, it might take more disk space in the developers' computers

Local version control



Git



Git is a distributed version control system

Created by Linus Torvalds, the creator of Linux, in 2005

Git

“The best way to learn git is probably to first only do very basic things and not even look at some of the things you can do until you are familiar and confident about the basics.”

– Linus Torvalds

Git

To be studied and practiced:
Slides "SEP2024-Lecture5-git-full.pdf"

Git and GitHub

What is the difference between them?

Git and GitHub



Git	GitHub
It's a system	It's a service
It's locally installed in the computer	It's on the web
It's a command line tool	It has a graphical interface
It's a version control system	It's a code hosting platform for git repositories
It has features to control the version of files	It has features to control the version of files plus other features such as issue tracking, code review, and continuous integration

Types of repositories

Public: anyone on the internet can see it.

Private: you choose who can see and commit to this repository.

GitHub repository “body”

The screenshot shows the GitHub interface for a repository named 'example' by user 'fermadeiral'. The repository is public and has 1 branch (main) and 0 tags. The 'Code' tab is selected, showing the file structure. A red box highlights the 'README.md' file, which contains the text 'example'. A speech bubble points to the file structure, and another points to the README file content.

fermadeiral / example Public

Pin Unwatch 1 Fork 0 Star 0

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags Go to file Add file <> Code

fermadeiral Initial commit fba179a 2 minutes ago 1 commit

README.md Initial commit 2 minutes ago

README.md

example

Readme Activity 0 stars 1 watching

README file

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

Repository URL: <https://github.com/fermadeiral/example>

GitHub repository commit history

The screenshot displays the GitHub interface for the repository 'fermadeiral / example'. At the top, navigation links include Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Below these, repository statistics show 1 branch and 0 tags. A red box highlights the '1 commit' link in the commit history section, which is pointed to by a callout bubble labeled 'Commit history'. The commit history shows a single commit by 'fermadeiral' with the message 'Initial commit', committed '2 minutes ago'. Below the commit history, the 'README.md' file is shown with the content 'example'. On the right side, the 'About' section is visible, along with repository statistics: 0 stars, 1 watching, and 0 forks. The 'Releases' section at the bottom indicates 'No releases published' and provides a link to 'Create a new release'.

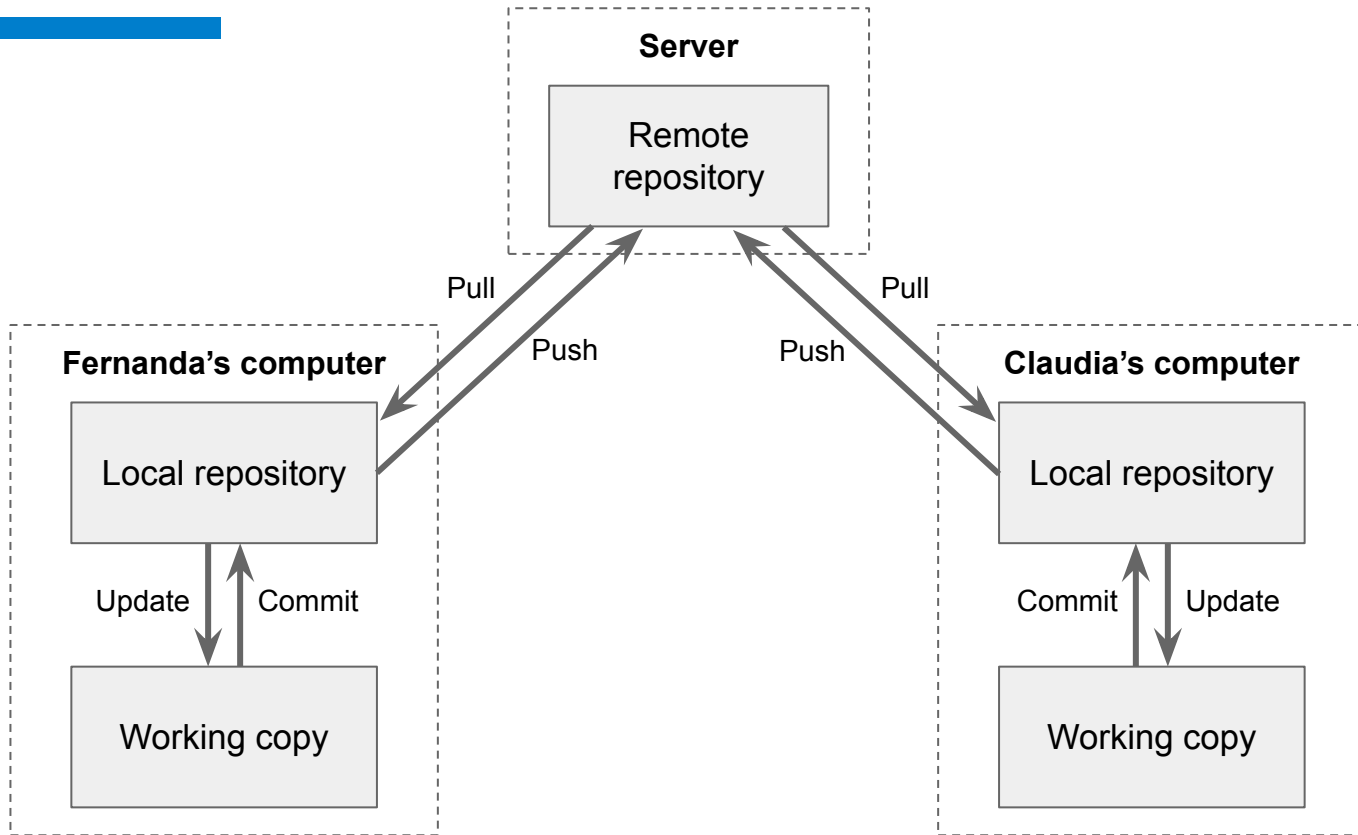
Repository URL: <https://github.com/fermadeiral/example>

GitHub – how to perform changes

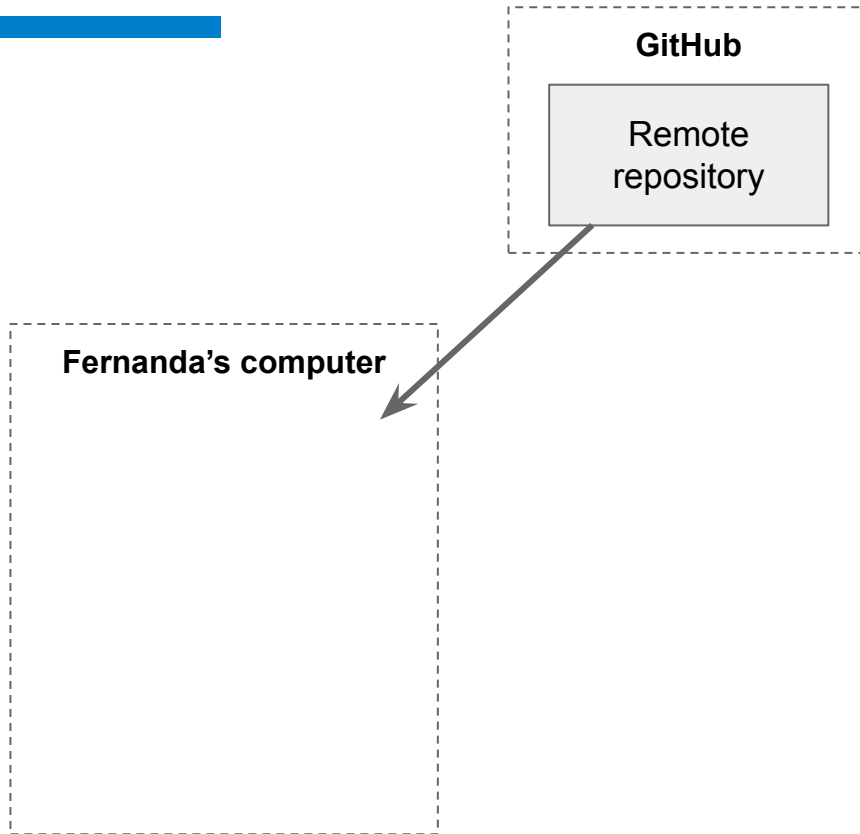
The screenshot shows the GitHub interface for a repository named 'example' by user 'fermadeiral'. The repository is public. At the top, there are buttons for Pin, Unwatch (1), Fork (0), and Star (0). Below this is a navigation bar with links to Code (highlighted with a red box), Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. The main content area shows the 'main' branch with 1 branch and 0 tags. There are buttons for 'Go to file', 'Add file', and a green 'Code' button. Below this, a commit by 'fermadeiral' is shown, labeled 'Initial commit', with hash 'fba179a' and timestamp '2 minutes ago'. A file named 'README.md' is listed as part of the initial commit. The README content is displayed in a large box, showing the word 'example' in a large font. On the right side, there is an 'About' section with a note that no description, website, or topics are provided. Below this are statistics: 0 stars, 1 watching, and 0 forks. At the bottom right, there is a 'Releases' section stating 'No releases published' with a link to 'Create a new release'.

Repository URL: <https://github.com/fermadeiral/example>

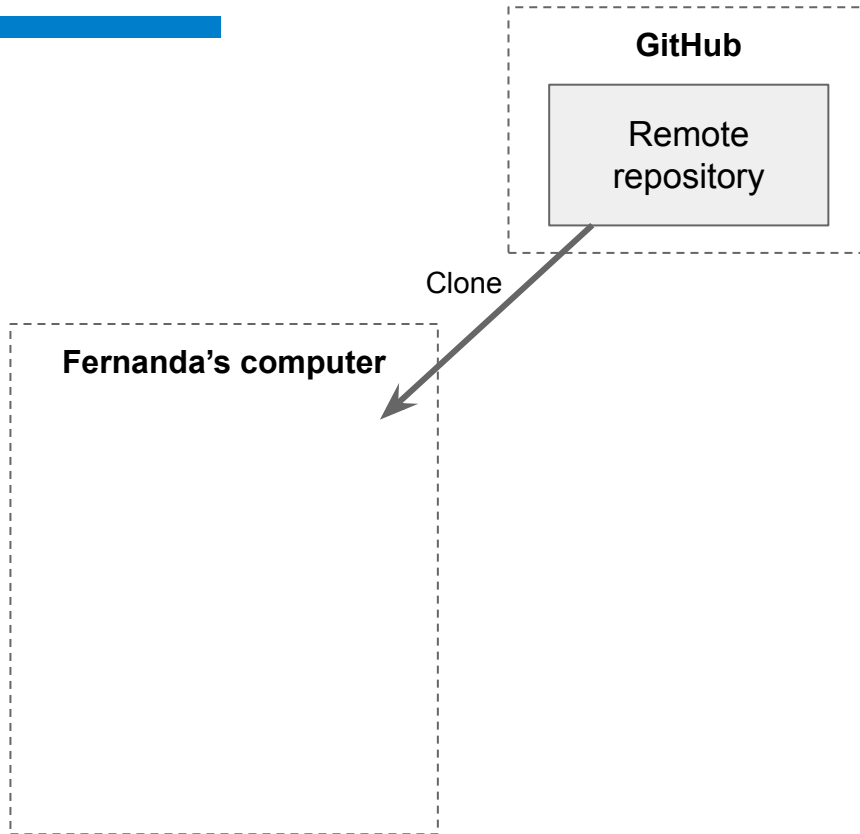
Distributed version control



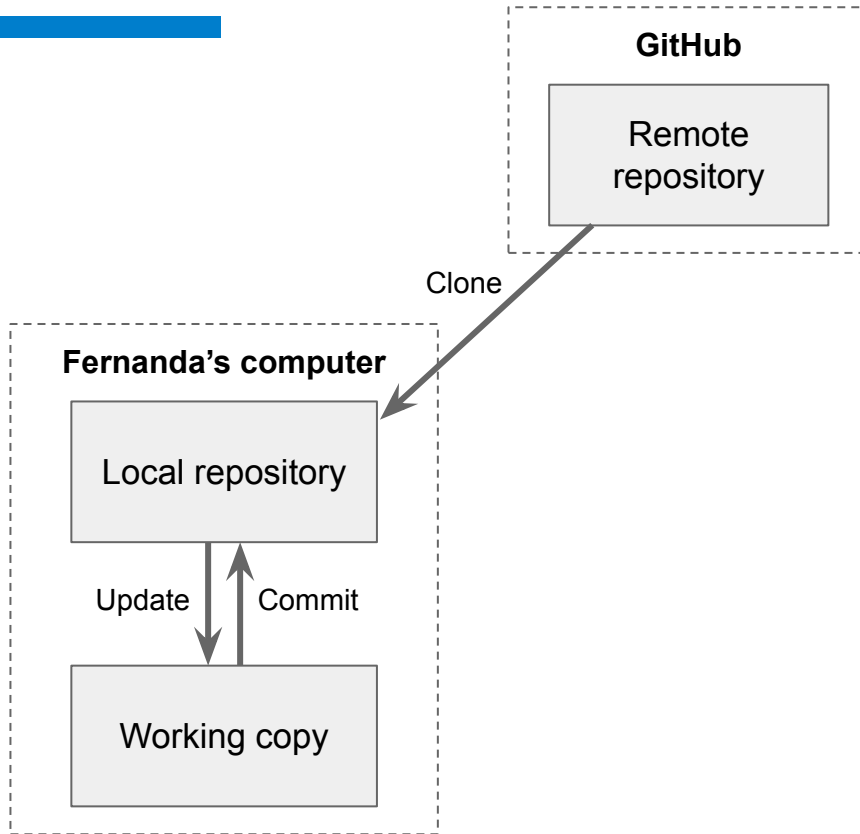
How to start collaborating in a project



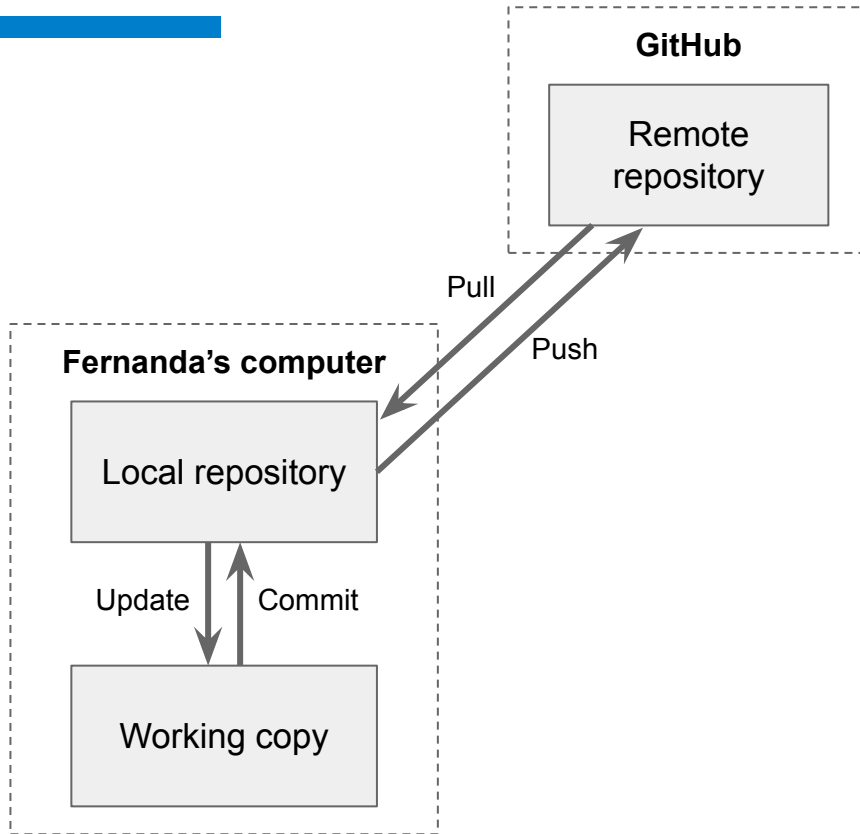
How to start collaborating in a project



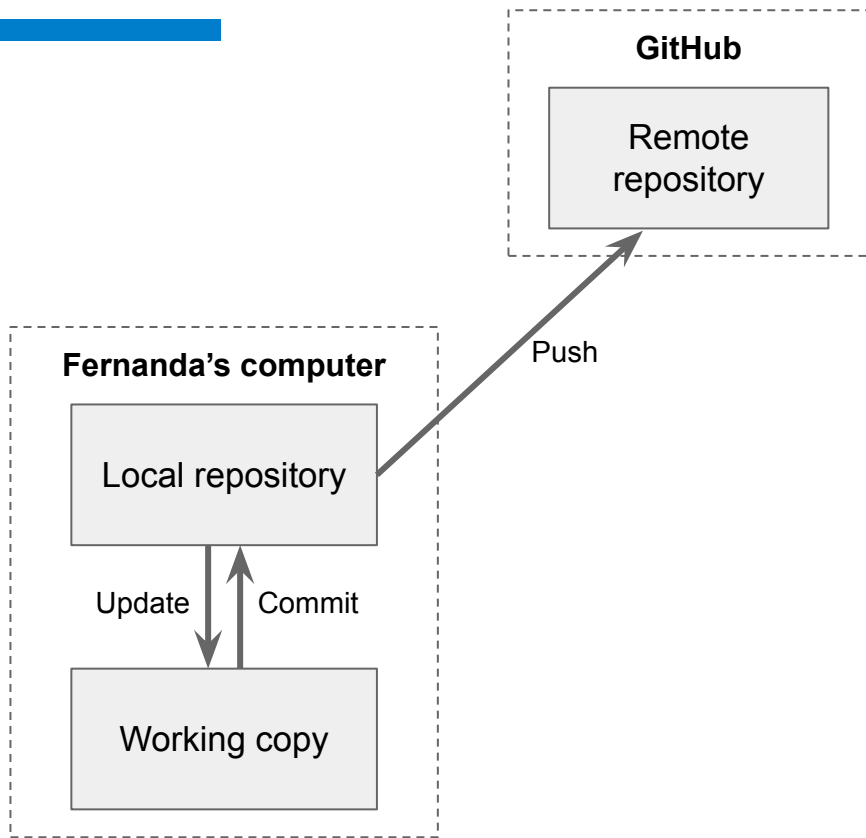
How to start collaborating in a project



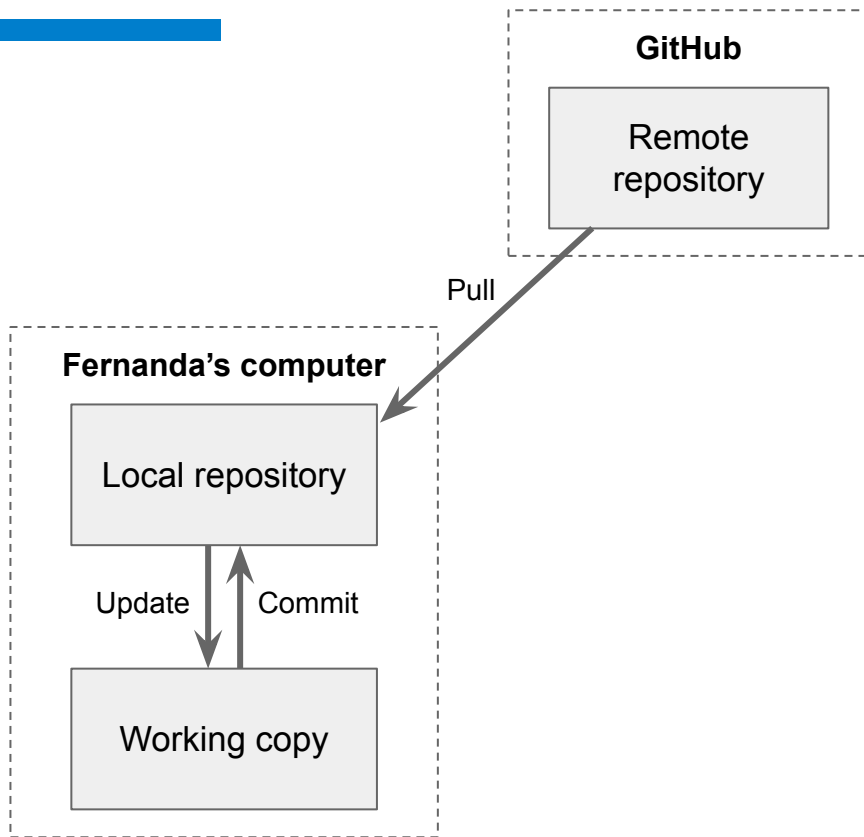
How to start collaborating in a project



How to start collaborating in a project (assuming you have permission)



How to update your local version after changes in the remote by others



GitHub features

The screenshot shows the GitHub interface for a repository named 'example' by user 'fermadeiral'. The repository is public. At the top, there are navigation links: Code, Issues, Pull requests, and Actions. The 'Issues' link is highlighted with a red box. Below the navigation bar, there are buttons for 'Go to file', 'Add file', and 'Code'. The repository has 1 branch (main) and 0 tags. A commit by 'fermadeiral' is shown, titled 'Initial commit', with a commit hash of 'fba179a' and a timestamp of '2 minutes ago'. Below the commit, a file named 'README.md' is listed, also titled 'Initial commit' and timestamped '2 minutes ago'. The content of the README.md file is displayed, showing the word 'example'. On the right side, there is an 'About' section with a description: 'No description, website, or topics provided.' Below this, there are statistics: 0 stars, 1 watching, and 0 forks. At the bottom, there is a 'Releases' section with the text 'No releases published' and a link to 'Create a new release'.

fermadeiral / example Public

Pin Unwatch 1 Fork 0 Star 0

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 1 branch 0 tags Go to file Add file <> Code

fermadeiral Initial commit fba179a 2 minutes ago 1 commit

README.md Initial commit 2 minutes ago

README.md

example

About

No description, website, or topics provided.

Readme Activity 0 stars 1 watching 0 forks

Releases

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

Issues



Issues

To track:

- Ideas
- Feedback
- Tasks
- Bugs

...

The screenshot shows the GitHub Issues interface for the Kotlin project. The top navigation bar includes links for Code, Issues (379), Pull requests (24), Discussions, Actions, Projects, Wiki, and Security. Below the navigation bar, there's a search bar with the filter 'is:issue is:open' and buttons for Labels (24) and Milestones (5). A 'New issue' button is also present. The main content area displays a list of 379 open issues, with 1,741 closed issues. The issues are sorted by default, and the list includes columns for Author, Label, Projects, Milestones, Assignee, and Sort. The visible issues are:

- @Condition badly implemented** (bug) #3270 opened on May 10 by fraf, target 1.6.0
- WriteAccessors prefers 'later' candidates to earlier ones?** #3258 opened on Apr 28 by facboy
- Support for Java 21 Sequenced Collections** (feature) #3240 opened on Apr 16 by filiphr, target 1.7.0
- MapStruct Support idea快捷键shift+f6 替换错误** (bug) #3233 opened on Apr 11 by yuchen-zheng
- Implement `InjectionStrategy.SETTER`** (enhancement, good first issue, up-for-grabs) #3229 opened on Apr 4 by MLNW
- Disable source presence checking for a single field using annotations** #3222 opened on Mar 31 by mjustin
- @InheritConfiguration not support non-abstract method** #3221 opened on Mar 30 by blackc716
- kapt fails with `java.lang.reflect.InvocationTargetException` (no error message)** (bug, kotlin) #3221

Issues – bug report


[Code](#) [Issues 379](#) [Pull requests 24](#) [Discussions](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#)

@Condition badly implemented #3270

Open

fraf opened this issue on May 10 · 1 comment

New issue



fraf commented on May 10

...

Expected behavior

@condition should be a wrapper of the existing generated code (generated without this annotation). Not an overwriter of existing generated code.

Actual behavior

When using @condition, the generated code is rewritten which is bad (in comparison when not present). It needs to wrap it, not rewrite it.

Steps to reproduce the problem

1. Create a classical mapper class :

```
@Mapper(componentModel = "spring", uses = {TaxeMapper.class})
public interface FormuleMapper {

    Formule formuleEntityToDTO(FormuleOption formule);
```

Assignees

No one assigned

Labels

bug

Projects

None yet

Milestone

1.6.0

Development

No branches or pull requests

Issues

Organization:

- Labels
- Milestones
- Assignee
- Traceability

The screenshot shows the GitHub Issues interface for the Kotlin project. The top navigation bar includes links for Code, Issues (379), Pull requests (24), Discussions, Actions, Projects, Wiki, and Security. Below the navigation bar, there's a search bar with the filter 'is:issue is:open' and buttons for 'Labels 24' and 'Milestones 5'. A 'New issue' button is also present. The main content area displays a list of 379 open issues, with a total of 1,741 closed issues. The issues are sorted by default, and the list includes columns for Author, Label, Projects, Milestones, Assignee, and Sort. The issues listed are:

- @Condition badly implemented** (bug) #3270 opened on May 10 by fraf, target 1.6.0
- WriteAccessors prefers 'later' candidates to earlier ones?** #3258 opened on Apr 28 by facboy
- Support for Java 21 Sequenced Collections** (feature) #3240 opened on Apr 16 by filiphr, target 1.7.0
- MapStruct Support idea快捷键shift+f6 替换错误** (bug) #3233 opened on Apr 11 by yuchen-zheng
- Implement InjectionStrategy.SETTER** (enhancement, good first issue, up-for-grabs) #3229 opened on Apr 4 by MLNW
- Disable source presence checking for a single field using annotations** #3222 opened on Mar 31 by mjustin
- @InheritConfiguration not support non-abstract method** #3221 opened on Mar 30 by blackc716
- kapt fails with java.lang.reflect.InvocationTargetException (no error message)** (bug, kotlin) #3221 opened on Mar 30 by blackc716

Pull requests



Pull requests

Main goals:

- To allow **code review** and
- To perform **automated checks** on changes

...before changes are merged in the main branch

Pull requests




Create a new local branch

Perform changes in your working directory and commit them to the local repository, as many times as you want (i.e., multiple commits)


Push the branch to the remote repository

The branch will appear on GitHub

Pull requests

 **branch_for_pr** had recent pushes less than a minute ago


Compare & pull request

 main ▾


Go to file

Add file ▾

<> Code ▾

 **fermadeiral** Update README.md ...


3 hours ago ⌚ 2

 README.md

Update README.md


3 hours ago

README.md





example


Adding changes...


About 


No description, website, or topics provided.

 Readme

 Activity

 0 stars

 2 watching

 0 forks

Releases

No releases published


[Create a new release](#)

Packages

Pull requests


Open a pull request

Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#).

 base: main

← compare: branch_for_pr











✓ **Able to merge.** These branches can be automatically merged.



Update README.md


Write

Preview

H B I       @    

(I'm writing this comment because I don't want to be a bad example in terms of following practices).

This pull request updates the README.md file for showing an example of pull request.

Attach files by dragging & dropping, selecting or pasting them. 

Create pull request

Reviewers

No reviews

Assignees

No one—assign yourself

Labels

None yet

Projects


None yet

Milestone

No milestone

Development

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

 Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

36

Pull requests

Update README.md #1

[Open](#) fermadeiral wants to merge 1 commit into `main` from `branch_for_pr`


Conversation 0

Commits 1

Checks 0

Files changed 1

+1 -1




fermadeiral commented now


Owner

...

(I'm writing this comment because I don't want to be a bad example in terms of following practices).

This pull request updates the README.md file for showing an example of pull request.







Update README.md

414d968

Add more commits by pushing to the `branch_for_pr` branch on **fermadeiral/example**.



Continuous integration has not been set up
[GitHub Actions](#) and [several other apps](#) can be used to automatically catch bugs and enforce style.



This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request

or view [command line instructions](#).

Reviewers

No reviews

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

37

Pull requests

Update README.md #1

[Open](#) fermadeiral wants to merge 1 commit into `main` from `branch_for_pr`

Conversation 0

Commits 1

Checks 0

Files changed 1

+1 -1

Changes from all commits File filter Conversations Jump to

0 / 1 files viewed [Review changes](#)

2 README.md

<> Viewed

...	...	@@ -1,3 +1,3 @@
1	1	# example
2	2	
3		- Adding changes...
	3	+ This is just a change to show an example of pull request.

Pull requests

If the button
“Merge pull
request” is
pressed, what will
happen?

The screenshot shows a GitHub pull request titled "Update README.md #1". At the top, it indicates that user "fermadeiral" wants to merge 1 commit into the `main` branch from the `branch_for_pr` branch. Below this, there are tabs for "Conversation" (0), "Commits" (1), "Checks" (0), and "Files changed" (1). A comment from "fermadeiral" is visible, stating: "(I'm writing this comment because I don't want to be a bad example in terms of following practices). This pull request updates the README.md file for showing an example of pull request." To the right of the comment, there are sections for "Reviewers" (No reviews), "Assignees" (No one—assign yourself), "Labels" (None yet), "Projects" (None yet), "Milestone" (No milestone), and "Development" (Successfully merging this pull request may close these issues.). At the bottom, a green button labeled "Merge pull request" is visible, along with a link to "view command line instructions".

Pull-based development

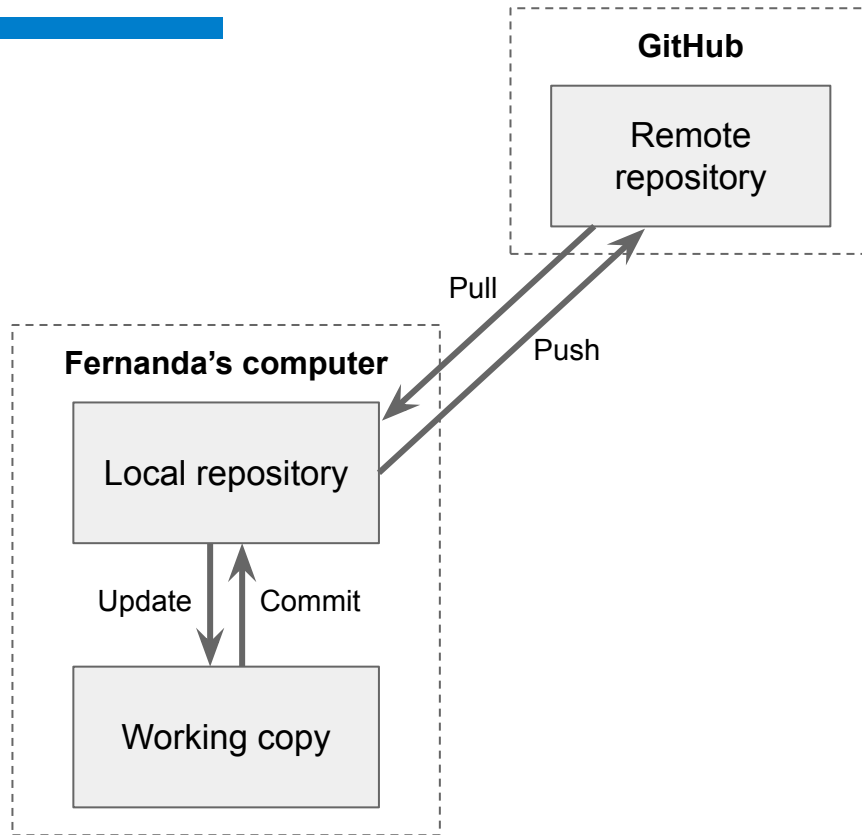


Pull-based development

Main goal:

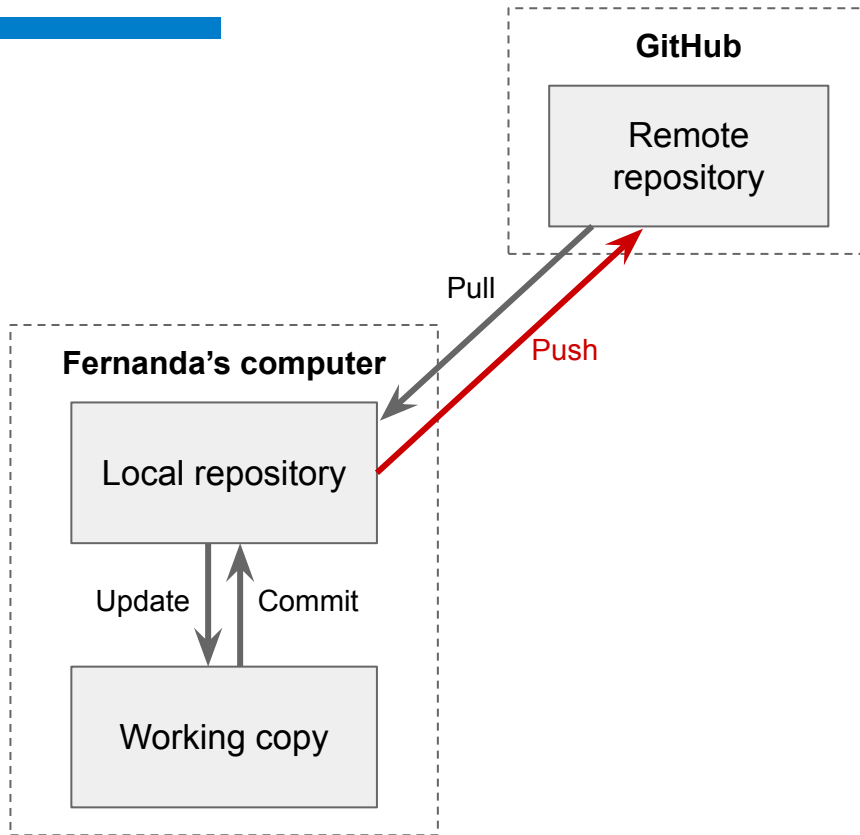
- To allow **external contributions in the repository**

Let's go back...



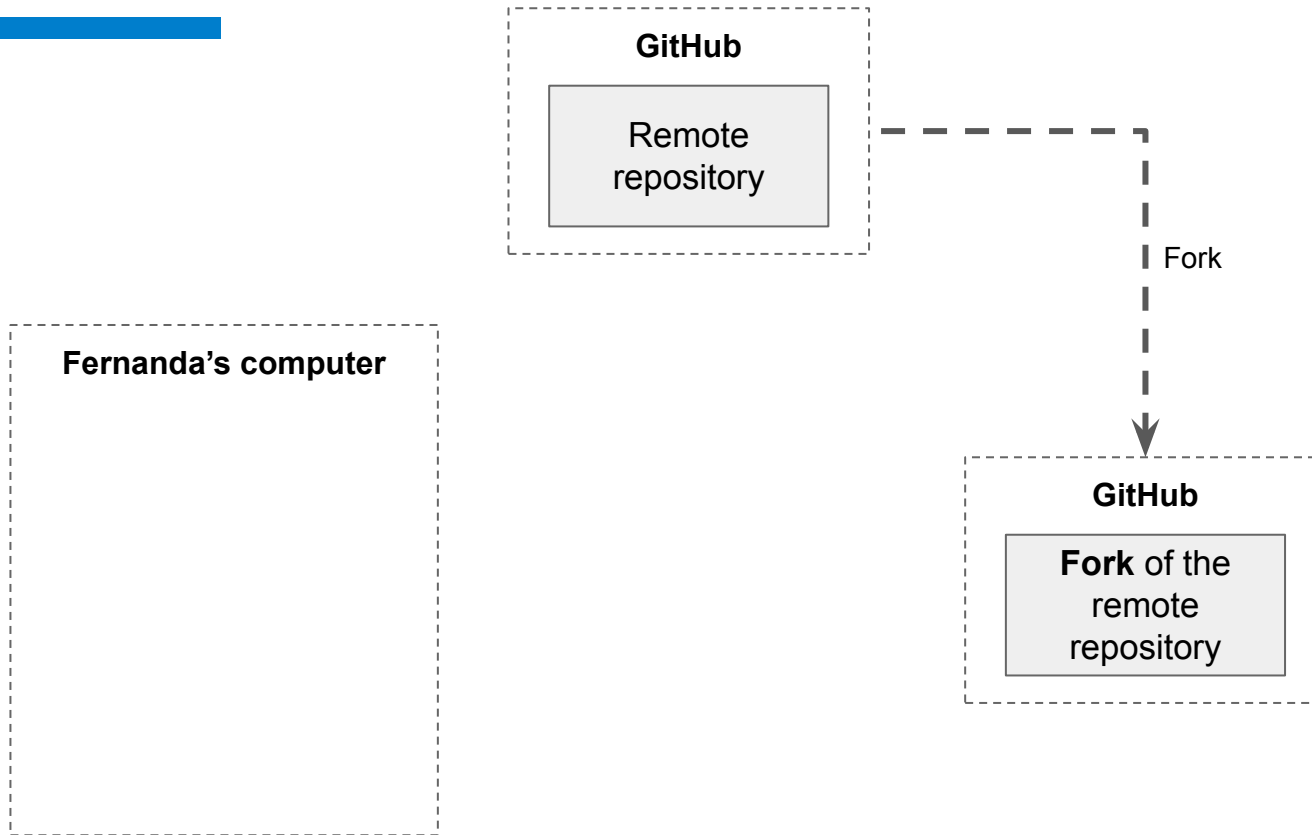
If you don't have permissions on the remote repository, what can't you do here?

Let's go back...

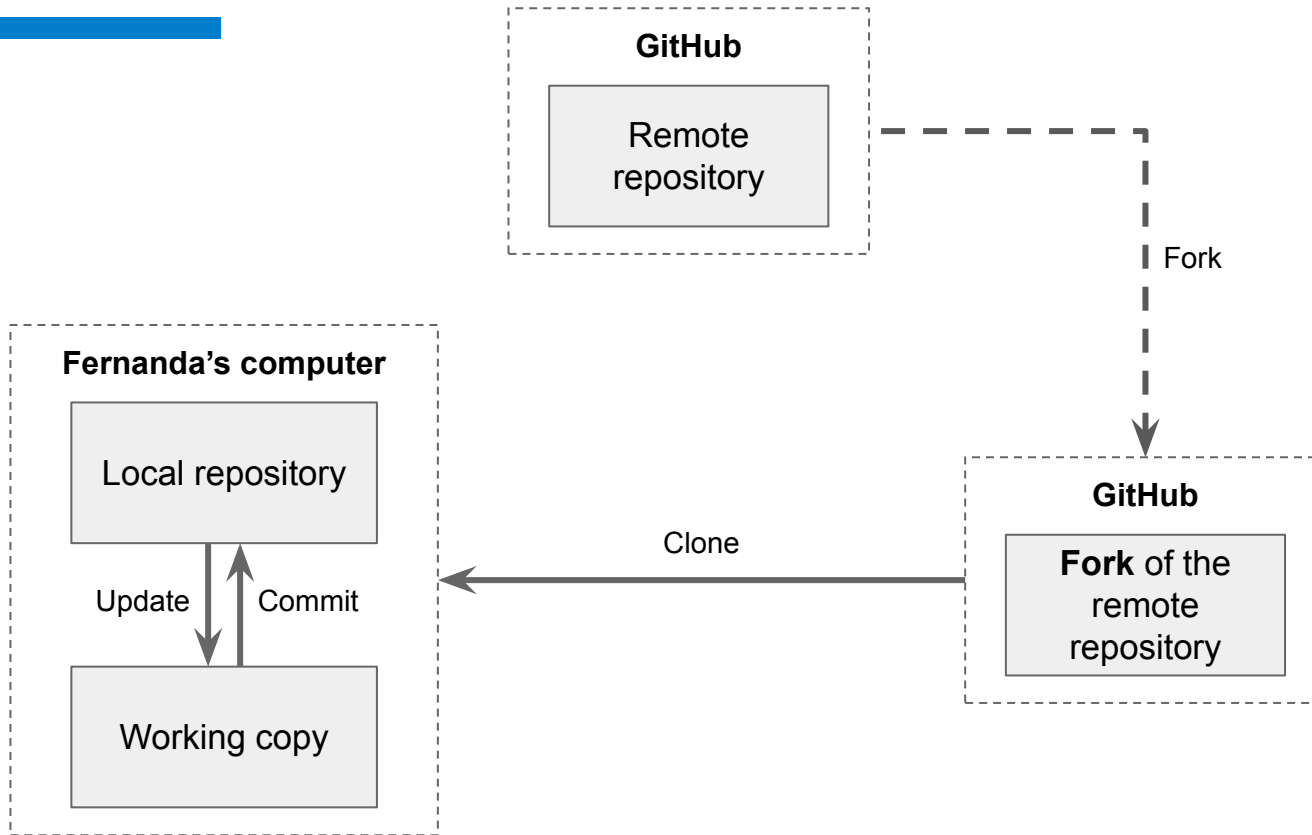


If you don't have permissions on the remote repository, what can't you do here?

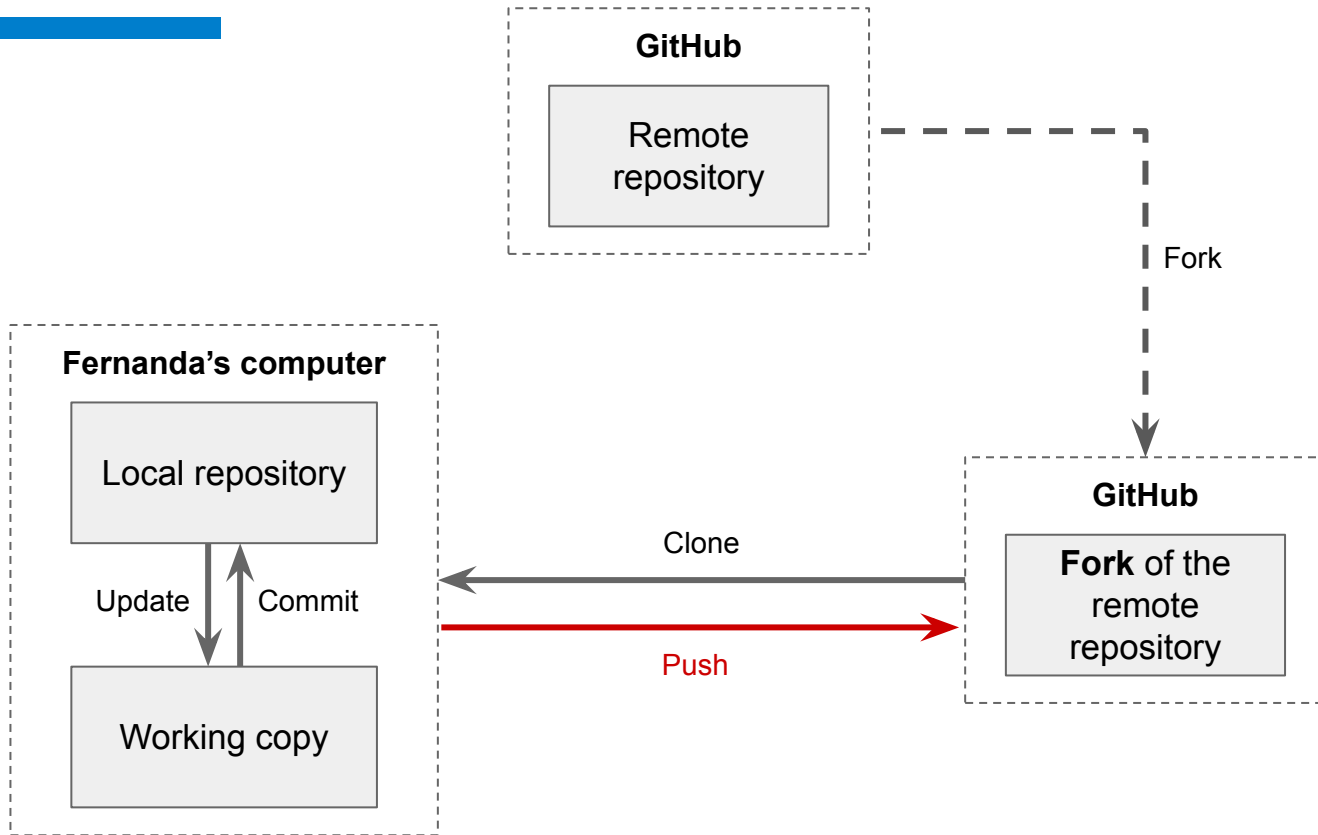
Pull-based development



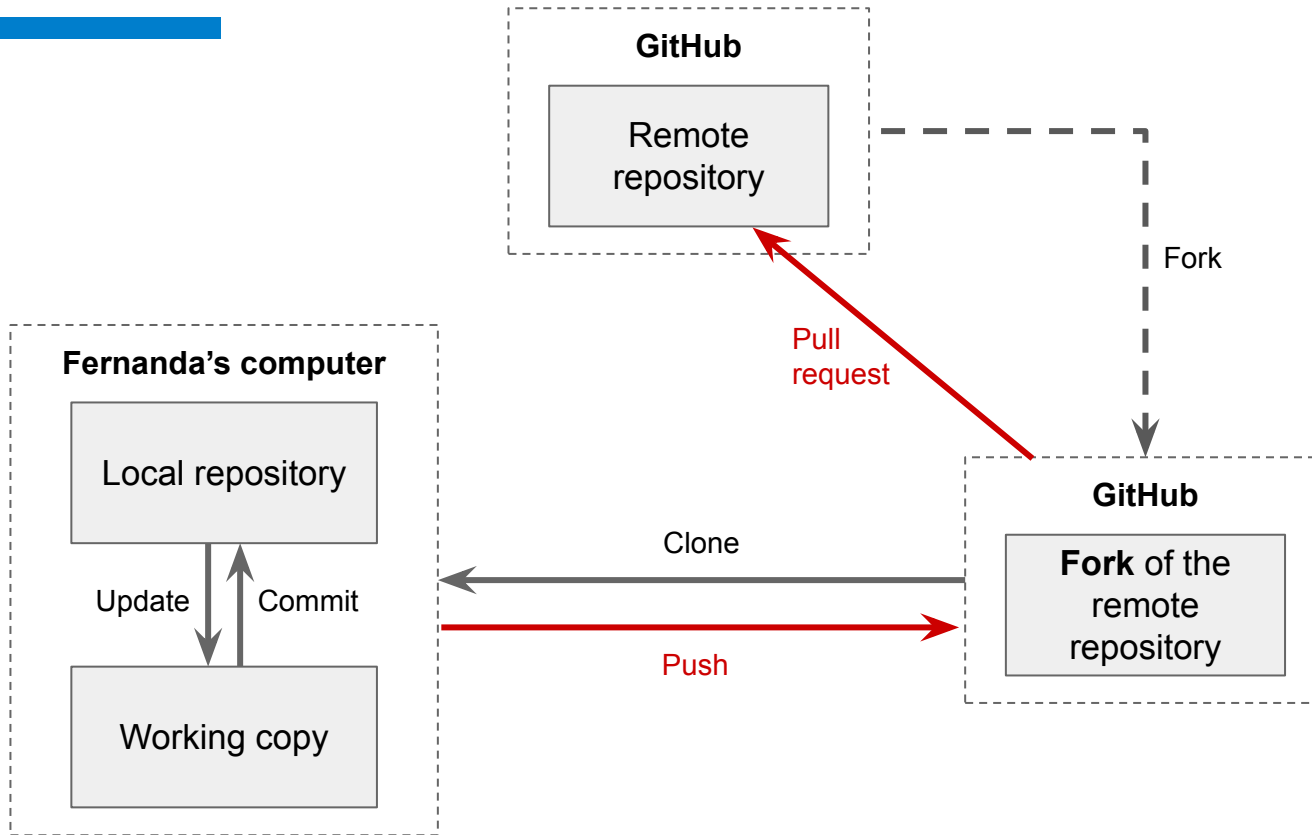
Pull-based development



Pull-based development



Pull-based development



Pull-based development

Main goal:

- To allow **external contributions in the repository**

What other goal do you see with this approach?

GitHub Actions

GitHub Actions

Main goal:

- To **automate** software development tasks and processes

GitHub Actions: workflow

A workflow is a configurable automated process that will run one or more jobs

Workflows are defined in a .yml file checked in to your repository

Workflows are defined in the “.github/workflows” directory in the repository

Workflow basics

A workflow must contain the following basic components:

- One or more **events** that will trigger the workflow
- One or more **jobs**, each of which will execute on a runner machine and run a series of one or more **steps**
- Each step can either **run** a script defined by the developer or a predefined action

GitHub Actions: example



```
name: hello-world
on: push
jobs:
  my-job:
    runs-on: ubuntu-latest
    steps:
      - name: My step
        run: echo "Hello World!"
```

GitHub Actions: example



```
name: hello-world
on: push
jobs:
  my-job:
    runs-on: ubuntu-latest
    steps:
      - name: My step
        run: echo "Hello World!"
```

Event

Job

Step

GitHub Actions: example

```
name: hello-world
on: push
jobs:
  my-job:
    runs-on: ubuntu-latest
    steps:
      - name: My step
        run: echo "Hello World!"
```

The screenshot displays the GitHub Actions interface for a repository named 'hello-world'. The top navigation bar includes links for Code, Issues, Pull requests, Actions (highlighted with a red box), Projects, Wiki, Security, and a menu icon. Below the navigation bar, the workflow run is titled 'Create hello-world.yml #1' with a green checkmark icon. To the right of the title are buttons for 'Re-run all jobs' and a three-dot menu. The left sidebar contains a 'Summary' section with links for 'Jobs', 'Run details', 'Usage', and 'Workflow file'. The 'Jobs' section shows a single job named 'my-job' with a green checkmark. The 'Run details' section shows the job 'my-job' as 'succeeded' with a search bar and a refresh icon. The job details show a list of steps: 'Set up job' (1s), 'My step' (0s), and 'Complete job' (0s). The 'My step' is expanded, showing a log with two lines: '1 Run echo "Hello World!"' and '4 Hello World!'.

GitHub Actions: example



```
name: Java CI
on: push
jobs:
  build:
    runs-on: ubuntu-latest

    steps:
      - uses: actions/checkout@v3
      - name: Set up JDK 17
        uses: actions/setup-java@v3
        with:
          java-version: '17'
          distribution: 'temurin'
      - name: Build with Maven
        run: mvn --batch-mode --update-snapshots package
```

Build and test
with Maven
on push (CI)

More of this on the CI/CD
lecture!

GitHub Actions: example

```
on:
  issues:
    types: opened

jobs:
  comment:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/github-script@v6
        with:
          script: |
            github.rest.issues.createComment({
              issue_number: context.issue.number,
              owner: context.repo.owner,
              repo: context.repo.repo,
              body: '👏 Thanks for reporting!'
            })
```

Comment on an issue
with the predefined
action
“actions/github-script”

Basic collaboration practices



Basic collaboration practices

- Each commit is an atomic change

Each commit does one, and only one simple thing, that can be summed up in a simple sentence

Basic collaboration practices

- Each commit message is clear, follows a pattern, and is traceable

Pattern: For example, using a prefix to state the kind of change in the commit (e.g., "fix:", "feature:")

Traceable: For example, using a suffix with the number of issue or pull request that is linked to an issue

Basic collaboration practices

- Creation of issues to organize small tasks
- Issues should be updated, i.e., closed once addressed, and contain information about the commit(s) in which they were addressed

Basic collaboration practices

- Every commit on the master branch should only be done through the merge of a pull request
- Every pull request should be merged by a different person than who opened it

Basic collaboration practices

In open-source repositories, you can find instructions for collaborations/contributions:

- Sometimes in the README.md
- Sometimes there is a file named CONTRIBUTING.md

Open Source



Free Software



Freedom 0: The freedom to run the program as you wish, for any purpose.

Freedom 1: The freedom to study how the program works, and change it, so it does your computing as you wish. Access to the source code is a precondition for this.

Freedom 2: The freedom to redistribute copies so you can help others in your community.

Freedom 3: The freedom to distribute copies of your modified versions to others. By doing this, you can give the whole community a chance to benefit from your changes. Access to the source code is a precondition for this.

Open Source

Open source stands for criteria a little looser than those of
Free Software

Free Software vs. Open Source

They stand for views based on fundamentally different values:

- For the free software movement, free software is an ethical imperative, essential **respect for the users' freedom**.
- By contrast, the philosophy of open source considers issues in terms of **how to make software “better”**, in a practical sense only.

Licenses

They outline the terms and conditions under which the software can be used, modified, and distributed

Licenses



Free		Proprietary
Permissive	Copyleft (protective or restrictive)	
E.g., MIT	E.g., GPL	No copying, no modification

Licenses (only the ones related to free and open source)

	Permissive	Copyleft (protective or restrictive)
Right to use	Yes	Yes
Right to modify	Yes	Yes
Right to distribute original	Yes, with the same license	Yes, with the same license
Right to distribute modified	Yes, with any license	Yes, with the same license
Right to proprietization	Yes	No

Final remarks

Free and open-source software are similar, but have different values/focus

Licenses are important:

- They specify how software can be used, modified, and distributed
- Unlicensed software cannot be used

TODO



Reading

Exam material:

- Slides “SEP2024-Lecture5-git-full.pdf”
- Slides “SEP2024-Lecture5-GitHub-full.pdf”
- Ian Sommerville, “Engineering Software Products”, 2020: Chapter 10 - “DevOps and Code Management”, Section 10.1 - “Code management¹”

Recommended material:

- <https://git-scm.com/>
- GitHub Docs: <https://docs.github.com/en>
- Book: Gustavo Pinto, “Open Source Licensing 101”, 2020

¹ “Code management systems” is the same as “version control systems”

Takeaways?

