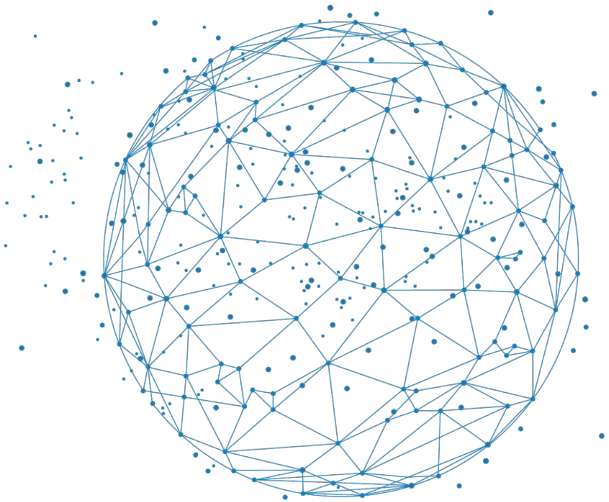
2025 November 27th



Improving the FAIRness of Your GitLab

DENECKER Thomas
CNRS—IFB
0000-0003-1421-7641

MESSAK Imane
CNRS—IFB
0000-0002-1654-6652

ROUSSEAU Baptiste
CNRS—IFB
0009-0002-1723-2732

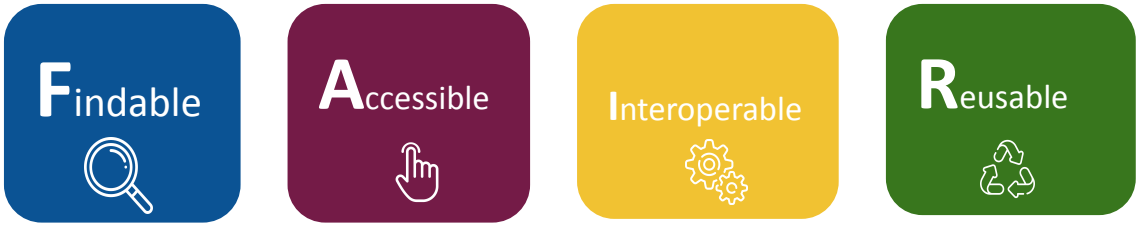
Réseau MERIT MetiER en bioinformaTique

IFB INSTITUT FRANÇAIS DE BIOINFORMATIQUE

cnrs

INRAE

FAIR principles



FAIR principles

Findable

Persistent IDentifier (PID)

Allows for the stable and unique identification of web resources

There are many forms, the best known of which is the DOI

DOI 10.5281/zenodo.7729199

A PID creates a unique and persistent hyperlink

PIDs facilitate data citation

Metadata

Metadata finely describe the ressource (data).

Examples of metadata: title, author, publication date.

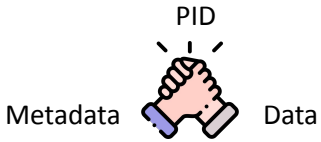
Metadata facilitate searching (sort by, filter, categories,...)

Be generous in your description of your data !

Metadata with PID

The metadata describing the data are often separate files.

By adding the PID to the metadata, the link between the data and the metadata becomes explicit.



Data Repository

Data must be deposited in a data repository

A repository is a web service for hosting, searching, and downloading data

Repositories offer services such as PID assignment, secure storage, a licence, long-term archiving, ...

FAIR principles

Accessible

Standard protocol

Data must be retrievable via a standardized communication protocol

A communication protocol is a set of procedures for communication between machines.

Examples: HTTP (for web pages) and FTP (for files)

Free and Open protocol

Freely usable and interoperable

Facilitating free access to data

HTTP, FTP, SMTP ✓

Microsoft Exchange Server ✗

Authentification

Accessible ≠ free and open

"*As open as possible, as closed as necessary*"

Authentication may be necessary for sensitive data (personal data, national security, …)

Secure protocol (HTTPS, SFTP)

Metadata access

Maintain access to metadata even if the data is no longer accessible

Data can disappear, metadata will therefore be very useful for understanding why and taking precautions (in the event of damage, for example).

FAIR principles

Controlled vocabulary

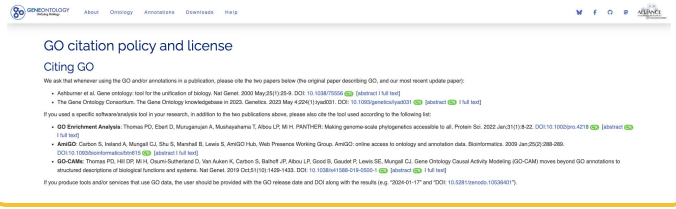
Use a controlled vocabulary to index and find knowledge

A controlled vocabulary is a list of predefined terms

Commun language for machines

FAIR vocabulary

A vocabulary that respects the FAIR principles must be used (ie: PID)



Interoperabl

Linked Metadata

Link data to enrich their context

The FAIR principles rely on the Web of data (RDF)

Links between data follow a three-part structure (triplet): subject, predicate, object

Repositories like Nakala use RDF technology to link data

FAIR principles

Metadata with attributes

Data must be described with metadata and relevant attributes.

Provide as much information as possible on the context of data production/collection

All information are important, even those that seem unnecessary !

Licence

In France, publicly funded research data must be open and freely reusable (with some exceptions)

Add a license even to say that they are freely accessible and reusable

Choosing the right license isn't always easy, but there are a number of resources to help you find your way around

Reusable

Provenance

how were the data obtained?

Tools used, parameters used, associated data, degree of uncertainty, etc.

Add READ ME, notebook, requirement file, …

Community Standards

Use community standards to facilitate data reuse

Use open file formats to make your data available to your peers

Reminder of the good practices

Versioning → **Versioning / sharing** → **Sharing** → **Archiving**

GitLab

GitHub

Atlassian Bitbucket

zenodo

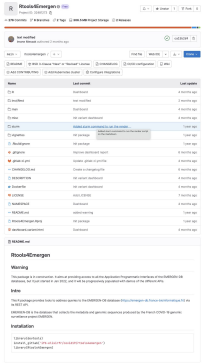
Software Heritage

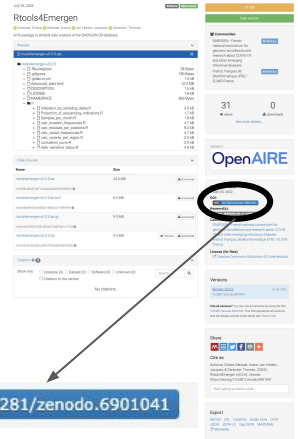
VS Code

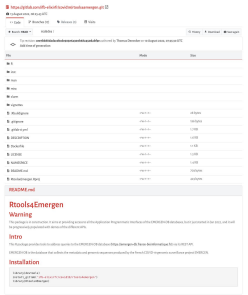
Studio

Sourcetree

...





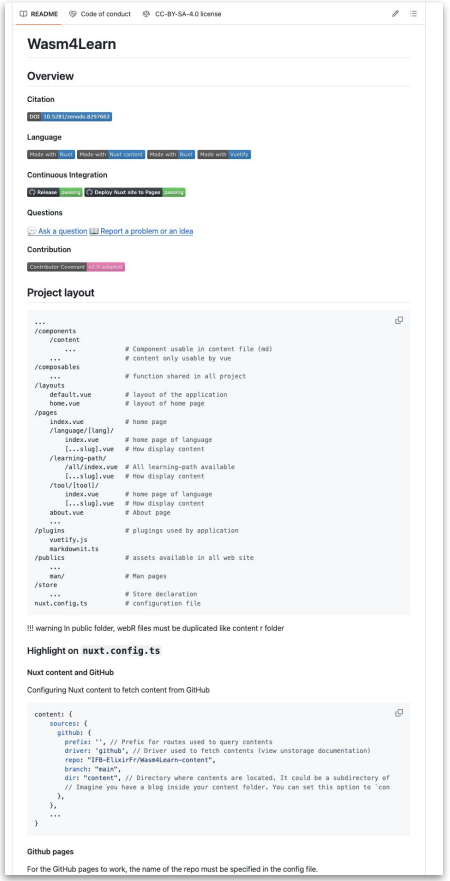


README

The README is the **first thing** users and contributors see → **it's your project's entry point.**

A README file is essential for guiding users, documenting the project, encouraging contribution, and promoting development best practices.

Resources: *https://www.makeareadme.com/*
https://readme.so/



License

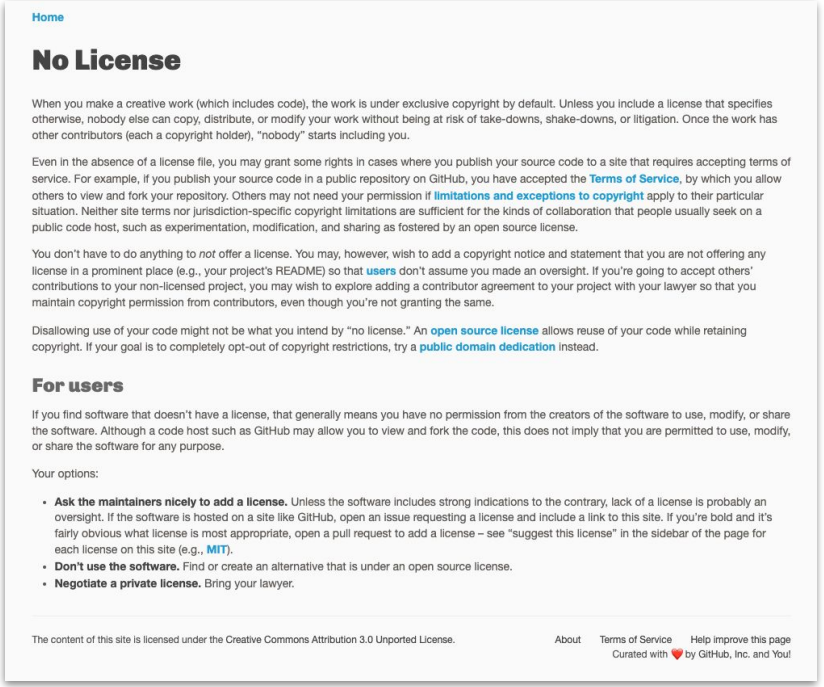
Why add a license?

- Grants users legal clarity on how they can use, share, or modify your code
- Essential for reuse and open collaboration

What happens when you add a code with no license, by default:

- Nobody has legal permission to use, modify, or share it, not even for personal or academic use.
- The code is considered “all rights reserved” under copyright law.
- Even open access (e.g. on GitHub) does not mean open source.

This creates uncertainty and discourages reuse, collaboration, and reproducibility, especially in science and open-source communities.



Home

No License

When you make a creative work (which includes code), the work is under exclusive copyright by default. Unless you include a license that specifies otherwise, nobody else can copy, distribute, or modify your work without being at risk of take-downs, shake-downs, or litigation. Once the work has other contributors (each a copyright holder), “nobody” starts including you.

Even in the absence of a license file, you may grant some rights in cases where you publish your source code to a site that requires accepting terms of service. For example, if you publish your source code in a public repository on GitHub, you have accepted the **Terms of Service**, by which you allow others to view and fork your repository. Others may not need your permission if **limitations and exceptions to copyright** apply to their particular situation. Neither site terms nor jurisdiction-specific copyright limitations are sufficient for the kinds of collaboration that people usually seek on a public code host, such as experimentation, modification, and sharing as fostered by an open source license.

You don't have to do anything to *not* offer a license. You may, however, wish to add a copyright notice and statement that you are not offering any license in a prominent place (e.g., your project's README) so that **users** don't assume you made an oversight. If you're going to accept others' contributions to your non-licensed project, you may wish to explore adding a contributor agreement to your project with your lawyer so that you maintain copyright permission from contributors, even though you're not granting the same.

Disallowing use of your code might not be what you intend by “no license.” An **open source license** allows reuse of your code while retaining copyright. If your goal is to completely opt-out of copyright restrictions, try a **public domain dedication** instead.

For users

If you find software that doesn't have a license, that generally means you have no permission from the creators of the software to use, modify, or share the software. Although a code host such as GitHub may allow you to view and fork the code, this does not imply that you are permitted to use, modify, or share the software for any purpose.

Your options:

- **Ask the maintainers nicely to add a license.** Unless the software includes strong indications to the contrary, lack of a license is probably an oversight. If the software is hosted on a site like GitHub, open an issue requesting a license and include a link to this site. If you're bold and it's fairly obvious what license is most appropriate, open a pull request to add a license – see “suggest this license” in the sidebar of the page for each license on this site (e.g., **MIT**).
- **Don't use the software.** Find or create an alternative that is under an open source license.
- **Negotiate a private license.** Bring your lawyer.

The content of this site is licensed under the Creative Commons Attribution 3.0 Unported License.

About Terms of Service Help improve this page

Curated with ❤ by GitHub, Inc. and You!

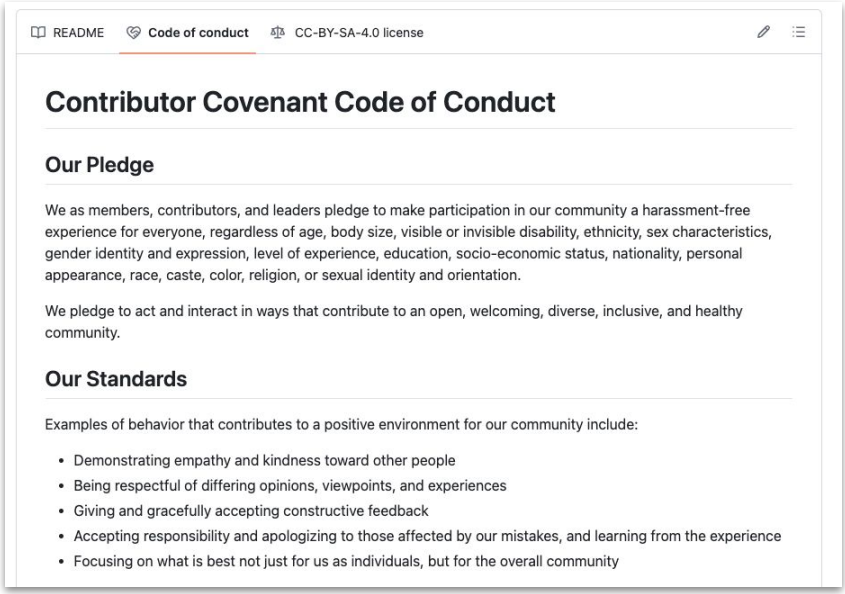
source: https://choosealicense.com/no-permission/

Code of Conduct

A code of conduct helps to create a positive and respectful environment where all community members can contribute meaningfully and feel safe.

Where to find one for your repository ?

https://www.contributor-covenant.org/version/2/1/code_of_conduct/


README Code of conduct CC-BY-SA-4.0 license

Contributor Covenant Code of Conduct

Our Pledge

We as members, contributors, and leaders pledge to make participation in our community a harassment-free experience for everyone, regardless of age, body size, visible or invisible disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, caste, color, religion, or sexual identity and orientation.

We pledge to act and interact in ways that contribute to an open, welcoming, diverse, inclusive, and healthy community.

Our Standards

Examples of behavior that contributes to a positive environment for our community include:

- Demonstrating empathy and kindness toward other people
- Being respectful of differing opinions, viewpoints, and experiences
- Giving and gracefully accepting constructive feedback
- Accepting responsibility and apologizing to those affected by our mistakes, and learning from the experience
- Focusing on what is best not just for us as individuals, but for the overall community

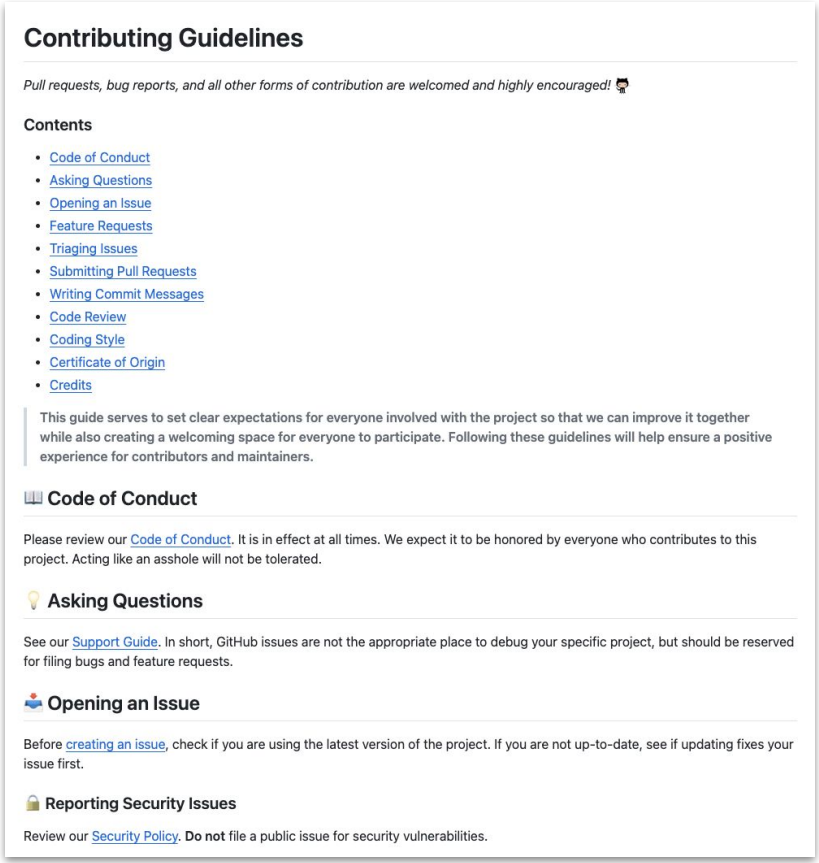
Contributing file

A CONTRIBUTING file serves as a valuable resource for guiding contributors, clarifying expectations, promoting collaboration, and fostering a positive and inclusive community around the project.

What to include:

- How to submit an issue / MR
- Commit message guidelines
- Testing, CI, review process

Documentation
https://docs.github.com/fr/communities/setting-up-your-project-for-healthy-contributions/setting-guidelines-for-repository-contributors



Contributing Guidelines

Pull requests, bug reports, and all other forms of contribution are welcomed and highly encouraged!

Contents

- Code of Conduct
- Asking Questions
- Opening an Issue
- Feature Requests
- Triaging Issues
- Submitting Pull Requests
- Writing Commit Messages
- Code Review
- Coding Style
- Certificate of Origin
- Credits

> This guide serves to set clear expectations for everyone involved with the project so that we can improve it together while also creating a welcoming space for everyone to participate. Following these guidelines will help ensure a positive experience for contributors and maintainers.

Code of Conduct

Please review our Code of Conduct. It is in effect at all times. We expect it to be honored by everyone who contributes to this project. Acting like an asshole will not be tolerated.

Asking Questions

See our Support Guide. In short, GitHub issues are not the appropriate place to debug your specific project, but should be reserved for filing bugs and feature requests.

Opening an Issue

Before creating an issue, check if you are using the latest version of the project. If you are not up-to-date, see if updating fixes your issue first.

Reporting Security Issues

Review our Security Policy. **Do not** file a public issue for security vulnerabilities.

What is CodeMeta?

- A JSON-LD vocabulary / schema to describe software metadata.
- Helps make your software more findable, citable, and interoperable

How to use it:

- Use the *CodeMeta* generator (https://codemeta.github.io/codemeta-generator/) or manually create a codemeta.json at the repo root
- Include key metadata: authors, license, repository URL, keywords, CI info


```
{
  "@context": "https://w3id.org/codemeta/3.0",
  "type": "SoftwareSourceCode",
  "author": [
    {
      "id": "https://orcid.org/0000-0003-1421-7641",
      "type": "Person",
      "affiliation": {
        "type": "Organization",
        "name": "CNRS - IFB"
      },
      "email": "thomas.denecker@france-bioinformatique.fr",
      "familyName": "Denecker",
      "givenName": "Thomas"
    },
    {
      "id": "https://orcid.org/0000-0002-1654-6652",
      "type": "Person",
      "affiliation": {
        "type": "Organization",
        "name": "CNRS - IFB"
      },
      "email": "imane.messak@france-bioinformatique.fr",
      "familyName": "Messak",
      "givenName": "Imane"
    },
    {
      "id": "https://orcid.org/0009-0002-1723-2732",
      "type": "Person",
      "affiliation": {
        "type": "Organization",
        "name": "CNRS - IFB"
      },
      "email": "baptiste.rousseau@france-bioinformatique.fr",
      "familyName": "Rousseau",
      "givenName": "Baptiste"
    }
  ],
  "codeRepository": "https://gitlab.com/anf-workflow-et-reproductibilite/improving-the-fairness-of-your-gitlab",
  "dateCreated": "2025-11-20",
  "datePublished": "2025-11-20",
  "description": "Repository for improving the FAIRness and reproducibility of a GitLab projects.",
  "keywords": [
    "FAIR",
    "Reproducibility",
    "GitLab",
    "Best Practices",
    "Open Science"
  ],
  "license": "https://spdx.org/licenses/BSD-3-Clause",
  "name": "Improving the FAIRness of your GitLab",
  "operatingSystem": [
    "Linux",
    "Windows",
    "macOS"
  ],
  "programmingLanguage": "Python",
  "softwareRequirements": "https://gitlab.com/anf-workflow-et-reproductibilite/improving-the-fairness-of-your-gitlab/-/blob/main/pyproject.toml",
  "continuousIntegration": "https://gitlab.com/anf-workflow-et-reproductibilite/improving-the-fairness-of-your-gitlab/-/pipelines",
  "developmentStatus": "active",
  "isSourceCodeOf": "Improving the FAIRness of your GitLab projects",
  "issueTracker": "https://gitlab.com/anf-workflow-et-reproductibilite/improving-the-fairness-of-your-gitlab/-/issues"
}
```

Conventional Commits

A convention for commit messages using prefixes such as feat:, fix:, docs:, chore:, etc.

Why it's useful ?

- Provides a structured commit history
- Enables automation (e.g., changelog generation, release versioning)

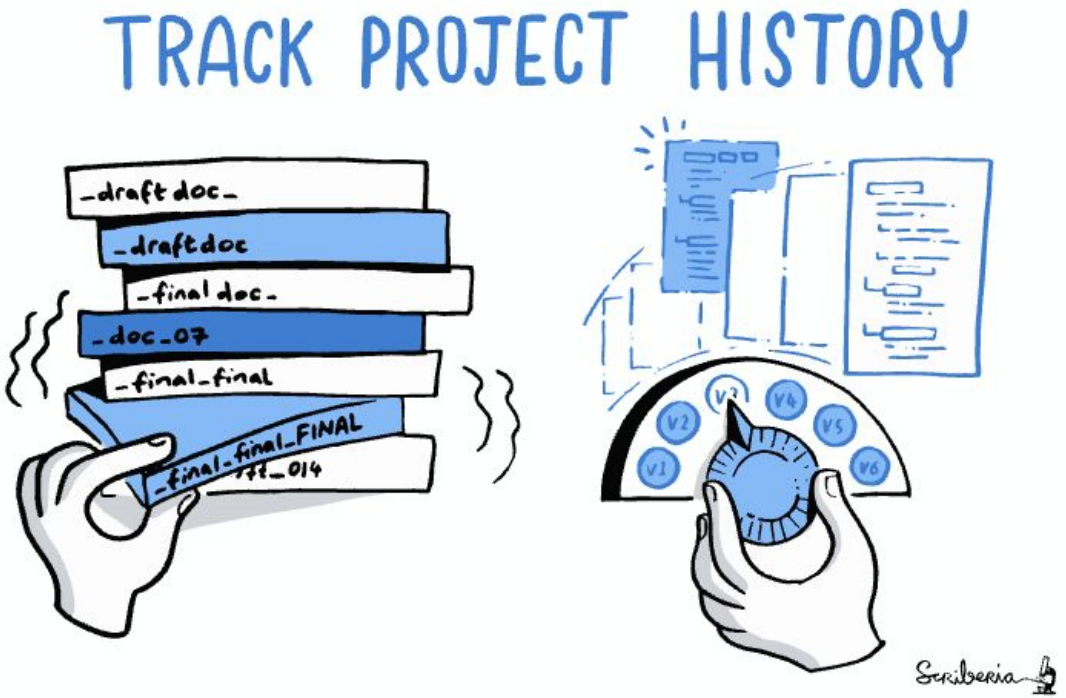
All you have to do is write your commits following the Conventional Commits
https://www.conventionalcommits.org/en/v1.0.0/

```
<type>[optional scope]: <description>

[optional body]

[optional footer(s)]
```

Example

```
feat: allow provided config object to extend other configs
```

Release


TRACK PROJECT HISTORY
draft doc
_draftdoc
final doc
_doc_07
_final_final
_final_final_FINAL
V1
V2
V3
V4
V5
V6
Scribbria

Goal

Provide users with a version of your code that has been fixed in time and labeled.

All steps are detailed here

https://docs.gitlab.com/user/project/releases/

Semantic of a release number

1.0.0

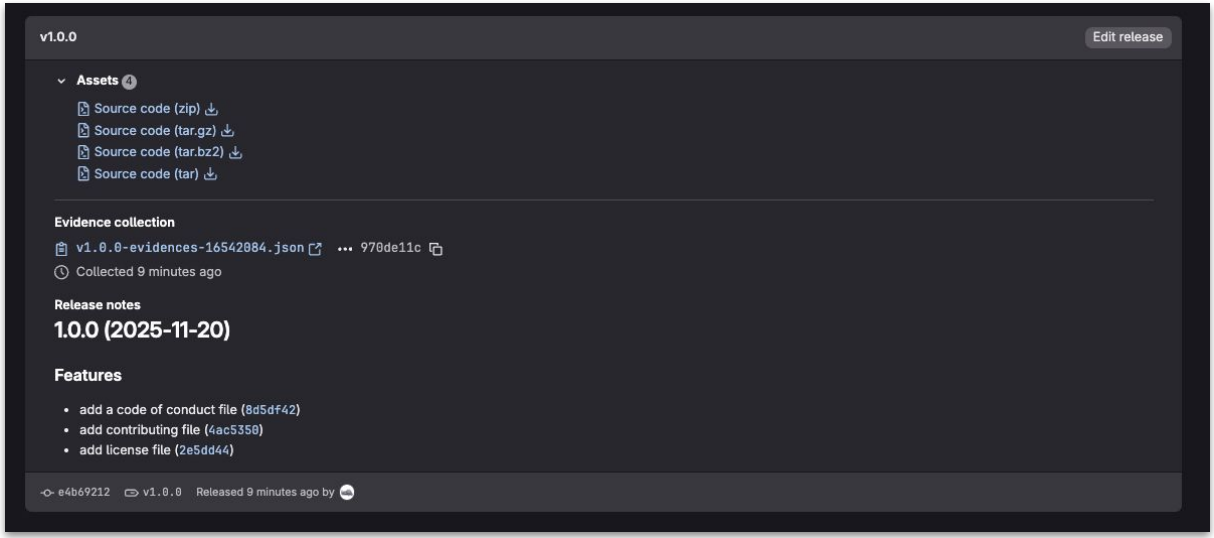
MAJOR.MINOR.PATCH

MAJOR : changes not backwards-compatible

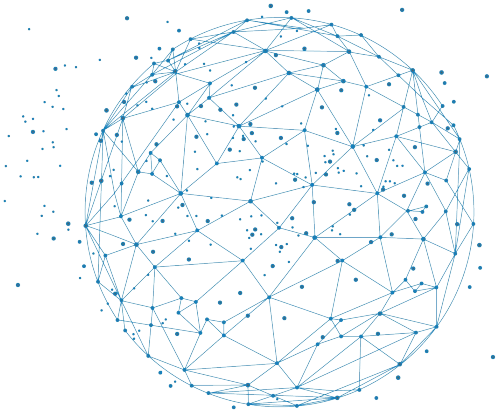
MINOR : new/modified functionalities, backwards-compatible

PATCH : bug fixes, backwards-compatible More details : https://semver.org/

Make a release



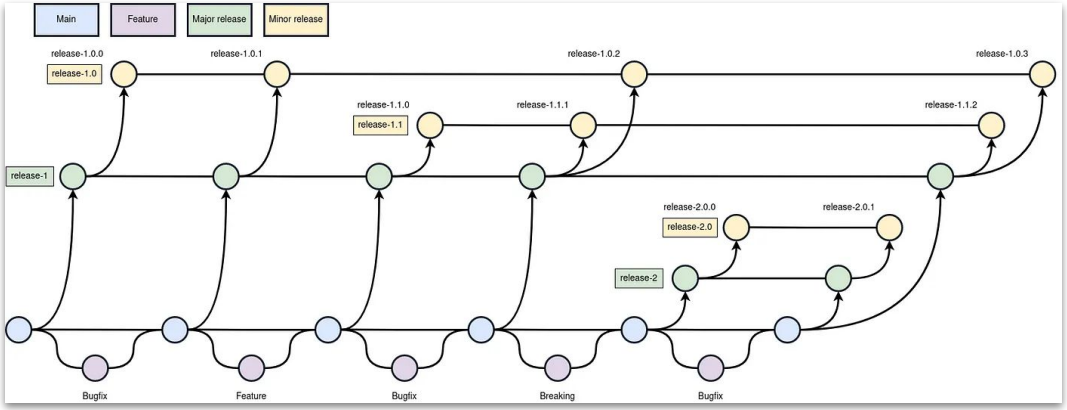
Semantic-release



semantic-release

semantic-release automates the whole package release workflow, including determining the next version number, generating the release notes, and publishing the package.

Documentation available here: https://semantic-release.gitbook.io/semantic-release



Source: https://medium.com/@gordon.messmer/semantic-releases-part-1-an-example-process-7b99d6b872ab

How to set it up in GitLab CI:

1. Use a .gitlab-ci.yml with a release job that runs semantic-release when on a release branch (e.g., main)

2. Provide a GitLab Token (GL_TOKEN or GITLAB_TOKEN) in CI so semantic-release can push tags and create releases

3. Use the *@semantic-release/gitlab* plugin to publish GitLab releases

```yaml
release:
  stage: release
  image: node:20
  before_script:
    - npm ci
  script:
    - npm run semantic-release
  rules:
    - if: $CI_COMMIT_BRANCH == "main"
```

Example of gitlab-ci.yml file

Release: Why conventional commits is useful

- Commit messages structured like

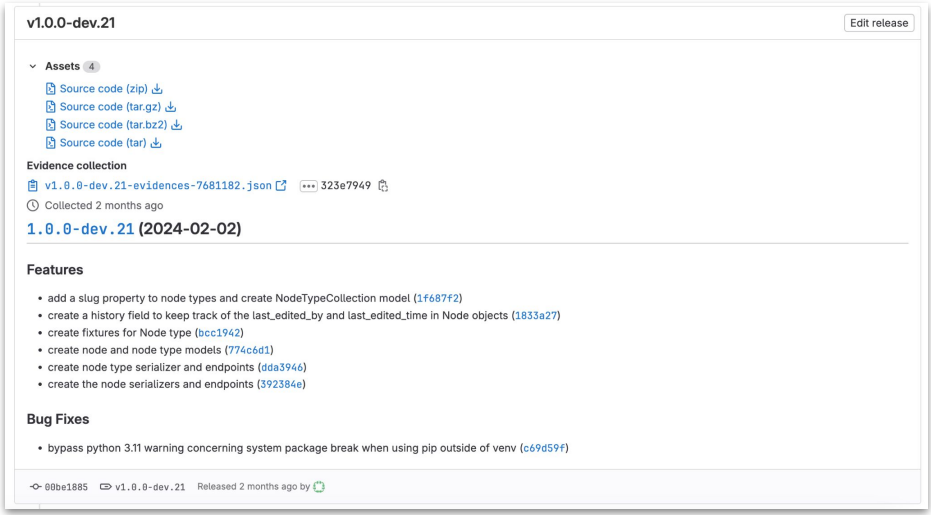
```
fix: .. or feat: … + Breaking changes
```

  let semantic- release know whether to create a patch, minor or major release.

- This helps remove guesswork and ensure automated version bumping is consistent.

Release and changelog

Releases



v1.0.0-dev.21

Edit release

Assets 4

- Source code (zip)
- Source code (tar.gz)
- Source code (tar.bz2)
- Source code (tar)

Evidence collection

v1.0.0-dev.21-evidences-7681182.json ••• 323e7949

Collected 2 months ago

1.0.0-dev.21 (2024-02-02)

Features

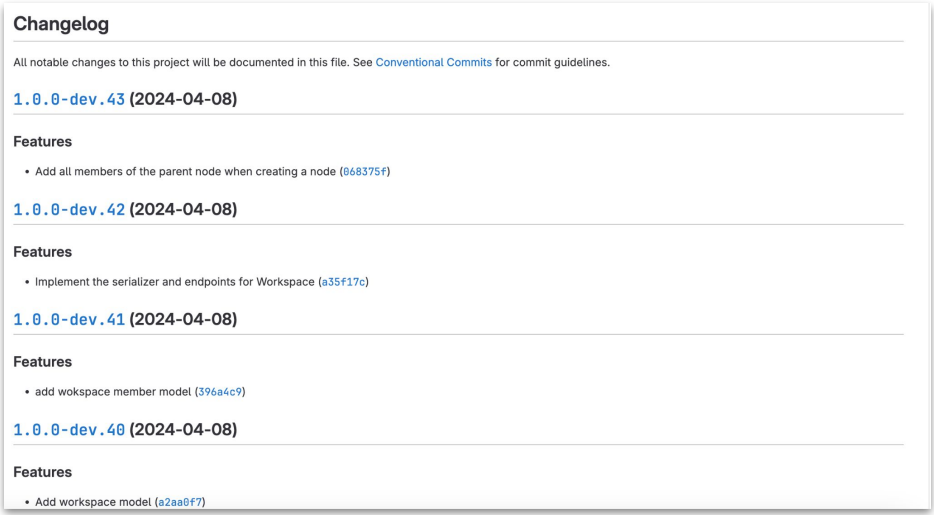
- add a slug property to node types and create NodeTypeCollection model (1f687f2)
- create a history field to keep track of the last_edited_by and last_edited_time in Node objects (1833a27)
- create fixtures for Node type (bcc1942)
- create node and node type models (774c6d1)
- create node type serializer and endpoints (dda3946)
- create the node serializers and endpoints (392384e)

Bug Fixes

- bypass python 3.11 warning concerning system package break when using pip outside of venv (c69d59f)

00be1885 v1.0.0-dev.21 Released 2 months ago by

Changelog file



Changelog

All notable changes to this project will be documented in this file. See Conventional Commits for commit guidelines.

1.0.0-dev.43 (2024-04-08)

Features

- Add all members of the parent node when creating a node (868375f)

1.0.0-dev.42 (2024-04-08)

Features

- Implement the serializer and endpoints for Workspace (a35f17c)

1.0.0-dev.41 (2024-04-08)

Features

- add wokspace member model (396a4c9)

1.0.0-dev.40 (2024-04-08)

Features

- Add workspace model (a2aa0f7)

Thank you for your attention !

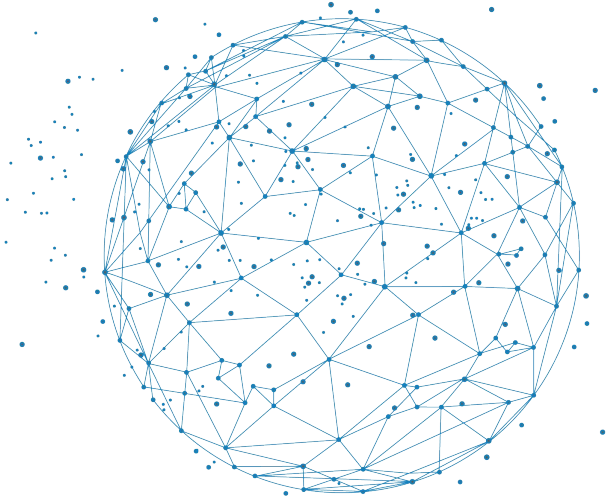
Réseau MERIT MetiER en bioinformaTique

IFB INSTITUT FRANÇAIS DE BIOINFORMATIQUE

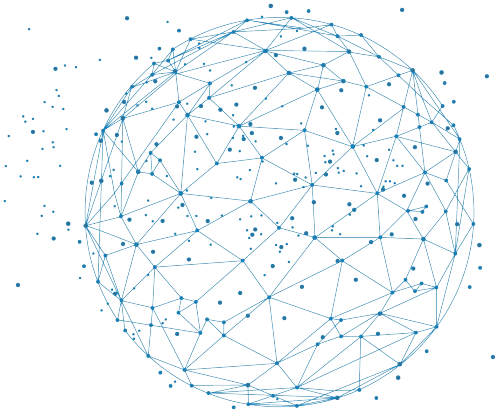
cnrs

INRAE

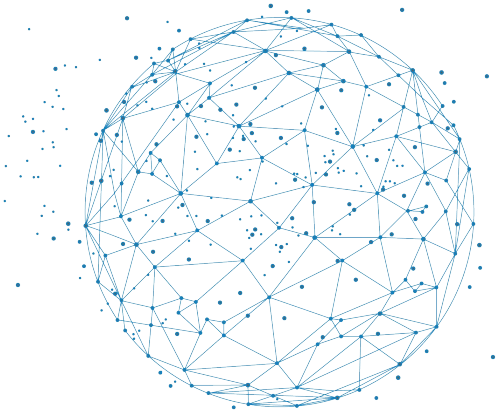
Practice session



Additional slides



Releases, on GitHub

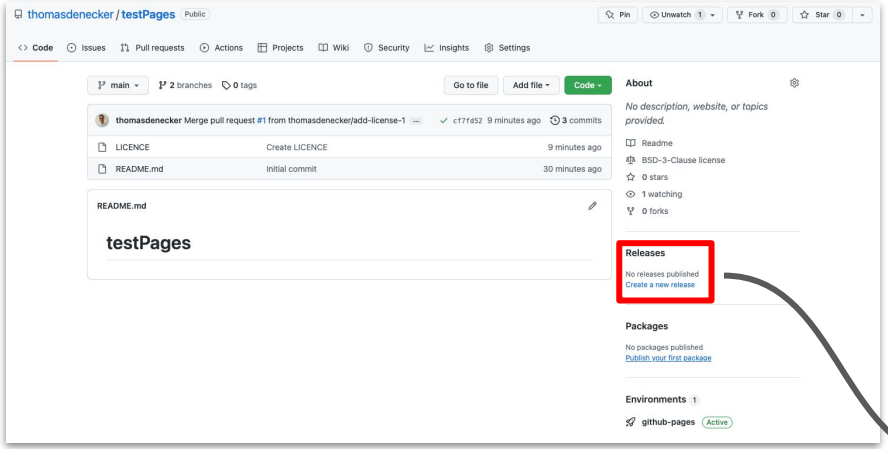


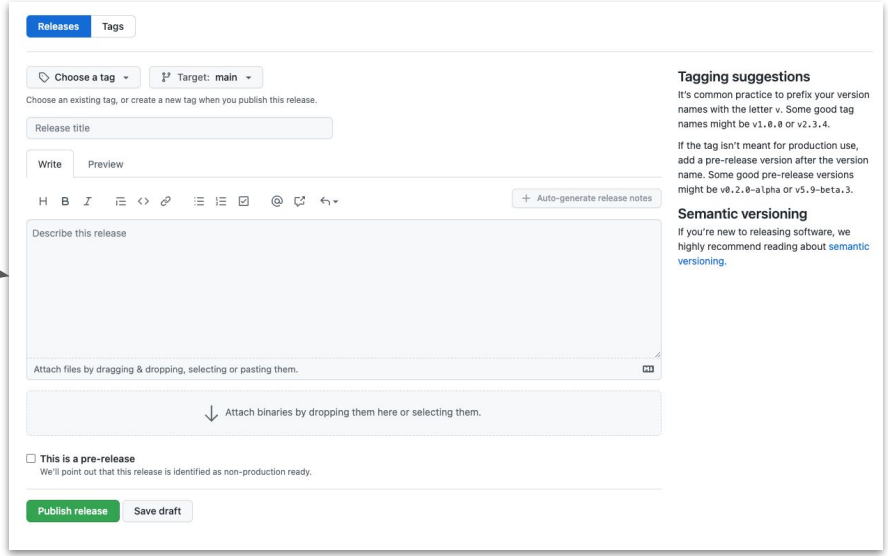
Goal

provide users with a version of your code that has been fixed in time and labelled. All the steps are detailed here:

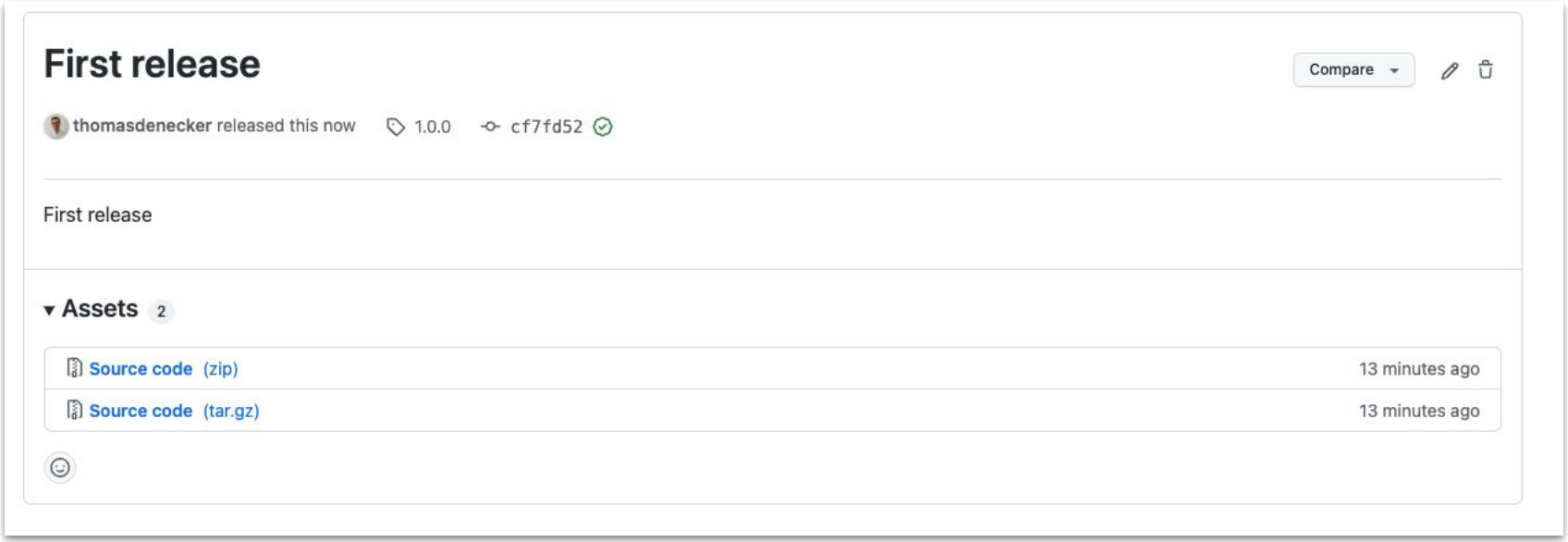
https://help.github.com/en/articles/creating-releases

Make a release





First release for test page



How to set it up in GitHub actions:

1. Use a GitHub Actions workflow file (e.g., `.github/workflows/ci.yml`)

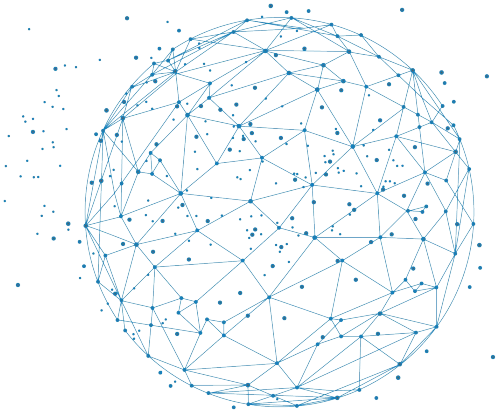
2. The GITHUB_TOKEN is automatically provided by GitHub Actions, so no manual token setup is required

3. Use the @semantic-release/github plugin to publish GitHub releases

```yaml
name: CI/CD Pipeline

on:
  push:
    branches:
      - main
    tags:
      - 'v*'
  pull_request:
    branches:
      - main

jobs:
  release:
    name: Semantic Release
    runs-on: ubuntu-latest
    needs: docker-build
    if: github.ref == 'refs/heads/main' && github.event_name == 'push'
    permissions:
      contents: write
      issues: write
      pull-requests: write
    steps:
      - name: Checkout code
        uses: actions/checkout@v4
        with:
          fetch-depth: 0
          token: ${{ secrets.GITHUB_TOKEN }}

      - name: Setup Node.js
        uses: actions/setup-node@v4
        with:
          node-version: '20'
          cache: 'npm'

      - name: Install dependencies
        run: npm ci

      - name: Run semantic-release
        env:
          GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
        run: npx semantic-release --extends ./.releaserc.github.json
```

Example of .github/workflows/ci.yml file

Obtain a DOI



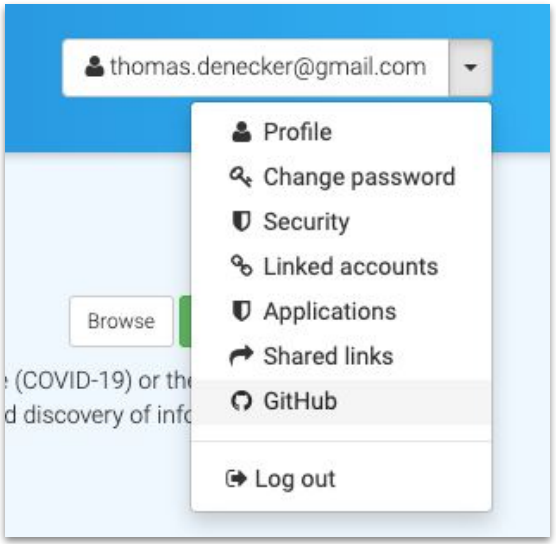
Digital Object Identifier

Reference system to cite an object (A GitHub project in our case)

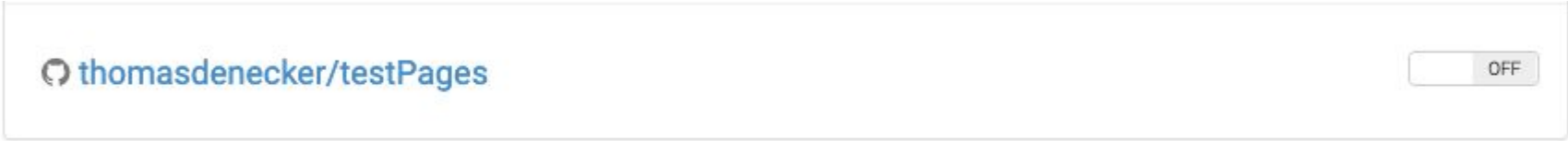
https://docs.github.com/en/repositories/archiving-a-github-repository/referencing-and-citing-content

Obtain a DOI

1. Sign in to Zenodo
   - With your GitHub account
   - With your ORCID account (add a ”Linked account” to GitHub afterwards)
2. Go to the Settings page ! GitHub tab

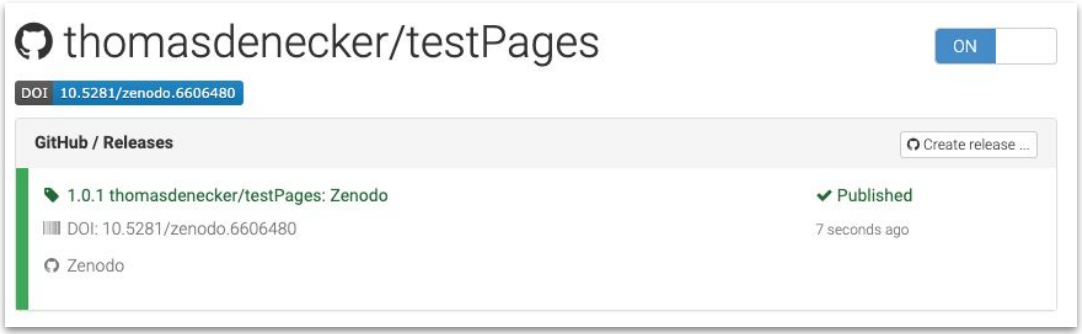


3. In the list below, find the project you want to link to Zenodo. Flip the switch.



4. Create a new release

5. Et voilà !



You can add the doi badge in your README

DOI Badge

This badge points to the latest released version of your repository. If you want a DOI badge for a specific release, please follow the DOI link for one of the specific releases and grab badge from the archived record.

Markdown

```
[![DOI](https://zenodo.org/badge/499069909.svg)](https://zenodo.org/bad
```

reStructedText

```
.. image:: https://zenodo.org/badge/499069909.svg
   :target: https://zenodo.org/badge/latestdoi/499069909
```

HTML

```
<a href="https://zenodo.org/badge/latestdoi/499069909"><img src="https:
```

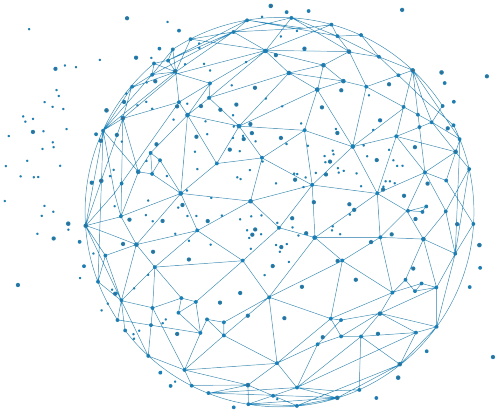
Image URL

```
https://zenodo.org/badge/499069909.svg
```

Target URL

```
https://zenodo.org/badge/latestdoi/499069909
```

Citations

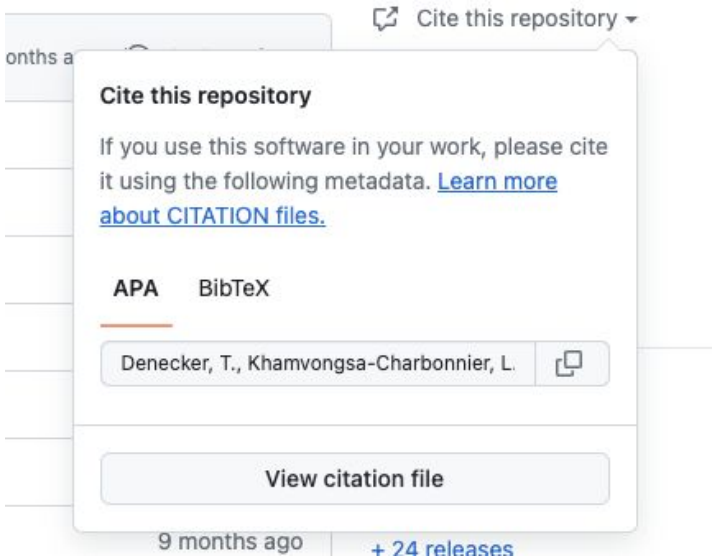


CITATION files

You can add a CITATION file to your repository to help users correctly cite your software.

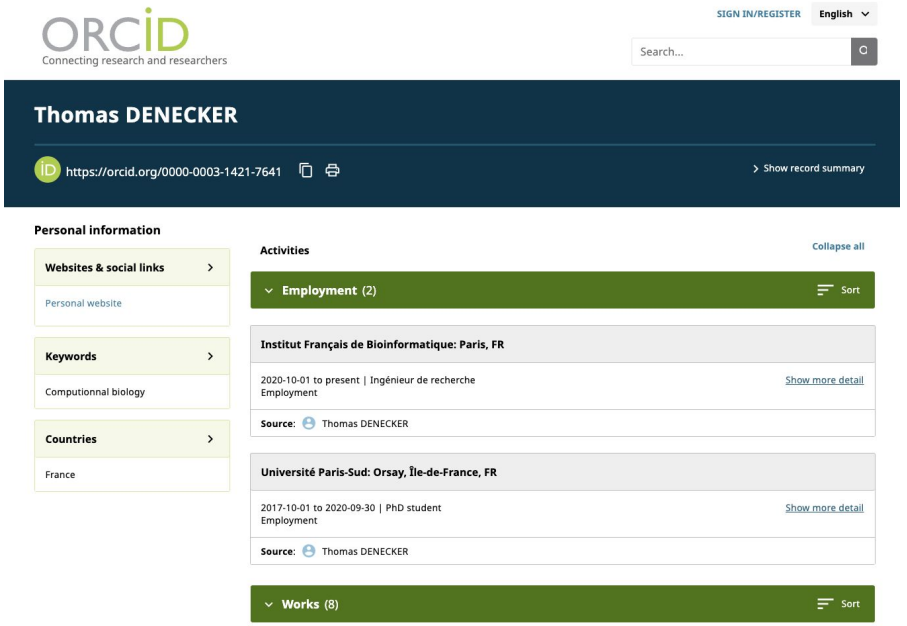
Example of CITATION.cff

```
cff-version: 1.2.0
message: "If you use this software, please cite it as below."
authors:
- family-names: "Lisa"
  given-names: "Mona"
  orcid: "https://orcid.org/0000-0000-0000-0000"
- family-names: "Bot"
  given-names: "Hew"
  orcid: "https://orcid.org/0000-0000-0000-0000"
title: "My Research Software"
version: 2.0.4
doi: 10.5281/zenodo.1234
date-released: 2017-12-18
  url: "https://github.com/github-linguist/linguist"
```

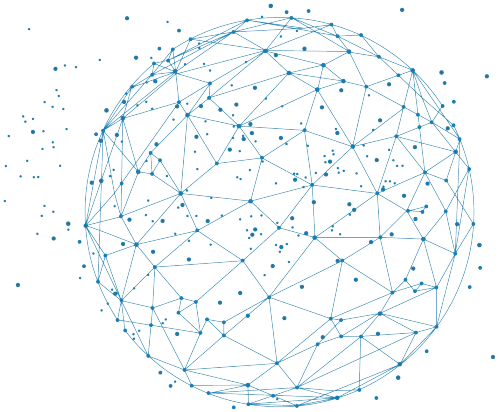


Focus on ORCID

ORCID (Open Researcher and Contributor ID) is a unique digital identifier assigned to a researcher or academic contributor. It aims to solve the problem of name ambiguity in research, where multiple authors may share the same name or similar names.



Archive your code



Software Heritage – The Universal Software Archive

Software Heritage is an international initiative to collect, preserve, and share all publicly available source code.

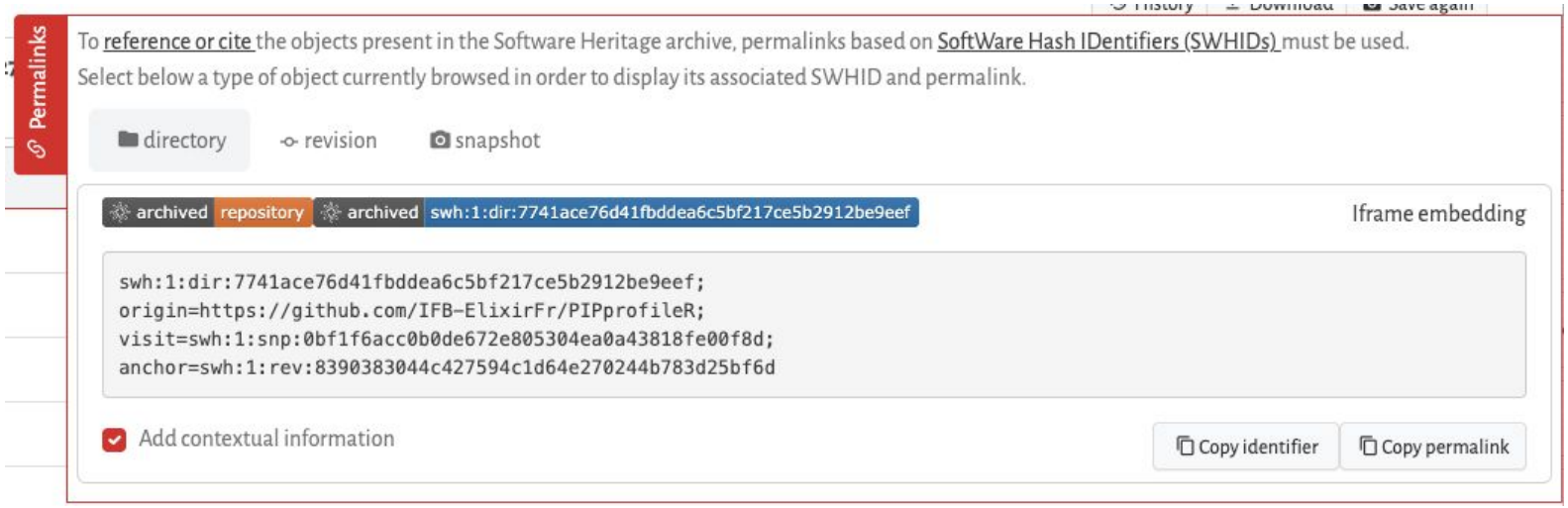
Its mission is to ensure that software, as a key part of our scientific and technological heritage, remains accessible over time.

By archiving millions of projects from platforms like GitHub, GitLab, and others, Software Heritage enables long-term reproducibility, transparency, and traceability in science and industry.

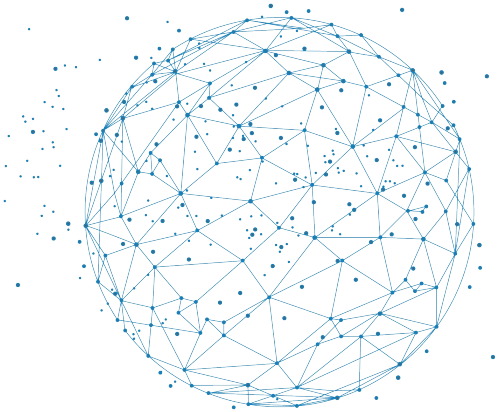
Learn more: softwareheritage.org

Bitbucket 2,861,844 origins	56,983 origins	git 33,515 origins
29,433 origins	debian 143,849 origins	91,965 origins
GitHub 278,117,567 origins	gitiles 24,738 origins	GitLab 5,839,728 origins
git 3,867 origins	Gogs 430 origins	2,065,035 origins
Guix 63,133 origins	GNU 354 origins	heptapod 1,363 origins
launchpad 657,631 origins	Maven 425,519 origins	NixOS 34,404 origins
npm 4,008,896 origins	4,988 origins	Packagist 380,879 origins
fedora PAGURE 72,459 origins	Phabricator 199 origins	pub.dev 65,249 origins
python 639,635 origins	SOURCEFORGE 382,323 origins	stagit 344 origins

Obtain a permanent Identifiers (SWHIDs)



Go further with Git*…

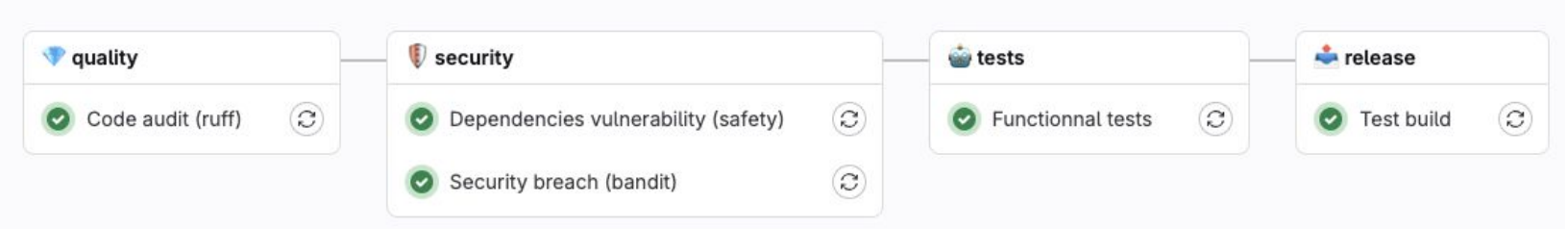


CI / CD - Continuous Integration & Continuous Deployment

CI/CD is a development practice that automates code integration, testing, and deployment.

- Continuous Integration (CI): Automatically tests and integrates code changes to detect issues early.
- Continuous Deployment (CD): Automatically deploys tested code to production or staging environments.

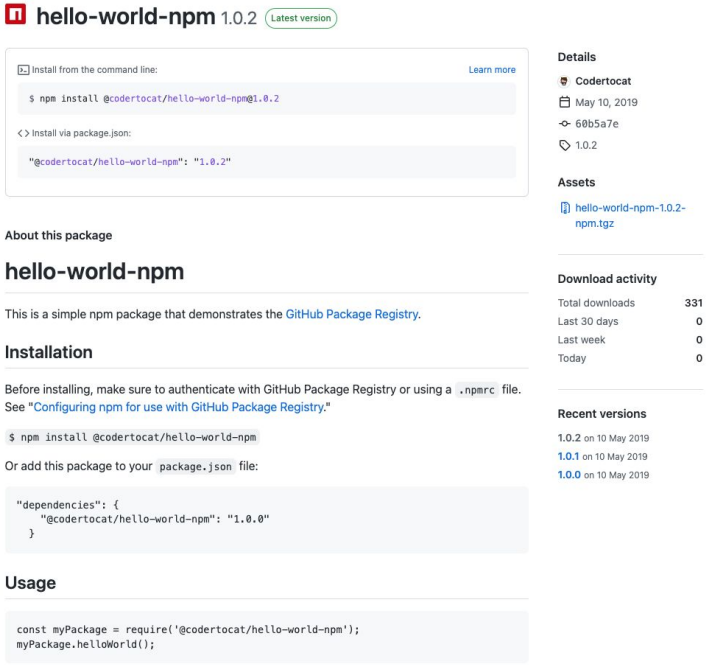
Both GitHub Actions and GitLab CI/CD provide built-in tools to set up pipelines, run tests, and deploy applications—helping teams release faster, more reliably, and with less manual work.

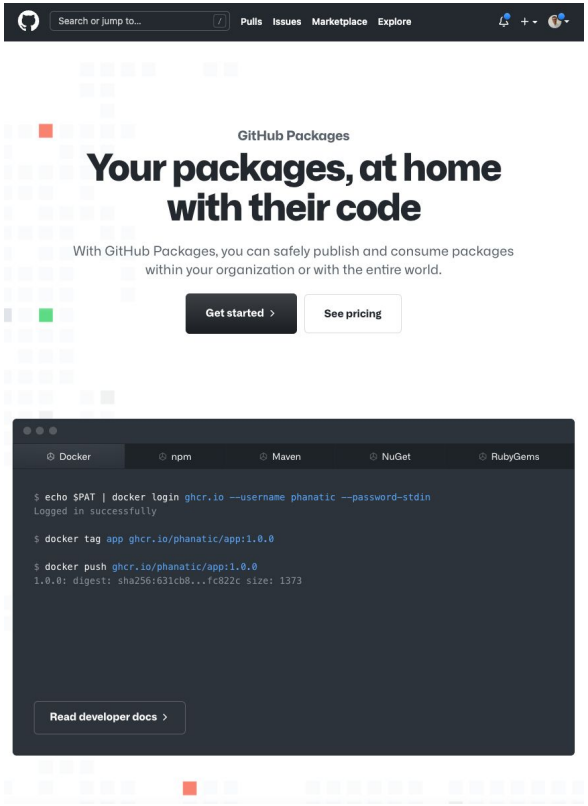


Docker image & package hosting

GitHub GitLab

https://docs.github.com/en/packages





Examples

Working with a GitHub Packages registry

Learn how to use a supported GitHub Packages registry.

Working with the Container registry
You can store and manage Docker and OCI images in the Container registry.

Working with the Docker registry
The Docker registry has now been replaced by the Container registry.

Working with the RubyGems registry
You can configure RubyGems to publish a package to GitHub Packages and to use packages stored on GitHub Packages as dependencies in a Ruby project with Bundler.

Working with the npm registry
You can configure npm to publish packages to GitHub Packages and to use packages stored on GitHub Packages as dependencies in an npm project.

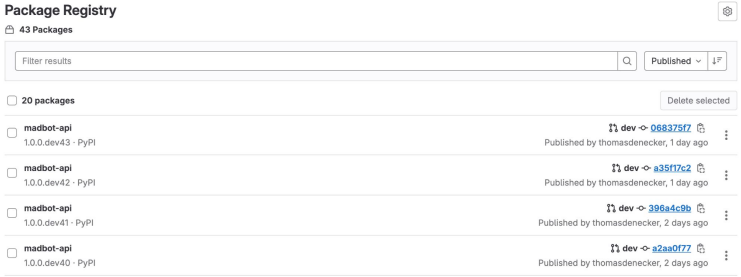
Working with the Apache Maven registry
You can configure Apache Maven to publish packages to GitHub Packages and to use packages stored on GitHub Packages as dependencies in a Java project.

Working with the Gradle registry
You can configure Gradle to publish packages to the GitHub Packages Gradle registry and to use packages stored on GitHub Packages as dependencies in a Java project.

Working with the NuGet registry
You can configure the `dotnet` command-line interface (CLI) to publish NuGet packages to GitHub Packages and to use packages stored on GitHub Packages as dependencies in a .NET project.

Migrating to the Container registry from the Docker registry
GitHub will migrate Docker images previously stored in the Docker registry on GitHub to the Container registry.

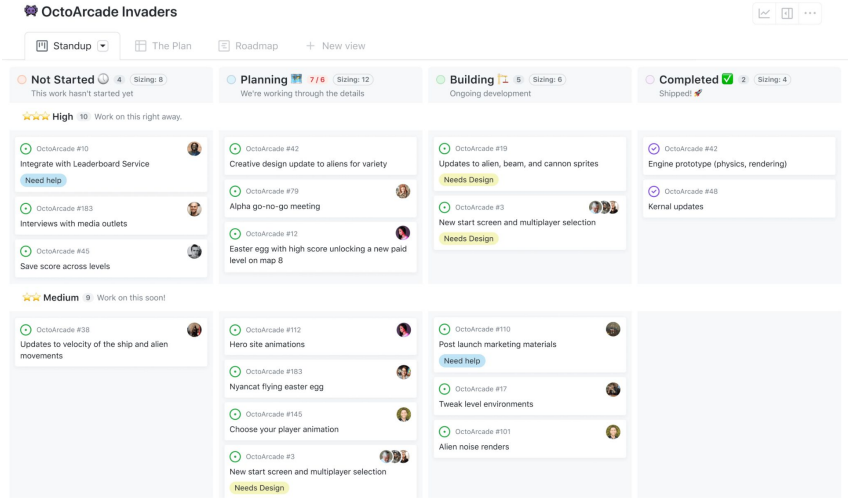
Packages (ie: python)

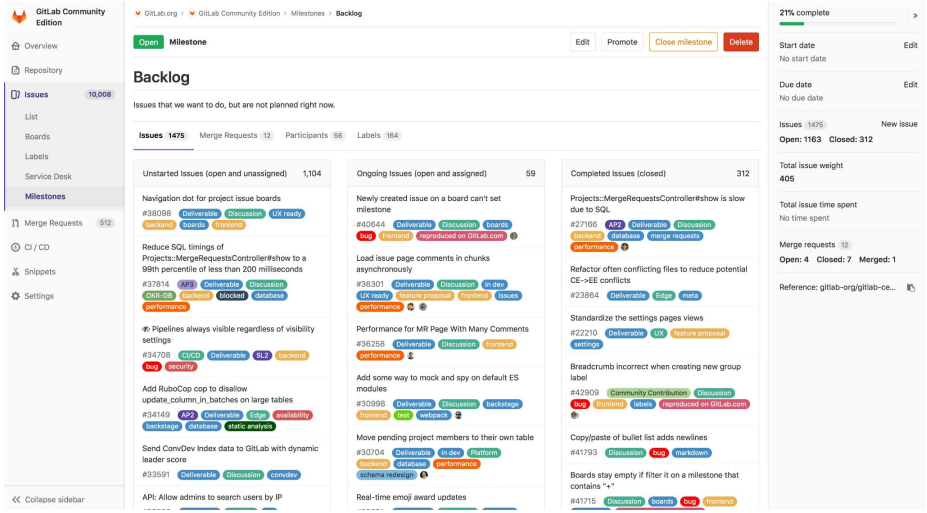


Container (ie: docker)

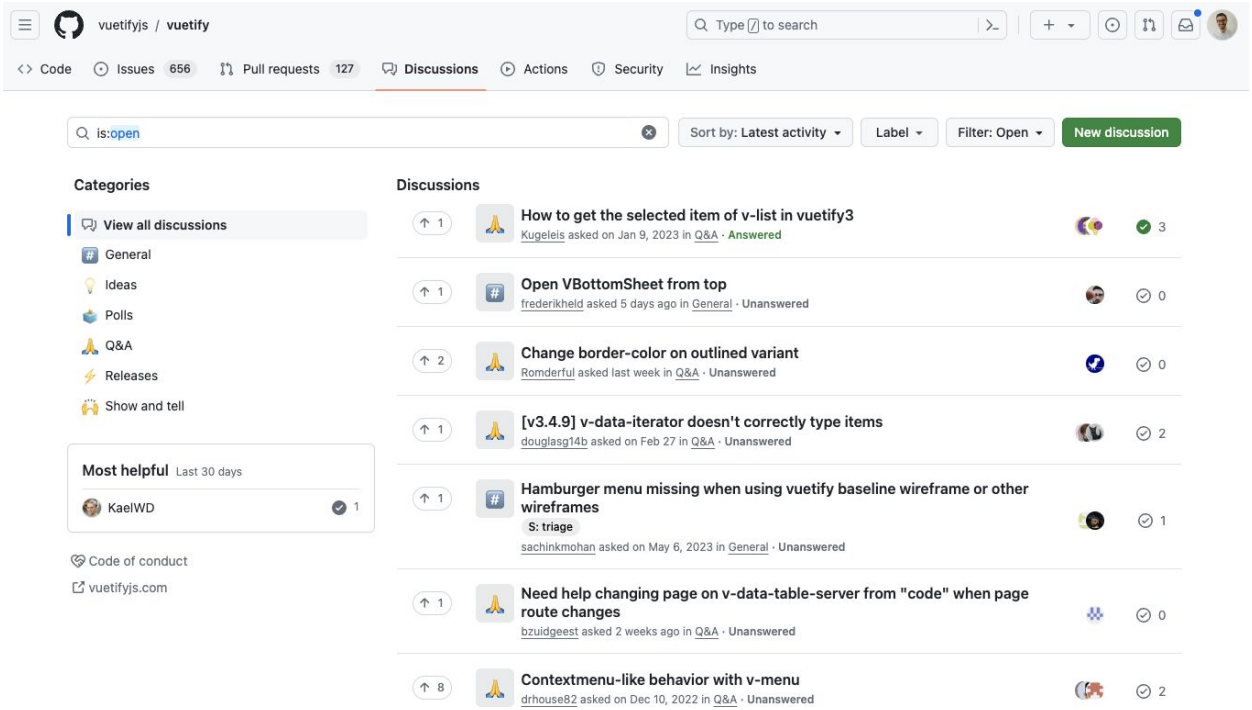
Project management

GitHub GitLab

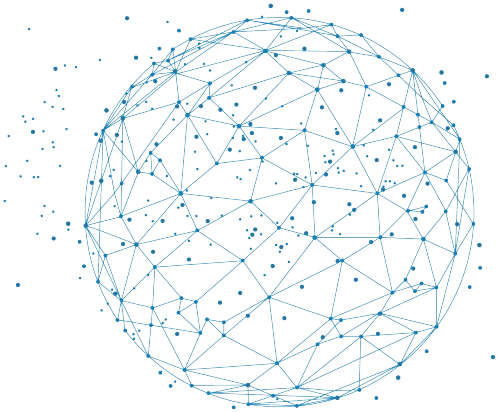




Github-specific : Discussion



Showcase your work



Why ?

- Your project is simpler to share and find

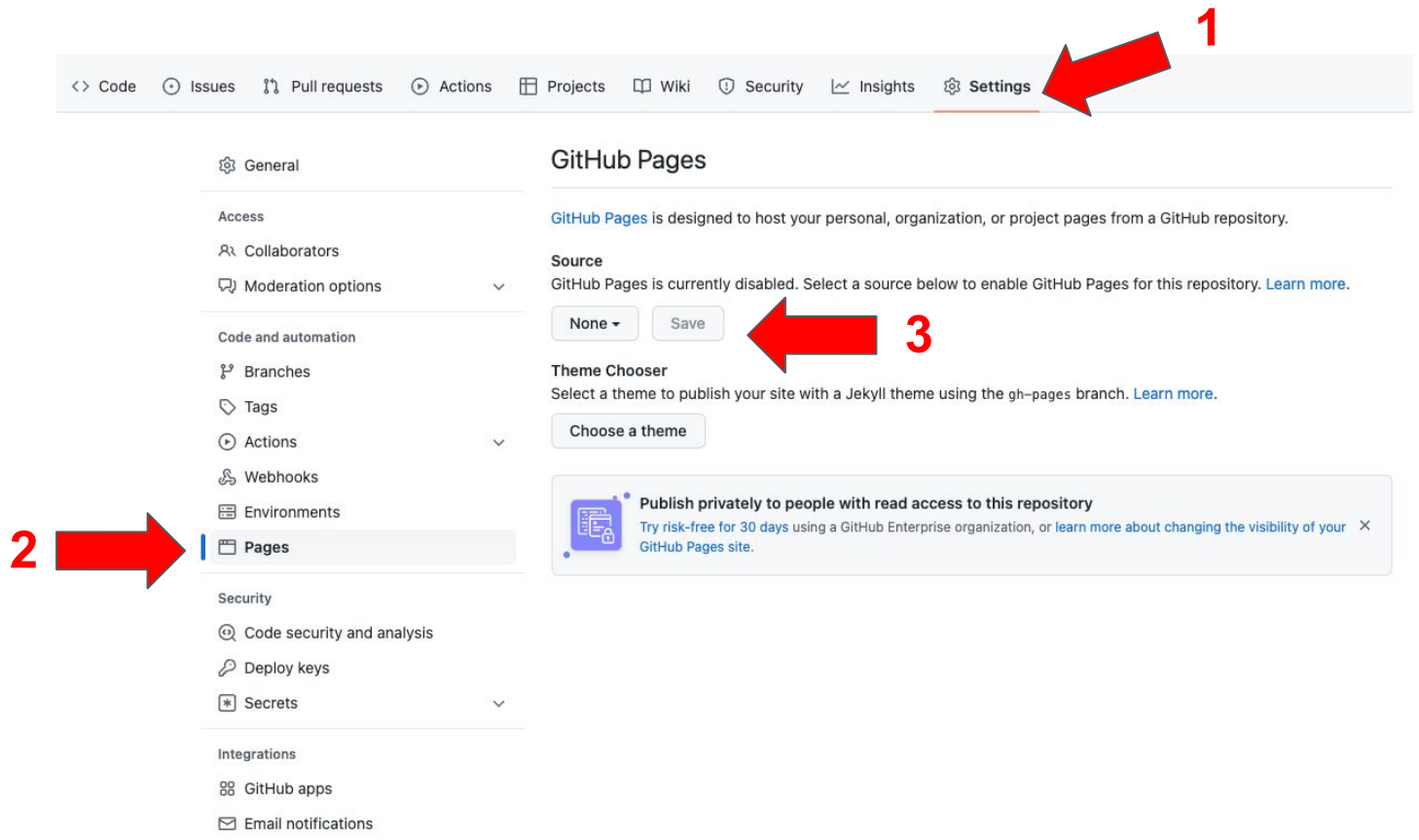
Advantages

- Free hosting of static websites
- Able to convert Markdown into a website

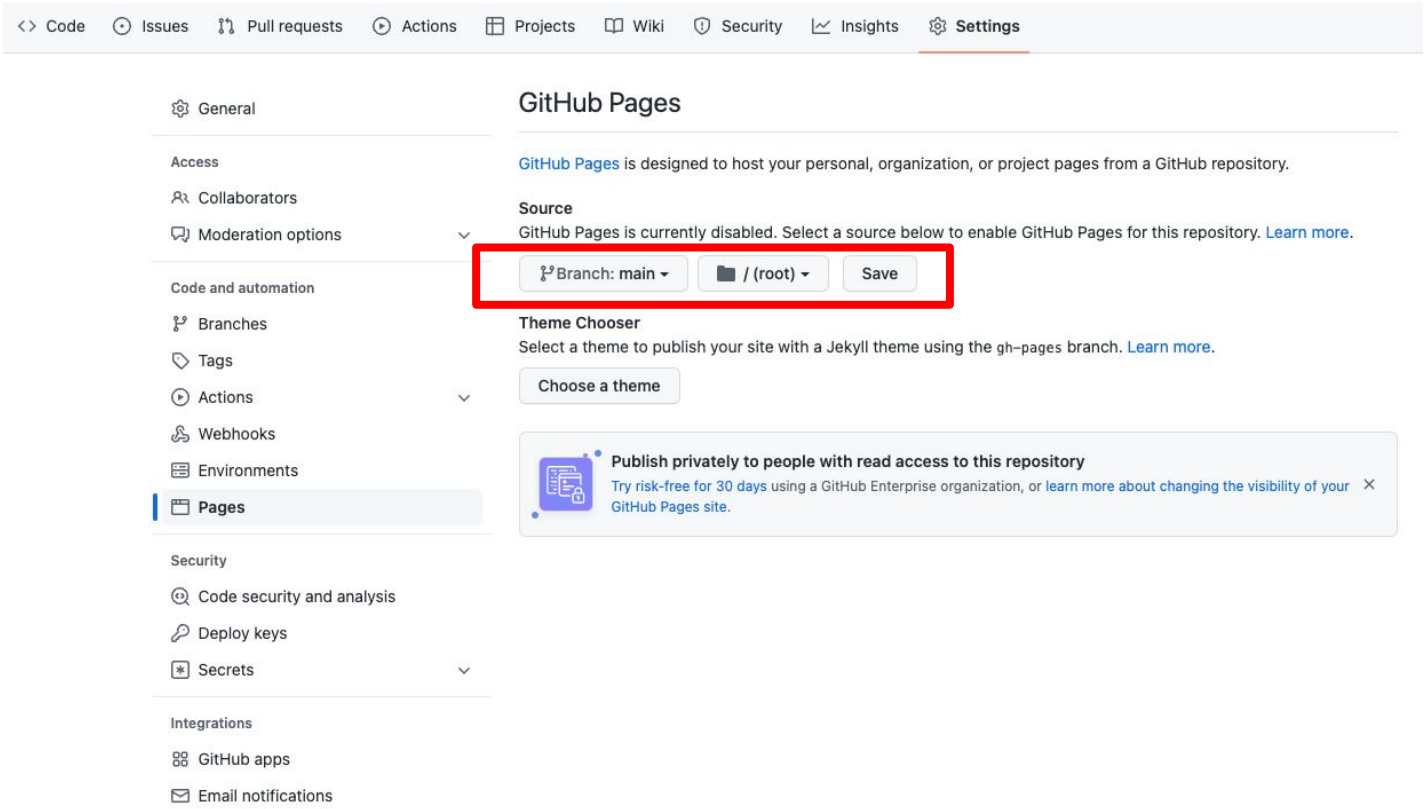
Documentations :

- https://pages.github.com/
- https://docs.gitlab.com/user/project/pages/

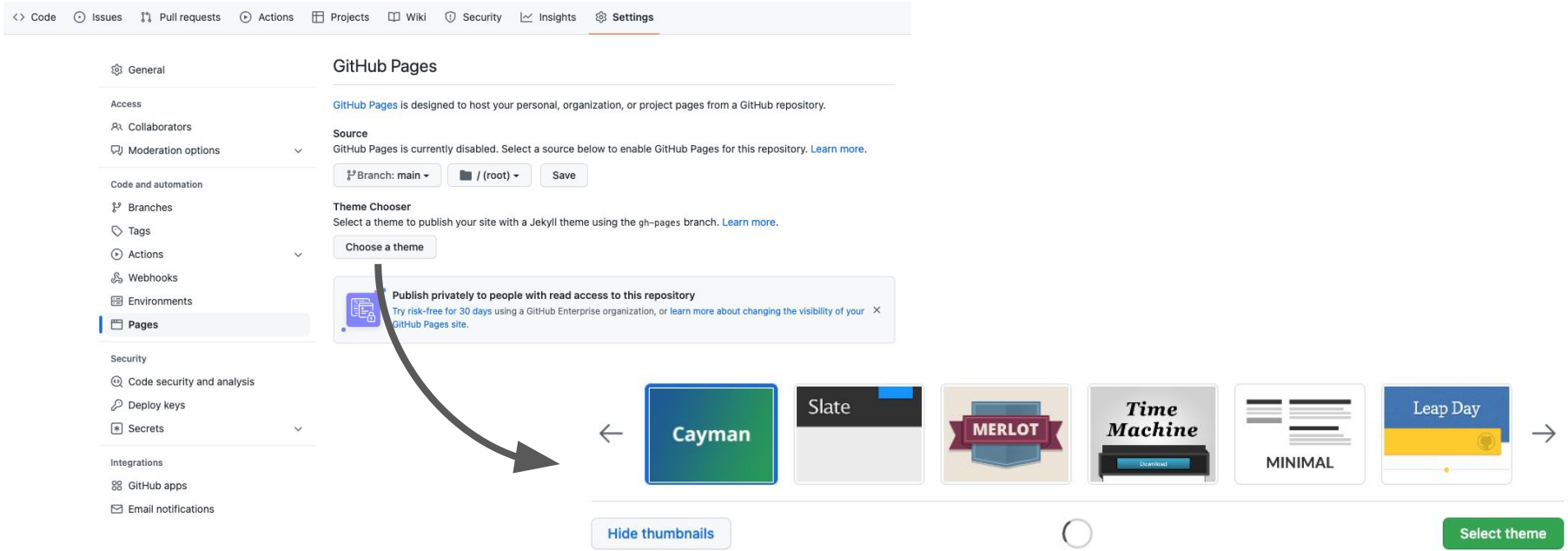
Showcase your work - In practice



Showcase your work - In practice

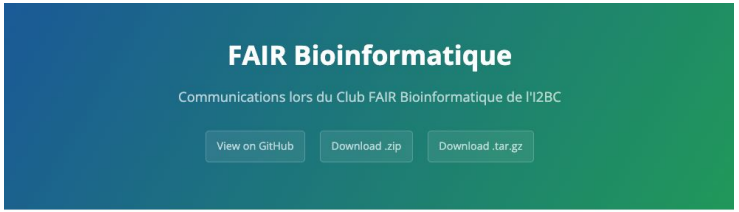


Showcase your work - In practice



Convert Markdown into HTML !

```
# Welcome !

**FAIR_Bioinfo** - [![DOI](https://zenodo.org/badge/164655551.svg)](https://zenodo.org/badge/latestdoi/164655551)

**Gitbook** - [![](https://img.shields.io/badge/Gitbook-FAIR_Bioinfo-blue.svg)](https://fair-bioinfo.gitbook.io/fair-bioinfo/)
[![](https://img.shields.io/badge/Github-FAIR_Bioinfo_Gitbook-blue.svg)](https://github.com/thomasdenecker/FAIR_Bioinfo-
Gitbook)
[![DOI](https://zenodo.org/badge/197582632.svg)](https://zenodo.org/badge/latestdoi/197582632)

**Docker image** - [![](https://img.shields.io/badge/Docker-FAIR_Bioinfo-blue.svg)]
(https://hub.docker.com/r/tdenecker/fair_bioinfo) [![]
(https://images.microbadger.com/badges/image/tdenecker/fair_bioinfo.svg)]
(https://microbadger.com/images/tdenecker/fair_bioinfo)

**Licence pour le code** - [![](https://img.shields.io/badge/LICENCE-CeCILL%202.1-brightgreen.svg)]
(https://github.com/thomasdenecker/FAIR_Bioinfo/blob/master/LICENCE)

**Licence pour le GitBook** -
[![CC BY-SA 4.0][cc-by-sa-image]][cc-by-sa]

[cc-by-sa]: http://creativecommons.org/licenses/by-sa/4.0/
[cc-by-sa-image]: https://licensebuttons.net/l/by-sa/4.0/88x31.png
[cc-by-sa-shield]: https://img.shields.io/badge/License-CC%20BY--SA%204.0-lightgrey.svg

**Bienvenue à FAIR_bioinfo**

FAIR_Bioinfo est une formation initialement pensée pour la communauté francophone. En effet, les ressources sont nombreuses
concernant la reproductibilité en anglais mais un manque se faisait sentir en français. Vous trouverez tout le contenu du
cours présenté dans les différentes sessions (slides en français). Nous proposons aussi la version retranscrite de ces cours
au format gitbook en anglais : https://fair-bioinfo.gitbook.io/fair-bioinfo/

*FAIR_Bioinfo is a training course initially designed for the French-speaking community. Indeed, there are many resources
concerning reproducibility in English but there was a lack in French. You will find all the course content presented in the
different sessions (slides in French). We also offer the transcribed version of these courses in gitbook format in English:
https://fair-bioinfo.gitbook.io/fair-bioinfo/*

**Informations pratiques**
- Quand ? : le dernier vendredi après midi de chaque mois (sauf juillet à définir), rdv 12h30
- Durée ? : 1h30 (questions incluses)
- Lieu ? : Salle de conférence A.Kalogeropoulos, b. 400, campus Orsay

**Objectifs**

L'objectif est de proposer et d'utiliser un panel d'outils permettant la réalisation d'un projet complet de bio-info en
partant de rien et aboutissant à la création d'un conteneur (technologie Docker). Le partage, la valorisation et l'analyse
dynamique des données seront inclus dans le panel.
FAIR correspond à l'acronyme anglais "Findable, Accessible, Interoperable, & Reusable", initialement défini pour les données
mais que nous détournons ici pour leurs protocoles d'analyse.
Le projet support est une étude "d'expression différentielle de gènes" à partir de données RNAseq d'O.tauri.
```



Welcome !

FAIR_Bioinfo - DOI 10.5281/zenodo.5105349

Gitbook - Gitbook FAIR Bioinfo | Github FAIR Bioinfo Gitbook | DOI 10.5281/zenodo.3624622

Docker image - Docker FAIR Bioinfo

Licence pour le code - LICENCE CeCILL 2.1

Licence pour le GitBook -

Bienvenue à FAIR_bioinfo

FAIR_Bioinfo est une formation initialement pensée pour la communauté francophone. En effet, les ressources sont nombreuses concernant la reproductibilité en anglais mais un manque se faisait sentir en français. Vous trouverez tout le contenu du cours présenté dans les différentes sessions (slides en français). Nous proposons aussi la version retranscrite de ces cours au format gitbook en anglais : https://fair-bioinfo.gitbook.io/fair-bioinfo/

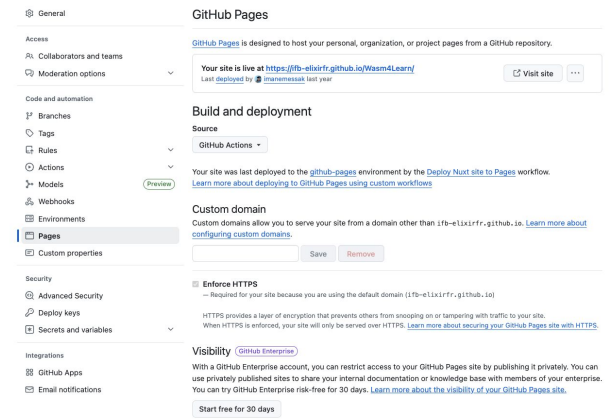
FAIR_Bioinfo is a training course initially designed for the French-speaking community. Indeed, there are many resources concerning reproducibility in English but there was a lack in French. You will find all the course content presented in the different sessions (slides in French). We also offer the transcribed version of these courses in gitbook format in English: https://fair-bioinfo.gitbook.io/fair-bioinfo/

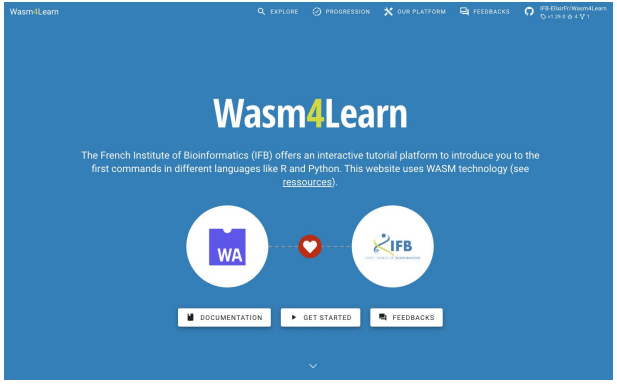
Informations pratiques

- Quand ? : le dernier vendredi après midi de chaque mois (sauf juillet à définir), rdv 12h30
- Durée ? : 1h30 (questions incluses)
- Lieu ? : Salle de conférence A.Kalogeropoulos, b. 400, campus Orsay

Also works directly from HTML

1. Create a folder named ”docs” main file must be named index.html
2. ”Settings” → ”Options” → ”GitHub Pages”





Example :

https://ifb-elixirfr.github.io/Wasm4Learn/