

GIT

Guida Introduttiva

A cura di :

Salvatore Diprima

Luca Aloï

13/03/2018



Argomenti trattati

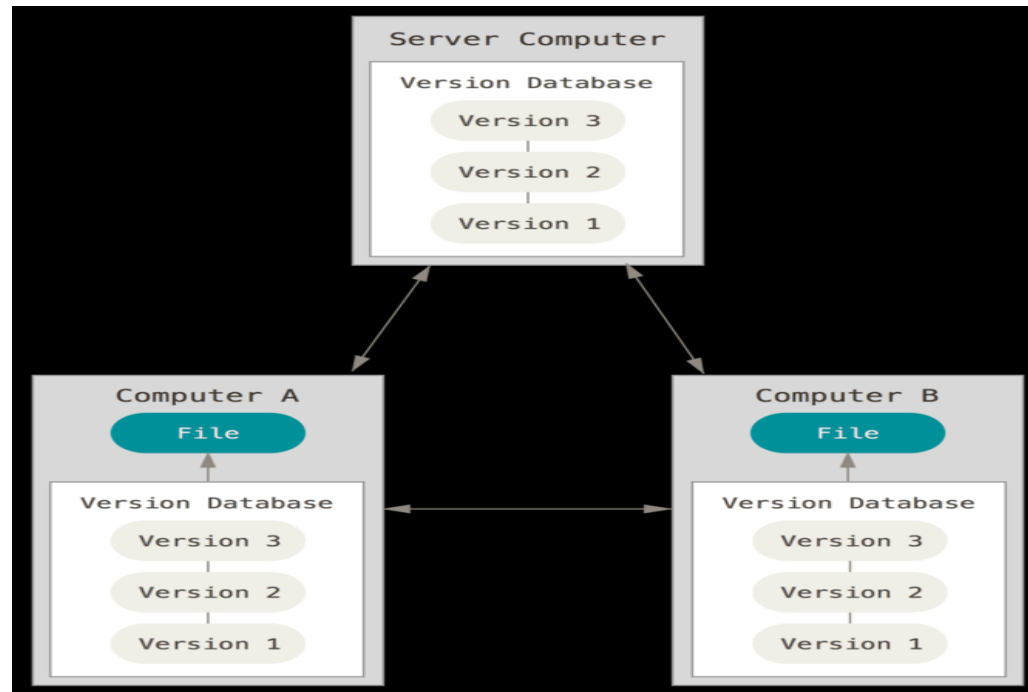
- I sistemi di controllo di versione
- Git: Storia e Caratteristiche principali
- Git: Repository, Divergenze, Fusioni
- Git: Server remoti e GitHub

Il Controllo di Versione(VCS)

- Il controllo di versione è un sistema che tiene traccia, nel tempo, di tutte le modifiche a un file o un insieme di file, così che sia possibile recuperare una qualsiasi versione precedente in qualsiasi momento.
- Sistemi locali di controllo di versione
- Sistemi centralizzati di controllo di versione(CVCS)
- Sistemi distribuiti di controllo di versione(DVCS)

Sistemi distribuiti di controllo di versione(DVCS)

- Permette di tenere traccia delle modifiche e delle versioni apportate al codice sorgente del software, senza la necessità di dover utilizzare un server centrale.



Sistemi distribuiti di controllo di versione(DVCS)

- **Bazaar**

Software libero per il controllo versione distribuito, ideato da Canonical Ltd. È scritto in Python e fa parte del progetto GNU.

- **BitKeeper**

Software di controllo di versione distribuito per il codice sorgente dei programmi, prodotto da BitMover Inc.

Originariamente era un software proprietario, ma dal 9 maggio 2016 è diventato open source come i suoi concorrenti Git, Bazaar e Mercurial.

- **Mercurial**

Software multiplatforma di controllo di versione distribuito creato da Matt Mackall e disponibile sotto GNU General Public License 2.0.

È quasi completamente scritto in Python, ma include anche una implementazione diff binaria scritta in C.

Git-Storia

GIT

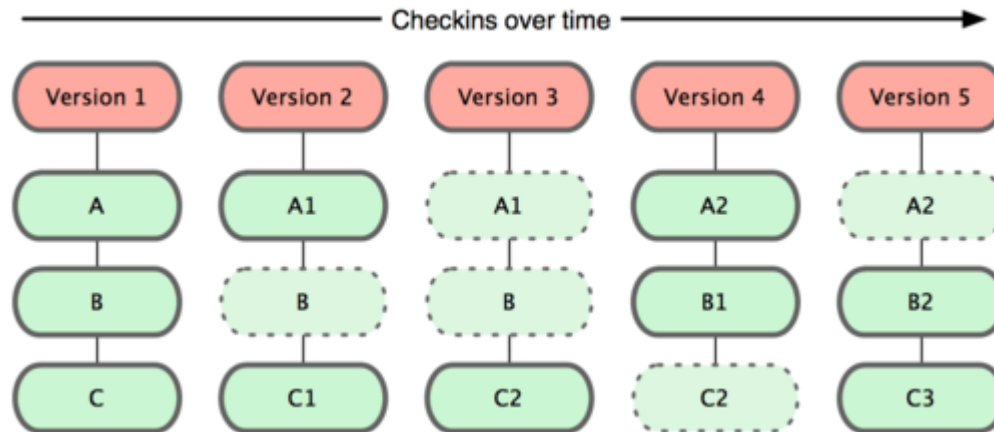
- Git è un software di controllo versione distribuito utilizzabile da interfaccia a riga di comando, creato da Linus Torvalds nel 2005.
- Git (che nello slang americano significa «idiota») nacque per essere un semplice strumento per facilitare lo sviluppo del kernel Linux ed è diventato uno degli strumenti di controllo versione più diffusi.

Git-Caratteristiche

- Veloce
- Design semplice
- Ottimo supporto allo sviluppo non-lineare (migliaia di rami paralleli)
- Completamente distribuito
- Capace di gestire in modo efficiente (per velocità e dimensione dei dati) grandi progetti come il kernel Linux

Git-Caratteristiche

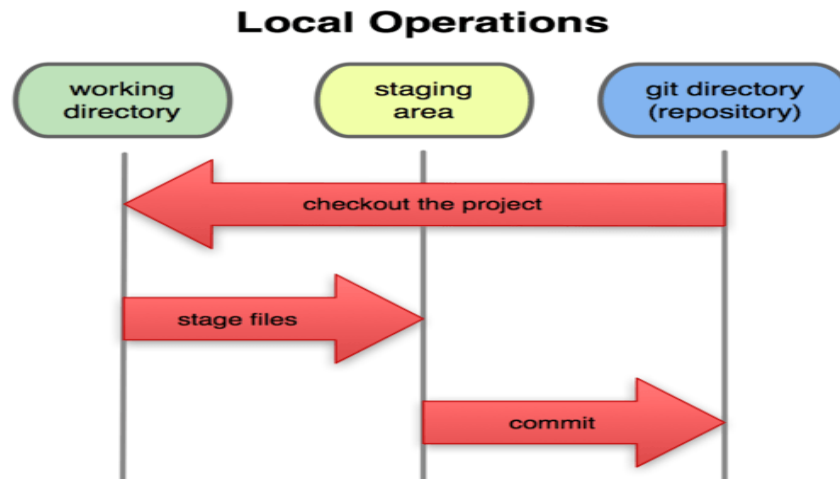
- Git considera i propri dati come una serie di istantanee (snapshot) di un mini filesystem. Ogni volta che committi, o salvi lo stato del tuo progetto in Git, lui fa un'immagine di tutti i file in quel momento, salvando un riferimento allo snapshot. Se alcuni file non sono cambiati, Git non li risalva, ma crea semplicemente un collegamento al file precedente già salvato.



Git-Caratteristiche

I Tre Stati

- I file in Git possono essere in tre stati: committed (committati), modified (modificati) e staged (in stage o Index). Committato significa che il file è al sicuro nel database locale. Modificato significa che il file è stato modificato, ma non è ancora stato committato nel database. In stage significa che hai contrassegnato un file, modificato nella versione corrente, perché venga inserito nello snapshot alla prossima commit.



Git-Configurazioni

//Configura account

- `$ git config --global user.name "John Doe"`
- `$ git config --global user.email johndoe@example.com`

//Configura editor

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

//Configura diff

- `$ git config --global merge.tool vimdiff`

//Visualizza configurazioni

- `$ git config --list`

Git-Help

//Elenco dei comandi principali

- \$ git

//Elenco di tutti i comandi

- \$ git help -a

//Dettagli sul comando scelto

- \$ git help nome comando

Git-Repository

Creare un repository in una directory preesistente

//Crea sottodirectory .git con la struttura del repository

- `$ git init`

Clonare un repository esistente

//clonare il repository in una directory con nome diverso da grit

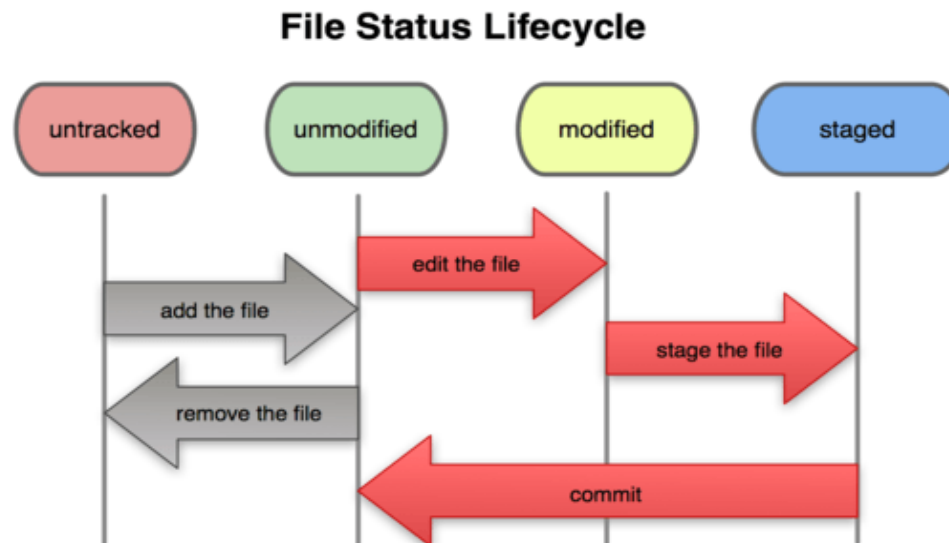
- `$ git clone https://github.com/schacon/grit.git mygrit`

Git-Repository

Ogni file della directory di lavoro può essere in uno dei due stati seguenti: *tracked*(tracciato) o *untracked* (non tracciato, fuori dall'ultimo snapshot).

//Verifica lo stato dei file

- `$ git status`



Git-Repository

//Traccia un nuovo file README

- `$ git add README`

//Committa quello che è in stage con commento

- `$ git commit -m 'initial project version'`

//Committa quello che è già tracciato(eseguendo in automatico il git add)

- `$ git commit -a -m 'added new benchmarks'`

//Rimuove il file gemspec

- `$ git rm gemspec`

//Rimuove il file readme solo dallo stage(lo mantiene sulla cartella di lavoro)

- `$ git rm --cached readme.txt`

Git-Repository

Ignorare File

Spesso hai dei file che non vuoi che Git aggiunga automaticamente e nemmeno che te li mostri come tracciati, come log o file di sistema. In questi casi puoi creare un file chiamato `.gitignore` con la lista di pattern dei file che vuoi ignorare.

//Ignora i file che finiscono in .o, .a o con ~

- `$ cat .gitignore`
- `*.[oa]`
- `*~`

Git-Repository

Mostra le modifiche dentro e fuori lo stage

Se vuoi sapere cos'è stato effettivamente modificato e non solo quali file(`git status`) puoi usare il comando `git diff`.

//Per vedere cosa hai modificato, ma non ancora nello stage

- `$ git diff`

//Per vedere cosa c'è nello stage e che farà parte della prossima commit

- `$ git diff --cached`

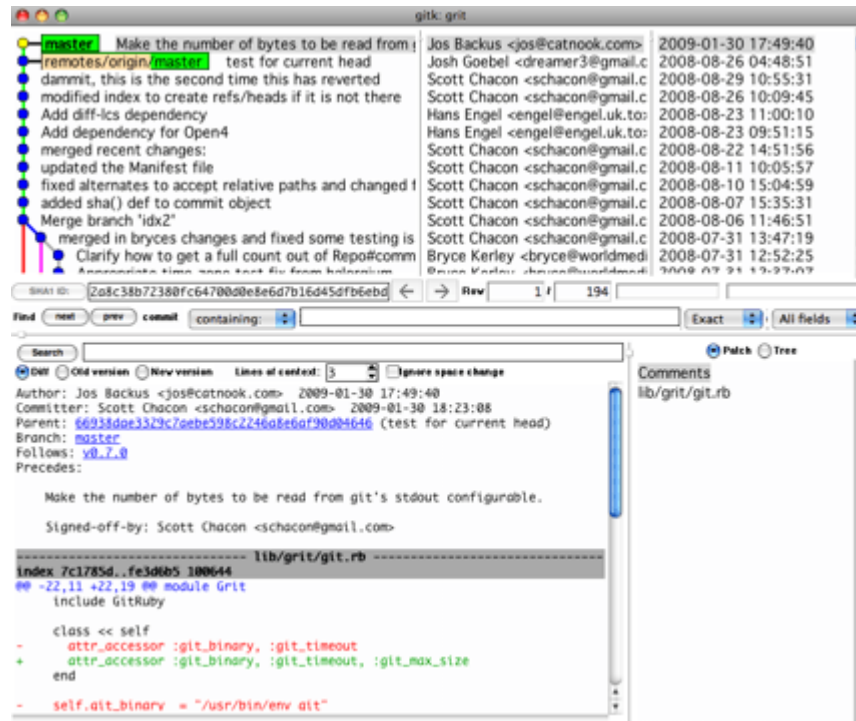
Git-Repository

//Vedere la cronologia delle commit

- `$ git log`

//Vedere la cronologia delle commit graficamente

- `$ gitk`



Git-Repository

Annullare qualcosa

//Modifica l'ultima commit

\$ git commit --amend

//Rimuove il file benchmarks.rb dallo stage

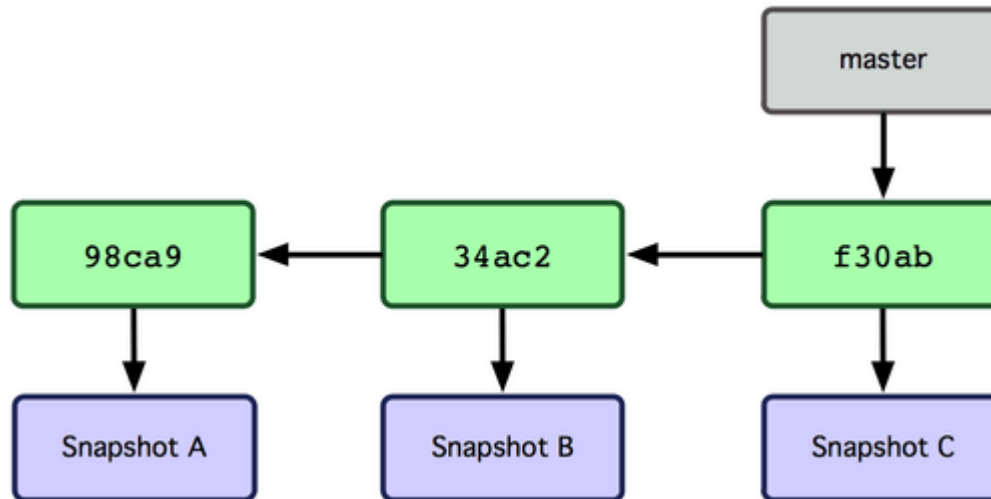
\$ git reset HEAD benchmarks.rb

//Annulla le modifiche al file committato

\$ git checkout -- benchmarks.rb

Git-Diramazioni

- Diramazione(Branch) significa divergere dal flusso principale di sviluppo continuando a lavorare senza correre il rischio senza pasticciare il flusso principale.
- Il nome del ramo principale in Git è master, che punta all'ultimo commit eseguito.

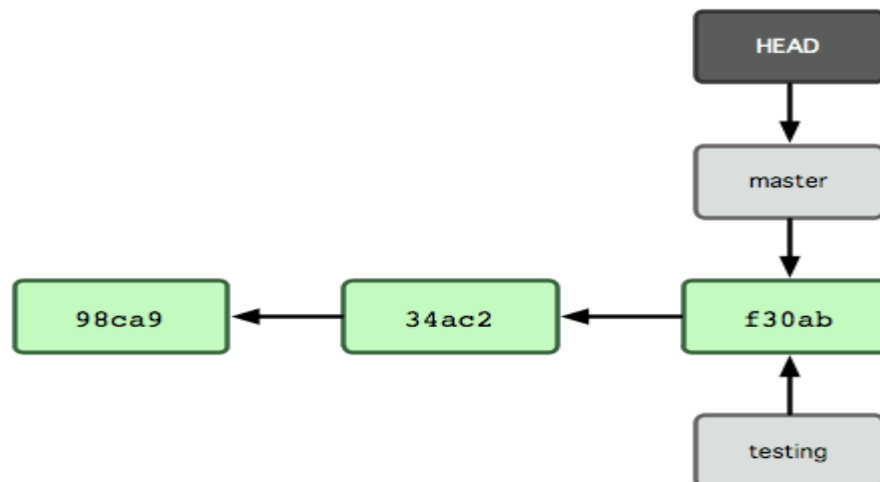


Git-Diramazioni

//Crea un nuovo ramo testing

- `$ git branch testing`

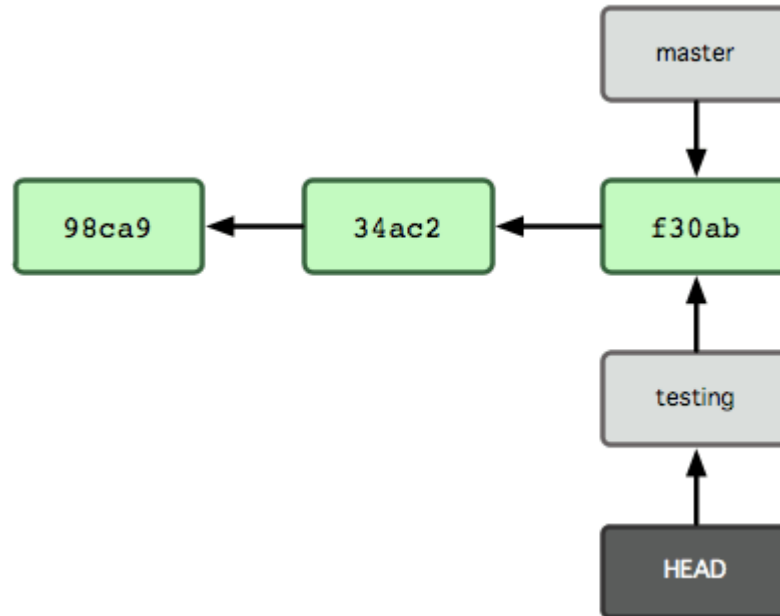
HEAD è un puntatore al ramo locale su cui ti trovi.



Git-Diramazioni

//Per spostarsi sul nuovo ramo testing

- \$ git checkout testing



Git-Fusioni

//Fondere il branch testing al ramo master

- \$ git checkout master
- \$ git merge testing

//Dopo il merge si può eliminare il ramo secondario

- \$ git branch -d testing

Git-Fusioni

Conflitti

Se modifichi la stessa parte di uno stesso file in modo differente nei due rami che stai fondendo assieme, Git non è in grado di unirli in modo pulito.

Git aggiunge dei marcatori standard di conflitto-risoluzione (<<<<<<, =====) ai file che hanno conflitti, così puoi aprirli manualmente e risolvere i conflitti.

Git-Server Remote

- Per poter collaborare con un qualsiasi progetto Git, devi sapere come amministrare i tuoi **repository remoti**.
- I **repository remoti** sono versioni dei progetti ospitate da qualche parte su Internet o sulla rete locale. Puoi averne molti e normalmente avrai un accesso in sola lettura o anche in scrittura.
- Collaborare con altri implica di sapere amministrare questi **repository remoti**, inviarne e prelevarne dati per condividere il lavoro.

Git-Server Remote

//Per vedere i server remoti che hai configurato

- `$ git remote -v`

//Aggiunge un nuovo repository remoto con un nome breve(pb)

`$ git remote add pb https://github.com/paulboone/ticgit.git`

//Per scaricare tutto dal repository remoto

- `$ git fetch pb`

//Per caricare tutto sul server

- `$ git push pb master`

Git-Server Remote

Differenza tra Pull, Fetch e Clone

Git Pull

Tira giù dal server quello che chiedi e fa la merge nel branch in cui ti trovi(fetch + merge)

Git Fetch

Simile al pull ma non fa il merge in automatico

Git Clone

Clona un repository su una cartella locale

Git-Hosting

- Puoi ospitare i tuoi progetti **Git** su un sito esterno e dedicato. Un sito di hosting è generalmente veloce da configurare ed è facile avviare un progetto su questo.
- Oggi, hai un'enorme quantità di opzioni di hosting tra cui scegliere, ognuna con differenti vantaggi e svantaggi. Per vedere una lista aggiornata, controlla la pagina **GitHosting** sul wiki principale di Git:
- **<https://git.wiki.kernel.org/index.php/GitHosting>**

GitHub

- **GitHub** è il più grande hosting Git di progetti open source ed è anche uno dei pochi che offre sia un *hosting pubblico* sia *privato*.
- Moltissime aziende che offrono servizi a livello internazionale usano Github, le principali sono Google, Apple, Microsoft, NASA, Facebook, Twitter, NodeJS, Ruby Rails, JetBrains, JQuery.
- GitHub, Inc. ha sede legale a San Francisco in California.



- Creare un nuovo repository

Search GitHub Pull requests Issues Marketplace Explore

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner Repository name

ilsalvador83 /

Great repository names are short and memorable. Need inspiration? How about `solid-octo-engine`.

Description (optional)

☒ **Public**
Anyone can see this repository. You choose who can commit.

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

☐ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** Add a license: **None** ⓘ

Create repository



Repository Public

Visibile da tutti. Decidi tu chi può effettuare commit.

Repository Private

Decidi tu chi può vedere e committare il repository (A pagamento).



LICENZA

I repository pubblici su GitHub sono usati per condividere software open source. Affinché il tuo repository sia veramente open source, devi licenziarlo in modo che altri siano liberi di usare, modificare e distribuire il software.

A screenshot of the GitHub 'Create repository' form. The form includes options for repository visibility (Public or Private), a checkbox to initialize with a README, and dropdown menus for adding a .gitignore file and a license. The 'Add a license' dropdown is open, showing a list of available licenses. The 'Create repository' button is green and prominent.

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

☐ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▼ | Add a license: **None** ▼ ⓘ

Create repository

Licenses

Filter licenses...

- None**
- Apache License 2.0
- GNU General Public License v3.0
- MIT License
- BSD 2-Clause "Simplified" License
- BSD 3-Clause "New" or "Revised" License
- Eclipse Public License 2.0
- GNU Affero General Public License v3.0

[Terms](#) [Privacy](#) [Security](#) [Status](#) [Contact](#)

GitHub

- Il sito choosealicense.com, ti aiuta a capire come concedere in licenza il tuo codice. Una licenza software dice ad altri cosa possono e non possono fare con il tuo codice sorgente, quindi è importante prendere una decisione informata.
- Non sei obbligato a scegliere una licenza. Tuttavia, senza una licenza, si applicano le leggi sul copyright di default, nel senso che conservi tutti i diritti sul tuo codice sorgente e nessuno può riprodurre, distribuire o creare opere derivate dal tuo lavoro.
- La maggior parte delle persone inserisce il proprio testo di licenza in un file denominato `LICENSE.txt`(o `LICENSE.md`) nella radice del repository.

GitHub

Branch: master ▾

Progetto-Topix / LICENSE

Find file

Copy path



ilsalvador83/Progetto-Topix is licensed under the
Mozilla Public License 2.0

Permissions of this weak copyleft license are conditioned on making available source code of licensed files and modifications of those files under the same license (or in certain cases, one of the GNU licenses). Copyright and license notices must be preserved. Contributors provide an express grant of patent rights. However, a larger work using the licensed work may be distributed under different terms and without source code for files added in the larger work.

Permissions

- ✓ Commercial use
- ✓ Modification
- ✓ Distribution
- ✓ Patent use
- ✓ Private use

Limitations

- ✗ Liability
- ✗ Trademark use
- ✗ Warranty

Conditions

- ① Disclose source
- ① License and copyright notice
- ① Same license (file)

This is not legal advice. [Learn more about repository licenses.](#)



ilsalvador83 Initial commit

c55b840 a minute ago

1 contributor

374 lines (293 sloc) | 16.3 KB

Raw

Blame

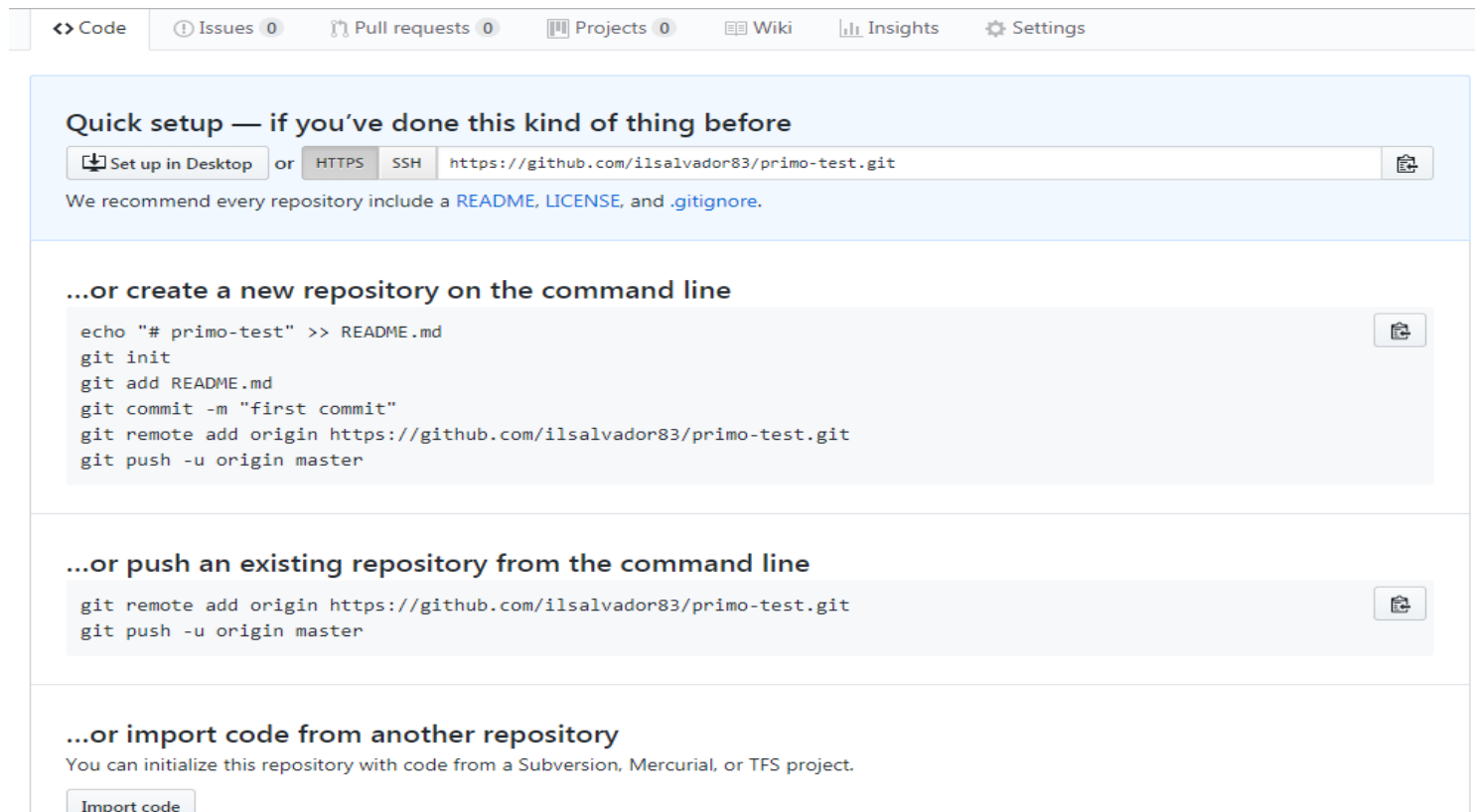
History



```
1  Mozilla Public License Version 2.0
2  =====
3
4  1. Definitions
5  -----
```

GitHub

Dato che non hai ancora nessun codice, GitHub ti mostrerà alcune istruzioni:



The screenshot shows the GitHub interface for a new repository. At the top is a navigation bar with links for Code, Issues (0), Pull requests (0), Projects (0), Wiki, Insights, and Settings. Below this is a light blue box titled "Quick setup — if you've done this kind of thing before". It contains a button "Set up in Desktop" followed by "or" and a dropdown menu with "HTTPS" and "SSH" selected. The URL "https://github.com/ilsalvador83/primo-test.git" is shown next to a copy icon. Below this, it says "We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#)." Below the blue box is a section titled "...or create a new repository on the command line" with a copy icon. It contains a code block with the following commands:

```
echo "# primo-test" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/ilsalvador83/primo-test.git
git push -u origin master
```



 Below this is another section titled "...or push an existing repository from the command line" with a copy icon. It contains a code block with the following commands:

```
git remote add origin https://github.com/ilsalvador83/primo-test.git
git push -u origin master
```


 At the bottom is a section titled "...or import code from another repository" with the text "You can initialize this repository with code from a Subversion, Mercurial, or TFS project." and an "Import code" button.

<> Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings


Quick setup — if you've done this kind of thing before

 Set up in Desktop or **HTTPS** **SSH** `https://github.com/ilsalvador83/primo-test.git` 

We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

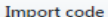
...or create a new repository on the command line 

```
echo "# primo-test" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/ilsalvador83/primo-test.git
git push -u origin master
```

...or push an existing repository from the command line 

```
git remote add origin https://github.com/ilsalvador83/primo-test.git
git push -u origin master
```

...or import code from another repository
You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

 Import code

GitHub

- Creare un nuovo branch 'nuovoramo'

The screenshot shows the GitHub interface for the repository 'ilsalvador83 / primo-test'. The top navigation bar includes links for 'This repository', 'Search', 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. The repository name is displayed as 'ilsalvador83 / primo-test' with 'Watch', 'Star', and 'Fork' buttons showing 0 counts. Below the repository name, there are tabs for 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Insights', and 'Settings'. The 'Code' tab is active, showing the file 'AI.doc' in the 'primo-test' branch. A 'Switch branches/tags' dialog box is open, with 'nuovoramo' entered in the search field. The dialog shows a list of branches, with 'Create branch: nuovoramo from 'master'' highlighted in blue. The background shows the file content area with a 'View Raw' link at the bottom.

GitHub

- Creare un nuovo branch 'nuovoramo'

The screenshot displays the GitHub interface for a repository named 'primo-test' by user 'ilsalvador83'. The top navigation bar includes links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the repository name, there are buttons for 'Watch', 'Star', and 'Fork', each with a count of 0. A secondary navigation bar shows tabs for 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Insights', and 'Settings'. The main content area is divided into sections: 'Default branch' (showing 'master' as the default), 'Your branches' (listing 'nuovoramo'), and 'Active branches' (also listing 'nuovoramo'). Each branch entry shows the last update time and a 'New pull request' button. The 'nuovoramo' branch is highlighted in blue, indicating it is the current branch.

This repository Search

Pull requests Issues Marketplace Explore

ilsalvador83 / primo-test

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Overview Yours Active Stale All branches Search branches...

Default branch

master Updated a day ago by ilsalvador83 Default Change default branch

Your branches

nuovoramo Updated a day ago by ilsalvador83 0 | 0 New pull request

Active branches

nuovoramo Updated a day ago by ilsalvador83 0 | 0 New pull request

GitHub

- Modifca file sulla branch

The screenshot shows the GitHub interface for a repository named 'progetto-tpx' by user 'ilsalvador83'. The top navigation bar includes links for 'This repository', 'Search', 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. On the right, there are icons for notifications, a dropdown menu, and a repository icon. Below the navigation bar, the repository name 'ilsalvador83 / progetto-tpx' is displayed, followed by 'Watch' (0), 'Star' (0), and 'Fork' (0) buttons. A secondary navigation bar contains links for 'Code', 'Issues' (0), 'Pull requests' (0), 'Projects' (0), 'Wiki', 'Insights', and 'Settings'. The main content area shows the 'Branch: nuovoramo' and the file 'progetto-tpx / README.txt'. There are 'Find file' and 'Copy path' buttons. Below this, a commit by 'ilsalvador83' is shown with the message 'Initial Project Version' and a commit hash 'af56200' from '4 days ago'. It lists '1 contributor'. The file content area shows '1 lines (1 sloc) | 7 Bytes'. At the bottom, there is a table with one row: '1 read me'. On the right side of the file content area, there are buttons for 'Raw', 'Blame', and 'History', along with icons for a monitor, a pencil (labeled 'Edit this file'), and a trash can.

This repository Search Pull requests Issues Marketplace Explore

ilsalvador83 / progetto-tpx Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

Branch: nuovoramo progetto-tpx / README.txt Find file Copy path

ilsalvador83 Initial Project Version af56200 4 days ago

1 contributor

1 lines (1 sloc) | 7 Bytes Raw Blame History Edit this file

1	read me
---	---------

GitHub

- Committa la modifica



Commit changes

Update README.txt

Prima modifica

- ☒ Commit directly to the `nuovoramo` branch.
- ☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes

Cancel



PULL REQUEST

- Una volta effettuate le modifiche del tuo lavoro sulla branch, puoi effettuare una "richiesta di pull" (pull request), ovvero puoi proporre agli altri le modifiche che hai inviato chiedendo di aggiungerle al repository.
- È possibile aggiungere un riepilogo delle modifiche proposte, aggiungere etichette, e tramite una @mention chiedere opinioni a singoli collaboratori o team.

GitHub

- Puoi farlo tramite lo stesso sito - GitHub ha un pulsante "pull request" che automaticamente notifica i mantenitori - o eseguire il comando `git request-pull` e inviare manualmente via email l'output.
- Il comando `request-pull` riceve come parametri il branch di base sul quale vuoi far applicare le modifiche e l'URL del repository Git da cui vuoi che le prendano, e produce il sommario di tutte queste modifiche in output.
- `$ git request-pull nuovoramo https://github.com/ilsalvador83/progetto-tpx.git master`



Aprire una pull request

2 commits

2 branches

0 releases

1 contributor

Your recently pushed branches:

nuovoramo (2 minutes ago) [Compare & pull request](#)

Branch: nuovoramo [New pull request](#) [Create new file](#) [Upload files](#) [Find file](#) [Clone or download](#)

This branch is 1 commit ahead of master. [Pull request](#) [Compare](#)

ilsalvador83	Update README.txt	Latest commit 9d24fd0 2 minutes ago
README.txt	Update README.txt	2 minutes ago
prova.php	Initial Project Version	4 days ago

README.txt



```
read me
'modifica'
```




Aprire una pull request

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: master  compare: nuovoramo ✓ Able to merge. These branches can be automatically merged.



Update README.txt


Write

Preview

AA ▾ B i “ <> 🔗 ☰ ☷ ☶ ↶ @ 📌

Prima modifica

Attach files by dragging & dropping, [selecting them](#), or pasting from the clipboard.

 Styling with Markdown is supported

Create pull request

Reviewers

No reviews—request one

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone


GitHub

Effettuare un merge e/o chiudere la pull


Update README.txt #1 Edit

Open ilsalvador83 wants to merge 1 commit into `master` from `nuovoramo`


Conversation 0 Commits 1 Files changed 1 +2 -1

 ilsalvador83 commented 12 minutes ago Owner + (user) edit


Prima modifica

 Update README.txt Verified 9d24fd8


Add more commits by pushing to the `nuovoramo` branch on ilsalvador83/progetto-tpx.


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

Merge pull request You can also open this in GitHub Desktop or view command line instructions.

 Set up continuous integration to automatically test your code
Catch bugs, enforce style, and increase confidence in your code before you merge.

Explore GitHub Marketplace






Notifications

Unsubscribe

You're receiving notifications because you authored the thread.

1 participant



Lock conversation

Write Preview AA B i “ <> ↔ ⋮ ⋮ ⋮ ↶ @ 🔖

Leave a comment


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

Styling with Markdown is supported




Close pull request Comment





Aggiungere collaboratori


 This repository


[Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)


  


 [ilsalvador83](#) / [progetto-tpx](#)


 Watch 0


 Star 0


 Fork 0


 Code


 Issues 0

 Pull requests 0

 Projects 0

 Wiki

 Insights

 Settings

[Options](#)

[Collaborators](#)

[Branches](#)

[Webhooks](#)

[Integrations & services](#)

[Deploy keys](#)

Collaborators

Push access to the repository

This repository doesn't have any collaborators yet. Use the form below to add a collaborator.

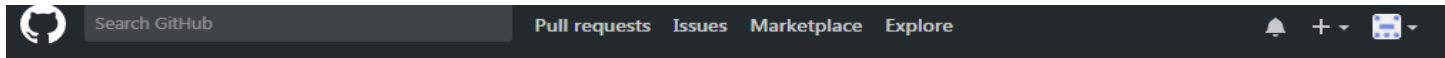
Search by username, full name or email address

You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead.

Add collaborator



Creare una organizzazione



Sign up your team



Completed
Create personal account



Step 2:
Create organization



Step 3:
Invite members

Create an organization account

Organization name

This will be your organization name on <https://github.com/>.

Billing email

We'll send receipts to this inbox.

Organization accounts allow your team to plan, build, review, and ship software — all while tracking bugs and discussing ideas.

Choose your plan

☒ Free

Unlimited users and public repositories

\$0

The credit card and plan you choose will be billed to the organization — not `ilsalvador83` (your user account).

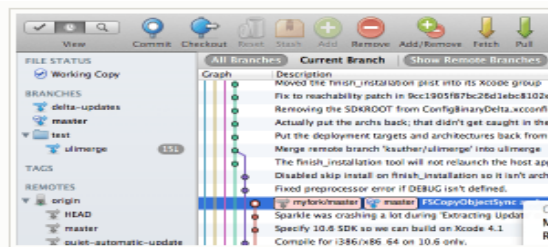
GitHub

Biforcare un progetto

- Se vuoi contribuire ad un progetto esistente a cui non hai un accesso per l'invio, GitHub incoraggia la biforcazione del progetto. Il forking è uno strumento proprio di GitHub e serve a creare una copia server del repository originario.
- Quando vai sulla pagina di un progetto che credi interessante e vuoi lavorarci un po' su, puoi cliccare sul pulsante "fork" nella parte superiore del progetto per avere una copia su GitHub nel tuo utente a cui puoi inviare le modifiche.

Git-Client

- Oltre alla riga di comando, esistono in commercio numerosi Client per interagire con i repository GIT.

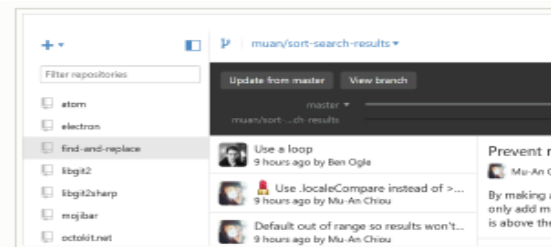


SourceTree

Platforms: Mac, Windows

Price: Free

License: Proprietary

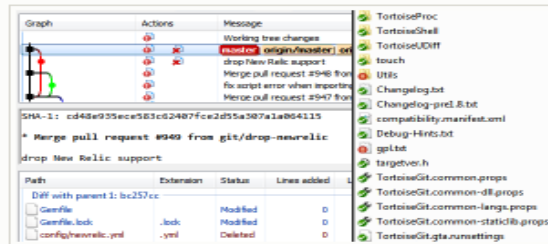


GitHub Desktop

Platforms: Mac, Windows

Price: Free

License: MIT

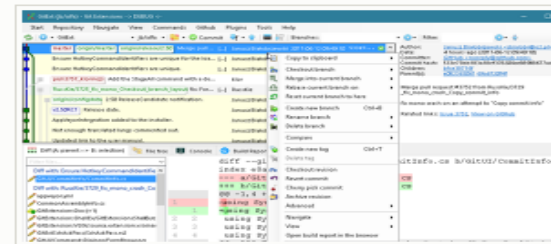


TortoiseGit

Platforms: Windows

Price: Free

License: GNU GPL



Git Extensions

Platforms: Linux, Mac, Windows

Price: Free

License: GNU GPL

Git-Client

- Apache License 2.0
- GNU General Public License v3.0
- MIT License
- Licenza BSD Semplificata/ Licenza FreeBSD (2 clausole)
- Licenza BSD modificata/Nuova licenza BSD (3 clausole)
- Eclipse Public License
- GNU Affero General Public License v3.0
- GNU General Public License v3.0
- GNU Lesser General Public License v3.0
- Mozilla Public License 2.0
- The Unlicense

Contatti e Fonti

GitHUB

- IIsalvador83
- cryptoluca
- <https://git-scm.com/book>