

# Introduzione a GitHub

# Cos'è GitHub?

In breve...

- È una piattaforma web per la condivisione e gestione di progetti Software.
- Nasce come implementazione di **Git**, uno strumento per il *version control*, che permette di gestire e tenere traccia delle versioni e delle modifiche apportate al codice sorgente del software.



# Installare Git

**Git** può essere  
scaricato presso:

<https://git-scm.com/downloads>



# GitHub Desktop

**GitHub** può essere utilizzato sia tramite interfaccia grafica che da linea di comando.

La versione desktop è disponibile presso:

<https://desktop.github.com/>



# Creare un Repository

Un **repository** (letteralmente deposito o ripostiglio) è un ambiente di un in cui vengono gestiti i metadati del progetto.

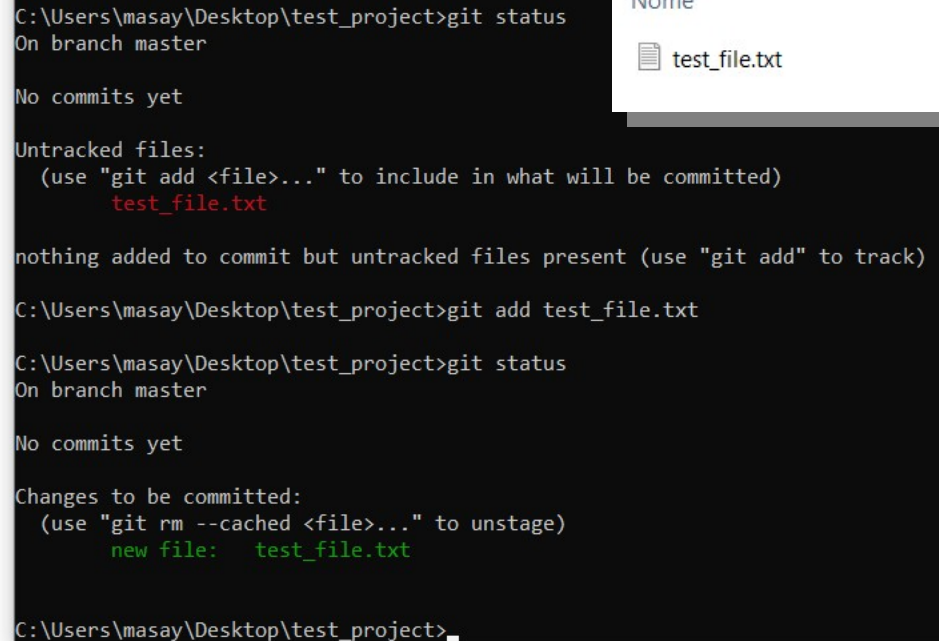
È possibile creare un repository all'interno di una cartella tramite il comando: *git init*

```
C:\Users\masay\Desktop>cd test_project
C:\Users\masay\Desktop\test_project>git init
Initialized empty Git repository in C:/Users/masay/Desktop/test_project/.git/
C:\Users\masay\Desktop\test_project>_
```

# Aggiungere un File al Repository

Aggiungendo o modificando un **file** all'interno della cartella che contiene il repository, Git si accorgerà della sua presenza, ma bisogna dirgli esplicitamente di tenere traccia dello stato di un file.

- Il comando `git status` mostra lo stato dei file all'interno del repository;
- Il comando `git add nomefile` dice a Git di tracciare quel file.



The image shows a terminal window with the following text:

```
C:\Users\masay\Desktop\test_project>git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    test_file.txt

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

C:\Users\masay\Desktop\test_project>git add test_file.txt

C:\Users\masay\Desktop\test_project>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   test_file.txt

C:\Users\masay\Desktop\test_project>
```

In the background, a Windows File Explorer window is open to the 'test\_project' directory, showing a file named 'test\_file.txt'.

# La Commit: cos'è?

Una **Commit** è una registrazione dei cambiamenti avvenuti dall'ultima Commit fatta.

In poche parole: è come salvare un **checkpoint**.

Quando vengono modificati dei file all'interno del repository, facendo una Commit crei uno stato del progetto a cui poter tornare. I file coinvolti nella Commit sono solo quelli aggiunti tramite il comando *git add*.

# Fare una Commit

Tramite il comando *git status* possiamo vedere quali cambiamenti non sono ancora stati “committati”.

- Il comando per fare una **Commit** è:  
*git commit -m "messaggio"*

```
C:\Users\masay\Desktop\test_project>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   test_file.txt

C:\Users\masay\Desktop\test_project>git commit -m "La mia prima commit!"
[master (root-commit) b8d0618] La mia prima commit!
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test_file.txt

C:\Users\masay\Desktop\test_project>git status
On branch master
nothing to commit, working tree clean

C:\Users\masay\Desktop\test_project>
```



# Creare un Branch

Un **Branch** è una copia del progetto a partire da una particolare commit. Le modifiche al Branch del progetto non vengono eseguite sulla copia originale.

Il comando per passare ad un nuovo Branch è:

*git checkout -b nomebranch*

```
C:\Users\masay\Desktop\test_project>git checkout -b nuovo_branch  
Switched to a new branch 'nuovo_branch'  
  
C:\Users\masay\Desktop\test_project>
```

# Merging di un Branch

Dopo aver effettuato delle modifiche “convincenti” sul nuovo Branch possiamo aggiornare il progetto principale e mantenere una sola versione, facendo una **Merge**.

- Il comando `git branch` mostra l'elenco dei branch esistenti evidenziando quella su cui si sta lavorando;
- Il comando `git checkout nomebranch` permette di cambiare branch (la copia originale è chiamata `master` di default);
- Il comando `git merge nomebranch`, aggiorna il branch corrente con le modifiche di quello citato nel comando.

```
C:\Users\masay\Desktop\test_project>git checkout -b nuovo_branch
Switched to a new branch 'nuovo_branch'

C:\Users\masay\Desktop\test_project>git branch
  master
* nuovo_branch

C:\Users\masay\Desktop\test_project>git status
On branch nuovo_branch
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   test_file.txt

no changes added to commit (use "git add" and/or "git commit -a")

C:\Users\masay\Desktop\test_project>git add test_file.txt

C:\Users\masay\Desktop\test_project>git commit -m "commit sul nuovo branch"
[nuovo_branch 3b09763] commit sul nuovo branch
 1 file changed, 1 insertion(+)

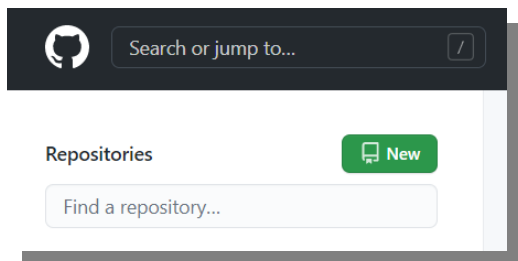
C:\Users\masay\Desktop\test_project>git checkout master
Switched to branch 'master'

C:\Users\masay\Desktop\test_project>git merge nuovo_branch
Updating b8d0618..3b09763
Fast-forward
 test_file.txt | 1 +
 1 file changed, 1 insertion(+)

C:\Users\masay\Desktop\test_project>_
```

# Creare un Repository su GitHub

Per condividere il proprio progetto su **GitHub** è necessario creare un nuovo repository sul proprio profilo personale.



## Create a new repository



A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner \*  VOrigatti / Repository name \* nuovorepo ✓

Great repository names are short and memorable. Need inspiration? How about [verbose-fiesta?](#)

Description (optional)

Il mio nuovo repository!

- ☒  **Public**  
Anyone on the internet can see this repository. You choose who can commit.
- ☐  **Private**  
You choose who can see and commit to this repository.

### Initialize this repository with:

Skip this step if you're importing an existing repository.

- ☐ **Add a README file**  
This is where you can write a long description for your project. [Learn more.](#)
- ☐ **Add .gitignore**  
Choose which files not to track from a list of templates. [Learn more.](#)
- ☐ **Choose a license**  
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

# Push di un Repository su GitHub

Per caricare un repository locale in un repository GitHub bisogna settare l'indirizzo di quest'ultimo tramite il comando:  
*git remote add origin indirizzorepositorygithub*

E caricare il repository locale con un comando di **Push**:  
*git push -u origin nomebranch*

```
C:\Users\masay\Desktop\test_project>git remote add origin https://github.com/VDorigatti/nuovorepo.git

C:\Users\masay\Desktop\test_project>git push -u origin master
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (6/6), 450 bytes | 225.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/VDorigatti/nuovorepo.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.

C:\Users\masay\Desktop\test_project>
```

Se viene fatta la **Push** di un *branch diverso*, verrà automaticamente creato un nuovo branch anche sul repository GitHub.

# Fare una Pull da GitHub

Se sono stati effettuati dei cambiamenti sul repository GitHub, ma non su repository locale, è possibile aggiornare il repository locale tramite una **Pull**.

Il comando corrispondente è:  
*git pull origin nomebranch*

```
C:\Users\masay\Desktop\test_project>git pull origin master
From https://github.com/VDorigatti/nuovorepo
* branch          master      -> FETCH_HEAD
Already up to date.

C:\Users\masay\Desktop\test_project>
```

In questo caso non ci sono aggiornamenti perché il nostro repository locale è già **aggiornato**.

# Cos'è una Pull Request?

Una **Pull Request** viene fatta per informare il proprietario di un repository che vogliamo fare dei cambiamenti al suo progetto.

Sarà il proprietario del repository a **revisionare** i cambiamenti proposti all'interno della Pull Request e decidere se approvare questi cambiamenti nel progetto principale.

# Esempio di Pull Request

Proviamo a...

- **Creare un nuovo branch** “PROVA” sul repository locale;
- **Fare una commit** di alcune modifiche;
- **Fare una push** del nuovo branch sul nostro repository GitHub.

```
C:\Users\masay\Desktop\test_project>git status
On branch PROVA
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   nuovo_file.txt
        modified:   test_file.txt

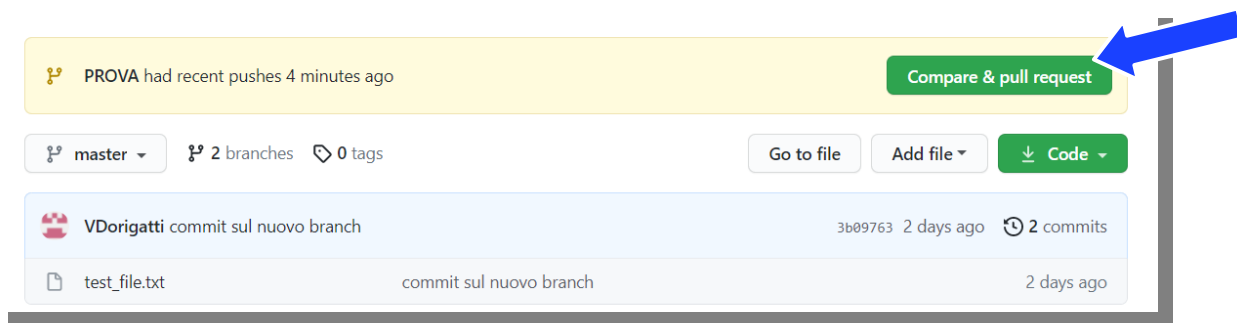
C:\Users\masay\Desktop\test_project>git commit -m "modifiche al nuovo branch"
[PROVA 00f2b57] modifiche al nuovo branch
 2 files changed, 2 insertions(+), 1 deletion(-)
 create mode 100644 nuovo_file.txt

C:\Users\masay\Desktop\test_project>git push -u origin PROVA
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 309 bytes | 154.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'PROVA' on GitHub by visiting:
remote:   https://github.com/VDorigatti/nuovorepo/pull/new/PROVA
remote:
To https://github.com/VDorigatti/nuovorepo.git
 * [new branch]      PROVA -> PROVA
Branch 'PROVA' set up to track remote branch 'PROVA' from 'origin'.

C:\Users\masay\Desktop\test_project>
```

# Esempio di Pull Request

Sul nostro repository GitHub vedremo comparire il nuovo branch di cui abbiamo fatto la push con l'opzione di richiedere una **Pull Request**.





# Esempio di Pull Request

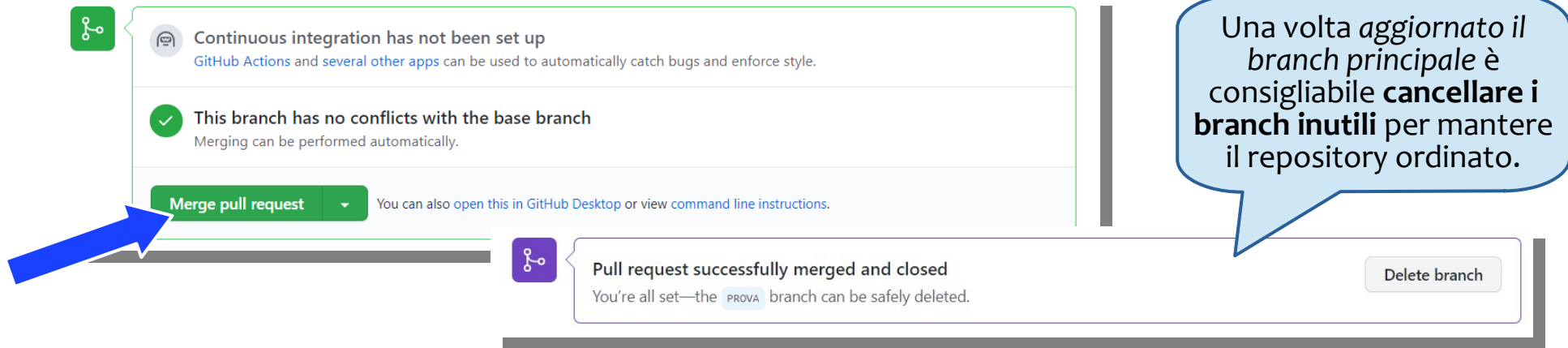
Sul nostro repository GitHub vedremo comparire il nuovo branch di cui abbiamo fatto la push con l'opzione di richiedere una **Pull Request**.

The screenshot displays the GitHub web interface for a repository. At the top, a yellow notification bar states "PROVA had recent pushes 4 minutes ago" with a green button labeled "Compare & pull request". Below this, the repository's main view shows the "master" branch selected, with "2 branches" and "0 tags" indicated. A commit by "VDorigatti" is shown, titled "commit sul nuovo branch", with a file named "test\_file.txt" added. To the right, the "Open a pull request" section is active, showing a comparison between "base: master" and "compare: PROVA". A green checkmark indicates "Able to merge". Below the comparison, there is a text input field for the pull request title, a "Write" tab, a "Preview" tab, and a large text area for comments. At the bottom right, a green button labeled "Create pull request" is highlighted with a blue arrow.

# Merging di una Pull Request

Una volta effettuata una la **Pull Request** vedremo comparire la richiesta pronta per essere revisionata.

(dato che in questo caso i proprietari siamo noi)



The screenshot shows the GitHub Pull Request interface. A green box highlights the status section, which includes a green checkmark indicating 'This branch has no conflicts with the base branch' and a green 'Merge pull request' button. A blue arrow points to this button. Below this, a purple box shows a confirmation message: 'Pull request successfully merged and closed'. To the right, a blue speech bubble contains Italian text advising to delete unnecessary branches after updating the main branch.

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.

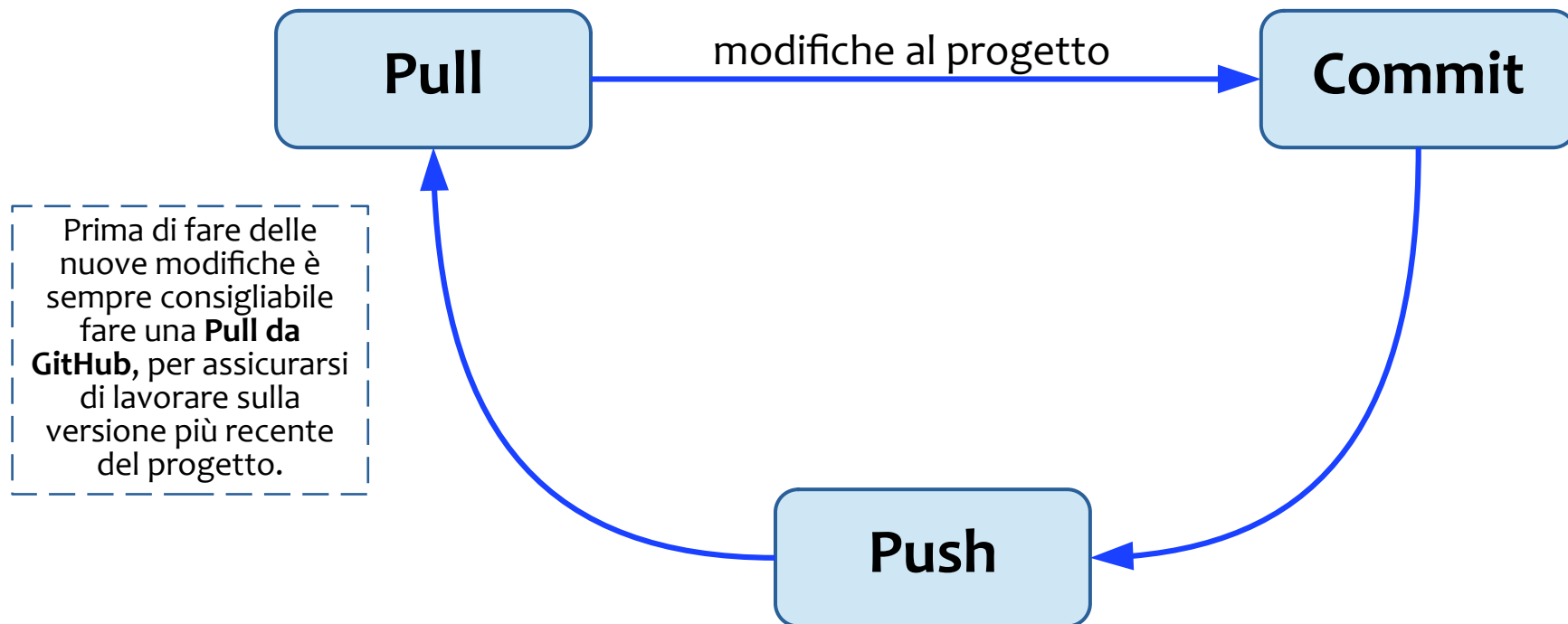
Merge pull request You can also open this in GitHub Desktop or view command line instructions.

Pull request successfully merged and closed  
You're all set—the `PROVA` branch can be safely deleted.

Delete branch

Una volta *aggiornato il branch principale* è consigliabile **cancellare i branch inutili** per mantenere il repository ordinato.

# Best Practice: il ciclo di utilizzo



# Grazie dell'attenzione!

Per qualsiasi domanda potete contattarmi all'indirizzo:  
[vdorigatti@uninsubria.it](mailto:vdorigatti@uninsubria.it)