

VCS-GIT-LAB-4

Gitlab

MARCELLO MISSIROLI
**Tecnologia
e Progettazione**
per il mondo **digitale**
e per il **web II**



 digital docet

Prerequisiti

COMPUTER DI LABORATORIO O LAPTOP PERSONALE

Avendo completato l'esperienza precedente:

- ▶ Git installato
- ▶ Un repo git locale
- ▶ Accesso a internet

LA DURATA PREVISTA DI QUESTA ATTIVITÀ È DI 1 ORA DI 50 MINUTI.

1.

Account su Gitlab

Il nostro deposito remoto.

Collegatevi a Gitlab (www.gitlab.com)


Solutions ▾ Resources ▾ Partners ▾ Pricing Support ▾


Welcome to the DevOps Platform era. GitLab CEO Sid Sijbrandij discusses the next iteration of DevOps. [Learn more.](#)

GitLab is the DevOps Platform


Bring velocity with confidence, security without sacrifice, and visibility into DevOps success.

[Try GitLab for FREE](#) [Watch a demo](#)




 **Single source of truth**

Manage projects, not tools. With GitLab, you get a DevOps platform delivered as a single application—one interface, one

 **Continuous everything**

Bridge Dev and Ops once and for all. Our industry-leading CI/CD empowers all teams to work together efficiently. Powerful,

 **Real-time security**

See everything that matters. Built-in everything brings automated security, quality, and vulnerability management. With

We're looking for people to take part in an online survey to help us improve our website. [Take survey](#)

Username or email

Password

☐ Remember me [Forgot your password?](#)

[Sign in](#)

Don't have an account? [Register now](#)

Sign in with

[Google](#)

[GitHub](#)

[Twitter](#)

[Bitbucket](#)

[Salesforce](#)

☐ Remember me

Registrarevi


Resources ▾ Partners ▾ Pricing Support ▾

Get free trial Login

Welcome to the DevOps Platform era. GitLab CEO Sid Sijbrandij discusses the next iteration of DevOps. [Learn more.](#)

GitLab is the DevOps platform

Bring velocity with confidence, security without sacrifice, and visibility into DevOps processes.



Single source of truth

Manage projects, not tools. With GitLab, you have a DevOps platform delivered as a single application—one interface, one source of truth.

Continuous everything

Bridge Dev and Ops once and for all. Our industry-leading CI/CD empowers all teams to work together efficiently. Powerful, continuous everything.

Real-time security

See everything that matters. Built-in everything brings automated security, code quality, and vulnerability management. With real-time security.

[Take survey](#)

We're looking for people to take part in an online survey to help us improve our website.

[GitLab for FREE](#) [Watch a demo](#)


First name Last name

Username

Email

Password

Minimum length is 8 characters.

☐ Non sono un robot 

[reCAPTCHA Privacy - Terms](#)

[Register](#)

By clicking Register, I agree that I have read and accepted the [GitLab Terms of Use and Privacy Policy](#)

or

Create an account using:

[Google](#) [GitHub](#)

[Twitter](#) [Bitbucket](#)

[Salesforce](#)

Already have login and password? [Sign in](#)

I protocolli remoti di Git utilizzano una crittografia a chiave pubblica. Occorre pertanto generare queste chiavi. Non servono oggi, ma visto che siamo in fase di configurazione ne approfittiamo. Una volta create, vanno conservate con cura.

Chiave pubblica-privata (opzionale ma consigliata)

`ssh-keygen -C "alex-york@hotmail.co.uk" -t rsa`

Copiate la cartella `.ssh` per sicurezza

```
$prova@pc:~$ ssh-keygen -t rsa -b 2048 -C  
"alex-york@hotmail.co.uk"  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/prova/.ssh/id_rsa):  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/prova/.ssh/id_rsa  
Your public key has been saved in /home/prova/.ssh/id_rsa.pub  
The key fingerprint is:  
SHA256:cm3e5HfH98K6MvLX7so+IR85B95Kc1R20Xa1KHcigWU per gitlab
```

Copiate il contenuto del file `.ssh/id_rsa.pub` nella clipboard, quindi incollatela nella casella “Key” della pagina “User Settings > SSH Key”

Aggiungete la vostra chiave pubblica a Gitlab

User Settings > SSH Keys

Q Search settings

SSH Keys

SSH keys allow you to establish a secure connection between your computer and GitLab.

Add an SSH key

To add an SSH key you need to [generate one](#) or use an [existing key](#).

Key

Paste your public SSH key, which is usually contained in the file '~/.ssh/id_ed25519.pub' or '~/.ssh/id_rsa.pub' and begins with 'ssh-ed25519' or 'ssh-rsa'. Do not paste your private SSH key, as that can compromise your identity.

```
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAQAC/h55TKdCU4Yi2Qm+XCmAnyMejCOOPOp5fXeJ/g4
c7nyk7VJ2Jr4CYt3R09FsA5+ZmNXlyRMSsuSvJ8VlvkhEU2WoXddE8ZRNpkIDDD7Y6tNk52LmgN
7sDZJVnF6LQs2k+X/ChjZmQyTA+CWVRHvgRhK9WmnGOQKfwVPY8RnKLTCaZf1Ffli+6sdqlpM
Q3wlWZzsb0DHnBf60BlvjYFEb/ORCPssz0GOYwAYtWdrQmqjqeiVKWDdAHF3XtSiclwxx/3eqyX
5CwfQz6a56UubjgDpTo90opS/eS+yBhzLmj438MhD2kiKJLJYaZ2ONFgDqBdepKN2ljw00IwjdfA
V/d per gitlab
```

Title

per gitlab

Give your individual key a title. This will be publicly visible.

Expires at

gg/mm/aaaa

Key will be deleted on this date.

Add key

Verifica

```
$prova@pc:~$ssh -T git@gitlab.com  
Welcome to GitLab, @alex!
```

“

Checkpoint #1:

- ▶ *Accesso a Gitlab eseguito e configurato*

”

2.

Il repo remoto

Il primo remoto non si scorda mai

New Project > Blank project



Create blank project

Create a blank project to house your files, plan your work, and collaborate on code, among other things.

New project > Create blank project

Project name


Project URL

Project slug

Want to house several dependent projects under the same namespace? [Create a group](#).

Project description (optional)

Visibility Level

☒  Private

Project access must be granted explicitly to each user. If this project is part of a group, access will be granted to members of the group.

☐  Public

The project can be accessed without any authentication.

☒ Initialize repository with a README

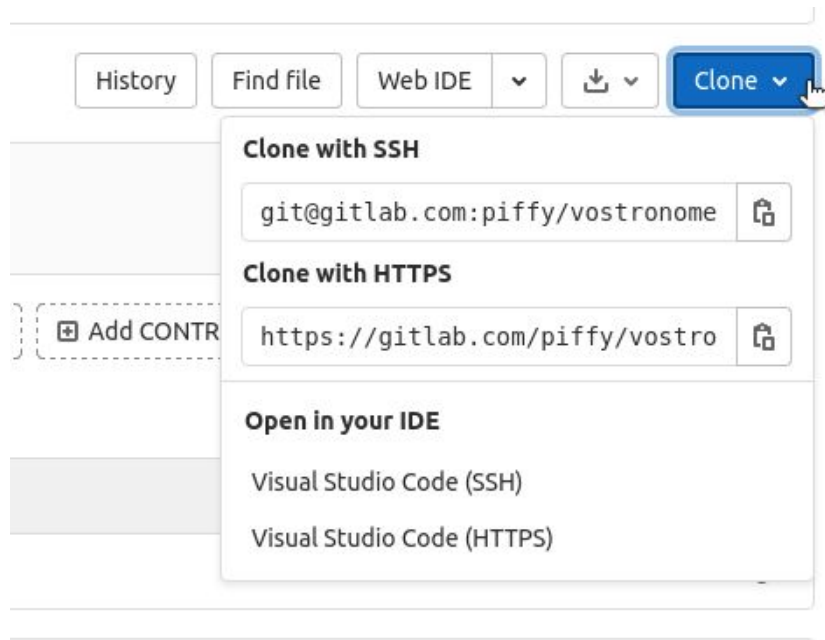
Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

Create project

Cancel

Collegiamo al remoto 1:

Identificare l'URL del progetto su Gitlab andando alla home page del progetto, fare click su clone, quindi copiarlo nella clipboard.



Collegiamo al remoto 2:

Dare i seguenti comandi:

à errore ignorate

```
$prova@pc:~/demo$  
git branch -m master main  
git remote add origin [URL DI GITLAB]  
  
git push -u origin --all --force  
  
$prova@pc:~/demo$
```

Collegiamo al remoto 3:

Risultato: qualcosa di simile a questo (dopo avervi chiesto la password)

```
Enumerazione degli oggetti in corso: 17, fatto.  
Conteggio degli oggetti in corso: 100% (17/17), fatto.  
Compressione delta in corso, uso fino a 8 thread  
Compressione oggetti in corso: 100% (9/9), fatto.  
Scrittura degli oggetti in corso: 100% (17/17), 1.37 KiB |  
1.37 MiB/s, fatto.  
17 oggetti totali (0 delta), 0 riutilizzati (0 delta)  
[...]  
To https://gitlab.com/piffy/vostronome.git  
* [new branch]      main -> main  
Branch 'main' impostato per tracciare il branch remoto 'main'  
da 'origin'
```


Pulliamo:

Dare i seguenti comandi e controllate su Gitlab:

```
$prova@pc:~/demo$  
  
git remote add origin [URL DI  
GITLAB]  
  
git push -u origin --all --force  
  
$prova@pc:~/demo$
```

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

Name	Last commit
 temp	gitignore
 .gitignore	gitignore
 README.md	Initial commit
 altrodemo.txt	secondo file
 demo.txt	terzo commit

Verifica

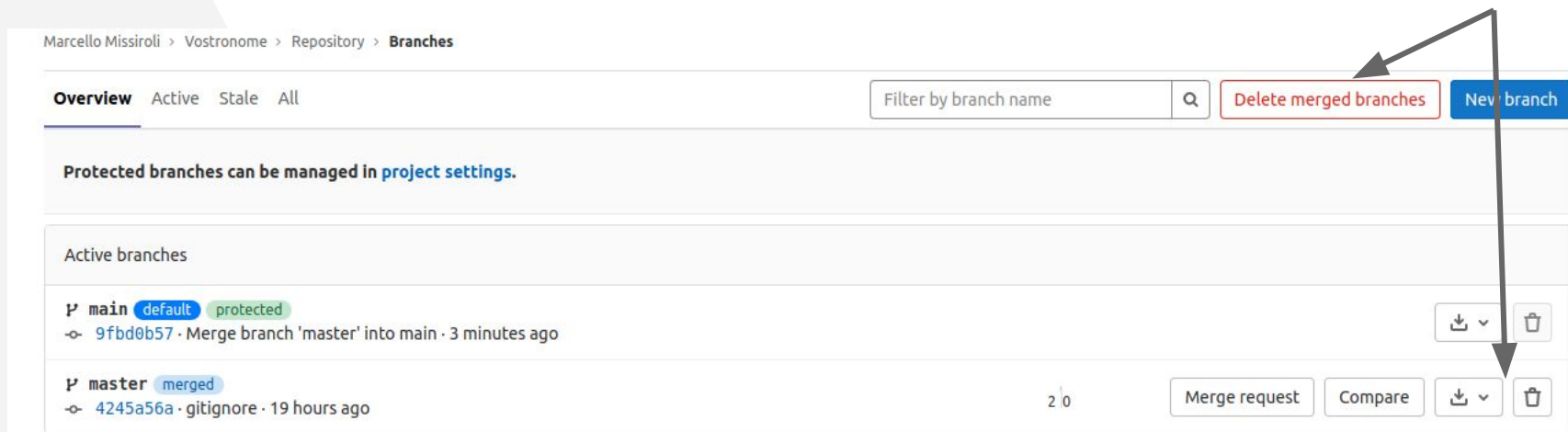
- ▶ Modificare un file (tipo demo.txt), poi commitare veloce con `git commit -am "commento"`.
- ▶ Dare `git push`
- ▶ Controllare sul sito remoto che il file sia stato modificato

Troubleshooting (fix, 1/2)

- ▶ Sul sito sono presenti due branch (main e master)?
Per prima cosa:
 - ▷ `git checkout main`
 - ▷ `git merge master`
`--allow-unrelated-histories`
 - ▷ `git branch -d master`

Troubleshooting (fix, 2/2)

Quindi cancellare il branch obsoleto su gitlab con `git push origin --delete master` oppure




Troubleshooting: protected branches

- ▶ Nelle ultime versioni di Gitlab, il branch main è classificato “protected”, e non è possibile fare direttamente il push dall'esterno (ammesso solo ai maintainer).
- ▶ Viene indicato da questo errore:

```
S.Silvestre@LAB-INFO-19 MINGW64 ~/demo (main)
$ git push -u origin --all --force
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (9/9), 741 bytes | 370.00 KiB/s, done.
Total 9 (delta 0), reused 0 (delta 0), pack-reused 0
remote: GitLab: You are not allowed to force push code to a protected branch on this project.
To gitlab.com:silve961/demo1.git
! [remote rejected] main -> main (pre-receive hook declined)
error: failed to push some refs to 'gitlab.com:silve961/demo1.git'
```

Troubleshooting: protected branches (fix)

- ▶ "Settings" → "Repository" → click sul pulsante [Expand] alla destra di "Protected branches"
- ▶ Premere su "Unprotect" a lato del branch "main"



Branch	Allowed to merge	Allowed to push	Allowed to force push ?	Code owner approval ?	
main default	Maintainers ▾	Maintainers ▾	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	Unprotect

“

Checkpoint #2:

- ▶ *Sapete inizializzare un repo remoto*
- ▶ *Sapete collegarlo a un repo esistente*
- ▶ *Sapete tenere sincronizzati i due repo*

”

3.

Collaboratori

L'unione fa la forza

Se avete un repo pubblico, tutti possono clonare il vostro repo, ma nessuno può modificarlo.

Per poterlo fare, occorre espressamente autorizzare gli utenti coinvolti.

Organizzatevi a coppie ed eleggete un “capo” e un “dipendente”

Capo: andare su Project Information > Member

Project members

You can invite a new member to **Vostronome** or invite another group.

Invite member

Invite group

GitLab member or Email address

Select a role



Developer

Learn more about roles.

Access expiration date

Invite

Import

Filter members						
Account	Source	Access granted	Access expires	Max role	Expiration	
 Marcello Missiroli It's you @piffy	Direct member	19 hours ago by Marcello Missiroli	No expiration set	Maintainer	Expiration date	

Se avete un repo pubblico, tutti possono clonare il vostro repo, ma nessuno può modificarlo. Per poterlo fare, occorre espressamente autorizzare gli utenti coinvolti.

Per questo esempio Francesco è eletto “capo”. Gli altri saranno “dipendenti”

Controllate se siete stati inseriti come collaboratori (dovrebbe apparire tra i vostri progetti).

Capo: first push

Creare localmente un repo.

Collegarlo al repo remoto (remote add)

Creare e committare un file chiamato
“demo3.txt” contenente una riga di testo

Pushare sul remoto.

Dipendente: clone

Aprire un terminale e clonate il repo.

```
$prova2@altropc:git clone [Gitlab URL]
git clone [Gitlab URL]
Clone in 'vostronome' in corso...
remote: Enumerating objects: 22, done.
remote: Counting objects: 100% (22/22), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 22 (delta 1), reused 0 (delta 0), pack-reused 0
Ricezione degli oggetti: 100% (22/22), 4.41 KiB | 4.41 MiB/s, fatto.
Risoluzione dei delta: 100% (1/1), fatto.
$prova2@altropc:~/vostronome$
```

Dipendente: aggiungete un nuovo file

Nell'ordine:

- ▶ Create un nuovo file chiamato `demo[vostrome].txt`
- ▶ Aggiungetelo alla staging area
- ▶ Committate
- ▶ Pushate

Capo: pullare e controllare

Nell'ordine:

- ▶ Pullare
- ▶ Controllare che il nuovo file sia arrivato

“

Checkpoint #3:

- ▶ *Sapete aggiungere collaboratori*
- ▶ *Sapete clonare un repo*
- ▶ *Sapete modificare un repo da qualsiasi postazione*

”

4.

Conflitti

Non sempre tutto fila liscio

Capo: modifica

Modificare il file demo.txt (aggiungete il vostro nome e cognome).

Salvare, committare, pushare

Dipendente: modifica

Modificare il file demo.txt (aggiungete il vostro nome e cognome). Salvare, committare, pushare.

```
$prova2@altropc:git push
git push
To [Gitlab repo]
 ! [rejected]
error: push di alcuni riferimenti su '[Gitlab repo]' non riuscito
suggerimento: Gli aggiornamenti sono stati rifiutati perché il remoto
contiene delle modifiche che non hai localmente. Ciò solitamente è
causato da un push da un altro repository allo stesso riferimento.
Potresti voler integrare le modifiche remote (ad es. con 'git pull ...')
prima di eseguire nuovamente il push.
Vedi la 'Nota sui fast forward' in 'git push --help' per ulteriori
suggerimento: dettagli.
$prova2@altropc:~/vostronome$
```

Dipendente: modifica

Seguire il suggerimento (git pull) provoca il conflitto.

```
$prova2@altropc:git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Decompressione degli oggetti in corso: 100% (3/3), 260 byte | 260.00
KiB/s, fatto.
Da gitlab.com:piffy/vostronome
   9a06d64..d19a0f3  main      -> origin/main
Merge automatico di demo3.txt in corso
CONFLITTO (contenuto): conflitto di merge in demo3.txt
Merge automatico fallito; risolvi i conflitti ed esegui il commit
del risultato.
$prova2@altropc:~/vostronome$
```

Risoluzione del conflitto: metodo 1 (manuale)

- ▶ Editare il file.

```
GNU nano 4.8 demo3.txt  
Un file di Bill Gates e Alex York
```

Risoluzione del conflitto: metodo 1 (manuale)

▶ Committare il file

```
prova2@altropc:~/vostronome$ git add demo.txt
prova2@altropc:~/vostronome$ git commit -m "risoluzione conflitto"
[main c73fafb] risoluzione conflitto

$prova2@altropc:~/vostronome$
```

Risoluzione del conflitto: metodo 1 (manuale)

► Pushare

```
prova2@altropc:~/vostronome$ git add demo3.txt
prova2@altropc:~/vostronome$ git commit -m "risoluzione conflitto"
[main c73fafb] risoluzione conflitto

$prova2@altropc:~/vostronome$
```

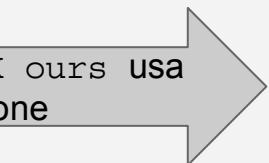

Risoluzione del conflitto: metodo 2 (forzante)

- ▶ Azzerare il pull

```
prova2@altropc:~/vostronome$ git reset --hard HEAD
HEAD ora si trova a c73fafb Modifica prima riga per Bill Gates
prova2@altropc:
```

Risoluzione del conflitto: metodo 2 (forzante)

- ▶ Pullare accettando le modifiche remote e pushare



```
prova2@altropc:~/vostronome$ git reset --hard HEAD
HEAD ora si trova a c73fafb Modifica prima riga per Bill Gates
prova2@altropc:git pull -X theirs
Aggiornamento di c73fafb..b33d03c
Fast-forward
 demo3.txt | 3 +--
 1 file changed, 1 insertion(+), 2 deletions(-)
```

Risoluzione del conflitto: metodo 3 (distruttivo)

Resettare e pushare. QUESTO ELIMINA PER SEMPRE LE MODIFICHE NON ACCETTATE.

```
$prova@pc:~/demo$ git reset --hard HEAD  
$prova@pc:~/demo$ git push -f  
$prova@pc:~/demo$
```

“

Checkpoint #4:

- ▶ *Sapete quando si generano i conflitti*
- ▶ *Sapete risolvere i conflitti*

”

5.

Filesystem

Alla fine, si parla di questo

Git come filesystem

Linus Torvalds aveva in mente un filesystem quando lavorava su Git. E in effetti molti comandi permettono in pratica di “lavorare” sul filesystem git

Git come filesystem

- ▶ `git rm`: equivale a Unix `rm`
- ▶ `git mv`: equivale a Unix `mv` (e serve quindi anche a cambiare il nome, mantenendo il tracciamento)
- ▶ `git clean --force`: elimina tutti i file NON TRACCIATI dalla directory di lavoro

(le directory non esistono in git, scompaiono se sono vuote. Se volete tenerle, la convenzione è quello di avere un file fittizio chiamato `.gitkeep`)

Miniesercizio

- ▶ Cambiate il nome al file demo.txt in demo4.txt
- ▶ Committare e guardare il log
- ▶ Create un nuovo file “inutile.txt”
- ▶ Ripulite la directory

“

Checkpoint #5:

- ▶ *Sapete rinominare e spostare i file*
- ▶ *Sapete ripulire le cartelle di lavoro*

”

GRAZIE!

Torniamo alla teoria!

Credits

Special thanks to all the people who made and released these awesome resources for free:

- ▶ Presentation template by [SlidesCarnival](#)
- ▶ Photographs by [Startupstockphotos](#)
- ▶ Anil Gupta (www.guptaanil.com)
- ▶ Pete Nicholls (github.com/Aupajo)
- ▶ Armando Fox

Questo documento è distribuito con licenza CreativeCommon BY-SA 3.0

Presentation design

This presentation uses the following typographies and colors:

- ▶ Titles: **Dosis**
- ▶ Body copy: **Roboto**

You can download the fonts on these pages:

<https://www.fontsquirrel.com/fonts/dosis>

<https://material.google.com/resources/roboto-noto-fonts.html>

- ▶ Orange **#ff8700**

You don't need to keep this slide in your presentation. It's only here to serve you as a design guide if you need to create new slides or download the fonts to edit the presentation in PowerPoint®

SlidesCarnival icons are editable shapes.

This means that you can:

- Resize them without losing quality.
- Change line color, width and style.

Isn't that nice? :)

Examples:

