



# **Introduzione a GitHub**

## **Corso di Laboratorio di Gestione progetto e organizzazione d'impresa**

Prof. Dei Rossi, Leonardo Essam

## Che cos'è GitHub?

GitHub è un servizio di hosting per progetti software, di proprietà della società GitHub Inc., con sede legale a San Francisco in California, controllata da Microsoft.

Il nome deriva dal fatto che GitHub è una implementazione dello strumento di controllo versione distribuito Git.

**GitHub è la parte che rende Git un VCS ibrido!**

# Creazione della repository (1)

La creazione della repository su GitHub equivale al creare una repository in locale con il comando `git init`.

Tramite la procedura guidata di GitHub:


## Create a new repository

Repositories contain a project's files and version history. Have a project elsewhere? [Import a repository](#).

*Required fields are marked with an asterisk (\*).*

### 1 General

Owner \*

 leonardodeirosi ▾

Repository name \*

/ progetto-5inb

✔ progetto-5inb is available.

Great repository names are short and memorable. How about [sturdy-guide?](#)

Description

0 / 350 characters

## Creazione della repository (2)

È possibile poi definire delle impostazioni sulla visibilità del repository:

### 2 Configuration

#### Choose visibility \*

Choose who can see and commit to this repository

 Public ▾

#### Add README

READMEs can be used as longer descriptions. [About READMEs](#)

Off ☐

#### Add .gitignore

.gitignore tells git which files not to track. [About ignoring files](#)

No .gitignore ▾

#### Add license

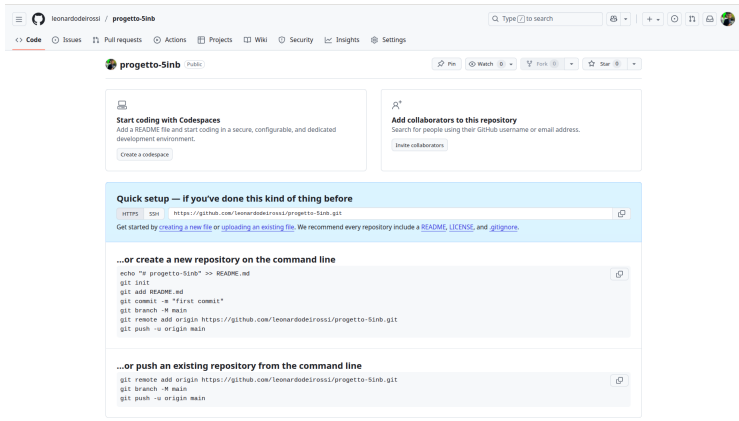
Licenses explain how others can use your code. [About licenses](#)

No license ▾

Create repository

# Creazione della repository (3)

Di default il repository viene creato vuoto:



## La sicurezza prima di tutto (1)

Per poter integrare Git Bash e GitHub non basta la combinazione di nome utente e password, da qualche anno infatti è diventato obbligatorio l'uso di un token di accesso per aggiungere un layer in più di sicurezza.

Per generare un nuovo token si procede da:

- Profile > Settings > Developer settings

E poi:

- Personal access tokens > Tokens (classic)

Infine:

- Generate new token > Generate new token (classic)

## La sicurezza prima di tutto (3)

Ogni token è caratterizzato da:

- Un nome descrittivo;
- Una scadenza;
- L'insieme di *scope* per cui può essere utilizzato.

Gli *scope* dei token (non solo in GitHub, ma anche in generale<sup>1</sup>) sono le azioni che il singolo token è autorizzato a compiere. Ad esempio, un token può gestire solo il push/pull per/da i repository mentre un altro token può gestire anche operazioni amministrative usando le REST API di GitHub.

---

<sup>1</sup>Lo vedremo in informatica (forse...)

## La sicurezza prima di tutto (4)

Per l'uso in laboratorio, è sufficiente il set di scope chiamato "repo":

### Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

<input checked="" type="checkbox"/> <b>repo</b>	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events



## La sicurezza prima di tutto (5)

Per non dover fare ogni volta copia-incolla del token, è possibile configurare Git in modo che vada a salvare le credenziali di accesso dopo il primo utilizzo.

Vale il comando:

```
git config --global credential.helper store
```

## Lavorare sul repository (1)

Per poter interagire con la repository GitHub, il modo migliore è utilizzare Git a livello locale.

GitHub nella pagina di presentazione del repository vuoto suggerisce infatti proprio alcuni comandi per integrare Git con GitHub.

Esistono due opzioni:

- Comando: `git clone [...]`;
- Comando: `git remote add [...]`.

## Lavorare sul repository (2)

Il comando `git clone` viene usato per scaricare dal server remoto un repository (vuoto o meno che sia) in modo che sia già configurato per sincronizzare le modifiche con il repository remoto su GitHub.

La sintassi è:

```
git clone [link]
```

## Lavorare sul repository (3)

In alternativa, è possibile configurare un repository (locale) esistente per sincronizzarsi con un repository remoto.

Questo è utile soprattutto quando la sincronizzazione su GitHub inizia ad uno stadio avanzato dello sviluppo del progetto.

La sintassi del comando `git remote add` è:

```
git remote add origin [link]
```

La parola `origin` rappresenta il server remoto (vedasi poi `git log`) e `[link]` è il link al repository remoto.

## Lavorare sul repository (4)

Le operazioni su Git sono sempre le stesse: add, commit, checkout, ecc.

Ma con la presenza ora di una nuova sorgente/destinazione remota è necessario sincronizzarsi e aggiornarsi con il server remoto.

In Git esistono due comandi:

- `git push`
- `git pull`

## "Pushare" sul repository (1)

Per inviare le modifiche svolte sul branch corrente (verificare con `git branch`) si usa il comando:

```
git push
```

In questo modo le modifiche andranno a sincronizzarsi con il repository remoto.

## "Pushare" sul repository (2)

Output atteso:

```
rory@ThinkBook-14: ~/Docker/Altalena/htdocs/~leode
rory@ThinkBook-14:~/Docker/Altalena/htdocs/~leode$ git push
Enumerazione degli oggetti in corso: 5, fatto.
Conteggio degli oggetti in corso: 100% (5/5), fatto.
Compressione delta in corso, uso fino a 16 thread
Compressione oggetti in corso: 100% (3/3), fatto.
Scrittura degli oggetti in corso: 100% (3/3), 321 byte | 321.00 KiB/s, fatto.
3 oggetti totali (2 delta), 0 riutilizzati (0 delta), 0 riutilizzati nel file pack
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/leonardodeirossi/leonardodeirossi.github.io
  01f931e..f97efd9  master -> master
```

## "Pullare" dal repository (1)

Per scaricare le modifiche fatte sul server remoto (magari da altri utenti), invece, esiste il comando `git pull`:

```
rory@ThinkBook-14: ~/Docker/Altalena/htdocs/~leode
rory@ThinkBook-14:~/Docker/Altalena/htdocs/~leode$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
Decompressione degli oggetti in corso: 100% (3/3), 960 byte | 480.00 KiB/s, fatto.
Da https://github.com/leonardodeirossi/leonardodeirossi.github.io
   f97efd9..309a816  master      -> origin/master
Aggiornamento di f97efd9..309a816
Fast-forward
 index.html | 2 --
1 file changed, 2 deletions(-)
```



