GitLab

Since July 2016 we use GitLab to collaborate.

UNIBZ users have to login to GitLab and their account will be automatically enabled a few minutes after they try their first login.

External users can sign up with their Google, Github, Twitter, Facebook, Bitbucket or Microsoft account.

Here are some instructions on GitLab to ease its use.

If you need support on old repositories created via SVN please continue reading here.

Initial setup

Git is usually preinstalled in Linux and Mac, while in Windows you have to install a client.

You can have different hosts for different Git repositories which you access via specific username/e-mail. You have to configure your local Git with these settings.

You can either clone/push/pull via HTTPS or SSH. If you chose to use HTTPS, you'll have to enter your credentials every time, if you choose to use SSH you only insert the password of your SSH key. Official documentation

The steps for pushing and pulling via SSH Key

- Step 1: Identify or create SSH Key - Official Documentation
- Step 2: Add SSH Key - Official Documentation
- Step 3: Configure your local Git client

Identify or create the SSH key

You need an SSH key that resides on your workstation/notebook.

Check for an existing SSH key

If you already have an SSH key in your computer you will probably find it here:

Linux/macOS : ~/.ssh/id_rsa.pub

Windows : %userprofile%\.ssh\id_rsa.pub

Open the file with a text editor and check if it has contents. Typically it starts with ssh-rsa or ssh-ed25519. If it has contents add it to Gitlab if it does not exist or is empty generate an SSH key.

Generate an SSH key

Open a terminal on Linux or macOS, or Git Bash on Windows and run the command:

```
ssh-keygen -t rsa -C "your@email" -b 4096
```

or

```
ssh-keygen -t ed25519 -C "your@email"
```

You can check your e-mail on your Gitlab Profile Settings → Main Settings → Email

When you are asked for a location and file name you can either keep the default location or choose a new one. If this is your only key we advice you accept the defaults and maintain the file name id_rsa. Should you have several keys you can name them according to your own preferences. GitLab Official Documentation

Add the key to your ssh agent by issuing this commands on Linux/macOS or Git Bash for Windows (adapt to your id_rsa path):

```
$ eval "$(ssh-agent -s)"
$ ssh-add -K ~/.ssh/id_rsa
```

Note: You can also use a key generator such as PuTTYgen instead of using the terminal. Please refer to the guidelines of the product on how to create the RSA key and the location of the file.

Add an SSH key

Login to Gitlab and go in Gitlab Profile Settings → SSH Keys.

In the field Key copy the contents of the file: id_rsa.pub (or any other file containing your key).

Choose a name to identify the key (ex. YourComputerName) and click Add Key, you will see the it listed.

Configure your local Git Client

Before proceeding check if your GIT client is already configured. In Linux/macOS terminal or Windows Git Bash, type the following:

```
git config --global --list
```

```
git config --list
```

This command lists any existing GIT repositories configurations. If you have an already existing configuration choose a local configuration (folder specific) for GIT. If you have no other GIT

configurations you can configure it globally.

Global configuration

To configure GIT globally using the SSH key inside the terminal/command line, type:

```
git config --global user.name "Firstname Lastname"
git config --global user.email "your@email"
git config --list
```

Add your server data in your ssh configuration file. Verify the path is correct

Linux/Mac file ~/.ssh/config:

```
host gitlab.inf.unibz.it
     HostName gitlab.inf.unibz.it
     user your_username
     IdentityFile ~/.ssh/id_rsa
```

Windows file ~/.ssh/config:

```
Host gitlab.inf.unibz.it
     RSAAuthentication yes
     IdentityFile ~/.ssh/config/id_rsa
```

The data on the global Git configuration is stored in the file (Linux/macOS) ~/.gitconfig or (Windows) C:\Users\username\.gitconfig

The content would be (if configured):

```
[user]
        name = Lastname Firstname
        email = your@email
```

Local Configuration

You have to run the commands inside the directory where you want your repository to be. From terminal/command line, type:

```
cd repository_folder_in_your_computer
git config user.name "Firstname Lastname"
git config user.email "your@email"
```

Should you receive this message **fatal: not in a git directory**, you have to initialize the folder as a git folder. Issue command :

```
git init
```

and run git commands again.

Basic Git

The logic of git requires you to checkout a project via CLONE. You ensure the project files are up to date by issuing a PULL command. Once the modifications on the project are done, you ADD the changes, you COMMIT them and you PUSH them to the gitlab server.

Create a new project via the web interface

Create a new project via the web interface. https://gitlab.inf.unibz.it/projects/new

Send the checkout URL to other collaborators. You can copy/paste it from the project settings (as SSH or as HTTPS).

When you checkout and you are asked for a password/username it means your configuration has to be edited. Do not insert your username/password.

Clone in new folder

```
git clone git@gitlab.inf.unibz.it:firstname-lastname/my-first-project.git
cd my-first-project
touch README.md
git add README.md
git commit -m "add README"
git push -u origin master
```

Clone in existing folder

```
cd existing_folder
git init
git remote add origin git@gitlab.inf.unibz.it:firstname-lastname/my-first-project.git
git add .
git commit
git push -u origin master
```

Go to the master branch to pull the latest changes from there

```
git checkout master
```

Download the latest changes in the project

This is for you to work on an up-to-date copy (it is important to do every time you work on a project), while you setup tracking branches.

```
git pull REMOTE NAME-OF-BRANCH -u
```

(REMOTE: origin) (NAME-OF-BRANCH: could be “master” or an existing branch)

Create a branch

Spaces won't be recognized, so you need to use a hyphen or underscore.

```
git checkout -b NAME-OF-BRANCH
```

Work on a branch that has already been created

```
git checkout NAME-OF-BRANCH
```

View the changes you've made

It's important to be aware of what's happening and what's the status of your changes.

```
git status
```

Add changes to commit

You'll see your changes in red when you type “git status”.

```
git add CHANGES IN RED
git commit -m "DESCRIBE THE INTENTION OF THE COMMIT"
```

Send changes to gitlab.inf.unibz.it

```
git push REMOTE NAME-OF-BRANCH
```

An example:

```
git add .
git commit
git push -u origin master
```

Delete all changes in the Git repository, but leave unstaged things

```
git checkout .
```

Delete all changes in the Git repository, including untracked files

```
git clean -f
```

Merge created branch with master branch

You need to be in the created branch.

```
git checkout NAME-OF-BRANCH
git merge master
```

Some support cases

SourceTree failing to connect

- Delete entry in keychain for gitlab.inf.unibz.it
- Add key again to ssh client : ssh-add -K <key_path>