GitLab

Since July 2016 we use GitLab to collaborate.

UNIBZ users have to login to GitLab and their account will be automatically enabled.

External users can sign up with their Google, Twitter, or Facebook account. The external user has to motivate the request by sending an e-mail to cs-tech@lists.inf.unibz.it.

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, check the version

```
git --version
```

You have to instruct Git to use a specific username/e-mail combination for gitlab.inf.unibz.it or any other hosts your need. Git allows you to have a global configuration or local configurations. To check if you have any configurations saved:

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

If your global configuration is empty then you can create a global configuration, unless create a local configuration for the repository you need.

The steps for pushing and pulling via SSH Key

- Step 1: Identify or create SSH Key
- Step 2: Add SSH Key
- Step 3: Configure your local client (global/local configuration)

Global GIT configuration

Check if your GIT client is already configured: LINUX/MACOSX Inside a terminal or in Windows inside your GIT bash command line , type the following:

```
git config --list
```

If there is no account/content, configure GIT globally. Inside the terminal/command line, type:

```
git config --global user.name "Firstname Lastname"
git config --global user.email "your_email@unibz.it"
git config --list
```

The last command should show the data you entered and is your gitlab global configuration. This data is saved in the file ~/.gitconfig:

```
[user]
        name = Lastname Firstname
        email = your_email@unibz.it
```

Try to checkout repo. Should you experience trouble check the contents of file: .ssh/config. You can manually insert:

```
host gitlab.inf.unibz.it
     user your_username
```

Local Configuration

You have to do it in the root of the repository. From terminal/command line, type:

```
cd repository_folder
git config user.name "Firstname Lastname"
git config user.email "your_other@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 then configure again the user.name and user.email parameters.

Add an SSH key

If you need support on how to generate the SSH key follow our guide on How to add an SSH Key for GITLAB.

If you already have an SSH key login to GitLab and go to → Profile Settings → SSH Keys. In the field Key copy the contents of the file: id_rsa.pub (or any other file containing your key). Once you click Add Key, you will see the key listed in your SSH Keys.

Basic Git commands

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
```

From:
https://wiki.inf.unibz.it/ - **Engineering-Tech Wiki**

Permanent link:
https://wiki.inf.unibz.it/doku.php?id=auth:gitlab&rev=1497347210

Last update: **2019/01/16 10:03**