UNIVERSITÀ
DI TRENTO
Department of
Information Engineering and Computer Science

Software Development Tools

Edoardo Lamon

Software Development for Collaborative Robots

Academic Year 2025/26

Git

- Git is a distributed version control system that tracks changes in any set of computer files, usually used for coordinating work among programmers who are collaboratively developing source code during software development.

- It is supported by various cloud-based code hosts such as GitHub, GitLab, Bitbucket etc.

Git configuration

- To install Git:
```
sudo apt update; sudo apt install git
```
- Configure it:
```
git config --global user.name "Your Name“
git config --global user.email "youremail@domain.com"
git config --global init.defaultBranch main
```
- Initialize the folder you want to be a Git repo (e.g. the ROS WS):
```
cd ros2_ws/src; git init
```
- Link your Git repo to a remote server (that we’re calling origin):
```
git remote add origin <remote repo URL>
```

Git useful commands

- Add a file (all) files to the list of Git tracked files:
  `git add file1 (git add .)`
- Verify status:
  `git status`
- Commit a local change to the repo with **message**:
  `git commit -m “msg" file1 file2 (git commit -m “msg" -a)`
- Pushing local changes to the main branch of remote server origin:
  `git push origin main`
- Update the local branch using the remote branch main:
  `git pull origin main`

Git useful commands

- Temporary save edits on tracked files who have not been committed yet: `git stash`
- Reapply previously stashed changes: `git stash pop`
- View current branch (all branches): `git branch (git branch –a)`
- Switch to existing branch called develop: `git checkout develop`
- Create a new branch develop and switch to it: `git checkout –b develop`
- Merging locally code between branches, keeping all commit messages. In main: `git merge develop --no-ff`

Git useful GUI

Git GUI (recommended to add, commit and push)

- Install: `sudo apt update; sudo apt -y install git-gui`
- Run: `git gui`

Gitg (to visualize repository history, branches, commits etc.)

- Install: `sudo apt update; sudo apt -y install gitg`
- Run: `gitg`

Documentation

- To make sure your code can be used by your teammate, and in the future also by external users, it is a good practice to provide a **README** and **documentation**;
- The **README** should include:
  1. A description of the project;
  2. Instructions on how to install or start the program;
  3. A tutorial or example of how to use the program.
- The most practical way is to add specifically **formatted comments** to the code (using VS code extensions for example), as in the case of **Doxygen**, or markdown files in case of **Sphinx,** and use a parser to automatically generate the documentation. They can generate an on-line documentation browser (in HTML) and/or an off-line reference manual (in LaTex/PDF).

Continuous Integration and Deployment (CI/CD)

- The process of running the test can be automated!
- Continuous integration is about automatically **building** and **testing** every change a developer makes to an application (and generating an updated documentation).
- The CI concept was first introduced over two decades ago to avoid *"integration hell"*, which happens when integration is put off till the end of a project.
- In practice, a list of instruction (YAML file) will be run on host (runner) to prepare the environment and build/test/generate the documentation. The runner has to be configured in the repo/project/organization.
- Some external tools are available, such as Travis CI and Jenkins.
- Recently also cloud-based code hosts (GitHub, GitLab) provide a fully integrated CI/CD pipeline, e.g. Github Actions.
- ROS 2 provides a template for writing the YAML file with some predefined actions.

Useful Links

- CMake build system:
  https://docs.ros.org/en/humble/How-To-Guides/Ament-CMake-Documentation.html
  https://colcon.readthedocs.io/en/released/index.html
- Code style:
  https://docs.ros.org/en/humble/The-ROS2-Project/Contributing/Code-Style-Language-Versions.html
- Documentation, testing:
  https://docs.ros.org/en/humble/The-ROS2-Project/Contributing/Developer-Guide.html#documentation