

# Version control system (VCS)

- Remember that you are required to **keep** a process-log-book of the whole development
  - solutions with just one commit or with **incomplete** process-log-book (where it is not possible to follow the progress of the project) will not be accepted
  - it does not matter that mistakes are done while developing. What matters is that we can **track** how you solved the problems
- A VCS is ideal to maintain a process-log-book

# Version control system (VCS)

- Currently there many VCSs available: svn, git, mercurial, cvs, ...
  - for the class project (for easing the supervision) you are required to use git and the services provided by bitbucket
  - you are also required to use only **private** repositories
- In this guide we will use git as well
  - we will also use bitbucket which offers free (multiuser) private git repositories
  - there are many other free repositories services such as Google code, Sourceforge, etc.

# Git: basic documentation

- If you are new to git we recommend you to have a look to the following introductory material:
  - a very basic guide  
<http://rogerdudler.github.io/git-guide/>
  - a more elaborated tutorial for beginners:  
<https://www.atlassian.com/git/tutorials>
  - a good graphical tutorial:  
<http://marklodato.github.io/visual-git-guide/index-en.html>

# Git: creating the central repository

- Create a new repository in bitbucket for that you should:
  - sign up: <https://bitbucket.org/>
  - log in in your account and create new **private** repository called *fraction\_tutorial*
  - see next slide ...

# Git: creating the central repository

The screenshot shows the Bitbucket 'Create a new repository' page. A red arrow points to the 'Create' button in the top navigation bar. Another red arrow points to the 'Create repository' button at the bottom. Red circles highlight the 'Name' field (containing 'fraction\_tutorial'), the 'Access level' checkbox (checked, labeled 'This is a private repository'), the 'Repository type' radio button (selected, labeled 'Git'), and the 'Language' dropdown menu (showing 'C++').

Bitbucket Dashboard Teams Repositories **Create** Find a repository

## Create a new repository

You can

**Name\*** fraction\_tutorial

Description Code for the tutorial of the VDS class project

Access level ☒ This is a private repository

Forking Allow only private forks

Repository type ☒ Git ☐ Mercurial

Project management ☐ Issue tracking ☐ Wiki

Language C++

**Repository integrations**

HipChat ☐ Enable HipChat notifications

**Create repository** Cancel

**New to Bitbucket**  
Learn the basics of using Git and Mercurial by exploring the Bitbucket 101.

**Working in a team**  
Create a team account to consolidate your repository and organize your team's work.

# Git tips (1)

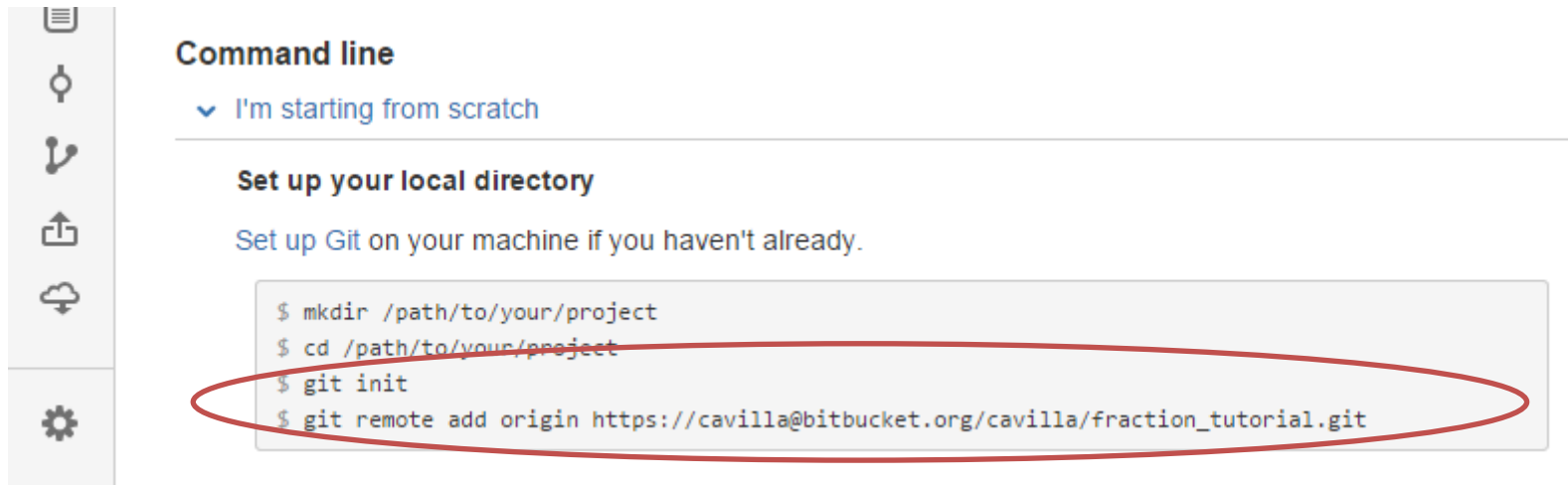
- Try to save in the git repositories only text files: .cpp, .hpp, scons scripts, doxygen configuration files, makefiles, ...
  - do not save binary files, object files (\*.o), executables and libraries
- Before starting configure git in your account (set user name, email, text editor, ...). For this follow the steps presented in: <https://www.atlassian.com/git/tutorials/setting-up-a-repository/git-config>

# Git: creating working directory

- Now that you have created the central repository it is time to create your working directory.
  - go to the directory *work/fraction\_tutorial* which is available on your account. On the linux console type:  
*cd work/fraction\_tutorial*
  - This directory has a template of the code (c++ code, scons scripts, makefiles and doxygen template)
  - Note the directory structure:
    - /doc* doxygen documentation
    - /src* source files
    - /src/unittest* code with unit tests
    - /test* code to run the tests

# Git: creating the working directory

- Use the indications provided by bitbucket in order to connect the local working directory with the central repository



The screenshot shows the 'Command line' section of a Bitbucket interface. On the left is a vertical sidebar with icons for repository, commit, pull request, upload, cloud, and settings. The main content area is titled 'Command line' and includes a dropdown menu 'I'm starting from scratch'. Below this is the section 'Set up your local directory' with the instruction 'Set up Git on your machine if you haven't already.' A code block contains the following commands: 

```
$ mkdir /path/to/your/project  
$ cd /path/to/your/project  
$ git init  
$ git remote add origin https://cavilla@bitbucket.org/cavilla/fraction_tutorial.git
```

 The last two lines of the code block are circled in red.

**Command line**

▼ I'm starting from scratch

**Set up your local directory**

Set up Git on your machine if you haven't already.

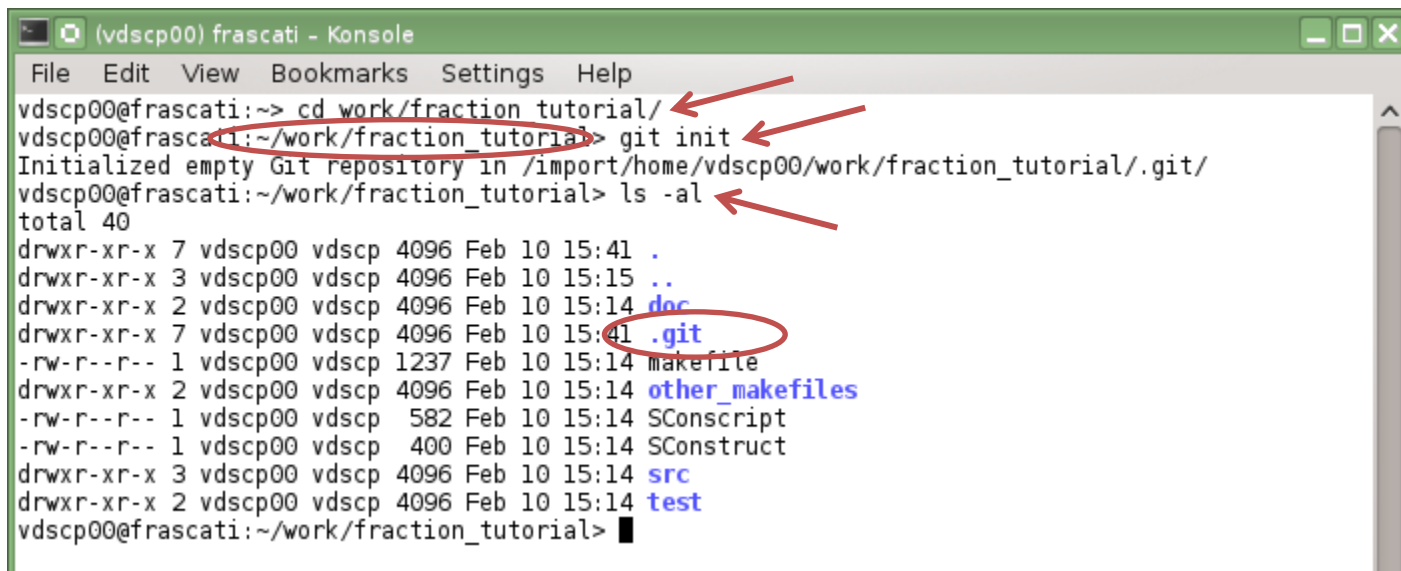
```
$ mkdir /path/to/your/project  
$ cd /path/to/your/project  
$ git init  
$ git remote add origin https://cavilla@bitbucket.org/cavilla/fraction_tutorial.git
```

- Next slide gives more details ...



# Git: creating working directory

- On the console type (remember you should be in the *fraction\_tutorial* directory): *git init*
- This creates a new local repository (so from now we have the central and local repositories). If you type on the console *ls -al* you should be able to see the new hidden *.git* folder



```
(vdscp00) frascati - Konsole
File Edit View Bookmarks Settings Help
vdscp00@frascati:~> cd work/fraction_tutorial/
vdscp00@frascati:~/work/fraction_tutorial> git init
Initialized empty Git repository in /import/home/vdscp00/work/fraction_tutorial/.git/
vdscp00@frascati:~/work/fraction_tutorial> ls -al
total 40
drwxr-xr-x 7 vdscp00 vdscp 4096 Feb 10 15:41 .
drwxr-xr-x 3 vdscp00 vdscp 4096 Feb 10 15:15 ..
drwxr-xr-x 2 vdscp00 vdscp 4096 Feb 10 15:14 doc
drwxr-xr-x 7 vdscp00 vdscp 4096 Feb 10 15:41 .git
-rw-r--r-- 1 vdscp00 vdscp 1237 Feb 10 15:14 makefile
drwxr-xr-x 2 vdscp00 vdscp 4096 Feb 10 15:14 other_makefiles
-rw-r--r-- 1 vdscp00 vdscp 582 Feb 10 15:14 SConscript
-rw-r--r-- 1 vdscp00 vdscp 400 Feb 10 15:14 SConstruct
drwxr-xr-x 3 vdscp00 vdscp 4096 Feb 10 15:14 src
drwxr-xr-x 2 vdscp00 vdscp 4096 Feb 10 15:14 test
vdscp00@frascati:~/work/fraction_tutorial>
```

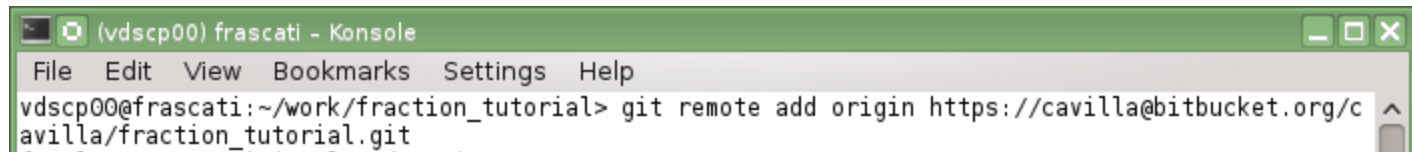
The screenshot shows a terminal window titled "(vdscp00) frascati - Konsole". The user navigates to the directory `~/work/fraction_tutorial/` and runs `git init`. The output indicates that an empty Git repository has been initialized in `~/work/fraction_tutorial/.git/`. Subsequently, the user runs `ls -al`, and the output lists the directory contents, including the newly created hidden `.git` folder. Red arrows and circles highlight the directory path, the `git init` command, and the `.git` folder in the `ls -al` output.

# Git: creating working directory

- It is time to connect the local repository with the central one, for that type:

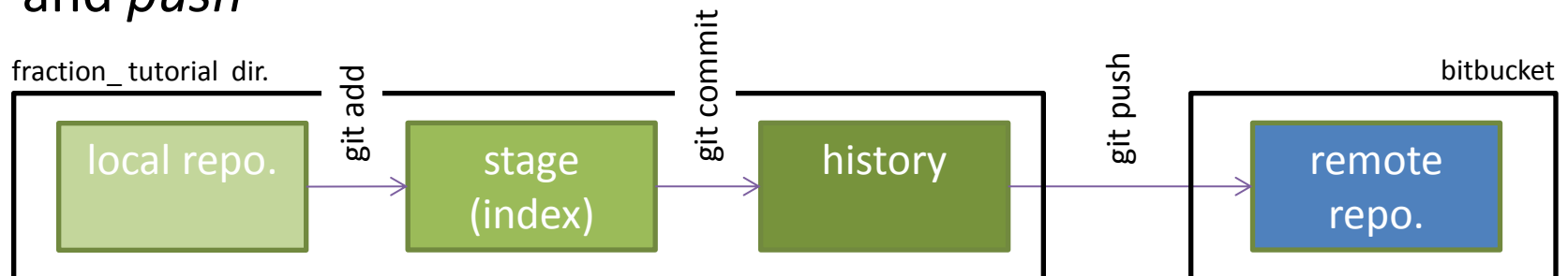
*git remote add origin https://username@bitbucket.org/username/fraction\_tutorial.git*

- adjust *username* accordingly



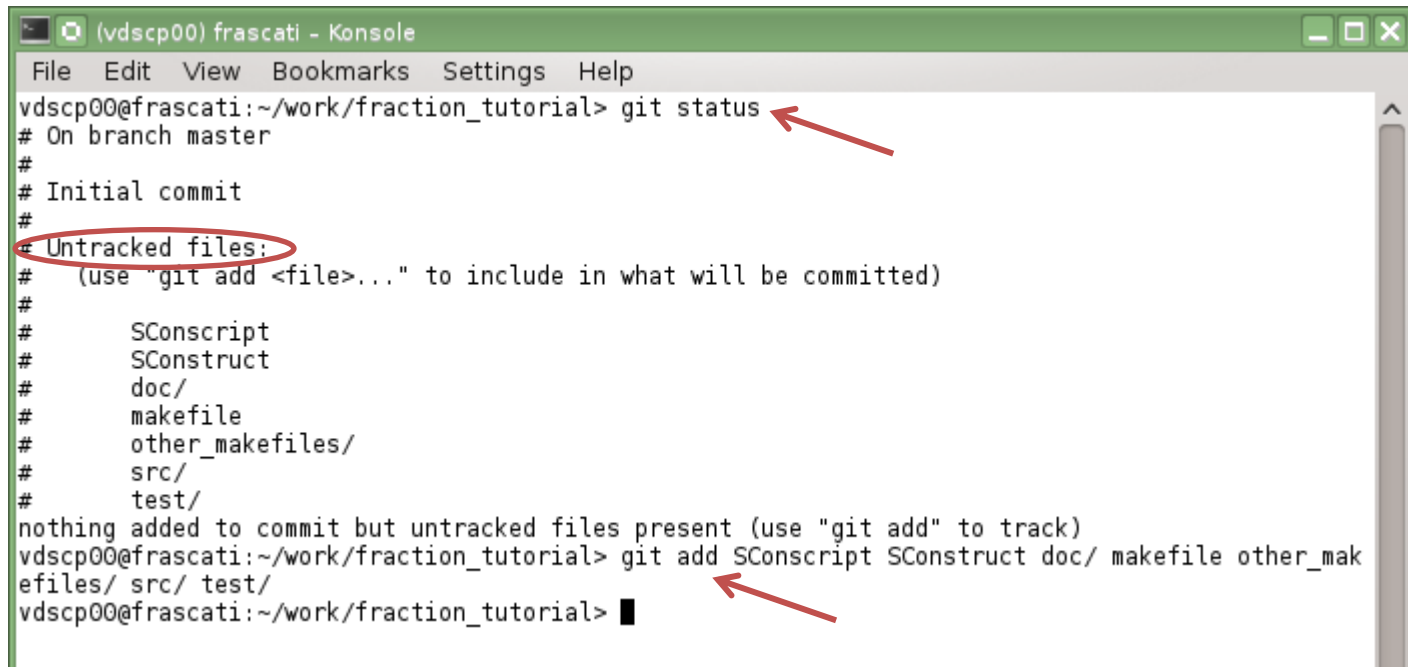
```
(vdscp00) frascati - Konsole
File Edit View Bookmarks Settings Help
vdscp00@frascati:~/work/fraction_tutorial> git remote add origin https://cavilla@bitbucket.org/cavilla/fraction_tutorial.git
```

- Since the current code in the local repository does not exist in the central one (type *git status* to check this) we need to save it
- This can be done by following three steps in git: *add*, *commit* and *push*



# Git: saving into the central repo.

- Add files and folders to the stage using *git add*



The screenshot shows a terminal window titled "(vdscp00) frascati - Konsole". The terminal output is as follows:

```
vdscp00@frascati:~/work/fraction_tutorial> git status
# On branch master
#
# Initial commit
#
# Untracked files:
#   (use "git add <file>..." to include in what will be committed)
#
#       SConscript
#       SConstruct
#       doc/
#       makefile
#       other_makefiles/
#       src/
#       test/
nothing added to commit but untracked files present (use "git add" to track)
vdscp00@frascati:~/work/fraction_tutorial> git add SConscript SConstruct doc/ makefile other_mak
efiles/ src/ test/
vdscp00@frascati:~/work/fraction_tutorial> █
```

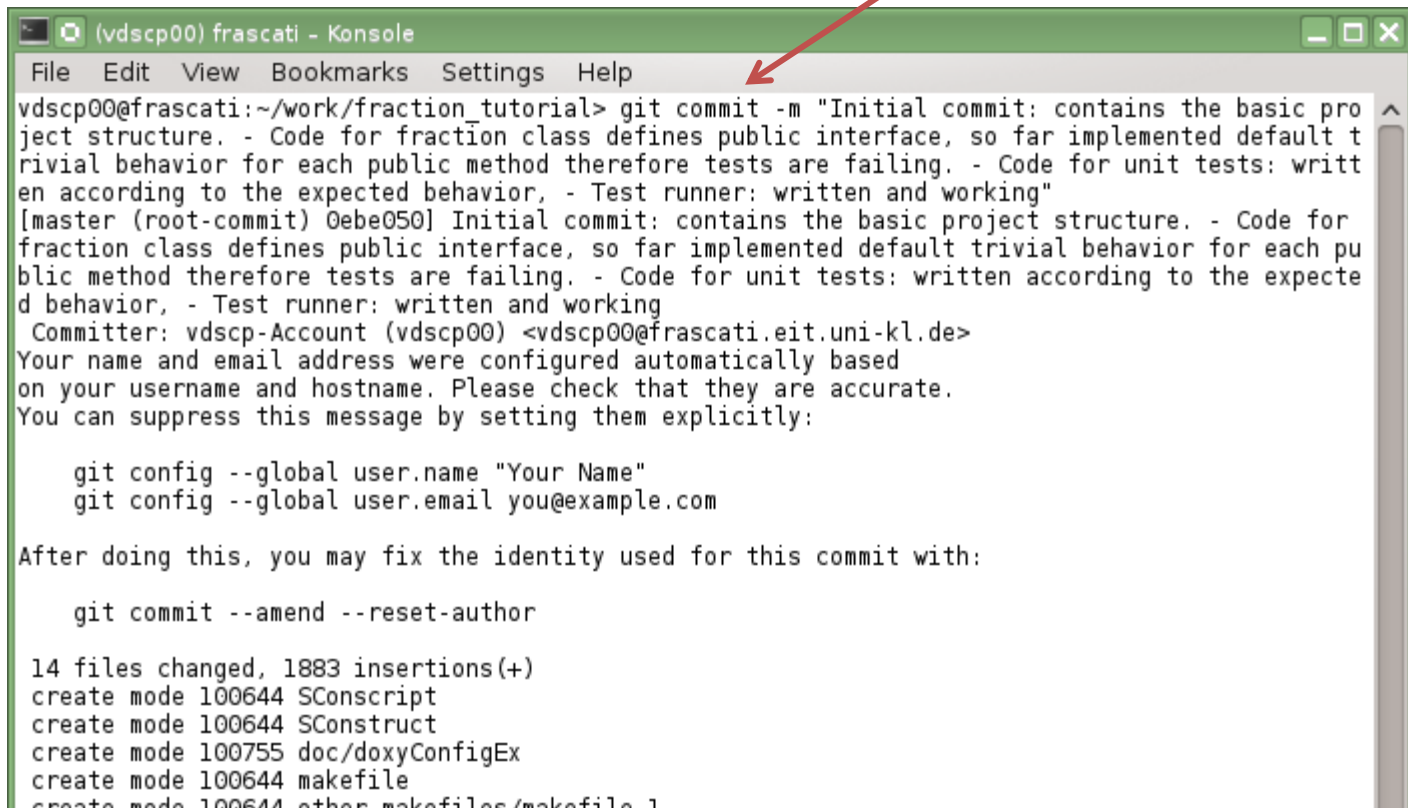
Two red arrows are present: one pointing to the `git status` command and another pointing to the `git add` command.

- If you type again *git status* you will notice that the files we added are marked as new and that they are ready to be committed
- Note that the new files are in the stage but not in the history

# Git: saving into the central repo.

- In order to commit into the history type:

*git commit -m "meaningful message so that we can track and assess what you did"*



```
(vdscp00) frascati - Konsole
File Edit View Bookmarks Settings Help
vdscp00@frascati:~/work/fraction_tutorial> git commit -m "Initial commit: contains the basic project structure. - Code for fraction class defines public interface, so far implemented default trivial behavior for each public method therefore tests are failing. - Code for unit tests: written according to the expected behavior, - Test runner: written and working"
[master (root-commit) 0e0e050] Initial commit: contains the basic project structure. - Code for fraction class defines public interface, so far implemented default trivial behavior for each public method therefore tests are failing. - Code for unit tests: written according to the expected behavior, - Test runner: written and working
Committer: vdscp-Account (vdscp00) <vdscp00@frascati.eit.uni-kl.de>
Your name and email address were configured automatically based on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:

    git config --global user.name "Your Name"
    git config --global user.email you@example.com

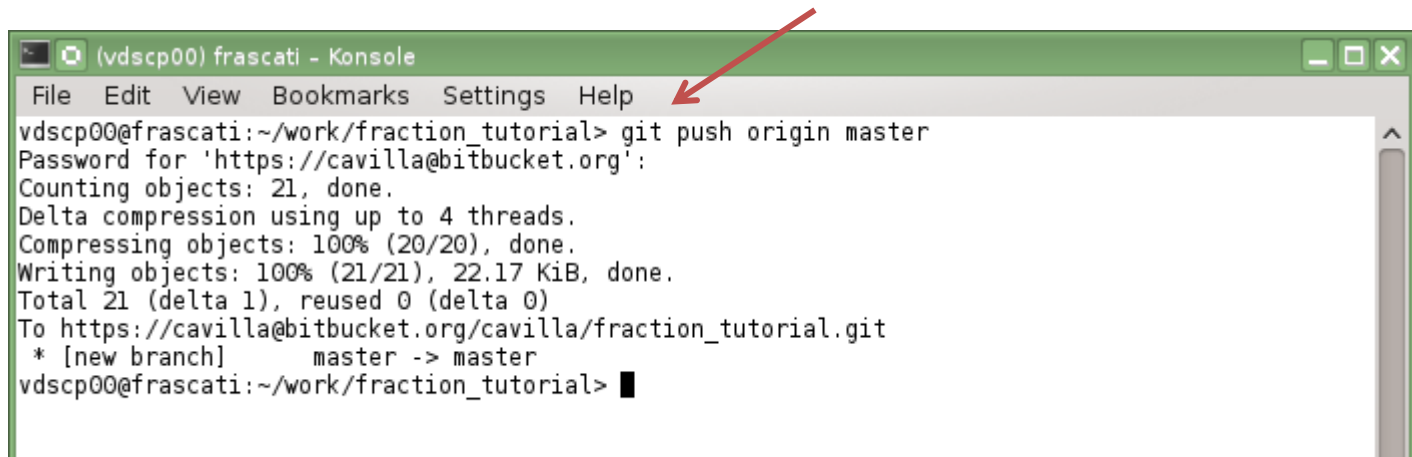
After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

14 files changed, 1883 insertions(+)
create mode 100644 SConscript
create mode 100644 SConstruct
create mode 100755 doc/doxyConfigEx
create mode 100644 makefile
create mode 100644 other/makefiles/makefile.1
```

# Git: saving into the central repo.

- Now the files are in the history and need to be pushed into the central repository
- Type: *git push origin master*
  - *origin* refers to the central repository and *master* to the current branch
  - when asked type the password of your bitbucket account

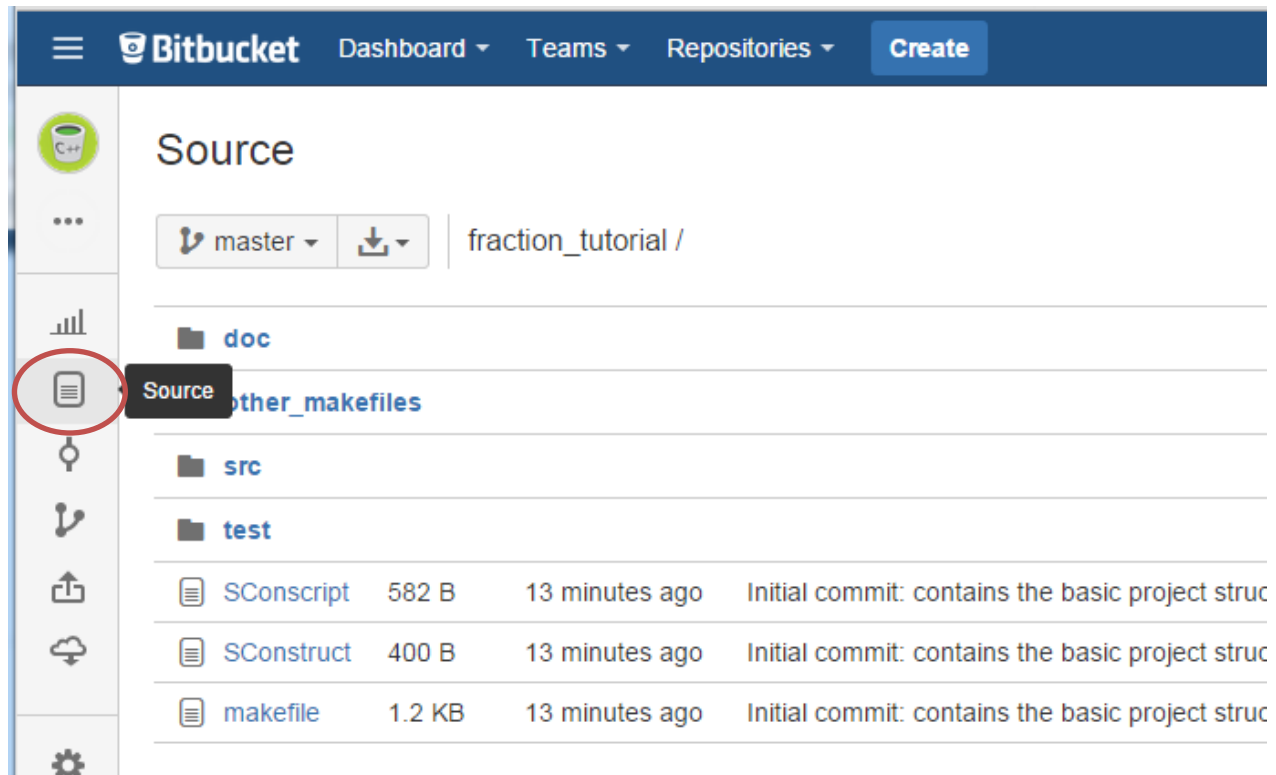


```
(vdscp00) frascati - Konsole
File Edit View Bookmarks Settings Help
vdscp00@frascati:~/work/fraction_tutorial> git push origin master
Password for 'https://cavilla@bitbucket.org':
Counting objects: 21, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (20/20), done.
Writing objects: 100% (21/21), 22.17 KiB, done.
Total 21 (delta 1), reused 0 (delta 0)
To https://cavilla@bitbucket.org/cavilla/fraction_tutorial.git
 * [new branch]      master -> master
vdscp00@frascati:~/work/fraction_tutorial>
```

A terminal window titled "(vdscp00) frascati - Konsole" with a menu bar (File, Edit, View, Bookmarks, Settings, Help) and standard window controls. A red arrow points to the "Help" menu item. The terminal shows the execution of the command `git push origin master`. It prompts for a password for the Bitbucket account `https://cavilla@bitbucket.org`. The output shows the progress of pushing 21 objects, including delta compression and writing to the repository. The final output indicates that the `master` branch has been successfully pushed to the central repository.

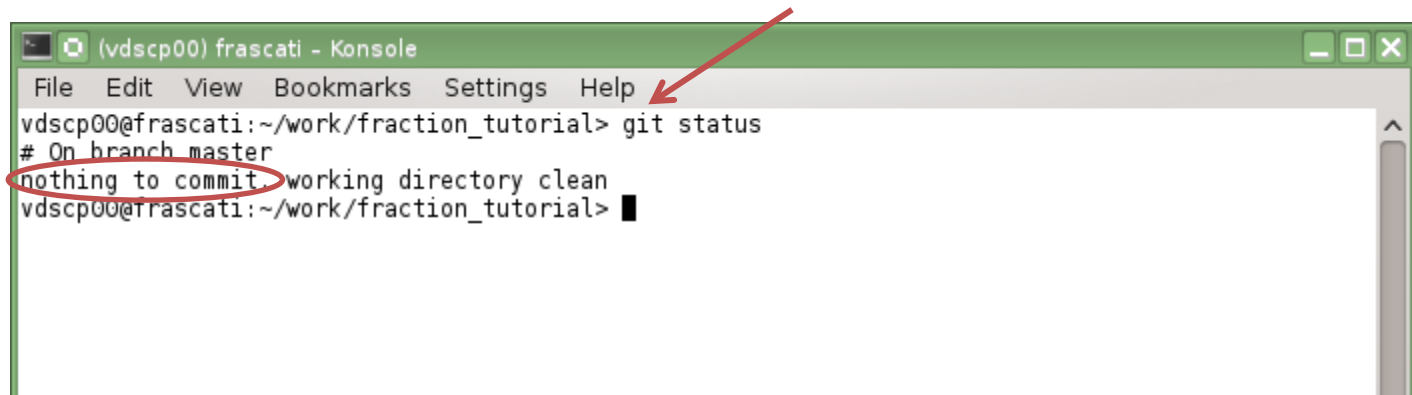
# Git: saving into the central repo.

- You may check that the changes done are now saved in the central repository:



# Git: saving into the central repo.

- Type again *git status* to check that now everything is up-to-date



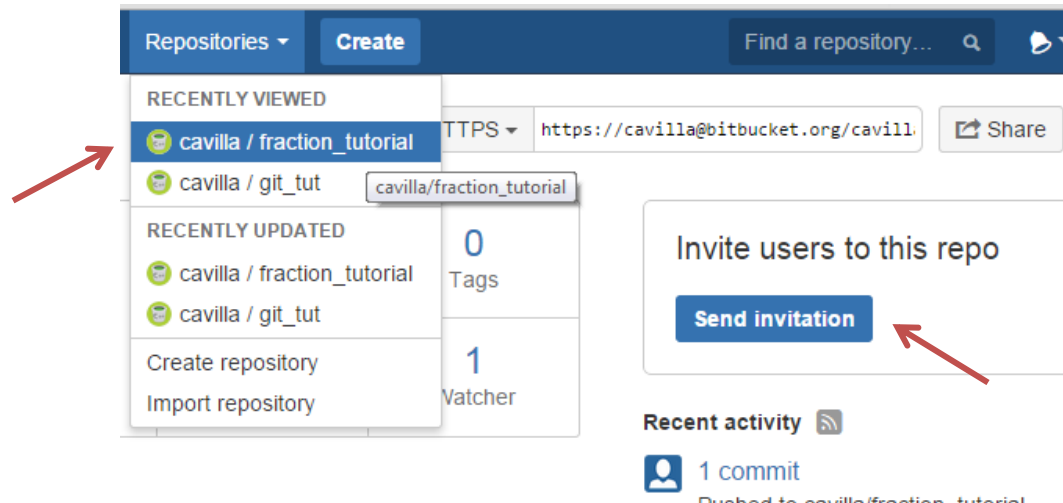
The screenshot shows a terminal window titled "(vdscp00) frascati - Konsole". The terminal has a menu bar with "File", "Edit", "View", "Bookmarks", "Settings", and "Help". A red arrow points to the "Help" menu. The terminal content shows the command "git status" being executed. The output is "# On branch master" followed by "nothing to commit, working directory clean". The phrase "nothing to commit" is circled in red. The prompt "vdscp00@frascati:~/work/fraction\_tutorial>" is visible at the end of the line.

```
(vdscp00) frascati - Konsole
File Edit View Bookmarks Settings Help
vdscp00@frascati:~/work/fraction_tutorial> git status
# On branch master
nothing to commit, working directory clean
vdscp00@frascati:~/work/fraction_tutorial>
```

- So far you are done with git stuff!
- Now lets have a look to what you just pushed ...

# Git tips (2)

- Here some extra tips:
- If you want to give access to a team member you can send an invitation in bitbucket



- Your team colleague can then use *git clone* (see tutorials) in order to work on the files
- To learn about git flows read this documentation:  
<https://www.atlassian.com/git/tutorials/comparing-workflows>