

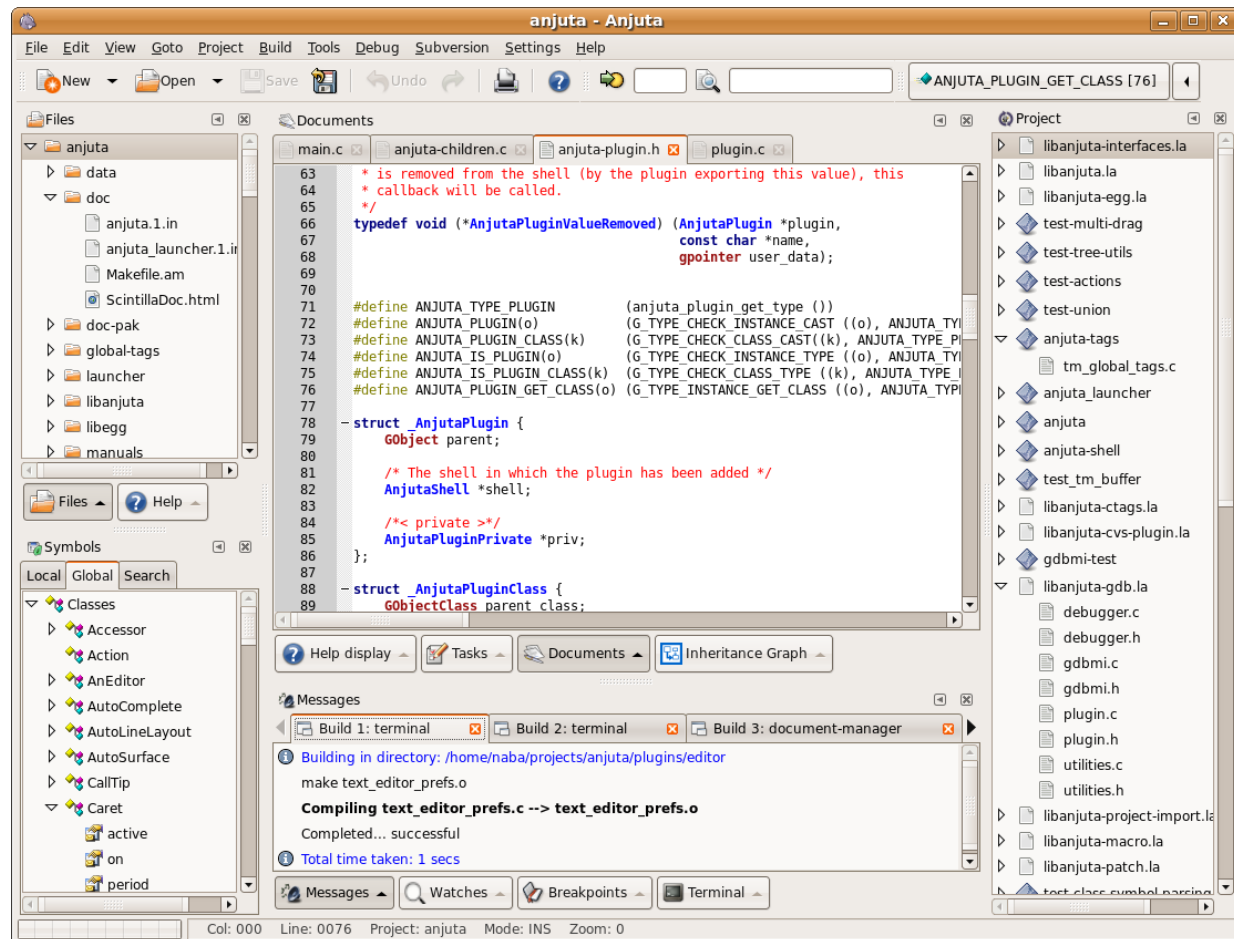


INTRODUZIONE ALL'USO DI AMBIENTI IDE: ANJUTA VERSIONING CON GIT

- ☐ IDE: Anjuta
- ☐ Debug mediante IDE
- ☐ Programmazione modulare mediante IDE
- ☐ Versioning: git

- Un ambiente di sviluppo integrato è un software che consente di
 - Editare un programma
 - Gestire più file appartenenti ad uno stesso progetto
 - Creare automaticamente i file di configurazione/build/deploy
 - Lanciare l'esecuzione di un programma
 - Avviare il debug di un programma consentendo
 - Di eseguire per passi il programma
 - Monitorare l'evoluzione di alcune variabili

- Poiché è l'ambiente installato in laboratorio
- E' leggero
- Integra g++ e gdb (debugger)



□ Da linea di comando:

```
sudo apt-get update
```

```
sudo apt-get install anjuta
```

- Nel caso in cui venisse successivamente segnalato un errore su libtool

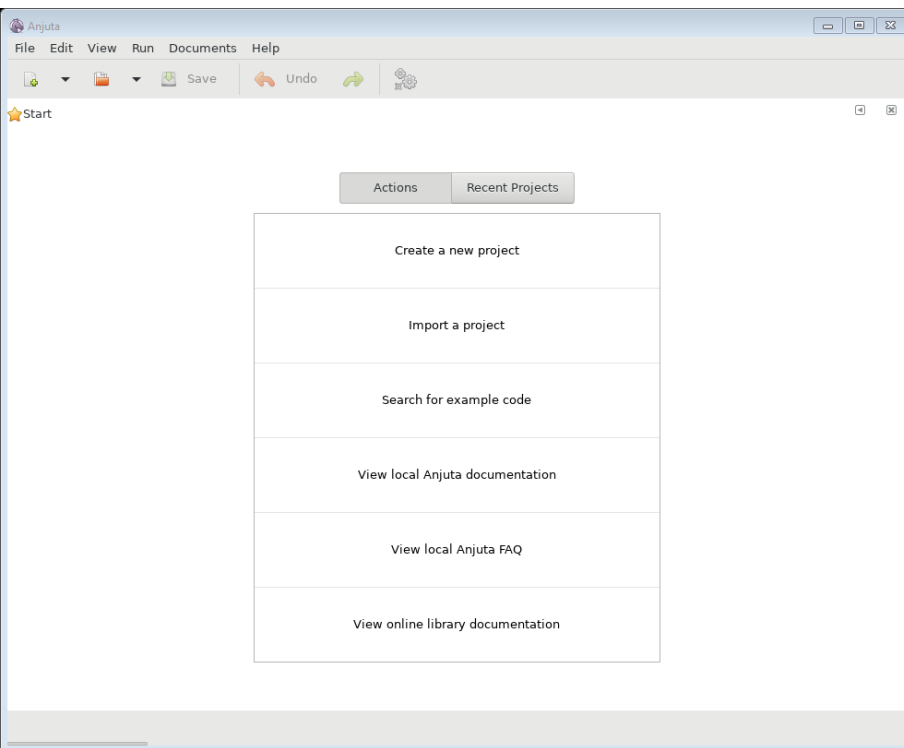
- Installare libtool e libtool-bin

```
sudo apt-get install libtool
```

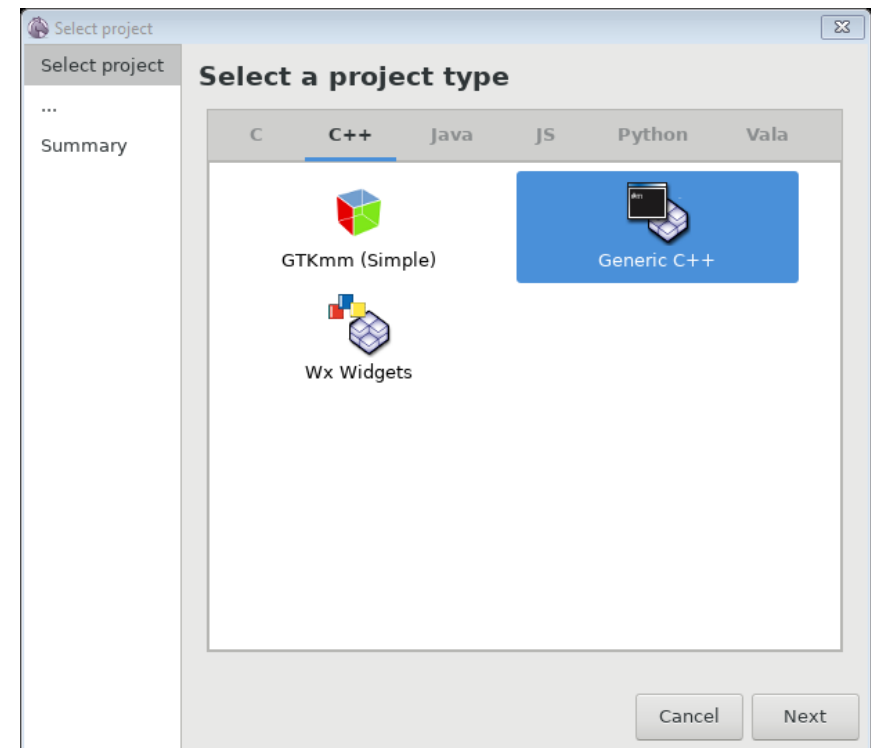
```
sudo apt-get install libtool-bin
```

- Lanciare anjuta

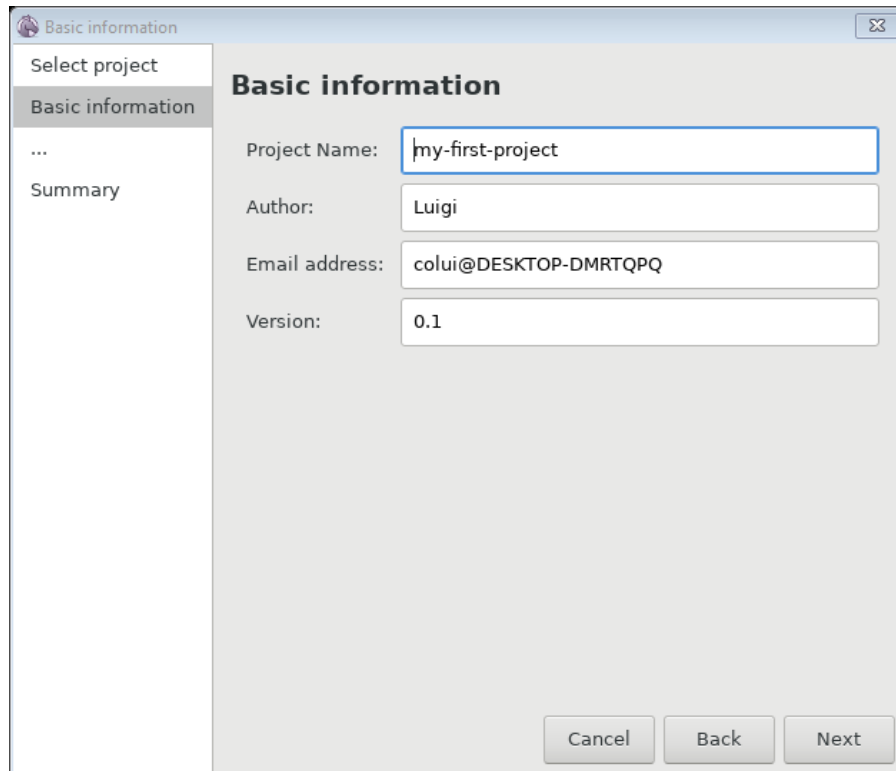
Create a new project



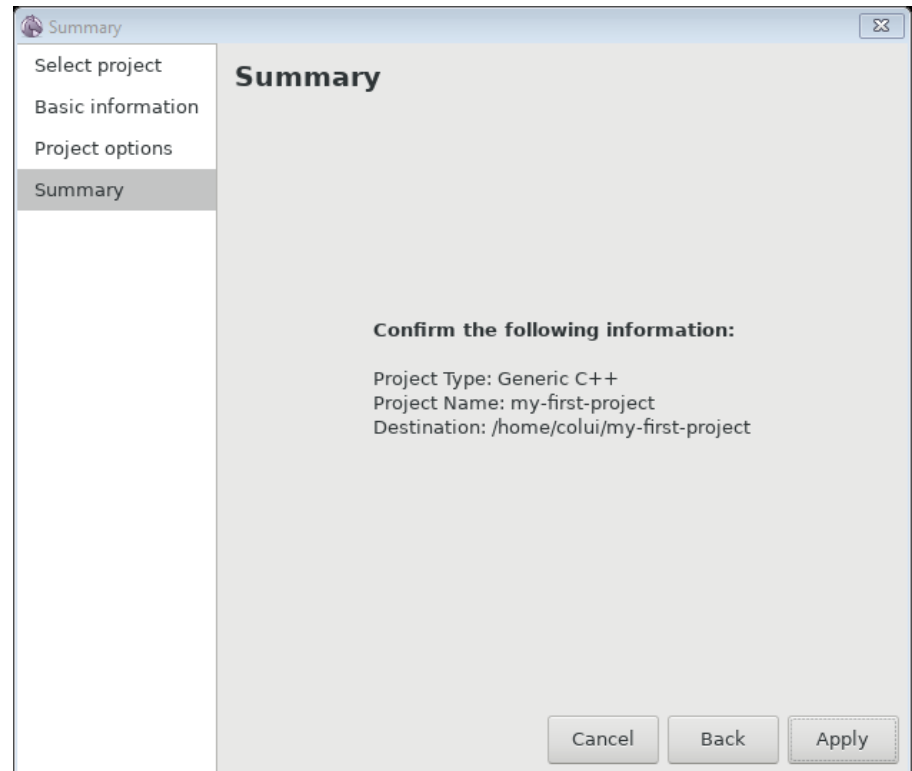
Quindi selezionare C++ e next



- Inserire il nome del progetto e andare avanti (next) fino alla fine

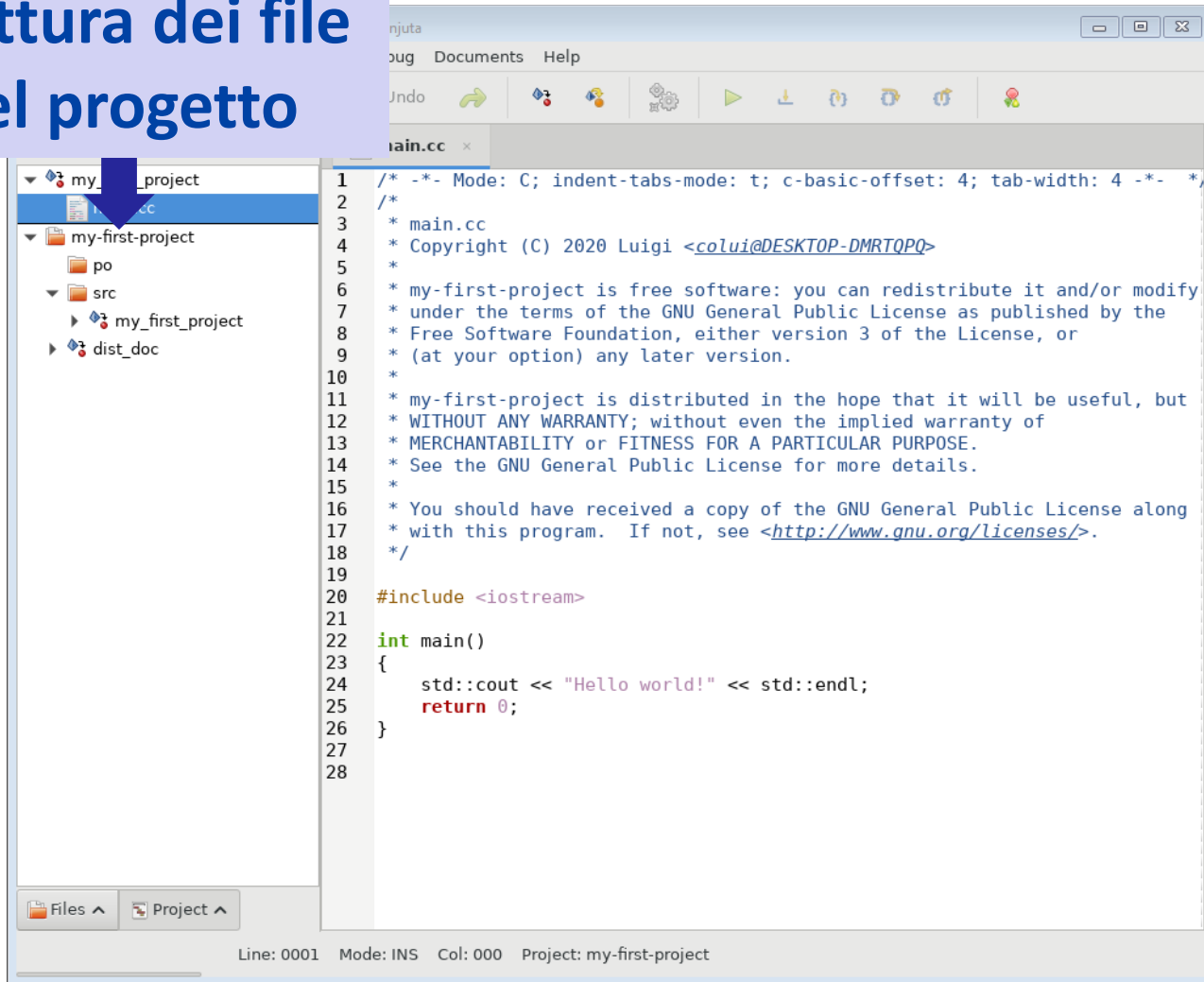


The 'Basic information' dialog box is shown with a sidebar on the left containing 'Select project', 'Basic information' (selected), '...', and 'Summary'. The main area is titled 'Basic information' and contains four text input fields: 'Project Name' with the value 'my-first-project', 'Author' with 'Luigi', 'Email address' with 'colui@DESKTOP-DMRTQPQ', and 'Version' with '0.1'. At the bottom right are 'Cancel', 'Back', and 'Next' buttons.

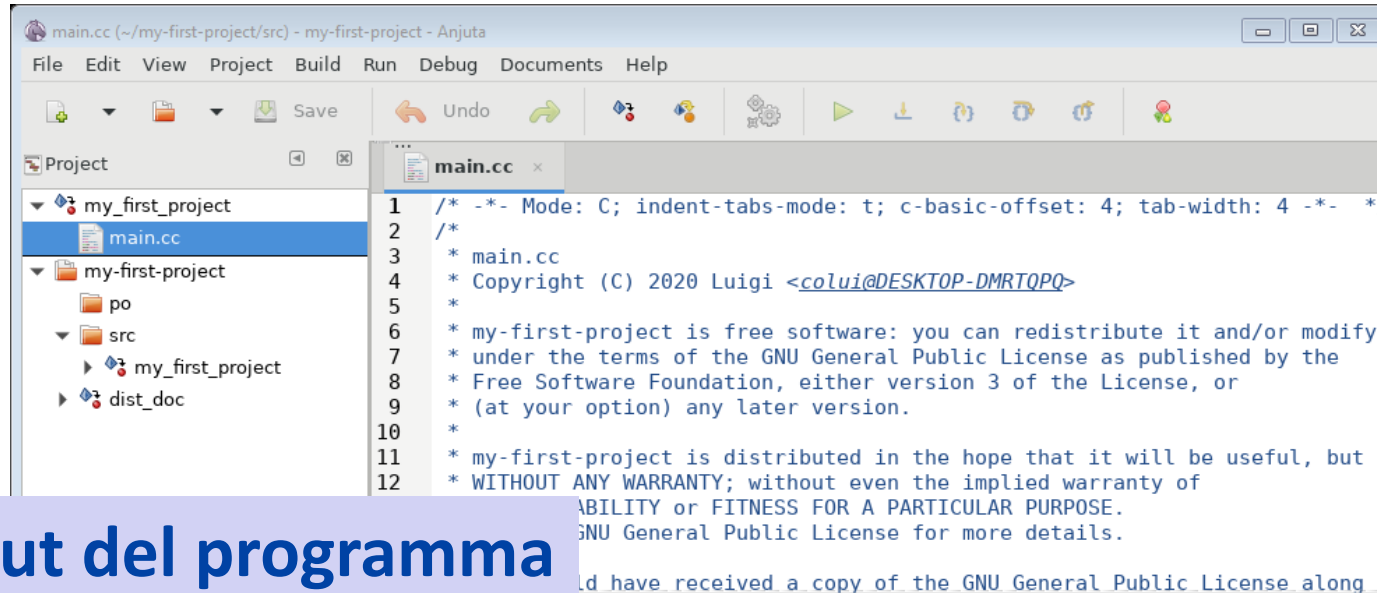


The 'Summary' dialog box is shown with a sidebar on the left containing 'Select project', 'Basic information', 'Project options', and 'Summary' (selected). The main area is titled 'Summary' and contains the text 'Confirm the following information:' followed by a list of project details: 'Project Type: Generic C++', 'Project Name: my-first-project', and 'Destination: /home/colui/my-first-project'. At the bottom right are 'Cancel', 'Back', and 'Apply' buttons.

Struttura dei file del progetto

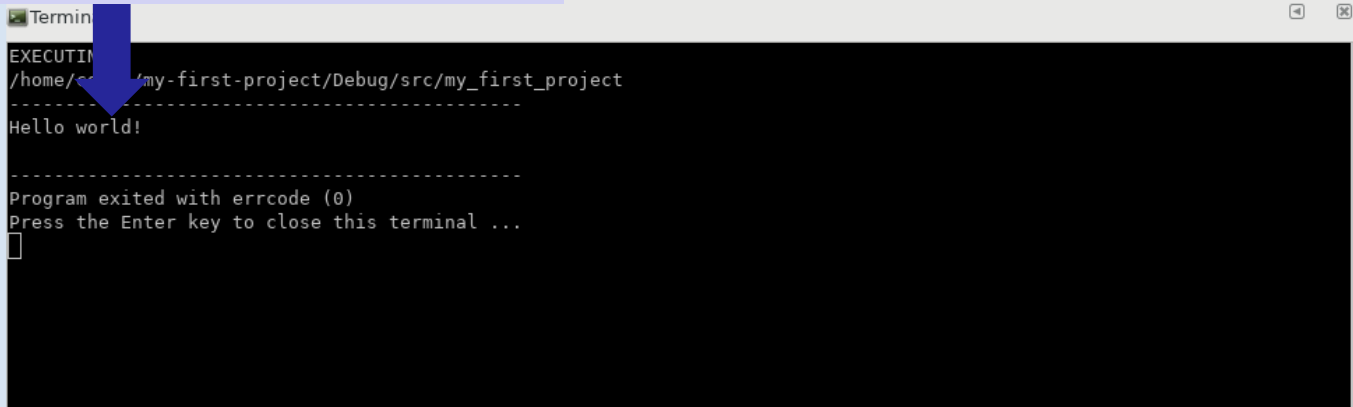


- Dal menu RUN selezioniamo EXECUTE



```
1  /* -*- Mode: C; indent-tabs-mode: t; c-basic-offset: 4; tab-width: 4 -*- */
2  /*
3   * main.cc
4   * Copyright (C) 2020 Luigi <colui@DESKTOP-DMRTQPQ>
5   *
6   * my-first-project is free software: you can redistribute it and/or modify
7   * under the terms of the GNU General Public License as published by the
8   * Free Software Foundation, either version 3 of the License, or
9   * (at your option) any later version.
10  *
11  * my-first-project is distributed in the hope that it will be useful, but
12  * WITHOUT ANY WARRANTY; without even the implied warranty of
   ABILITY or FITNESS FOR A PARTICULAR PURPOSE.
   GNU General Public License for more details.
   d have received a copy of the GNU General Public License along
```

Output del programma

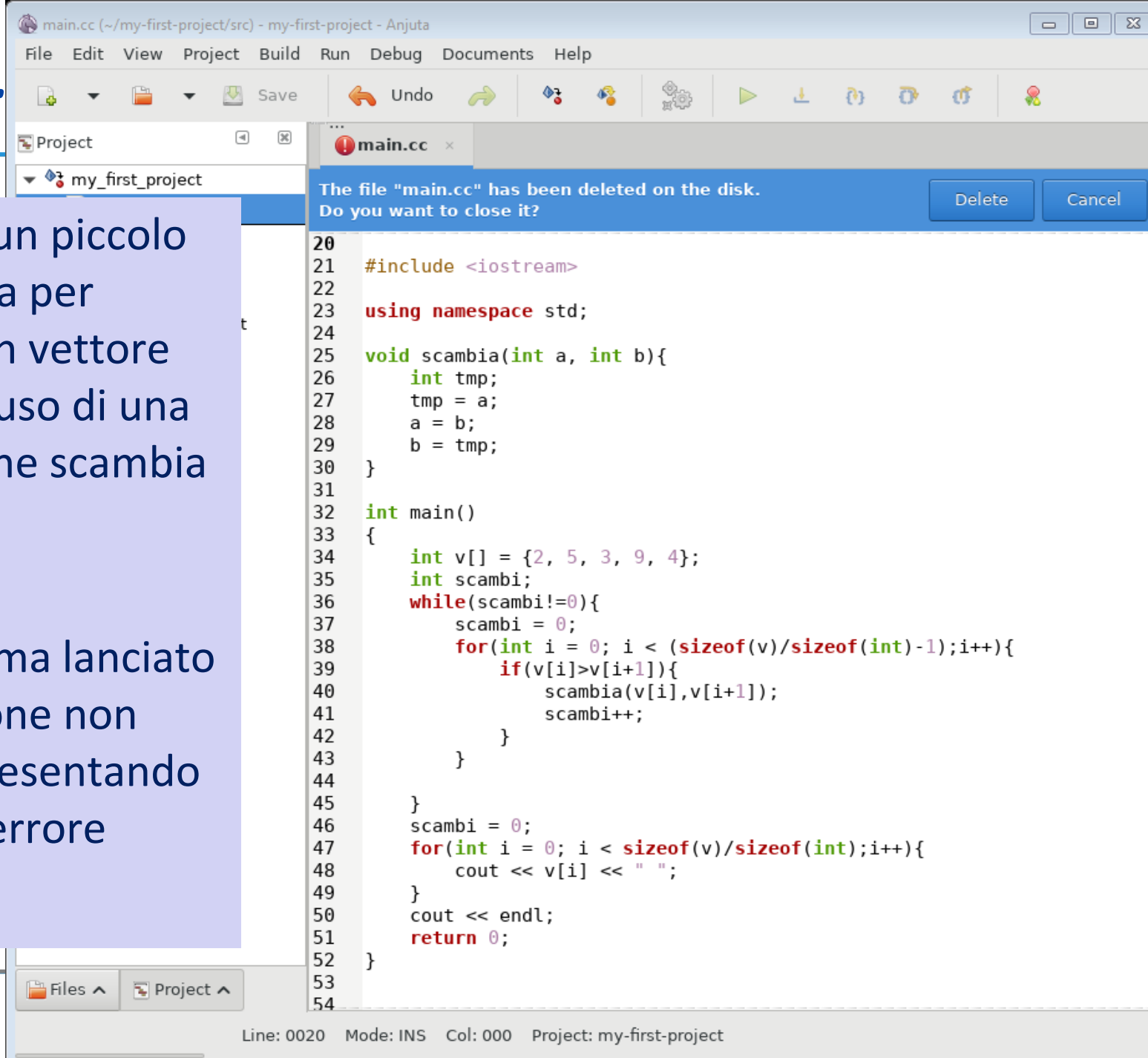


```
EXECUTING
/home/.../my-first-project/Debug/src/my_first_project
-----
Hello world!
-----
Program exited with errcode (0)
Press the Enter key to close this terminal ...
```

DEBUG ...

Scriviamo un piccolo programma per ordinare un vettore che faccia uso di una funzione che scambia due interi

Il programma lanciato in esecuzione non termina presentando quindi un errore

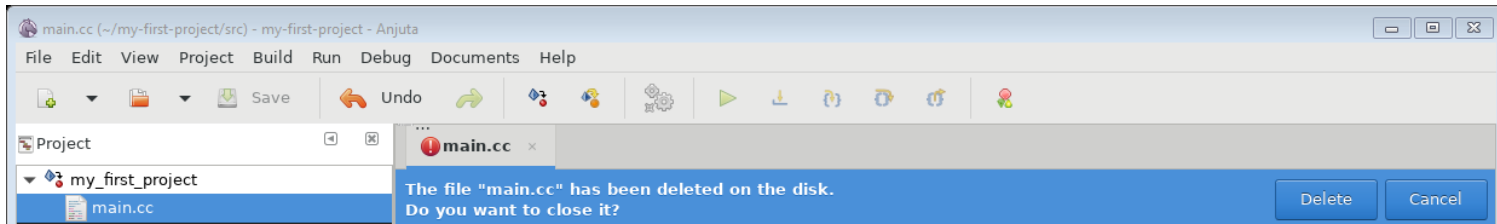


The screenshot shows an IDE window titled "main.cc (~my-first-project/src) - my-first-project - Anjuta". The menu bar includes File, Edit, View, Project, Build, Run, Debug, Documents, and Help. The toolbar contains icons for Save, Undo, and various development tools. The Project pane on the left shows "my_first_project". A blue error dialog box is displayed, stating: "The file 'main.cc' has been deleted on the disk. Do you want to close it?" with "Delete" and "Cancel" buttons. The main editor displays the following C++ code:

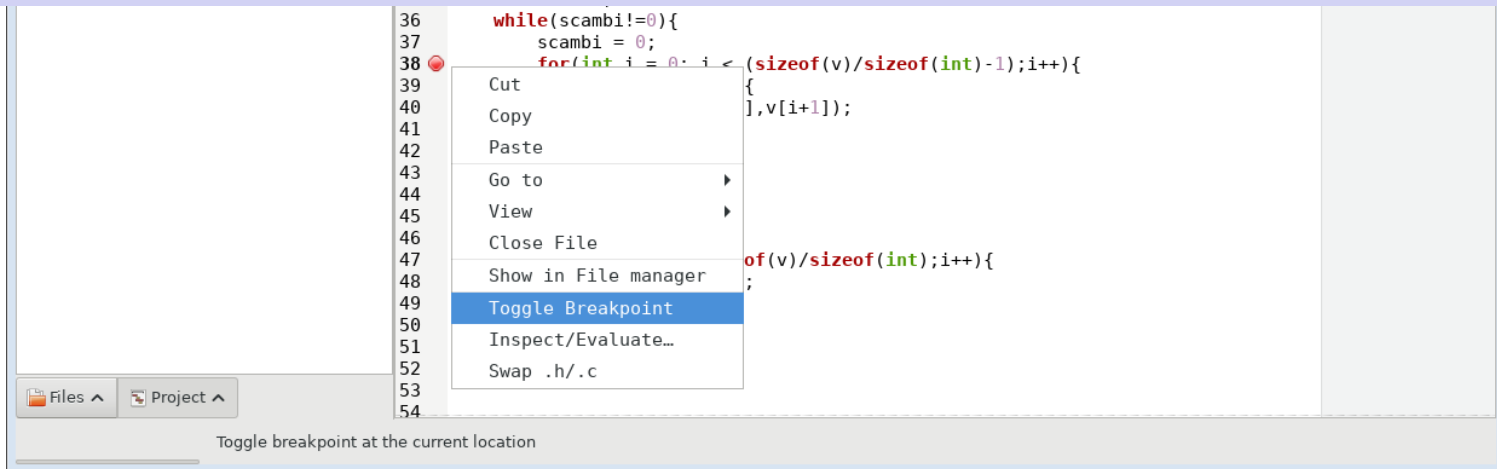
```
20
21 #include <iostream>
22
23 using namespace std;
24
25 void scambia(int a, int b){
26     int tmp;
27     tmp = a;
28     a = b;
29     b = tmp;
30 }
31
32 int main()
33 {
34     int v[] = {2, 5, 3, 9, 4};
35     int scambi;
36     while(scambi!=0){
37         scambi = 0;
38         for(int i = 0; i < (sizeof(v)/sizeof(int)-1);i++){
39             if(v[i]>v[i+1]){
40                 scambia(v[i],v[i+1]);
41                 scambi++;
42             }
43         }
44     }
45     scambi = 0;
46     for(int i = 0; i < sizeof(v)/sizeof(int);i++){
47         cout << v[i] << " ";
48     }
49     cout << endl;
50     return 0;
51 }
52
53
54
```

The status bar at the bottom indicates "Line: 0020 Mode: INS Col: 000 Project: my-first-project".





- 1) Aggiungiamo un punto di arresto in corrispondenza del for
- 2) Mandiamo in esecuzione in modalità debug (Run – Debug Program)



ISPEZIONARE IL VALORE DELLE VARIABILI

12

Eseguiamo passo passo (andando nelle funzioni –stepinto– o saltandole –stepover)

The screenshot shows a debugger interface with two main panels. The top panel, titled 'Registers', lists 16 registers (rax to r10) with their current values in hexadecimal. The bottom panel, titled 'Locals', shows the local variables of the current function. It includes a table with columns 'Variable', 'Value', and 'Type'. The variable 'i' has a value of 1 and type 'int'. The variable 'v' is an array of 5 integers, with values [2, 5, 3, 9, 4]. The variable 'scambi' has a value of 0 and type 'int'. A blue arrow points from the text 'E osserviamo come variano le variabili: Con i=1 v[1]=5 > v[2]=3 quindi c'è uno scambio' to the 'v' array in the Locals panel. Another blue arrow points from the text 'Eseguiamo passo passo (andando nelle funzioni –stepinto– o saltandole –stepover)' to the 'Step Over' button in the debugger's toolbar.

Register	Value
rax	0x0000000000000003
rbx	0x0000000000000000
rcx	0x00000000000000a0
rdx	0x0000000000000005
rsi	0x00007ffffffe3e8
rdi	0x0000000000000001
rbp	0x00007ffffffe300
rsp	0x00007ffffffe2d0
r8	0x00007ffff05cd80
r9	0x0000000000000000
r10	0x0000000000000006

Variable	Value	Type
i	1	int
v	[5]	int [5]
0	2	int
1	5	int
2	3	int
3	9	int
4	4	int
scambi	0	int

Line: 0040 Mode: INS Col: 000 Debugger: Stopped Project: my-first-project

- Entriamo nella funzione e osserviamo che i valori di a e b sono stati scambiati

The screenshot shows the Anjuta IDE interface. At the top, the title bar reads "main.cc (~my-first-project/src) - my-first-project - Anjuta". Below it is a menu bar (File, Edit, View, Project, Build, Run, Debug, Documents, Help) and a toolbar with icons for Save, Undo, and various debugging actions.

On the left, there is a "Registers" panel with a table of CPU registers and their values:

Register	Value
rax	0x0000000000000005
rbx	0x0000000000000000
rcx	0x00000000000000a0
rdx	0x0000000000000003
rsi	0x0000000000000003
rdi	0x0000000000000005
rbp	0x00007ffffffe2c0
rsp	0x00007ffffffe2c0
r8	0x00007ffff05cd80

Below the registers is a "Locals" panel with a table of local variables:

Variable	Value	Type
a	3	int
b	5	int

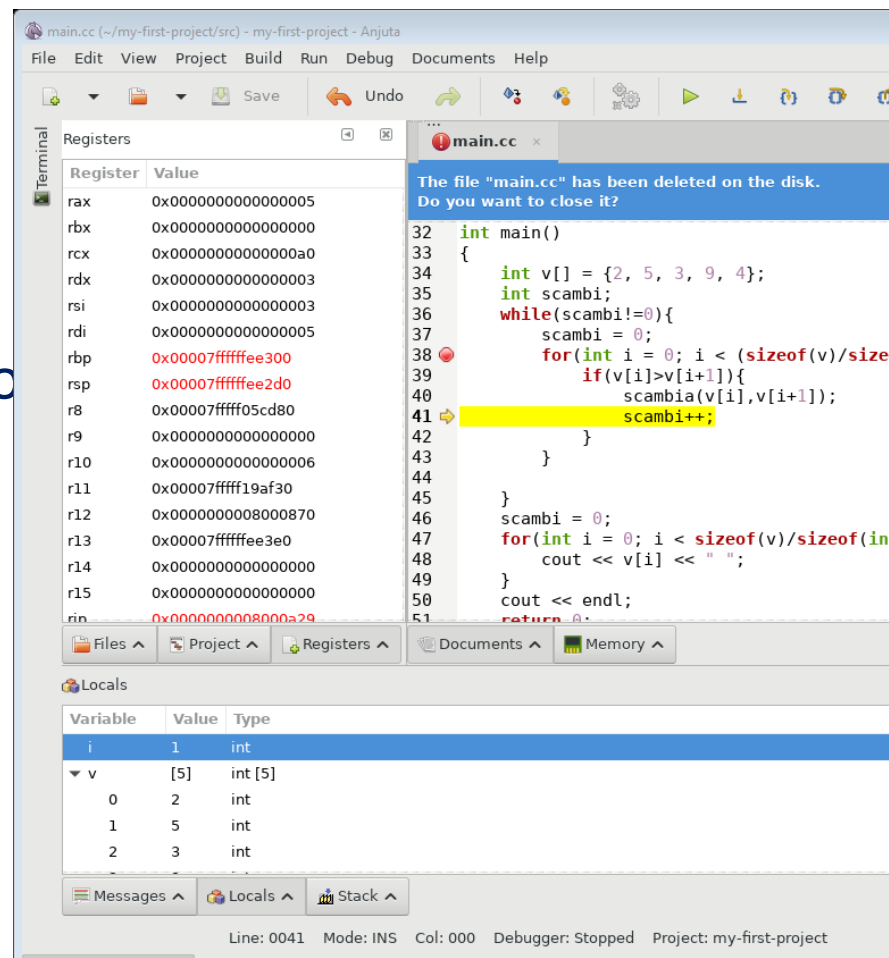
At the bottom, a status bar shows "Line: 0030 Mode: INS Col: 000 Debugger: Stopped Project: my-first-project".

In the center, a code editor shows the following C code:

```
24
25 void scambia(int a, int b){
26     int tmp;
27     tmp = a;
28     a = b;
29     b = tmp;
30 }
31
32 int main()
33 {
```

A blue error dialog box is overlaid on the code editor, stating: "The file 'main.cc' has been deleted on the disk. Do you want to close it?". It has "Delete" and "Cancel" buttons.

- ❑ Tuttavia i valori in v restano immutati => avremmo dovuto passare i valori per riferimento...
- ❑ Run-> Stop Debugger
- ❑ Togliamo il break point (menu Debug)
- ❑ Correggiamo la funzione indicando che i parametri sono passati per riferimento



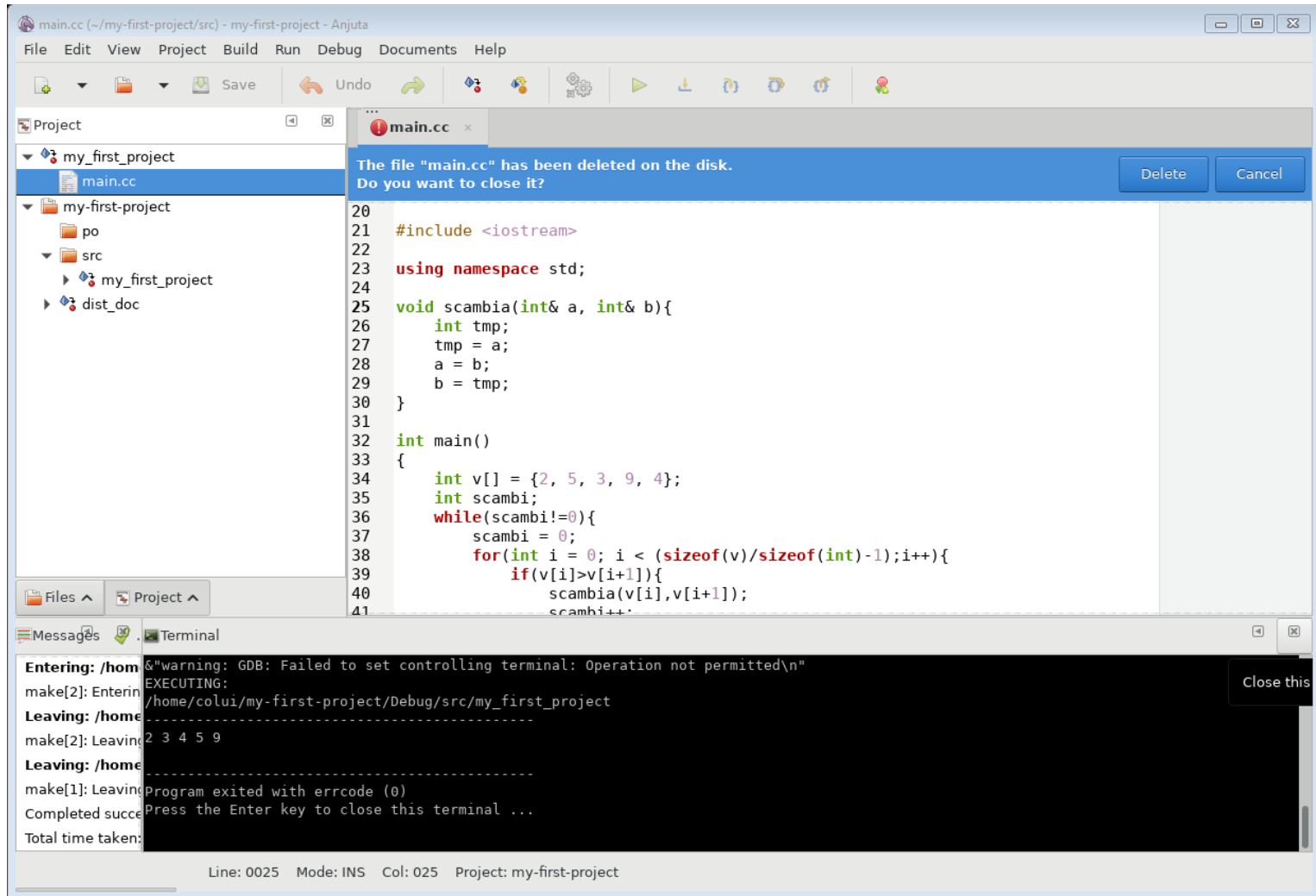
The screenshot shows the Anjuta IDE with a C++ program. The main window displays the source code of `main.cc`. The program defines a `swap` function and a `main` function. The `main` function initializes an array `v` with values {2, 5, 3, 9, 4} and a variable `scambi` to 0. It enters a `while` loop that calls `swap` and increments `scambi` until it reaches the size of `v`. A breakpoint is set at line 41, where `scambi++` is executed. The `Registers` window on the left shows the state of the CPU registers. The `Locals` window at the bottom shows the variables `i`, `v`, and `scambi`. The status bar at the bottom indicates the debugger is stopped at line 0041.

```
32 int main()
33 {
34     int v[] = {2, 5, 3, 9, 4};
35     int scambi;
36     while(scambi!=0){
37         scambi = 0;
38         for(int i = 0; i < (sizeof(v)/sizeof(int)); i++){
39             if(v[i]>v[i+1]){
40                 scambia(v[i],v[i+1]);
41                 scambi++;
42             }
43         }
44     }
45     scambi = 0;
46     for(int i = 0; i < sizeof(v)/sizeof(int); i++){
47         cout << v[i] << " ";
48     }
49     cout << endl;
50     return 0;
51 }
```

Register	Value
rax	0x0000000000000005
rbx	0x0000000000000000
rcx	0x00000000000000a0
rdx	0x0000000000000003
rsi	0x0000000000000003
rdi	0x0000000000000005
rbp	0x00007ffffee300
rsp	0x00007ffffee2d0
r8	0x00007ffff05cd80
r9	0x0000000000000000
r10	0x0000000000000006
r11	0x00007ffff19af30
r12	0x00000000000000870
r13	0x00007ffffee3e0
r14	0x0000000000000000
r15	0x0000000000000000
rip	0x0000000000000029

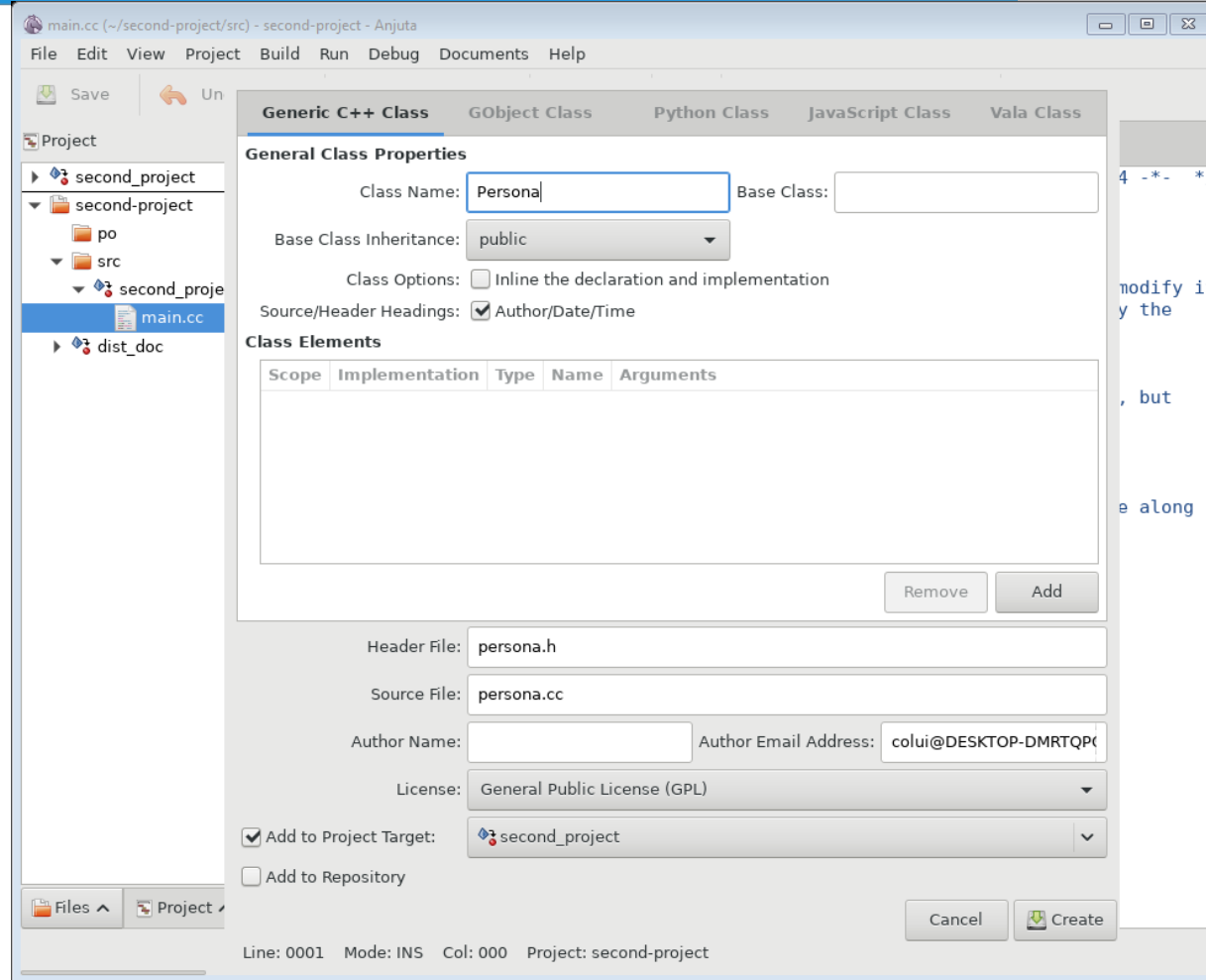
Variable	Value	Type
i	1	int
v	[5]	int [5]
0	2	int
1	5	int
2	3	int

Line: 0041 Mode: INS Col: 000 Debugger: Stopped Project: my-first-project



Progetto due:
New-> Project -> C++
second-project

□ File-> new Class



□ Completiamo l'interfaccia di Persona

The screenshot shows an IDE window titled '*persona.h (~/.second-project/src) - second-project - Anjuta'. The interface includes a menu bar (File, Edit, View, Project, Build, Run, Debug, Documents, Help), a toolbar with icons for Save, Undo, and various development actions, and a Project Explorer on the left. The Project Explorer shows a tree structure with 'second-project' expanded, containing 'po' and 'src' subfolders. Inside 'src', there is a 'second-project' subfolder containing 'main.cc', 'persona.cc', and 'persona.h' (which is selected). Below the Project Explorer are buttons for 'Files' and 'Project'. The main editor area displays the content of 'persona.h' with line numbers 20 to 39. The code defines the 'PERSONA_H' macro, includes the string header, uses the std namespace, and declares the 'Persona' class with public methods 'getName()' and a constructor, a protected attribute 'nome', and a private section. The status bar at the bottom indicates 'Line: 0025 Mode: INS Col: 020 Project: second-project'.

```
20 #ifndef _PERSONA_H_
21 #define _PERSONA_H_
22
23 #include <string>
24
25 using namespace std;
26
27 class Persona
28 {
29 public:
30     string getName();
31     Persona(string nome="");
32 protected:
33     string nome;
34 private:
35
36 };
37
38 #endif // _PERSONA_H_
39
```

- E forniamo un'implementazione della classe

The screenshot shows an IDE window titled "*persona.cc (~/.second-project/src) - second-project - Anjuta". The interface includes a menu bar (File, Edit, View, Project, Build, Run, Debug, Documents, Help), a toolbar with icons for Save, Undo, and other actions, and a Project Explorer on the left. The Project Explorer shows a tree structure with "second-project" containing "po" and "src" folders, with "src" containing "second-project" which includes "main.cc", "persona.cc", and "persona.h". The "persona.cc" file is selected and its content is displayed in the main editor. The code in "persona.cc" is as follows:

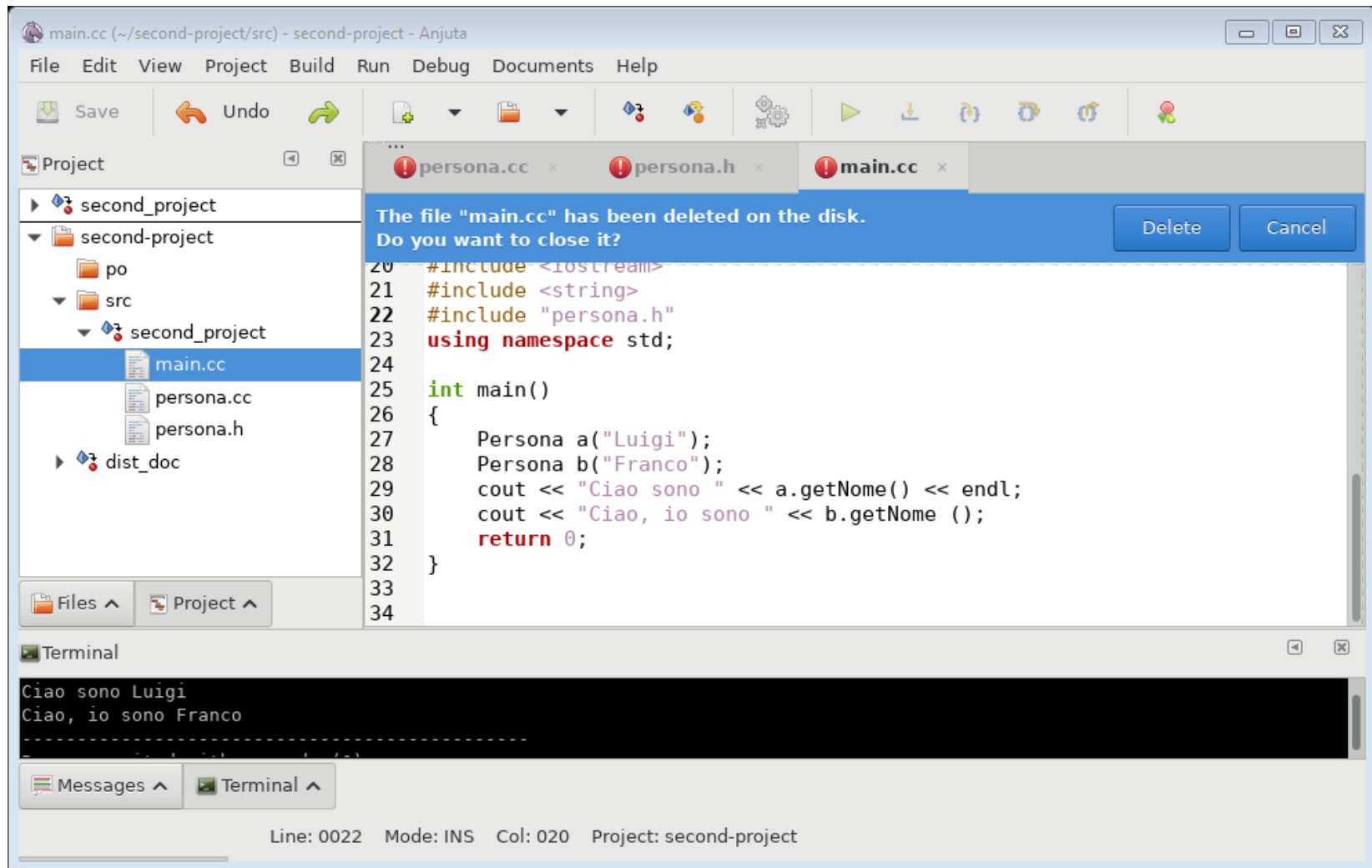
```

17  * with this program. If not, see <http://www.gnu.org/licenses/>.
18  */
19
20  #include "persona.h"
21
22  Persona::Persona(string nome){
23      this->nome = nome;
24  }
25
26  string Persona::getNome(){
27      return nome;
28  }
29

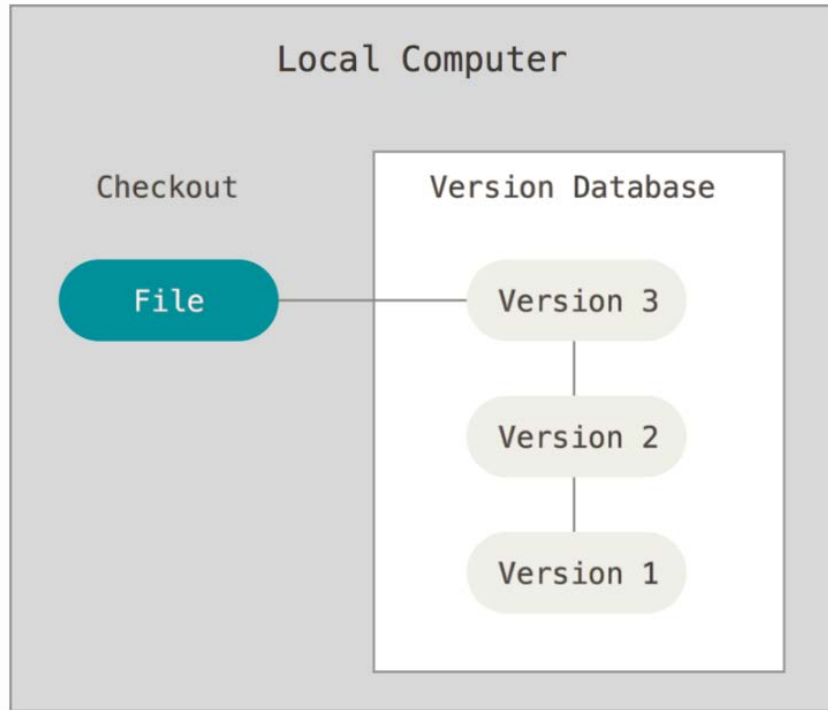
```

The status bar at the bottom indicates "Line: 0020 Mode: INS Col: 020 Project: second-project".

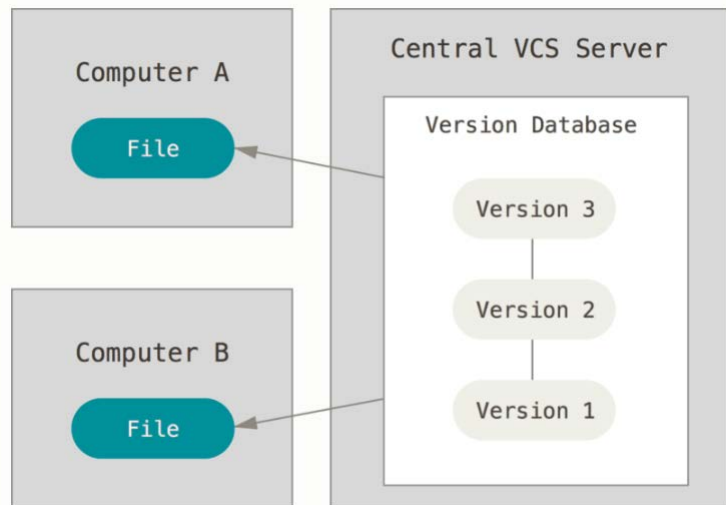
- E il main



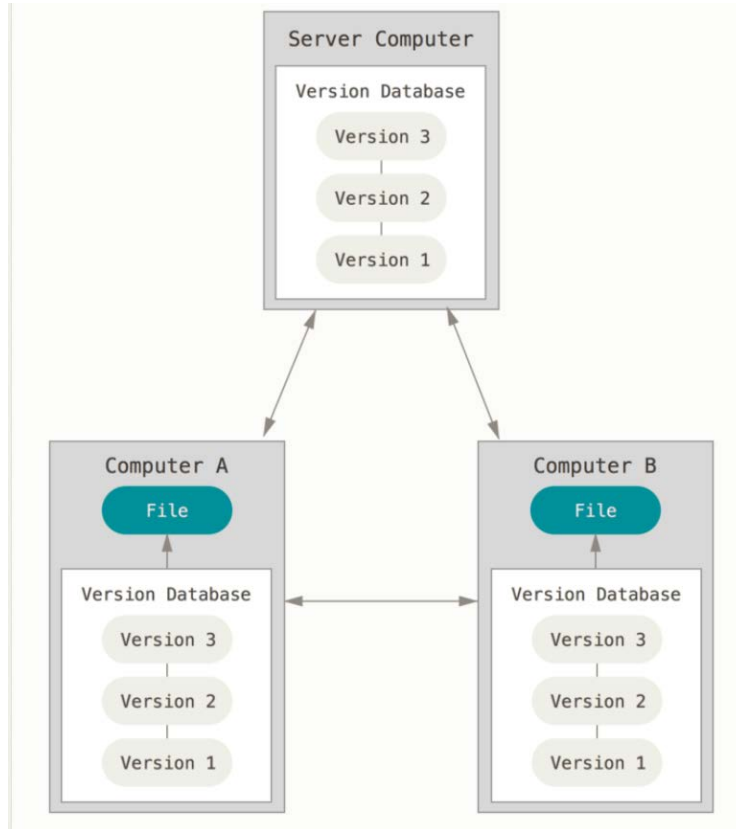
- Sistema che tiene traccia dei cambiamenti di un file o di un intero progetto nel tempo
 - Questo permette di ricavare versioni specifiche del file a nostra discrezione
 - Esempi
 - Ripristina file o un intero progetto a uno stato precedente
 - Confronta i cambiamenti che ci sono stati in un file
 - Vedi chi e cosa è stato modificato in un file
 - Nota: in caso di un problema, si può facilmente all'autore di una modifica quando è avvenuta e in quale parte del codice



- Metodo di version control:
 - Una volta modificati i file, copiali in una directory a parte
 - É soggetto a errori
- Ci sono VCS locali che hanno database che mantengono traccia di tutti cambiamenti di un file o di un insieme di file



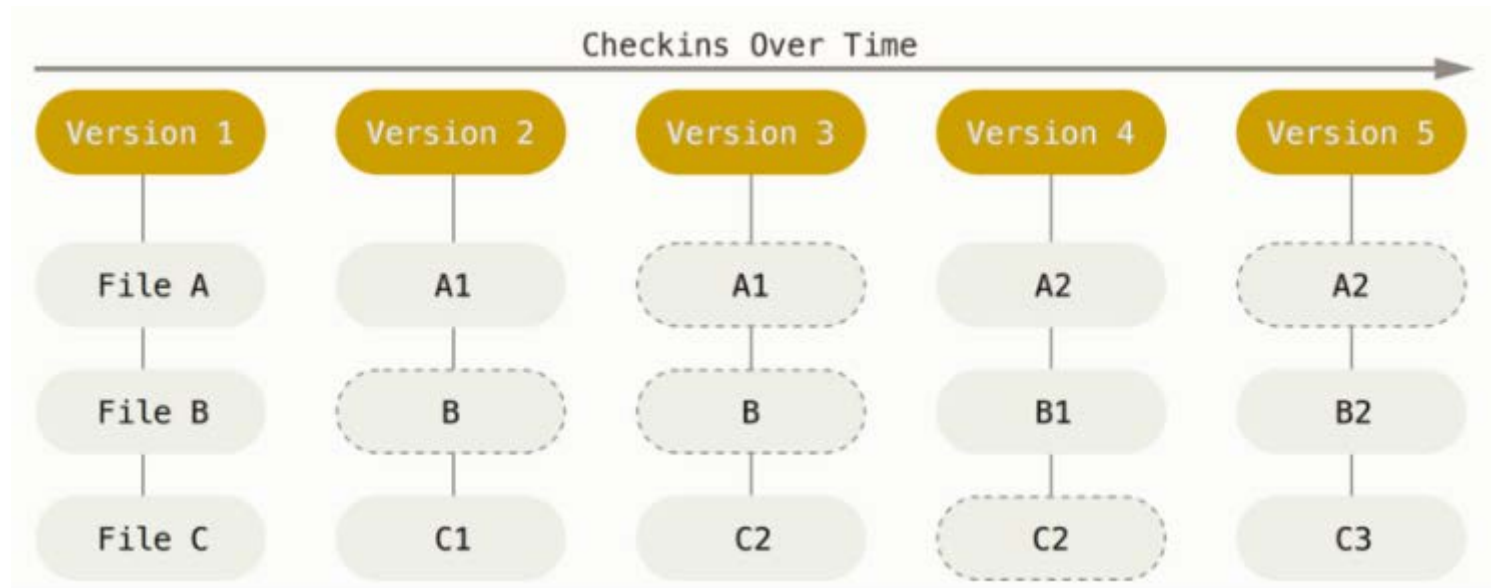
- Spesso è necessario lavorare su uno stesso pezzo di codice
- In un Centralized VCS esiste un server singolo che contiene tutte le versioni del file e diversi client che scaricano le versioni da questo server
- Se però il server va giù (*single point of failure*)
 - non sarà possibile aggiornare la versione di un file
 - I cambiamenti risulteranno solo in locale dove essi non vengono tracciati



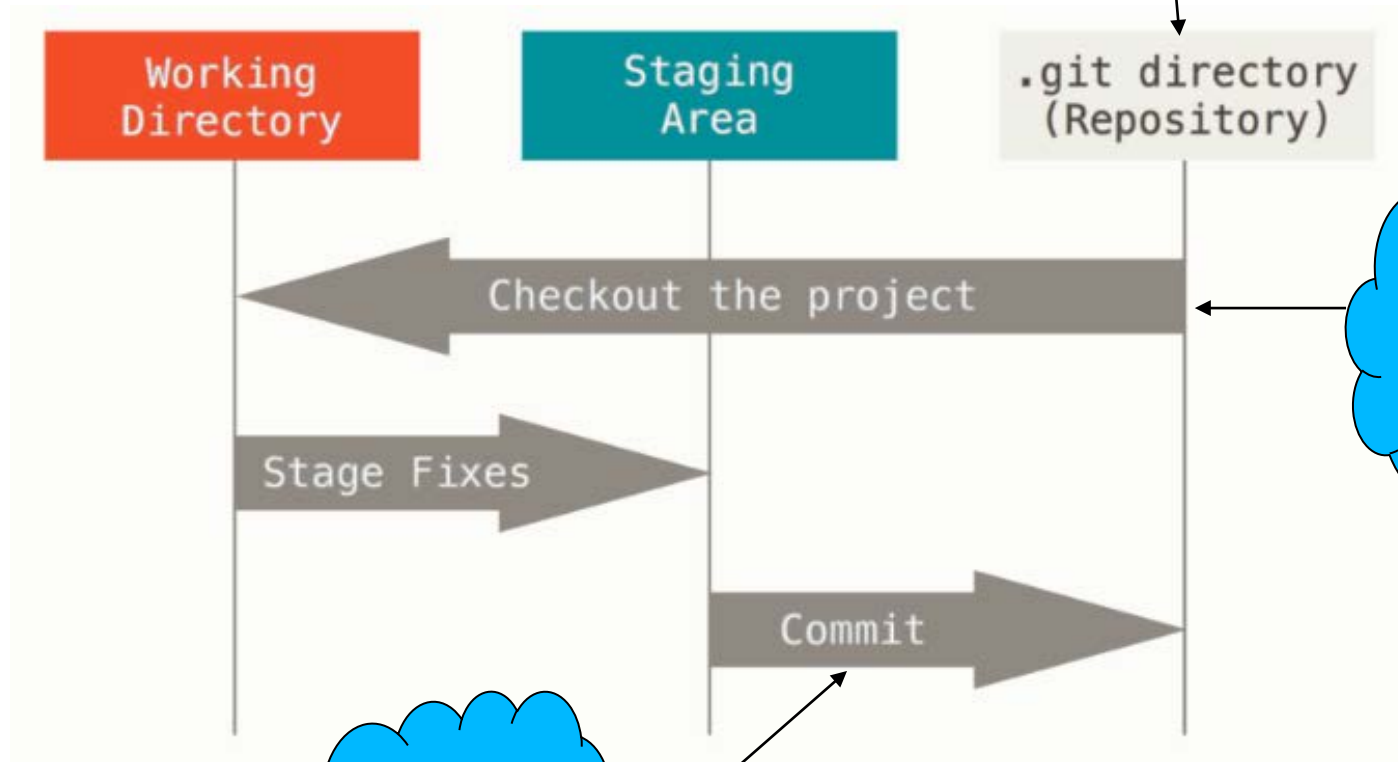
- In questo caso i client posseggono la stessa struttura, un **clone**, che si vede lato server
 - Ovvero essi mantengono le version dei file all'interno di un Progetto
 - Ogni client è un back up del server
- Quindi, se il server va giù, si può ripristinare il suo contenuto da qualsiasi client che lo aveva clonato.

□ Struttura dati

- Git mantiene uno snapshot del file system per ogni versione, è come uno stream di snapshot
 - Se un file non cambia da una versione a un'altra, esso non viene ricreato



- Le operazioni in locale
 - È possibile lavorare completamente in locale, senza dipendenze da server in remoto
 - Perché abbiamo tutta la storia del progetto
 - Ciò permette di evitare il possibile ritardo sulla rete
 - Si può lavorare offline, apportare delle modifiche al codice, tenendone traccia in locale, e solo quando si ritorna on line, applicare le modifiche sul server in remoto.
- Integrità dei dati
 - Ogni elemento in git è referenziato mediante il suo checksum (SHA-1), per cui se qualcosa cambia, esso viene immediatamente riconosciuto da git
 - I riferimenti sono rappresentati da 40 caratteri esadecimali, ad esempio:
 - *24b9da6552252987aa493b52f8696cd6d3b00373*



Il git directory è in locale

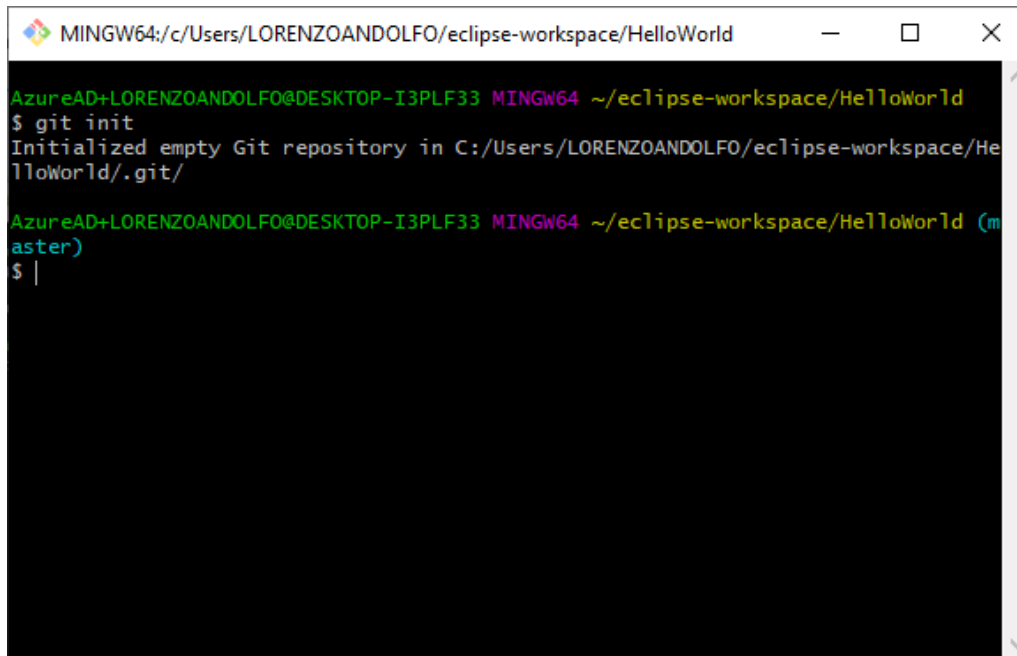
Porto nella WD una versione, lo snapshot del progetto

Aggiungo alla repository lo snapshot contenente le modifiche

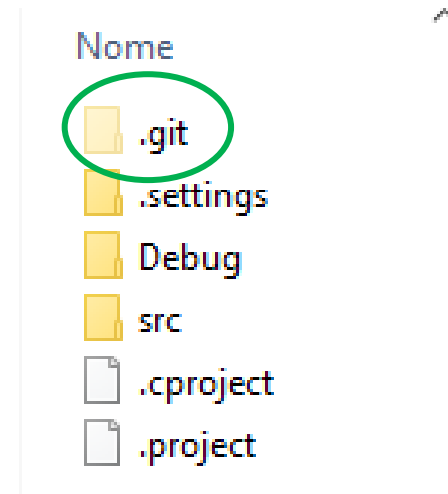
- *Working directory*
 - È il nostro workspace, dove scriviamo il codice
 - Visualizza una versione, uno snapshot, del progetto
- *Staging area*
 - Per tenere traccia di quello che ho intenzione di portare sulla *git directory (repository)*
- *git directory (repository)*
 - Dove git immagazzina tutte le informazioni del progetto.
 - Mediante l'operazione di “*clone*” di un repository permette di clonare il contenuto di questo ultimo su un altro pc

- Scarica git dal seguente sito
 - <https://git-scm.com/downloads>
 - Può essere utilizzato sia da windows o da linux

- ❑ Ottieni un repository del progetto
 - Posizionatevi nella directory in locale del progetto, apri la shell di git e digita:
 - *Comando: git init*
 - Viene creata una sottodirectory chiamata git che contiene la struttura vuota poiché non è ancora stato tracciato nulla



```
MINGW64:/c/Users/LORENZOANDOLFO/eclipse-workspace/HelloWorld
$ git init
Initialized empty Git repository in C:/Users/LORENZOANDOLFO/eclipse-workspace/HelloWorld/.git/
$
```



- È possibile visualizzare lo stato dei file
 - *Not staging area* -> file di colore rosso
 - Staging area -> file di colore verde
 - *Comando: git status*

Al momento tutti i file presenti nel repository non sono nella staging area

```
AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  .cproject
  .project
  .settings/
  Debug/
  src/

nothing added to commit but untracked files present (use "git add" to track)
```

- Comando: *git add fileName.cpp ...*

```
AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git add src/HelloWorld.cpp
warning: LF will be replaced by CRLF in src/HelloWorld.cpp.
The file will have its original line endings in your working directory

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   src/HelloWorld.cpp

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .cproject
    .project
    .settings/
    Debug/
    src/Num.cpp
    src/Num.h
    src/Vector.cpp
    src/Vector.h
```

File aggiunto alla staging area

- È possibile aggiungere direttamente il contenuto di una directory

```
src/Vector.h

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git add src/*

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   src/HelloWorld.cpp
    new file:   src/Num.cpp
    new file:   src/Num.h
    new file:   src/Vector.cpp
    new file:   src/Vector.h

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .cproject
    .project
    .settings/
    Debug/
```

Directory aggiunta

- È possibile inserire tutti i file alla staging area
 - Comando: *git add .*

```
AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git add .

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   .cproject
    new file:   .project
    new file:   .settings/language.settings.xml
    new file:   .settings/org.eclipse.cdt.managedbuilder.core.prefs
    new file:   Debug/HelloWorld.exe
    new file:   Debug/src/HelloWorld.o
    new file:   Debug/src/Num.o
    new file:   Debug/src/Vector.o
    new file:   src/HelloWorld.cpp
    new file:   src/Num.cpp
    new file:   src/Num.h
    new file:   src/Vector.cpp
    new file:   src/Vector.h
```

- Si possono rimuovere file dalla staging area se essi sono stati aggiunti per sbaglio. Supponiamo di mantenere solo la cartella src
 - Comando: *git rm --cached -r directory/* file.cpp*

```
AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git rm --cached -r Debug/* -r .settings/* .project .cproject
rm '.cproject'
rm '.project'
rm '.settings/language.settings.xml'
rm '.settings/org.eclipse.cdt.managedbuilder.core.prefs'
rm 'Debug/HelloWorld.exe'
rm 'Debug/src/HelloWorld.o'
rm 'Debug/src/Num.o'
rm 'Debug/src/Vector.o'

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   src/HelloWorld.cpp
    new file:   src/Num.cpp
    new file:   src/Num.h
    new file:   src/Vector.cpp
    new file:   src/Vector.h

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .cproject
    .project
    .settings/
    Debug/
```

```
AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git commit -m "Aggiunta progetto su git"
[master (root-commit) affcd26] Aggiunta progetto su git
Committer: LORENZO ANDOLFO <lorenzo.andolfo001@studenti.uniparthenope.it>
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. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

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

    git commit --amend --reset-author

5 files changed, 143 insertions(+)
create mode 100644 src/HelloWorld.cpp
create mode 100644 src/Num.cpp
create mode 100644 src/Num.h
create mode 100644 src/Vector.cpp
create mode 100644 src/Vector.h

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .cproject
        .project
        .settings/
        Debug/

nothing added to commit but untracked files present (use "git add" to track)

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git log
commit affcd269a1fdc230575cfa9fb4c196e8dcc8af1f (HEAD -> master)
Author: LORENZO ANDOLFO <lorenzo.andolfo001@studenti.uniparthenope.it>
Date:   Wed Jun 3 22:07:20 2020 +0200

    Aggiunta progetto su git
```

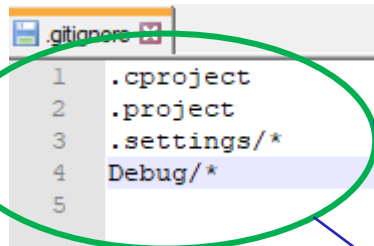
Commit su
repository locale

File andati sul
repository locale

Git log per
visualizzare la storia
dei commit

- Supponiamo di voler considerare solo determinati file nel processo di versioning
 - E' possibile elencare in un file di configurazione le cartelle e i file che non vogliamo includere
 - File .gitignore
 - Questo file sarà committato successivamente assieme agli altri

Aggiungo File che
non voglio tracciare



```
AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .cproject
        .project
        .settings/
        Debug/

nothing added to commit but untracked files present (use "git add" to track)

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore

nothing added to commit but untracked files present (use "git add" to track)
```

```
AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git add .gitignore

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git commit -m "Aggiunta .gitignore"
[master 31c7040] Aggiunta .gitignore
Committer: LORENZO ANDOLFO <lorenzo.andolfo001@studenti.uniparthenope.it>
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. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

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

    git commit --amend --reset-author

1 file changed, 4 insertions(+)
create mode 100644 .gitignore

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git log
commit 31c704096fc5e7843c4341b1481a46787defd4c0 (HEAD -> master)
Author: LORENZO ANDOLFO <lorenzo.andolfo001@studenti.uniparthenope.it>
Date:   Wed Jun 3 22:51:05 2020 +0200

    Aggiunta .gitignore

commit affcd269a1fdc230575cfa9fb4c196e8dcc8af1f
Author: LORENZO ANDOLFO <lorenzo.andolfo001@studenti.uniparthenope.it>
Date:   Wed Jun 3 22:07:20 2020 +0200

    Aggiunta progetto su git

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git status
On branch master
nothing to commit, working tree clean
```

- Supponiamo di voler visualizzare la differenza tra i due commit
 - Comando: *git diff commit1..commit2*

```
AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git diff 31c704096fc5e7843c4341b1481a46787defd4c0..affcd269a1fdc230575cfa9fb4c196e8dcc8af1
diff --git a/.gitignore b/.gitignore
deleted file mode 100644
index 0cda78b..0000000
--- a/.gitignore
+++ /dev/null
@@ -1,4 +0,0 @@
-.cproject
-.project
-.settings/*
-Debug/*
```

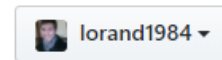
- I repository remoti contengono tutte le version dei tuoi progetti in remote
 - Operazioni più comuni sui repository remoti sono quelle di
 - Push: Un commit viene inoltrato dal mio repository locale a quello remoto
 - Pull: I commit presenti sul repository remoto vengono messi sul mio repository locale

- Creare un profilo da github:
 - <https://github.com/>
- Creare un repository
 - Andare sull'icona del «+» a destra per crearlo

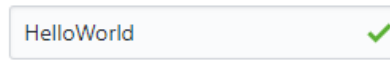
Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Owner



Repository name *



Great repository names are short and memorable. Need inspiration? How about **stunning-computing-machine**?

Description (optional)

☐  **Public**

Anyone can see this repository. You choose who can commit.

☒  **Private**

You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☐ **Initialize this repository with a README**

This will let you immediately clone the repository to your computer.


Add .gitignore: **None** ▼

Add a license: **None** ▼



Create repository

Quick setup — if you've done this kind of thing before

 Set up in Desktop or **HTTPS** SSH

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# HelloWorld" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/lorand1984/HelloWorld.git
git push -u origin master
```

Scegliamo questa
opzione perché già
abbiamo un
repository in locale

...or push an existing repository from the command line

```
git remote add origin https://github.com/lorand1984/HelloWorld.git
git push -u origin master
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

[Import code](#)

```
AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git remote add origin https://github.com/lorand1984/HelloWorld.git

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git push -u origin master
Logon failed, use ctrl+c to cancel basic credential prompt.
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 12 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (11/11), 1.64 KiB | 1.64 MiB/s, done.
Total 11 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/lorand1984/HelloWorld.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$
```

- ❑ Origin: rappresenta il repository remote
- ❑ Master: è il *branch* principale su cui abbiamo committato i cambiamenti
- ❑ Push: invia i dati su origin

The screenshot shows a GitHub repository interface. At the top, it displays '2 commits', '1 branch', '0 packages', and '0 releases'. Below this, there are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find file', and a green 'Clone or download' button. The repository name 'LORENZO ANDOLFO' is followed by 'Aggiunta .gitignore' and 'Latest commit 31c7040 1 hour ago'. A list of files is shown: 'src' (2 hours ago) and '.gitignore' (1 hour ago). A green oval highlights the commit history for 'src' and '.gitignore'. Below the file list, a blue box contains the text 'Add a README with an overview of your project.' and a green 'Add a README' button, which is also highlighted with a green oval. Two blue arrows point from text labels to these elements: one from 'Commit fatti in locale' to the commit history, and another from 'Possibilità di aggiungere un readme' to the 'Add a README' button.

2 commits 1 branch 0 packages 0 releases

Branch: master New pull request Create new file Upload files Find file Clone or download

LORENZO ANDOLFO Aggiunta .gitignore Latest commit 31c7040 1 hour ago

src 2 hours ago

.gitignore 1 hour ago

Aggiunta progetto su git

Aggiunta .gitignore

Add a README with an overview of your project.

Add a README


Commit fatti in locale

Possibilità di aggiungere un readme

HelloWorld / README.md Cancel

<> Edit new file Preview Spaces 2 No wrap

```
1 # HelloWorld
2 Istruzioni per lanciare Hello World...
```

 Commit new file

aggiunta readme

Add an optional extended description...

☒ Commit directly to the master branch.

☐ Create a new branch for this commit and start a pull request. [Learn more about pull requests.](#)

Commit new file Cancel

```
AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git fetch
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 731 bytes | 66.00 KiB/s, done.
From https://github.com/lorand1984/HelloWorld
 31c7040..558c7c4  master    -> origin/master

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git status
On branch master
Your branch is behind 'origin/master' by 1 commit, and can be fast-forwarded.
  (use "git pull" to update your local branch)

nothing to commit, working tree clean

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ git pull origin master
From https://github.com/lorand1984/HelloWorld
 * branch            master       -> FETCH_HEAD
Updating 31c7040..558c7c4
Fast-forward
 README.md | 2 ++
 1 file changed, 2 insertions(+)
 create mode 100644 README.md
```

Rende visibili i
branch remoti
in locale

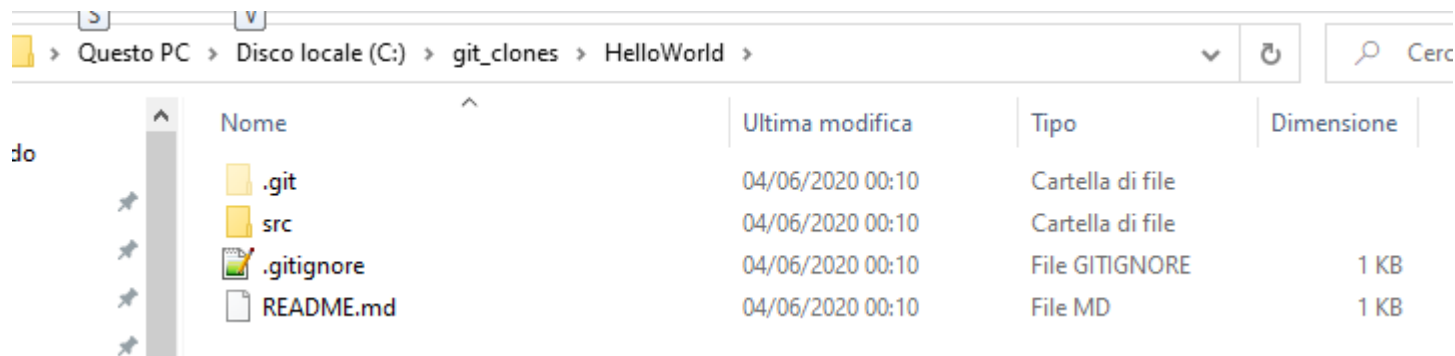
Prende da
origin master
il commit e lo
porta in locale

- È possibile clonare un repository remoto in locale
 - Comando: *git clone repo_name*

```
AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 ~/eclipse-workspace/HelloWorld (master)
$ cd c:\git_clones

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 /c/git_clones
$ git clone https://github.com/lorand1984/HelloWorld.git
Cloning into 'HelloWorld'...
remote: Enumerating objects: 14, done.
remote: Counting objects: 100% (14/14), done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 14 (delta 1), reused 11 (delta 1), pack-reused 0
Unpacking objects: 100% (14/14), 2.32 KiB | 71.00 KiB/s, done.

AzureAD+LORENZOANDOLFO@DESKTOP-I3PLF33 MINGW64 /c/git_clones
```



The screenshot shows a Windows File Explorer window with the address bar set to "Questo PC > Disco locale (C:) > git_clones > HelloWorld". The main area displays a list of files and folders:

Nome	Ultima modifica	Tipo	Dimensione
.git	04/06/2020 00:10	Cartella di file	
src	04/06/2020 00:10	Cartella di file	
.gitignore	04/06/2020 00:10	File GITIGNORE	1 KB
README.md	04/06/2020 00:10	File MD	1 KB

Comando git	descrizione
git add «nome_file»	muove «nome_file» nella staging area
git --cached rm «nome_file»	rimuove «nome_file» dalla staging area
git commit -m "nome messaggio commit"	Fai il commit sul repository locale
git commit --amend	Sovrascrivi l'ultimo commit (Per fare il push dopo devi usare l'opzione -f)
git checkout «nome_file»	ripristina «nome_file» che era «untracked» cioè non in staging
git push origin «local-name:remote-name»	Fai il push da un branch locale «local-name» a un branch remoto «remote-name»
git pull origin «remote-name»	Fai il pull dal branch remote-name al branch locale corrente
git checkout «nome_branch»	Ti sposti sul branch «nome_branch»
git checkout -b «nome_file»	Crea branch «nome_file» che parte dal branch corrente e ti sposti su di esso

Comando git	descrizione
git remote prune origin	Cancella tutti i riferimenti in locale ai branch remoti cancellati chiamati in gergo "stale branch".
git branch -a	Fai il list di tutti i branch in remoto
git fetch --all (-a)	Preleva tutti i branch da tutti i remote collegati
git branch -d -r origin/»miobranch»	Cancella il riferimento di un particolare branch (branch remoto cancellato).
git log	Stampa la storia dei commit sul branch corrente
git cherry-pick «nome_del_commit»	Prende il commit nome_del_commit e lo mette sul branch corrente.
git remote show origin	Mostra il repository remoto tracciato in fetch e in pull
git reset «commit»	Resetta tutto quello che veniva prima di «commit» portandolo come «untracked»

- ❑ <http://anjuta.org/>
- ❑ <https://git-scm.com/doc>
- ❑ <https://github.com/>