

Fondamenti di Informatica e Laboratorio T-AB
Ingegneria Elettronica e Telecomunicazioni

Lab 01

Introduzione a Codelite

Costruzione di un'Applicazione

Per costruire un'applicazione occorre:

- **compilare il file (o / file se più d'uno) che contengono il testo del programma (file sorgente)**

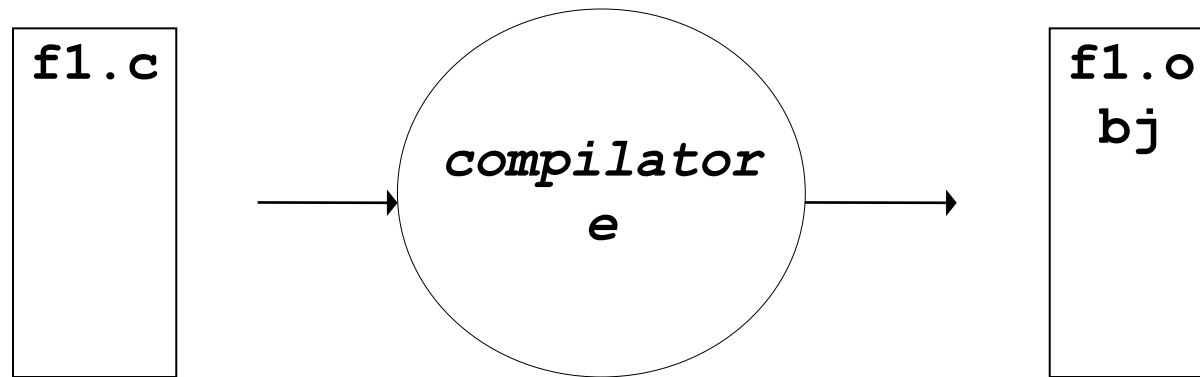
Il risultato sono uno o più file oggetto.

- **collegare i file oggetto l'uno con l'altro e con le librerie di sistema.**

Compilazione di un'Applicazione

1) Compilare il file (o *i* file se più d'uno) che contengono il testo del programma

- **File sorgente:** estensione **.c**
- **File oggetto:** estensione **.o** o **.obj**

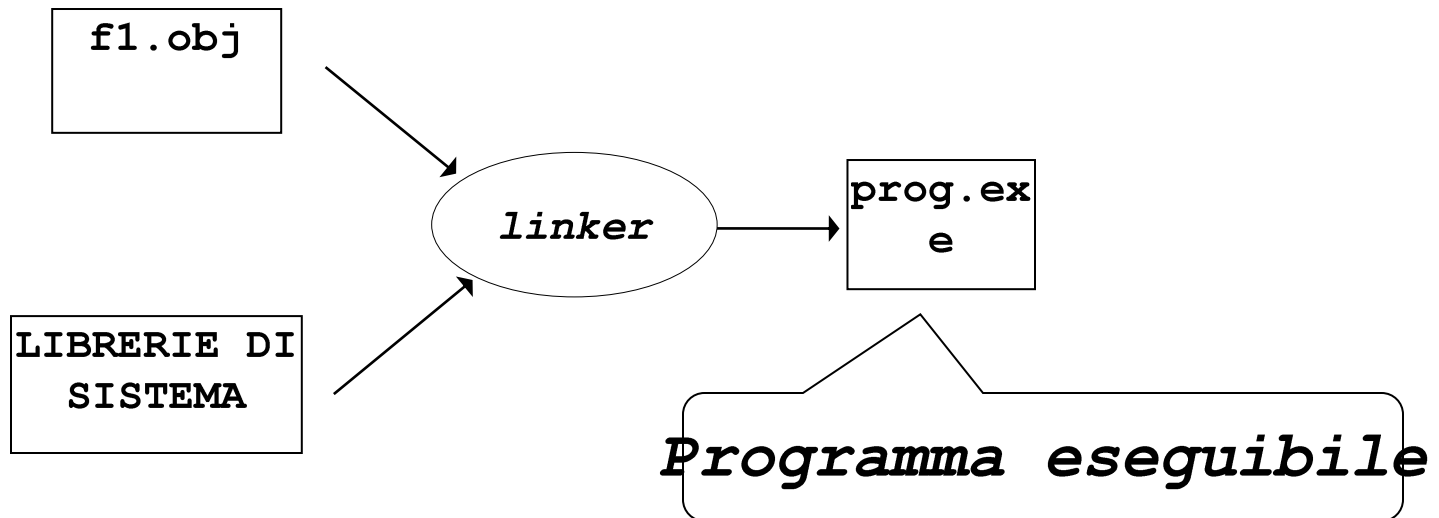


f1.obj: Una versione tradotta che però non è autonoma (e, quindi, non è direttamente eseguibile).

Collegamento (Linking) di un'Applicazione

2) Collegare il file (o *i* file) oggetto fra loro e con le librerie di sistema

- File oggetto: estensione **.o** o **.obj**
- File eseguibile: estensione **.exe** o nessuna



Collegamento (Linking) di un'Applicazione

LIBRERIE DI SISTEMA:

insieme di componenti software che consentono di interfacciarsi col sistema operativo, usare le risorse da esso gestite, e realizzare alcune "istruzioni complesse" del linguaggio

Ambienti Integrati

Oggi, gli *ambienti di lavoro integrati* automatizzano la procedura:

- **compilano i file sorgente (se e *quando necessario*)**
- **invocano il linker per costruire l'eseguibile**

ma per farlo devono sapere:

- ***quali file sorgente* costituiscono l'applicazione**
- ***il nome dell'eseguibile* da produrre.**

Progetti

È da queste esigenze che nasce il concetto di **PROGETTO**

- **un contenitore concettuale (e fisico)**
- **che elenca i file sorgente in cui l'applicazione è strutturata**
- **ed eventualmente altre informazioni utili.**

Oggi, *tutti* gli ambienti di sviluppo integrati, *per qualunque linguaggio*, forniscono questo concetto e lo supportano con idonei strumenti.

Installare Codelite

Download dal sito web: <http://downloads.codelite.org/>

CodeLite 9.1.0 - Stable Release released on Feb 1, 2016

 CodeLite 9.1.0 for Windows **64** bit Installer [Direct Link](#) | [GitHub](#)



Windows

 CodeLite 9.1.0 for Windows **32** bit Installer [Direct Link](#) | [GitHub](#)




 CodeLite 9.1.0 App Bundle for OSX 10.10 [Direct Link](#) | [GitHub](#)



Mac OS X

 [Download CodeLite 9.1.0 tar.gz from GitHub](#)

 [Setup CodeLite apt repository for Ubuntu / Debian and their derivatives e.g. Mint »](#)



Linux

 [CodeLite RPMs \(Fedora, openSUSE\) »](#)

Oppure installare la macchina virtuale fornita:

[Instruzioni per l'installazione](#)

Prima Configurazione (1)

Al primo avvio di Codelite **sotto Windows** sulle macchine in laboratorio, sarà necessario configurare l'ambiente di sviluppo

- È una procedura automatica avviata alla prima esecuzione
- È possibile scegliere alcune opzioni per specificare la modalità di visualizzazione del codice e altri dettagli
- **IMPORTANTE!** quando verrà chiesto che compiler occorre selezionare MingGW seguendo questa procedura:
 - cliccare sul menù "Add compilers" e selezionare "Scan computer for installed compilers"->"OK"->"OK"

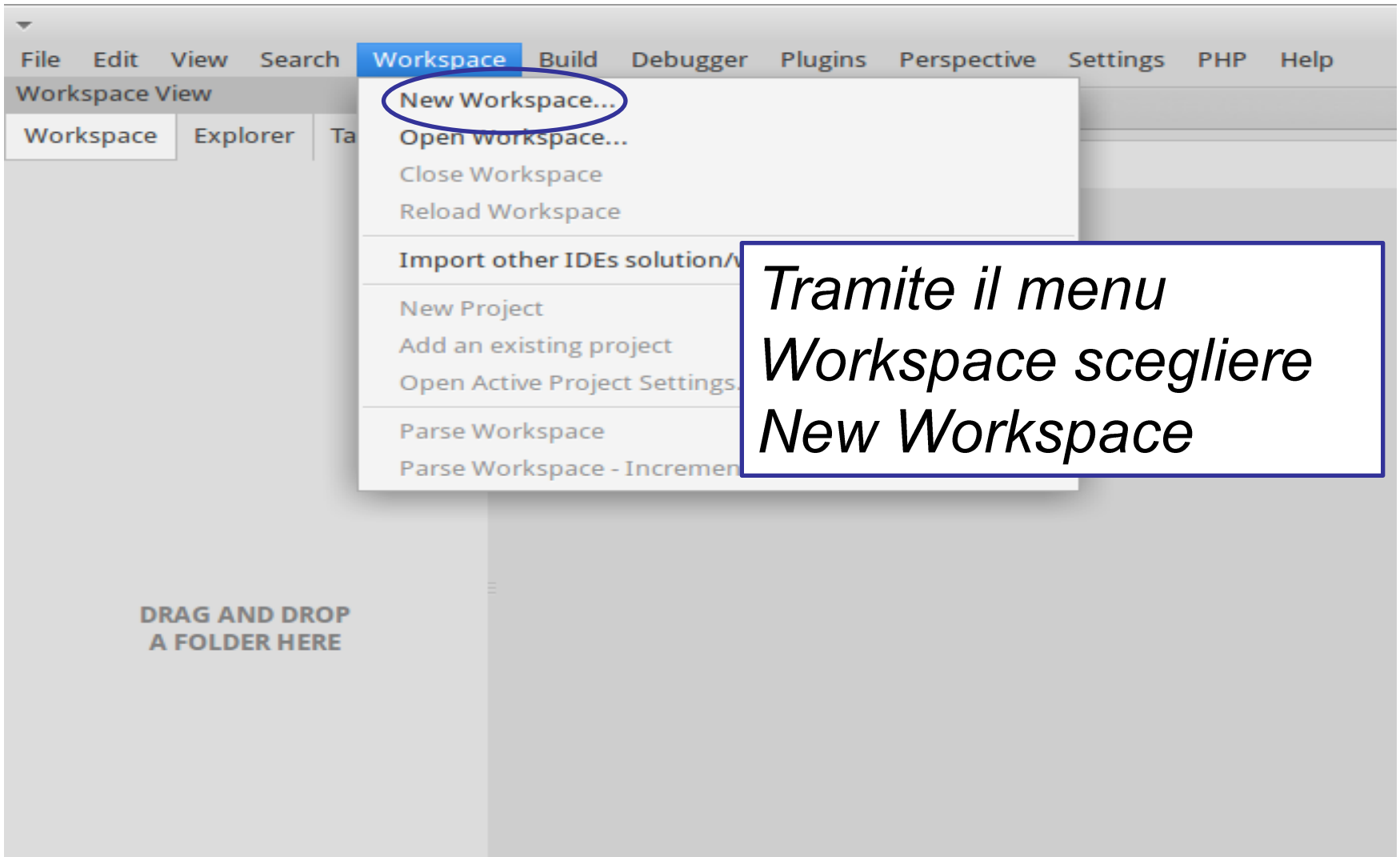
Prima Configurazione (2)

Nel caso la prima configurazione non fosse stata eseguita correttamente quando proverete ad usare il debugger verrà mostrato a video un errore. Per rimediare eseguire nuovamente la procedura di selezione del compiler (MinGW)

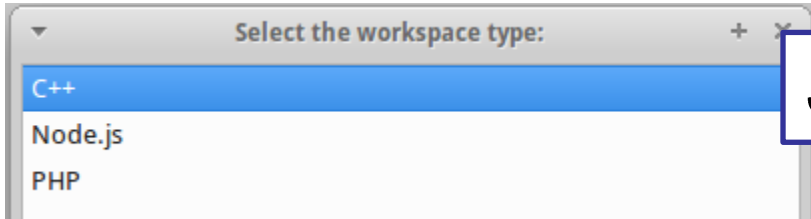
Selezionare dal menù "Settings"->"Build settings", nella scheda "Compiler", cliccare sul menù "Add compilers" e selezionare "Scan computer for installed compilers"->"OK"->"OK".

Se il problema persiste (o per eliminare ogni dubbio..) eliminare il workspace corrente e crearne uno nuovo

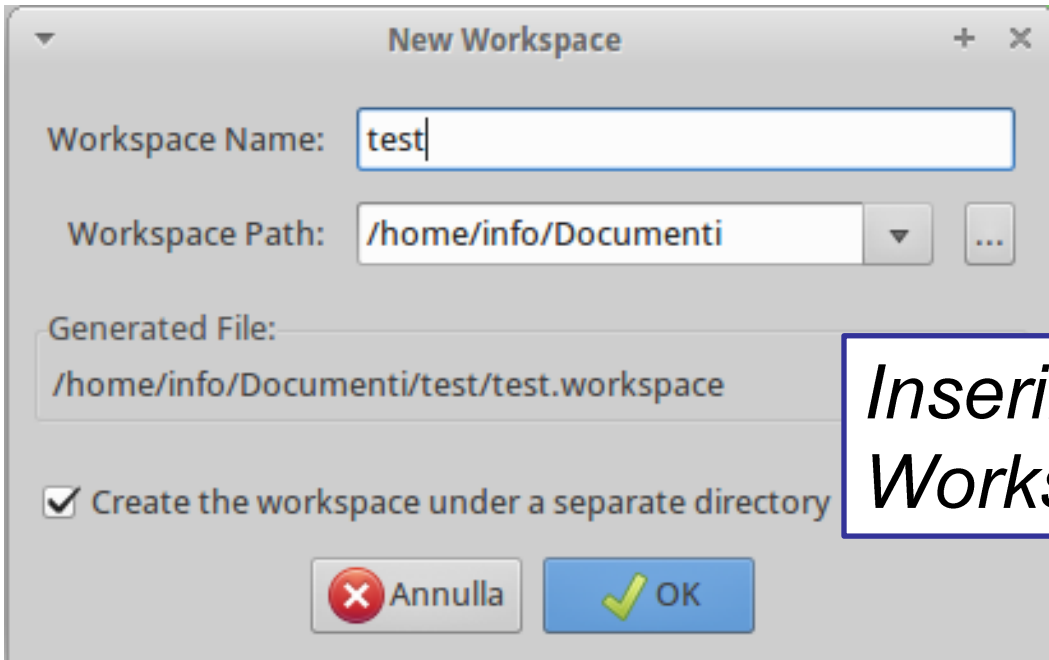
Progetti in Codelite



Progetti in Codelite

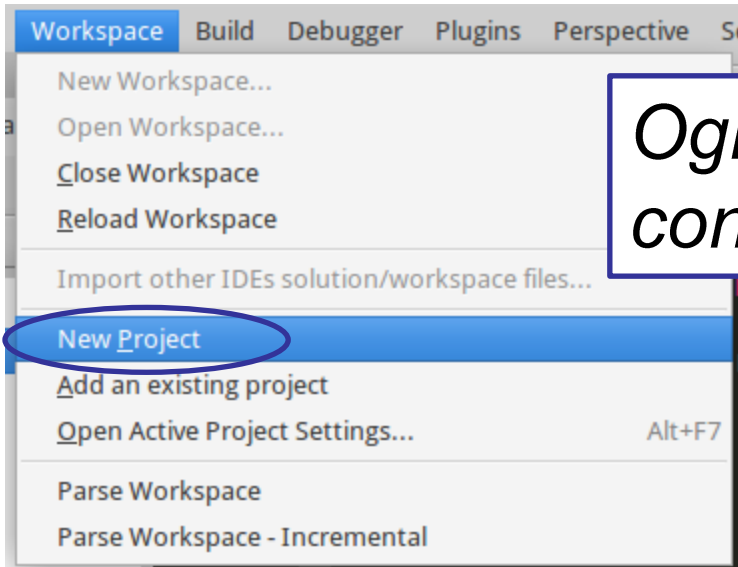


Selezionare C++



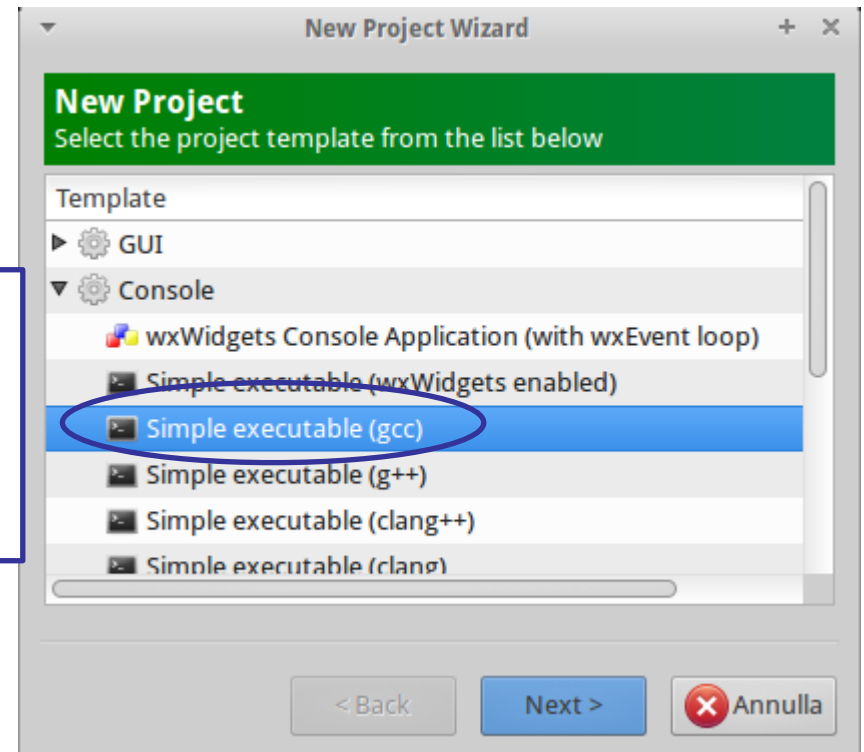
*Inserire il nome del
Workspace ed il percorso*

Progetti in Codelite

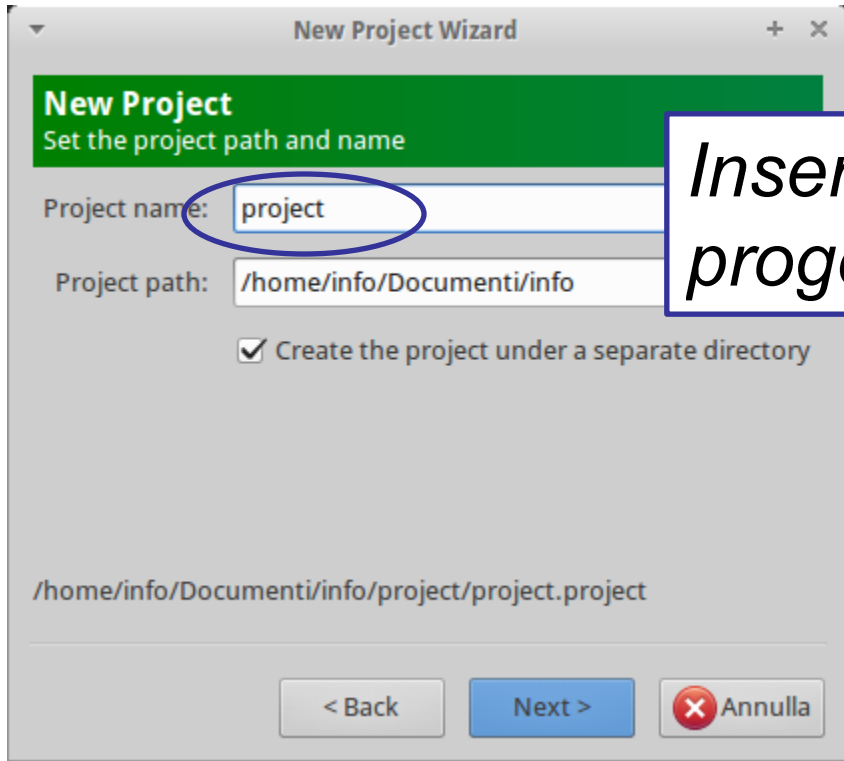


Ogni workspace può contenere uno o più progetti

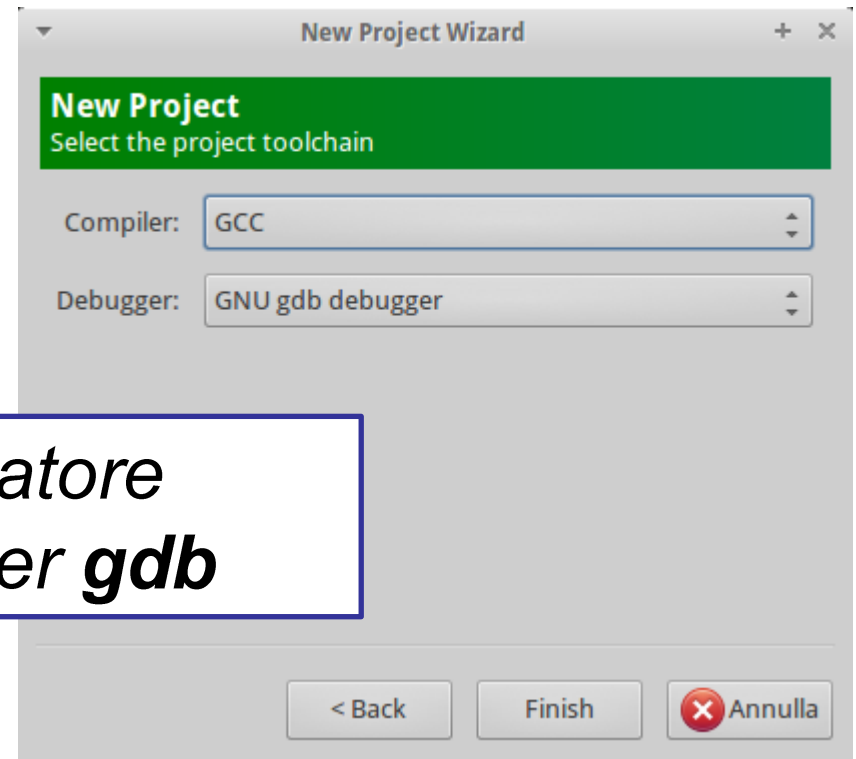
Selezionare la categoria 'Console' ed il template 'gcc'



Progetti in Codelite



Inserire il nome del progetto



*Selezionare il compilatore **MinGW** ed il debugger **gdb***

Progetti in Codelite

The screenshot displays the Codelite IDE interface. The top menu bar includes File, Edit, View, Search, Workspace, Build, Debugger, Plugins, Perspective, Settings, PHP, and Help. The main workspace is divided into three primary views:

- Workspace View:** Located on the left, it shows a project tree with folders 'info', 'test', and 'src'. The file 'main.c' is selected and highlighted in blue.
- EditorView:** The central area shows the code editor for 'main.c'. The code is as follows:

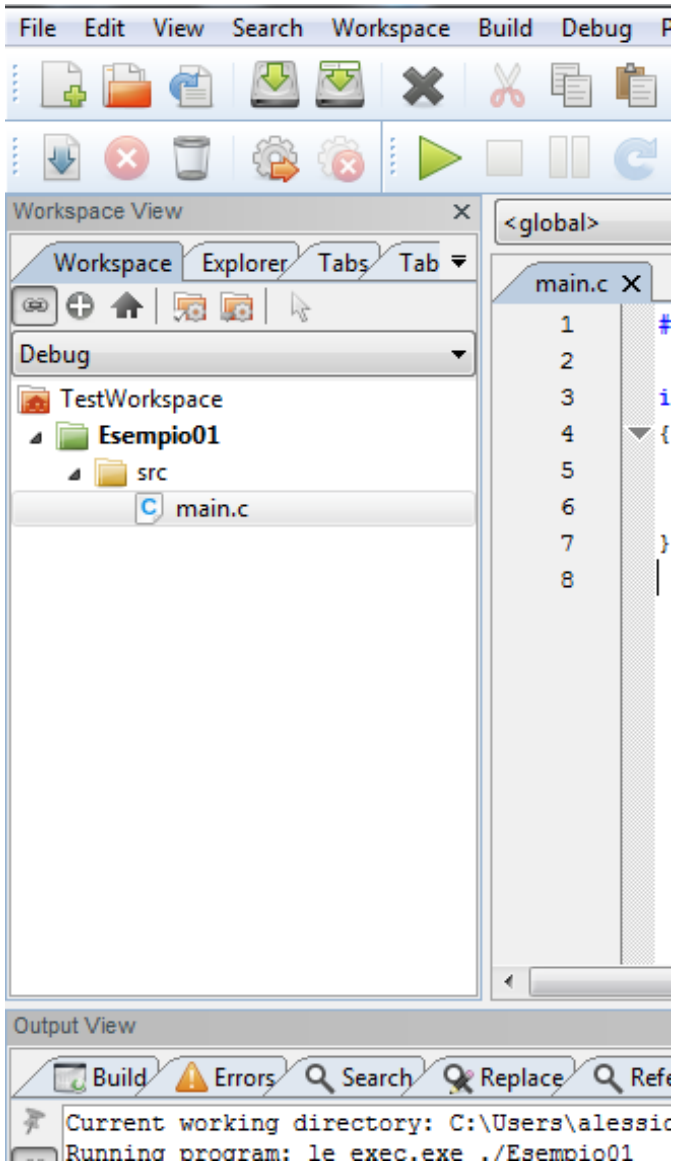
```
1 #include <stdio.h>
2
3 int main(int argc, char **argv)
4 {
5     printf("hello world\n");
6     return 0;
7 }
8
```
- Output View:** Located at the bottom, it displays the execution output:

```
Current working directory: /home/info/Documenti/info/test/Debug
Running program: /usr/bin/codelite_xterm './test' '/bin/sh -f /usr/bin/codelit
Program exited with return code: 0
```

Blue rounded rectangles highlight the Workspace View, EditorView, and Output View. White boxes with black text labels 'Workspace View', 'EditorView', and 'Output View' are connected to their respective views by blue lines.

At the bottom of the IDE, the status bar shows 'Ln 8, Col 0, Pos 93', 'SPACES', and 'C++'.

Progetti in Codelite



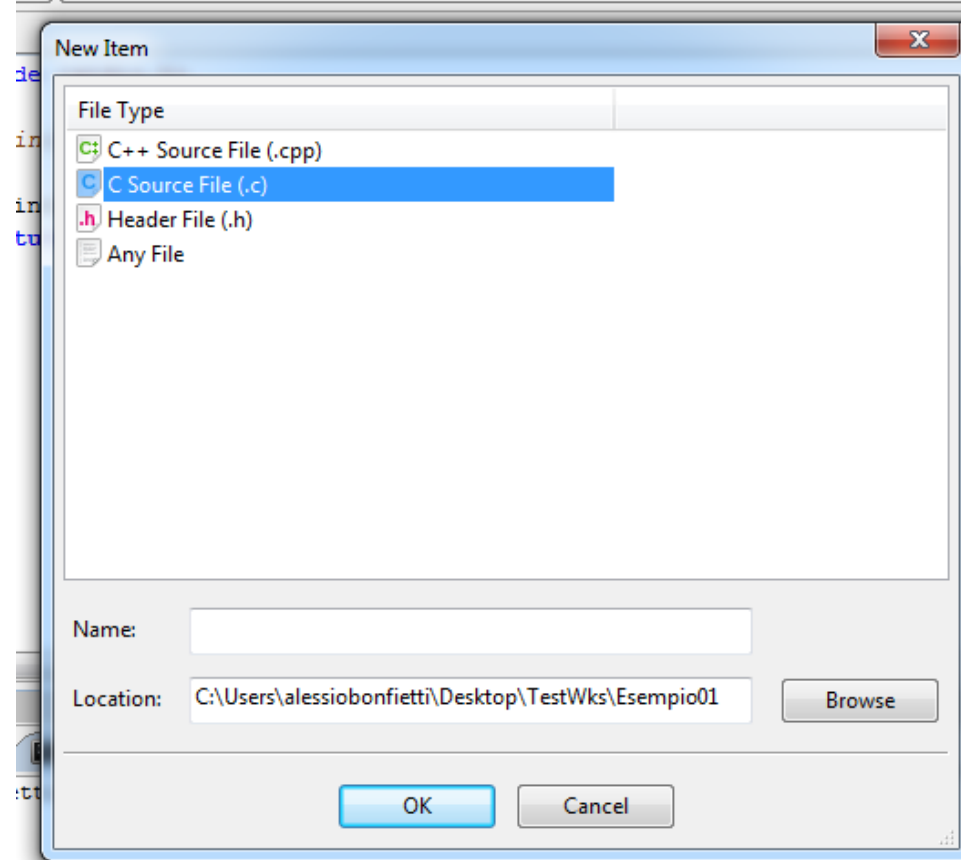
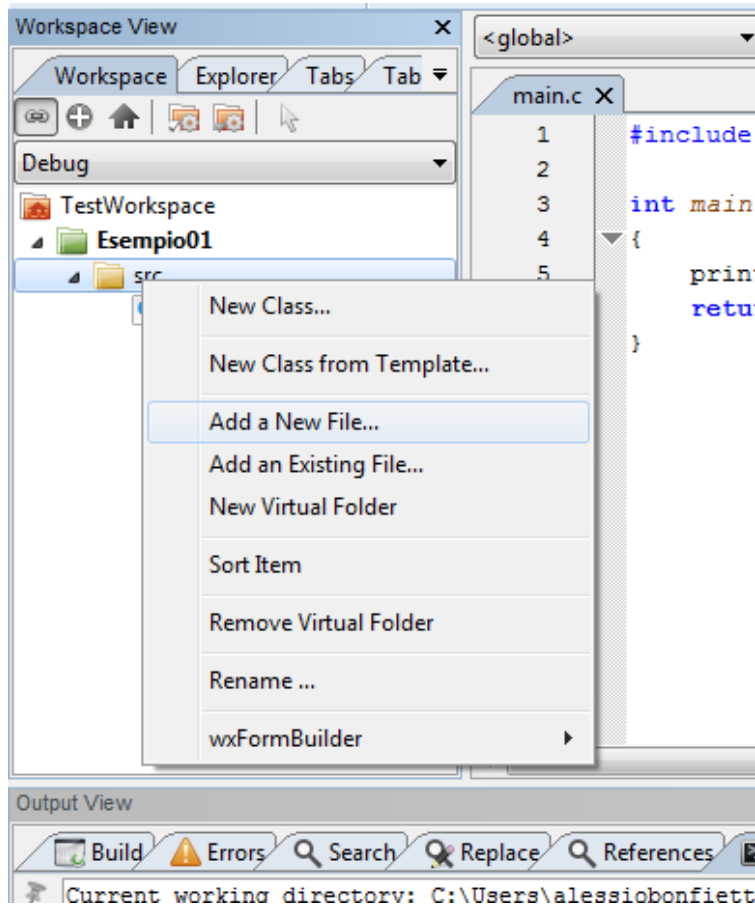
Workspace View

Alla creazione di un progetto, l'IDE **Codelite** crea automaticamente il file principale contenente la funzione main del programma.

Da questa interfaccia è dedicata alla gestione dei file sorgente

Progetti in Codelite

Click destro sulla directory 'src' per aggiungere un file sorgente



Progetti in Codelite

The screenshot shows the Codelite IDE interface. On the left, the 'View' menu is open, listing various panes and options. The main editor area displays a C++ source file named 'main.c' with line numbers 1 through 8. The function signature 'main(int argc, char **argv)' is visible in the toolbar above the editor. The status bar at the bottom shows the current directory and the process ID.

View Search Workspace Build Debugger Plugins

- Word Wrap
- Toggle Current Fold
- Toggle All Folds
- Toggle All Topmost Folds in Selection
- Toggle Every Fold in Selection
- Display EOL
- Show Whitespace
- Full Screen...
- Show Welcome Page
- ✓ Load Welcome Page at Startup
- ✓ Output Pane
- ✓ Workspace Pane
- ✓ **Navigation Bar**
- Debugger Pane
- ✓ Show Status Bar
- Show ToolBar

Build Debug Plugins Perspective Settings Help C++

Elenco delle funzioni: per raggiungere velocemente un punto nel codice

main(int argc, char **argv)

main.c X

```
1 #include <stdio.h>
2
3 int main(int
4 {
5     printf("
6     return 0;
7 }
8
```

Barra delle Tab: veloce accesso ai file sorgenti aperti

EditorView

Per attivare l'elenco delle funzioni

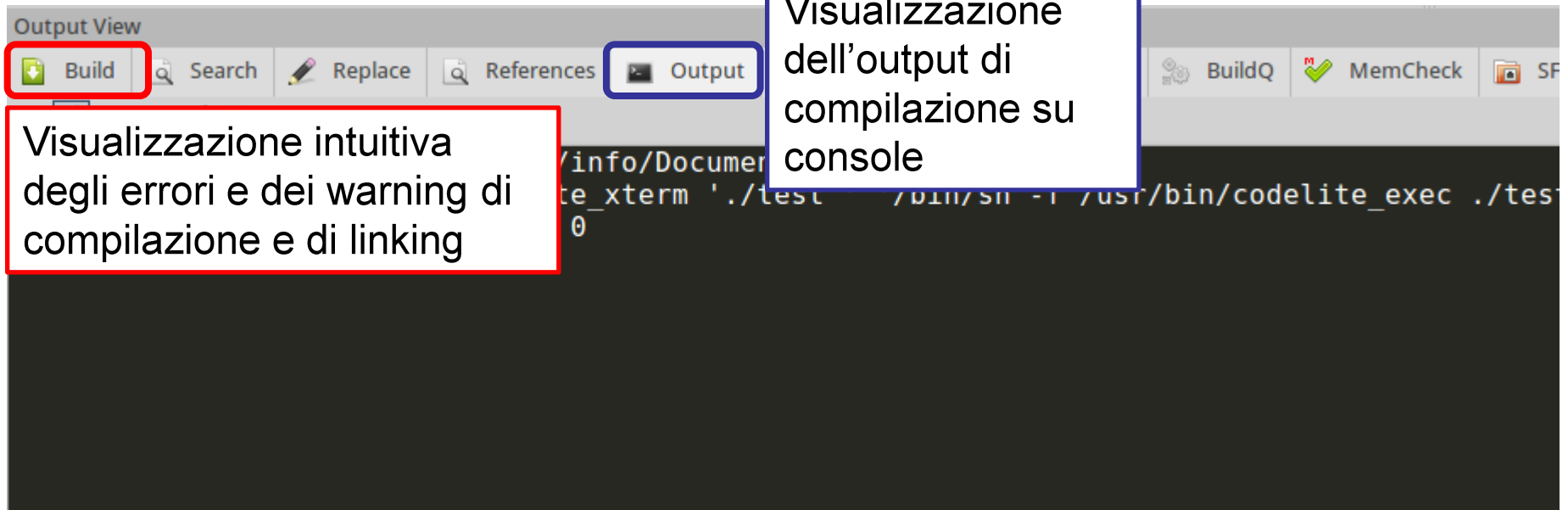
Numeri di linea

Replace References Output Trace Tasks BuildQ CppCheck CScope

\\Users\\alessiobonfietti\\Desktop\\TestWks\\Esempio01\\Debug
./Esempio01
id: -1073741510

Progetti in Codelite

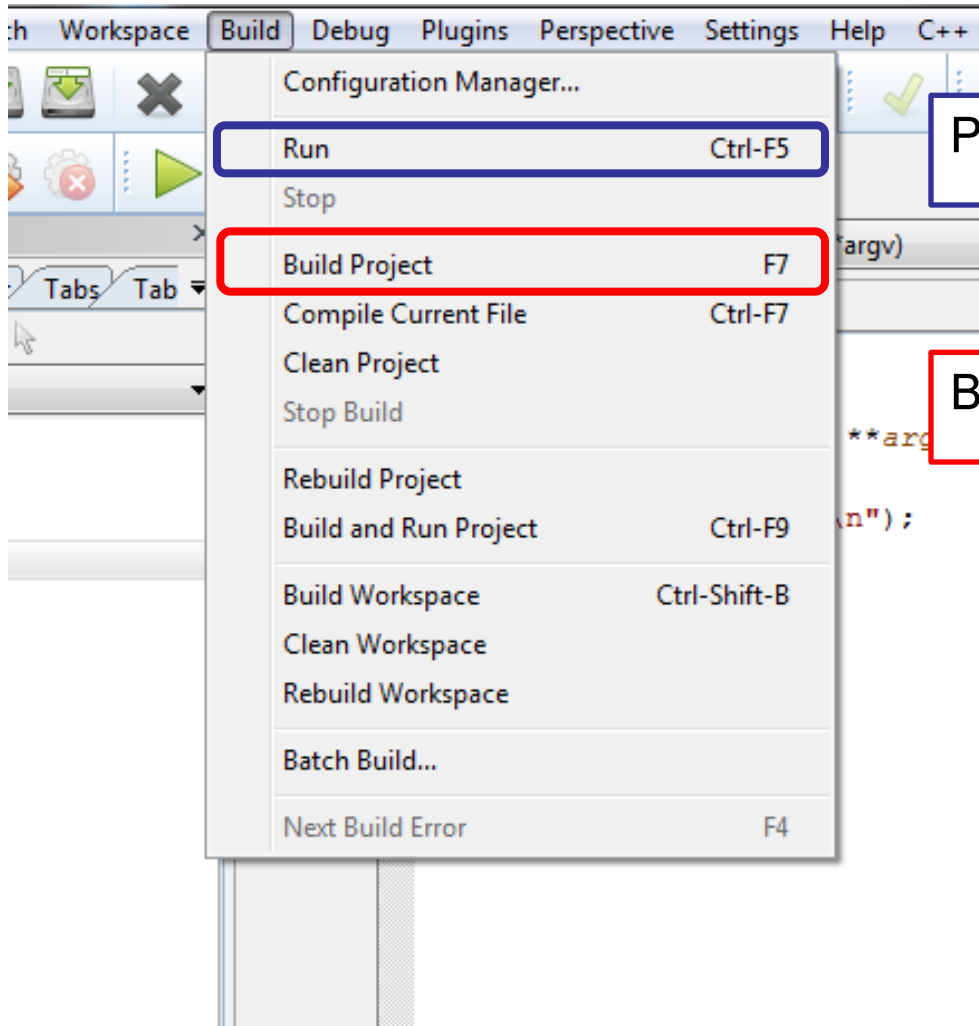
Output View



Visualizzazione intuitiva degli errori e dei warning di compilazione e di linking

Visualizzazione dell'output di compilazione su console

Progetti in Codelite



Per Eseguire il programma

Build = Compile + Link

Build: Warning

The screenshot shows a code editor with a C program in `main.c`. The code is as follows:

```
1 #include <stdio.h>
2
3 int main(int argc, char **argv)
4 {
5     int i;
6     printf("hello world\n");
7     return 0;
8 }
9
```

A red box highlights the `int i;` line, and a yellow arrow points to it. A text box on the right contains the text "Indicazione del warning".

The Output View at the bottom shows the following output:

```
make[1]: Leaving directory `/home/info/Documenti/info/test'
make[1]: Entering directory `/home/info/Documenti/info/test'
/usr/bin/gcc -c "/home/info/Documenti/info/test/main.c" -g -O0 -Wall -o ./Debug/main.c.o -I
/home/info/Documenti/info/test/main.c: In function 'main':
/home/info/Documenti/info/test/main.c:5:9: warning: unused variable 'i' [-Wunused-variable]
    int i;
    ^
```

A red box highlights the warning message in the output view. The status bar at the bottom indicates "Ln 6, Col 15, Pos 77" and "SPACES C++".

Build: Errors

The screenshot shows an IDE window with the following components:

- Workspace View:** Shows a project structure with folders 'info', 'test', and 'src', and a file 'main.c'.
- Code Editor:** Displays the content of 'main.c':

```
2  
3 int main(int argc, char **argv)  
4 {  
5     int i = 5  
6  
7     printf("hello world\n");  
8     i = 2;  
9     return 0;  
10 }  
11
```
- Output View:** Shows the compilation output:

```
make[1]: Leaving directory `/home/info/Documenti/info/test'  
make[1]: Entering directory `/home/info/Documenti/info/test'  
/usr/bin/gcc -c "/home/info/Documenti/info/test/main.c" -g -O0 -Wall -o ./Debug/main.c.o -I  
/home/info/Documenti/info/test/main.c: In function 'main':  
/home/info/Documenti/info/test/main.c:7:2: error: expected ',' or ';' before 'printf'  
    printf("hello world\n");  
    ^  
/home/info/Documenti/info/test/main.c:5:9: warning: variable 'i' set but not used [-Wunused-bu
```

A blue box highlights the error message in the Output View, and a white arrow points from it to the 'printf' line in the code editor.

Indicazione degli errori

ESAMIX

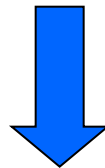
<http://esamix.labx>

Il Debugger

Una volta scritto, compilato e collegato il programma (ossia, costruito l'eseguibile)

occorre uno strumento che consenta di

- **eseguire il programma passo per passo**
- **vedendo le variabili e la loro evoluzione**
- **e seguendo le funzioni via via chiamate.**



Debugger

Debugger

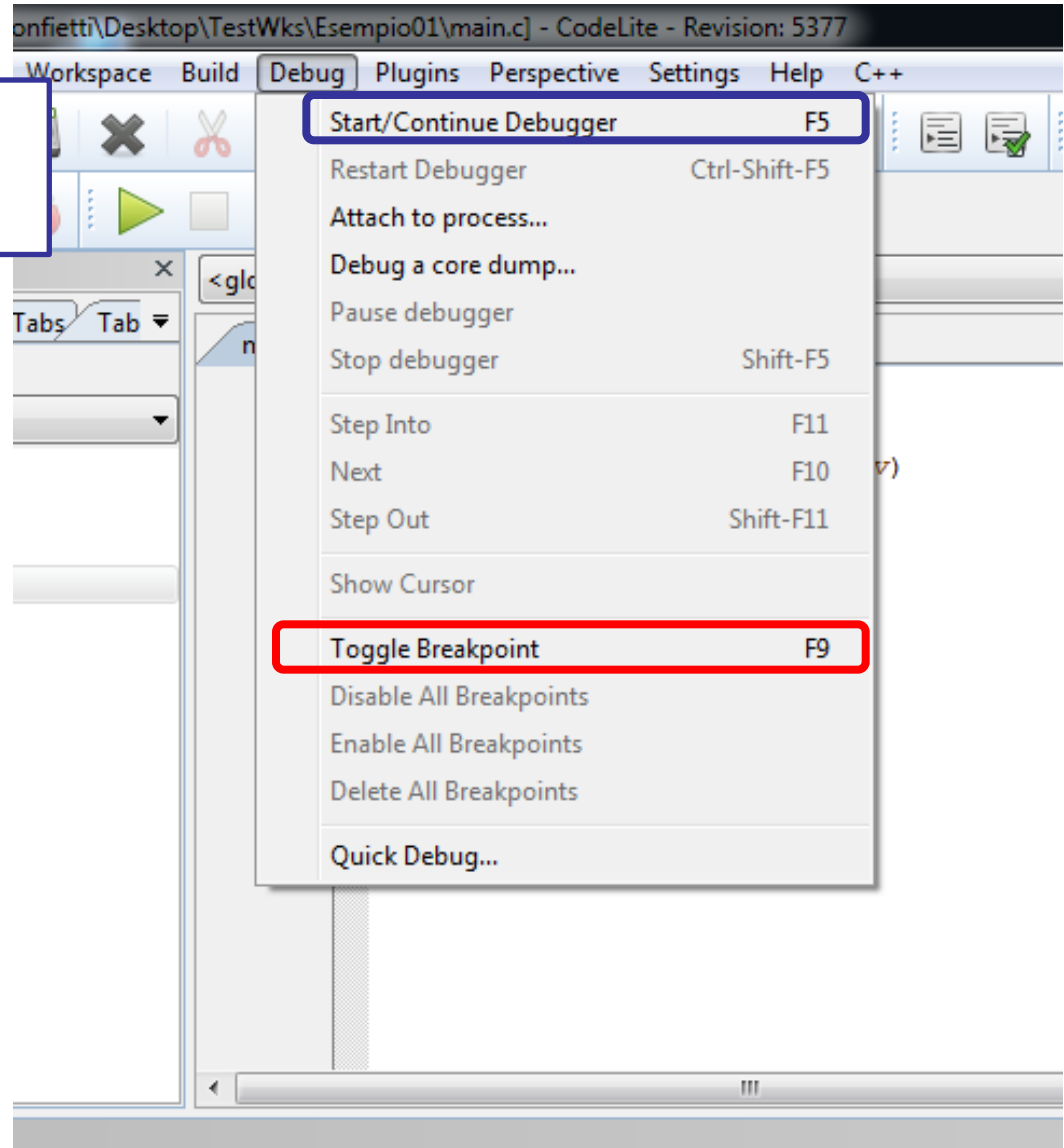
Sia **Codelite** sia altri ambienti di sviluppo incorporano un *debugger* con cui eseguire il programma,

- **riga per riga**
 - entrando anche dentro alle funzioni chiamate
 - oppure considerando le chiamate di funzione come una singola operazione
- oppure **inserendo breakpoints**

Progetti in CodeLite

Eseguire in modalità debug

Inserire un Breakpoint



Fase di Debugging

- **Prima di iniziare la sessione di debugging e' possibile inserire i cosiddetti *breakpoints***
 - *punti di interruzione nell'esecuzione del programma in cui il debugger fornisce una "fotografia" dello stato delle variabili*
- **Per inserire un breakpoint posizionare il cursore nel punto in cui si vuole fermare il debug e (alternative):**
 - *Utilizzare il comando da Menù*
 - *Premere F9*
 - *Singolo click a fianco del numero di riga*

Debugger

The image shows a screenshot of a debugger interface with several annotations:

- Comandi veloci Debug**: A red box highlights the top toolbar containing various debugging icons like play, stop, pause, and step-through.
- Debug Mode**: A black box highlights the 'Debug' button in the top-left toolbar.
- Indicatore di posizione Debug**: A blue box highlights a green arrow icon on the left side of the code editor, indicating the current execution position.
- Locals: Vista dello stato corrente di esecuzione Variabili-Valori-Tipo**: A green box highlights the 'Locals' tab and the variable list below it.

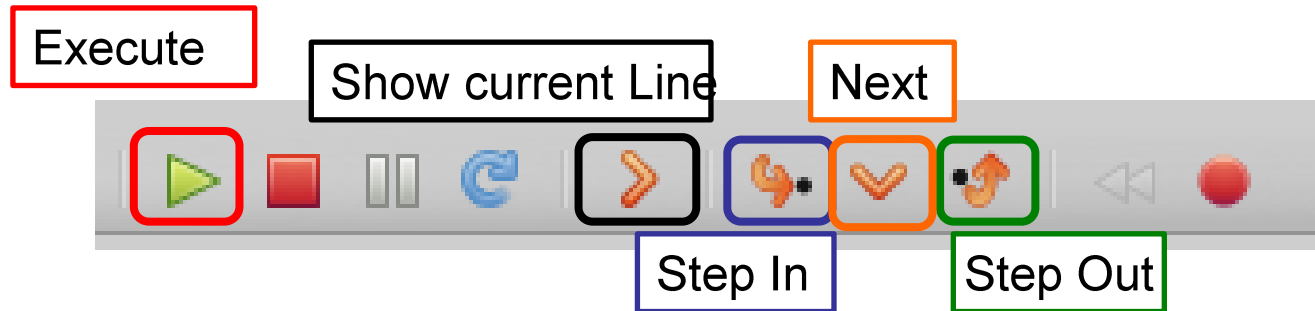
```
int main(int argc, char **argv)
{
    int i = 5;
    printf("hello wor
    i = 2;
    return 0;
}
```

Name	Value	Type
▶ argc	1	int
▶ argv	0x7fffffff5c8	char **
▶ i	5	int

Debugger: Come Procedere

- Nel menu Debug che compare quando il Debugger e' attivo ci sono alcune voci importanti:
 - **Execute**: esegue il programma fino al prossimo Debug
 - **Step in**: esegue passo passo le istruzioni di una funzione
 - **Step Out**: esegue l'istruzione e torna alla funziona chiamante
 - **Next**: esegue l'istruzione corrente
 - **Show current line**: permette di posizionare il cursore in una determinata posizione nel sorgente e esegue tutte le istruzioni fino ad arrestarsi al cursore.

Debugger



Debugger

The image shows a debugger interface with a code editor and a memory view. The code editor displays the following C code:

```
int main(int argc, char **argv)
{
    int i = 5;
    char C[4] = {'a', 'b', 'c', '\0'};
    printf("hello world\n");
    i = 2;
    return 0;
}
```

The line `char C[4] = {'a', 'b', 'c', '\0'};` is highlighted with a red box. Below the code editor, the Debugger window shows the memory view for the variable `C`. The memory view is also highlighted with a red box and contains the following data:

Name	Value	Type
argv	0x7711m3e30	char
▼ C	[4]	char [4]
0	97 'a'	char
1	98 'b'	char
2	99 'c'	char
3	0 "	char

Rappresentazione
Array statici

Debugger

The screenshot shows a debugger interface with a C program being debugged. The program code is as follows:

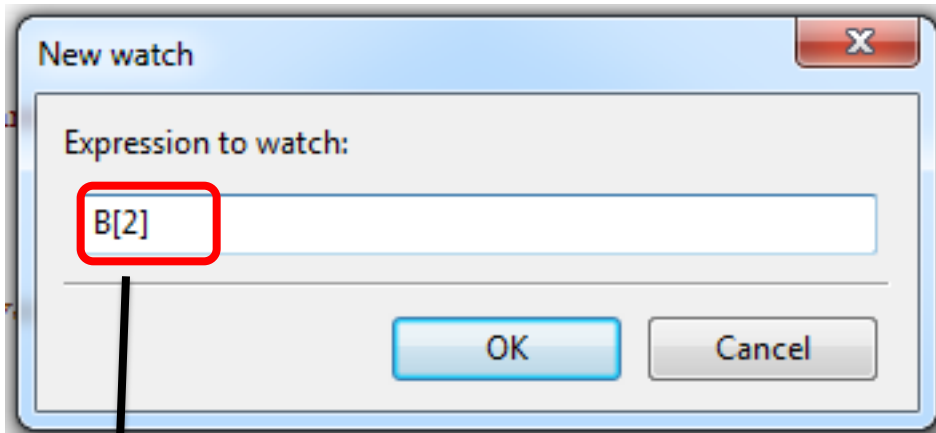
```
10 int var3 = 2;
11 char A[4] = {'a', 'b', 'c', '\0'};
12 int * B;
13
14 B = (int*)malloc(sizeof(int)*4);
15
16 B[0] = 5;
17 B[2] = 12;
18
19 printf("Programma di esempio\n");
20
21 stampa(var);
22 stampa(var2);
23 var3 = somma(var, var2);
24 stampa(var3);
25
26 return 0;
27 }
```

The line `B = (int*)malloc(sizeof(int)*4);` is highlighted with a red box. A callout box points to the `Watches` tab in the debugger, containing the text: "Rappresentazione parti non in stack: Watches".

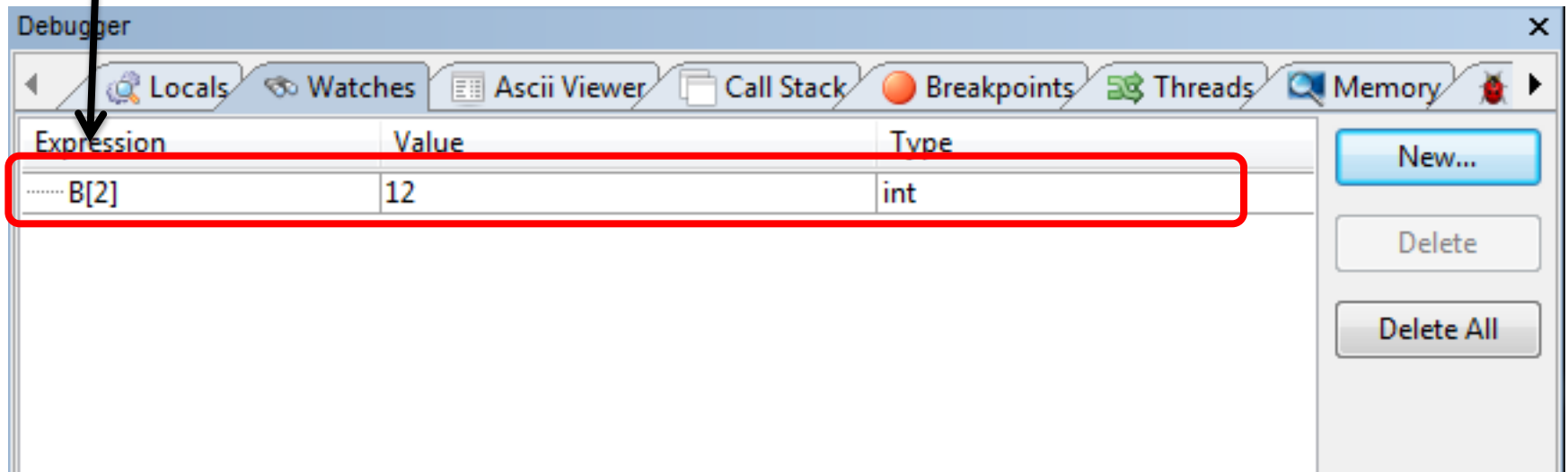
The `Watches` tab is active, showing a table of variables being monitored:

Name	Value	Type
B	0x3d1038	int *
.....*B	5	int
var	1	int
var2	5	int
var3	2	int
A	{...}	char [4]

Debugger



Rappresentazione
parti non in
stack:Watches



Note per l'installazione

- Nelle successive slide troverete le informazioni dettagliate per installare Codelite sulle vostre macchine, a seconda del sistema operativo utilizzato
- In realtà, per facilitarvi le cose, è possibile utilizzare delle macchine virtuali in cui Codelite è già stato installato e su cui potete iniziare a programmare senza ulteriori complicazioni
 - Per informazioni sulle macchine virtuali e su come installarle/usarle fare riferimento alle slide poste sul sito, in particolare nella sezione relativa al laboratorio trovate il riferimento “Installazione su macchina virtuale”

Mac OS X Notes

Per funzionare, Codelite, necessita del compilatore.

Per verificare se il compilatore è installato, aprire il *terminale* (si trova in /Applicazioni/Utility) e digitare (senza \$):

```
$ gcc
```

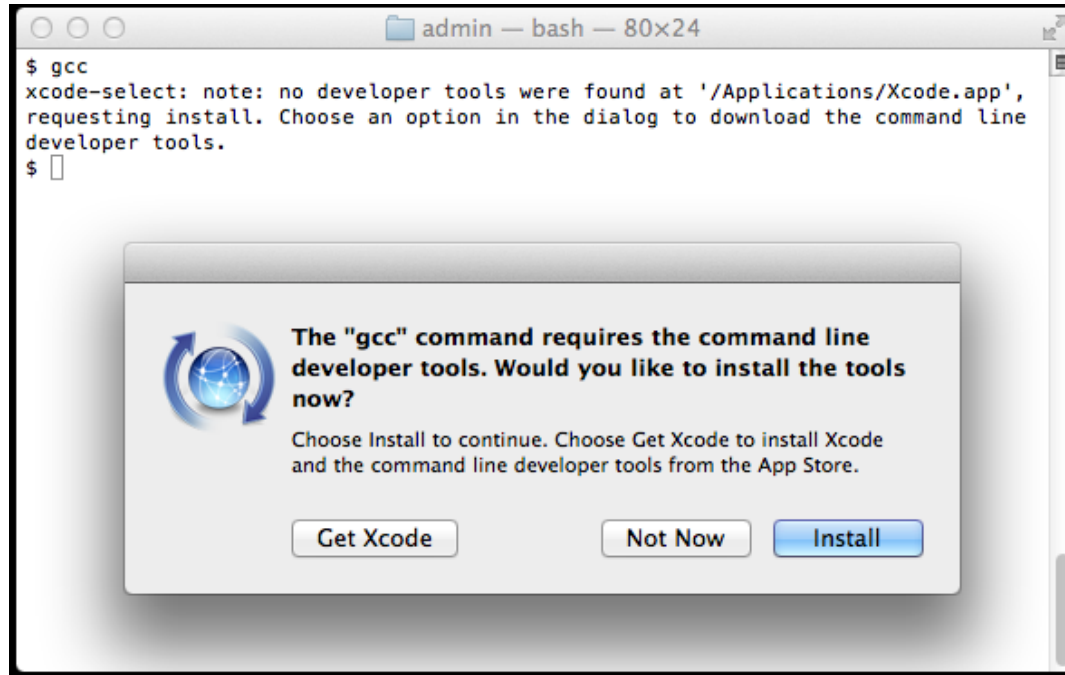
Se vi appare una scritta simile a questa va tutto bene:

```
clang: error: no input files
```

Significa che il compilatore è già installato

Mac OS X 10.10 Yosemite Notes

Altrimenti apparirà una finestra di installazione, tipo:

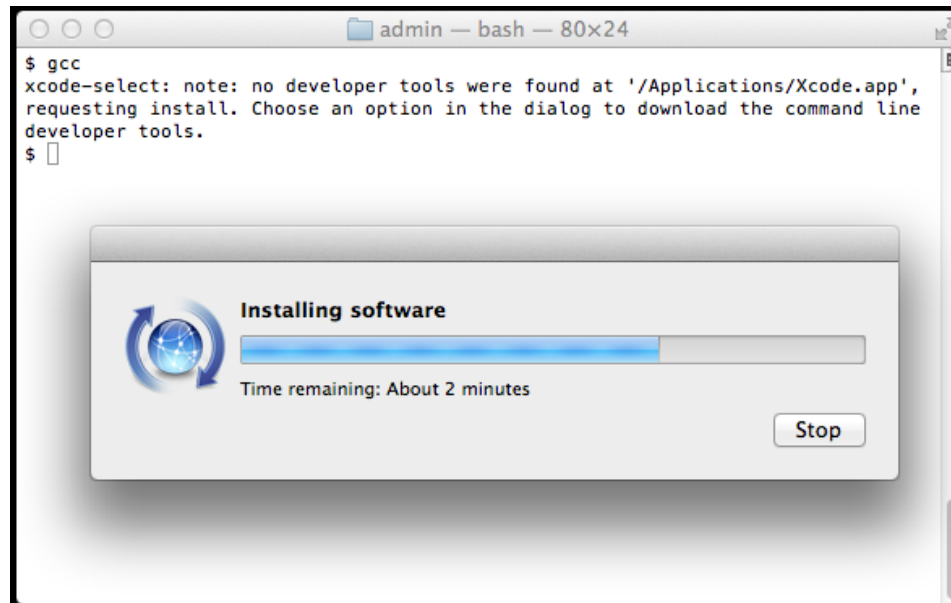


Cliccate **Install** per installare il compilatore (command line tools)

Cliccando *Get Xcode* verrà installato l'intero ambiente di sviluppo Mac Xcode

NOTA: Per eseguire Codelite NON è necessario Xcode ma solo il pacchetto command line tools

Mac OS X 10.10 Yosemite Notes



Per verificare se è installato correttamente digitare nel terminale:

```
$ xcode-select -p
```

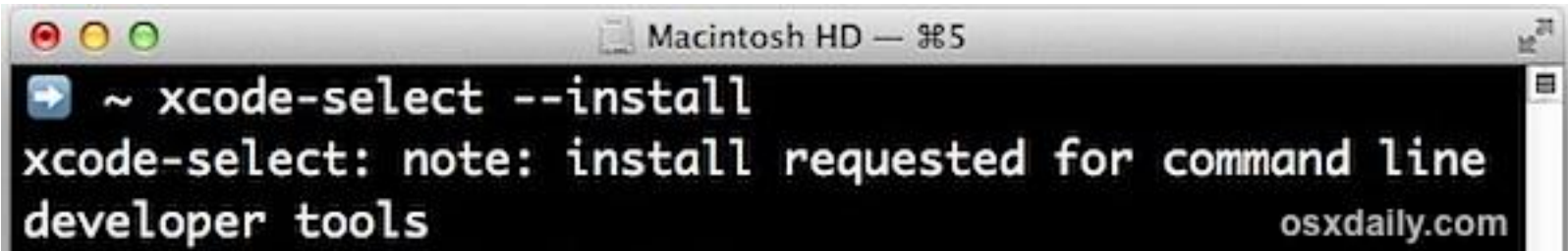
Si dovrebbe leggere una scritta tipo:

```
/Library/Developer/CommandLineTools
```

Mac OS X Notes

Se non appare la finestra di installazione provare a digirare nel terminale:

```
$ xcode-select -install
```

A screenshot of a Mac OS X terminal window. The title bar shows "Macintosh HD" and a keyboard shortcut. The terminal content shows a command being executed and its output.

```
~ xcode-select --install  
xcode-select: note: install requested for command line  
developer tools  
osxdaily.com
```

NOTA: se ancora non funziona, usare Google per risolvere il problema.

Mac OS X Notes

Al termine dell'installazione selezionare dal menù "Settings"->"Build settings", nella scheda "Compiler", cliccare sul menù "Add compilers" e selezionare "Scan computer for installed compilers"->"OK"->"OK".

Se il problema persiste, eliminare il workspace e crearne uno nuovo

Debian/Ubuntu Notes

Potete trovare la guida all'installazione nel sito:

[http://codelite.org/LiteEditor/Repositories#
toc1](http://codelite.org/LiteEditor/Repositories#toc1)

Da qualsiasi versione di Debian/Ubuntu, aprire il terminale e eseguire i seguenti comandi:

```
$ sudo apt-get purge codelite codelite-plugins
```

```
$ sudo apt-key adv --fetch-keys
```

<http://repos.codelite.org/CodeLite.asc>

Sempre da terminale ottenere il nome della vostra distribuzione per scegliere la giusta repository:

```
$ cat /etc/*-release | grep "DISTRIB_CODENAME="
```


Debian/Ubuntu Notes

In base al risultato del comando precedente eseguire:

- **Wheezy**

```
$ sudo apt-add-repository 'deb  
http://repos.codelite.org/ubuntu/ wheezy contrib'
```

- **Jessie**

```
$ sudo apt-add-repository 'deb  
http://repos.codelite.org/ubuntu/ Jessie contrib'
```

- **Trusty**

```
$ sudo apt-add-repository 'deb  
http://repos.codelite.org/ubuntu/ trusty universe'
```

- **Utopic**

```
$ sudo apt-add-repository 'deb  
http://repos.codelite.org/ubuntu/ utopic universe'
```

In fine eseguire sempre da terminale:

```
$ sudo apt-get update
```

```
$ sudo apt-get install codelite wxcrafter
```

Windows Notes

Prima di installare CodeLite è necessario installare diversi pacchetti MinGW:

- Scaricare MinGW dal sito <http://sourceforge.net/projects/mingw/>
- Installare MinGW
- All'interno di MinGW selezionare i pacchetti mingw-developer-toolkit, mingw-base, mingw-gcc-g++, mingw-make (tutti i pacchetti mingw-make)

Una volta installato MinGW è possibile procedere con l'installazione di codelite

Windows Notes

Per verificare quale versione installare (32 o 64 bit), da “Pannello di controllo”, selezionare “Sistema” quindi leggere la versione del Sistema operativo:

Visualizza informazioni di base relative al computer

Edizione Windows

Windows 8.1 Pro

© 2013 Microsoft Corporation. Tutti i diritti riservati.



Ancora più funzionalità con una nuova edizione di Windows

Sistema

Processore:	Intel(R) Core(TM) i5-3210M CPU @ 2.50GHz 2.50 GHz
Memoria installata (RAM):	8,00 GB (7,60 GB utilizzabile)
Tipo sistema:	Sistema operativo a 64 bit, processore basato su x64
Penna e tocco:	Nessun input penna o tocco disponibile per questo schermo

Windows Notes

Al termine dell'installazione selezionare dal menù "Settings"->"Build settings", nella scheda "Compiler", cliccare sul menù "Add compilers" e selezionare "Scan computer for installed compilers"->"OK"->"OK".

Se il problema persiste, eliminare il workspace e crearne uno nuovo

Windows Notes

Provate a creare un nuovo progetto, compilarlo ed eseguirlo. Se tutto funziona, siete a posto.

Altrimenti..



Una volta creato il progetto dovrete inserire nelle opzioni del linker `-static-libgcc -static-libstdc++`

Tasto destro del mouse sul progetto, “Settings”, “Common settings”, “Linker”, alla voce “Linker Options” inserire `-static-libgcc -static-libstdc++`

Windows & Linux Notes

ATTENZIONE: Su Windows e Linux, CodeLite non controlla la presenza dei diritti di scrittura sulla cartella di salvataggio, controllare preventivamente la presenza dei diritti.

N.B.: In laboratorio le cartelle di Windows in cui CodeLite può salvare sono C:\Temp e Desktop

Esercizio

Copiare e provare il seguente programma

```
#include <stdio.h>
int main(int argc, char **argv)
{
    int base, altezza, area;
    printf("Calcolo area rettangolo\n");
    printf("Inserire la larghezza del rettangolo:\n");
    scanf("%d",&base);
    printf("Inserire l'altezza del rettangolo:\n");
    scanf("%d",&altezza);
    area = base * altezza;
    printf("Il rettangolo ha area uguale a %d\n", area);
    getchar();
    return 0;
}
```