INFORMÁTICA INDUSTRIAL
INDUSTRIAL COMPUTING

BASICS OF
PROGRAMMING WITH C++

Professor: Mohamed Abderrahim/Juan Gonzalez
Departamento de Ingeniería de Sistemas y Automática

1. INTRODUCTION

Introduction – Program Structure

C++

```
#include <iostream>


int main()
{
    std::cout << "\n Hello World”  << std::endl;
              return 0;
}
```

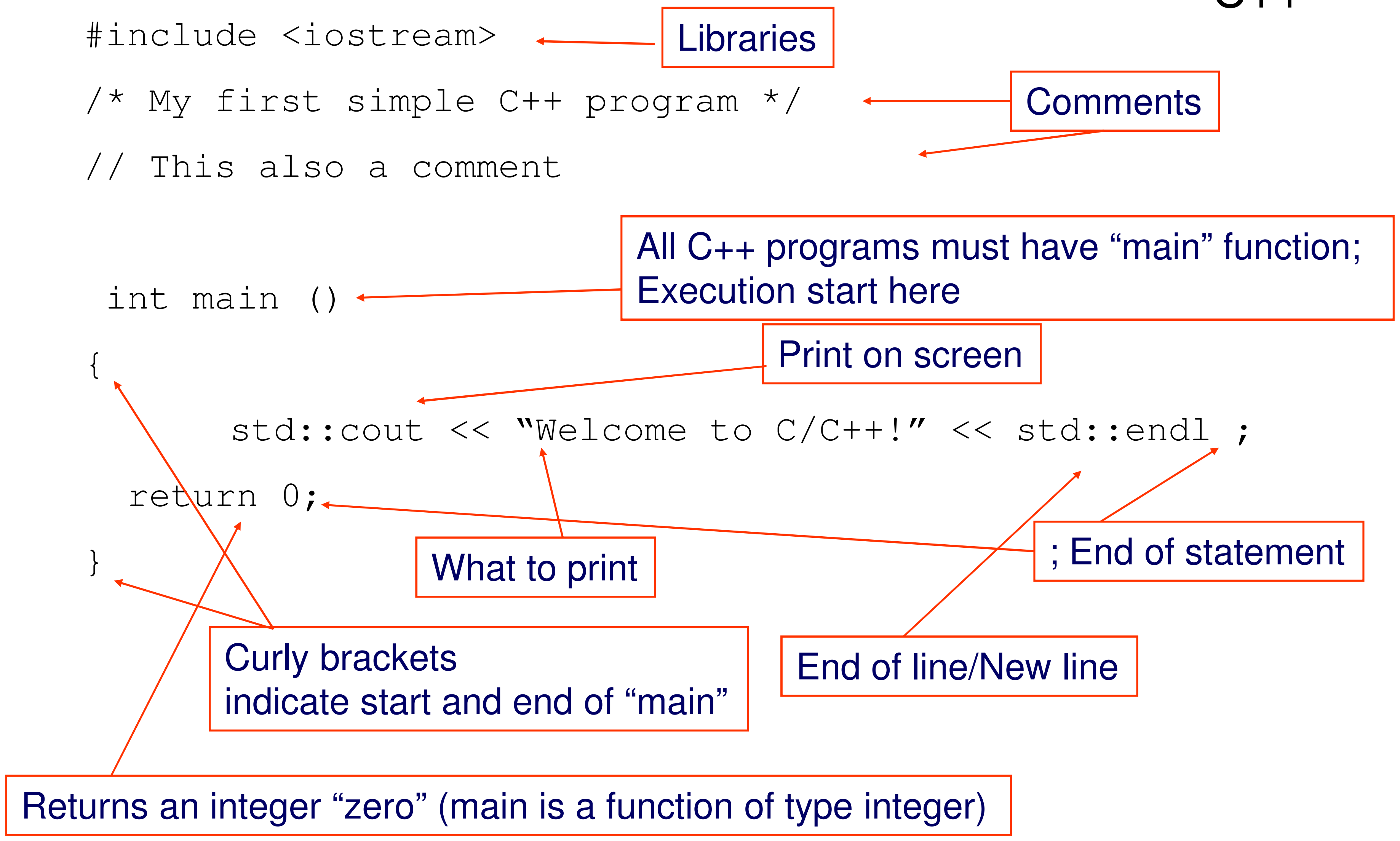
Introduction – Program Structure

C++

```
#include <iostream>

 using namespace std;
int main()
{
    cout << "\n Hello World"  << endl;
            return 0;
}
```

Introduction – Program Structure

C++



```
#include <iostream>

/* My first simple C++ program */

// This also a comment


 int main ()

{

        std::cout << "Welcome to C/C++!" << std::endl ;

  return 0;

}
```

Libraries
Comments
All C++ programs must have “main” function; Execution start here
Print on screen
What to print
; End of statement
End of line/New line
Curly brackets indicate start and end of “main”
Returns an integer “zero” (main is a function of type integer)

Introduction – Program Structure

```
#include <iostream>
#include <stdio.h>
    using namespace std;
/* Example program:  C comment */
// This is line comment in  C++
 int main()
{
          int num;

          num=1;

          cout <<"numero es " << num << endl;
          printf("numero es %d\n",num);
          return 0;

}
```

statement to pre-processor

Comments

Name of function

Declaration statement

Assign value;

Method invocation (call)

Function statement (call)

Use de tabulators

End of Sentence **;**

Use curly brackets

Introduction - COMPILING

– MS-DOS / Windows

• Text Editors	1. Source Code (program.cpp)
• Compiler	2. Object Code (program.o)
• Linker	3. Executable(programa.exe)

– UNIX

• Text Editors	1. Source Code (program.cpp)
• Compiler	2. Object Code (program.o)
• Linker	3. Executable (a.out)

Basic Structure of a C++ program

Example: Hola Mundo

Algorithm:

Show “¡Hola Mundo!”

Program C++:

```
#include <iostream>

using namespace std;
int main()
{
		cout<<"¡Hola Mundo!";

		return 0;
}
```

Basic Structure of a C program

Example: Hola Mundo

Include declarations of the Standard Library of Input and Output streams

C++ Program:

```
#include <iostream>
using namespace std;
int main()
{
   cout<<"¡Hola Mundo!";

   return 0;
}
```

Basic Structure of a C program

Ejemplo: Hola Mundo

Curly brackets for start and end of the function (bloc of statements)

C++ Program:

```
#include <iostream>
using namespace std;
int main()
{
   cout<<"¡Hola Mundo!";

   return 0;
}
```

Basic Structure of a C program

Ejemplo: Hola Mundo

Programa C:

```
#include <iostream>
using namespace std;
int main()
{
   cout<<"¡Hola Mundo!";

   return 0;
}
```

statement (**Function Call**) for writing "¡Hola Mundo!"

Basic Structure of a C program

Ejemplo: Hola Mundo

Statements ends with **semi column** (;)

Programa C:

```
#include <iostream>
using namespace std;
int main()
{
   cout<<"¡Hola mundo!";

   return 0;
}
```

Basic Structure of a C program

Ejemplo: Hola Mundo

Programa C:

```
#include <iostream>
using namespace std;
int main()
{
   cout<<"¡Hola mundo!";

   return 0;
}
```

Return zero (main is a function of type integer)

Example: Count until 10

Show the numbers from 0 to 9

```
Fix cuenta to 0
While (cuenta is less than 10 )
{
    Show cuenta
    add 1 to cuenta
}
```

```
int main()
{


   return 0;
}
```

Example: Count until 10

Show the numbers from 0 to 9

```
Fix the cuenta to 0
While (cuenta is less than 10 )
{
    Show cuenta
    add 1 to cuenta
}
```

```
#include <iostream>
 using namespace std;

int main()
{



  return 0;
}
```

Example: Count until 10

Mostrar los números del 0 al 9

```
Fix the cuenta to 0
While (cuenta is less than 10 )
{
    Show cuenta
    add 1 to cuenta
}
```

```
#include <iostream>
using namespace std;
/* Print out numbers 0 to 9 */

int main()
{



  return 0;
}
```

Comment

Example: Count until 10

Show the numbers from 0 to 9

```
Fix the cuenta to 0
While (cuenta is less than 10
    )
{
    Show cuenta
    add 1 to cuenta
}
```

```
#include <iostream>
using namespace std;

/* Print out numbers 0 to 9 */
int main()
{
   int cuenta;



  return 0;
}
```

Variable Declaration

Example: Count until 10

Show the numbers from 0 to 9

```
Fix the cuenta to 0
While (cuenta is less than 10 )
{
    Show cuenta
    add 1 to cuenta
}
```

```
#include <iostream>
using namespace std;

/* Print out numbers 0 to 9 */
int main()
{
   int cuenta;

   cuenta = 0;
   while ( cuenta < 10 )
   {
       cout << cuenta<< endl ;
       cuenta=cuenta+1;
   }
  return 0;
}
```

Example: Count until 10

Show the numbers from 0 to 9

```
Fix cuenta to 0
While (cuenta is less than 10 )
{
    Show cuenta
    add 1 to cuenta
}
```

```
#include <iostream>
using namespace std;

/* Print out numbers 0 to 9 */
int main()
{
   int cuenta;

   cuenta = 0;
   while   cuenta < 10 )
   {
                              l ;
}
```

Assign a value to a variable
(Left to right)

Example: Count until 10

Show the numbers from 0 to 9

```
Fix the cuenta to 0
While (cuenta is less than 10 )
{
    Show cuenta
    add 1 to cuenta
}
```

```
#include <iostream>
using namespace std;

/* Print out numbers 0 to 9 */
int main()
{
   int cuenta;

   cuenta = 0;
   while ( cuenta < 10 )
   {
       cout << cuenta<     endl ;
       cuenta=cuenta
   }
  return 0;
}
```

¡No semi column! (punto-y-coma)

Example: Count until 10

Show the numbers from 0 to 9

```
Fix the cuenta to 0
While (cuenta is less than 10 )
{
    Show cuenta
    add 1 to cuenta
}
```

```cpp
#include <iostream>
using namespace std;

/* Print out numbers 0 to 9 */
int main()
{
   int cuenta;

   cuenta = 0;
   while ( cuenta < 10 )
   {
       cout << cuenta<< endl ;
       cuenta=cuenta+1;
   }
  return 0;
}
```

Example: ¿What is the sign?

Find the sign of a number

```
Show "Type a number "

introduce num

if (num is less than 0)
then
{
    Show num " is -'ve"
}
otherwise
{
    Show num " es +'ve"
}
```

```
#include <iostream>
// Find the sign of a number
int main()
{
   float num;
   std::cout <<"Type a number :  " ;
   std::cin >> num;

   if ( num < 0 )
   {
      std::cout << num <<" is negative\n"
   ;
   }
   else
   {
      std::cout<< num << " is positive\n" ;
   }
   return 0;
}
```

2. Data Types

Basic Types

- ***bool*** bolean (true/false)
- ***char*** characters
- ***int*** integers
- ***float*** decimals
- ***short*** "short" integers
- ***long*** "long" integers
- ***double*** decimals with double precision

Types

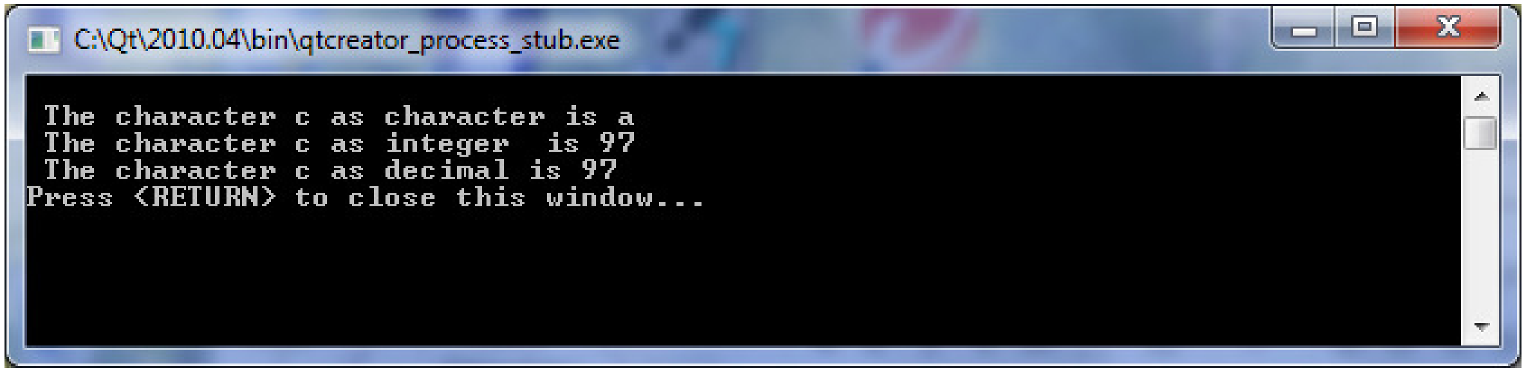
- It is important to choose well the type of variable: An int does not stores decimals and a float cannot allow integer remainder.
- For each variable, it is important to use the correct conversion code in functions.

Representaction & Conversion

```
cout << static_cast<int>(1.7);              // displays 1

cout << static_cast<double>(1) / 2;         // displays 0.5


cout << (int)1.7;                           // displays 1
```

Representaction & Conversion

```
Include<iostream>
 using namespace std;


 int main()
{
            char c ;
            c = ‘a’ ;
            cout << “\n The character c as character is ” << c ;
            cout << “\n The character c as integer  is ” <<  (int)c ;
            cout << “\n The character c as decimal is ” << (float) c << endl;

            return 0;
}
```

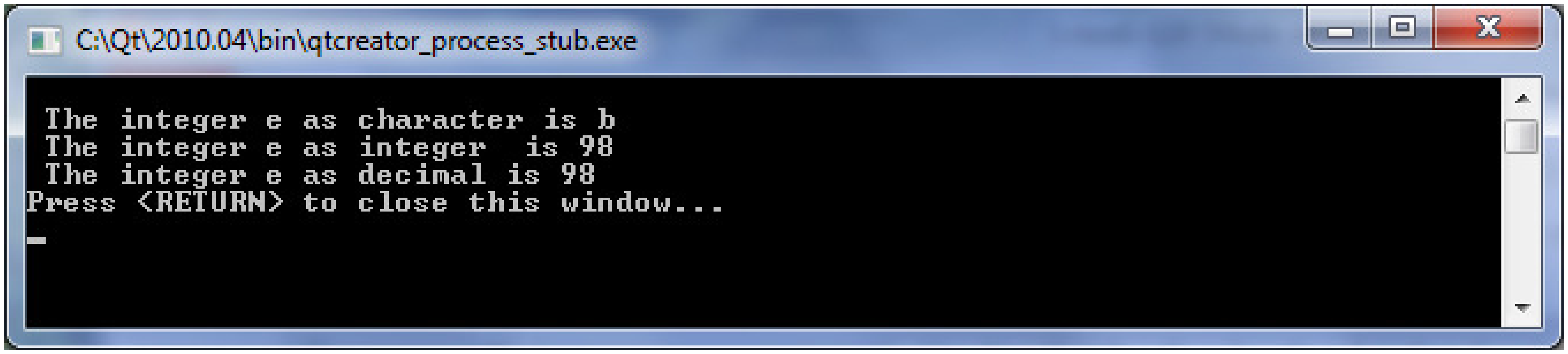


C:\Qt\2010.04\bin\qtcreator_process_stub.exe

```

 The character c as character is a
 The character c as integer  is 97
 The character c as decimal is 97
Press <RETURN> to close this window...
```

Representaction & Conversion

```
Include<iostream>
 using namespace std;


 int main()
{
              int e=98 ;

              cout << "\n The integer e as character is " << (char)e ;
              cout << "\n The integer e as integer  is " <<  (int)e ;
              cout << "\n The integer e as decimal is " << (float )e << endl ;

             return 0;
}
```

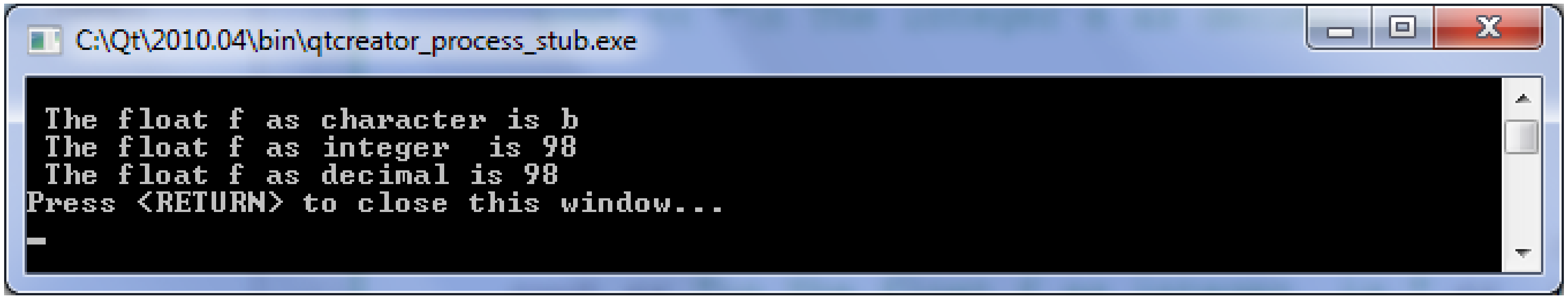


C:\Qt\2010.04\bin\qtcreator_process_stub.exe

```

 The integer e as character is b
 The integer e as integer  is 98
 The integer e as decimal is 98
Press <RETURN> to close this window...
_
```

Representaction & Conversion

```
Include<iostream>
 using namespace std;

int main()
{
               float f=98. ;

               cout << "\n The float f as character is " << (char) f ;
               cout << "\n The float f as integer  is " <<  (int) f ;
               cout << "\n The float f as decimal is " << (float ) f << endl ;

              return 0;
}
```



C:\Qt\2010.04\bin\qtcreator_process_stub.exe

```
 The float f as character is b
 The float f as integer  is 98
 The float f as decimal is 98
Press <RETURN> to close this window...
_
```

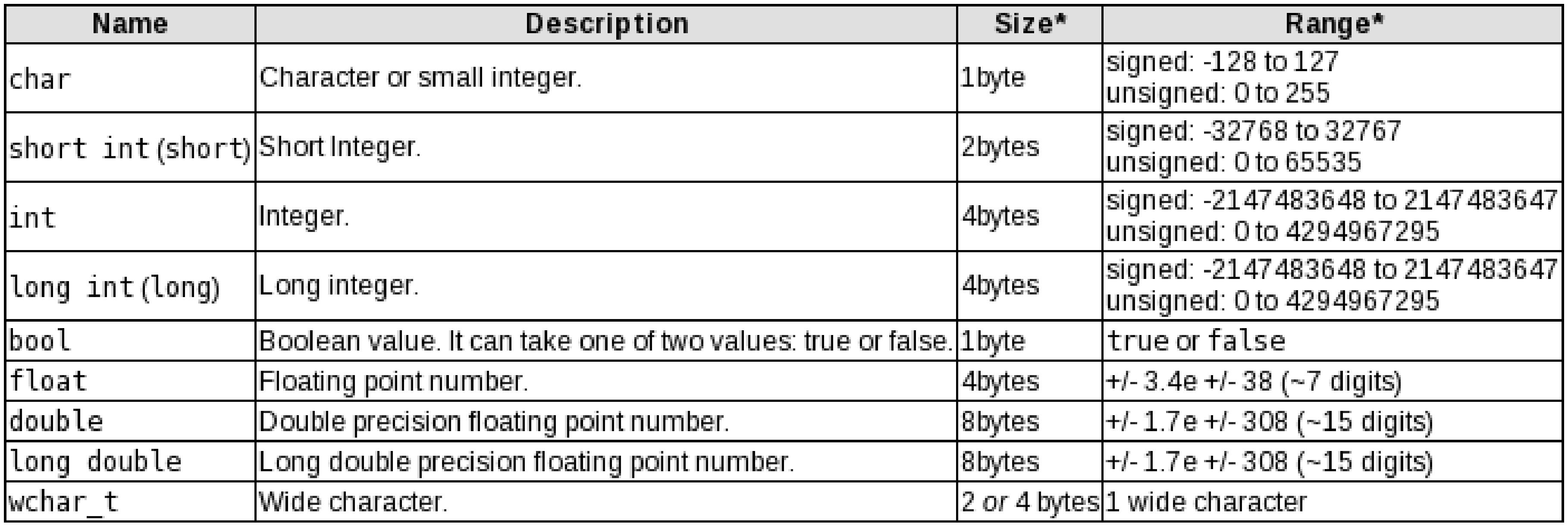
Basic Types

Size of types

	Ms-Dos	Unix
– char	8 bits	8 bits
– int	16bits	32bits
– float	32bits	32bits
– short	16bits	16bits
– long	32bits	32bits
– double	64bits	64bits
– bool	8 bits	8 bits

Basic Types

Range of types



Name	Description	Size*	Range*
`char`	Character or small integer.	1byte	signed: -128 to 127 unsigned: 0 to 255
`short int` (`short`)	Short Integer.	2bytes	signed: -32768 to 32767 unsigned: 0 to 65535
`int`	Integer.	4bytes	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
`long int` (`long`)	Long integer.	4bytes	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
`bool`	Boolean value. It can take one of two values: true or false.	1byte	`true` or `false`
`float`	Floating point number.	4bytes	+/- 3.4e +/- 38 (~7 digits)
`double`	Double precision floating point number.	8bytes	+/- 1.7e +/- 308 (~15 digits)
`long double`	Long double precision floating point number.	8bytes	+/- 1.7e +/- 308 (~15 digits)
`wchar_t`	Wide character.	2 *or* 4 bytes	1 wide character

• Table from http://www.cplusplus.com/doc/tutorial/variables/
• The data en columns *Size* and *Range* depend on the system and the architecture to which they were compiled.
• These values are common for the 32-bits architecture.

Basic Types

Range of types

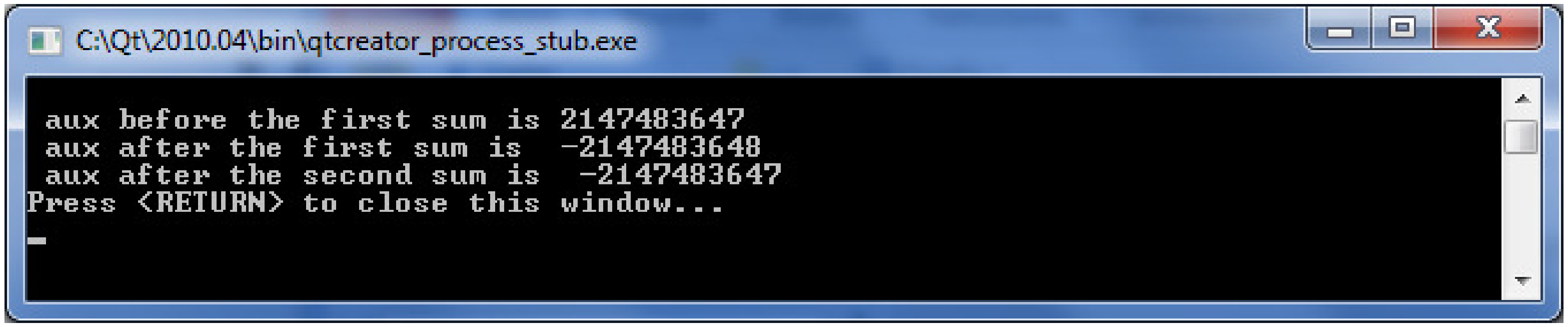
- If an integer is 4 bytes (32 bits)
  - From $-2^{31} = -2.147.483.648$
  - To $2^{31}-1 = 2.147.483.647$
    - » (passing by zero)

```
#include <iostream>

using namespace std;
/* Example program  */
int main()
{
        int num, Num;

        num=1;

        cout<<“number is "<<num<<endl;

        return 0;
}
```

- ***OJO*** overflow (desbordamiento)
- ***OJO*** capitals (mayúsculas)

Range of Types

```
Include<iostream>
 using namespace std;

int main()
{
            int aux=2147483647 ;

          cout << "\n aux before the first sum is " << aux ;
          aux= aux+1;
          cout << "\n aux after the first sum is " << aux ;
          aux= aux+1;
          cout << "\n aux after the second sum is " << aux << endl  ;

          return 0;
}
```

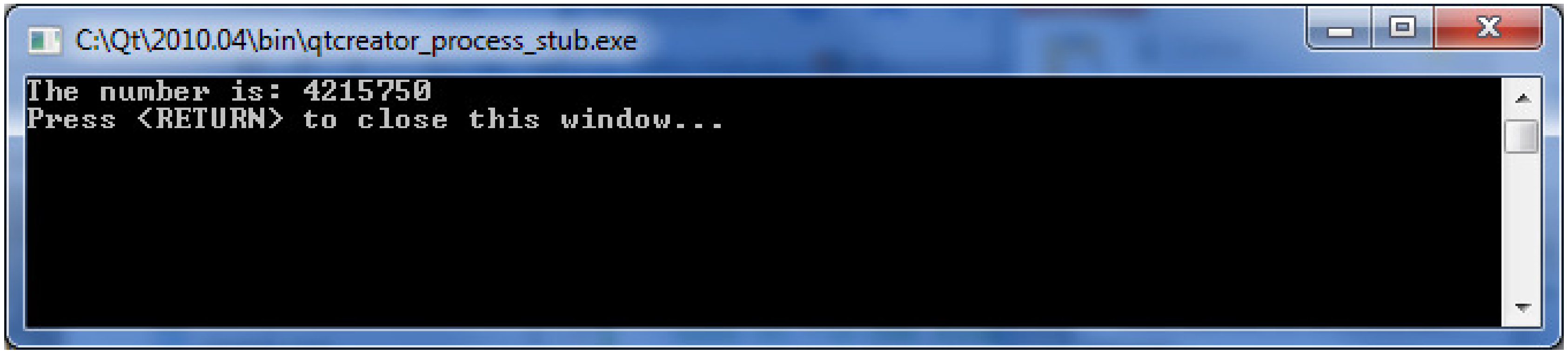

C:\Qt\2010.04\bin\qtcreator_process_stub.exe

```

 aux before the first sum is 2147483647
 aux after the first sum is  -2147483648
 aux after the second sum is  -2147483647
Press <RETURN> to close this window...
_
```

Range of Types

```
Include<iostream>
 using namespace std;

int main()
{
        int num, Num;
        num = 1 ;
        cout << "The number is: " << Num << endl ;

        return 0;
}
```

Initialize variables before use!


C:\Qt\2010.04\bin\qtcreator_process_stub.exe

```
The number is: 4215750
Press <RETURN> to close this window...
```

char type

- Adequate for:
  - characters
  - integers: -128/127
- The Values are defined by the ASCII code
  - ´a´ is 97
  - ´A´ is 65
  - ´2´ is 50 **not** 2

Constants/Literals

- Integers/Decimals

– 42	char/int
– 42L	long int
– 4.2F	float

- If begins with zero = octal
  - 042 is actually 34: 4*8+2.
- If begins with 0x =hexadecimal
  - 0x42 is actually 66: 4*16+2.

Sizeof

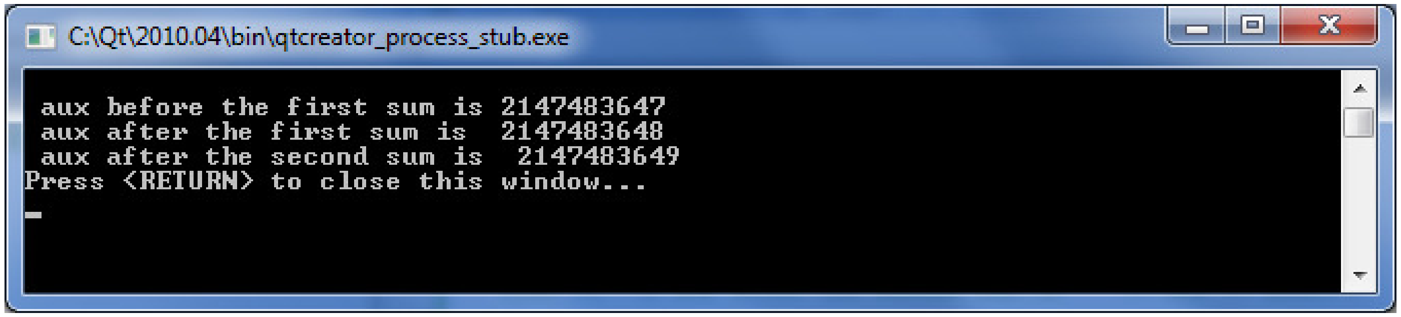
- The size of the integers depend on:
  - Operating System
  - Compiler
- sizeof() returns the number of bytes
  - sizeof(int)

UNSIGNED

- Indicate that all the numbers are going to be positive
- Increases the range of the variable
- Only positive numbers
  - unsigned char a   a=0-255
  - unsigned int a   a=0-4.294.967.295
  - unsigned short int a
  - unsigned long int a

UNSIGNED

```
Include<iostream>
 using namespace std;

int main()
{
                  unsigned int aux=2147483647 ;

                cout << "\n aux before the first sum is " << aux ;
                aux= aux+1;
                cout << "\n aux after the first sum is " << aux ;
                aux= aux+1;
                cout << "\n aux after the second sum is " << aux << endl  ;

                return 0;
}
```


C:\Qt\2010.04\bin\qtcreator_process_stub.exe

```
 aux before the first sum is 2147483647
 aux after the first sum is  2147483648
 aux after the second sum is  2147483649
Press <RETURN> to close this window...
_
```

Conversions and Cast

- If an expression contains variables of different types they would be converted *automatically* into the widest (biggest)type present in the expression.

  char ☑ short ☑ int ☑ long ☑ float ☑ double

- With the = sign the result will be converted to the type of the variable on the left of the sign

  if *i* is an integer variable i=3/9.0 ☑ i is equal 0

- The *cast* is an explicit conversion

  Despite that f is a float, f=3/9 ☑f=0

  f=3/**(float)**9 ☑f=3/9.0=0.333

Variables

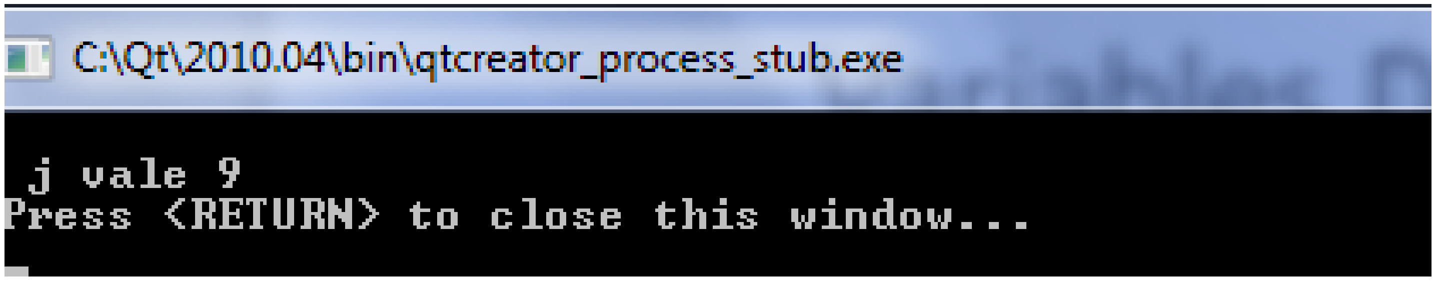
- Entities that contains values.
- They must be declared before using them
  - Inform the compiler
  - Reserve memory
- The names can be up to 254 characters
  - Only the first 31 used
- Characters that can be used A-Z a-z 0-9 y _
- Must begin with alphabetic chars

Variables Declaration

- Is a line (statement) with type followed by the name of the variable.
  - int i;
- More than one variable of the same type can be declared with same statement, separating them by comas.

Variables Declaration

```
#include <iostream>

 using namespace std;

int main()
{
   int i, j ;
   i = 3;
   j = 3 * i ;
   cout << "\n j vale " <<  j << endl;
   return 0;
}
```



```
C:\Qt\2010.04\bin\qtcreator_process_stub.exe

 j vale 9
Press <RETURN> to close this window...
```

reserved Names

auto	break	case	char
const	continue	default	do
double	else	enum	extern
float	for	**goto**	if
int	long	register	return
short	signed	sizeof	static
struct	switch	typedef	union
unsigned	void	volatile	while

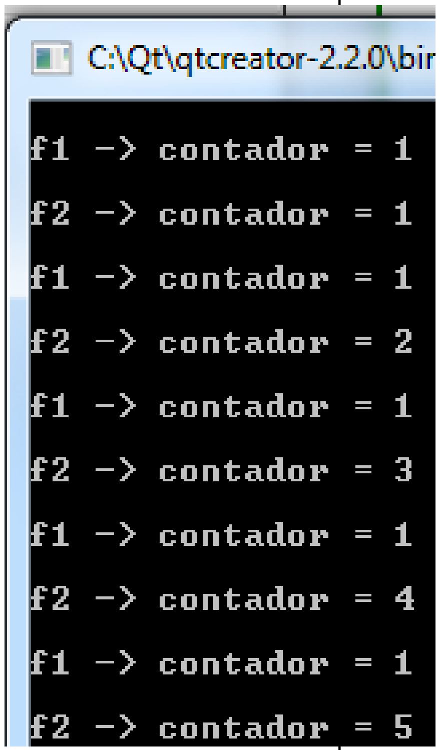
Variables' Atributes

- Type. When declared, int i;
- Scope!. Part of the program where it is used.
    - Inside function: from def. Until end of function.
    - Outside functions: from def. until end of file.
- Static/automatic
    – int i; static int j;
- Global/local
- Const: const double pi = 3.14159265;
- Register: Optimozation
- Volatile: interruptions

Example of *static* variable

```
#include <iostream>
 using namespace std;
void f1(void){
   int contador=0;
   cout << "\nf1 -> contador = " << ++contador << endl;
}

void f2(void){
   static int contador=0;
   cout << "\nf2 -> contador = " <<  ++contador <<endl;
}

…
```

```
…

int main()
{
   for(int i = 0; i < 5; i++ )
   {
      f1();
      f2();
   }
Return 0;
}
```


C:\Qt\qtcreator-2.2.0\bir

```
f1 -> contador = 1
f2 -> contador = 1
f1 -> contador = 1
f2 -> contador = 2
f1 -> contador = 1
f2 -> contador = 3
f1 -> contador = 1
f2 -> contador = 4
f1 -> contador = 1
f2 -> contador = 5
```

What is the result?

3. Mathematical Operators

Mathematical Operators

- Sum +
- subtraction -
- Multiplication *
- Division /
- Rest of division (remainder) (integers) %

Mathematical Operators

- precedence
  - Multiplication, division and rest **>** sum and subtraction
  - Multiplication = division=rest
  - sum = rest
    - Criterion: First operation from left
  - Use of parenthesis:
    - 5+4*3=5+12=17
    - (5+4)*3=9*3=27

Mathematical Operators

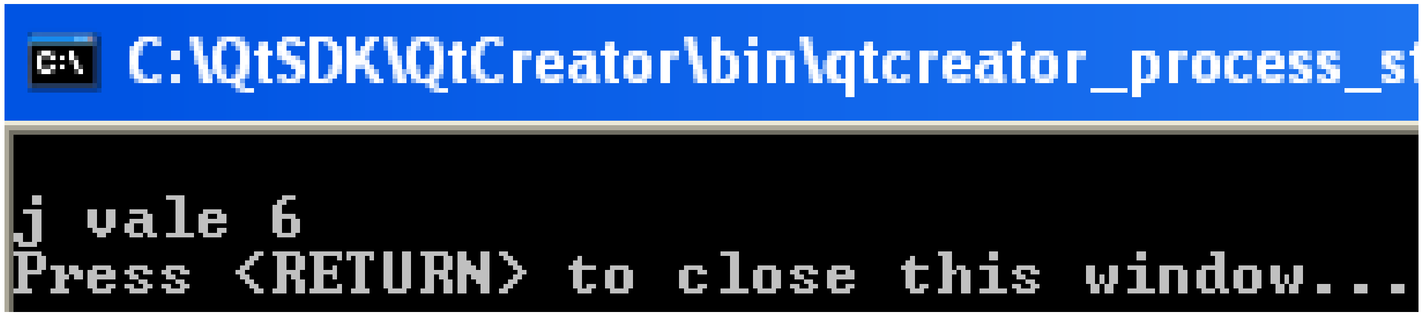
- Precedence and types

```
int i;
i=5*32/9;          i=160/9=17
i=5/9*32;          i=0*32=0
i=5/9.0*32;        i=0.555*32=17
```

Mathematical Operators

- Can be applied on constants
- Also on variables
- And combine both

```
#include <iostream>

 using namespace std;

int main()
{
   int i, j ;
   i = 3;
   j = 3 * i ;
   cout << "\n j vale " <<  j << endl;
   return 0;
}
```



C:\QtSDK\QtCreator\bin\qtcreator_process_st

```

j vale 6
Press <RETURN> to close this window...
```

++ -- Operators

++i is equivalent to i=i+1

i++ is equivalent to i=i+1

(i+j)++ ***is illegal***

--i is equivalent to i=i-1

i-- is equivalent to i=i-1

(i+j)-- ***is illegal***

- In expressions:
  - ++i first the sum and then assign
  - i++ Assign first and later execute sum (increment)

++ -- Operators

main()
{
int a,b,c;
a=b=c=0;
a=**++b**+ **++c**;
a=**b++** + **c++**;
a=**b --** + **--c**;
}

	a	b	c
a=b=c=0;	0	0	0
a=++b+ ++c;	?	?	?
++b	0	**1**	0
++c	0	1	**1**
a=b+c;	**2**	1	1
a=b+++c++;	?	?	?
a=b+c	**2**	1	1
b++	2	**2**	1
c++;	2	2	**2**
a=b--+ --c;	?	?	?
--c	2	2	**1**
a=b+c	**3**	2	1
b--;	3	**1**	1

Assignment Operators

- = += -= *= /=
- variable *assignment operator* expression

  k += 2
- variable = variable *operator* expression

  k = k + 2

  - Ej.:

  k*=2

  k*=3+x

Bit-wise Operations

- Complement to one **~**

```
~     00000101     (5)
=     11111010     (250)
```

- AND **&**

```
      00000101     (5)
      00000110     (6)
&     00000100     (4)
```

- OR **|**

```
      00000101     (5)
      00000110     (6)
|     00000111     (7)
```

- Exclusive OR **^**

```
      00000101     (5)
      00000110     (6)
^     00000011     (3)
```

Bit-wise Operations

- Displacement (shift operators)
  - variable *displacement* num_bits

i=5;

i=i << 2 (two bits to the left)

00000101 (5)

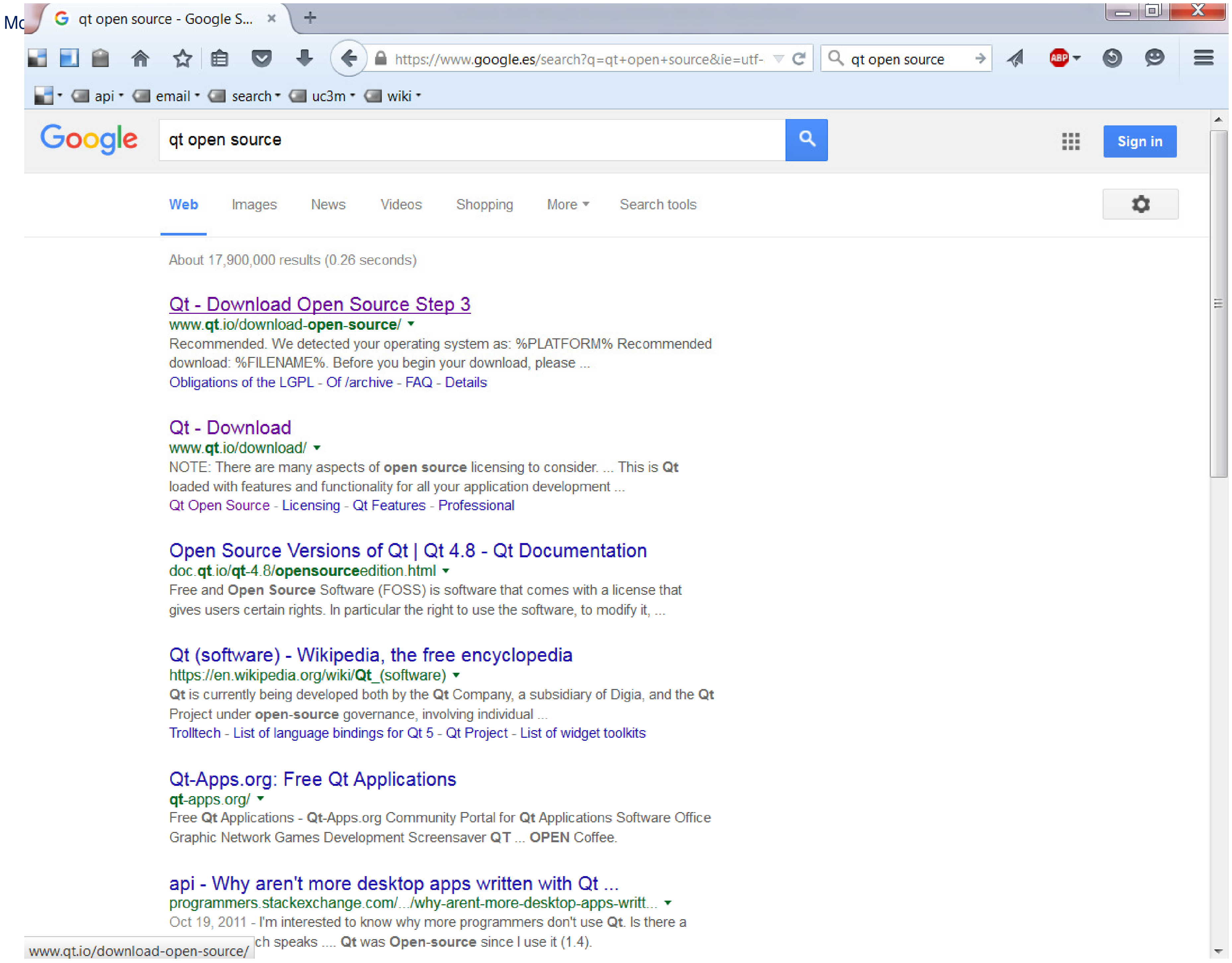
<< 2

i=i>>2 (two bits to the right)

00000101 (5)

>> 2

Download QT Creator



About 17,900,000 results (0.26 seconds)

Qt - Download Open Source Step 3
www.qt.io/download-open-source/
Recommended. We detected your operating system as: %PLATFORM% Recommended download: %FILENAME%. Before you begin your download, please ...
Obligations of the LGPL - Of /archive - FAQ - Details

Qt - Download
www.qt.io/download/
NOTE: There are many aspects of **open source** licensing to consider. ... This is **Qt** loaded with features and functionality for all your application development ...
Qt Open Source - Licensing - Qt Features - Professional

Open Source Versions of Qt | Qt 4.8 - Qt Documentation
doc.qt.io/qt-4.8/opensourceedition.html
Free and **Open Source** Software (FOSS) is software that comes with a license that gives users certain rights. In particular the right to use the software, to modify it, ...

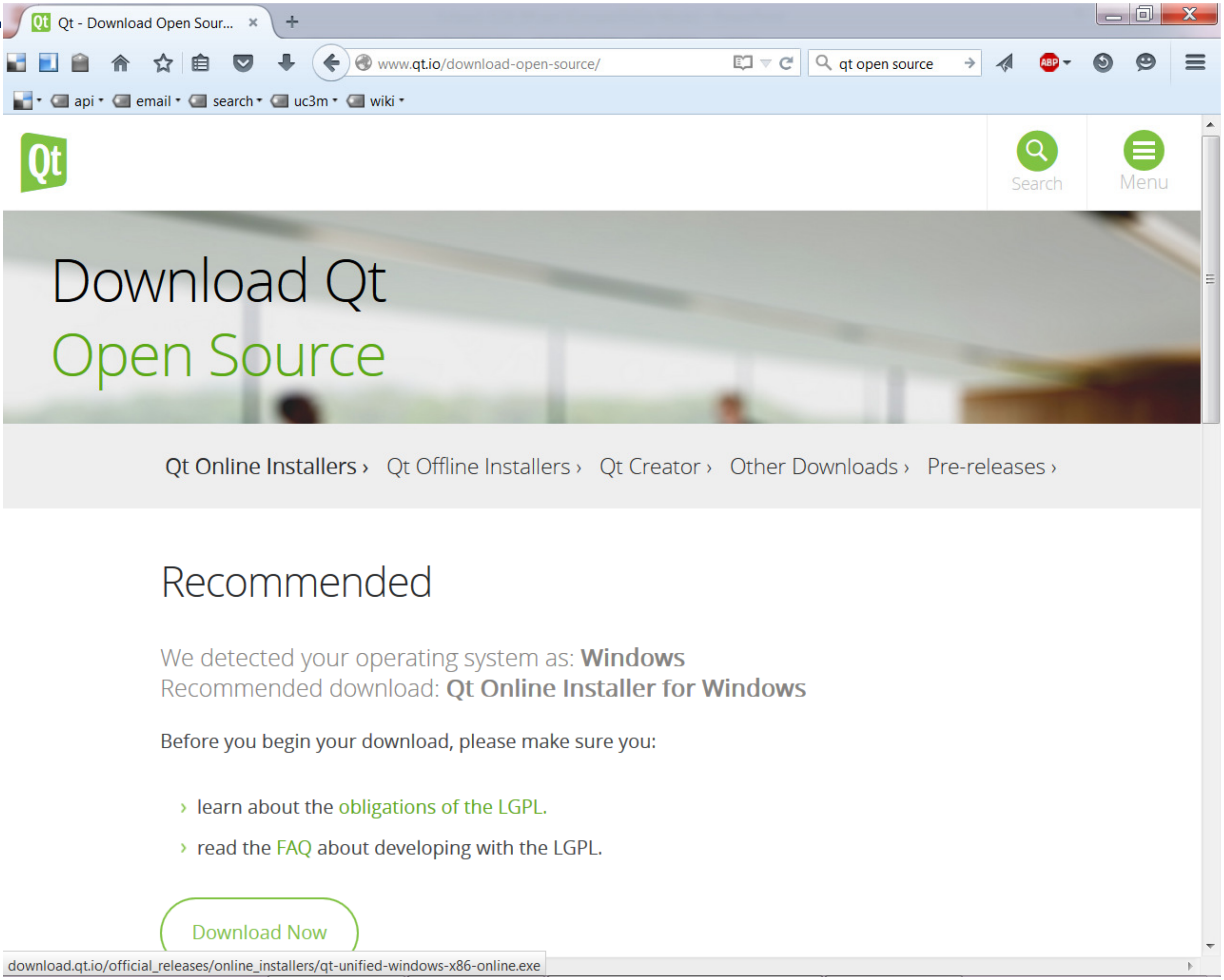
Qt (software) - Wikipedia, the free encyclopedia
https://en.wikipedia.org/wiki/Qt_(software)
Qt is currently being developed both by the **Qt** Company, a subsidiary of Digia, and the **Qt** Project under **open-source** governance, involving individual ...
Trolltech - List of language bindings for Qt 5 - Qt Project - List of widget toolkits

Qt-Apps.org: Free Qt Applications
qt-apps.org/
Free **Qt** Applications - **Qt**-Apps.org Community Portal for **Qt** Applications Software Office Graphic Network Games Development Screensaver **QT** ... **OPEN** Coffee.

api - Why aren't more desktop apps written with Qt ...
programmers.stackexchange.com/.../why-arent-more-desktop-apps-writt...
Oct 19, 2011 - I'm interested to know why more programmers don't use **Qt**. Is there a
ch speaks .... **Qt** was **Open-source** since I use it (1.4).

www.qt.io/download-open-source/

Mo



Download Qt
Open Source

Qt Online Installers › Qt Offline Installers › Qt Creator › Other Downloads › Pre-releases ›

Recommended

We detected your operating system as: **Windows**
Recommended download: **Qt Online Installer for Windows**

Before you begin your download, please make sure you:

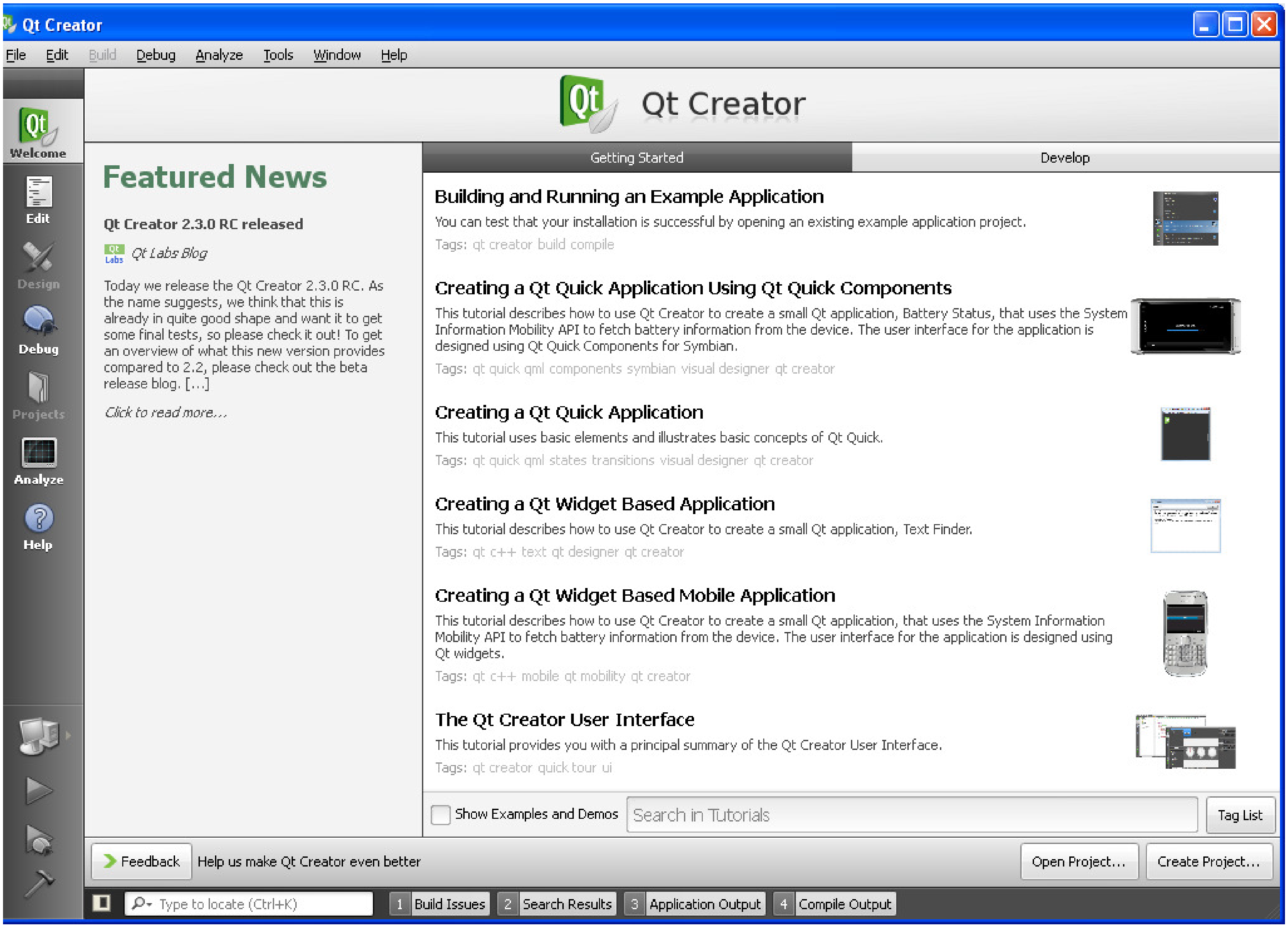
› learn about the obligations of the LGPL.

› read the FAQ about developing with the LGPL.

Download Now

download.qt.io/official_releases/online_installers/qt-unified-windows-x86-online.exe

Doing an example in QT Creator



Featured News

Qt Creator 2.3.0 RC released

Qt Labs Blog

Today we release the Qt Creator 2.3.0 RC. As the name suggests, we think that this is already in quite good shape and want it to get some final tests, so please check it out! To get an overview of what this new version provides compared to 2.2, please check out the beta release blog. […]

Click to read more…

Building and Running an Example Application

You can test that your installation is successful by opening an existing example application project.

Tags: qt creator build compile

Creating a Qt Quick Application Using Qt Quick Components

This tutorial describes how to use Qt Creator to create a small Qt application, Battery Status, that uses the System Information Mobility API to fetch battery information from the device. The user interface for the application is designed using Qt Quick Components for Symbian.

Tags: qt quick qml components symbian visual designer qt creator

Creating a Qt Quick Application

This tutorial uses basic elements and illustrates basic concepts of Qt Quick.

Tags: qt quick qml states transitions visual designer qt creator

Creating a Qt Widget Based Application

This tutorial describes how to use Qt Creator to create a small Qt application, Text Finder.

Tags: qt c++ text qt designer qt creator

Creating a Qt Widget Based Mobile Application

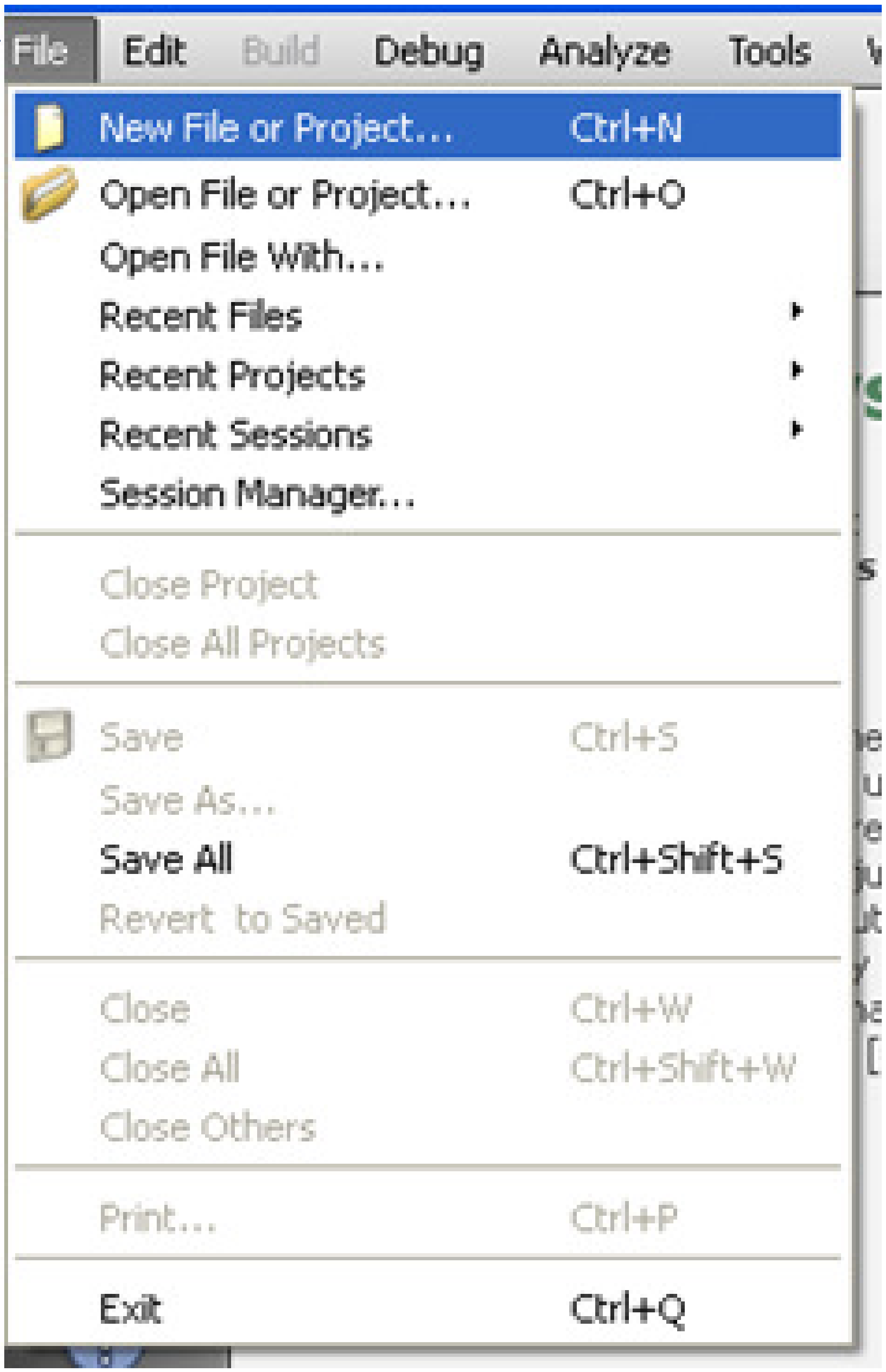
This tutorial describes how to use Qt Creator to create a small Qt application, that uses the System Information Mobility API to fetch battery information from the device. The user interface for the application is designed using Qt widgets.

Tags: qt c++ mobile qt mobility qt creator

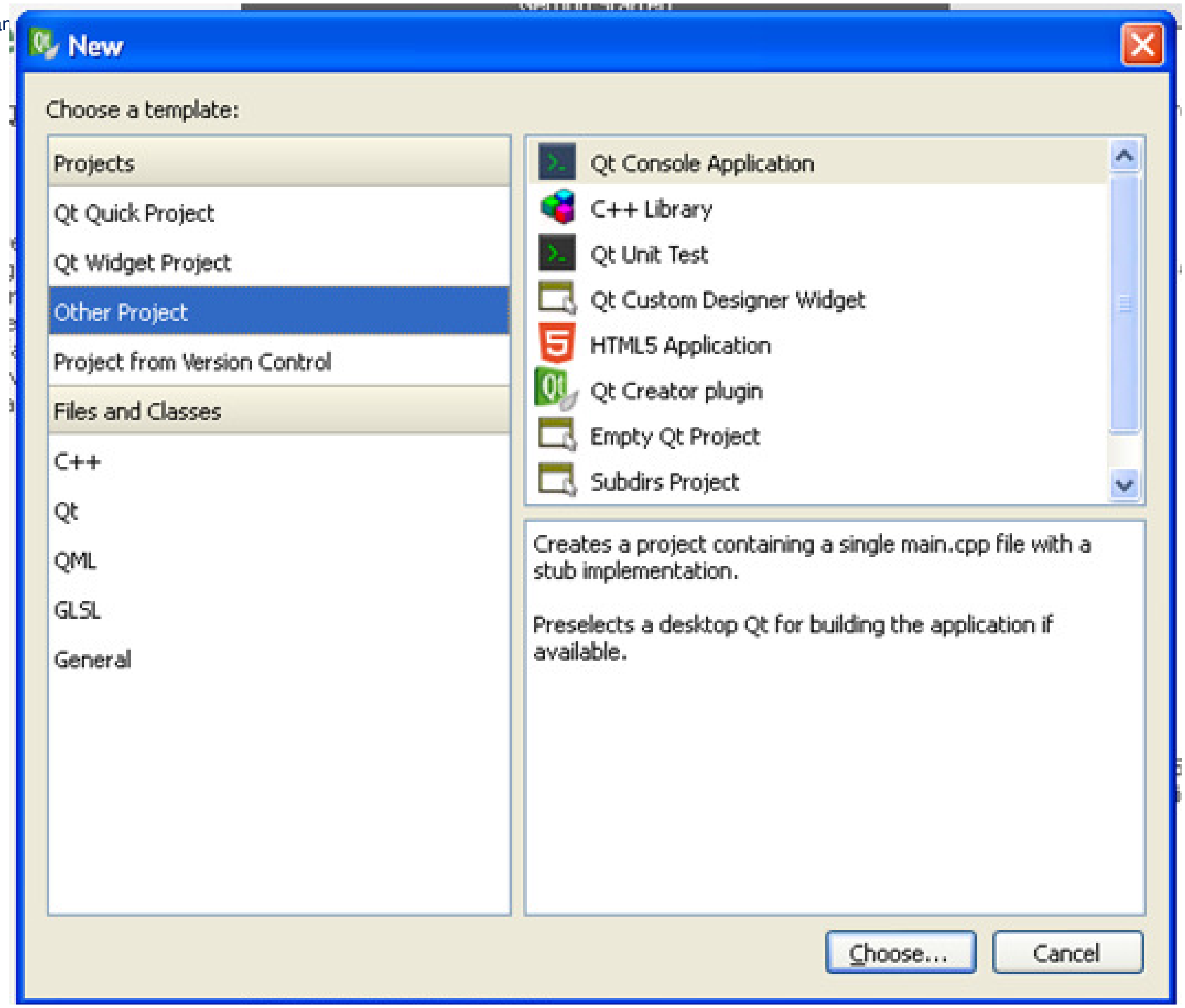
The Qt Creator User Interface

This tutorial provides you with a principal summary of the Qt Creator User Interface.

Tags: qt creator quick tour ui

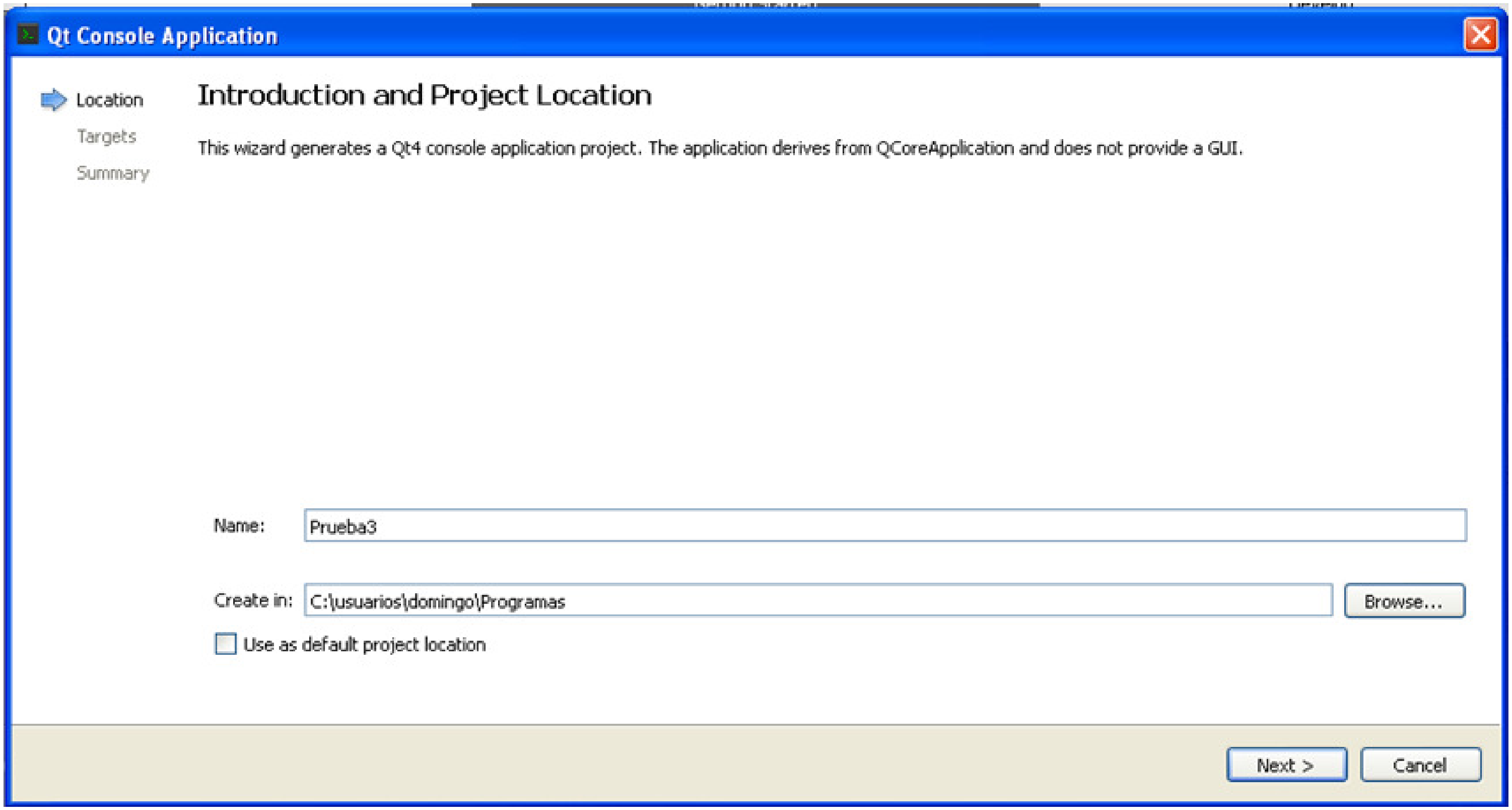

File
Edit
Build
Debug
Analyze
Tools
New File or Project... Ctrl+N
Open File or Project... Ctrl+O
Open File With...
Recent Files
Recent Projects
Recent Sessions
Session Manager...
Close Project
Close All Projects
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert to Saved
Close Ctrl+W
Close All Ctrl+Shift+W
Close Others
Print... Ctrl+P
Exit Ctrl+Q

Mohan



Creates a project containing a single main.cpp file with a stub implementation.

Preselects a desktop Qt for building the application if available.

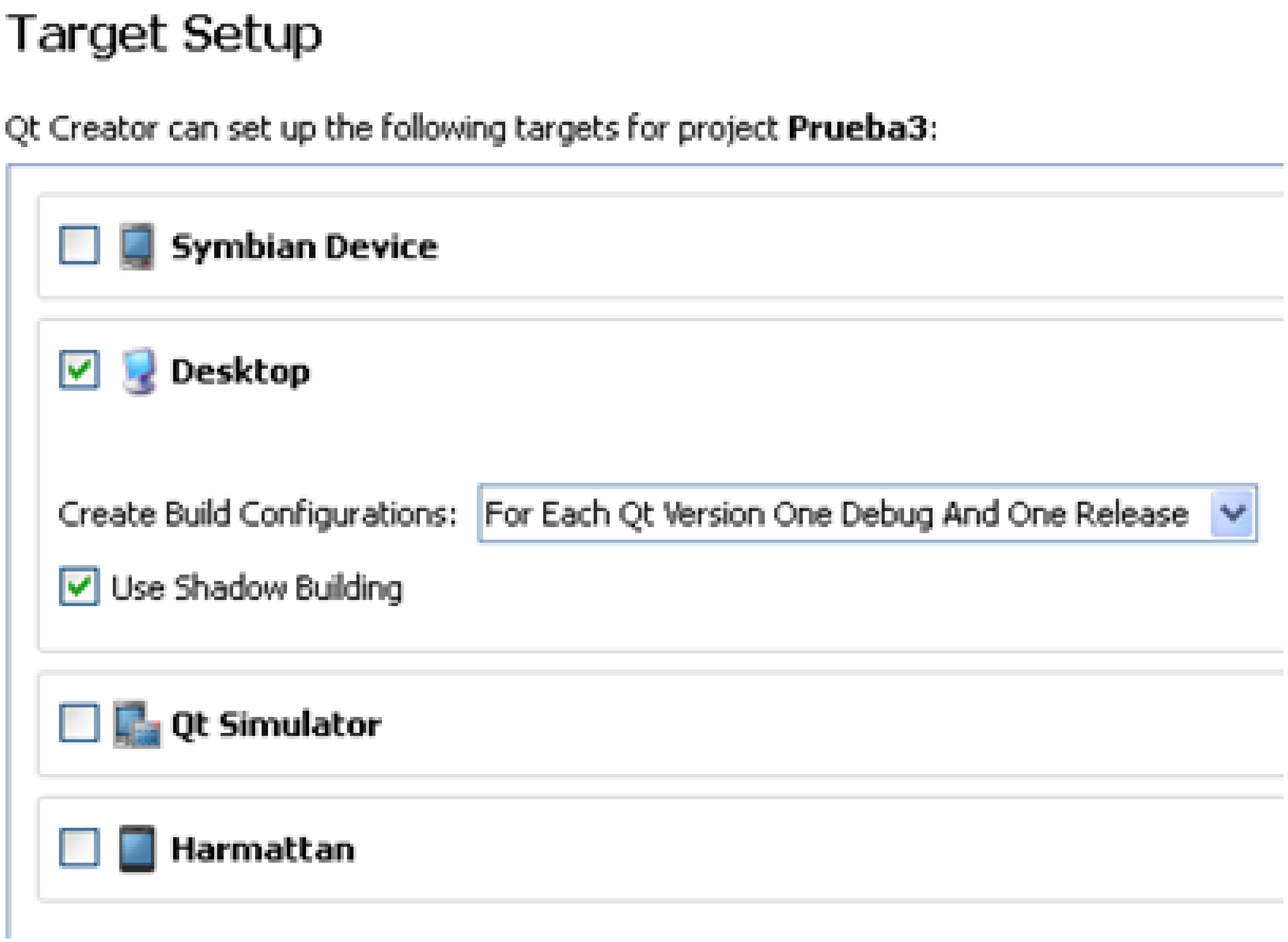


Introduction and Project Location

This wizard generates a Qt4 console application project. The application derives from QCoreApplication and does not provide a GUI.

Name: Prueba3

Create in: C:\usuarios\domingo\Programas



Target Setup

Qt Creator can set up the following targets for project **Prueba3**:

[ ] **Symbian Device**

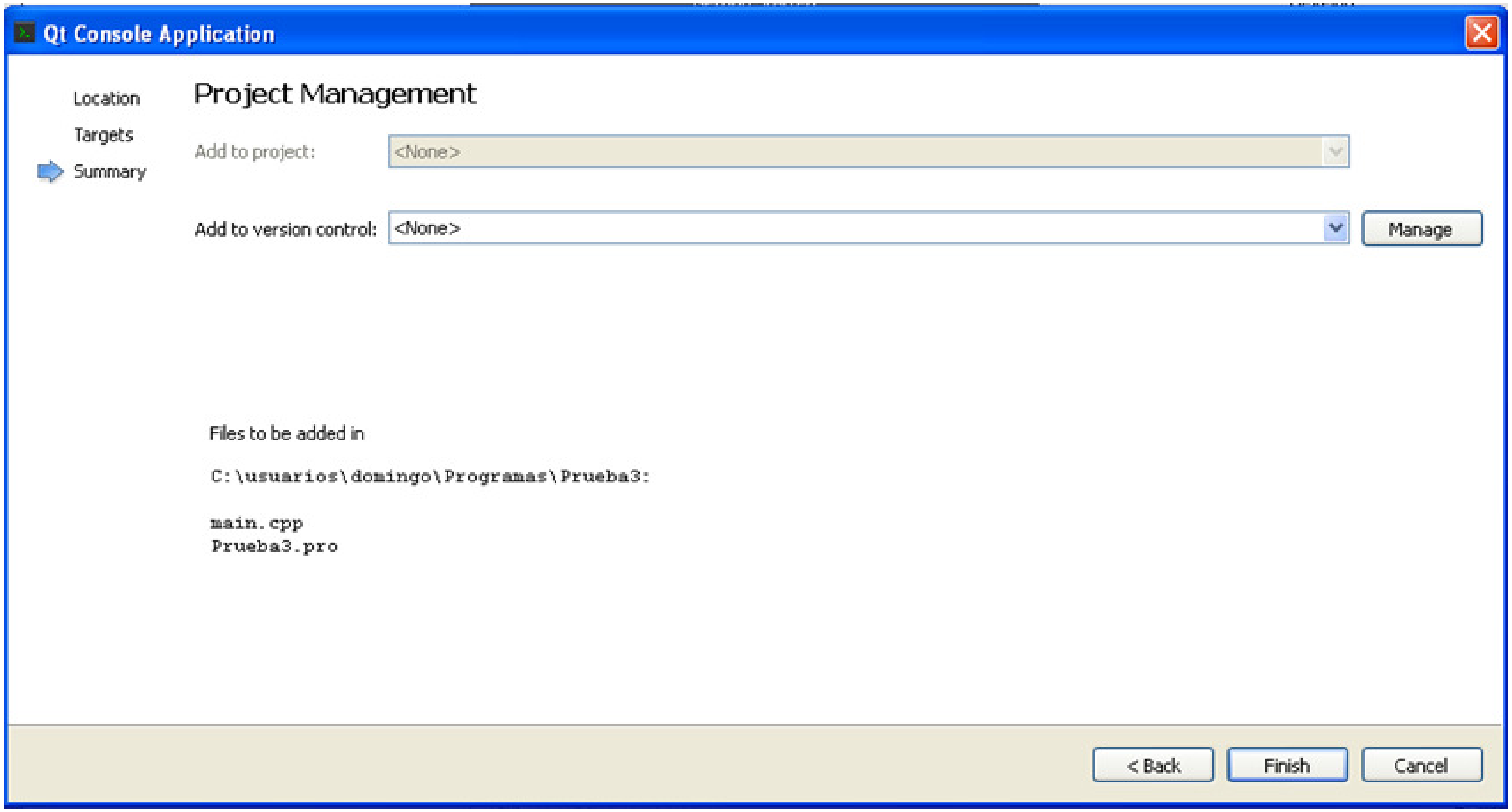
[✓] **Desktop**

Create Build Configurations: For Each Qt Version One Debug And One Release

[✓] Use Shadow Building

[ ] **Qt Simulator**

[ ] **Harmattan**



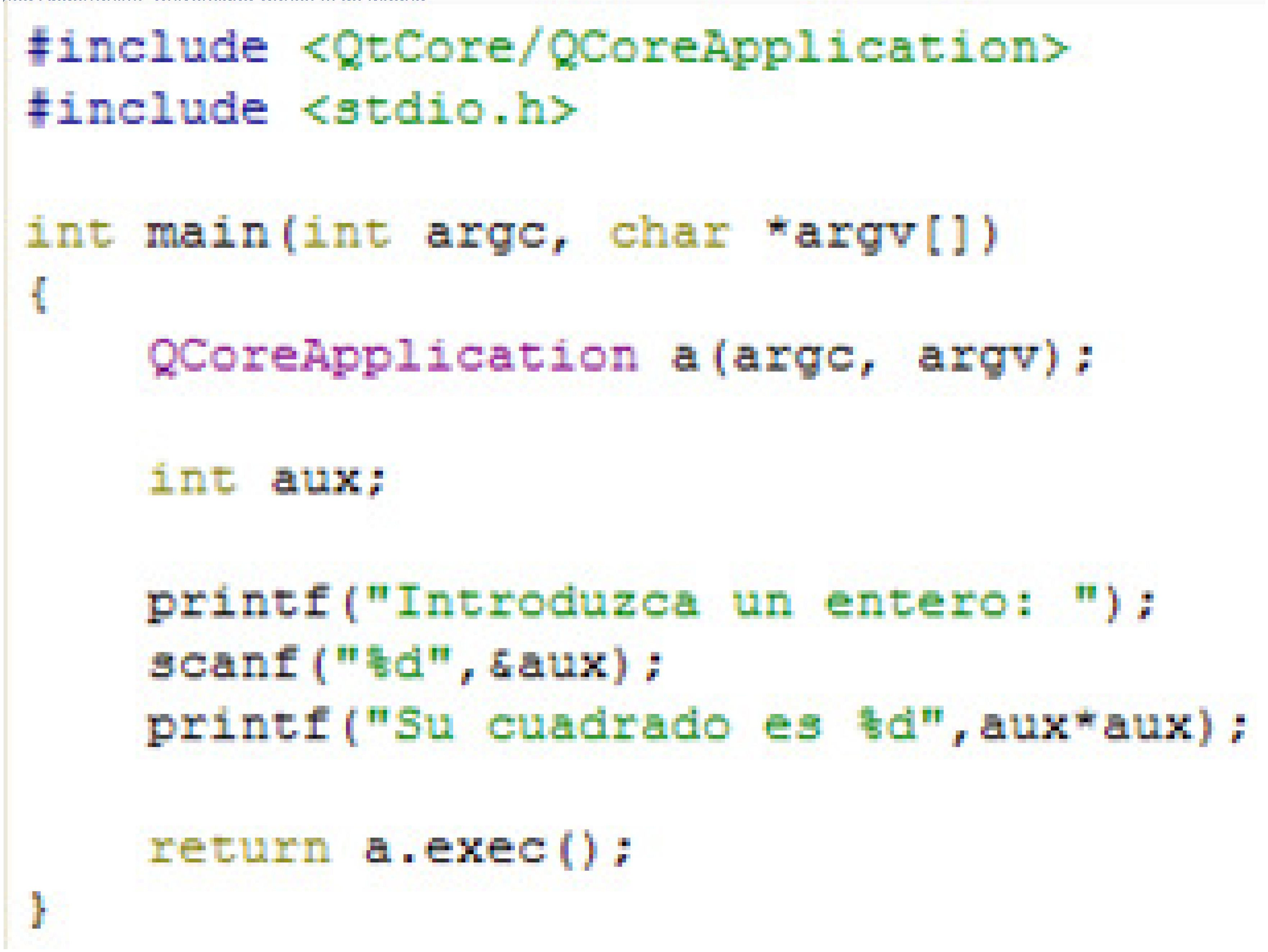
Project Management

Add to project: <None>

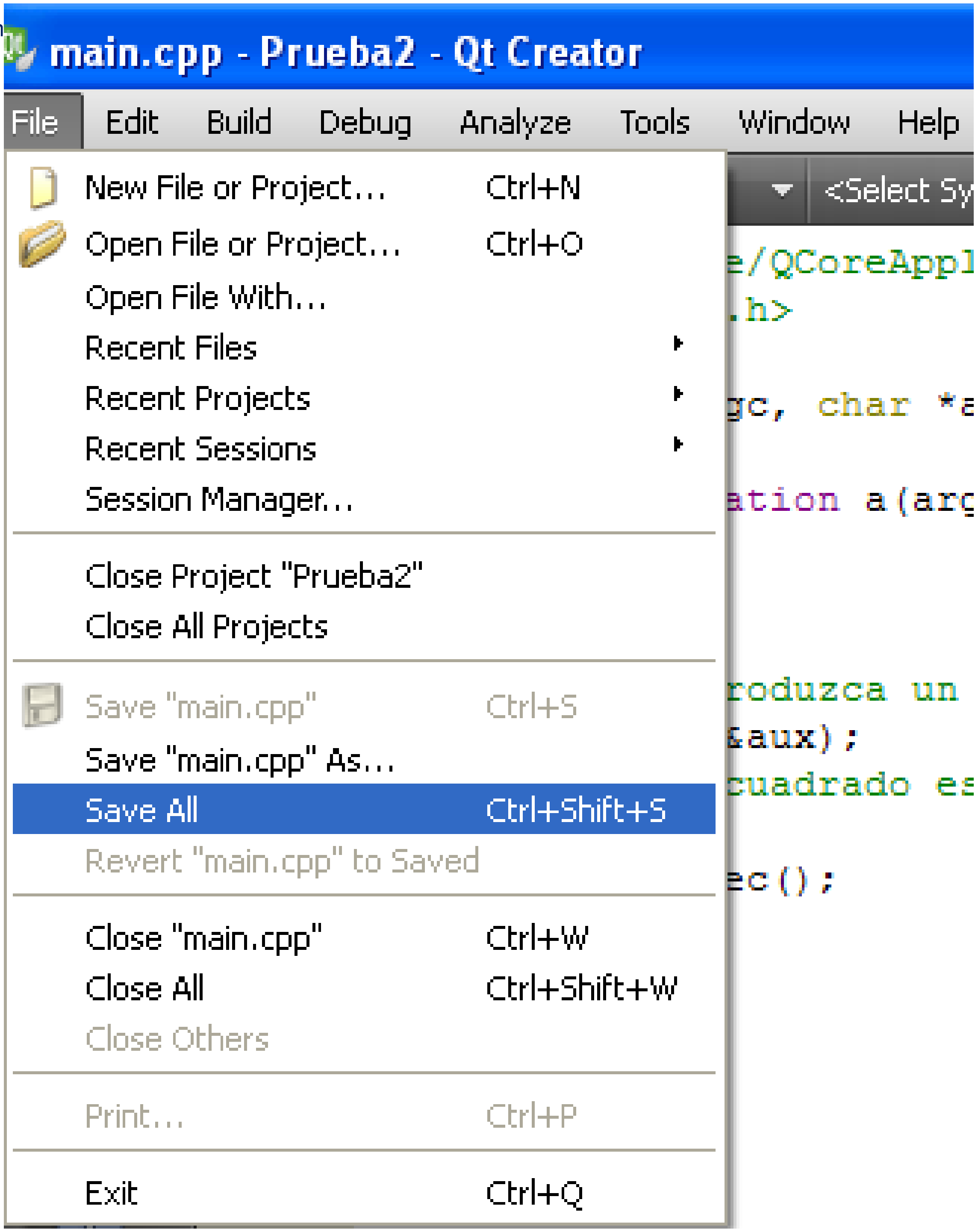
Add to version control: <None>

Files to be added in

```
C:\usuarios\domingo\Programas\Prueba3:

main.cpp
Prueba3.pro
```

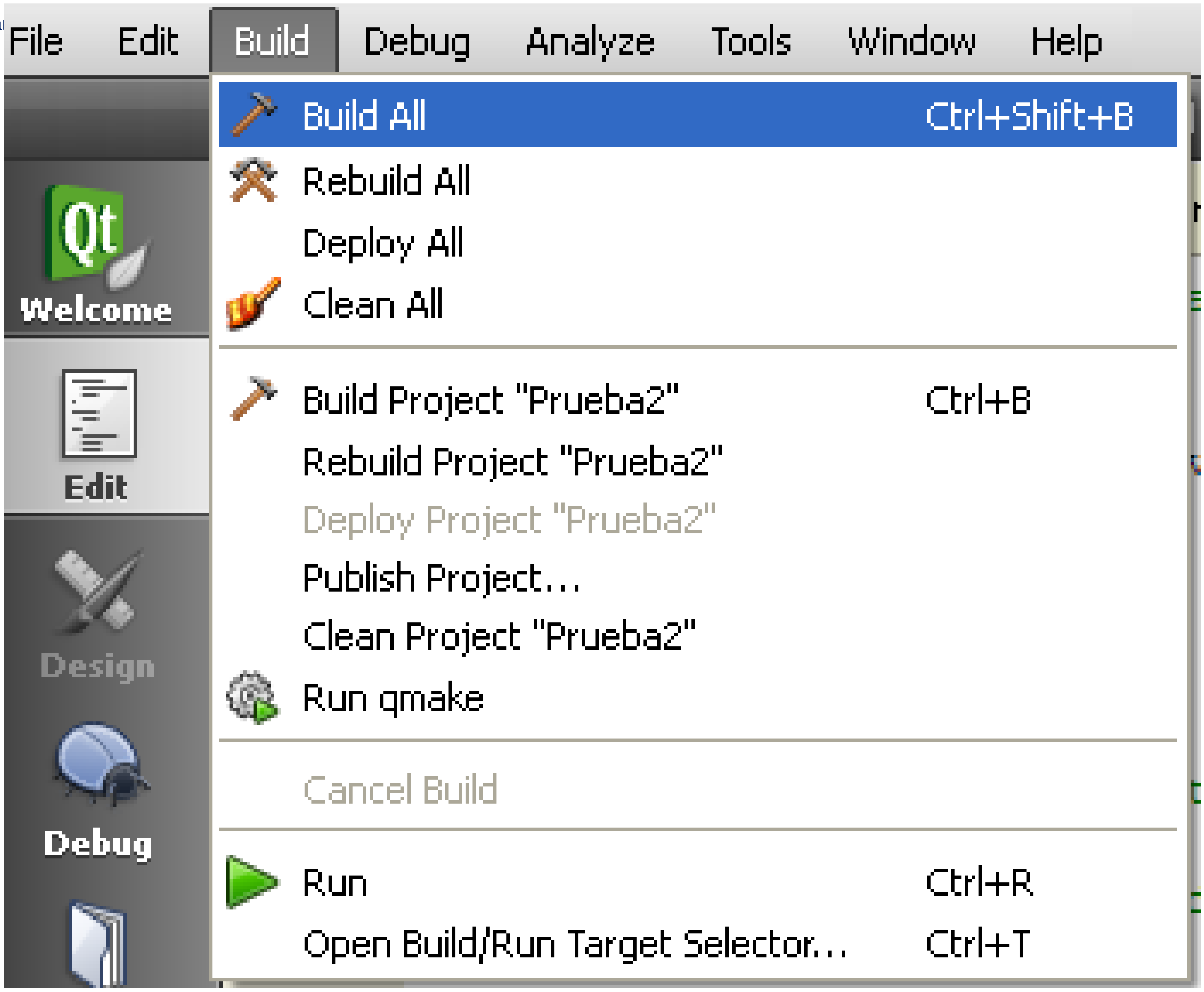


```
#include <QtCore/QCoreApplication>
#include <stdio.h>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc, argv);

    int aux;

    printf("Introduzca un entero: ");
    scanf("%d",&aux);
    printf("Su cuadrado es %d",aux*aux);

    return a.exec();
}
```



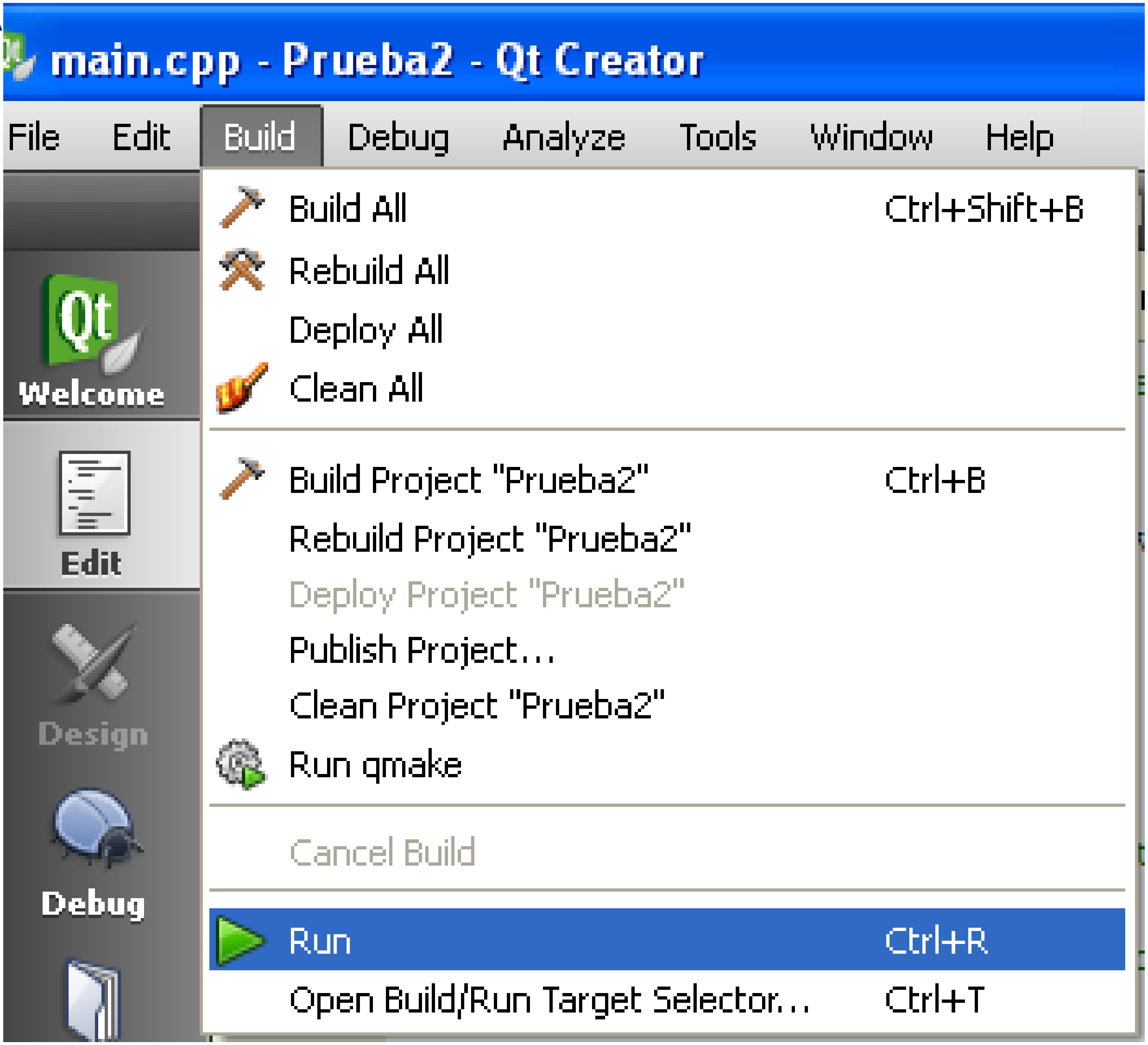
```
e/QCoreAppl
.h>

gc, char *a

ation a(arg



roduzca un
&aux);
cuadrado es

ec();
```

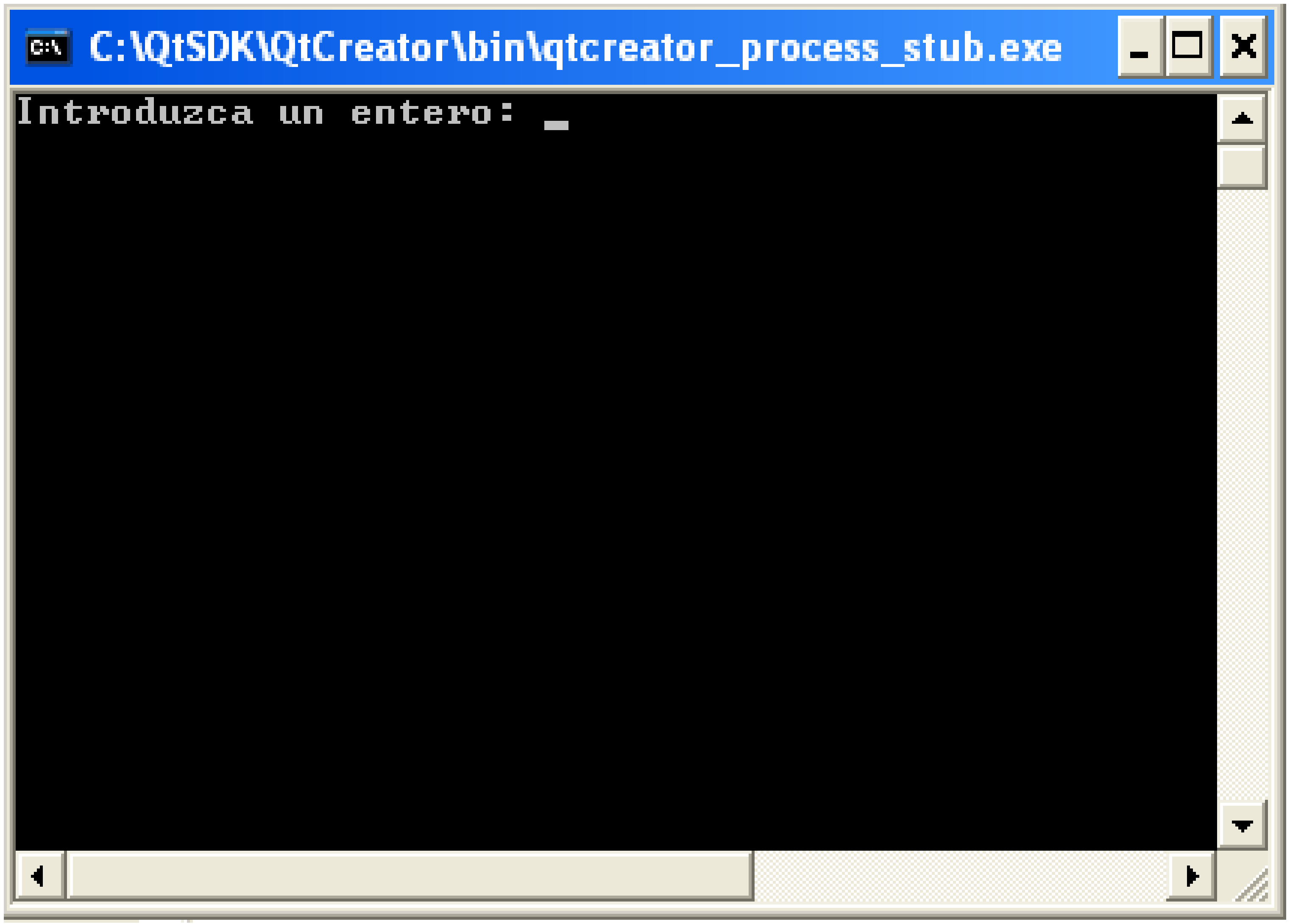
Moha



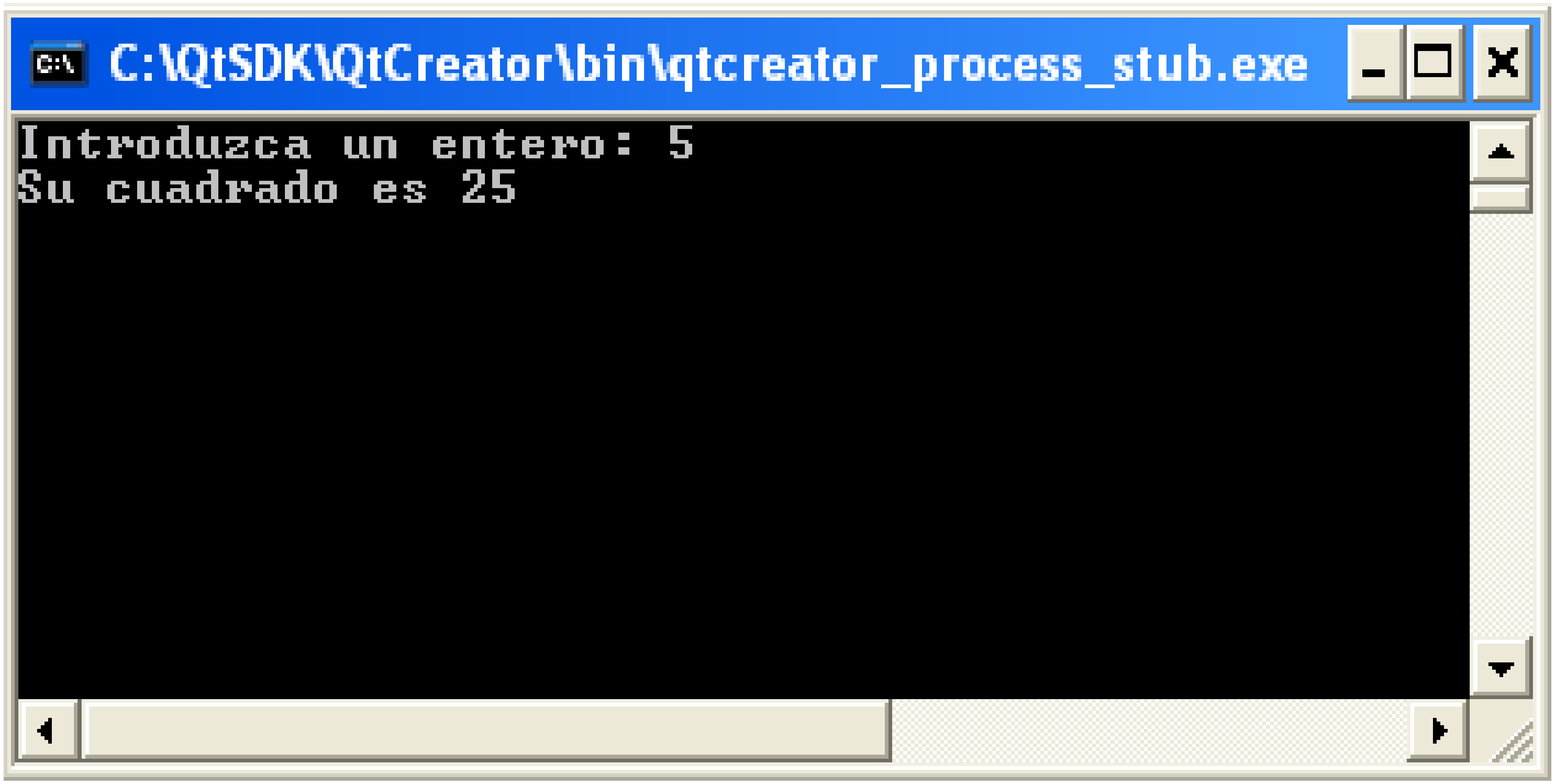
Build All Ctrl+Shift+B
Rebuild All
Deploy All
Clean All
Build Project "Prueba2" Ctrl+B
Rebuild Project "Prueba2"
Deploy Project "Prueba2"
Publish Project...
Clean Project "Prueba2"
Run qmake
Cancel Build
Run Ctrl+R
Open Build/Run Target Selector... Ctrl+T

Mohamed A


main.cpp - Prueba2 - Qt Creator
File
Edit
Build
Debug
Analyze
Tools
Window
Help
Build All Ctrl+Shift+B
Rebuild All
Deploy All
Clean All
Build Project "Prueba2" Ctrl+B
Rebuild Project "Prueba2"
Deploy Project "Prueba2"
Publish Project...
Clean Project "Prueba2"
Run qmake
Cancel Build
Run Ctrl+R
Open Build/Run Target Selector... Ctrl+T
Welcome
Edit
Design
Debug



```
Introduzca un entero: _
```



```
Introduzca un entero: 5
Su cuadrado es 25
```

Typical Problems

- If you get **printf** has not been declared, you probably forgot to include the library **stdio**.
- If you get **cout is not a member of std**, you probably forgot to include the library **iostream**
- If you get missing **;** before a function, you probably forgot to put it at the end of the previous.
- If you get missing **{** or **}** on given place, you may have forgotten putting it far before.