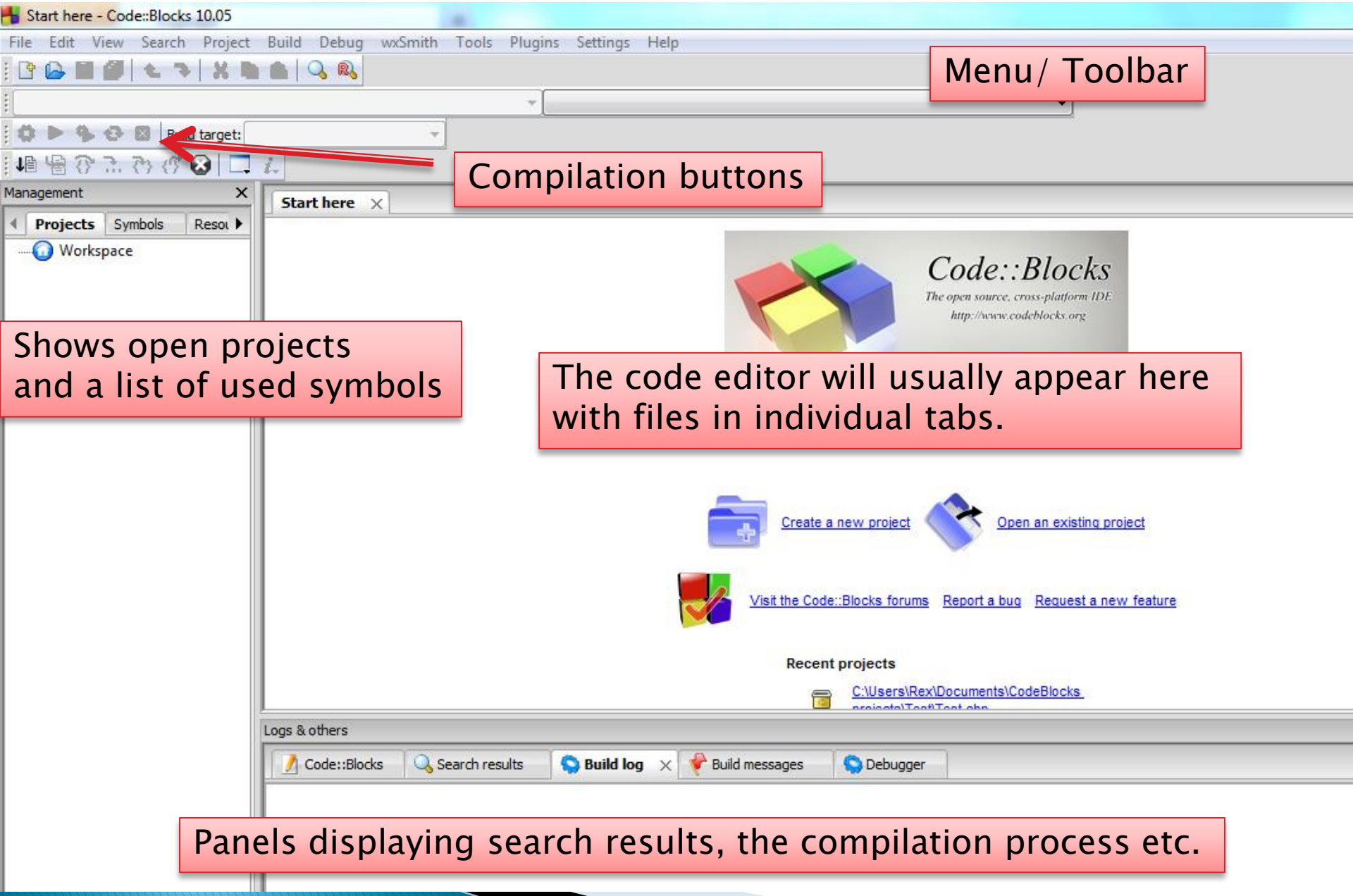
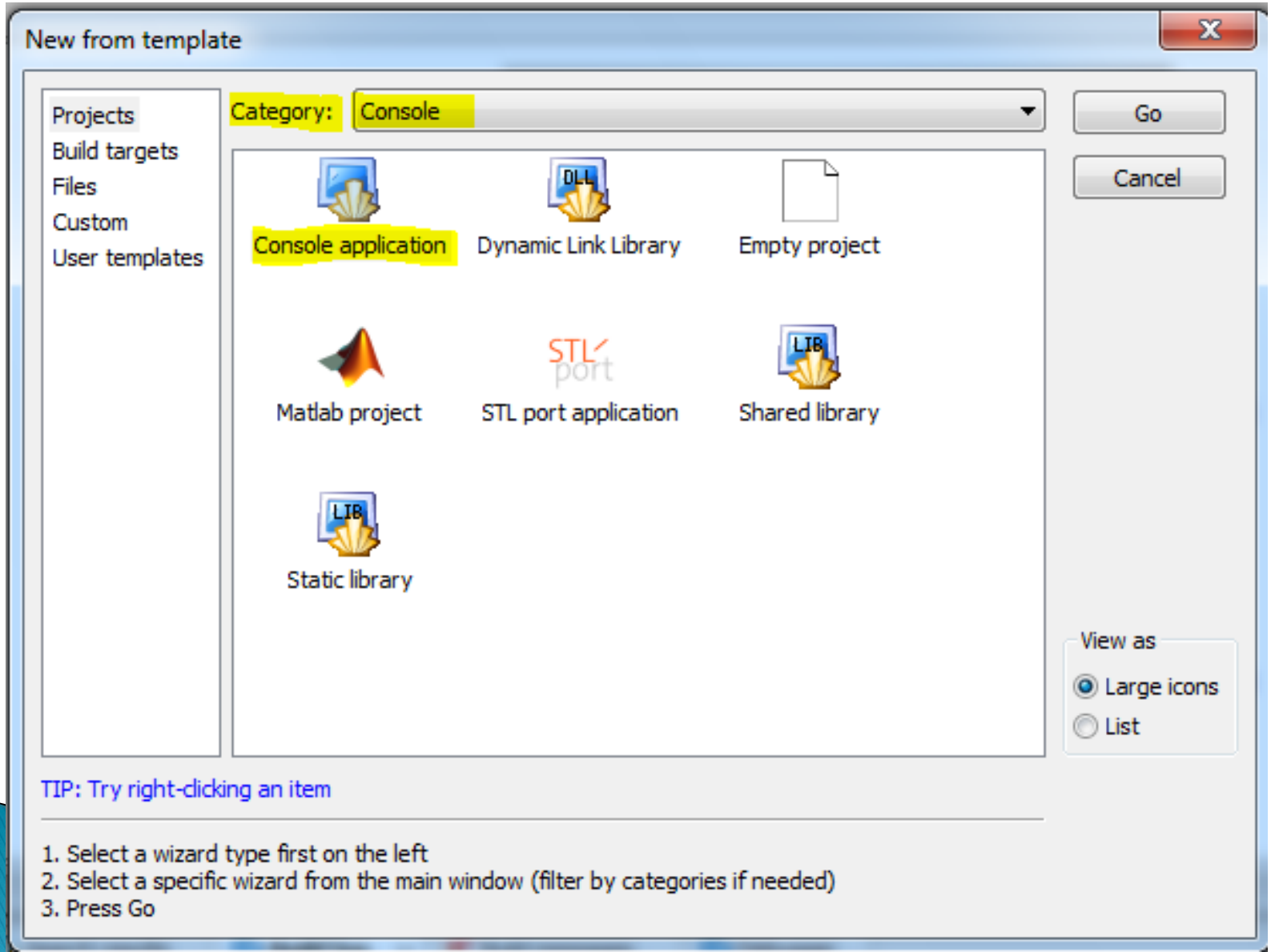
Programming languages WSTI

Types, operators, expressions

The appearance of the environment



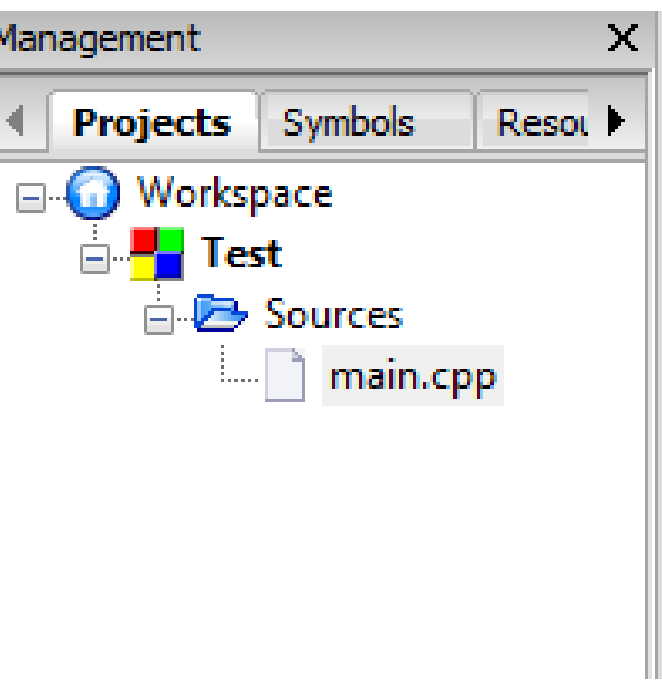
Creating a new project



The first application

Build & Run / F9

Header files



Declaration of namespace

```
main.cpp
1   #include <iostream>
2
3   using namespace std;
4
5   int main()
6   {
7       cout << "Hello world!" << endl;
8       return 0;
9   }
10
```

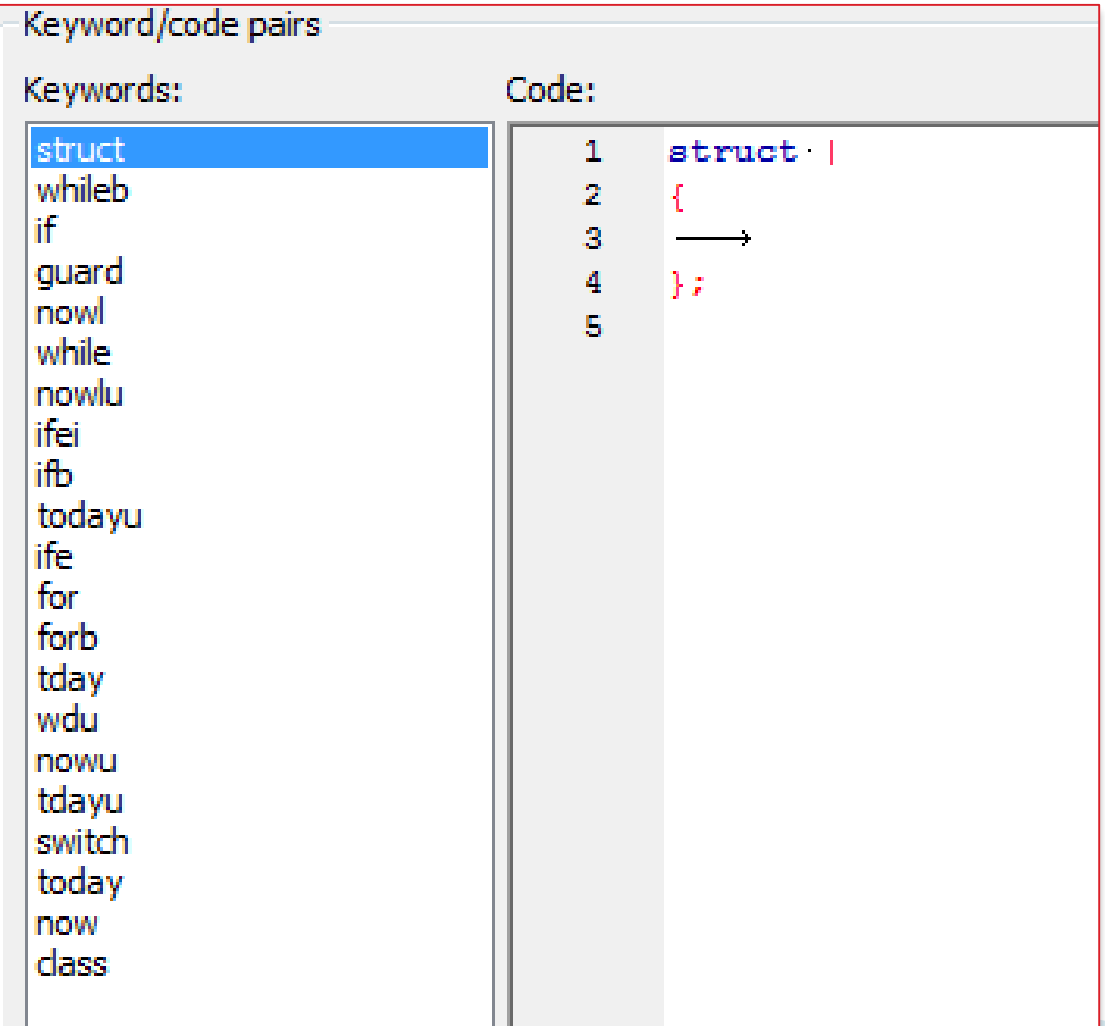
Exit code

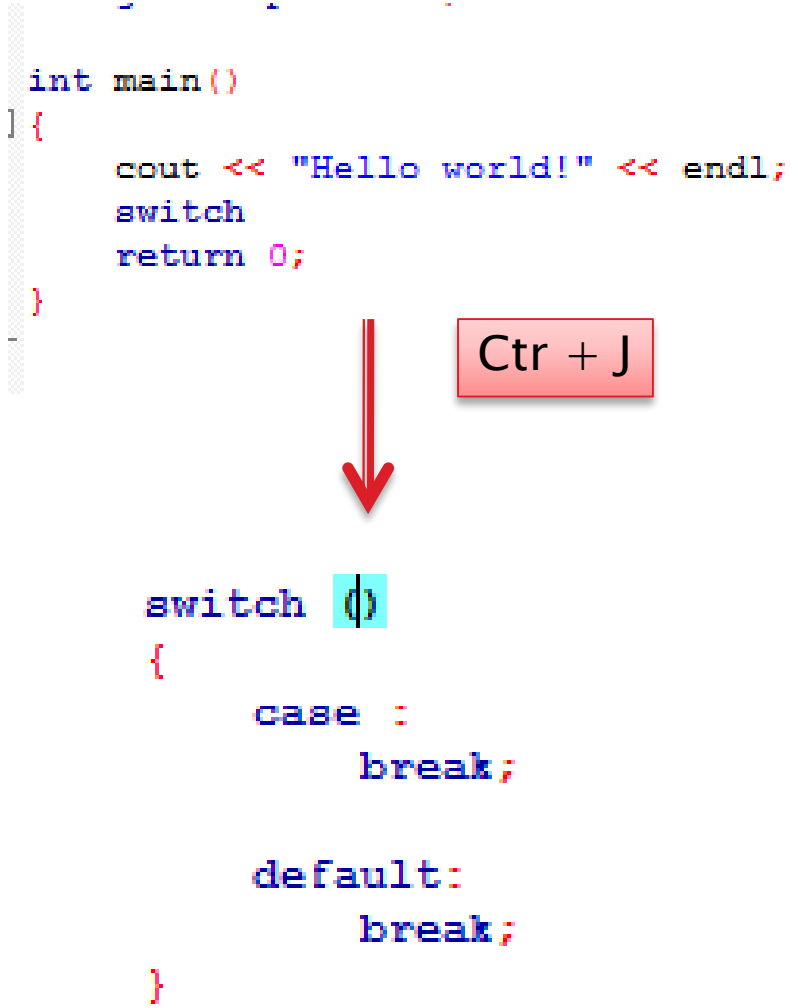
Output of the output stream content

Sources – displays files with the extensions *.c;*.cpp;.
ASM Sources – containes *.s;*.S;*.ss;*.asm files.
Headers – mainly header files – *.h;.
Resources – containes *.res;*.xrc files (for wxWidgets)

Get to know your environment, part 1

- Abbreviation mechanism

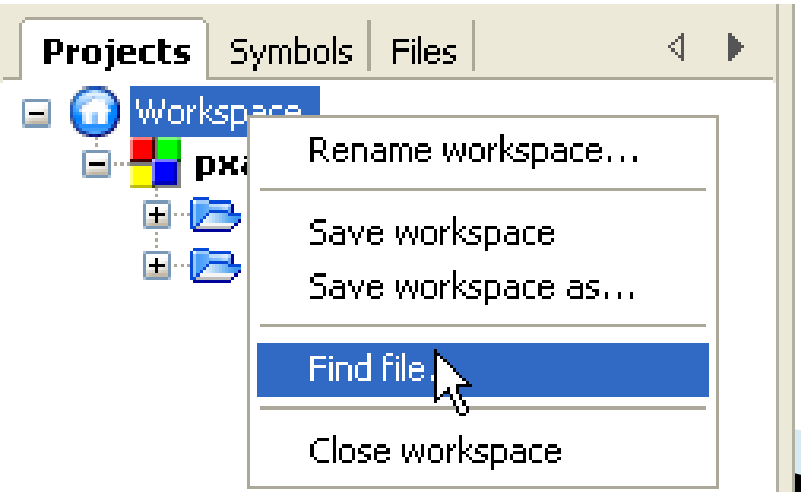


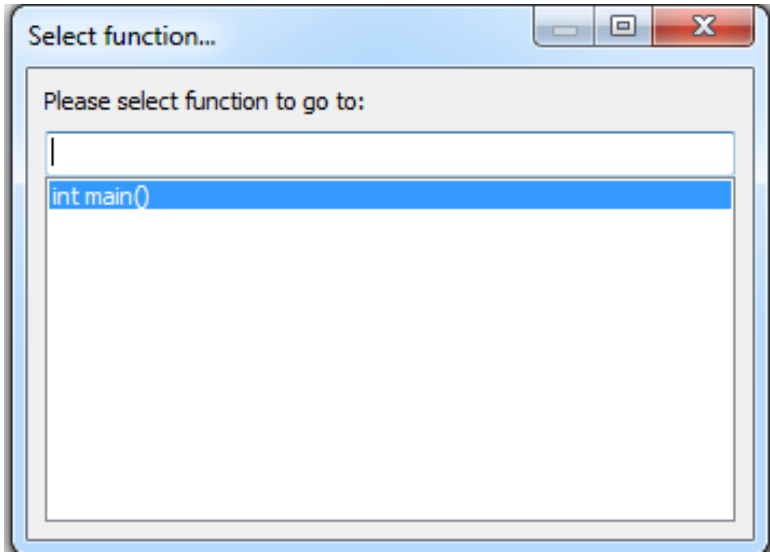


Get to know your environment, part 2

- Search and navigation in the code

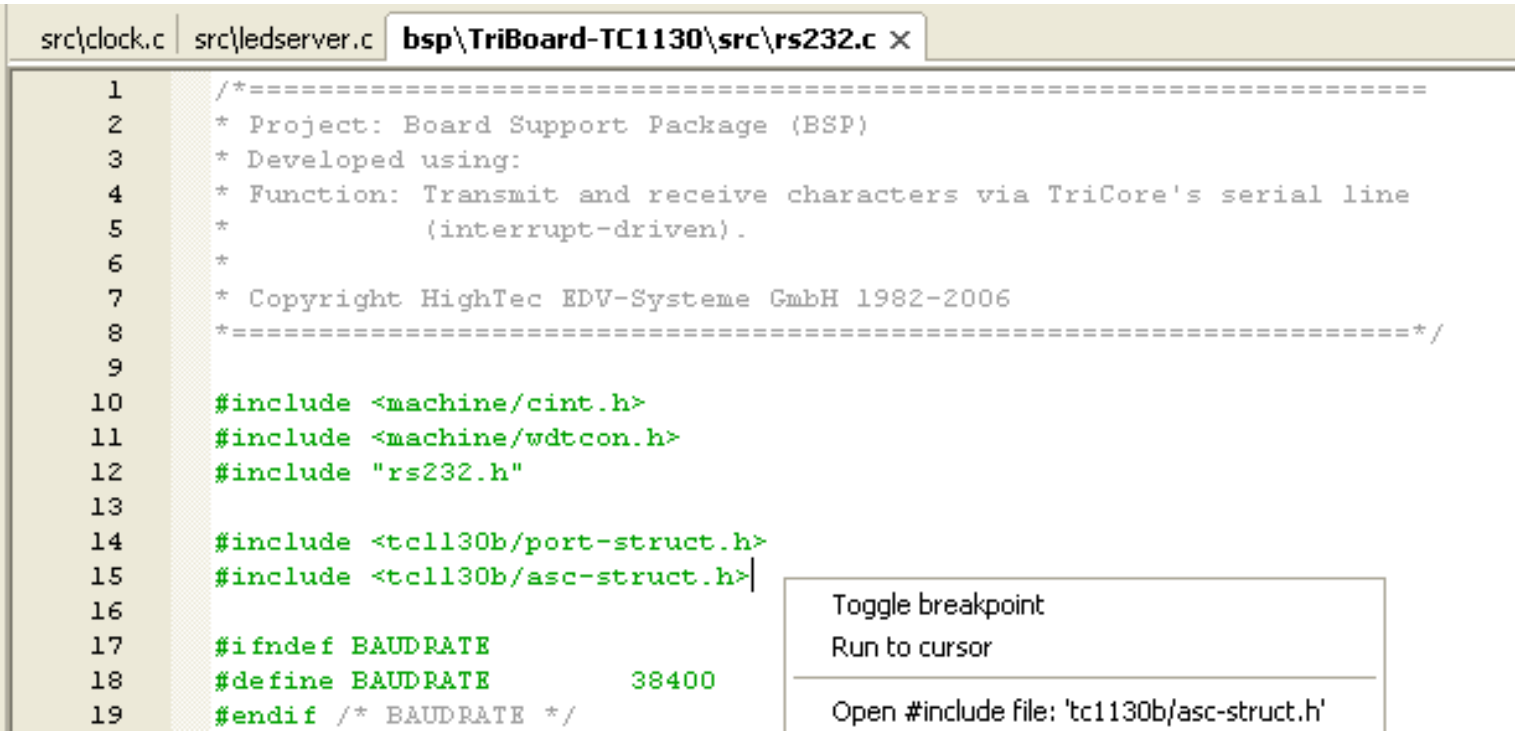
Ctr + **B** – create a bookmark
Alt + **PgUp**\ **Alt** + **PgDown** – tab navigation
Ctr + **F** – find in code
Ctr + **Shift** + **F** – find in files
Ctr + **Alt** + **G** – find a function
Ctr + **PgUp** \ **Ctr** + **PgDown** – function navigation
Ctr + **Alt** – find a file to open
Ctr + **I** – incremental search.





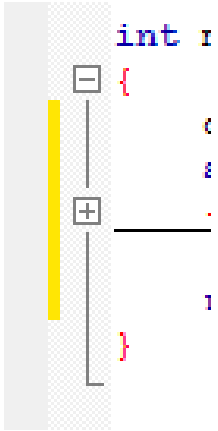
Get to know your environment, part 3

- Navigating header files
  - Place the cursor on the #**include** directive and select **open include file** from the pop-up menu.
  - Choose **Swap header/source**.



Get to know your environment, part 4

- **Code folding**
  //{
  Very long code
  //}
- **Code auto complete**
  - **Ctr + space** – forcing auto-complete.
- **Code export**
  - File / Export Menu
    - html
    - rtf
    - odt
    - pdf
- Line nubers
  - Settings -> General Settings-> **Show line numbers**



Keyboard shortcuts

Function	Shortcut Key
Undo last action	Ctrl-Z
Redo last action	Ctrl-Shift-Z
Swap header / source	F11
Comment highlighted code	Ctrl-Shift-C
Uncomment highlighted code	Ctrl-Shift-X
Auto-complete / Abbreviations	Ctrl-Space/Ctrl-J
Toggle bookmark	Ctrl-B
Goto previous bookmark	Alt-PgUp
Goto next bookmark	Alt-PgDown

Function	Shortcut Key
New file or project	Ctrl-N
Open existing file or project	Ctrl-O
Save current file	Ctrl-S
Save all files	Ctrl-Shift-S
Close current file	Ctrl-F4/Ctrl-W
Close all files	Ctrl-Shift-F4/Ctrl-Shift-W

Function	Shortcut Key
Find	Ctrl-F
Find next	F3
Find previous	Shift-F3
Find in files	Crtl-Shift-F
Replace	Ctrl-R
Replace in files	Ctrl-Shift-R
Goto line	Ctrl-G
Goto next changed line	Ctrl-F3
Goto previous changed line	Ctrl-Shift-F3
Goto file	Alt-G
Goto function	Ctrl-Alt-G
Goto previous function	Ctrl-PgUp
Goto next function	Ctrl-PgDn
Goto declaration	Ctrl-Shift-.
Goto implementation	Ctrl-.
Open include file	Ctrl-Alt-.

Create or delete a bookmark	Ctrl-F2
Go to next bookmark	F2
Select to next bookmark	Alt-F2
Find selection.	Ctrl-F3
Find selection backwards.	Ctrl-Shift-F3
Find matching preprocessor conditional, skipping nested ones.	Ctrl-K

Function	Shortcut Key
Build	Ctrl-F9
Compile current file	Ctrl-Shift-F9
Run	Ctrl-F10
Build and Run	F9
Rebuild	Ctrl-F11

Streams

```
std::cout << "I know " << 219 << " ways to make an
  omelette." << std::endl;
```

std::endl – represents the end of the line (which moves to the next one). Flushes the output stream buffer.

'\n' – special character sequence – represents the transition to a new line

cout – object representing a program's standard output stream

<< – inserter. Data is output in accordance with their type.

Types of variables

Type	Description	Usage
int	A signed integer	`int i = - 5;`
short (int)	Short integer (minimum 2 bytes)	`short a = -10;`
long (int)	Long integer (minimum 4 bytes)	`long z = 20;`
long long (int)	Very long integer (minimum 8 bytes)	`long long b = 30;`
unsigned … (int)	Numbers >= 0	`unsigned int i = 4;`
float	Single precision floating point numbers	`float f = 7.5f;`
double	Double precision floating point numbers	`double d = 7.5;`
long double	Large floating point numbers.	`long double d = 2.5L`
char	A single character	`char ch = 'a';`
char16_t	Single character (16-bit)	`char ch16 = u'a';`
char32_t	Single character (32-bit)	`char ch32 = U'a';`
wchar_t	A single character (size depends on the compiler)	`wchar_t = L'a'`
bool	true / false (zero or no 0)	`bool b = false;`
auto	The compiler will automatically select the type	`auto i = 20;`
decltype(wyrażenie)	The type that was given as a parameter	`decltype(int) j = 8;`

Special sequences

- They represent characters generally without graphic equivalents

\a	0x07	BEL	Audible bell
\b	0x08	BS	Backspace
\f	0x0C	FF	Formfeed
\n	0x0A	LF	Newline (linefeed)
\r	0x0D	CR	Carriage return
\t	0x09	HT	Tab (horizontal)
\v	0x0B	VT	Vertical tab
\\	0x5c	\	Backslash
\'	0x27	'	Apostrof
\"	0x22	"	Cudzysłów
\?	0x3F	?	Pytajnik
\O		any	O = łańcuch ósemkowych cyfr
\xH		any	H = łańcuch szesnastkowych cyfr
\XH		any	H = łańcuch szesnastkowych cyfr

Types of variables in practice

```
                        char: -128..127
                   short int: -32768..32767
                         int: -2147483648..2147483647
                    long int: -2147483648..2147483647
(C99)          long  long int: -9223372036854775808..9223372036854775807
                 unsigned char: 0..255
           unsigned short int: 0..65535
                  unsigned int: 0..4294967295
             unsigned long int: 0..4294967295
(C99) unsigned long long int: 0..18446744073709551615
```

- Additionally:

sizeof(char) <= sizeof(short int) <= sizeof(int) <= sizeof(long int)

- And:

unsigned char = odpowiednik bajta (byte)

unsigned short int = odpowiednik słowa (word)

Most compilers will display a **warning** when using uninitialized variables, and some development environments will display a **run-time error** when we want to retrieve an uninitialized value.

Operator	Description	Usage
=	A binary assigment operator	int i = 1;
!	Unary operator of logical negation	bool b = !true;

+
-
*
/
	Binary operators: addition
	subtraction
	multiplication
	division (result dependent on arguments)
		int i = 3 + 2;
		int i = 3 - 2;
		int i = 3 * 2;
		int i = 3 / 2;

%	Binary modulo operator	int reszta = 3 % 2;
++ --	Unarny incrementation operator Unarny decrementation operator	int i = 1; i++; or ++i;
~	Unary complement (bit inversion)	Int z = ~1
&	Bitwise AND	int z = 2 & 3;
|	Bitwise inclusive OR	int z = 2 | 3;
<< >>	Shift bits left Shift bits right	int z = 4 << 1;
^	Bitwise exclusive OR (XOR)	int z = 2 ^ 3;
&&	Logical AND	if (a > 5 && a < 10)
||	Logical OR	if(a == 2 || a == 3)

==
!=
<, <=
>, >=
	Relational and comparison operators: equal to
	not equal to
	Less then (or equal to)
	Greater then (or equal to)
		if (a == 3)
		if (a != 3)
		if (a < 3)
		if (a >= 3)

Identifiers

- An identifier is a sequence of letters, numbers and underscore characters beginning with a letter, with the underscore being treated as a letter.
- Uppercase and lowercase letters are distinguished.
- Note – Polish characters are not treated as letters! Keywords are also protected.
- Which are correct?

```
Control
K21
Pi
PI
_Control
J-92
_2Control
Wartość
2controls
This test
```

Exercise 1

1. Declare several variables: float, double, int. Assign them a staring value. Print them to the screen (using cout).
2. What is the largest integer that can be entered into a C++ program?
3. Declare a constant and try to change it. What will happen?

We declare constants with the use of const e.g.

```
const int i = 1;
```

Input and output

- **cin** — a stream representing the standard program input.
- **cout** — a stream representing the standard output of the program.
- **cerr** — **unbuffered** error output stream.
  clog — **buffered** error output stream.

```
#include <iostream>
#include <cstdlib>

using namespace std;

int main()
{
    int liczba;

    cout << "Podaj liczbe: ";
    cin >> liczba;
    cout << '\n' << flush;

    if (cin.good())
        cout << "Dobry strumien" << endl;
    else
        cout << "Bledy w strumieniu" << endl;

    cout << "Podales wartosc: " << liczba << endl;

    cin.ignore();
    cin.get();

    return EXIT_SUCCESS;
}
```

Exercise 2

1. Declare an int variable representing the user's age. Input your age (using the keyboard) and display it.
2. Try to type something incorrect (like „Alice has a cat"). What's happening?

Exercise 2

1. Declare an int variable representing the user's age. Input your age (using the keyboard) and display it.
2. Try to type something incorrect (like „Alice has a cat"). What's happening?

Validate user input!

Errors?

```
std::cin.ignore(std::numeric_limits<std::streamsize>::max(),'\n');
```

OR

```
std::cin.sync();
```

Streams and errors

```
#include <iostream>
#include <cstdlib>
#include <limits>

using namespace std;

int main()
{
    int liczba;

    cout << "Podaj liczbe: ";
    cin >> liczba;
    cout << '\n' << flush;
    cout << "Podales wartosc: " << liczba << endl;

    cout << "Jeszcze jedna prosze..." << endl;
    cin.clear();
    cin.ignore( numeric_limits< streamsize >::max( ), '\n' );
    cin >> liczba;
    cout << "Podana druga wartosc to: " << liczba << endl;

    return EXIT_SUCCESS;
}
```

Exercise 3

1. What will be the result of: *float a = 7 / 2; cout << a;*
2. How can I make the previous expression give the correct result?
3. What will be the result of the following pieces of code?

```
#include <iostream>
#include <cstdlib>

using namespace std;

int main()
{
    int a = 1;
    a = a++;
    a = ++a;
    a = a++ + ++a;
    cout << ++a << " " << ++a << endl;
    cout << a++ << " " << a++ << endl;

    return EXIT_SUCCESS;
}
```

```
#include <iostream>
#include <cstdlib>

using namespace std;

int main()
{
    int a = 2147483647; // MAX_INT = 2147483647
    cout << a++ << endl;
    cout << a << endl;

    return EXIT_SUCCESS;
}
```

http://c-faq.com/expr/evalorder2.html

Exercise 4

- What will be the result of:

```
#include <iostream>
#include <cstdlib>

using namespace std;

int main()
{
    int a = 1, b = 2;
    if (++a = b)
        cout << "Rowne" << endl;
    else
        cout << "Rozne" << endl;
    cout << a << " " << b << endl;

    return EXIT_SUCCESS;
}
```

Zero means **false**, **true** otherwise.

Exercise 5

1. Declare three variables named A, B, C, which will be the coefficients of the quadratic equation written in the form::

$$Ax^2 + Bx + C = 0$$

   Write a program that solves this quadratic equation for the coefficients given by the keyboard. Take care of the correctness (validity) of the input data.

Square root: *#include <cmath>*

sqrt()

Language reference: loops

- While loop:

```
int j = 2;
while (j < 9) {
    cout << j << " ";
    j += 2;
}
```

- For loop:

```
for (int i = 0; i < 10; i++) {
    cout << i << " ";
}
```

- Do.. while loop:

```
int i = 11;
do {
    i = i + 10;
    cout << i << " ";
} while (i < 10);
```

- Range-based loop:

```
vector<int> v = {0, 1, 2, 3, 4, 5};
 for (auto i : v)
    std::cout << i << ' ';
```

Break and continue works as expected

Exercise 6

1. What will be the result of the program:

```
int main()
{
    int i;
    for (i = 1; i <= 5; ++i ) cout << i << endl;
    for(; i >= 1; i--) cout << i << endl;

    return EXIT_SUCCESS;
}
```

2. What will the following loops do:

```
int a = 2;
for (;;) { }
for (;1;) {}
for (a;a;a) {}
while(1) { }
do { } while(1);
```

How to stop them?

Exercise 7

1. Write a program that reads natural numbers until you specify 0. Then it prints the minimum, maximum of the given numbers and their average.
2. Determine all prime numbers from 0 to 100.
3. Using loops, write a program that displays even integers from the range 31 to 52.
4. Write a program that displays on screen numbers between 1 and 100 divisible by 4, but indivisible by 8 and indivisible by 10. Use the continue statement for this purpose.
5. Write a program, defining how many years you have to save in a bank on a 5% deposit, with earnings of 12,000 $ per year to have at least 200000 zł. Assume that no tax is charged on interest or income.

Exercise 8

1. Write a program calculating the largest common divisor of two integers given by the user using the Euclid's algorithm.
2. Write a program which calculates a factorial of a given number.
3. Write a calculator program that will perform the following operations:
   - Addition of two numbers
   - Substraction of two numbers
   - Division of two numbers
   - Multiplication of two numbers
   - Determination of the square root of a number
   - Calculating the percentage from a number.
   - Determining the remainder of the division of two numbers.
   - Determination of any power of a given number.

   The calculator should allow selection of operations as long as the user wants.

You should use the instructions **cin.good()** or **cin.fail()** in your programs.