

Code::Blocks

1. 도구 개요

2. 설치 및 실행

3. 주요 기능

4. 활용 예제

1. 도구 개요

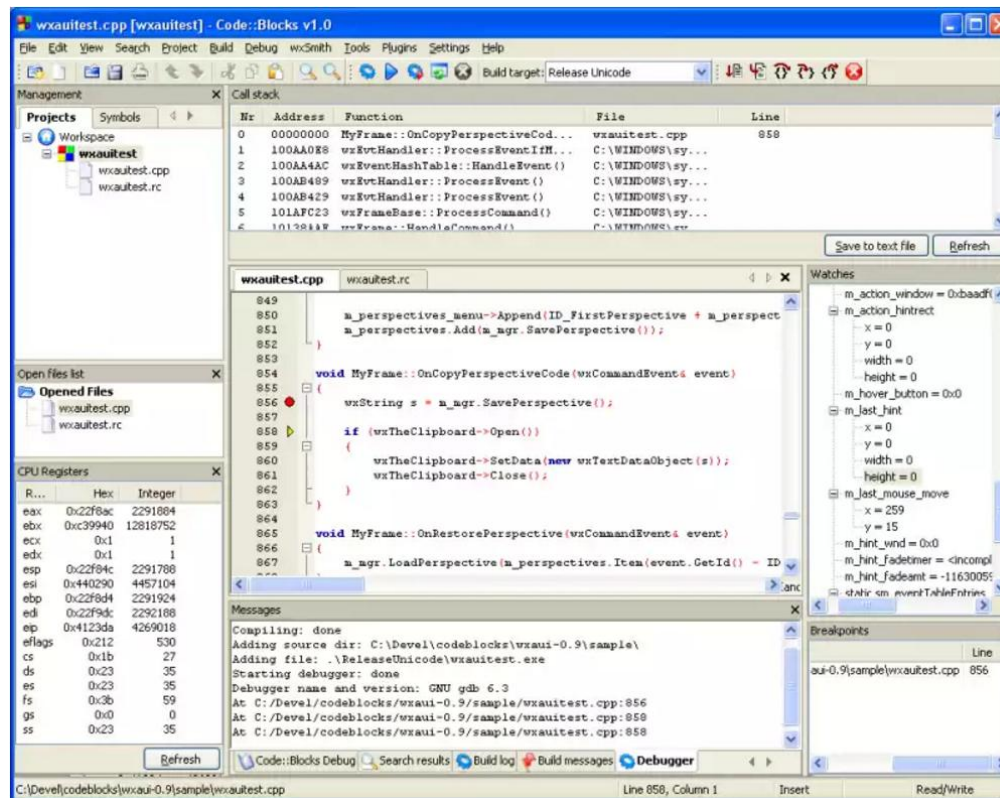
1.1 도구 정보 요약

도구명	Code::Blocks (http://www.codeblocks.org/)	라이선스	GNU Genreral Public License v3.0
소개	• C++로 작성된 C, C++, Fortran 개발 지원 공개SW IDE		
특징	• Console, DirectX, OpenGL, OpenCV, GTK, Matlab 등 다양한 프로젝트 템플릿 제공 • 리눅스, 윈도우, 맥OS 등 주요 운영체제 모두 지원		
주요기능	• 코드 편집, 문법 강조 기능, 코드 자동 완성 등 • 컴파일, 디버거, 플러그인 • 다른 IDE의 프로젝트 가져오기		
실행환경	• Linux, Windows, MacOS	사전설치도구	• 해당 없음
카테고리	• 구현	최신버전	• v13.12 (2015.11)
관련도구	• Eclipse, SharpDevelop		

1. 도구 개요

1.2 스크린 캡처 및 주요 기능

- C, C++, Fortran 개발을 지원하는 IDE
- GCC, Microsoft Visual C++, clang, Digital Mars, Borland C++ 등 다양한 컴파일러 지원
- GNU GDB 기반의 디버거 지원



2. 설치 및 실행

세부 목차

2.1 다운로드

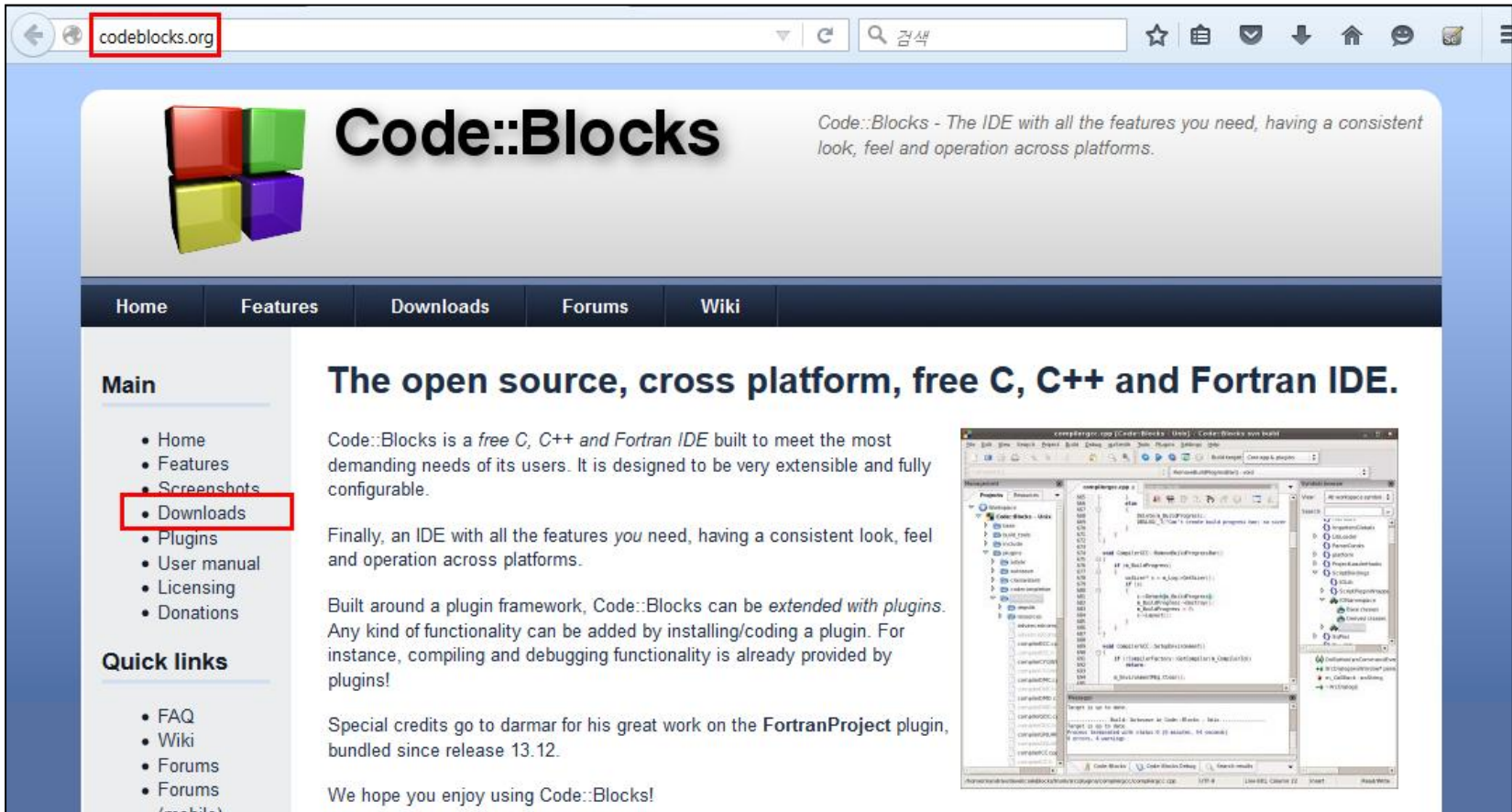
2.2 설치

2.3 설치 확인

2. 설치 및 실행

2.1 다운로드 (1/4)

- <http://codeblocks.org> 에 접속
- 왼쪽 Downloads 클릭



The screenshot shows the Code::Blocks website homepage. The browser's address bar shows "codeblocks.org". The website header features the Code::Blocks logo and the tagline "Code::Blocks - The IDE with all the features you need, having a consistent look, feel and operation across platforms." Below the header is a navigation bar with links: Home, Features, Downloads, Forums, and Wiki. The "Downloads" link is highlighted with a red box. On the left side, there is a "Main" section with a list of links: Home, Features, Screenshots, Downloads (highlighted with a red box), Plugins, User manual, Licensing, and Donations. Below this is a "Quick links" section with links to FAQ, Wiki, Forums, and Forums (mobile). The main content area has the heading "The open source, cross platform, free C, C++ and Fortran IDE." followed by a paragraph describing the IDE as a free, extensible, and fully configurable C, C++, and Fortran IDE. It mentions that the IDE is built around a plugin framework and can be extended with plugins. A screenshot of the Code::Blocks IDE interface is shown on the right. At the bottom, it says "Special credits go to darmar for his great work on the FortranProject plugin, bundled since release 13.12." and "We hope you enjoy using Code::Blocks!"

codeblocks.org

Code::Blocks

Code::Blocks - The IDE with all the features you need, having a consistent look, feel and operation across platforms.

Home Features Downloads Forums Wiki

Main

- Home
- Features
- Screenshots
- Downloads
- Plugins
- User manual
- Licensing
- Donations

Quick links

- FAQ
- Wiki
- Forums
- Forums (mobile)

The open source, cross platform, free C, C++ and Fortran IDE.

Code::Blocks is a *free C, C++ and Fortran IDE* built to meet the most demanding needs of its users. It is designed to be very extensible and fully configurable.

Finally, an IDE with all the features *you* need, having a consistent look, feel and operation across platforms.

Built around a plugin framework, Code::Blocks can be *extended with plugins*. Any kind of functionality can be added by installing/coding a plugin. For instance, compiling and debugging functionality is already provided by plugins!

Special credits go to darmar for his great work on the **FortranProject** plugin, bundled since release 13.12.

We hope you enjoy using Code::Blocks!

2. 설치 및 실행

2.1 다운로드 (2/4)

- Download the binary release 클릭



The screenshot shows the Code::Blocks website's 'Downloads' page. The browser address bar shows 'codeblocks.org/downloads'. The page has a navigation menu with 'Home', 'Features', 'Downloads', 'Forums', and 'Wiki'. The 'Downloads' section is active, displaying the title 'Downloads' and a list of download options. The first option, 'Download the binary release', is highlighted with a red box. Below it, a paragraph explains that this is the easiest way to install Code::Blocks. Other options include 'Download a nightly build' and 'Download the source code', each with a brief description. A 'Main' sidebar on the left contains links to Home, Features, Screenshots, Downloads (with sub-links for Binaries, Source, and SVN), Plugins, User manual, Licensing, and Donations. A 'Quick links' section at the bottom left lists FAQ, Wiki, and Forums.

codeblocks.org/downloads

Code::Blocks

Code::Blocks - The IDE with all the features you need, having a consistent look, feel and operation across platforms.

- Home
- Features
- Downloads
- Forums
- Wiki

Main

- Home
- Features
- Screenshots
- Downloads
 - Binaries
 - Source
 - SVN
- Plugins
- User manual
- Licensing
- Donations

Quick links

- FAQ
- Wiki
- Forums

Downloads

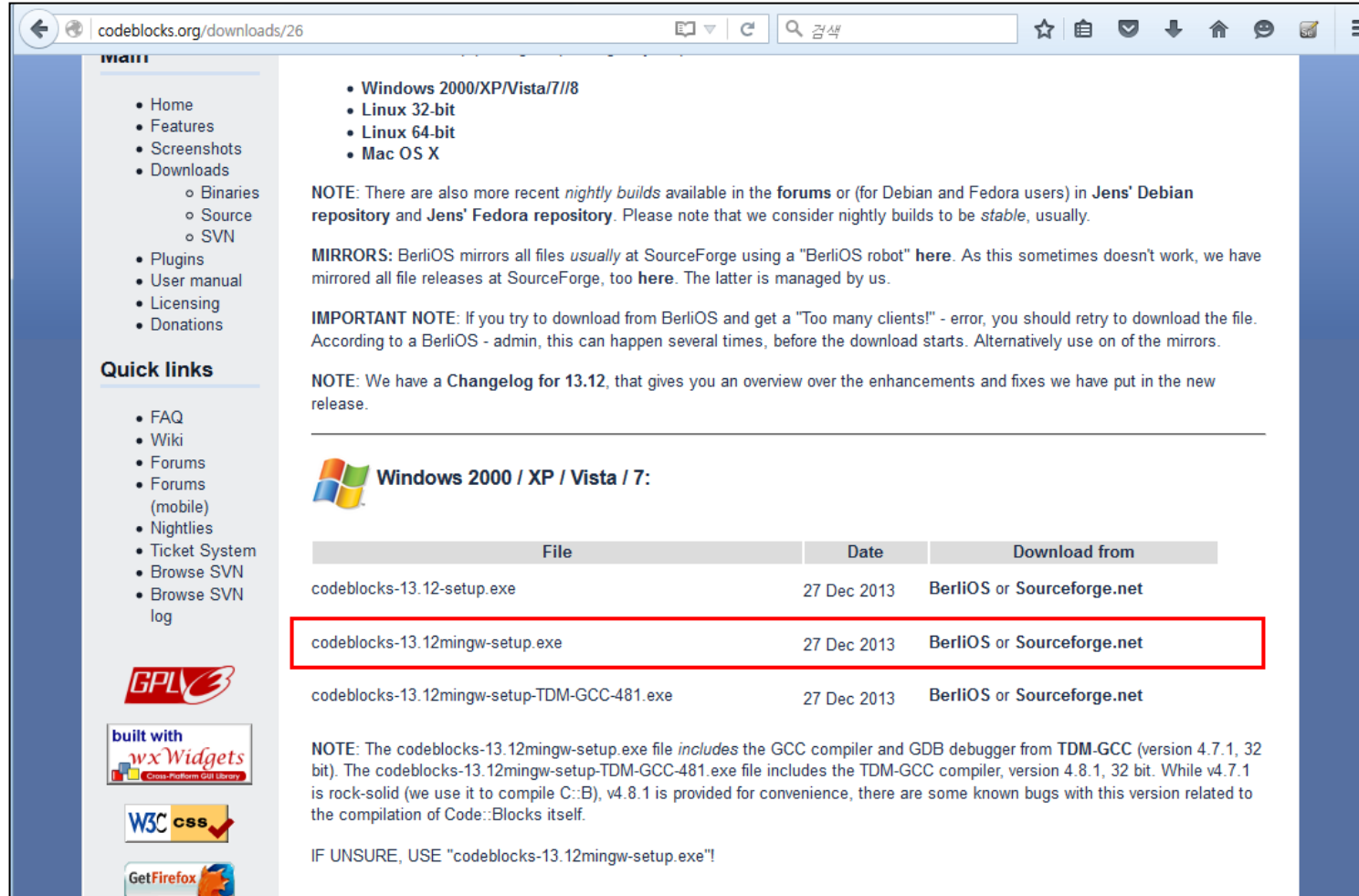
There are different ways to download and install Code::Blocks on your computer:

- **Download the binary release**
This is the easy way for installing Code::Blocks. Download the setup file, run it on your computer and Code::Blocks will be installed, ready for you to work with it. Can't get any easier than that!
- **Download a nightly build:** There are also more recent so-called *nightly builds* available in the **forums** or (for Debian and Fedora users) in **Jens' Debian repository** and **Jens' Fedora repository**. Other distributions usually follow provided by the community (Big "Thank you" for that!) Please note that we consider nightly builds to be *stable*, usually, unless stated otherwise.
- **Download the source code**
If you feel comfortable building applications from source, then this is the recommend way to download Code::Blocks. Downloading the source code and building it yourself puts you in great control and also makes it easier for you to update to newer versions or, even better, create patches for bugs you may find and contributing them back to the community so everyone benefits.

2. 설치 및 실행

2.1 다운로드 (3/4)

- GCC 컴파일러와 GDB 디버거가 포함되어 있는 mingw 포함 버전을 다운로드 한다. 우측의 Download from 에 있는 Sourceforge 링크 클릭



The screenshot shows the Code::Blocks download page. The left sidebar contains navigation links: Home, Features, Screenshots, Downloads (with sub-links for Binaries, Source, and SVN), Plugins, User manual, Licensing, and Donations. Below this is a 'Quick links' section with links to FAQ, Wiki, Forums, Forums (mobile), Nightlies, Ticket System, Browse SVN, and Browse SVN log. At the bottom of the sidebar are logos for GPL 3, wxWidgets, W3C CSS, and Get Firefox.

The main content area lists operating systems: Windows 2000/XP/Vista/7/8, Linux 32-bit, Linux 64-bit, and Mac OS X. It includes several notes: a note about nightly builds available in forums, a note about mirrors at SourceForge, an important note about download errors, and a note about the changelog for version 13.12.

Under the 'Windows 2000 / XP / Vista / 7:' section, there is a table with the following data:

File	Date	Download from
codeblocks-13.12-setup.exe	27 Dec 2013	BerliOS or Sourceforge.net
codeblocks-13.12mingw-setup.exe	27 Dec 2013	BerliOS or Sourceforge.net
codeblocks-13.12mingw-setup-TDM-GCC-481.exe	27 Dec 2013	BerliOS or Sourceforge.net

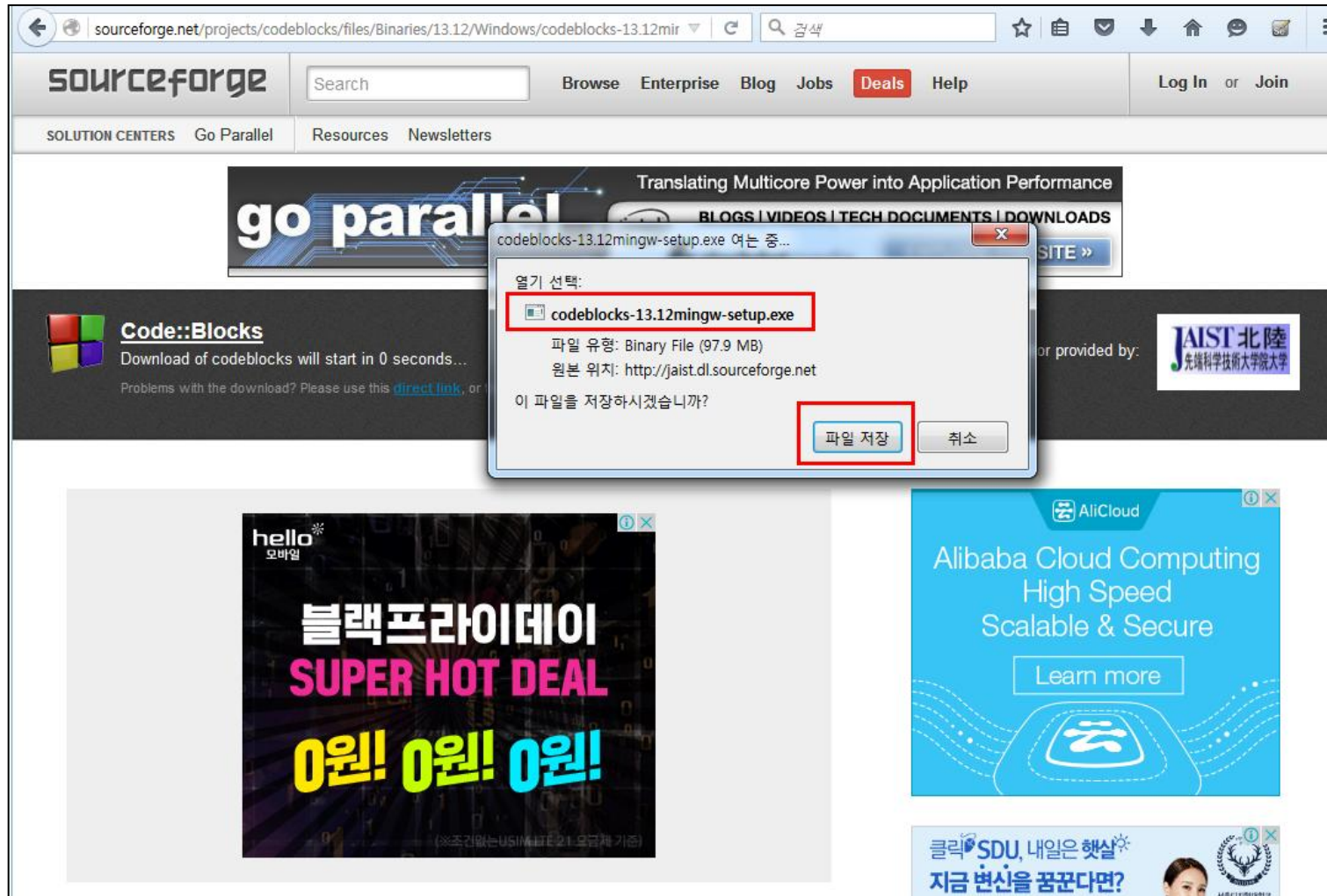
Below the table, there is a note stating that the codeblocks-13.12mingw-setup.exe file includes the GCC compiler and GDB debugger from TDM-GCC (version 4.7.1, 32 bit). It also mentions that the codeblocks-13.12mingw-setup-TDM-GCC-481.exe file includes the TDM-GCC compiler, version 4.8.1, 32 bit. Finally, it states that while v4.7.1 is rock-solid, v4.8.1 is provided for convenience, but there are some known bugs with this version related to the compilation of Code::Blocks itself.

At the bottom, it says: IF UNSURE, USE "codeblocks-13.12mingw-setup.exe"!

2. 설치 및 실행

2.1 다운로드 (4/4)

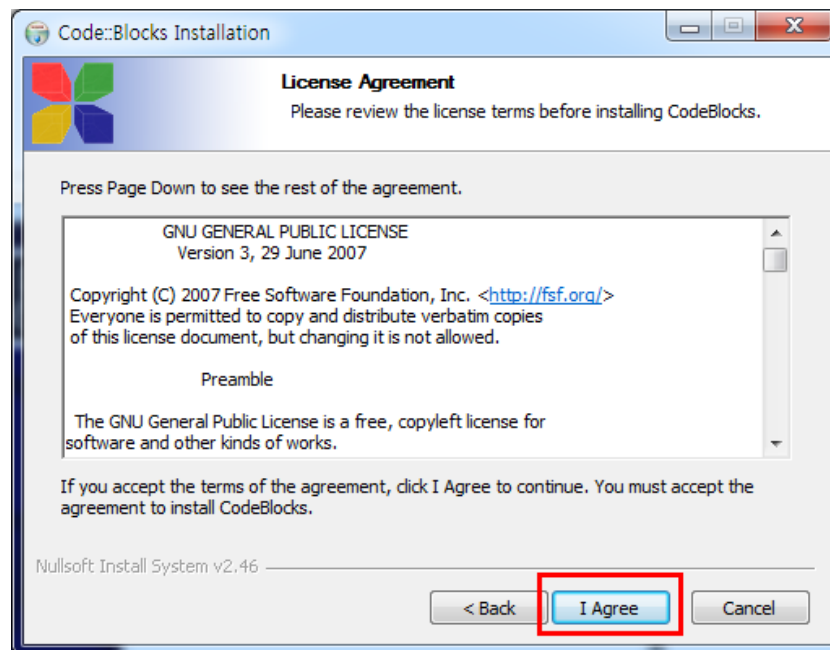
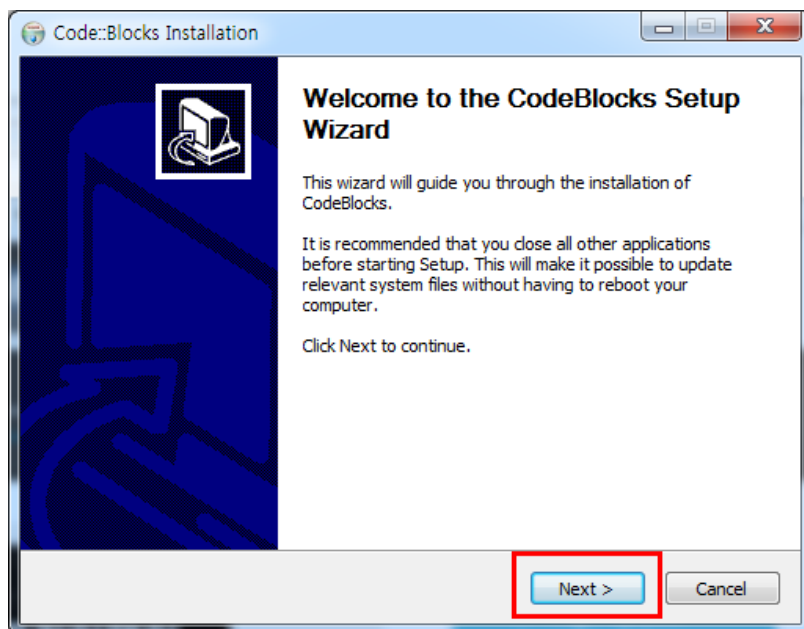
- 설치 파일 다운로드



2. 설치 및 실행

2.2 설치 (1/4)

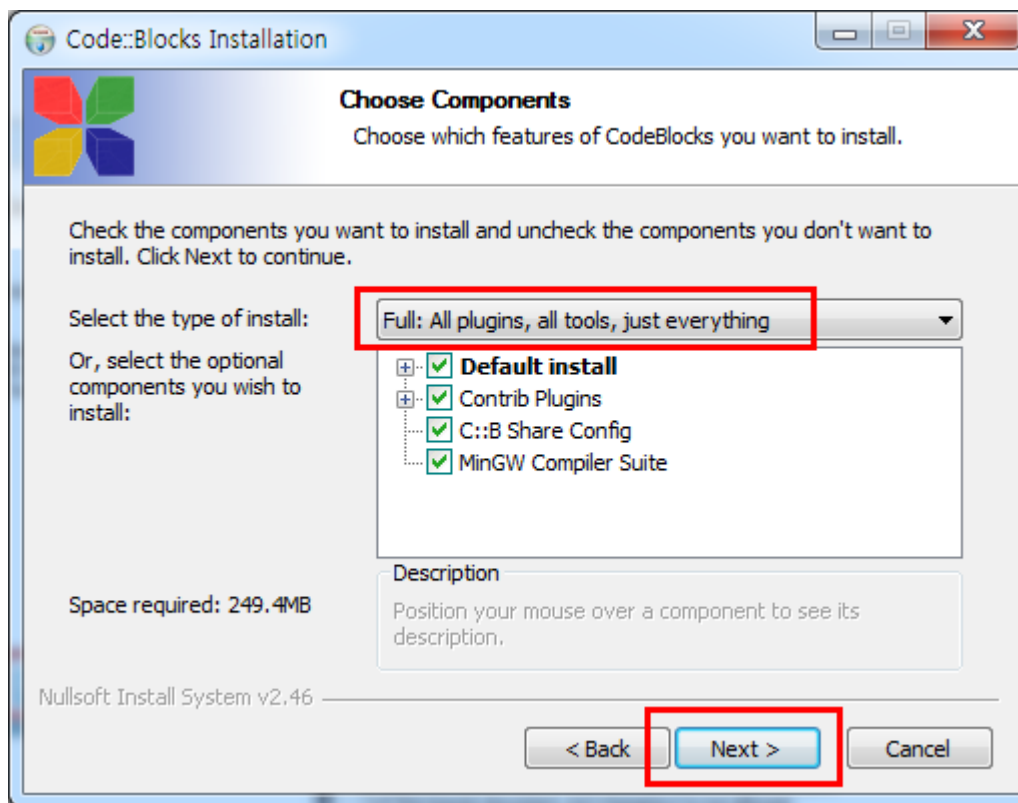
- 설치 파일 실행



2. 설치 및 실행

2.2 설치 (2/4)

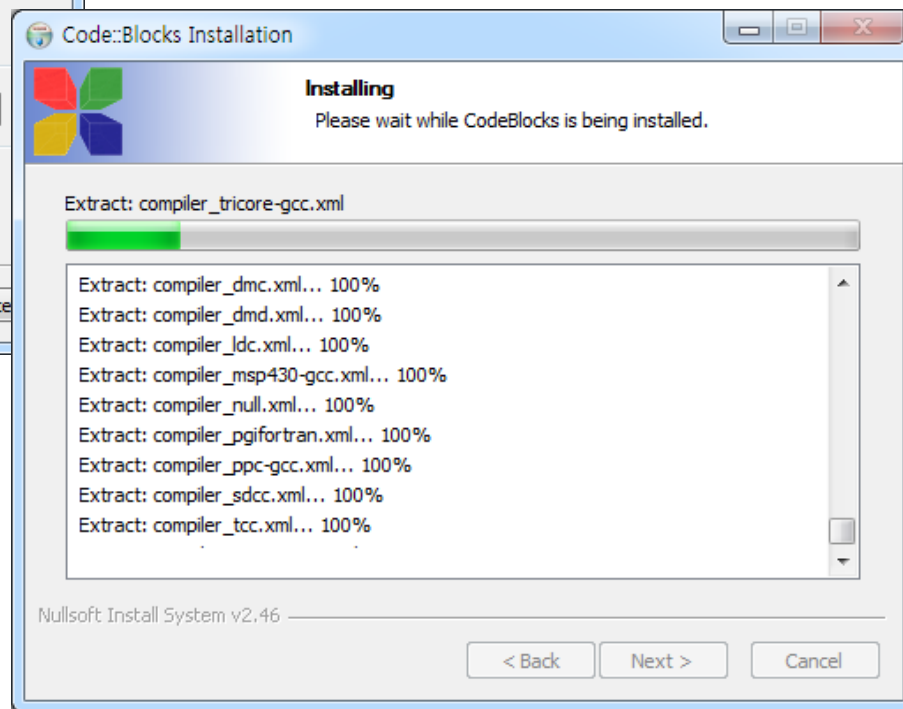
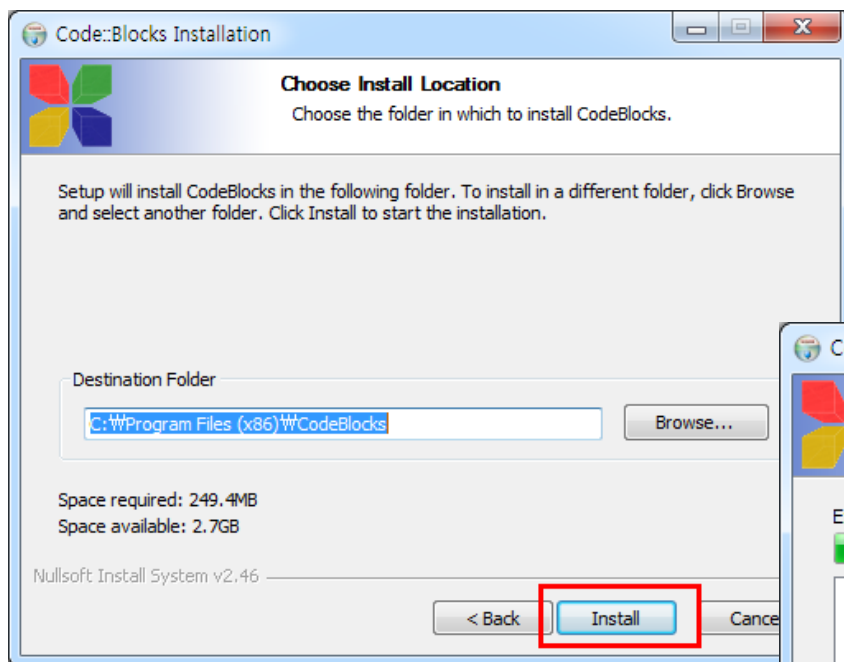
- 기본으로 선택되어 있는 전체 설치 옵션으로 설치



2. 설치 및 실행

2.2 설치 (3/4)

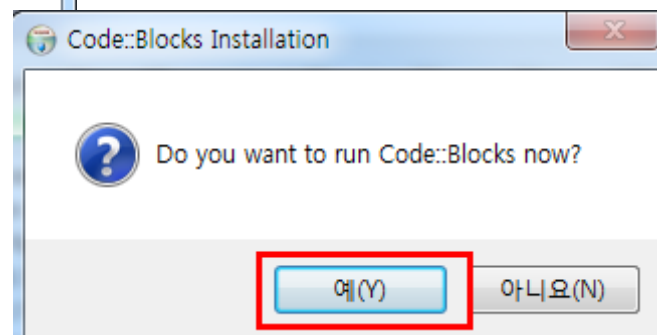
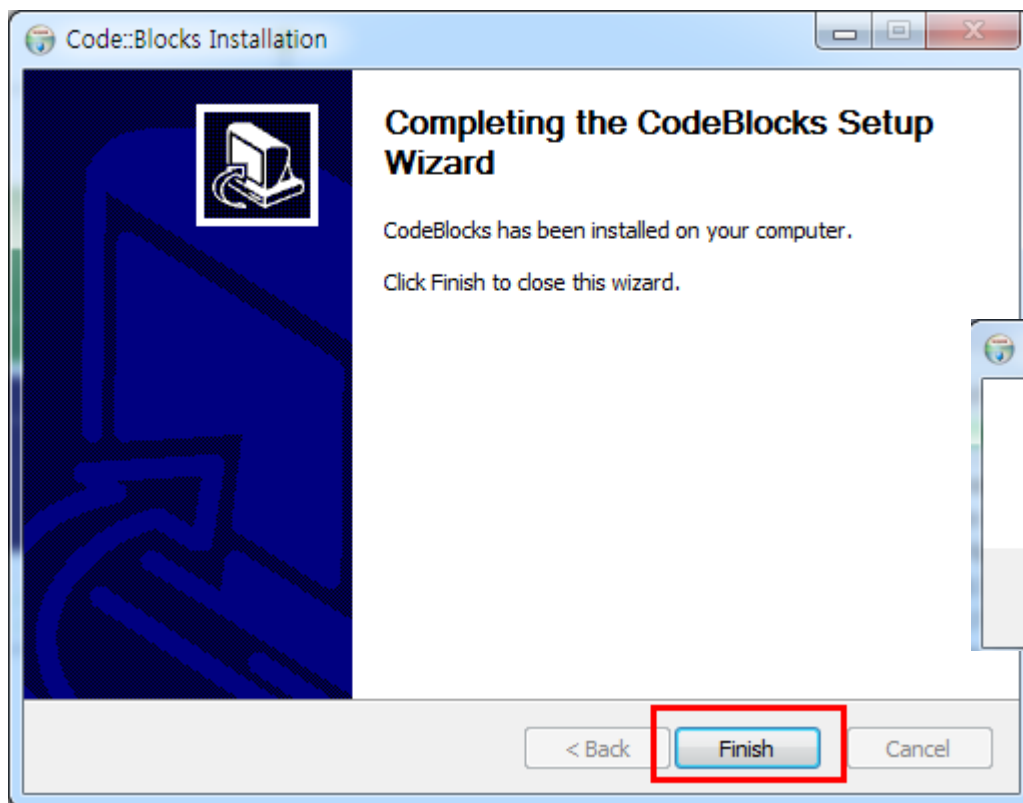
- 설치 위치 지정 및 설치 시작



2. 설치 및 실행

2.2 설치 (4/4)

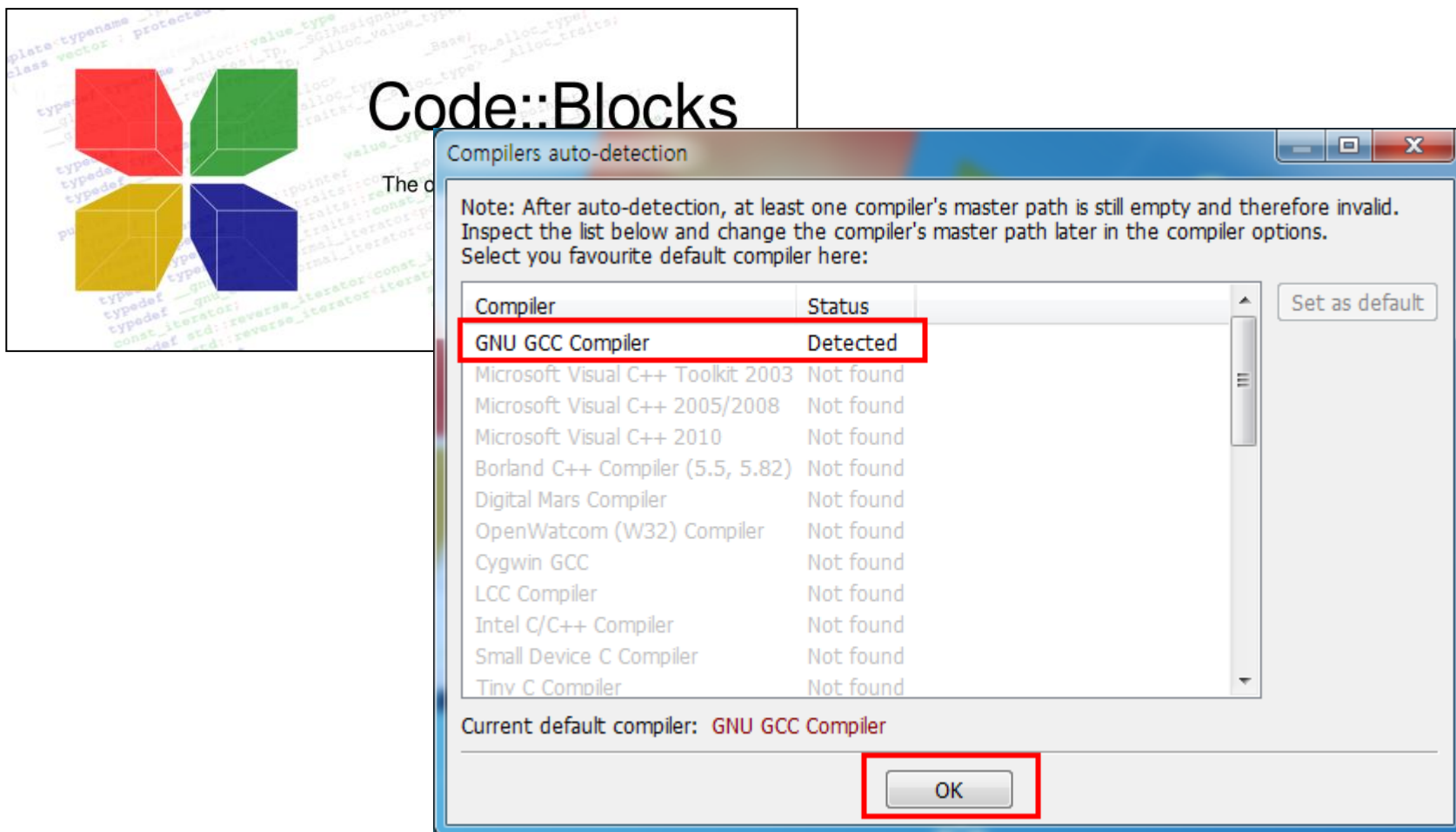
- 설치 완료 및 실행



2. 설치 및 실행

2.3 설치 확인 (1/3)

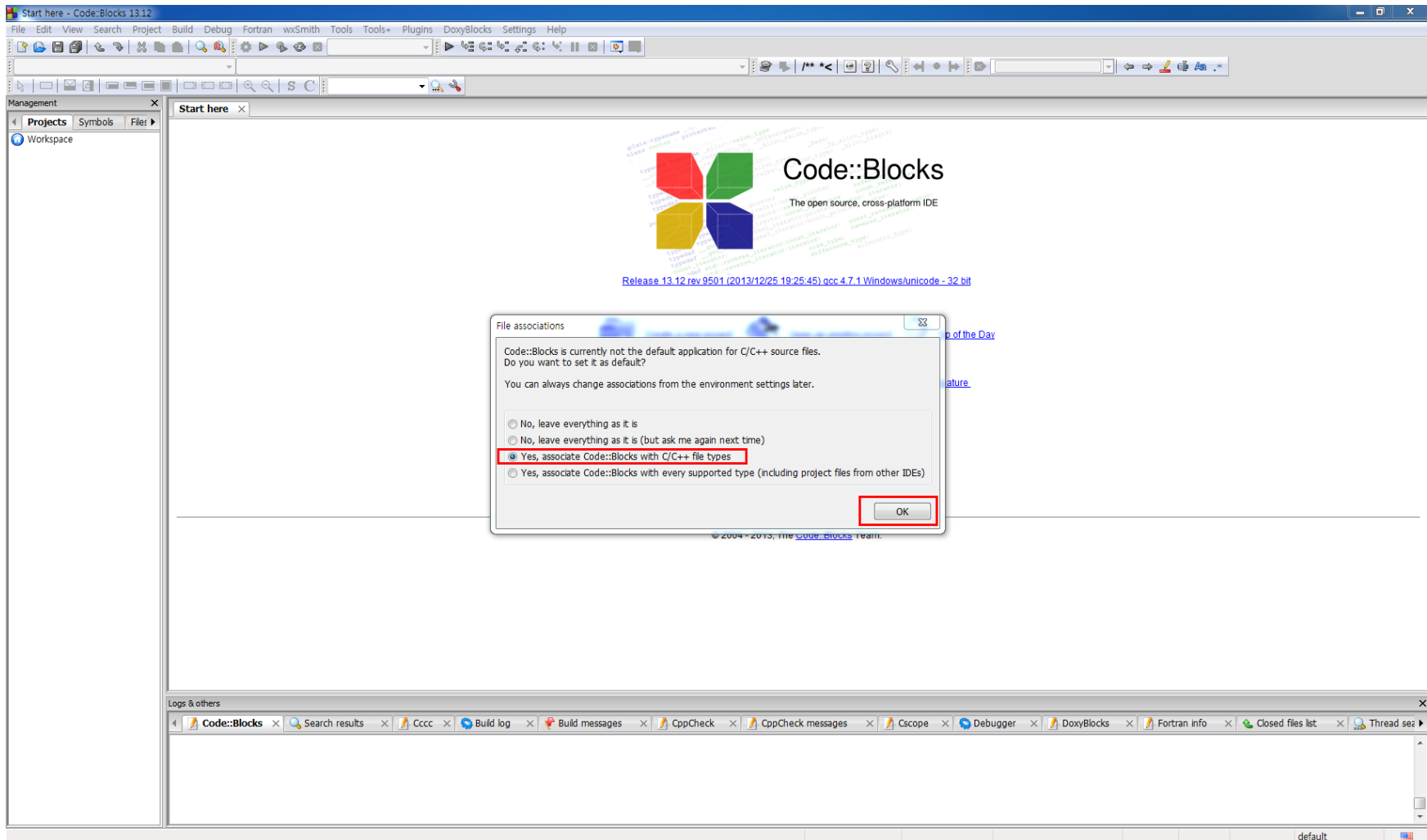
- 컴파일러 자동 감지 내용 확인 후 OK 클릭



2. 설치 및 실행

2.3 설치 확인 (2/3)

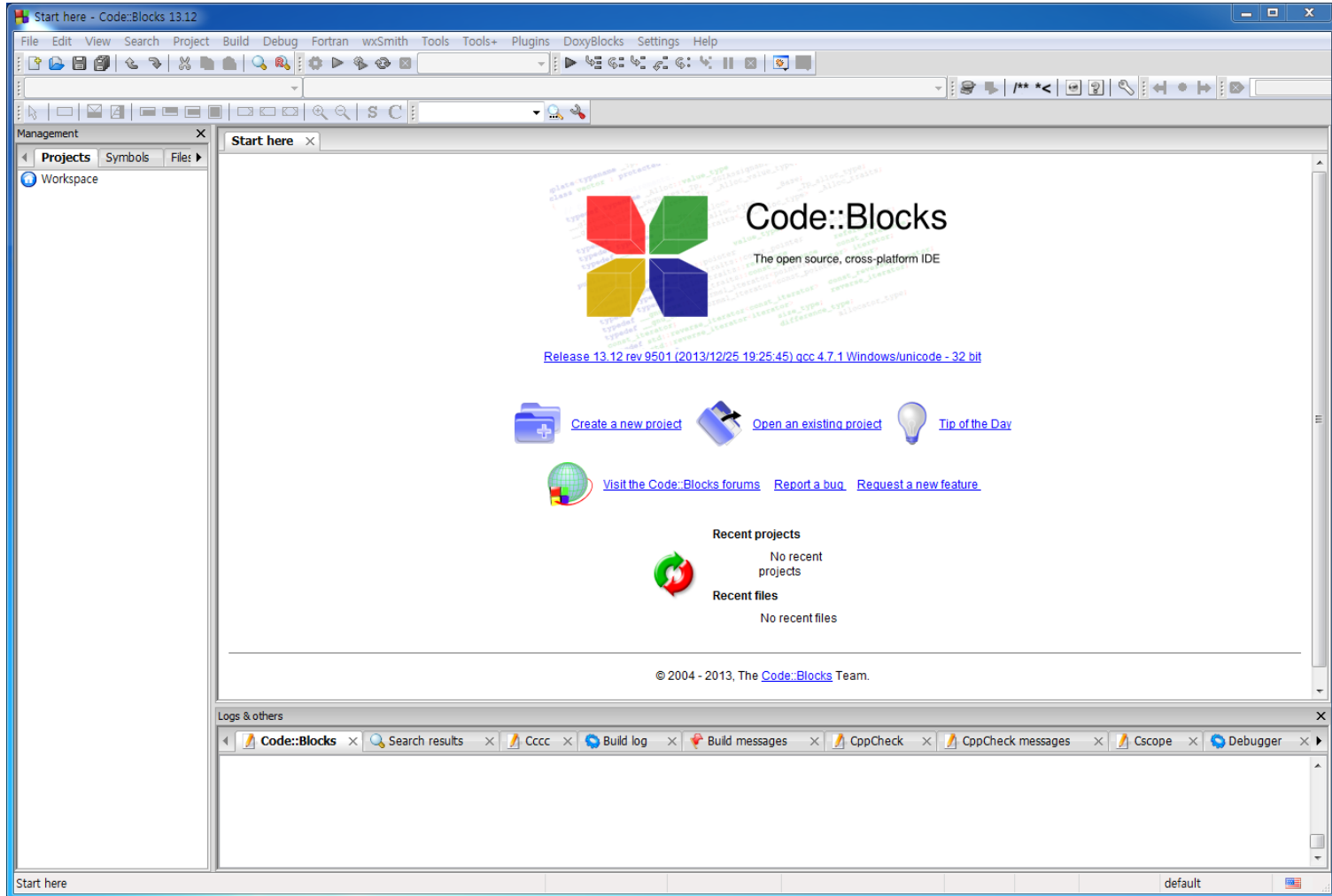
- 연결 파일 종류 설정 후 OK 클릭



2. 설치 및 실행

2.3 설치 확인 (3/3)

- 실행 화면



3. 주요 기능

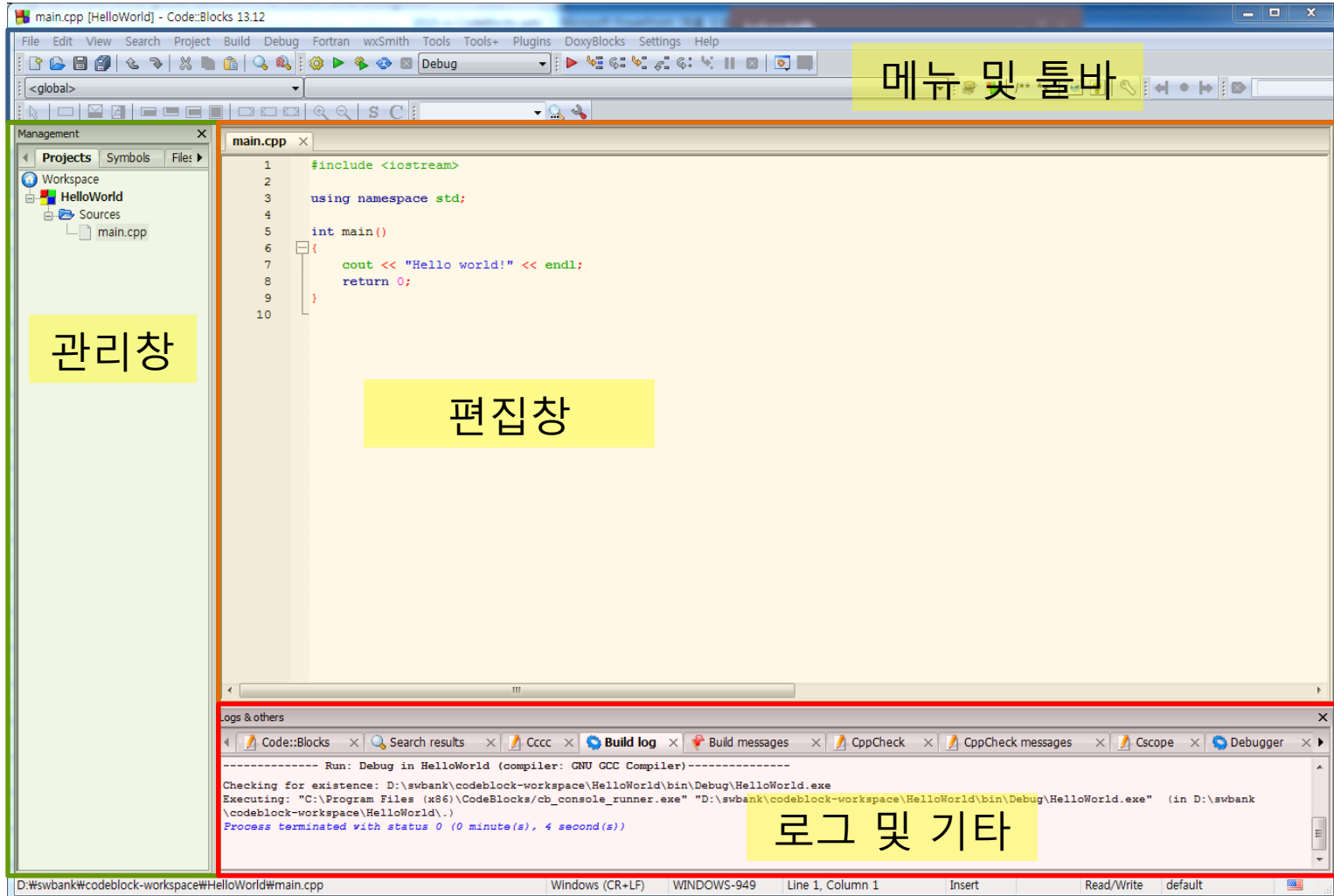
세부 목차

- 3.1 화면 레이아웃
- 3.2 소스 편집
- 3.3 컴파일 및 빌드
- 3.4 디버깅
- 3.5 다른 IDE 프로젝트 가져오기
- 3.6 플러그인

3. 주요 기능

3.1 화면 레이아웃

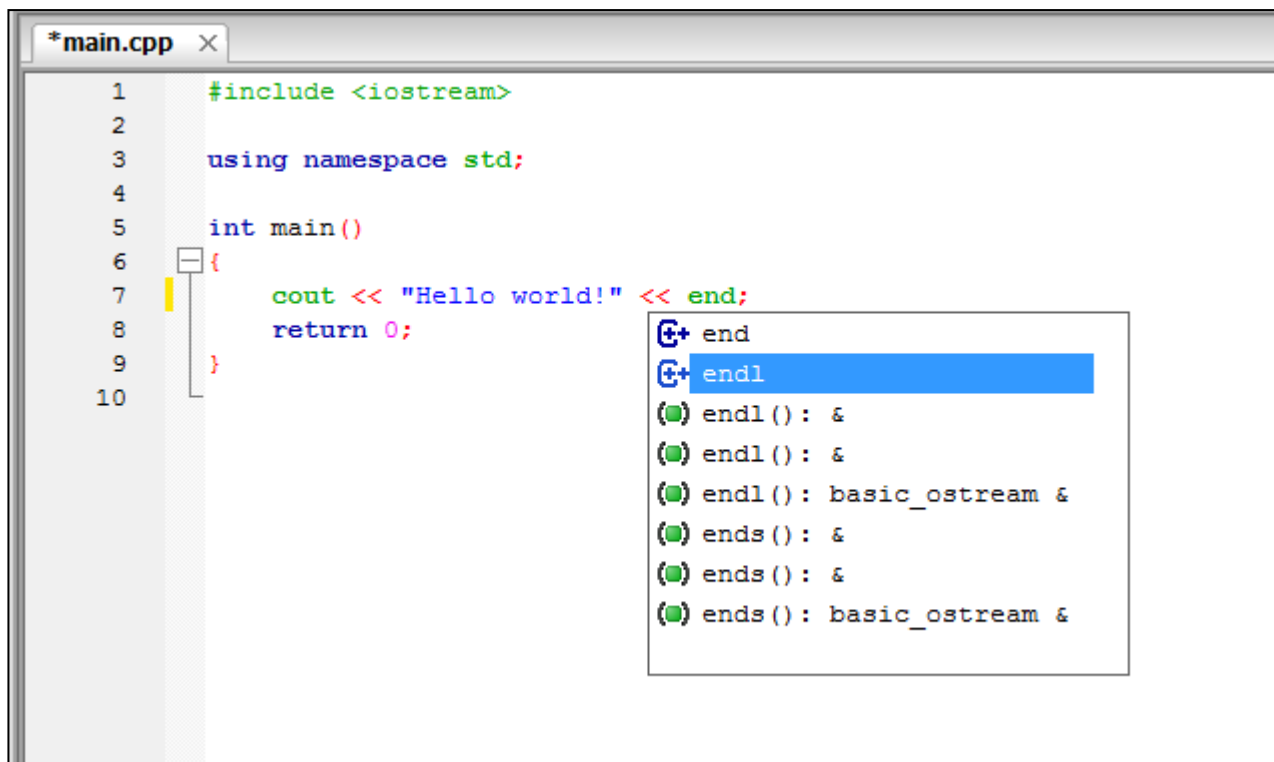
- 크게 메뉴 및 툴바, 관리창, 편집창, 로그 및 기타 창의 4구역으로 분류



3. 주요 기능

3.2 소스 편집

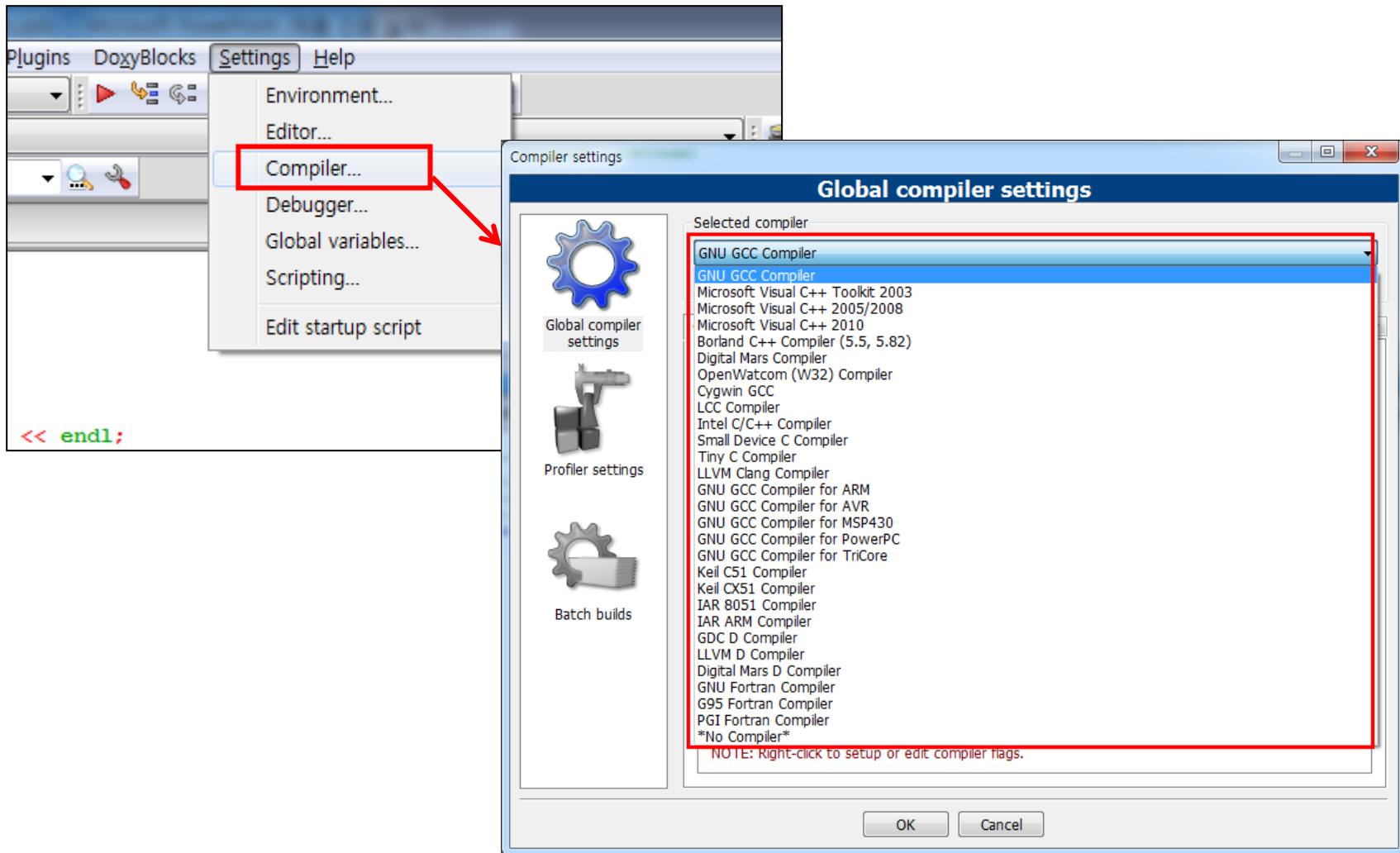
- 일반적인 IDE와 마찬가지로 탭을 이용한 다중 파일 편집, 코드 하이라이트, 코드 접기(folding), 코드 추천 기능 제공



3. 주요 기능

3.2 컴파일 및 빌드 (1/2)

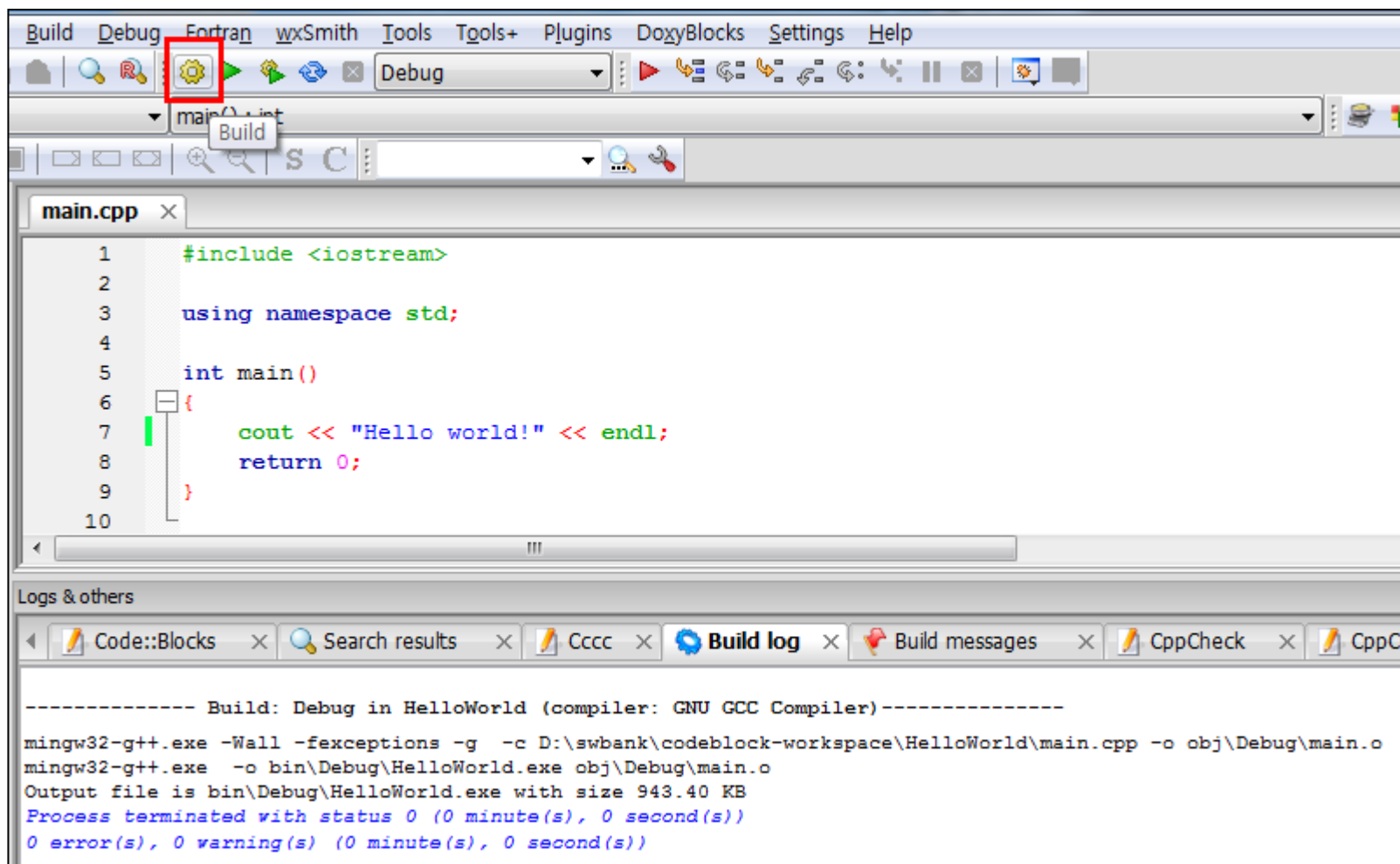
- 다양한 컴파일러를 지원한다.



3. 주요 기능

3.2 컴파일 및 빌드 (2/2)

- Build 명령을 통해 컴파일과 빌드를 수행



- 디버그 실행 모드 지원
 - Step into, Step out, Step over 등 실행 위치 이동 기능

Run to Cursor
커서 위치까지 실행

Next Line
다음 행까지 실행

Step into
루틴 내부로 이동

Step out
루틴 빠져나오기

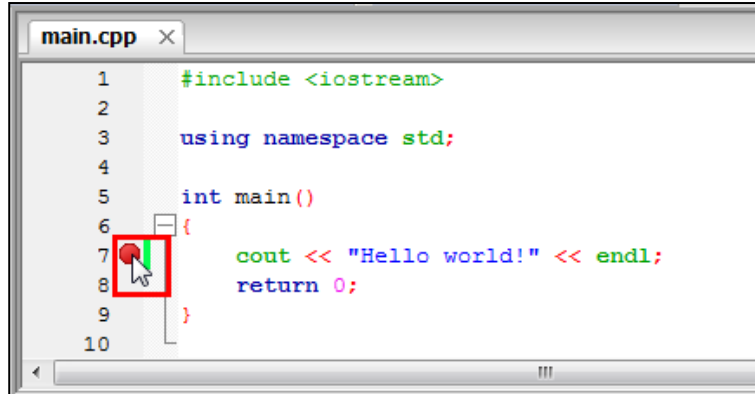
Next instruction
다음 명령

Step into instruction
명령 내부로 이동

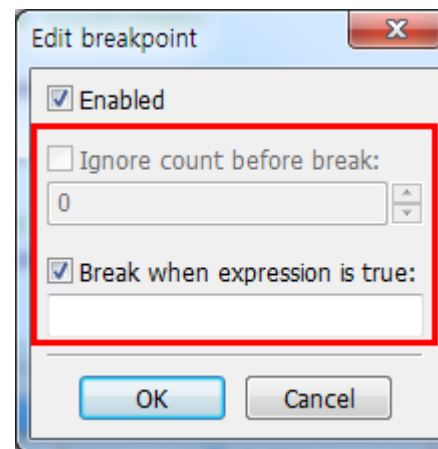
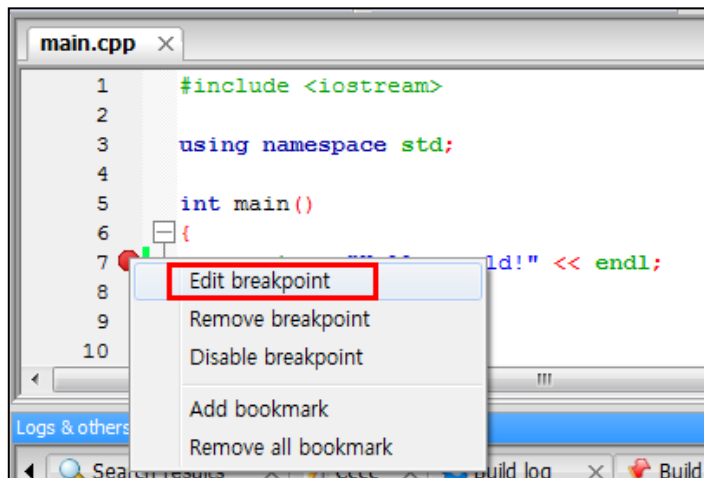
3. 주요 기능

3.4 디버깅 (2/4)

- Breakpoint(중단점) 설정
 - 행 번호 우측 지점을 클릭해서 설정



- Breakpoint를 우클릭 해서 조건 설정

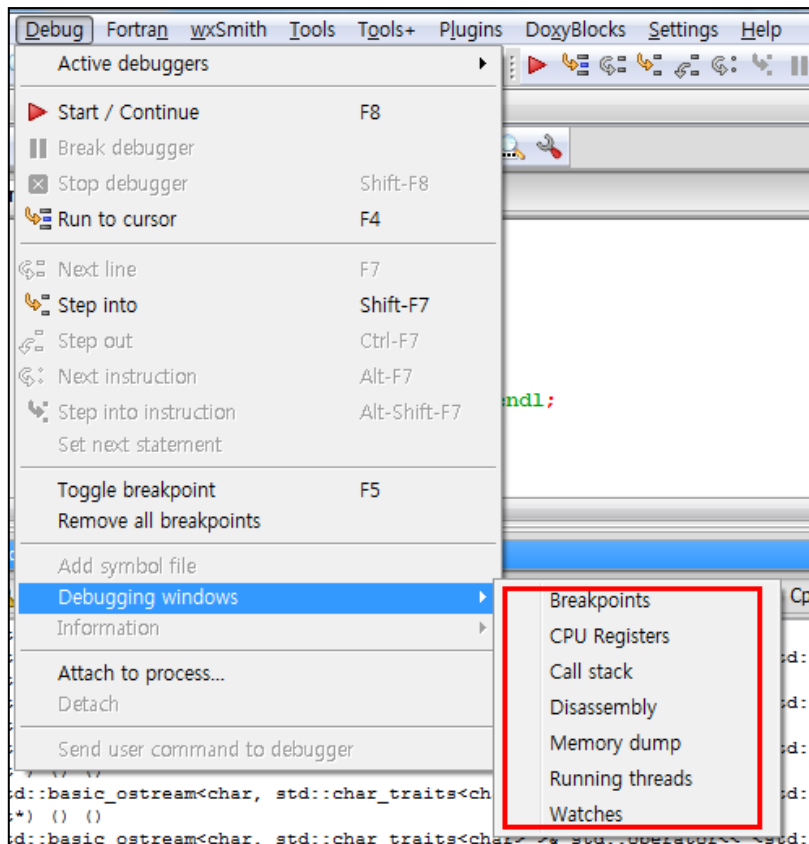


카운트와 조건
중 택일해서
적용 가능

3. 주요 기능

3.4 디버깅 (3/4)

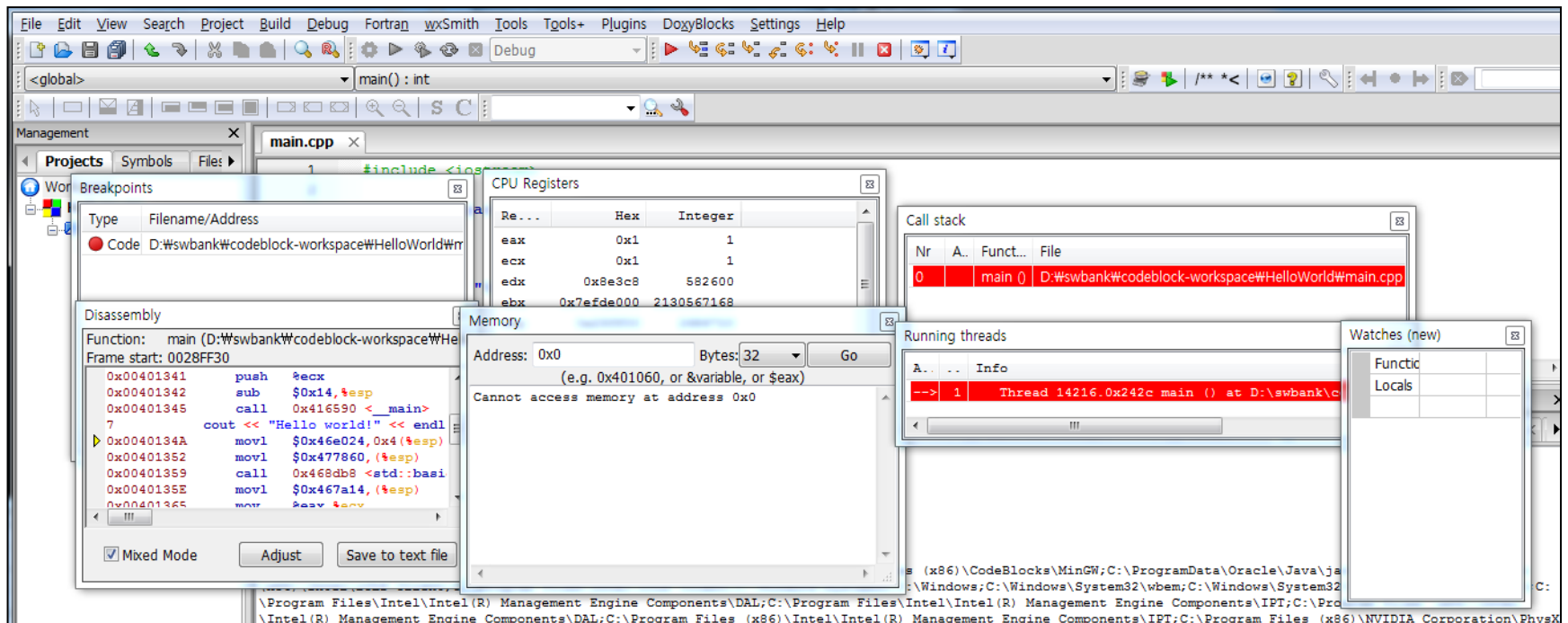
- 디버깅을 지원하는 다양한 기능 제공
 - Breakpoints 목록, CPU Register, Call Stack, Disassembly, Memory dump, Running threads, Watches



3. 주요 기능

3.4 디버깅 (4/4)

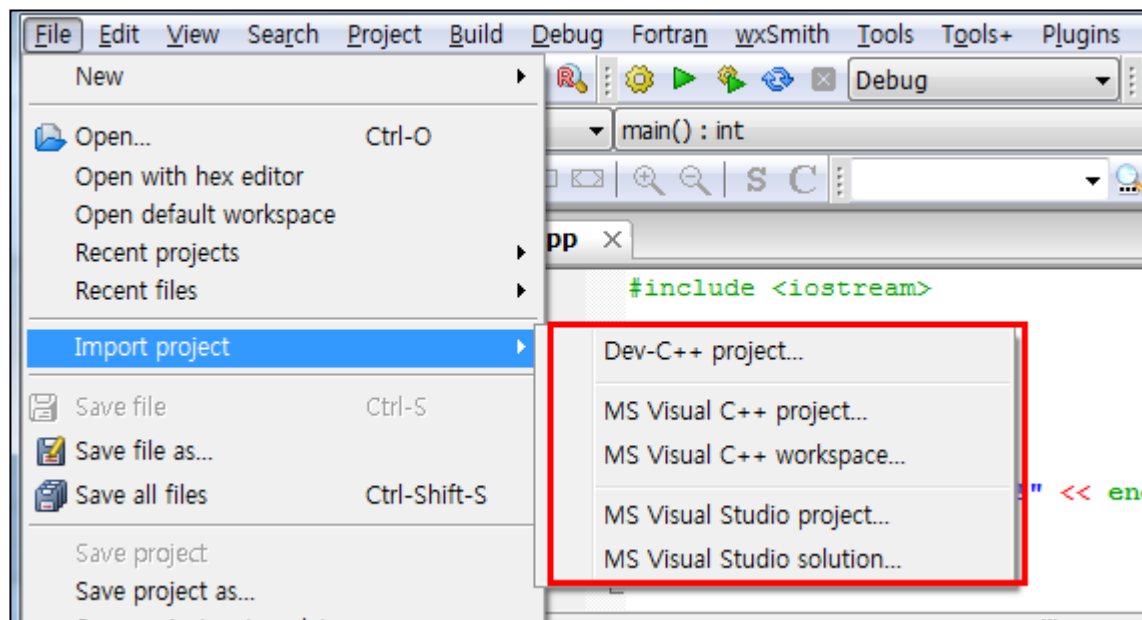
- 디버깅을 지원하는 다양한 기능 제공
 - Breakpoints 목록, CPU Register, Call Stack, Disassembly, Memory dump, Running threads, Watches



3. 주요 기능

3.5 다른 IDE 프로젝트 가져오기

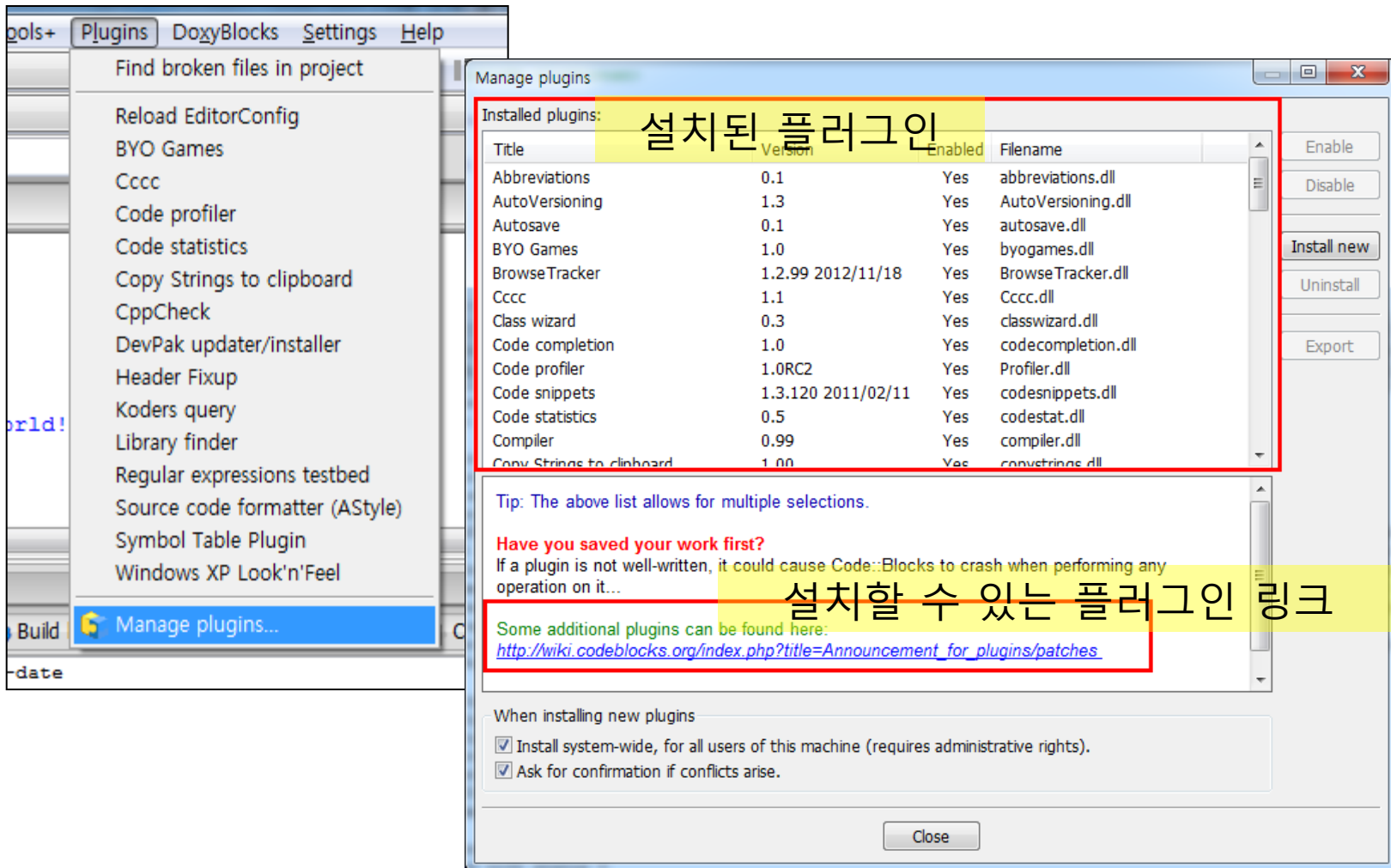
- 다른 IDE에서 작업 중인 프로젝트나 워크스페이스를 가져올 수 있다.



3. 주요 기능

3.6 플러그인 (1/2)

- 다른 IDE에서 작업 중인 프로젝트나 워크스페이스를 가져올 수 있다.



The screenshot shows the Code::Blocks IDE interface. The 'Plugins' menu is open, displaying a list of available plugins. The 'Manage plugins...' option is highlighted at the bottom. A 'Manage plugins' dialog box is overlaid on the main window. It features a table of installed plugins, a list of additional plugins with a link to a wiki page, and checkboxes for installation options.

설치된 플러그인

Title	Version	Enabled	Filename
Abbreviations	0.1	Yes	abbreviations.dll
AutoVersioning	1.3	Yes	AutoVersioning.dll
Autosave	0.1	Yes	autosave.dll
BYO Games	1.0	Yes	byogames.dll
BrowseTracker	1.2.99 2012/11/18	Yes	BrowseTracker.dll
Cccc	1.1	Yes	Cccc.dll
Class wizard	0.3	Yes	classwizard.dll
Code completion	1.0	Yes	codecompletion.dll
Code profiler	1.0RC2	Yes	Profiler.dll
Code snippets	1.3.120 2011/02/11	Yes	codesnippets.dll
Code statistics	0.5	Yes	codestat.dll
Compiler	0.99	Yes	compiler.dll
Copy Strings to clipboard	1.00	Yes	copystrings.dll

설치할 수 있는 플러그인 링크

Some additional plugins can be found here:
http://wiki.codeblocks.org/index.php?title=Announcement_for_plugins/patches

When installing new plugins

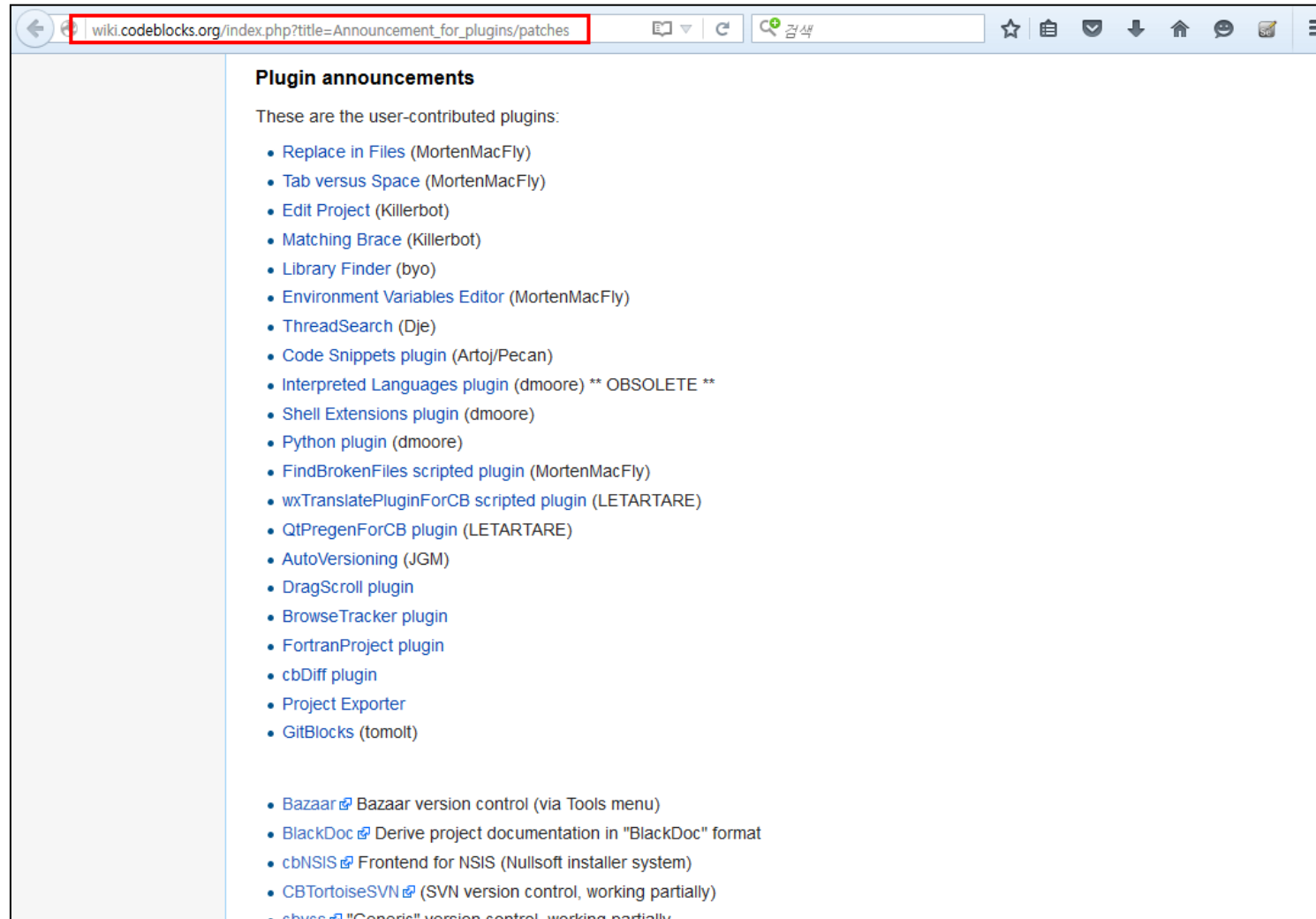
- ☒ Install system-wide, for all users of this machine (requires administrative rights).
- ☒ Ask for confirmation if conflicts arise.

Close

3. 주요 기능

3.6 플러그인 (2/2)

- 설치할 수 있는 플러그인 목록



4. 활용 예제

세부 목차

- 4.1 예제 소개
- 4.2 프로젝트 생성
- 4.3 예제 소스 작성
- 4.4 디버그
- 4.5 예제 실행

4. 활용 예제

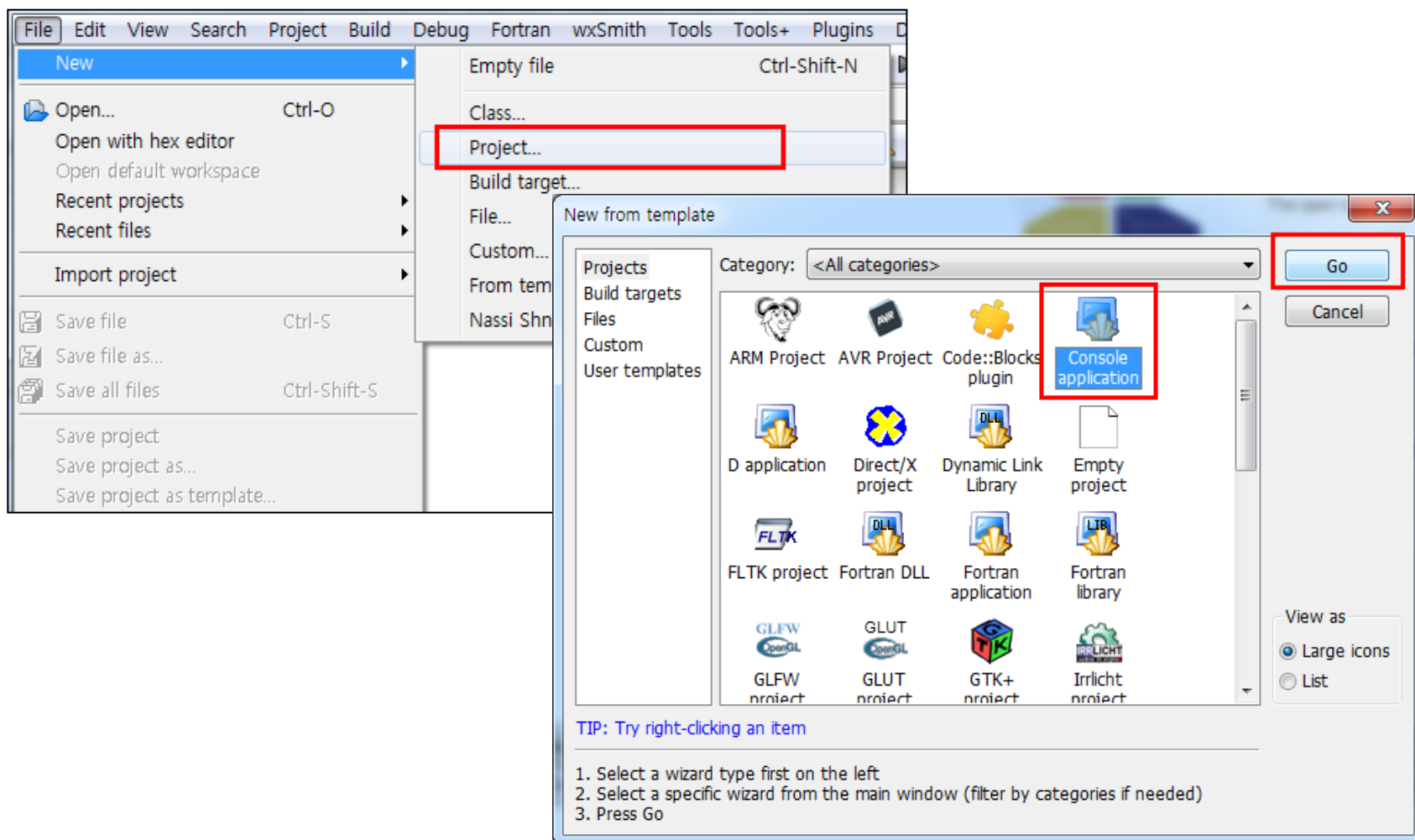
4.1 예제 소개

- 콘솔 프로그램을 C++로 작성할 수 있는 프로젝트를 생성하고,
- 디자인 패턴 중의 하나인 컴포지트(Composite) 패턴을 구현하고,
- 디버깅 방법을 알아보고,
- 예제를 실행해서 콘솔에서 결과를 확인한다.

4. 활용 예제

4.2 프로젝트 생성 (1/5)

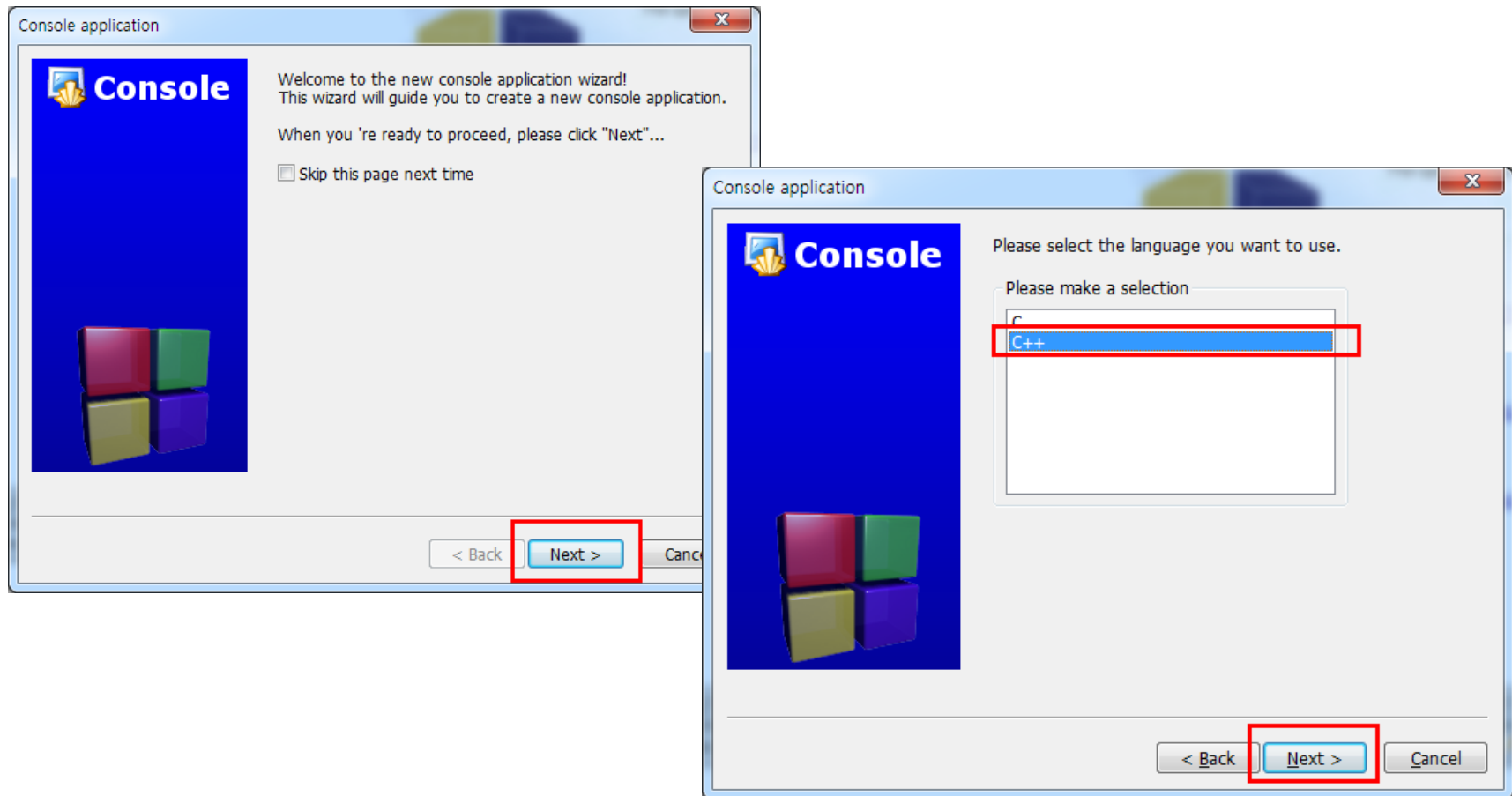
- 콘솔 프로그램을 작성할 수 있는 프로젝트를 생성한다.



4. 활용 예제

4.2 프로젝트 생성 (2/5)

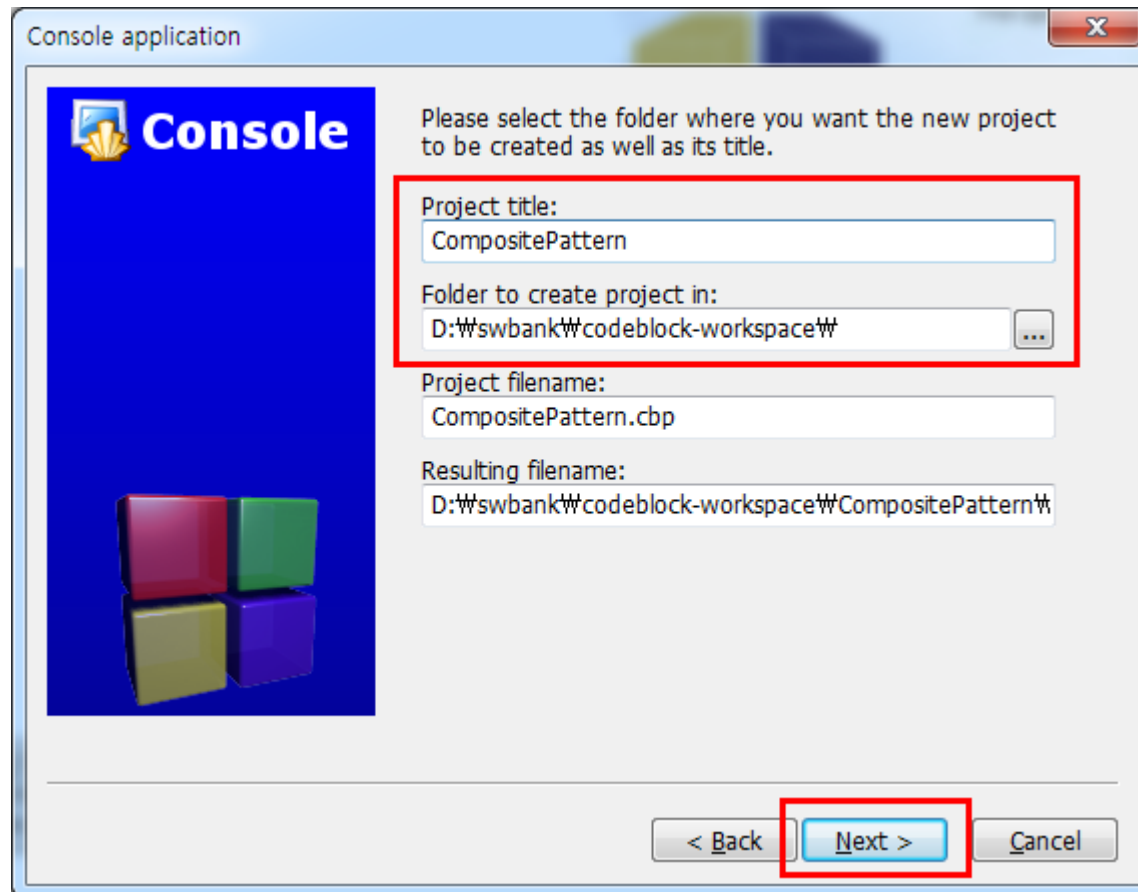
- 콘솔 애플리케이션 마법사가 실행된다.
- 언어는 C++을 선택한다.



4. 활용 예제

4.2 프로젝트 생성 (3/5)

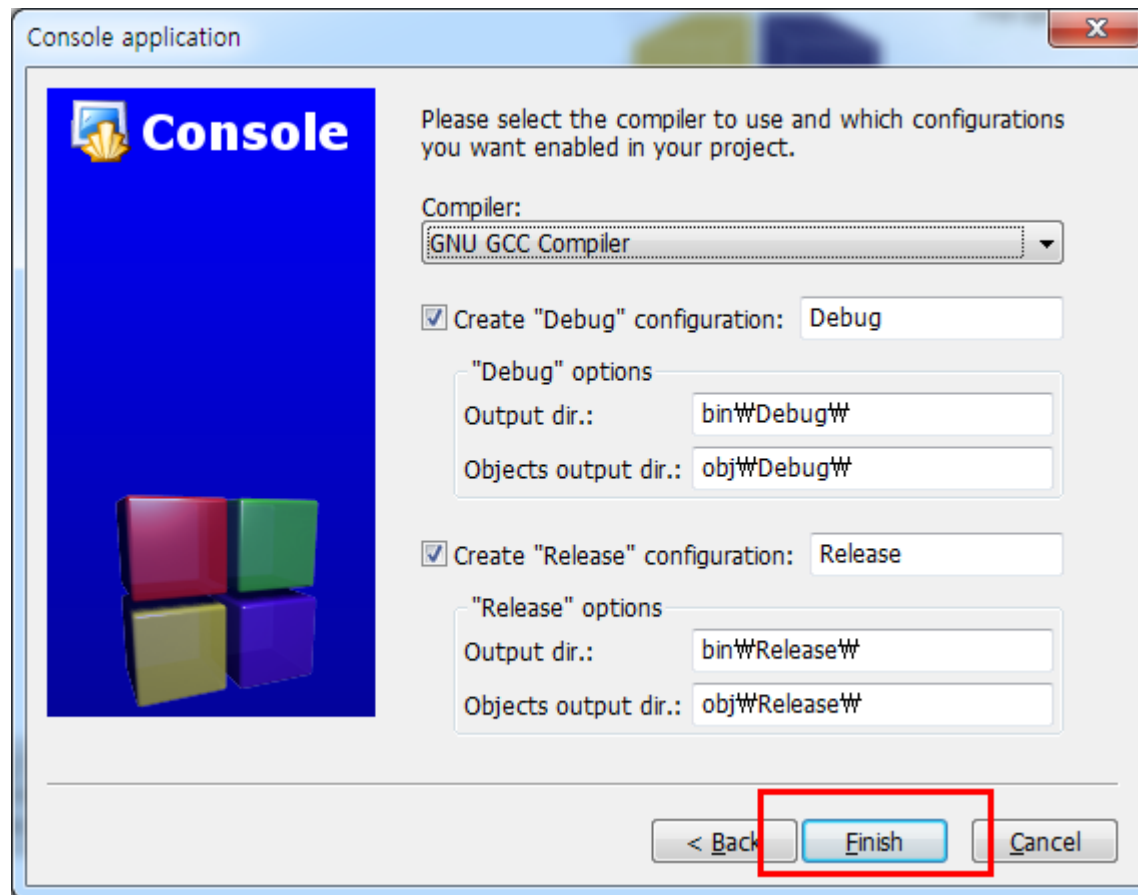
- 프로젝트 이름과 프로젝트 위치를 지정하고 Next를 클릭한다.
 - 아래의 두 가지 항목은 자동으로 입력된다.



4. 활용 예제

4.2 프로젝트 생성 (4/5)

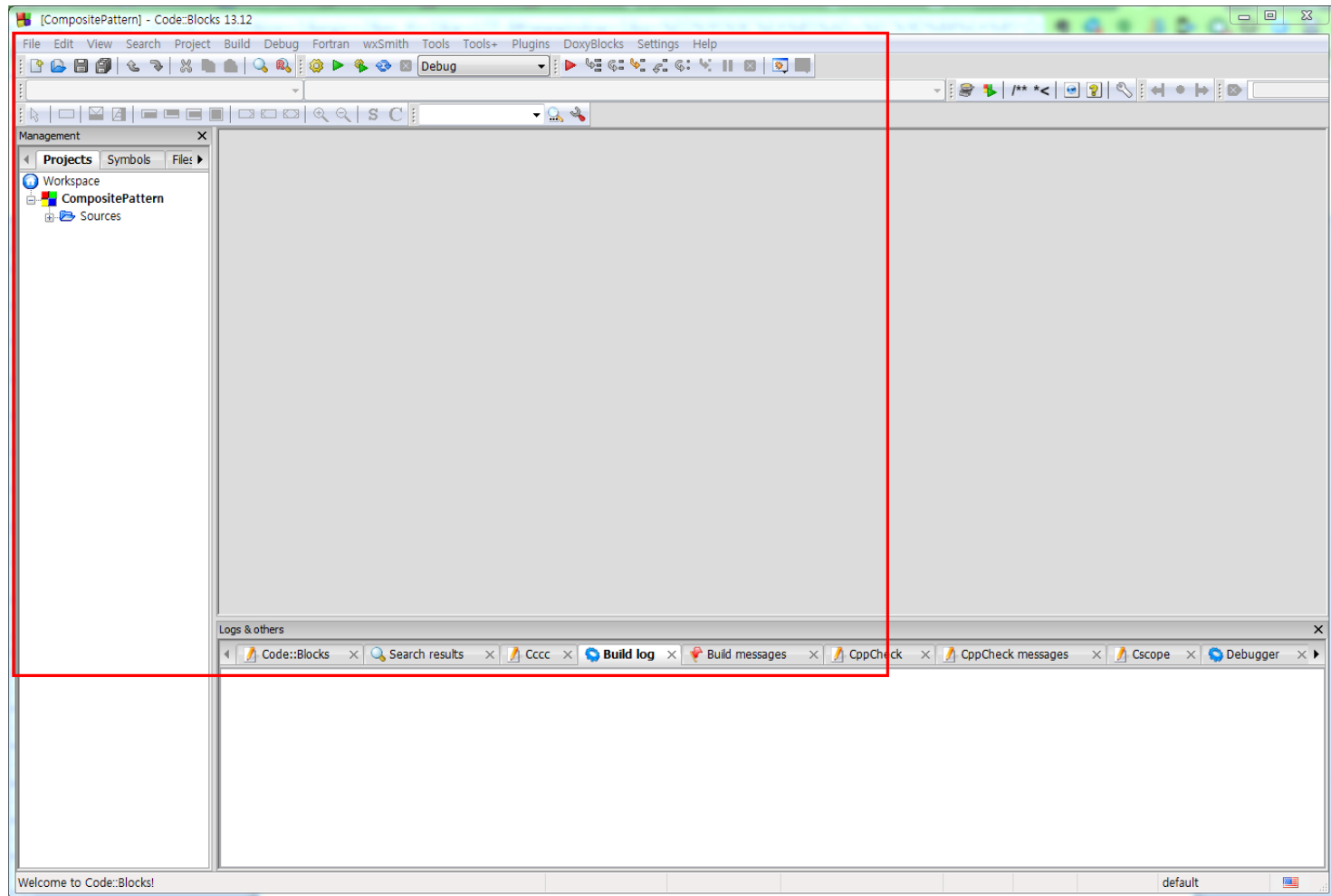
- 프로젝트에서 사용할 컴파일러와 디버그 및 배포 관련 내용을 설정한다.
 - 일단 주어진 기본값대로 설치한다. 나중에 변경 가능하다.



4. 활용 예제

4.2 프로젝트 생성 (5/5)

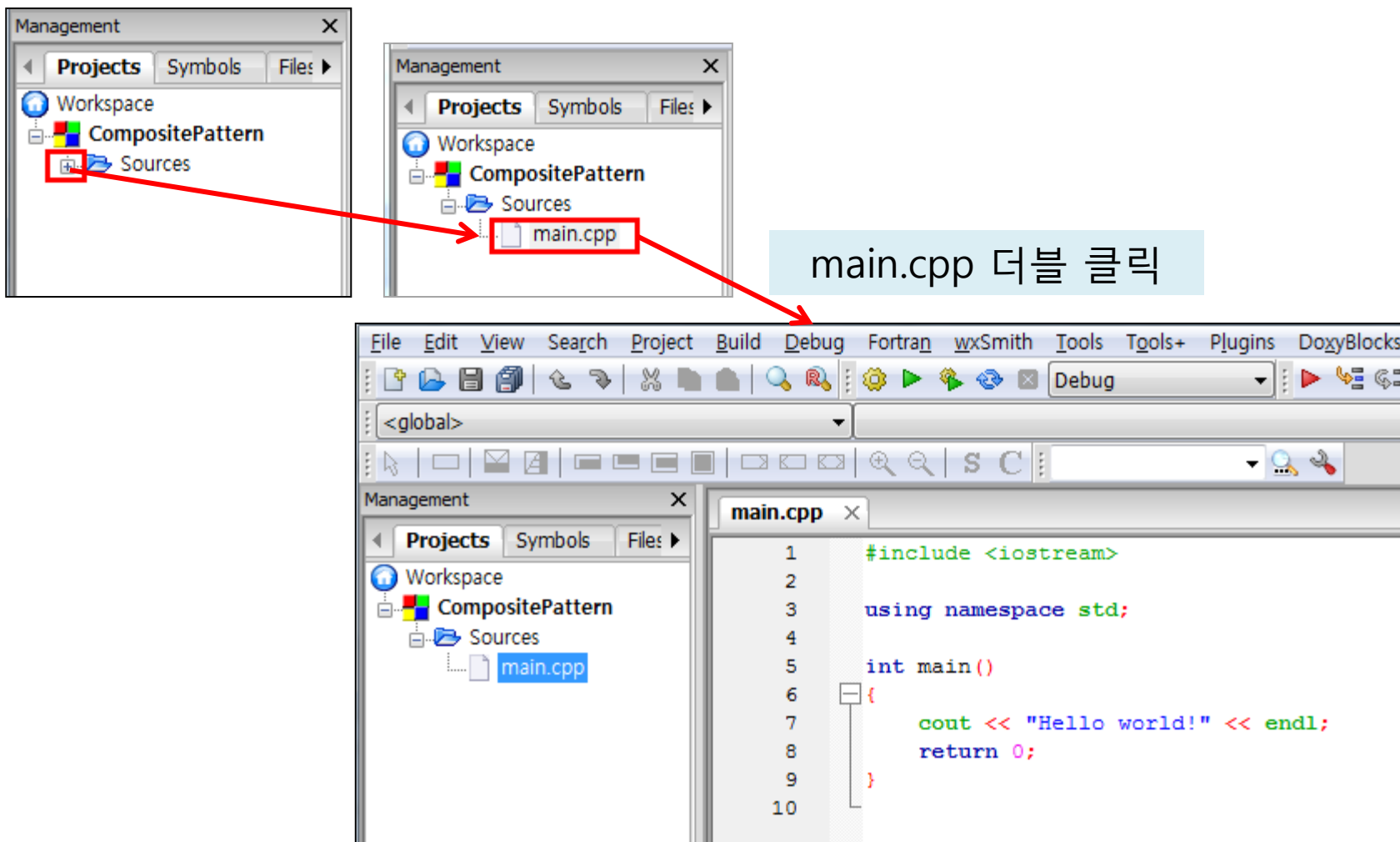
- 프로젝트 생성이 완료된다.



4. 활용 예제

4.3 예제 소스 작성 (1/2)

- Sources 폴더를 클릭하면 Hello World 예제 소스가 표시된다.



The screenshot illustrates the process of opening the `main.cpp` file in Code::Blocks. It shows two 'Management' panels. The first panel shows the 'Sources' folder selected under the 'CompositePattern' project. A red arrow points from this folder to the second 'Management' panel, which shows the `main.cpp` file selected. A text box with the text 'main.cpp 더블 클릭' (Double-click main.cpp) has an arrow pointing to the `main.cpp` file in the second panel. Below these panels is the main editor window, which displays the code for `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
```

4. 활용 예제

4.3 예제 소스 작성 (2/2)

- main.cpp 파일을 아래의 내용으로 대체한다.

```
#include <iostream>
#include <vector>
#include <string>

using std::cout;
using std::vector;
using std::string;

class Component
{
public:
    virtual void list() const = 0;
    virtual ~Component(){};
};

class Leaf : public Component
{
public:
    explicit Leaf(int val) : value_(val)
    {
    }
    void list() const
    {
        cout << "  " << value_ << "\n";
    }
private:
    int value_;
};
```

```
class Composite : public Component
{
public:
    explicit Composite(string id) : id_(id)
    {
    }
    void add(Component *obj)
    {
        table_.push_back(obj);
    }
    void list() const
    {
        cout << id_ << ":" << "\n";
        for (vector<Component*>::const_iterator
it = table_.begin();
it != table_.end(); ++it)
        {
            (*it)->list();
        }
    }
private:
    vector <Component*> table_;
    string id_;
};
```

```
int main()
{
    Leaf num0(0);
    Leaf num1(1);
    Leaf num2(2);
    Leaf num3(3);
    Leaf num4(4);
    Composite container1("Container
1");
    Composite container2("Container
2");

    container1.add(&num0);
    container1.add(&num1);

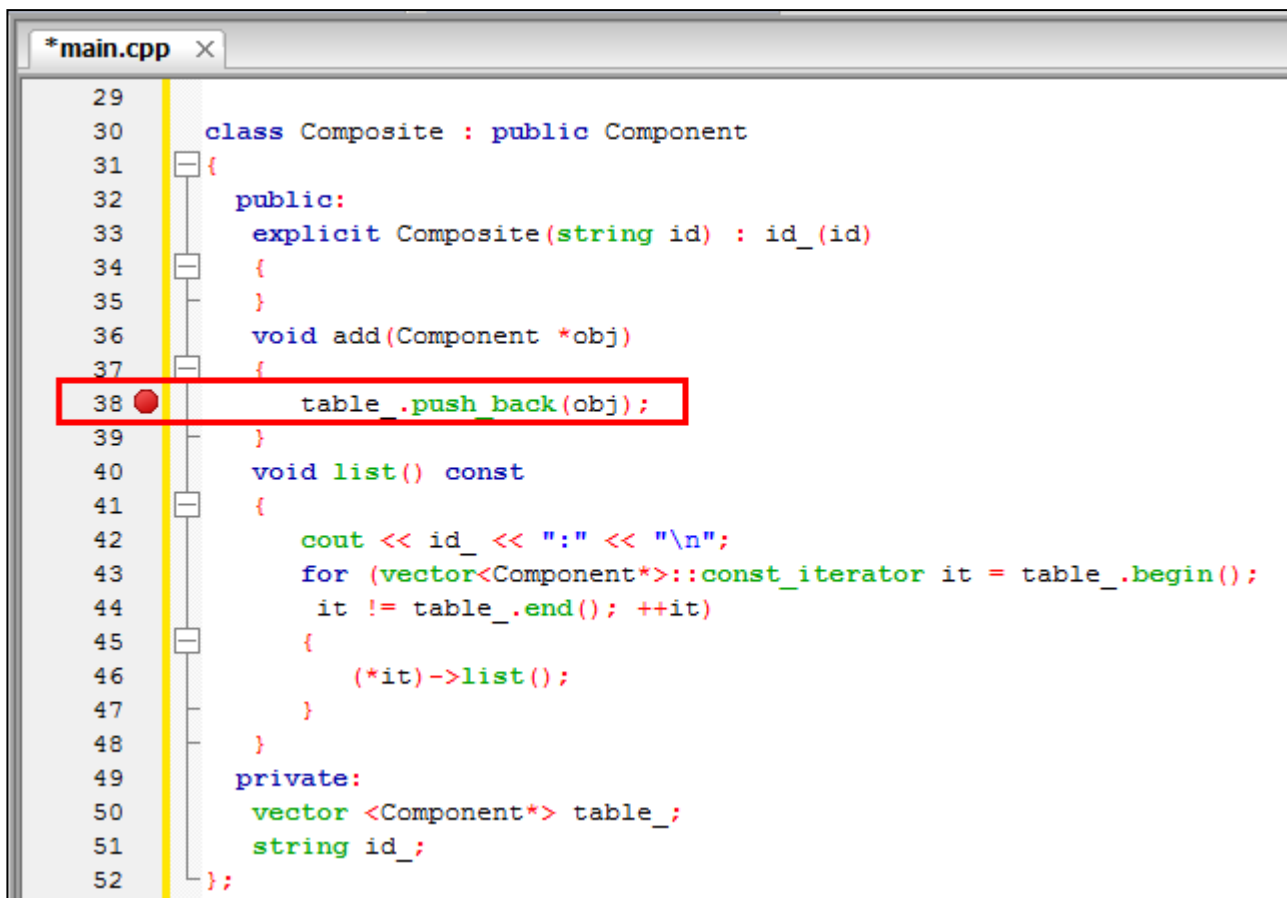
    container2.add(&num2);
    container2.add(&num3);
    container2.add(&num4);

    container1.add(&container2);
    container1.list();
    return 0;
}
```


4. 활용 예제

4.4 디버깅 (1/10)

- main.cpp 의 38행에 Breakpoint를 설정한다.

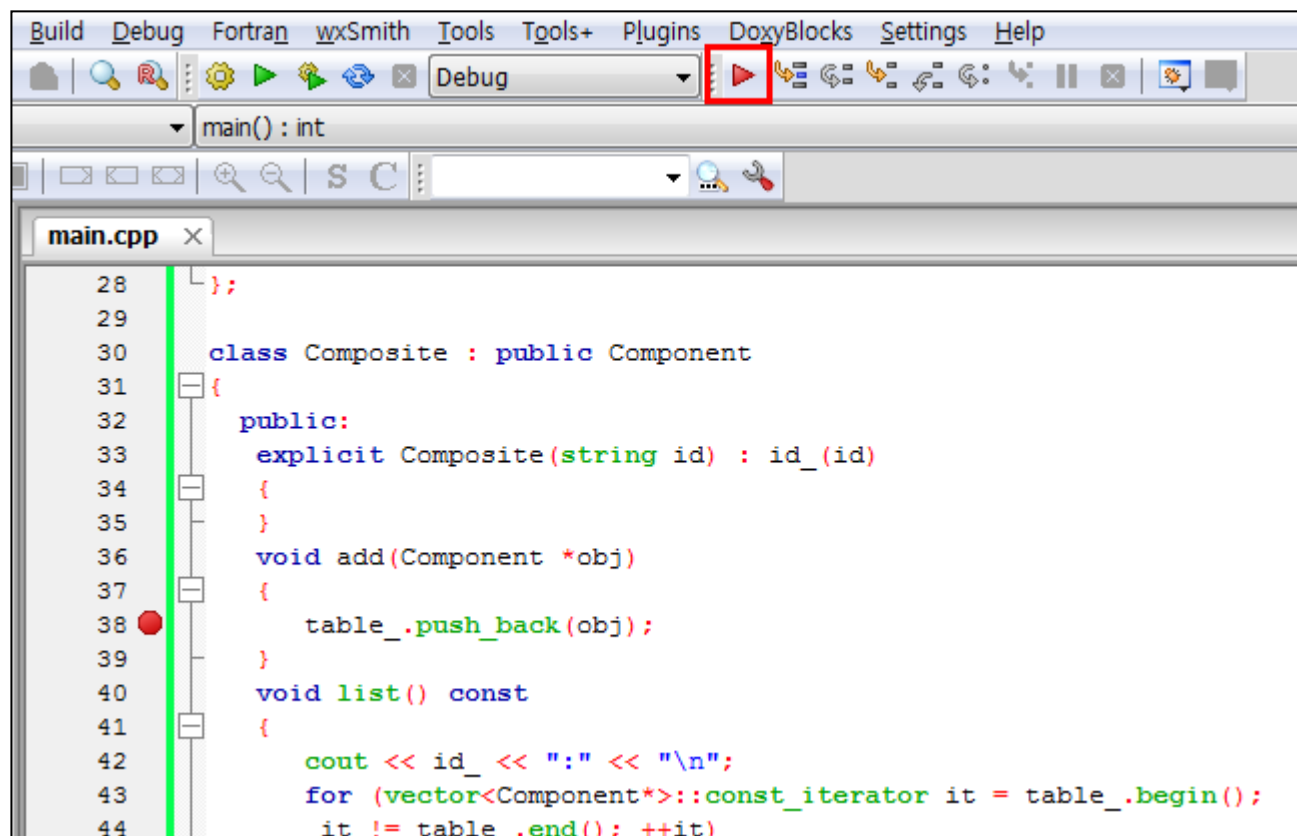


```
*main.cpp x
29
30 class Composite : public Component
31 {
32 public:
33     explicit Composite(string id) : id_(id)
34     {
35     }
36     void add(Component *obj)
37     {
38         table_.push_back(obj);
39     }
40     void list() const
41     {
42         cout << id_ << ":" << "\n";
43         for (vector<Component*>::const_iterator it = table_.begin();
44             it != table_.end(); ++it)
45         {
46             (*it)->list();
47         }
48     }
49 private:
50     vector<Component*> table_;
51     string id_;
52 };
```

4. 활용 예제

4.4 디버깅 (2/10)

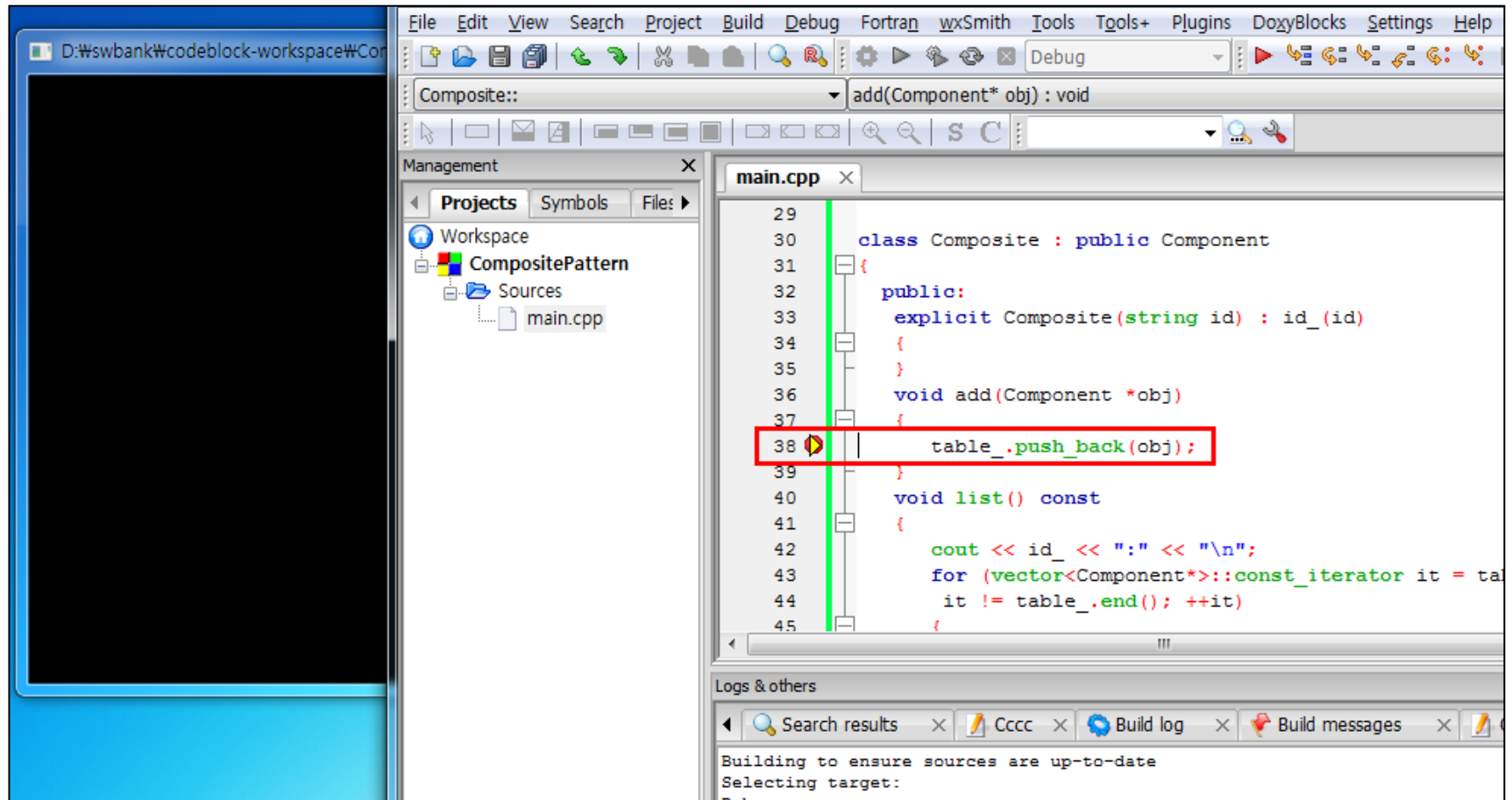
- 빨간색 삼각형 버튼을 클릭하여 디버그 모드로 실행



4. 활용 예제

4.4 디버깅 (3/10)

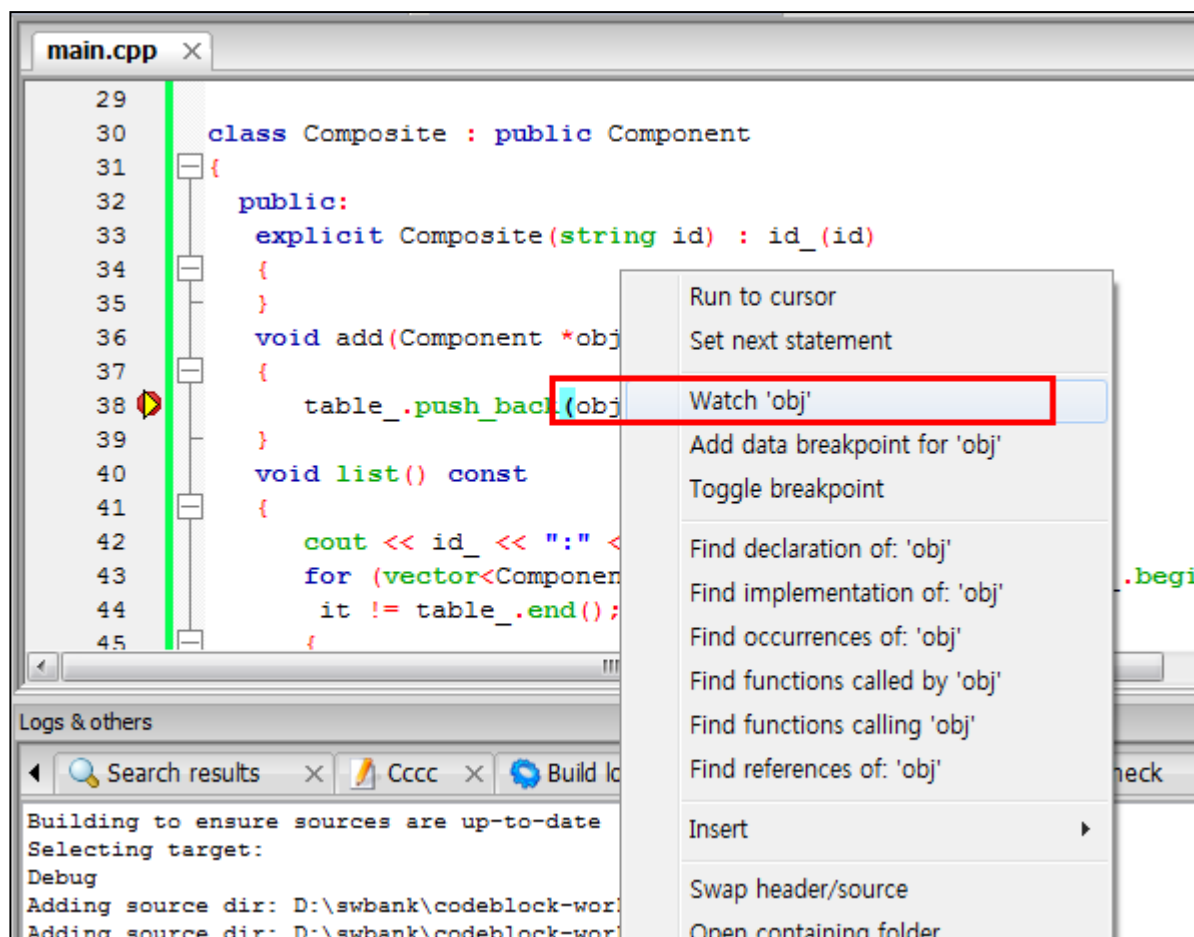
- 콘솔 창이 실행되며, 38행의 Breakpoint에 노란색 삼각형이 표시되며 실행이 멈춘다.



4. 활용 예제

4.4 디버깅 (4/10)

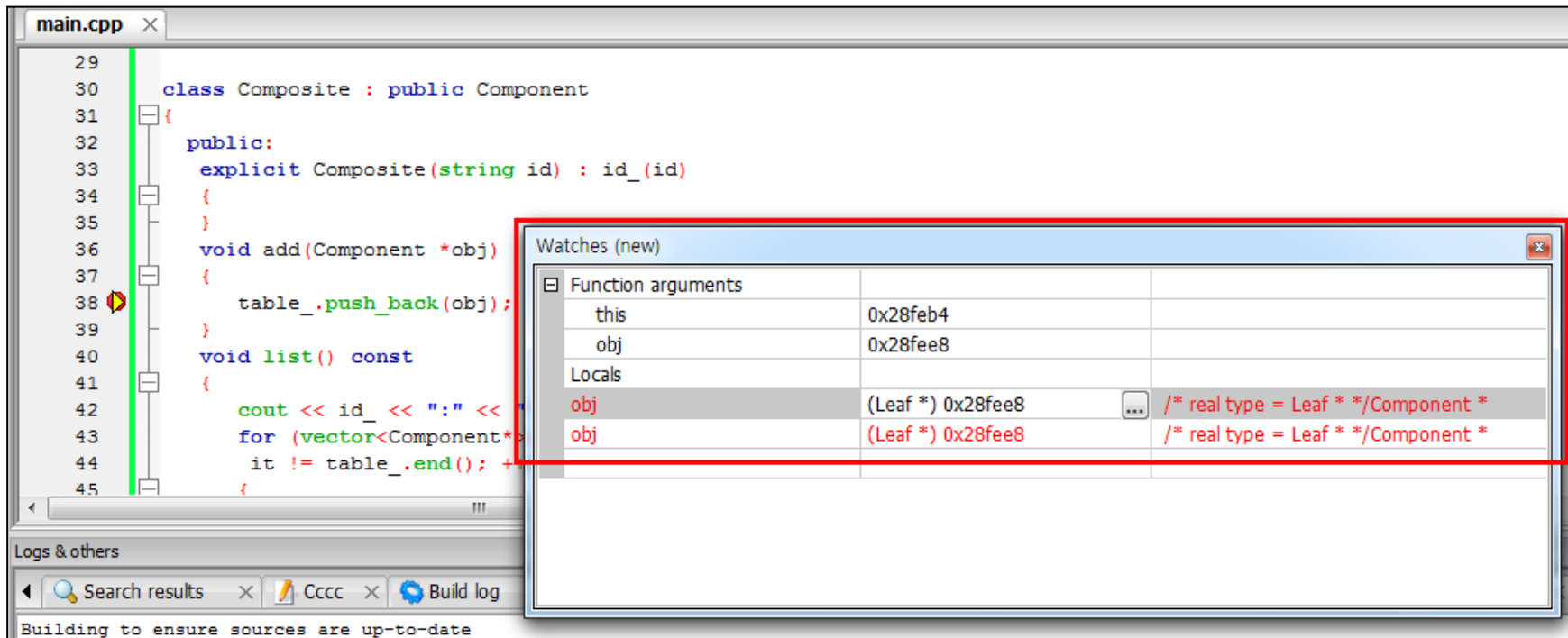
- obj의 값을 확인하기 위해 obj를 우클릭 해서 Watch 'obj'를 클릭한다.



4. 활용 예제

4.4 디버깅 (5/10)

- Watches (new) 창이 나타나며 obj의 상태를 확인할 수 있다.



The screenshot shows the Code::Blocks IDE with a C++ file named `main.cpp` open. The code defines a `Composite` class that inherits from `Component`. It has a public constructor `explicit Composite(string id) : id_(id)` and two methods: `add(Component *obj)` and `list() const`. The `add` method pushes `obj` onto a `table_` vector. The `list` method iterates over the `table_` vector and prints each element's ID. The cursor is positioned at line 38, where `table_.push_back(obj);` is executed. The `Watches (new)` window is open, showing the state of the program. It lists function arguments and local variables. The `obj` variable is shown with its memory address `0x28fee8` and its type `(Leaf *)`. The `real type` is also displayed as `(Leaf *)`.

```
29
30 class Composite : public Component
31 {
32 public:
33     explicit Composite(string id) : id_(id)
34     {
35     }
36     void add(Component *obj)
37     {
38         table_.push_back(obj);
39     }
40     void list() const
41     {
42         cout << id_ << ":" <<
43         for (vector<Component*>
44             it != table_.end(); +
45         {
```

Watches (new)

Function arguments		
this	0x28feb4	
obj	0x28fee8	
Locals		
obj	(Leaf *) 0x28fee8	/* real type = Leaf */Component *
obj	(Leaf *) 0x28fee8	/* real type = Leaf */Component *

Logs & others

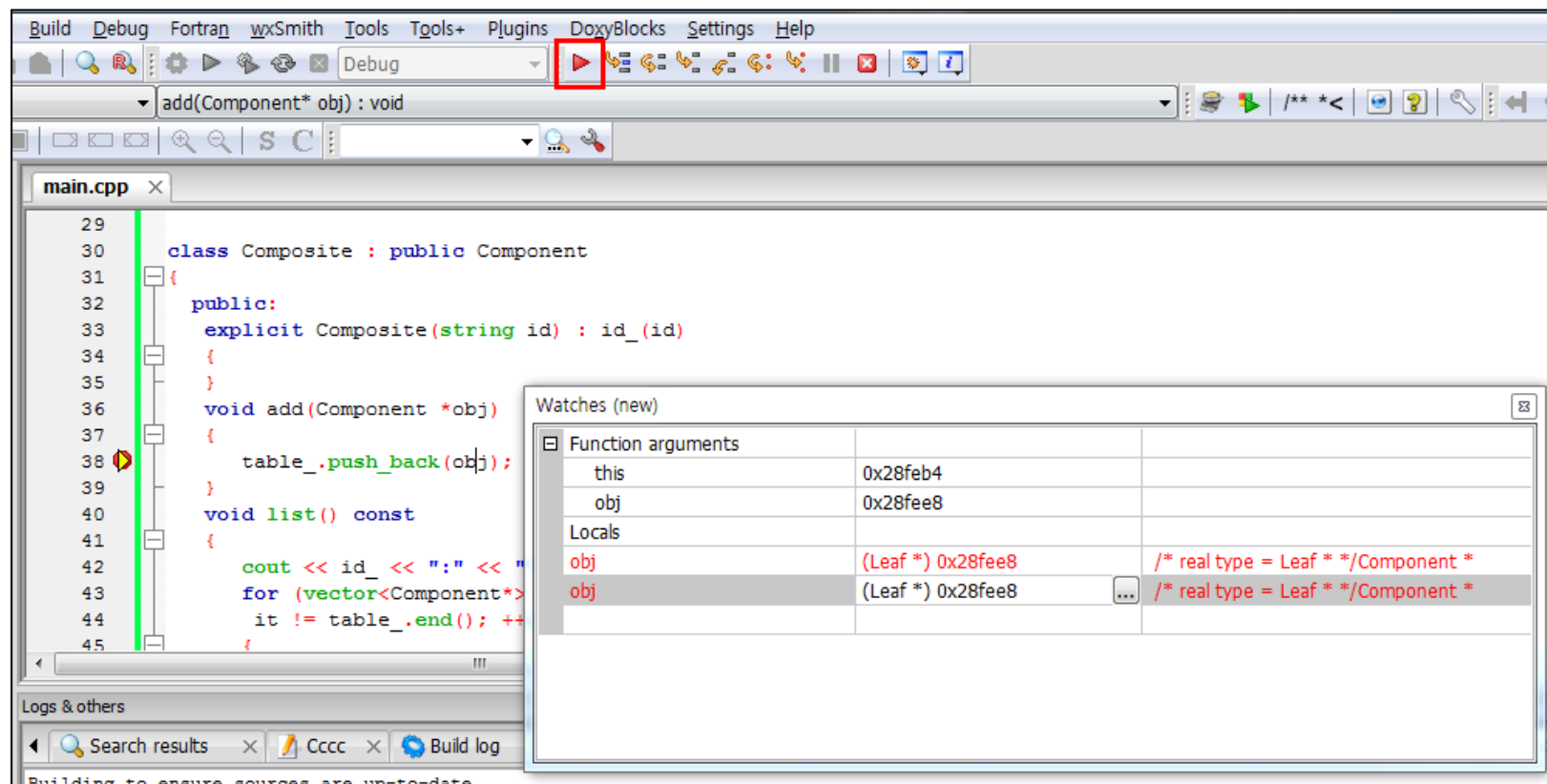
Search results × Cccc × Build log

Building to ensure sources are up-to-date

4. 활용 예제

4.4 디버깅 (6/10)

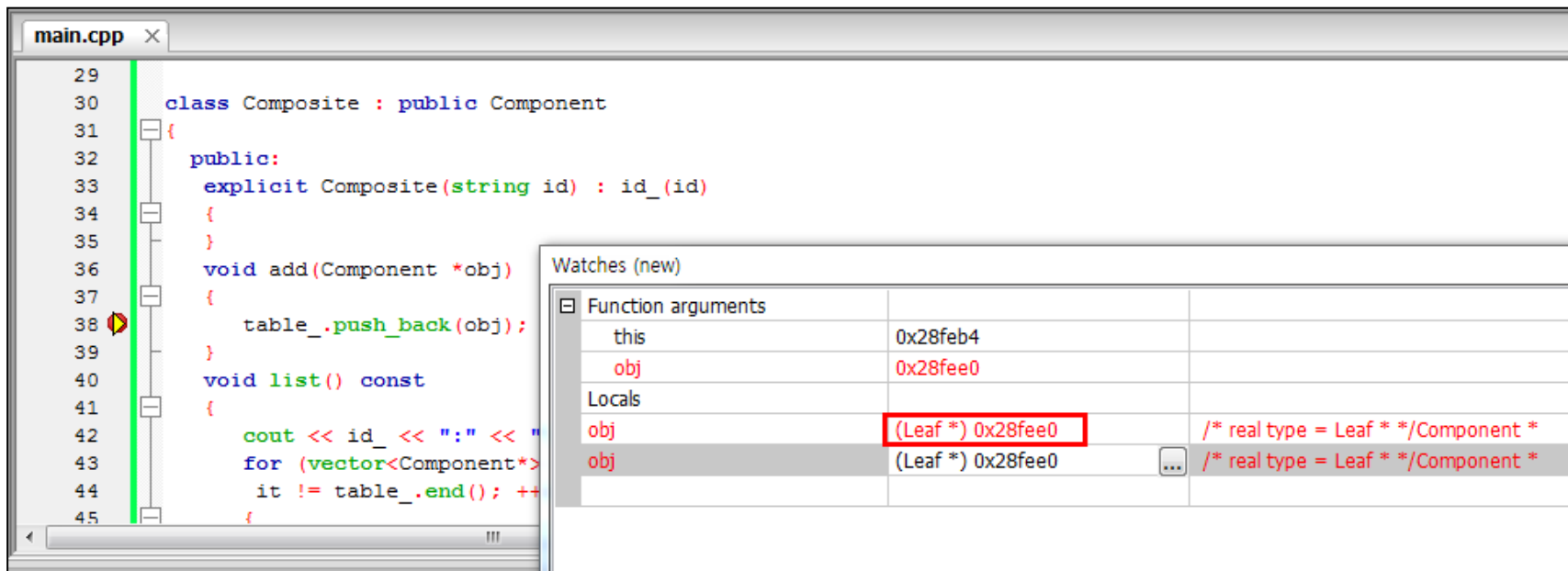
- 다시 빨간색 삼각형을 클릭하면 실행이 재개(resume)된다.



4. 활용 예제

4.4 디버깅 (7/10)

- Watches 창 내의 obj 값이 바뀌어 있다.



The screenshot shows the Code::Blocks IDE with a C++ program in `main.cpp` and the `Watches (new)` window open.

main.cpp

```
29
30 class Composite : public Component
31 {
32 public:
33     explicit Composite(string id) : id_(id)
34     {
35     }
36     void add(Component *obj)
37     {
38         table_.push_back(obj);
39     }
40     void list() const
41     {
42         cout << id_ << ":" << "
43         for (vector<Component*>
44             it != table_.end(); ++
45         {
```

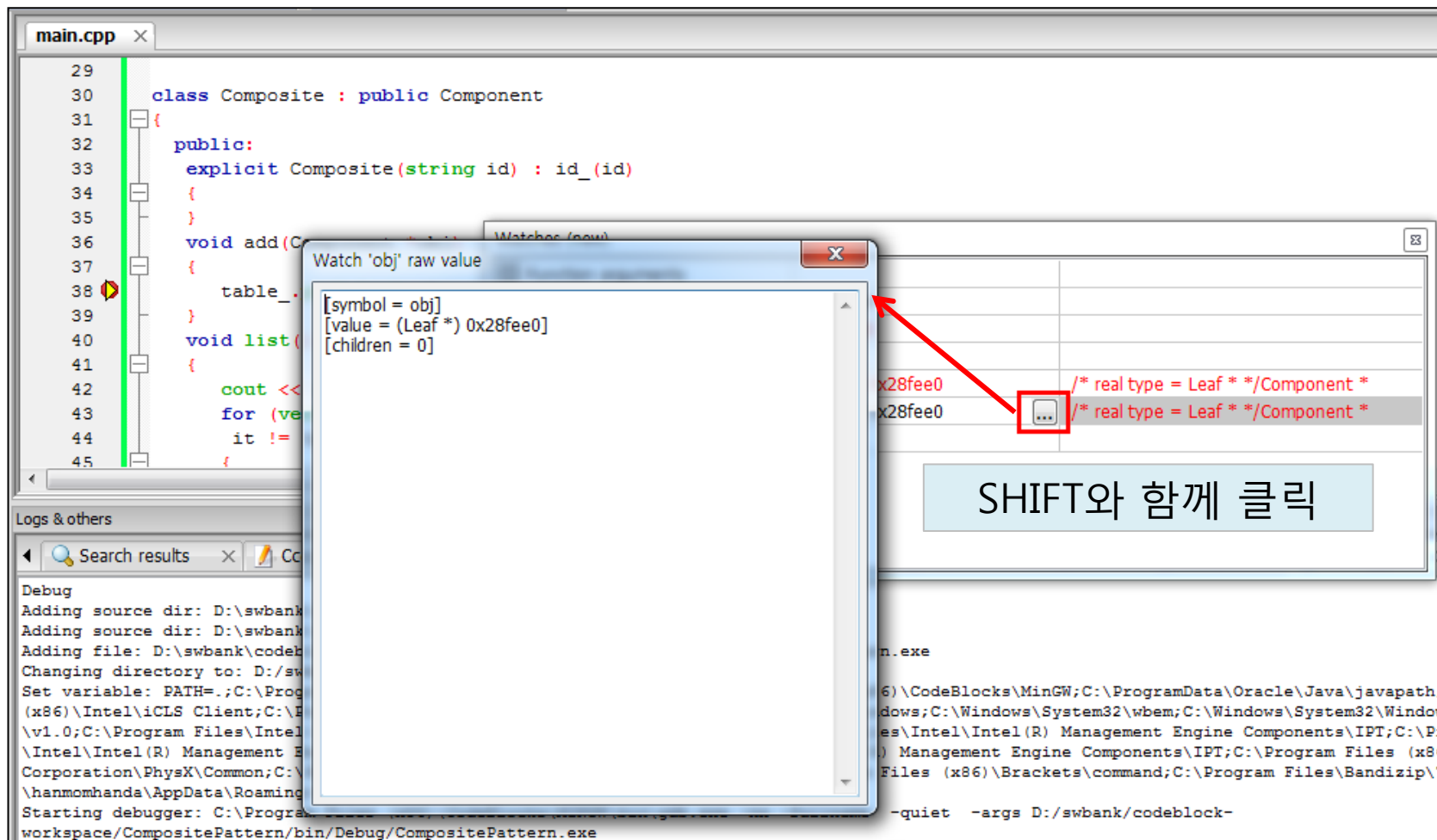
Watches (new)

Function arguments		
this	0x28feb4	
obj	0x28fee0	
Locals		
obj	(Leaf *) 0x28fee0	/* real type = Leaf */Component *
obj	(Leaf *) 0x28fee0	/* real type = Leaf */Component *

4. 활용 예제

4.4 디버깅 (8/10)

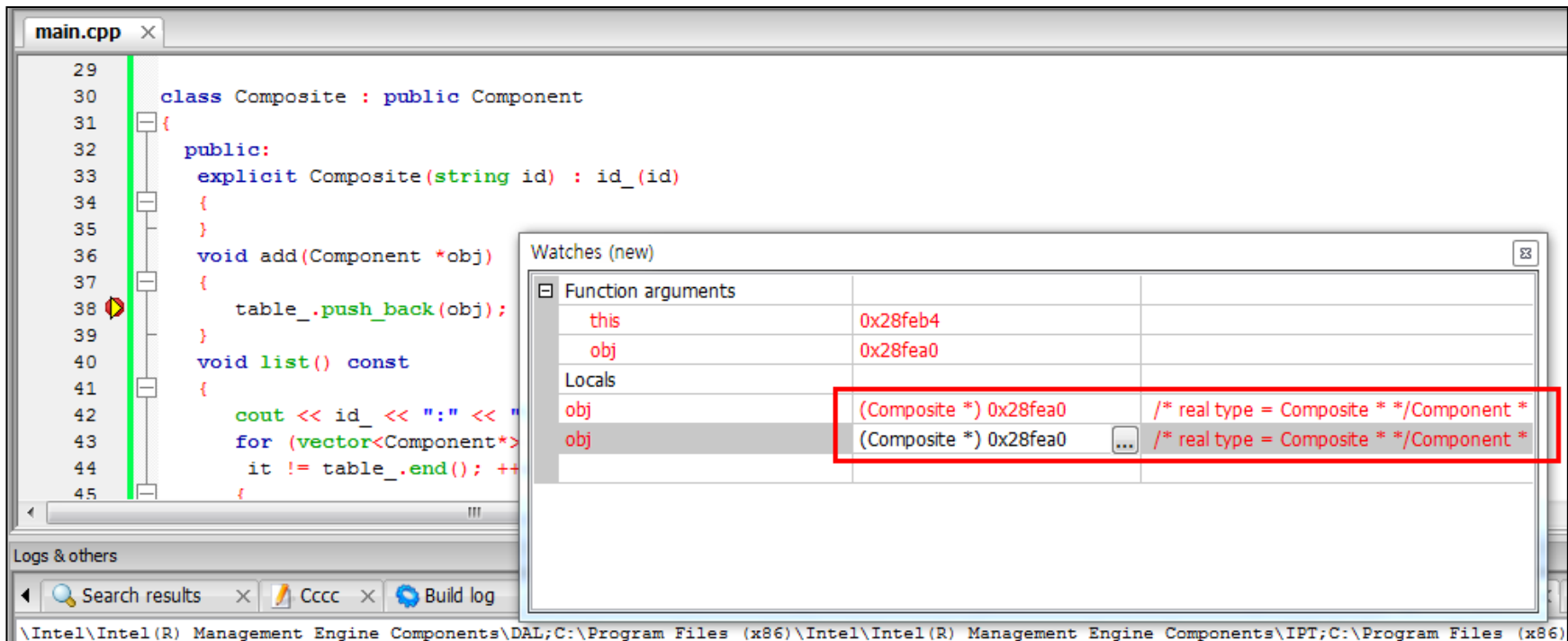
- Watches 창 내의 버튼을 SHIFT와 함께 누르면 상세한 상태를 볼 수 있다.



4. 활용 예제

4.4 디버깅 (9/10)

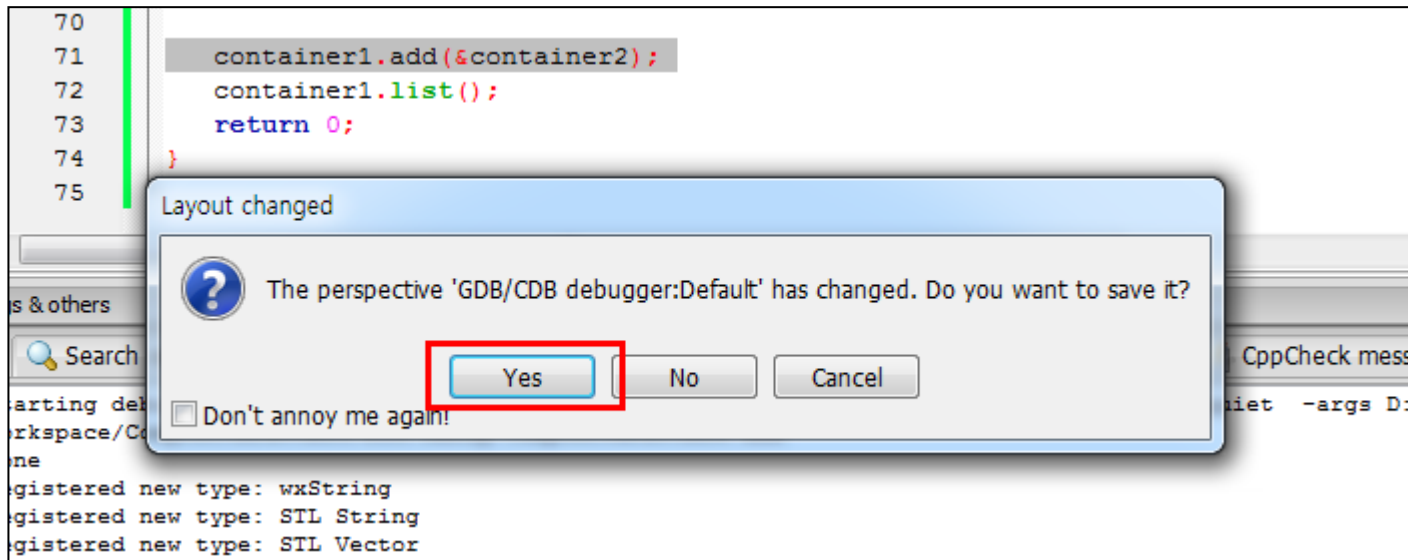
- 다시 빨간색 삼각형을 클릭하면 실행을 재개한다.
- 71행에서 container1에 container2를 추가했기 때문에, 계속 재개하다 보면 다음과 같이 Leaf가 아닌 Composite가 들어오는 것을 확인할 수 있다.



4. 활용 예제

4.4 디버깅 (10/10)

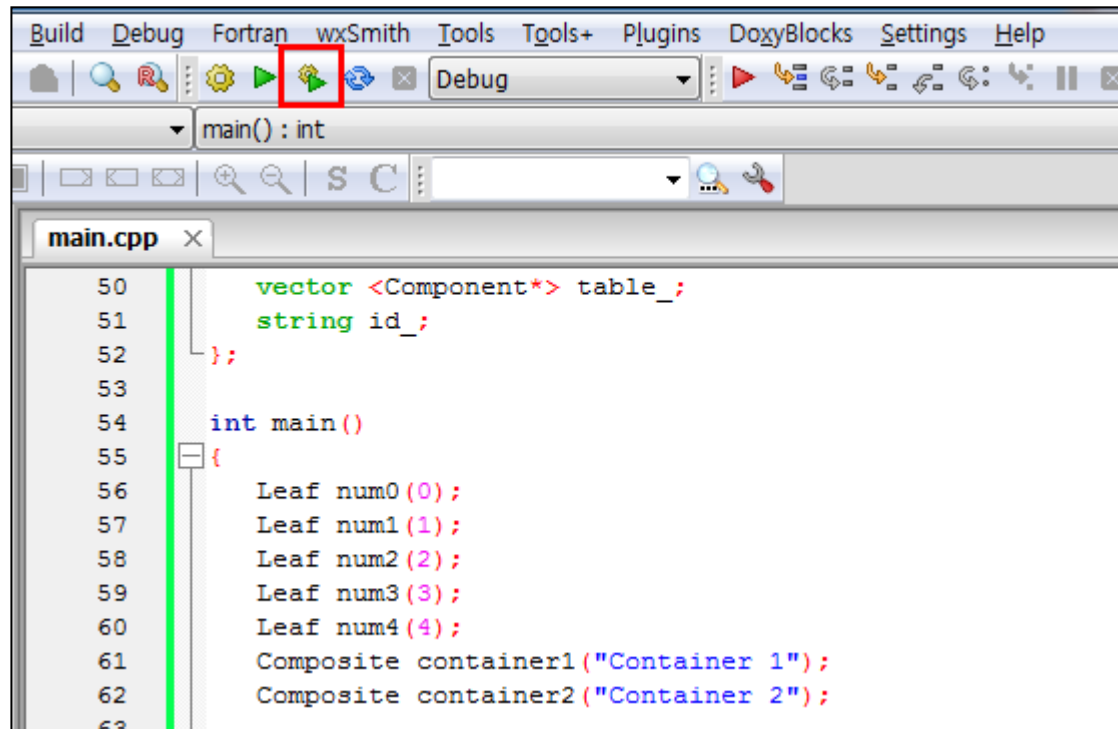
- 빨간색 삼각형을 한 번 더 클릭하면 프로그램 실행이 정상적으로 끝난다.
 - 아래와 같이 perspective를 저장하라는 알림창이 뜰 수 있는데, Yes를 눌러 저장한다.



4. 활용 예제

4.5 예제 실행 (1/2)

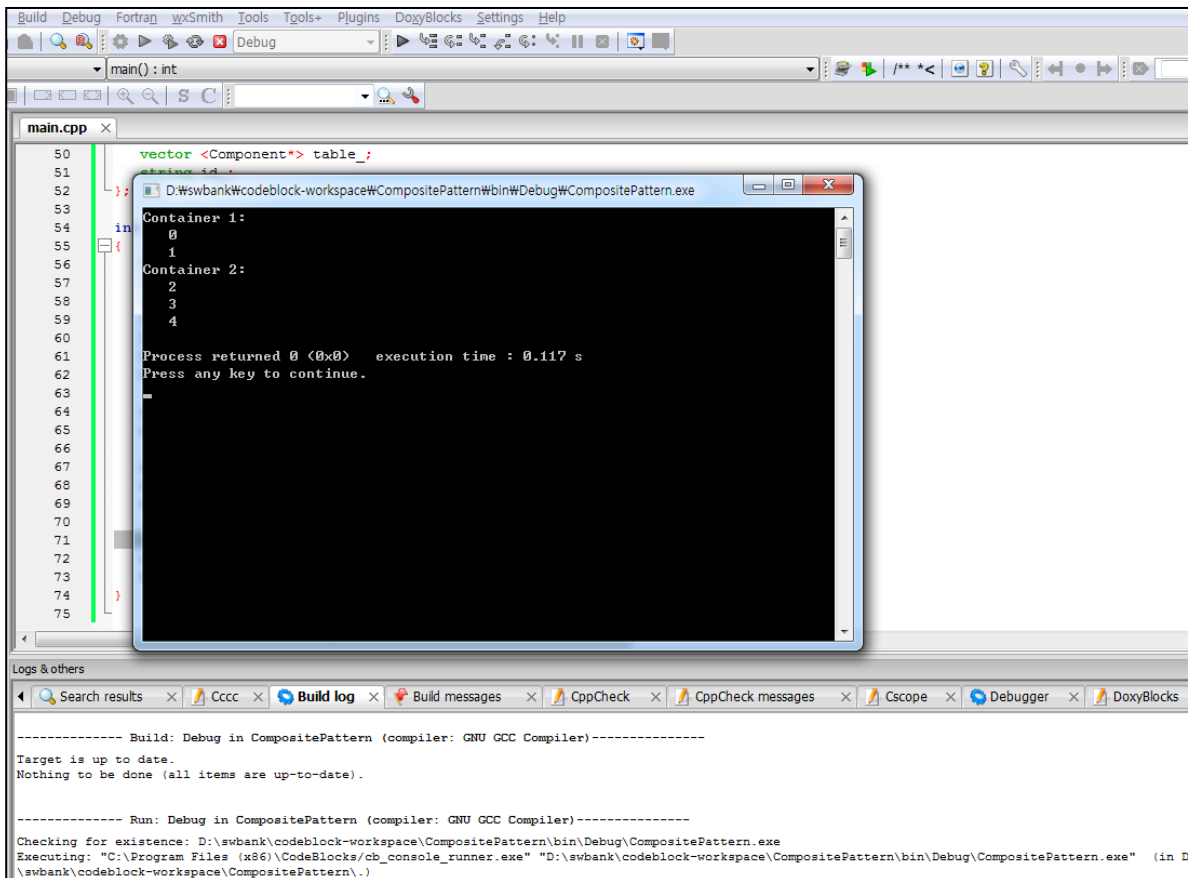
- 툴바에서 기어 버튼은 빌드를, 녹색 삼각형 버튼은 실행을 의미하며, 기어와 녹색 삼각형이 함께 있는 버튼은 빌드와 실행을 함께 수행한다.
- 빌드 및 실행 버튼을 클릭한다.



4. 활용 예제

4.5 예제 실행 (2/2)

- 툴바에서 기어 버튼은 빌드를, 녹색 삼각형 버튼은 실행을 의미하며, 기어와 녹색 삼각형이 함께 있는 버튼은 빌드와 실행을 함께 수행한다.
- 빌드 및 실행 버튼을 클릭하면 아래와 같이 콘솔 프로그램이 실행된다.



```
main.cpp
50 vector<Component*> table_;
51
52 };
53
54 in
55 {
56
57 Container 1:
58 0
59 1
60
61 Container 2:
62 2
63 3
64 4
65
66
67
68
69
70
71
72
73
74
75 }
```

```
Process returned 0 (0x0)   execution time : 0.117 s
Press any key to continue.
```

```
----- Build: Debug in CompositePattern (compiler: GNU GCC Compiler)-----
Target is up to date.
Nothing to be done (all items are up-to-date).

----- Run: Debug in CompositePattern (compiler: GNU GCC Compiler)-----
Checking for existence: D:\swbank\codeblock-workspace\CompositePattern\bin\Debug\CompositePattern.exe
Executing: "C:\Program Files (x86)\CodeBlocks\cb_console_runner.exe" "D:\swbank\codeblock-workspace\CompositePattern\bin\Debug\CompositePattern.exe" (in D:\swbank\codeblock-workspace\CompositePattern\.)
```