Chapter 01

C++ 프로그래밍의 첫 걸음

***학습목표**

- C++로 프로그래밍한다는 개념을 이해한다.
- C++를 이용해 간단한 프로그램을 작성하는 방법을 익힌다.
- C++ 프로그래밍의 기본 구조를 이해한다.

01. C++로 프로그래밍한다는 것의 의미

02. 세상에서 가장 간단한 C++ 프로그램

03. 간단한 출력을 하는 프로그램

요약

연습문제

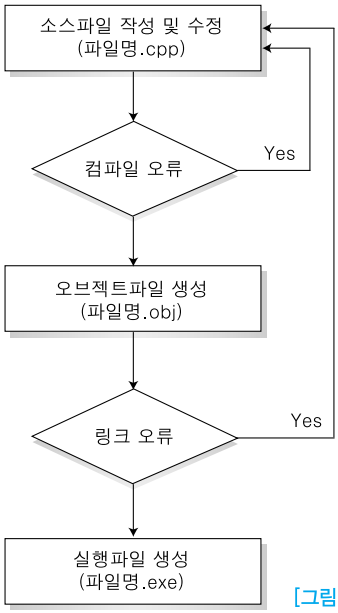
Section

1 C++로 프로그래밍한다는 것의 의미

C++로 프로그래밍한다는 것은 컴퓨터로 원하는 작업을 할 수 있도록 C++를 도구로 사용해 프로그램을 작성하는 것을 뜻한다. 컴퓨터는 스위치의 켜짐(On)이나 꺼짐(Off)에 의해서만 동작하고 이 두 가지 상태를 0과 1이라는 비트 형태로 저장하므로, 0과 1로 구성된 기계어 코드 상태만 이해하기 때문에 우리가 사용하는 언어를 이해하지 못한다.

C++는 사람의 언어를 컴퓨터가 이해할 수 있는 형태로 바꿔주기 위한 프로그램을 표현하는 언어다. 그러므로 C++를 이용해 프로그램을 작성하기 위해서는 먼저 C++ 문법 구조를 익혀야 하고, 그러기 위해서는 다양한 예제를 직접 작성해보는 것이 가장 좋다.

C++ 문법을 본격적으로 학습하기에 앞서 우선 C++ 프로그램을 작성하기 위해 어떤 과정을 거쳐야 하는지부터 살펴보자.



[그림 1-1] C++ 프로그래밍 단계

C++ 프로그램을 작성하는 과정은 크게 편집(Edit), 컴파일(Compile), 링크(Link), 실행(Execute) 4단계로 이루어진다. 그런데 각 단계가 항상 순조롭게 진행되는 것은 아니다. 컴파일이나 링크 단계에서 오류가 발생하면 다음 단계로 진행하지 못하므로 실행 가능한 파일을 얻기 위해서는 오류를 수정해야 한다. 각 단계에서 어떤 작업들이 일어나는지 자세히 살펴보자.

1. 편집 단계

프로그래머가 C++ 언어의 문법에 맞게 프로그램을 작성한 후 파일로 저장하는데 이렇게 작성된 파일을 소스파일이라 한다. 소스파일을 생성하는 편집 단계에서는 편집기가 필요하다. 다음 그림과 같이 윈도우에서 기본으로 제공하는 메모장을 이용할 수도 있지만 이 책에서는 비주얼 스튜디오에서 제공하는 편집기를 사용할 것이다. 어떤 편집기를 사용하든 C++ 소스파일을 작성한 후 저장할 때는 C++ 소스파일임을 의미하는 cpp 확장자를 꼭 붙여야 한다.

[그림 1-2] C++ 소스파일

2. 컴파일 단계

컴파일 단계에서는 편집 단계에서 작성한 소스파일이 C++ 컴파일러에 의해 C++ 문법에 맞는지를 먼저 검증한다. 그런 다음, 문법적 오류가 없음이 확인되면 소스파일을 기계어 상태로 변경해 오브젝트파일을 만들어 주는데, 오브젝트파일은 obj 확장자가 붙는다.

3. 링크 단계

프로그램에서 자주 사용되는 로직을 미리 정의해 제공하는 것을 라이브러리라고 한다. 한

개인이 프로그램에 대한 모든 내용을 기술하는 것은 불가능하므로 이러한 라이브러리에서 제공하는 정보(함수와 데이터 구조)를 사용하는데, 이들 라이브러리에서 제공하는 형식에 맞게 올바르게 사용하였는지를 링크 단계에서 검사한다. 그리고 올바르게 참조되었다면 컴파일 단계에서 만들어진 오브젝트파일을 실행이 가능한 파일로 만들어 주는데 실행파일에는 exe 확장자가 붙는다.

4. 실행 단계

링크 단계까지 성공적으로 진행해서 생성된 실행파일(확장자가 exe인 파일)은 컴퓨터에서 실행 가능하다. 이 실행파일을 콘솔창에서 입력하거나 비주얼 스튜디오에서 제공하는 툴을 이용해 실행시키는 단계가 바로 실행단계이다.

Section

02 세상에서 가장 간단한 C++ 프로그램

C++ 프로그램에는 기본적으로 따라야 하는 규칙이 있다. C++ 프로그램을 작성할 때 가장 먼저 주의할 점은 C++는 대문자와 소문자를 구분하므로 반드시 소문자로 기술해야 한다는 점이다. 이 절에서는 main 함수만 있는 가장 간단한 C++ 프로그램을 통해 C++ 프로그램의 구조를 살펴보고 기본적으로 따라야 하는 규칙을 알아본다.

다음 예제는 아무런 결과가 출력되지 않아서 프로그램이 수행되었는지 의심되지만 C++ 프로그램으로 전혀 손색이 없는 세상에서 가장 간단한 프로그램이다.

[예제 1-1] first.cpp

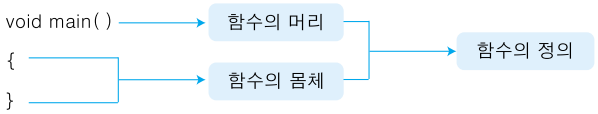
```
01  void main()
02  {
03  }
```

[예제 1-1]의 first.cpp는 C++ 프로그램에서 반드시 존재해야 하는 구성요소로만 구성되었기 때문에 이중 하나라도 빠지면 컴파일 에러가 발생한다. 이제 C++ 프로그램의 가장 기본이 되는 각 구성요소의 의미를 하나씩 살펴보자.

1. main() : 프로그램에서 반드시 하나 필요한 함수

C++ 프로그램에서는 기능별로 함수를 구현한 후 이 함수를 호출해서 사용한다. 컴파일러는 ()를 함수로 인식하도록 설계되어 있다. 위 예제에서는 main에만 ()가 있으므로 한 개의 main 함수로 구성된 예제이다. main은 C++ 컴파일러가 프로그램의 진입점으로 인식하는 함수이므로 C++로 작성한 프로그램은 반드시 가져야 하며 한 개만 존재해야 한다. C++ 프로그램을 실행시키면 main 함수를 찾아 함수 내부에 기술된 문장을 수행한다. 그런데 위 예제는 함수 내부에 내용이 없다.

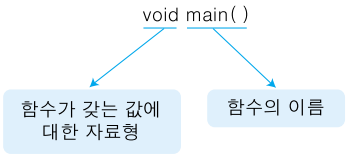
우리가 작성할 프로그램 역시 항상 main 함수가 등장하는데 이와 같은 형태로 main 함수를 기술한 것을 함수의 정의라고 한다. 함수의 정의는 함수의 머리와 몸체로 구성된다.



[그림 1-3] 함수의 구조

2. void : 함수가 값을 갖지 않도록 하는 자료형

함수의 머리 부분은 다음 예와 같이 자료형과 함수명으로 구성된다.

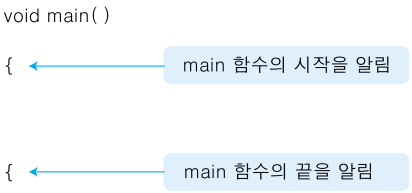


[그림 1-4] 함수의 머리 구조

모든 함수는 반드시 자료형을 지정해야 하는데 자료형은 함수 앞에 붙인다. 앞의 예에서는 main 함수 앞에 void 자료형을 지정하였는데, 이 void는 함수가 값을 갖지 않을 경우 붙이는 자료형이다.

3. {} : 함수의 시작과 끝

C++ 컴파일러는 함수의 시작과 끝을 알아야 하므로 이를 위해 {와} 기호를 사용한다.



[그림 1-5] 함수의 시작과 끝

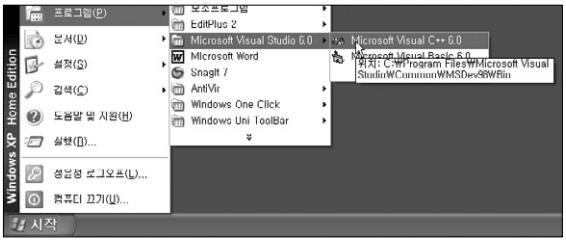
여기서는 main 함수의 시작과 끝을 알려주기 위해서 { 와 }가 사용되었다. C++로 작성한 프로그램을 실행시키면 main 함수의 { 와 } 사이에 기술한 내용이 차례대로 실행된다. 컴퓨터에게 시키고자 하는 일을 { 와 }사이에 기술하는데 이 기술 과정이 바로 프로그램을 작성하는 과정이다.

위에서 살펴본 소스파일 예제를 컴파일해 실행 가능한 파일을 생성하려면 C++ 컴파일러가 필요하다. 이 책에서는 비주얼 스튜디오 6.0을 사용하는데 비주얼 스튜디오 설치는 어렵지 않으므로 생략하고 이를 사용해 C++ 프로그램을 작성하는 방법만 살펴본다.

[실습하기 1-1] 비주얼 스튜디오로 C++ 프로그래밍하기

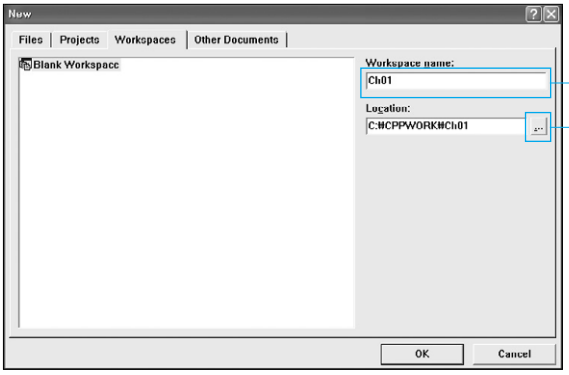
1 〈시작〉 버튼을 클릭한 후 [프로그램]-[Microsoft Visual Studio 6.0]-[Microsoft Visual C++ 6.0] 메뉴를 차례로 클릭해서 Visual C++ 프로그램을 실행한다.

http://www.hanbitbook.co.kr/exam/1467에서 비주얼 스튜디오 6.0 설치 방법을 설명한 문서를 다운로드 할 수 있다.



〈〈 Visual C++ 프로그램 실행

2 [File]-[New] 메뉴를 선택해 나오는 화면에서 [Workspaces] 탭을 클릭한 후 Workspace name: 입력란에 'Ch01'을 입력한다.



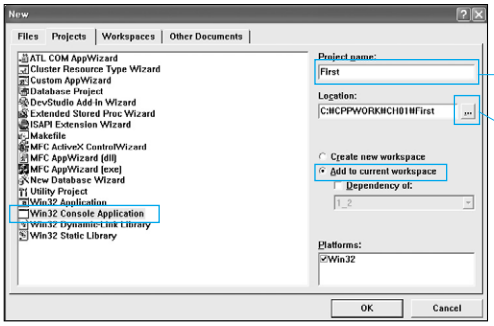
워크스페이스 이름을 직접 입력한다.

[…] 버튼을 클릭한 후 작업 폴더로 이동한다.
이 책에서는 C:₩CPPWORK를 작업 폴더로 이용한다.

〈〈 워크스페이스 생성

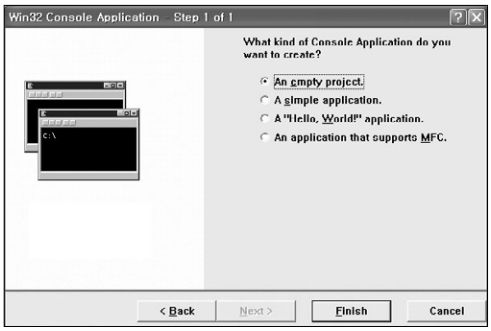
하나의 프로젝트에는 여러 개의 C++ 프로그램 파일을 포함할 수 있고, 다시 이 프로젝트를 여러 개 모아 워크스페이스에 포함시킨다. 이 책에서는 장별로 워크스페이스를 만들어 같은 장에서 다루는 모든 예제를 프로젝트 단위로 만들어 저장하고 있는데 프로젝트와 워크스페이스가 무엇인지는 다음 실습을 통해 알아볼 것이다.

3 같은 화면에서 [Projects] 탭을 클릭해 Win32 Console Application 항목을 선택한 후 경로(Location)와 프로젝트 이름(Project name)을 기입하고 〈OK〉 버튼을 누른다. 이때 이미 생성된 워크스페이스에 프로젝트를 포함시키려면 Add to current workspace를 선택해야 한다. 여기서 경로는 'C:\CPPWORK\Ch01'로 지정하고 프로젝트 이름은 'First'로 한다.



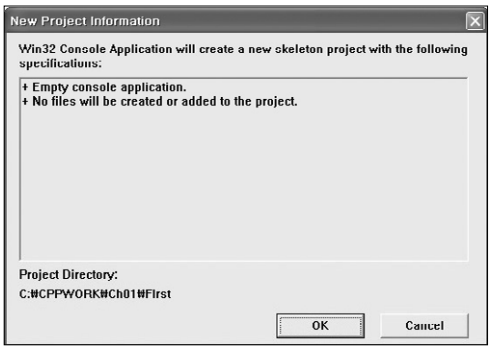
〈〈 프로젝트 생성

4 Console Application의 종류를 선택하는 화면에서 An empty project 항목을 선택한 후 〈Finish〉 버튼을 누른다.



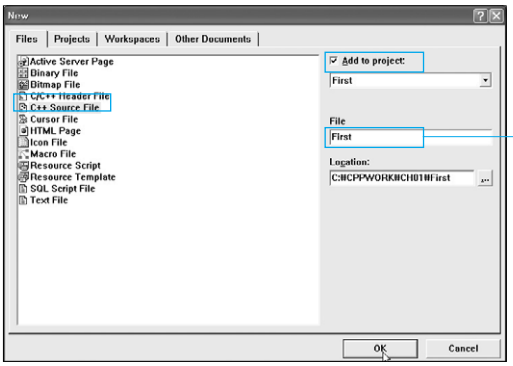
〈〈 Console Application의 종류 선택

5 새 프로젝트에 대한 정보를 확인한 후 〈OK〉 버튼을 누른다.



〈〈 새 프로젝트 정보 확인

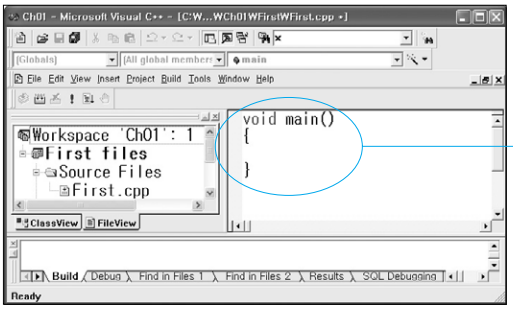
6 [File]-[New] 메뉴를 선택한 후 이번에는 [Files] 탭 클릭한 후 C++ Source File 항목을 선택하고 Add to Project 항목을 체크한 후 File란에 C++ 파일명을 입력한다. 여기서는 'First'라고 입력한다.



C++ 파일명을 직접 입력한다.

《 프로젝트에 C++ 소스파일

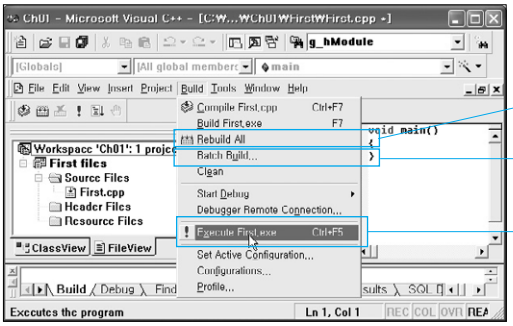
7 우측 편집창에 아래 그림과 같이 작성해보자.



소스 코드 입력

《 소스 코드 입력

8 소스 프로그램을 작성하였으면 컴파일, 링크한 후 실행해야 한다. 만일 여기서 컴파일, 링크 과정을 생략하고 바로 [Build]-[Execute First.exe] 메뉴를 선택하면 컴파일부터 실행까지 모두 이루어진다.



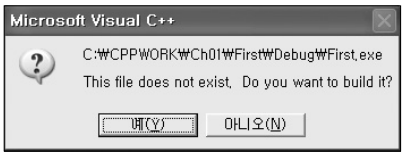
컴파일해 오브젝트파일 생성

링크해 실행파일을 생성

실행파일을 실행시켜서 결과를 확인

《 컴파일, 링크, 실행

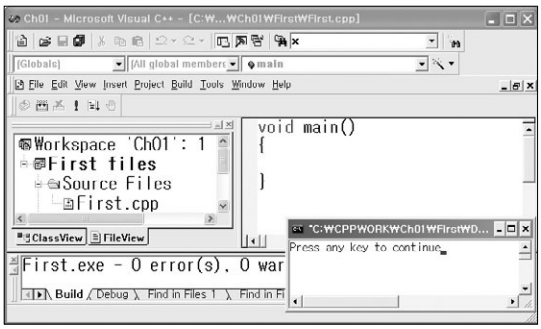
9 실행파일이 생성되지 않은 상태에서 [Build]–[Execute First.exe] 메뉴를 실행하면 실행파일을 생성하겠느냐고 묻는 화면이 나온다. 〈예〉 버튼을 클릭하면 컴파일부터 실행까지 모두 이루어진다.



〈〈 실행파일 생성 확인

컴파일 결과에 에러가 발생했을 시에는 이 에러를 수정하고 ❾와 같은 방법으로 다시 컴파일하고 실행해야 한다.

10 컴파일 및 링크 과정에서 에러가 발생하지 않으면 화면 하단에 'First.exe – 0 error(s), 0 warning(s)'을 출력한다. 실행 결과는 별도로 콘솔창이 떠서 거기에 출력된다.



〈〈 컴파일, 실행하여 결과 출력

저자 한마디

파일 열기 및 저장하기

프로그램을 새로 작성한 후 저장할 때 [파일]–[Save Workspace] 메뉴를 이용하면 워크스페이스 단위로 저장되므로 현재 활성 중인 파일을 저장할 때는 [Save]를 이용한다. 파일을 열 때는 [파일]–[Open Workspace] 메뉴를 이용해 워크스페이스 단위로 열어서 사용하며, 파일 단위로 열 때는 [Open] 메뉴를 이용한다. 파일을 닫을 때는 [파일]–[Close Workspace] 메뉴를 이용해 워크스페이스 단위로 닫으며 열어서 파일 단위로 닫고자 할 때는 [Close] 메뉴를 이용한다.