

# Getting Started with Visual Studio Code

1. VSCode 간단한 프로그램 테스트
2. VSCode 디버깅

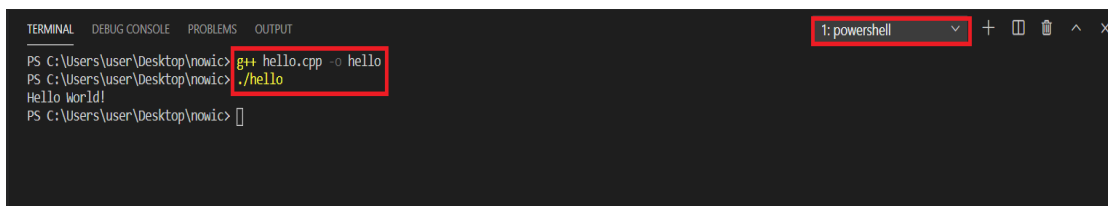
## 1. VSCode 간단한 프로그램 테스트

- 테스트를 하기 전에 반드시 VSCode Step2에서의 **Code Runner Extension이 설치되어 있어야 합니다.** 프로그램을 테스트하는 데에는 다양한 방법들이 있습니다. "hello.cpp" 파일을 이용하여 VSCode에서 C++에 정상적으로 작동하는지를 테스트해보아야 합니다.

- VSCode에서 디버깅을 사용하면서 만약 사용자의 이름이 한글로 되어 있다면 경로를 찾지 못하여 **"gdb failed with message: no such file or directory"** 오류가 발생하게 됩니다. 가장 좋은 방법은 사용자의 이름을 영문으로 바꾸어 다시 프로그램을 실행해보는 것이지만 <https://injunech.tistory.com/308>에 접속하여 해당 사이트의 방법을 따라 오류를 해결할 수도 있습니다.

- VSCode 터미널(Terminal)을 이용한 프로그램 실행


1. **Ctrl + ~** 단축키를 눌러 VSCode의 터미널을 실행시킵니다.
2. 터미널 **shell의 디폴트 Path와 유형을 확인해줍니다.** 이때 Path는 자신의 작업 폴더 Path이며, 유형은 PowerShell이어야 합니다.
3. GCC(g++) 프로그램 테스트 방법과 같이 VSCode의 터미널을 이용하여 빌드와 프로그램을 실행시켜줍니다. (VSCode Step1. GCC 및 MSYS2 설치 참조)

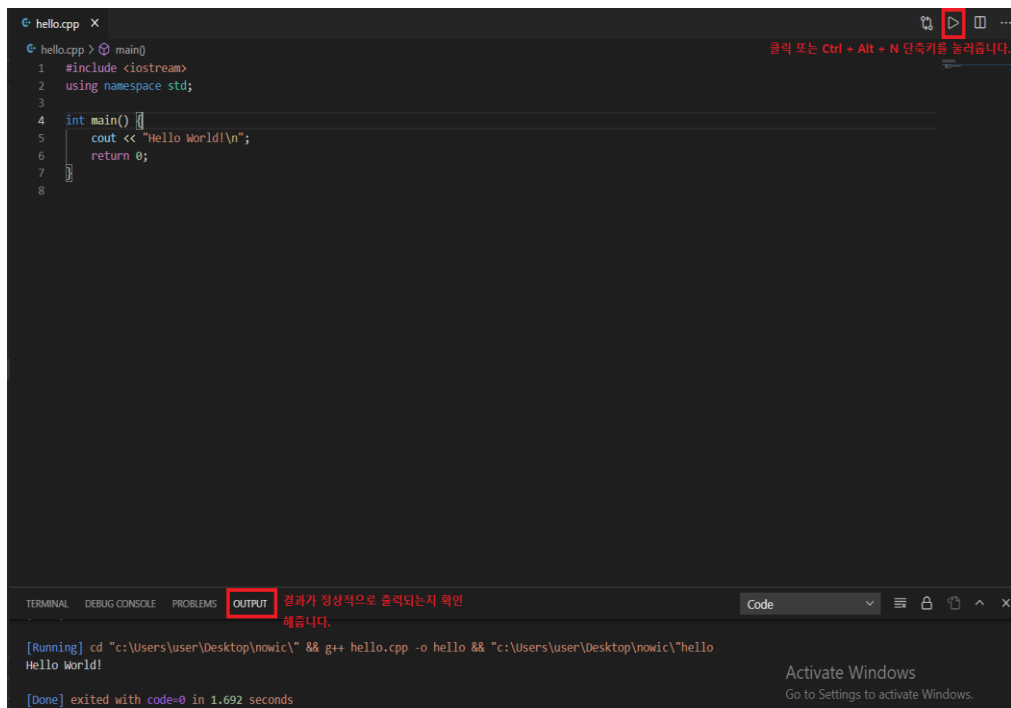


The screenshot shows a VS Code terminal window with the title bar '1: powershell'. The terminal content is as follows:

```
PS C:\Users\user\Desktop\nowic> g++ hello.cpp -o hello
PS C:\Users\user\Desktop\nowic> ./hello
Hello World!
PS C:\Users\user\Desktop\nowic> 
```

## ■ Code Runner Extension을 이용한 프로그램 실행

1. VSCode 우측 상단의 버튼  을 클릭해줍니다. 또는, **Ctrl + Alt + N** 단축키를 눌러 프로그램을 실행할 수도 있습니다.
2. [OUTPUT]에서 결과가 출력되는지 확인합니다. **Code Runner Extension을 이용하여 프로그램을 실행했을 때에는 출력만이 가능하며, arguments 입력은 불가능합니다.** 그러므로 간단히 출력만을 하는 프로그램을 실행할 때만 사용하여야 합니다.



The screenshot shows the VS Code interface with a C++ file named `hello.cpp` open. The code is as follows:

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     cout << "Hello World!\n";
6     return 0;
7 }
```

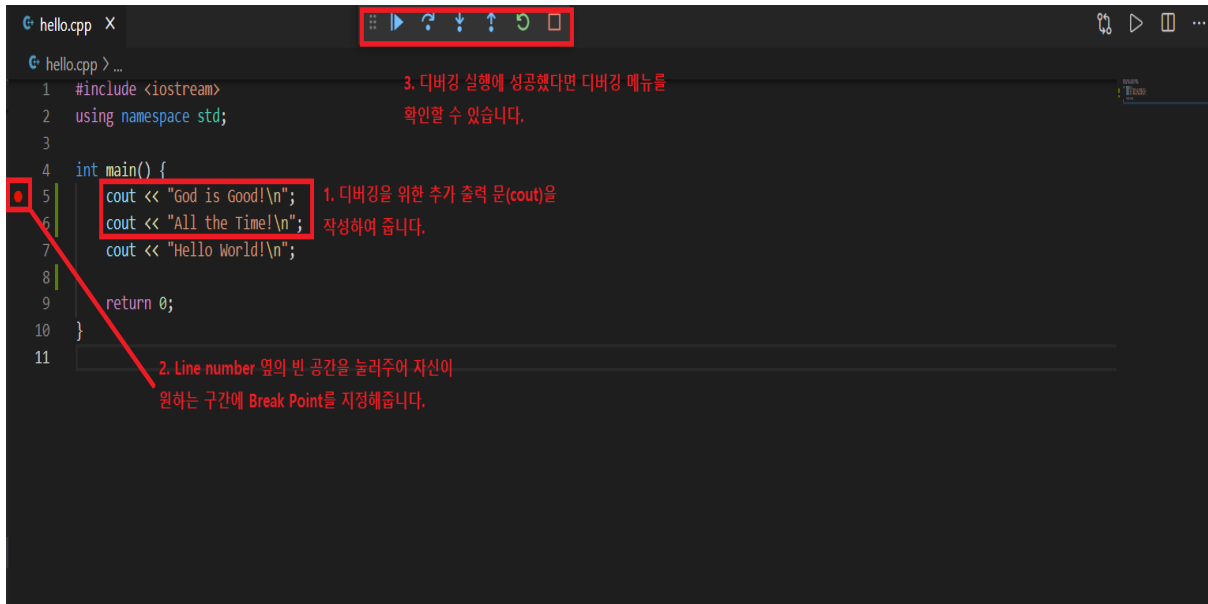
The Run button (a play icon) in the top right corner of the editor is highlighted with a red box. A tooltip above it says "클릭 또는 Ctrl + Alt + N 단축키를 눌러줍니다." (Click or press the Ctrl + Alt + N shortcut key).

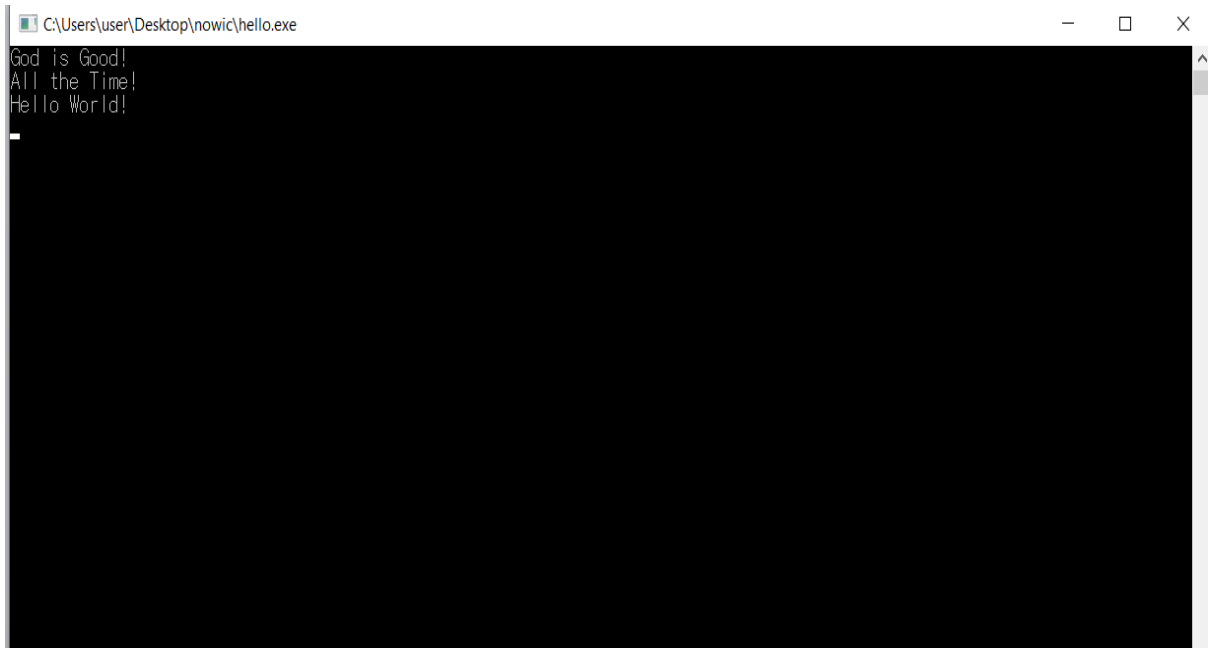
At the bottom, the OUTPUT panel is active and shows the following text:

```
[Running] cd "c:\Users\user\Desktop\nowic\" && g++ hello.cpp -o hello && "c:\Users\user\Desktop\nowic\hello
Hello World!
[Done] exited with code=0 in 1.692 seconds
```

## 2. 디버깅(Debugging) 실행

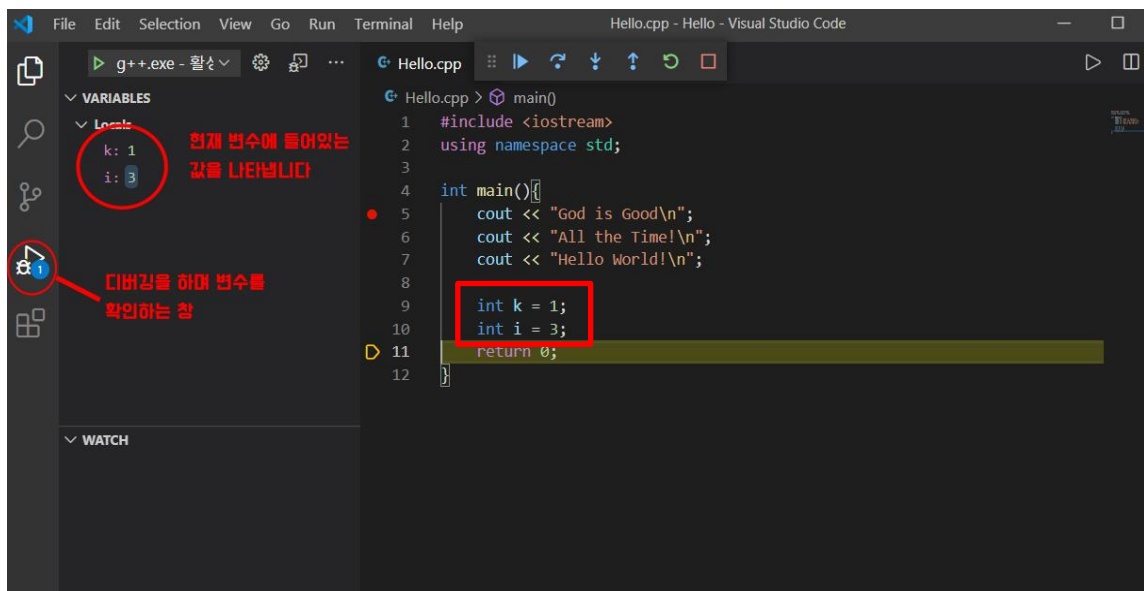
1. "hello.cpp" 소스 코드에서 디버깅을 위한 출력 문(cout)을 추가해줍니다. 기존에 존재하던 "Hello World!" 출력 문을 참고하여 자신이 추가하고 싶은 출력 문을 추가해줍니다. 이때에 "Wn"은 new line을 뜻하는 이스케이프 시퀀스로 줄 바꿈을 실행해줍니다.
2. 소스 코드에서 Line number의 옆 빈 공간을 클릭해주어 "Breakpoint"를 지정해줍니다. Breakpoint는 프로그램이 실행할 때에 멈추고 싶은 부분을 지정해줍니다. Breakpoint가 지정된 곳은 사용자의 추가적인 명령 전까지 일시적으로 흐름을 멈춥니다. (단축키: F9 [Run -> Toggle Breakpoint])
3. VSCode 상단의 옵션에서 [디버깅] 또는 [실행]을 눌러 [디버깅 시작]을 선택해줍니다. 또는, F5 단축키를 눌러주어 디버깅을 실행해줍니다. 디버깅이 성공적으로 실행했다면 새로운 콘솔 창이 나타남을 확인할 수 있습니다. ("externalConsole"의 값을 "true"로 지정했을 시에만 나타납니다.)
4. F10 단축키 (프로시저 단위 실행)(한 line씩 실행)를 계속해서 눌러주면서 콘솔 창에서 출력문이 할 줄씩 출력되는 것을 확인해줍니다.





(디버깅 실행 시에 나타나는 콘솔 창)

5. 디버깅의 가장 큰 장점은 프로그램의 흐름(프로시저)에 따라 현재 변수의 값을 실시간으로 확인할 수 있는 것입니다. "hello.cpp" 소스 코드에서 변수를 추가해준 후에 디버깅을 다시 실행해줍니다. 변수를 확인할 수 있는 창에서 현재 변수의 값이 나타나는지 확인해줍니다. 반복문(for 또는 while)을 추가하여 프로그램의 흐름에 따라 반복문에 선언된 변수의 값이 계속해서 변화하는 것을 확인하는 것도 좋습니다.



- 디버깅의 옵션은 아래와 같습니다.



1. **계속(Continue) – F5 [Run -> Start debugging]:** 다음 Breakpoint로 이동해줍니다. 두 개 이상의 Breakpoint가 있어야 합니다.
2. **프로시저 단위 실행(Step Over) – F10 [Run -> Step over]:** Breakpoint의 라인에서 다음 라인으로 이동해줍니다. 다음 라인이 함수일 경우는 해당 함수 내부로는 이동하지 않고 바로 실행하게 됩니다. 디버깅 옵션 중에 가장 많이 사용되는 옵션입니다.
3. **한 단계씩 코드 실행(Step Into) – F11 [Run -> Step into]:** 프로시저 단위 실행(Step Over)과 동일하게 다음 라인으로 이동합니다. 하지만, 다음 라인이 함수일 경우에는 해당 함수 안으로 들어가 함수 내부의 동작을 한 줄씩 이동합니다.
4. **프로시저 나가기(Step Out) – Shift + F11 [Run -> Step out]:** 현재 디버깅 중인 함수의 나머지 부분을 한 번에 실행시키고 함수의 리턴이 완료된 부분에서 멈춥니다. 한 단계씩 코드 실행(Step Into)을 통해 함수 안으로 들어간 뒤에 바로 리턴 문으로 넘어가고 싶을 때에 사용하면 유용합니다.
5. **재실행(Restart) – Ctrl + Shift + F5 [Run -> Restart debugging]:** 디버깅을 재실행시킵니다. 주로 디버깅 도중에 소스 코드를 변경하거나 Breakpoint를 변경해주었을 때에 디버깅을 새로 시작할 때에 사용됩니다.
6. **중지(Stop) – Shift + F5 [Run -> Stop debugging]:** 디버깅을 중지합니다.

- 디버깅의 옵션 중에 “Run without debugging”은 아직 C/C++ Extension에 적용되지 않아 실행되지 않습니다. 디버깅이 없이 프로그램을 실행하기 위해서는 Breakpoint를 모두 제거한 후에 실행하셔야 합니다. Breakpoint는 다시 한 번 눌러주면 제거됩니다.

- Visual Studio Code에 대한 설명, 단축키 설정, 그리고 디버깅 설정에 더 많은 관심이 있다면 아래의 URL 확인

<<https://goodgodgd.github.io/ian-flow/archivers/vscode-tutorial>>