

Micropython

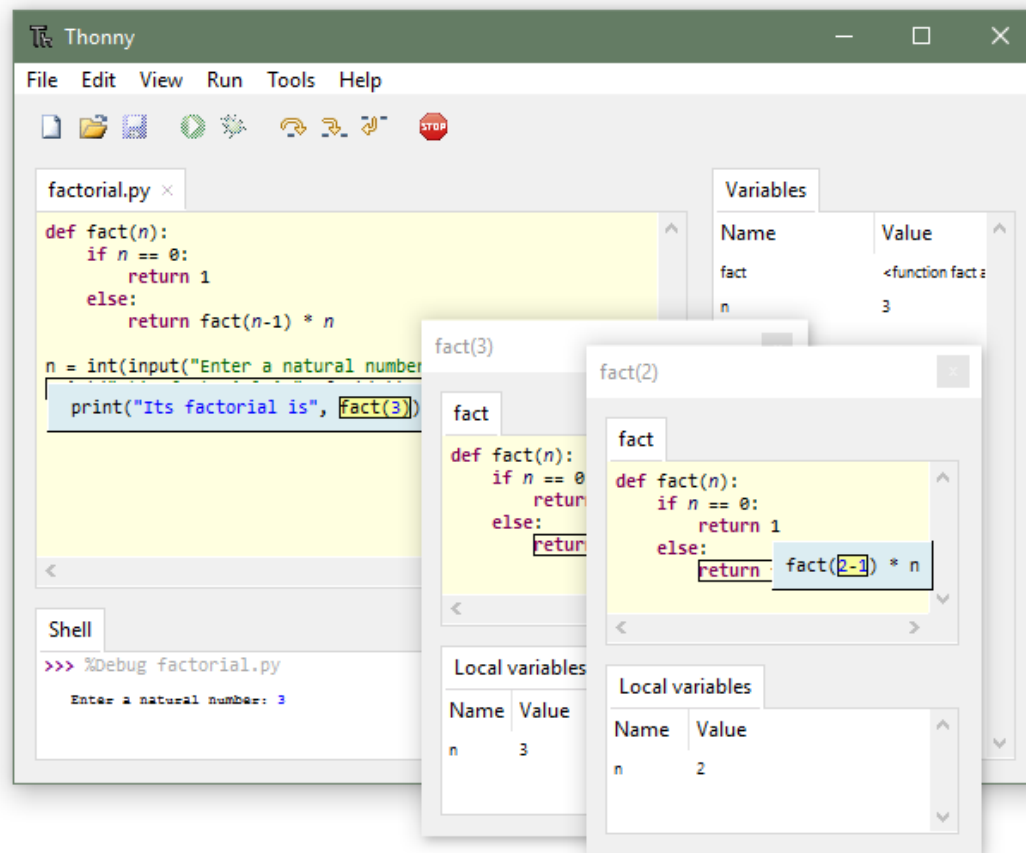
Thonny 설치

- <https://thonny.org/>

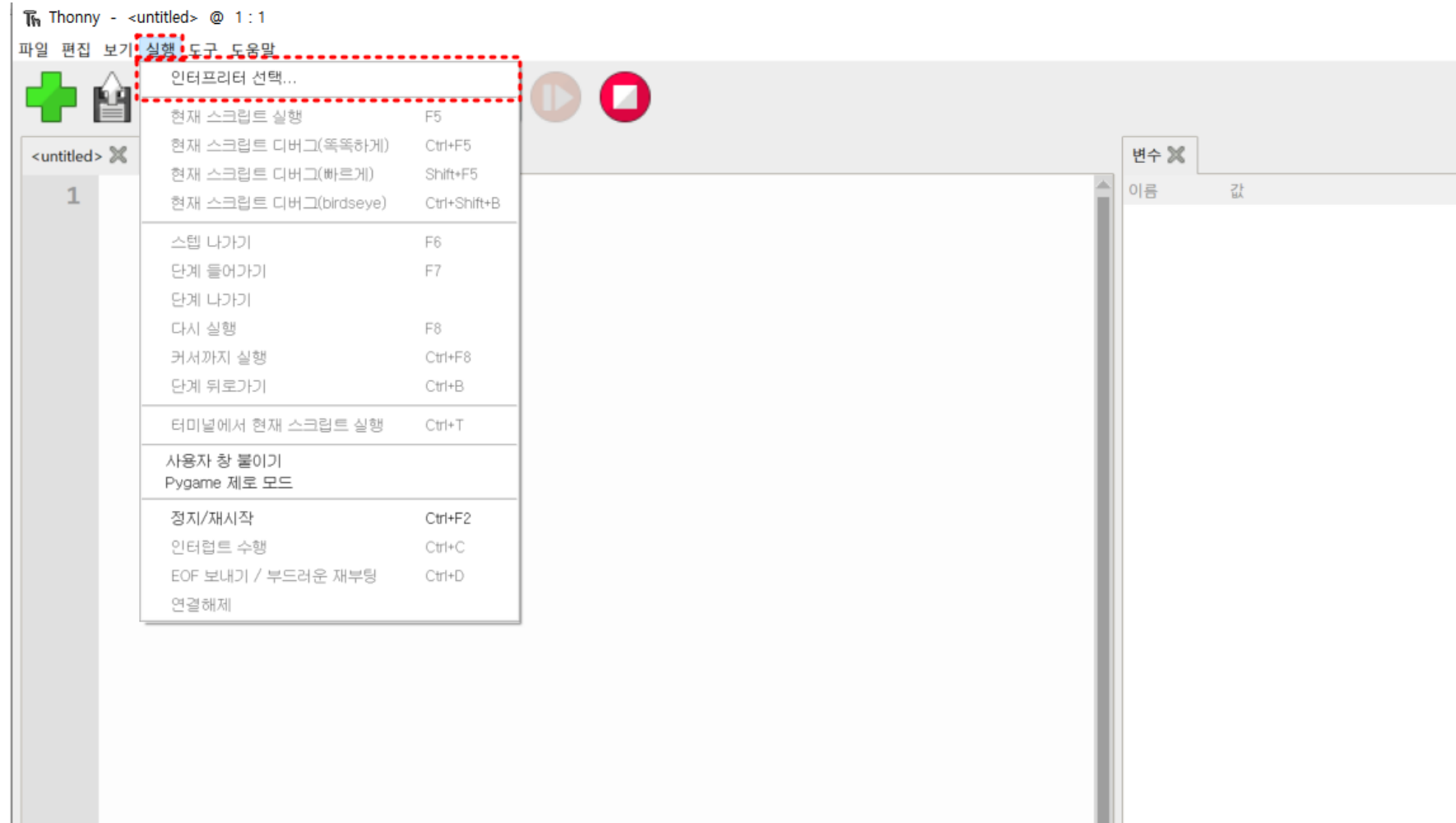
Thonny
Python IDE for beginners



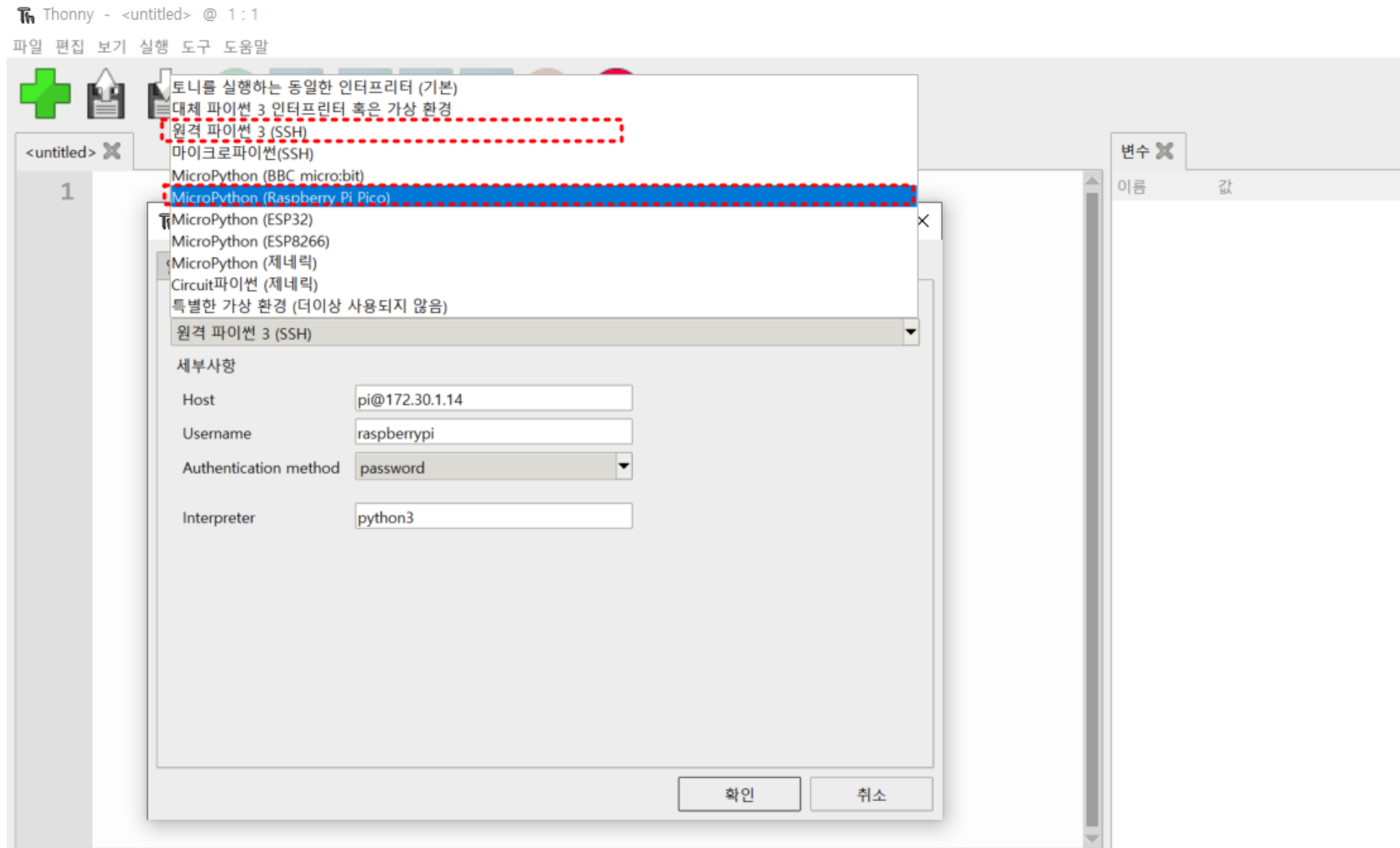
Download version **3.3.13** for
[Windows](#) • [Mac](#) • [Linux](#)



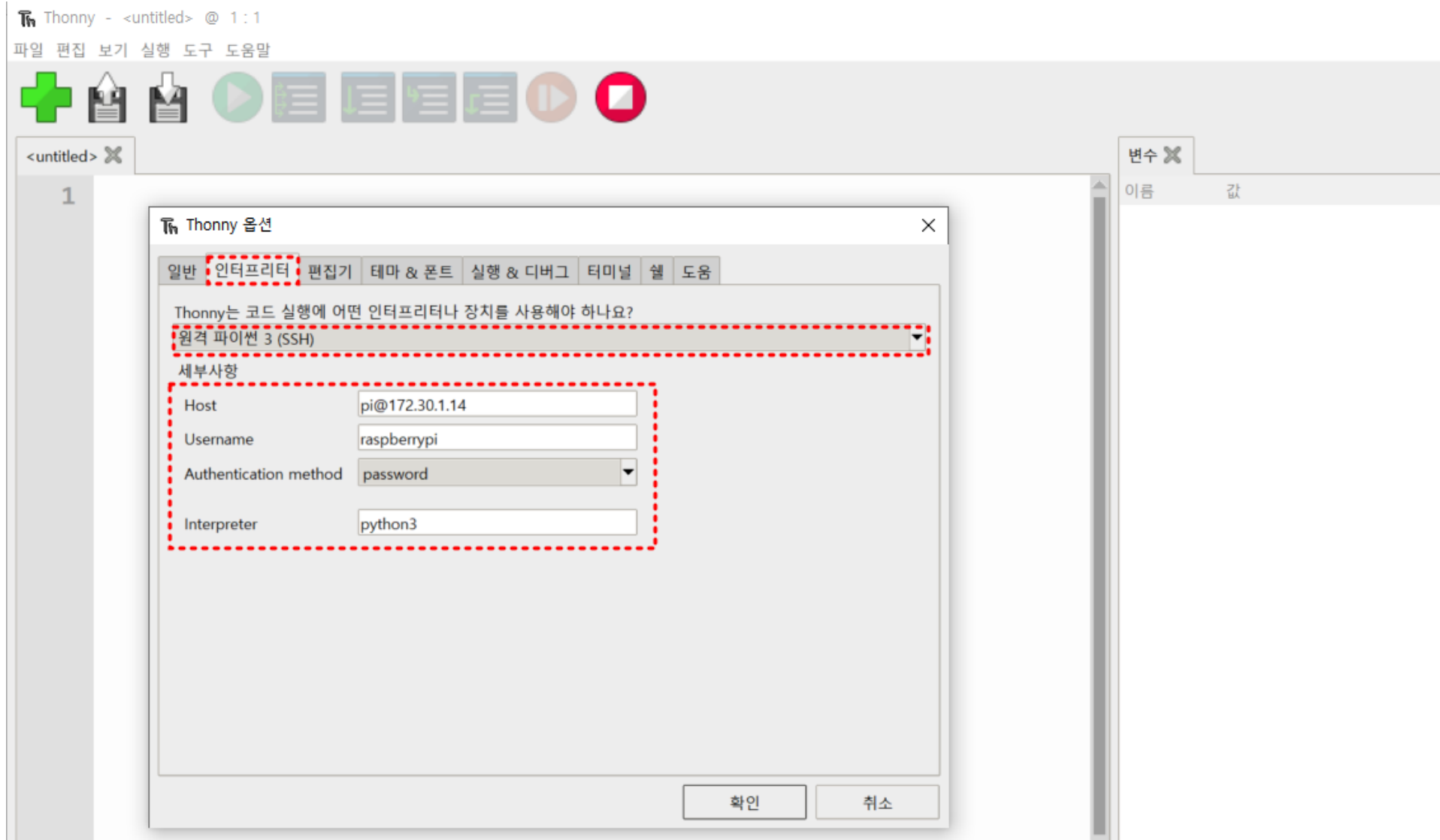
Raspberry Pi 및 PICO 환경설정



디바이스 선택



Raspberry Pi 4 원격 접속 방법



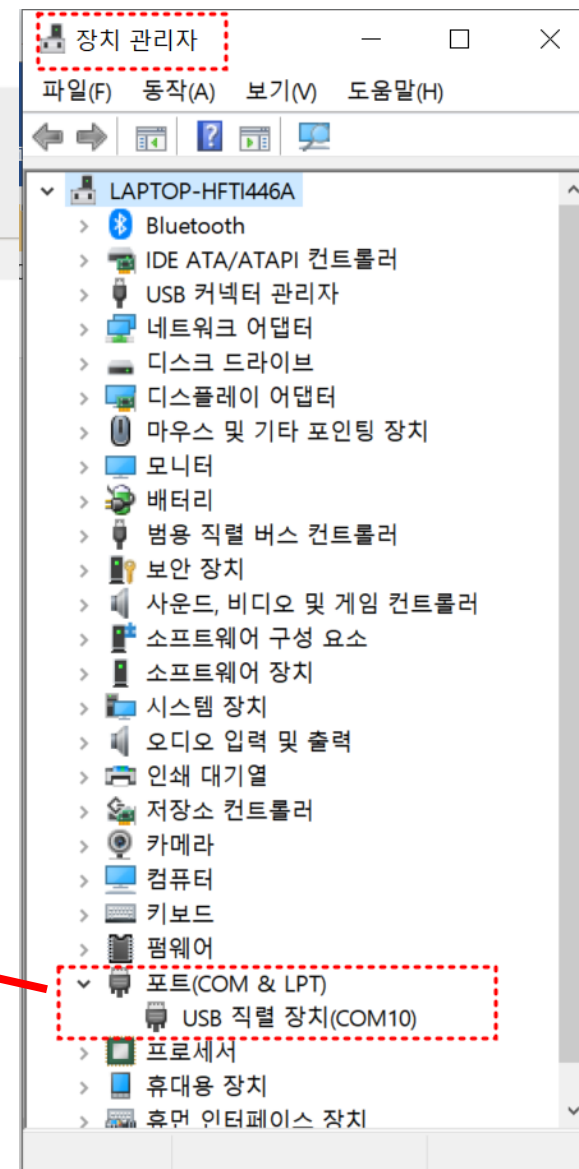
Thonny - <untitled> @ 1 : 1

파일 편집 보기 실행 도구 도움말

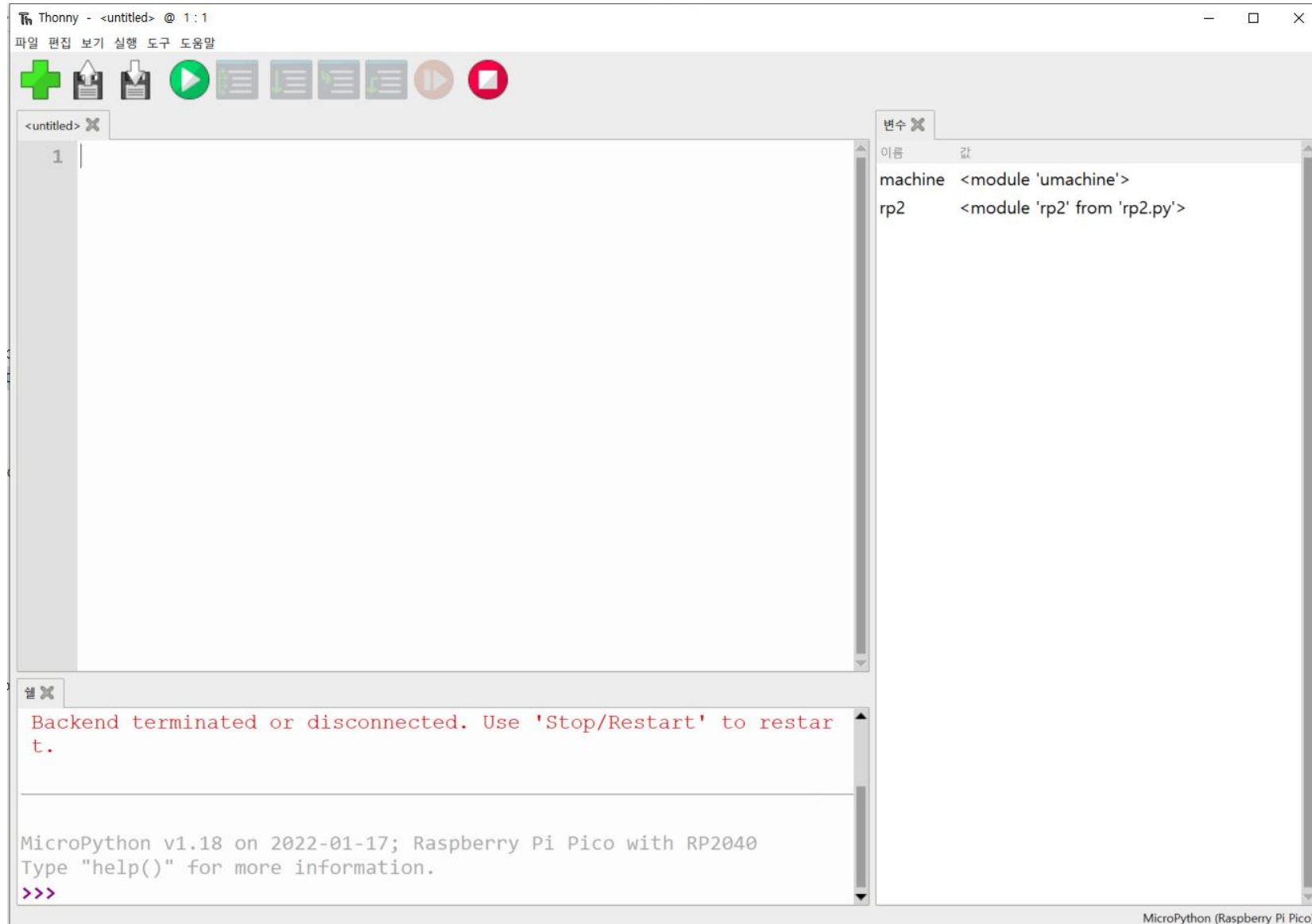


<untitled> X

1



PICO 와 연결된 작업 창



10번 LED 깜빡이는 예제

from machine import Pin

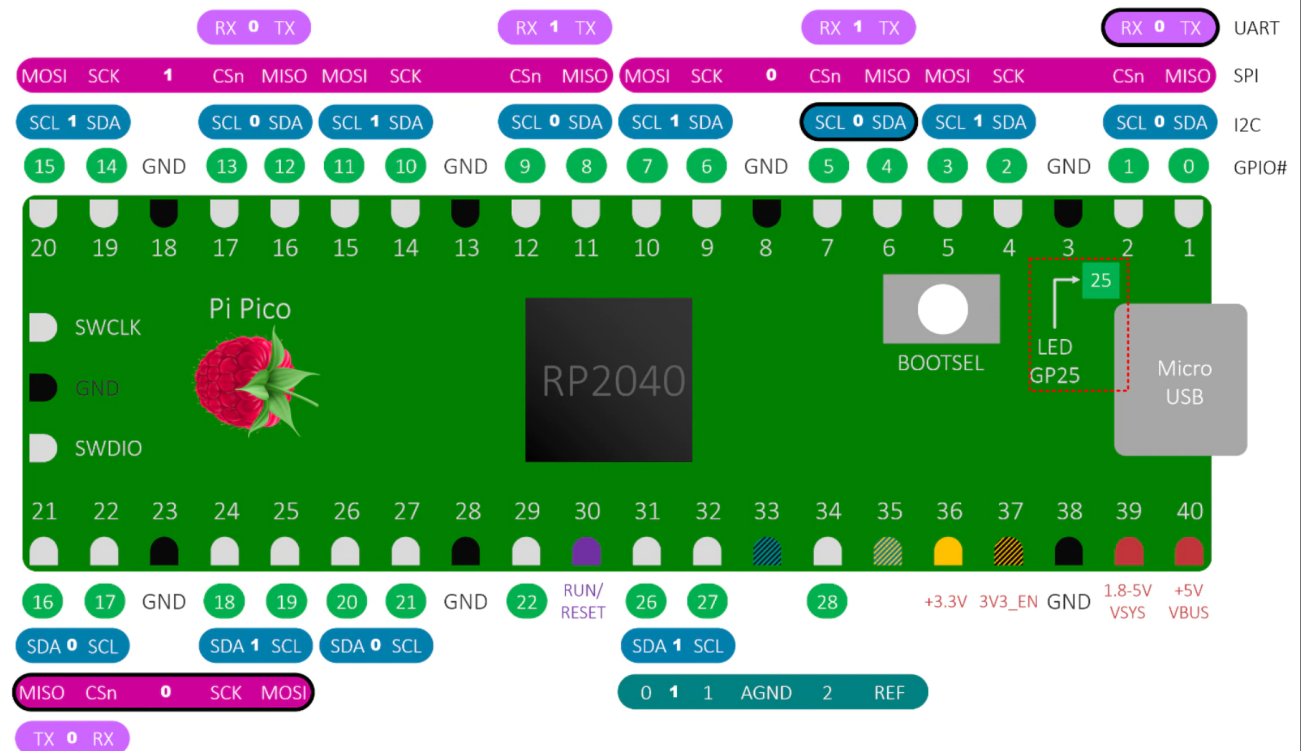
from time import sleep

led = Pin(25,Pin.OUT)

for i in range(10):

 led.toggle()

 sleep(0.5)



10번 LED 깜빡이는 예제

Thonny - Raspberry Pi Pico :: /blink1.py @ 8:14

파일 편집 보기 실행 도구 도움말

```
1 from machine import Pin
2 from time import sleep
3
4 led = Pin(25, Pin.OUT)
5
6 for i in range(10):
7     led.toggle()
8     sleep(0.5)
```

변수

이름	값
Pin	<class 'Pin'>
i	9
led	Pin(25, mode=OUT)
machine	<module 'umachine'>
rp2	<module 'rp2' from 'rp2.py'>
sleep	<function>

설

MicroPython v1.18 on 2022-01-17; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c \$EDITOR_CONTENT
>>> %Run -c \$EDITOR_CONTENT
>>>

MicroPython (Raspberry Pi Pico)

Windows 및 C/C++에서 Raspberry Pi Pico 작업

- <https://community.element14.com/products/raspberry-pi/b/blog/posts/working-with-the-raspberry-pi-pico-with-windows-and-c-c#jive-content-id-Install-Visual-Studio-Code>

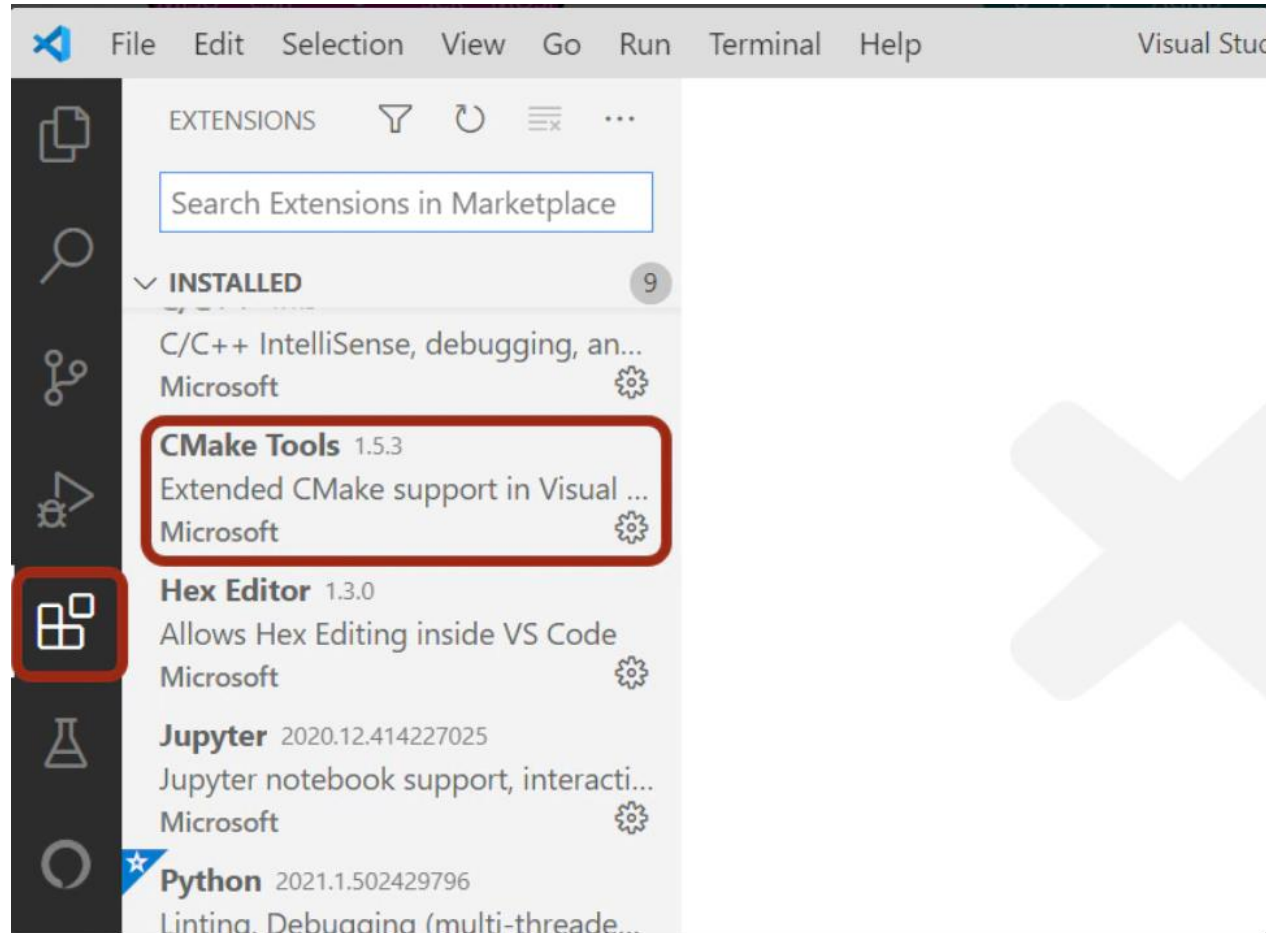
1. 소개
2. Visual Studio 코드 설치
3. CMake 설치
4. Visual Studio용 빌드 도구 설치
5. Python 3, Git 및 ARM GCC 컴파일러 설치
6. 코드 받기
7. 명령줄에서 빌드
8. NMake를 찾을 수 없는 문제 수정
9. 비주얼 코드로 빌드하기
10. 애플리케이션 실행
11. 실행 중인 애플리케이션 문제 해결
12. Intellisense(코드 지원) 사용
13. 나만의 프로젝트 폴더 만들기
14. 요약

1. 소개

- Raspberry Pi Pico 는 이중 ARM Cortex-M0 코어와 256kBytes RAM 이 있는 칩과 외부 하드웨어에 연결하기 위한 일반적인 주변 장치가 혼합된 저가형 마이크로 컨트롤러 보드
- 보드에는 2MB의 플래시 메모리와 USB 포트가 있다.
- Pi Pico는 다른 언어 중에서 C/C++ 또는 Python을 사용하여 프로그래밍할 수 있다.
- 다른 마이크로컨트롤러 보드와 마찬가지로 부팅 버튼을 누른 상태에서 USB 포트를 PC에 연결하면 USB 메모리 키처럼 드라이브 문자가 나타난다.
- C/C++를 사용하는 경우 보드 프로그래밍을 위해 실행 파일을 드라이브 문자로 끌어다 놓을 수 있다.
- 이 문서는 Pi Pico에서 실행할 C/C++ 응용 프로그램을 작성하고 빌드할 준비가 되도록 설정하기 위해 Windows PC에서 수행해야 하는 단계를 설명 한다.
- 여기의 단계는 Windows 10 64비트(x64)를 사용하여 수행수행방법을 기술하였다.

2. Visual Studio 코드 및 Cmake Tools 설치

- Visual Studio Code 웹사이트 정보에 따라 설치한다 .
- 다음과 같이 확장 CMake 도구를 설치한다



3. CMake 설치

- Cmake 다운로드 및 설치
- <https://cmake.org/download/>

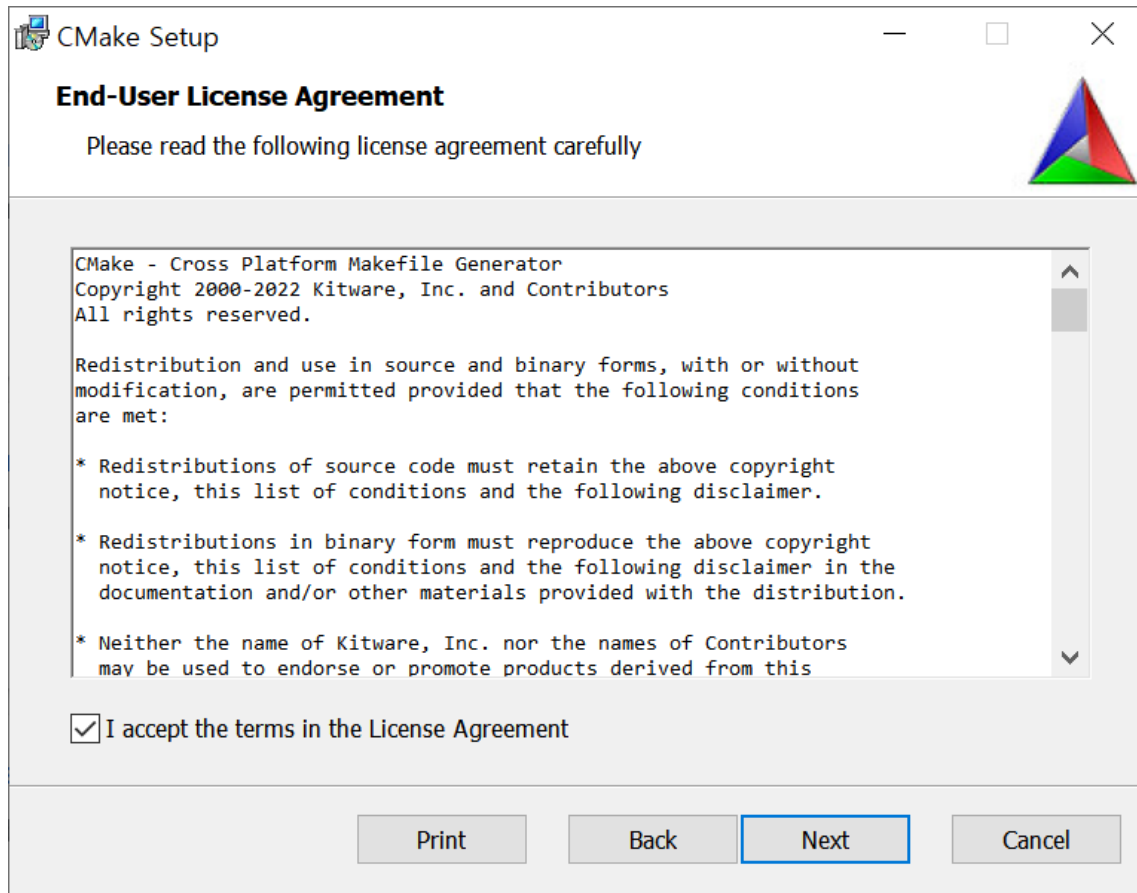
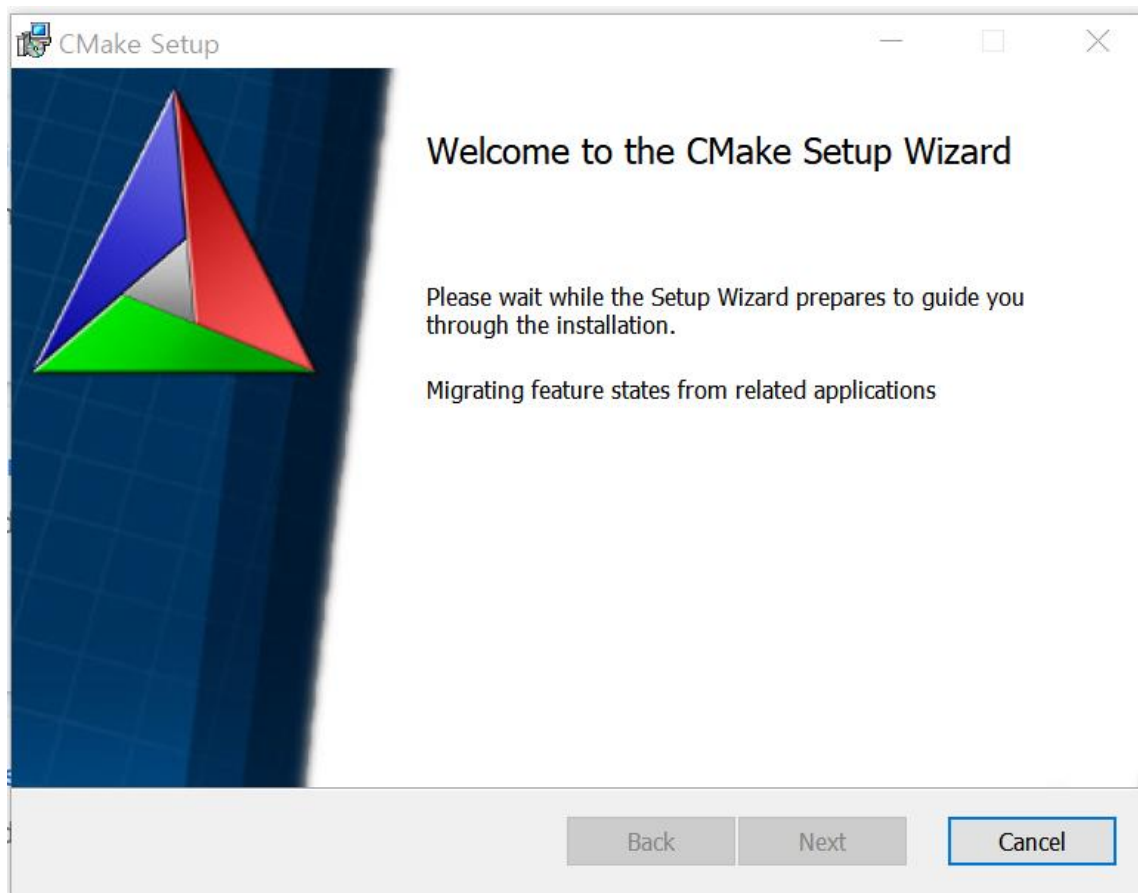
Source distributions:

Platform	Files
Unix/Linux Source (has \n line feeds)	cmake-3.23.0-rc2.tar.gz
Windows Source (has \r\n line feeds)	cmake-3.23.0-rc2.zip

Binary distributions:

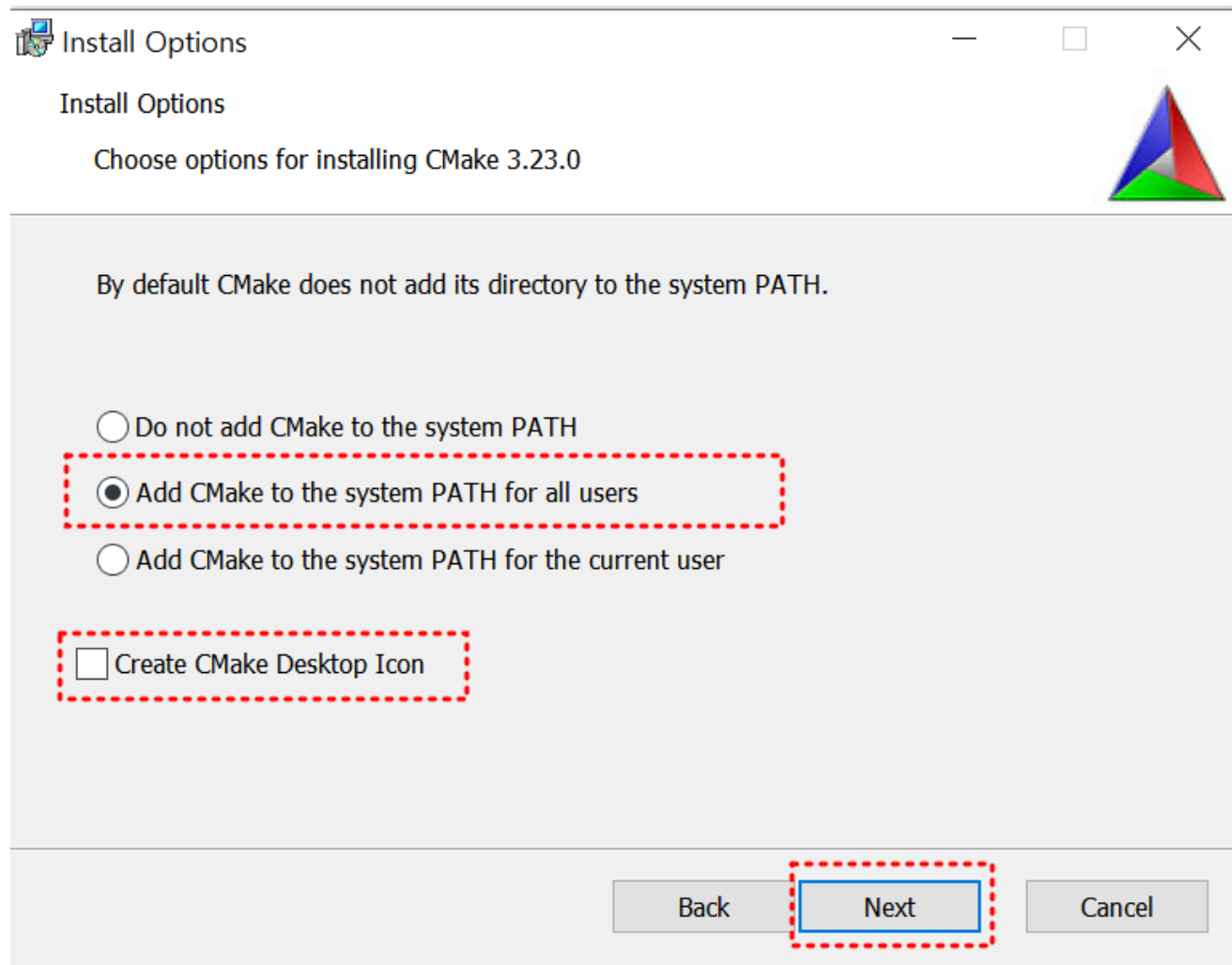
Platform	Files
Windows x64 Installer: Installer tool has changed. Uninstall CMake 3.4 or lower first!	cmake-3.23.0-rc2-windows-x86_64.msi
Windows x64 ZIP	cmake-3.23.0-rc2-windows-x86_64.zip
Windows i386 Installer: Installer tool has changed. Uninstall CMake 3.4 or lower first!	cmake-3.23.0-rc2-windows-i386.msi
Windows i386 ZIP	cmake-3.23.0-rc2-windows-i386.zip
macOS 10.13 or later	cmake-3.23.0-rc2-macos-universal.dmg
	cmake-3.23.0-rc2-macos-universal.tar.gz

3. CMake 설치

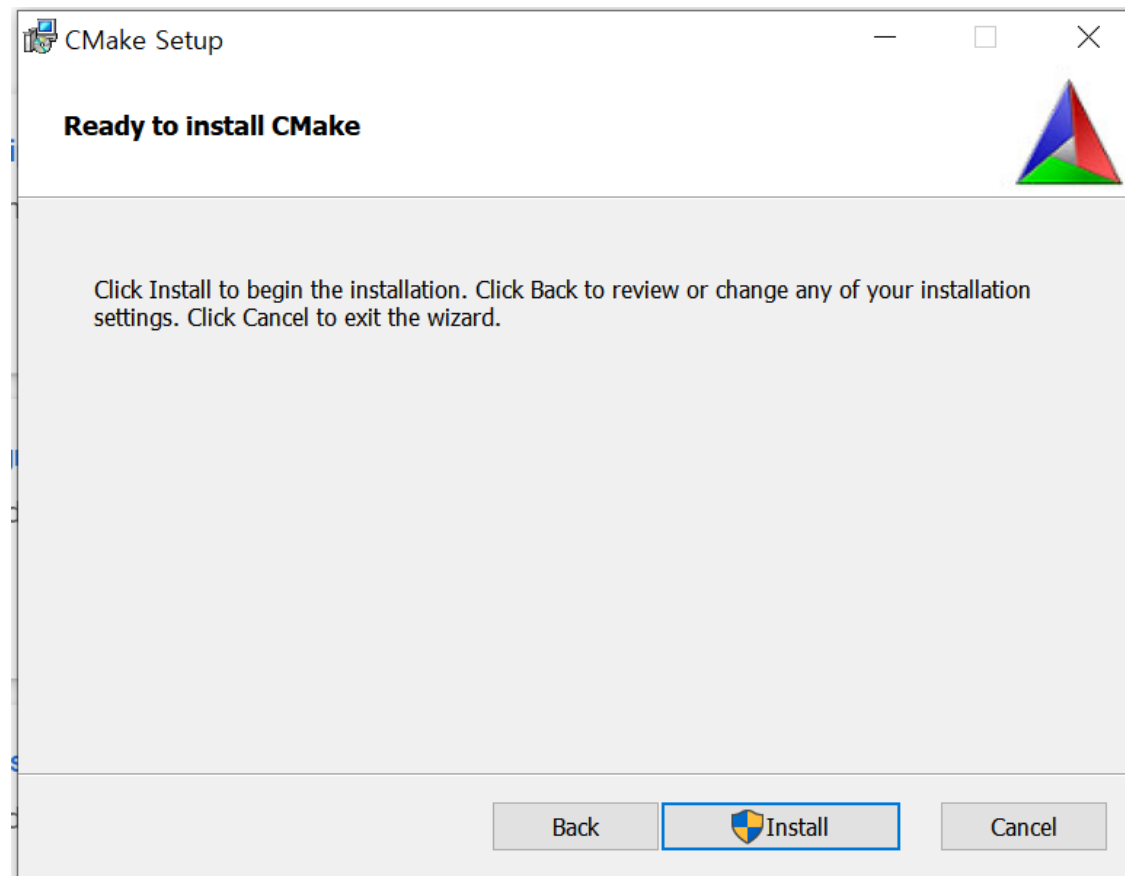
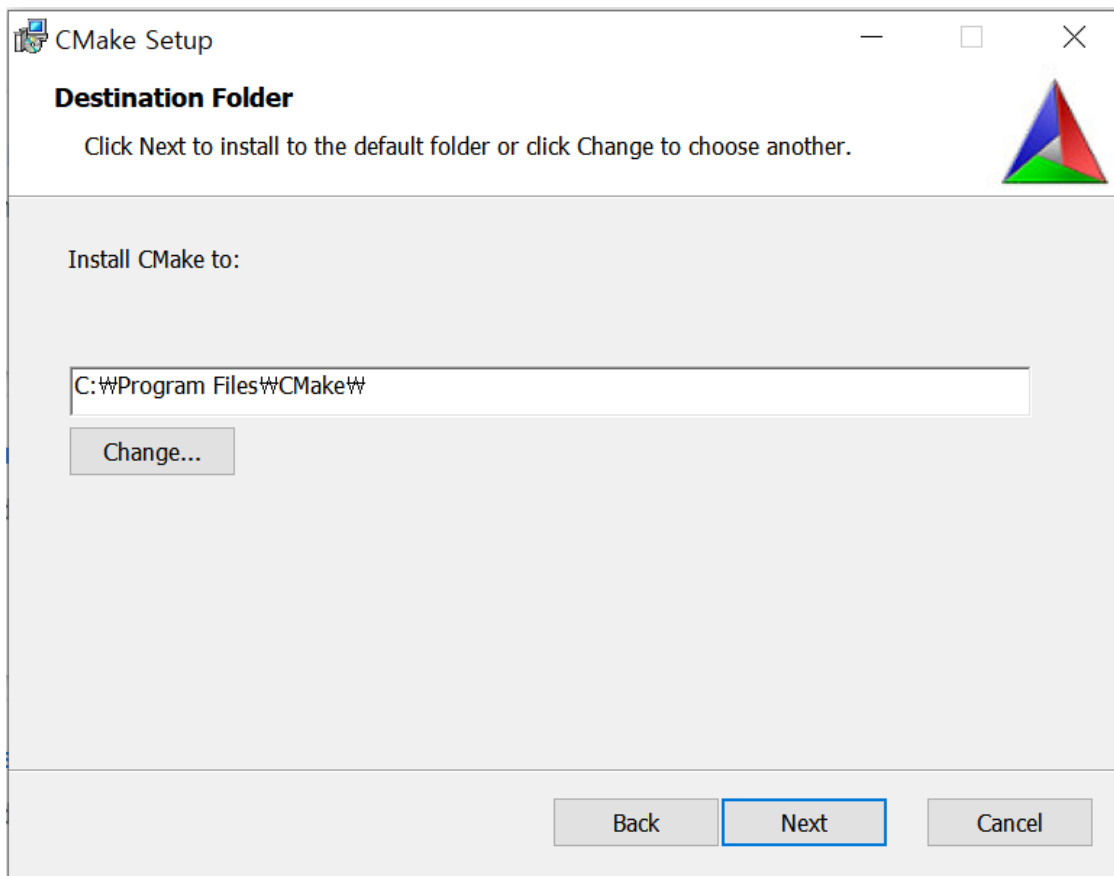


3. CMake 설치

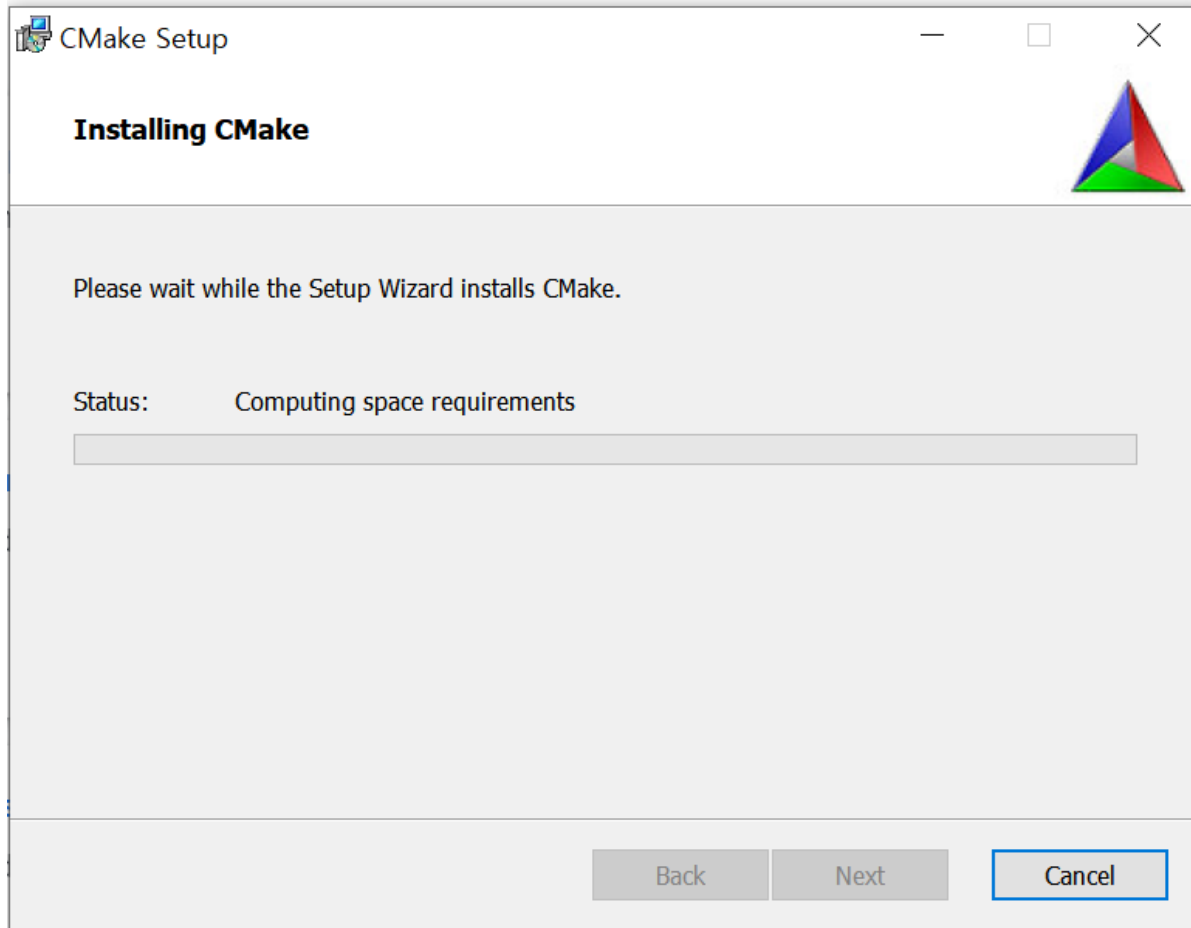
- 다음 옵션을 선택한다.




3. CMake 설치



3. CMake 설치



4. Visual Studio용 빌드 도구 설치




Visual Studio 2022

버전 17.1

Windows에서 .NET 및 C++ 개발자를 위한 최고의 포괄적인 IDE입니다. 소프트웨어 개발의 모든 단계를 향상시키고 개선할 수 있는 다양한 도구와 기능이 완벽하게 포함되어 있습니다.

릴리스 정보 > 버전 비교 > 오프라인 설치 방법 >




Mac용 Visual Studio 2019

버전 8.10

macOS에 기본 제공되는 .NET 개발자를 위한 포괄적인 IDE입니다. 웹, 클라우드 및 게임 개발에 대한 최고 수준의 지원과 함께 플랫폼 간 모바일 앱을 만들기 위한 매우 유용한 도구가 포함되어 있습니다.

[라이선스 활성화](#)에 대해 자세히 알아보기

[무료 다운로드](#)



Visual Studio Code

버전 16.4

Windows, macOS 및 Linux에서 실행되는 독립 실행형 소스 코드 편집기입니다. Java 및 웹 개발자를 위한 최고의 선택이며, 거의 모든 프로그래밍 언어를 지원할 수 있는 수많은 확장 기능을 제공합니다.

Visual Studio Code를 사용하면 [라이선스](#) 및 [개인정보처리방침](#)에 동의하는 것입니다.

[무료 다운로드](#)

커뮤니티

강력한 IDE, 학생, 오픈 소스 제공자 및 개인을 위해 무료로 제공

[무료 다운로드](#)

Professional

소규모 팀에 적합한 Professional IDE

[무료 평가판](#)

Enterprise

모든 규모의 팀을 위한 확장성 뛰어난 통합 솔루션

[무료 평가판](#)

미리 보기

아직 기본 릴리스에 없는 최신 기능에 조기에 액세스할 수 있습니다.

[자세한 정보 >](#)
[릴리스 노트 >](#)

Visual Studio 2019 버전 16.11 릴리스

2022년 10월에 사전이 종료될 예정입니다. 16.10보다 더 이상 지원되지 않습니다. 이러한 중단 릴리스는 다음 업데이트가 출시될 때까지 서비스 수정 사항을 수신합니다.

[my.visualstudio.com](#)의 [다운로드 섹션](#) 또는 Microsoft 카탈로그에서 가장 안전한 최신 버전인 Visual Studio 2019 버전 16.11을 다운로드할 수 있습니다. Visual Studio에서 지원 기준에 대한 자세한 내용은 [Visual Studio 2019 지원 정책](#)을 참조하세요.

- 2022년 2월 8일 Visual Studio 2019 버전 16.11.10 **NEW!**
- 2022년 1월 11일 Visual Studio 2019 버전 16.11.9
- 2021년 12월 14일 — Visual Studio 2019 버전 16.11.8
- 2021년 11월 16일 — Visual Studio 2019 버전 16.11.7
- 2021년 11월 9일 — Visual Studio 2019 버전 16.11.6
- 2021년 10월 12일 — Visual Studio 2019 버전 16.11.5
- 2021년 10월 5일 — Visual Studio 2019 버전 16.11.4
- 2021년 9월 14일 — Visual Studio 2019 버전 16.11.3
- 2021년 8월 25일 — Visual Studio 2019 버전 16.11.2
- 2021년 8월 16일 — Visual Studio 2019 버전 16.11.1
- 2021년 8월 10일 — Visual Studio 2019 버전 16.11.0

Visual Studio 2019 보관된 릴리스 정보

- Visual Studio 2019 버전 16.10 릴리스 정보
- Visual Studio 2019 버전 16.9 릴리스 정보
- Visual Studio 2019 버전 16.8 릴리스 정보
- Visual Studio 2019 버전 16.7 릴리스 정보
- Visual Studio 2019 버전 16.6 릴리스 정보
- Visual Studio 2019 버전 16.5 릴리스 정보
- Visual Studio 2019 버전 16.4 릴리스 정보
- Visual Studio 2019 버전 16.3 릴리스 정보
- Visual Studio 2019 버전 16.2 릴리스 정보
- Visual Studio 2019 버전 16.1 릴리스 정보
- Visual Studio 2019 버전 16.0 릴리스 정보

Visual Studio 2019 블로그

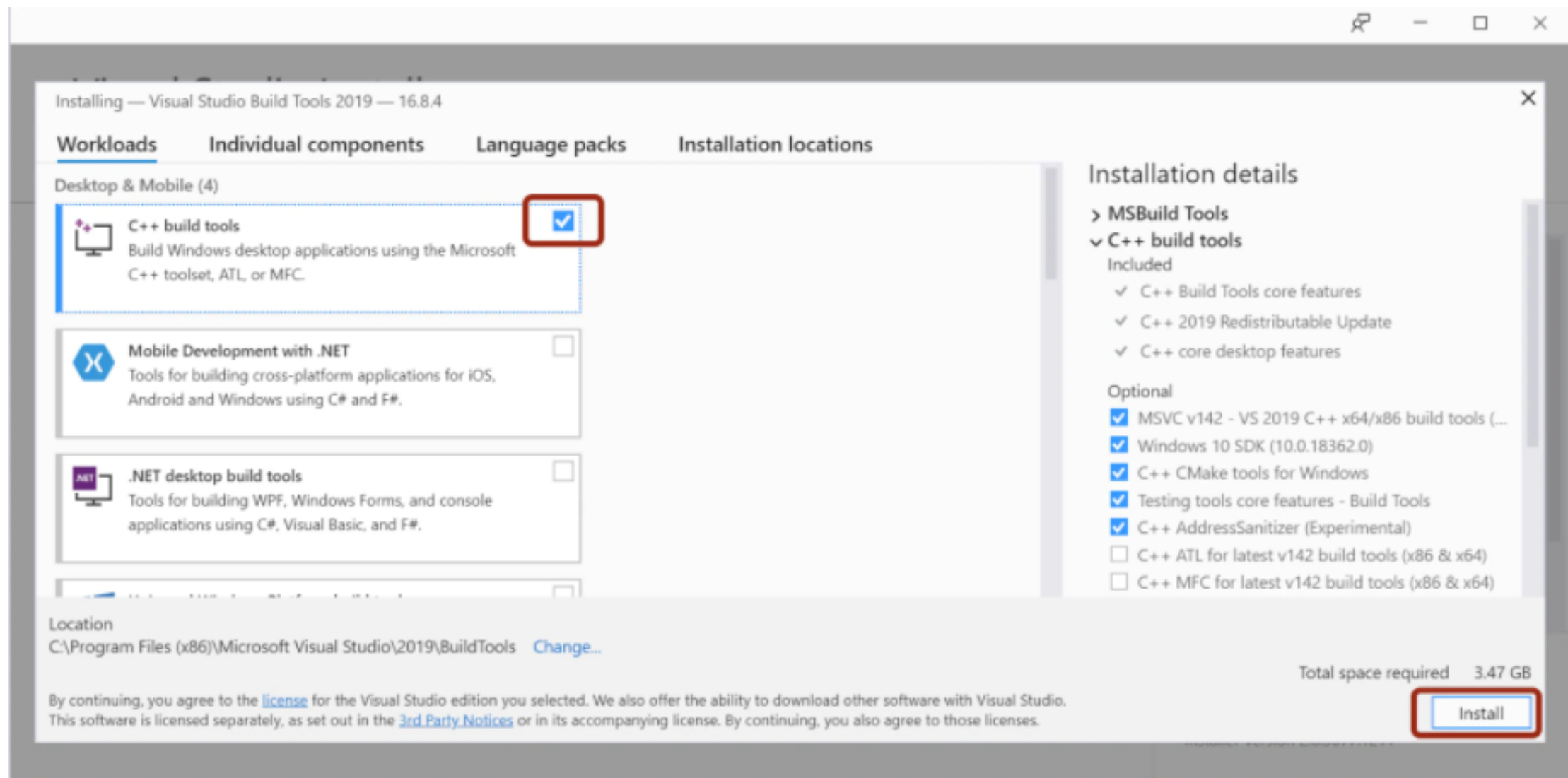
Visual Studio 2019 블로그는 Visual Studio 엔지니어링 팀에서 제공하는 제품 인사이트에 대한 공식적인 정보원입니다. 다음 게시물에서 Visual Studio 2019 릴리스에 대한 자세한 정보를 찾을 수 있습니다.

이 문서의 내용

- Visual Studio 2019 버전 16.11의 새로운 기능
- Visual Studio 2019 버전 16.11.10
- Visual Studio 2019 버전 16.11.9
- Visual Studio 2019 버전 16.11.8
- Visual Studio 2019 버전 16.11.7
- Visual Studio 2019 버전 16.11.6
- Visual Studio 2019 버전 16.11.5
- Visual Studio 2019 버전 16.11.4
- Visual Studio 2019 버전 16.11.3
- Visual Studio 2019 버전 16.11.2
- Visual Studio 2019 버전 16.11.1
- Visual Studio 2019 버전 16.11.0
- 알려진 문제
- 피드백 및 제안
- Blogs
- Visual Studio 2019 릴리스 정보 기록

4. Visual Studio용 빌드 도구 설치

- C++ 빌드 도구를 선택하고 설치한다.



5. Python 3, Git 및 ARM GCC 컴파일러 설치

- Pico 시작하기 PDF 가이드 를 참조
- **C/C++ SDK**
- SDK 설정
- C/C++ SDK를 사용하는 방법에 대한 전체 설명을 보려면 ' 시작하기 ' 설명서 를 본다.
- 그러나 Raspberry Pi 에서 Pico용으로 개발하려는 경우 명령줄에서 설정 스크립트 를 실행하여 C/C++ 도구 체인을 빠르게 설정할 수 있다 .
- 설정 스크립트를 실행하기 전에 Raspberry Pi의 OS가 최신 버전 인지 확인해야 한다 .

5. Python 3, Git 및 ARM GCC 컴파일러 설치

- **라즈베리 파이 피코 C/C++ SDK**

- 공식 C SDK는 명령줄에서 또는 Visual Studio Code, Eclipse 및 CLion과 같은 널리 사용되는 통합 개발 환경에서 사용할 수 있다.
- 시작하려면 C/C++ SDK 및 예제를 다운로드하고 '시작하기' 설명서를 살펴보고 시작한다.
- 또는 빠른 설정을 위해 다음 섹션을 참조하십시오.
- **SDK Github 리포지토리** : <https://github.com/raspberrypi/pico-sdk>
- 예제 Github 리포지토리
- 다음에서 C/C++ SDK에 대한 설명서를 찾을 수 있습니다.
- Raspberry Pi Pico 시작하기
- Raspberry Pi Pico 및 기타 RP2040 기반 마이크로컨트롤러 보드를 사용한 C/C++ 개발
- 라즈베리 파이 피코 C/C++ SDK
- RP2040 마이크로컨트롤러에서 C/C++ 개발을 위한 라이브러리 및 도구
- Raspberry Pi Pico C/C++ SDK에 대한 API 수준 Doxygen 설명서는 마이크로 사이트로도 제공됩니다 .

5. Python 3, Git 및 ARM GCC 컴파일러 설치

- 라즈베리 파이 피코 SDK
- Raspberry Pi Pico SDK(이하 SDK)는 Raspberry Pi Pico와 같은 RP2040 기반 장치용 프로그램을 C, C++ 또는 어셈블리어로 작성하는 데 필요한 헤더, 라이브러리 및 빌드 시스템을 제공한다.
- SDK는 비임베디드 C 개발자와 임베디드 C 개발자 모두에게 친숙한 API 및 프로그래밍 환경을 제공하도록 설계되어있다.
- 단일 프로그램은 한 번에 장치에서 실행되며 기존 `main()` 방법으로 시작한다.
- 표준 C/C++ 라이브러리는 PIO(Programmable IO)를 포함한 모든 RP2040 하드웨어에 액세스하기 위한 C 레벨 라이브러리/API와 함께 지원된다.
- 또한 SDK는 다양한 유틸리티와 함께 타이머, 동기화, USB(TinyUSB) 및 멀티 코어 프로그래밍을 처리하기 위한 더 높은 수준의 라이브러리를 제공한다.
- SDK는 간단한 애플리케이션에서 MicroPython과 같은 완전한 런타임 환경, RP2040의 온칩 부트롬 자체와 같은 저수준 소프트웨어에 이르기까지 무엇이든 구축하는 데 사용할 수 있다.
- 아직 SDK에 포함할 준비가 되지 않은 추가 라이브러리/API는 `pico-extras` 에서 찾을 수 있습니다 .

5. Python 3, Git 및 ARM GCC 컴파일러 설치

- 이에 대해서는 Pico 시작하기 PDF 가이드 를 참조
- neilk 블로그 게시물을 참조
- Adventures with the Raspberry Pi pico

6. 코드 받기

- Pico 폴더 생성(예 : C:\development\pico)
- **P**ico 사용자 가이드를 따르고
- git clone을 사용하거나 여기에 표시된 대로 수동으로 수행한다.
- 이상적으로는 git clone을 사용한다.
- 다음에서 코드의 zip 번들 다운로드:
- <https://github.com/raspberrypi/pico-sdk>
- <https://github.com/raspberrypi/pico-examples>

6. 코드 받기

- <https://github.com/raspberrypi/pico-sdk>

The screenshot shows the GitHub repository page for `raspberrypi / pico-sdk`. The repository has 25 watchers, 3 issues, 4 pull requests, and 4 actions. The 'Code' button is highlighted with a red box. The dropdown menu is open, showing options to clone the repository using HTTPS, SSH, or GitHub CLI, or to open it with GitHub Desktop. The 'Download ZIP' option is also highlighted with a red box.

Repository: `raspberrypi / pico-sdk` Watch 25

Code Issues 3 Pull requests 4 Actions Projects Wiki Security

master 4 branches 1 tag

bartekpacia and kilograham Update README.m

File	Version
cmake	Initial Release
docs	Initial Release
external	Initial Release
lib	Initial Release
src	Initial Release

Clone ?

HTTPS SSH GitHub CLI

`https://github.com/raspberrypi/pico-sd`

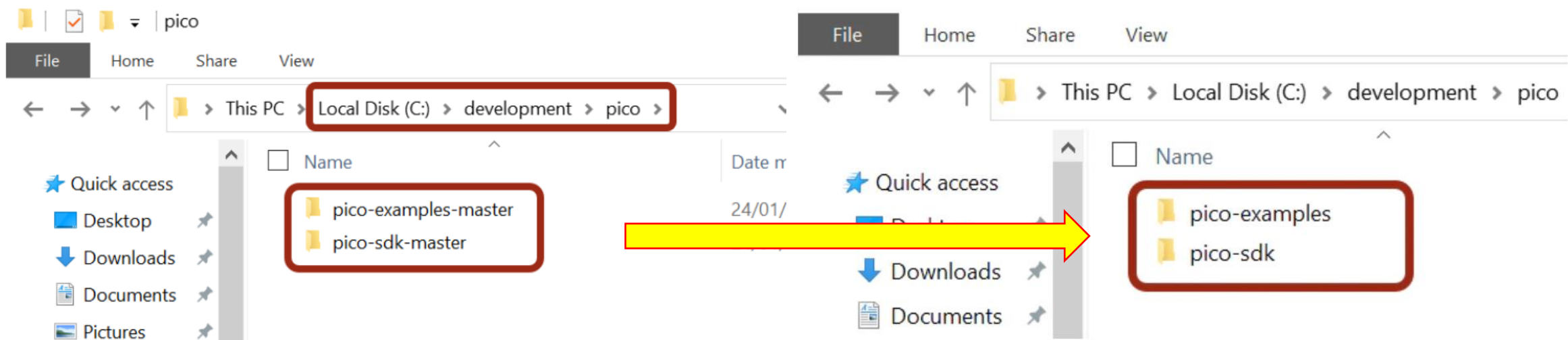
Use Git or checkout with SVN using the web URL.

Open with GitHub Desktop

Download ZIP

6. 코드 받기

- Pico 폴더 에 압축을 풀고
- 다음과 같이 이름을 변경한다.



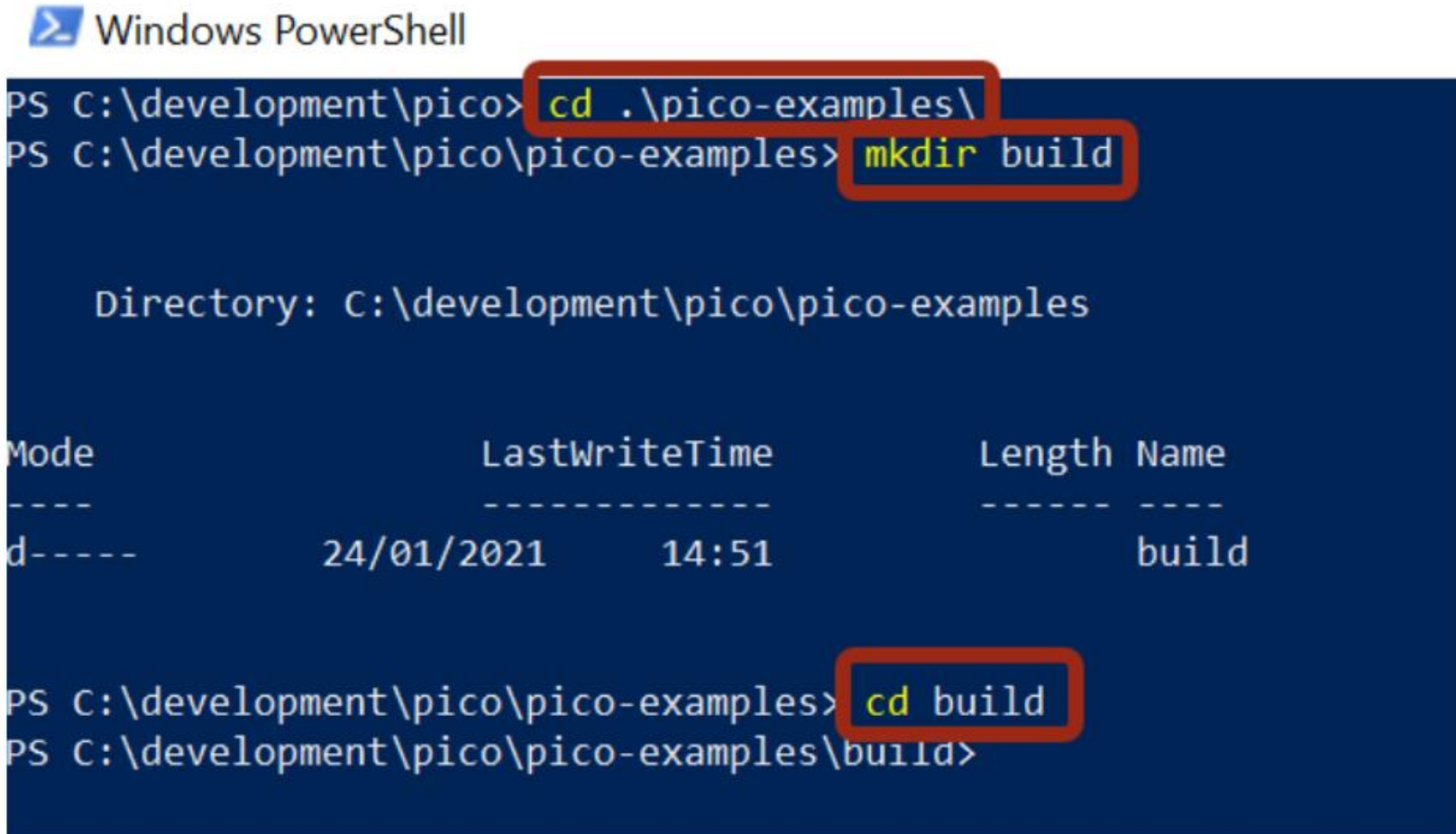
7. 명령줄에서 빌드

- PowerShell 을 시작하고 다음과 같이 압축을 푼 디렉토리로 이동한다.
- Pico디렉토리로 이동하여
- `setx PICO_SDK_PATH "c:\development\pico\pico-sdk"`

```
PS C:\development\pico> setx PICO_SDK_PATH "c:\development\pico\pico-sdk"
SUCCESS: Specified value was saved.
PS C:\development\pico>
```

7. 명령줄에서 빌드

- PowerShell을 닫고 새 PowerShell을 연다.
- pico-examples 내부에 빌드 폴더를 만든 다음 해당 폴더 로 이동한다.



```
Windows PowerShell
PS C:\development\pico> cd .\pico-examples\
PS C:\development\pico\pico-examples> mkdir build

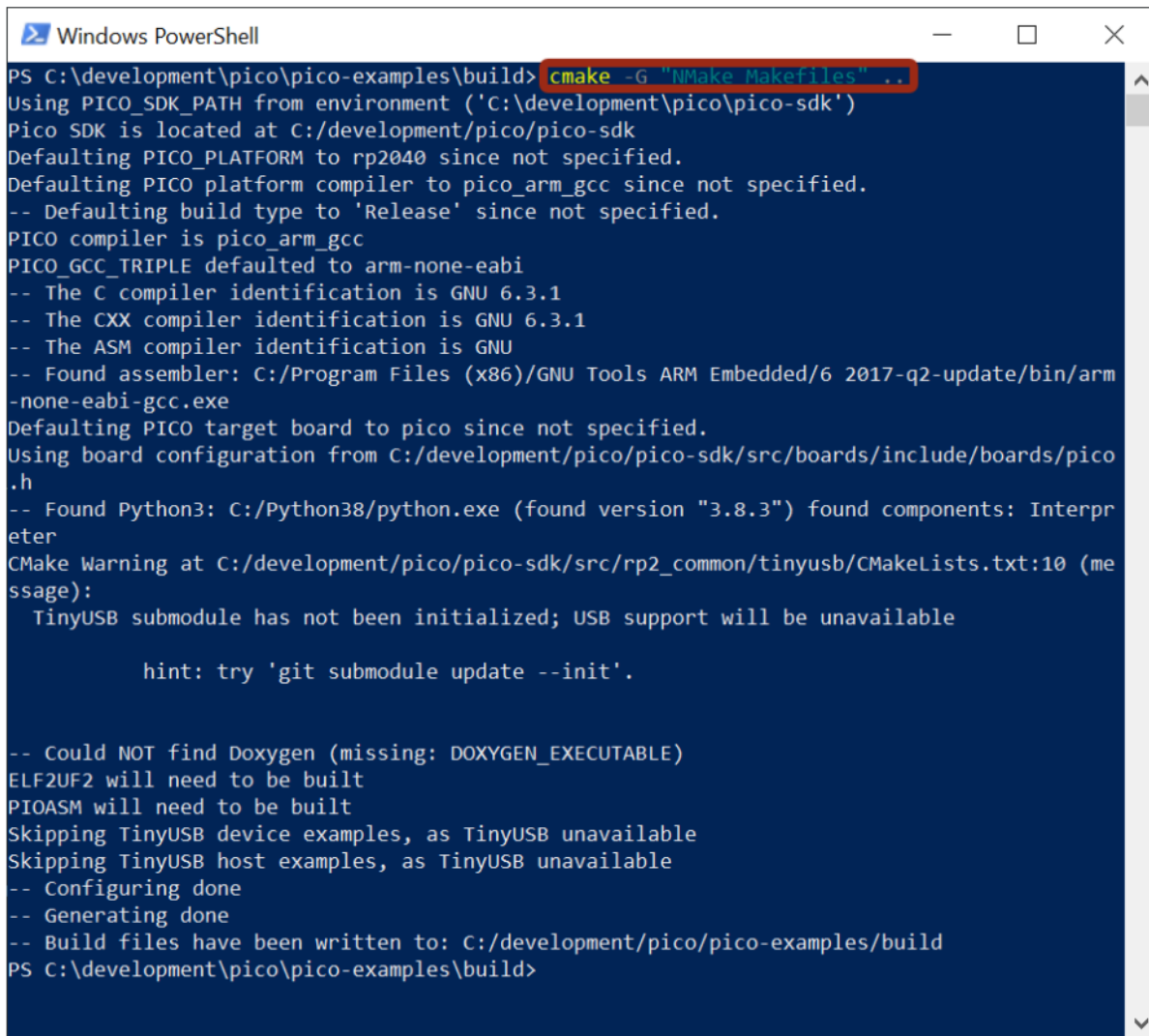
Directory: C:\development\pico\pico-examples

Mode                LastWriteTime         Length Name
----                -
d-----          24/01/2021   14:51             build

PS C:\development\pico\pico-examples> cd build
PS C:\development\pico\pico-examples\build>
```

7. 명령줄에서 빌드

- `cmake -G "NMake Makefiles" ..`



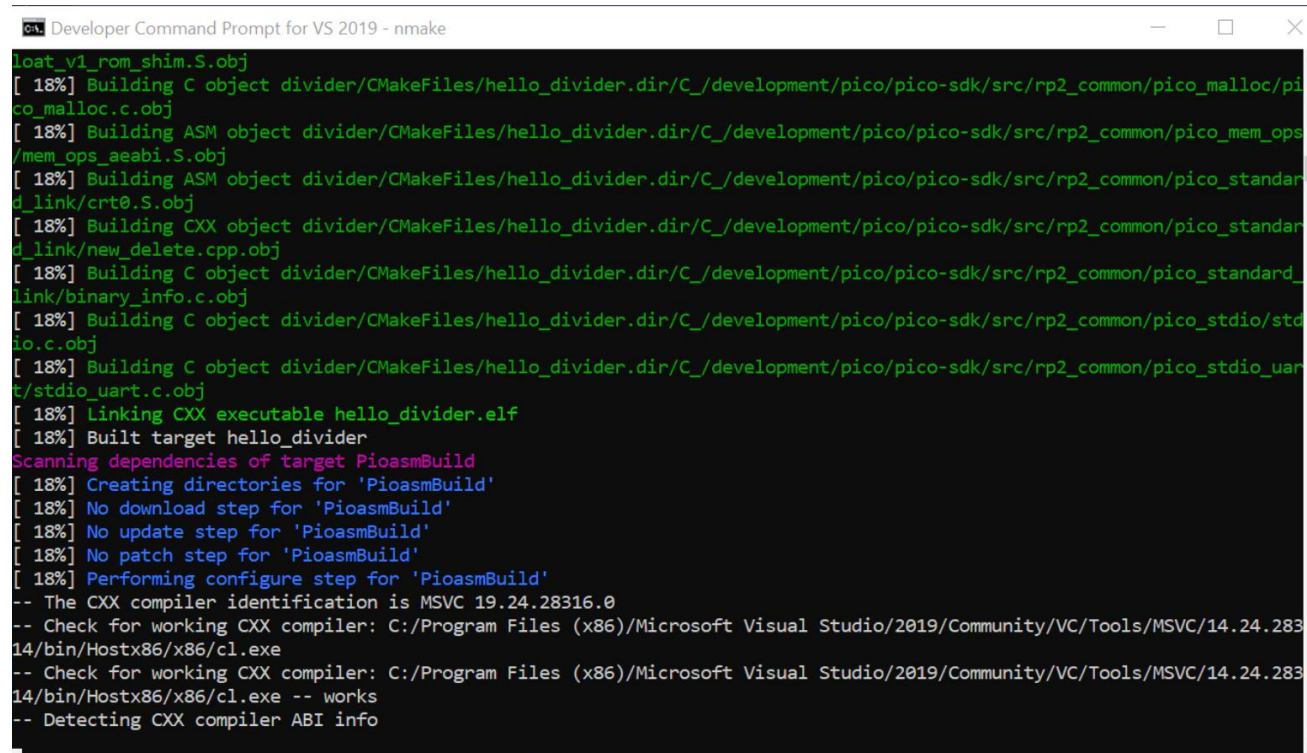
```
Windows PowerShell
PS C:\development\pico\pico-examples\build> cmake -G "NMake Makefiles" ..
Using PICO_SDK_PATH from environment ('C:\development\pico\pico-sdk')
Pico SDK is located at C:/development/pico/pico-sdk
Defaulting PICO_PLATFORM to rp2040 since not specified.
Defaulting PICO_platform_compiler to pico_arm_gcc since not specified.
-- Defaulting build type to 'Release' since not specified.
PICO compiler is pico_arm_gcc
PICO_GCC_TRIPLE defaulted to arm-none-eabi
-- The C compiler identification is GNU 6.3.1
-- The CXX compiler identification is GNU 6.3.1
-- The ASM compiler identification is GNU
-- Found assembler: C:/Program Files (x86)/GNU Tools ARM Embedded/6 2017-q2-update/bin/arm-none-eabi-gcc.exe
Defaulting PICO target board to pico since not specified.
Using board configuration from C:/development/pico/pico-sdk/src/boards/include/boards/pico.h
-- Found Python3: C:/Python38/python.exe (found version "3.8.3") found components: Interpreter
CMake Warning at C:/development/pico/pico-sdk/src/rp2_common/tinyusb/CMakeLists.txt:10 (message):
  TinyUSB submodule has not been initialized; USB support will be unavailable

  hint: try 'git submodule update --init'.

-- Could NOT find Doxygen (missing: DOXYGEN_EXECUTABLE)
ELF2UF2 will need to be built
PIOASM will need to be built
Skipping TinyUSB device examples, as TinyUSB unavailable
Skipping TinyUSB host examples, as TinyUSB unavailable
-- Configuring done
-- Generating done
-- Build files have been written to: C:/development/pico/pico-examples/build
PS C:\development\pico\pico-examples\build>
```

7. 명령줄에서 빌드

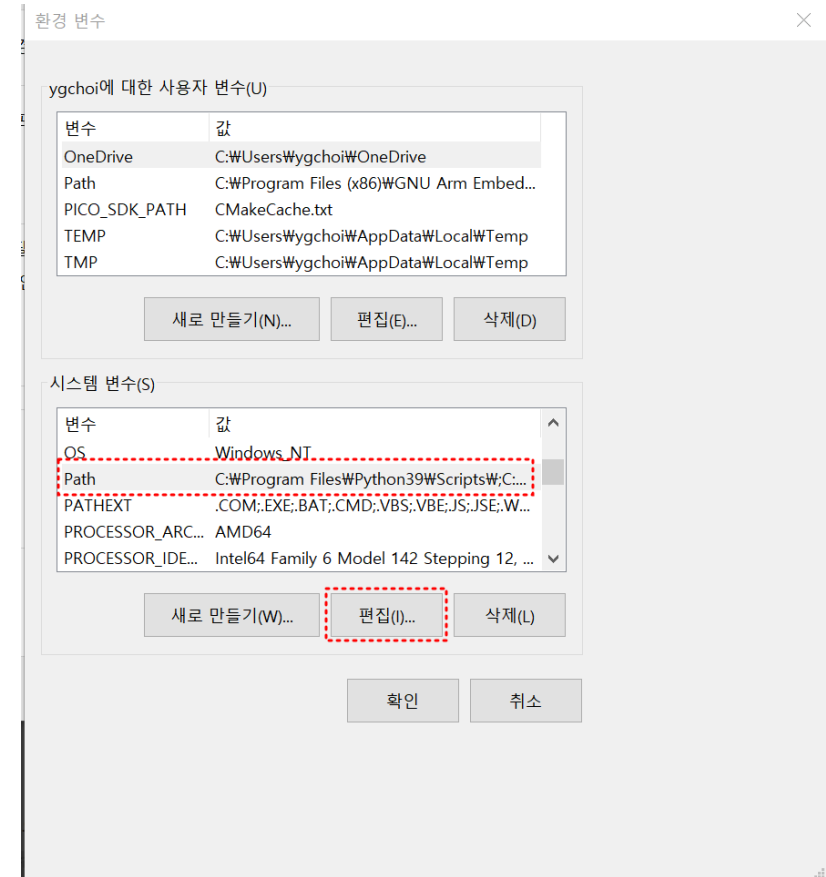
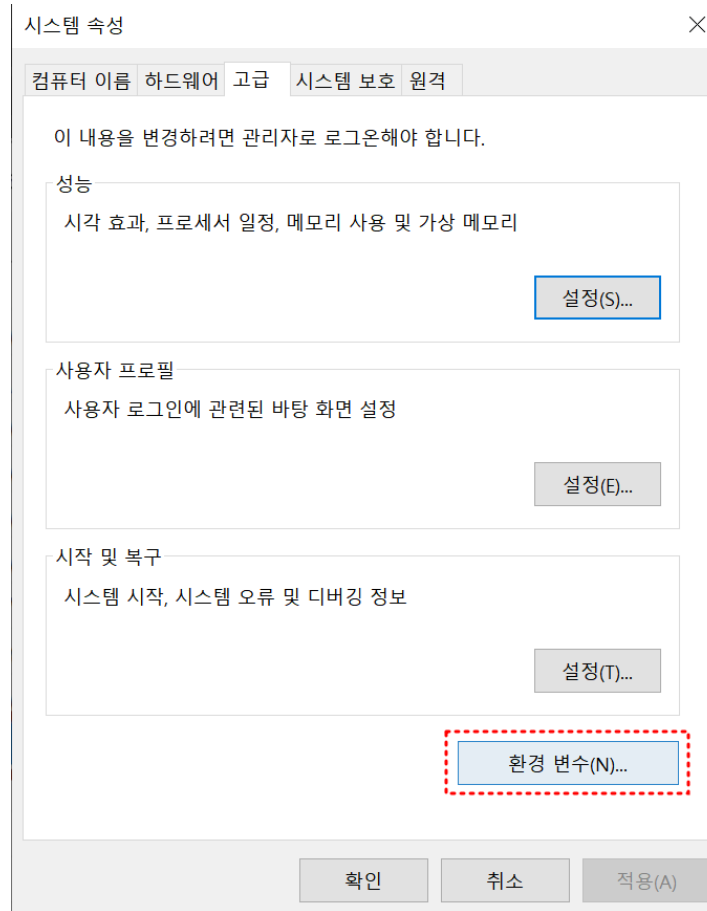
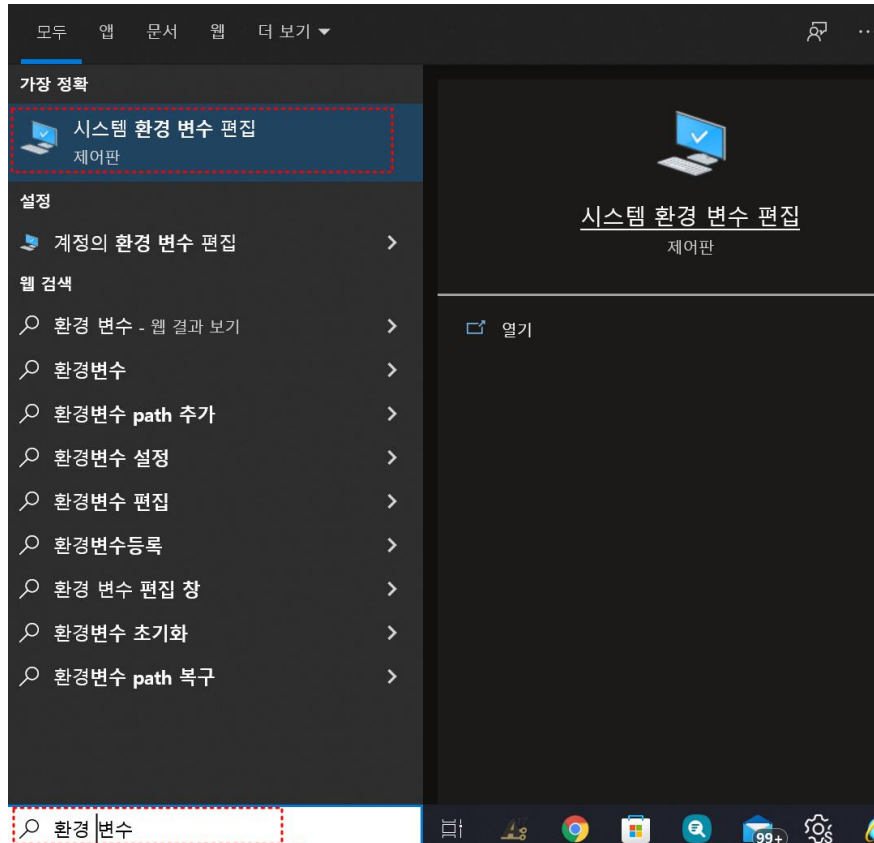
- 이제 코드를 빌드하기 위해 Microsoft 빌드 도구(특히 nmake)의 환경 변수가 PowerShell 에서 작동하도록 제대로 설정되지 않았을 수도 있다.
- 오류가 발생하면 대신 VS 2019용 개발자 명령 프롬프트를 사용하도록 전환한다.
- 프롬프트를 열고 이전과 같이 빌드 폴더로 이동한 다음 nmake 를 입력합니다.
- 모든 예제를 빌드하기 시작한다.



```
Developer Command Prompt for VS 2019 - nmake
load_v1_rom_shim.S.obj
[ 18%] Building C object divider/CMakeFiles/hello_divider.dir/C:/development/pico/pico-sdk/src/rp2_common/pico_malloc/pico_malloc.c.obj
[ 18%] Building ASM object divider/CMakeFiles/hello_divider.dir/C:/development/pico/pico-sdk/src/rp2_common/pico_mem_ops/mem_ops_eeabi.S.obj
[ 18%] Building ASM object divider/CMakeFiles/hello_divider.dir/C:/development/pico/pico-sdk/src/rp2_common/pico_standard_link/crt0.S.obj
[ 18%] Building CXX object divider/CMakeFiles/hello_divider.dir/C:/development/pico/pico-sdk/src/rp2_common/pico_standard_link/new_delete.cpp.obj
[ 18%] Building C object divider/CMakeFiles/hello_divider.dir/C:/development/pico/pico-sdk/src/rp2_common/pico_standard_link/binary_info.c.obj
[ 18%] Building C object divider/CMakeFiles/hello_divider.dir/C:/development/pico/pico-sdk/src/rp2_common/pico_stdio/stdio.c.obj
[ 18%] Building C object divider/CMakeFiles/hello_divider.dir/C:/development/pico/pico-sdk/src/rp2_common/pico_stdio_uart/stdio_uart.c.obj
[ 18%] Linking CXX executable hello_divider.elf
[ 18%] Built target hello_divider
Scanning dependencies of target PioasmBuild
[ 18%] Creating directories for 'PioasmBuild'
[ 18%] No download step for 'PioasmBuild'
[ 18%] No update step for 'PioasmBuild'
[ 18%] No patch step for 'PioasmBuild'
[ 18%] Performing configure step for 'PioasmBuild'
-- The CXX compiler identification is MSVC 19.24.28316.0
-- Check for working CXX compiler: C:/Program Files (x86)/Microsoft Visual Studio/2019/Community/VC/Tools/MSVC/14.24.28314/bin/Hostx86/x86/cl.exe
-- Check for working CXX compiler: C:/Program Files (x86)/Microsoft Visual Studio/2019/Community/VC/Tools/MSVC/14.24.28314/bin/Hostx86/x86/cl.exe -- works
-- Detecting CXX compiler ABI info
```

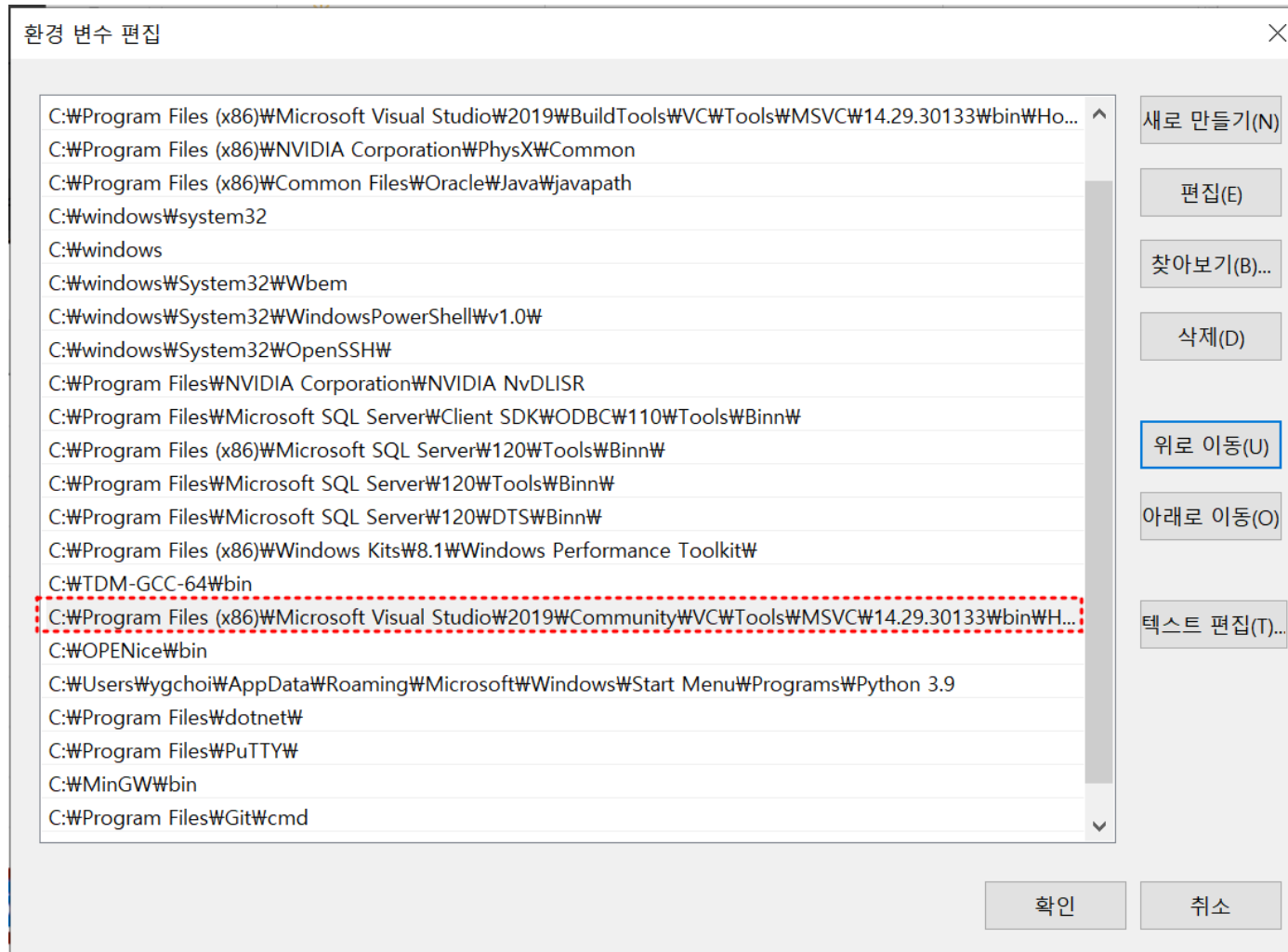
8. NMake를 찾을 수 없는 문제 수정

- 시스템 환경 변수 편집



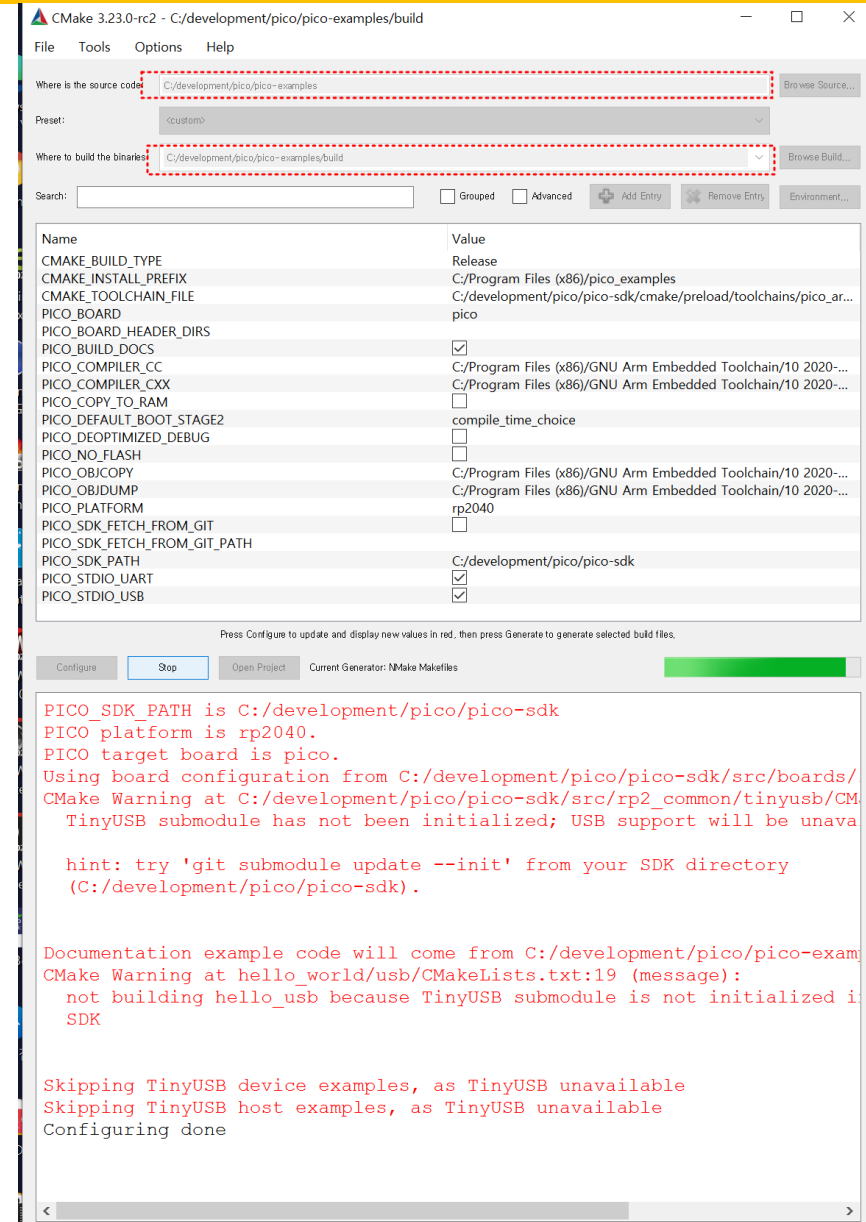
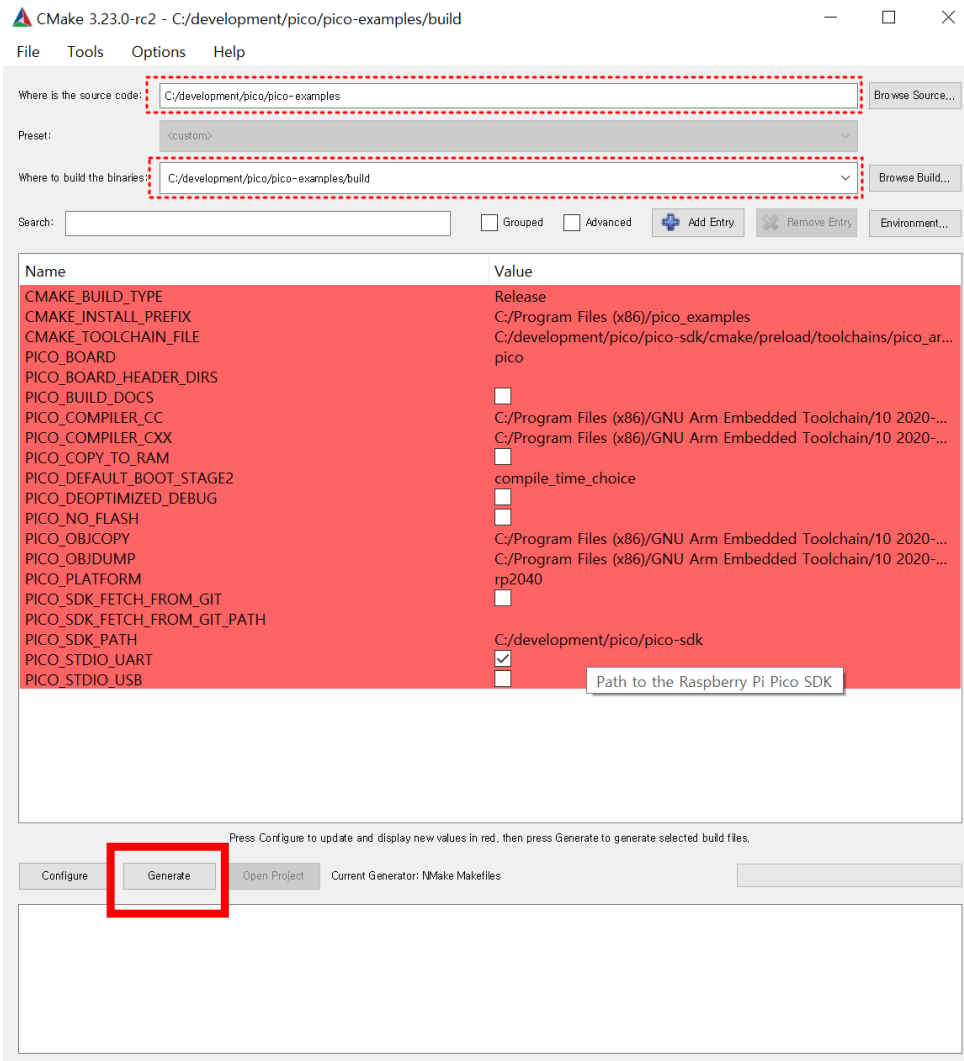
8. NMake를 찾을 수 없는 문제 수정

- 경로를 확인한후 복사하여 새로 만들기로 다음과 같이 삽입한다.



8. NMake를 찾을 수 없는 문제 수정

- cmake에 경로를 추가한다.



9. 비주얼 코드로 빌드하기

- 단계를 시도하기 전에 명령줄에서 성공적으로 빌드할 수 있는지 확인한다.
- 창의 왼쪽 하단에 있는 톱니바퀴를 클릭하고 설정 을 선택합니다 .

