C++ Project Setup and Build Guide

This document provides step-by-step instructions for setting up the development environment and building the project.

Table of Contents

Prerequisites

- Windows 10/11
- Administrator privileges (for software installation)
- 5GB+ free disk space

Environment Setup

1. Install MSYS2

1. Download the installer from MSYS2 official website
2. Run the installer and follow default installation steps
3. Launch "MSYS2 UCRT64" from Start Menu
4. Update packages:

   ```
   bash
   ```

   ```
   pacman -Syu
   ```

5. Install required toolchain:

   ```
   bash
   ```

   ```
   pacman -S --needed base-devel mingw-w64-ucrt-x86_64-toolchain
   ```

2. Install Visual Studio Code

1. Download from VSCode website
2. Run the installer with default options
3. Install these extensions:
   - C/C++ (ms-vscode.cpptools)
   - CMake Tools (ms-vscode.cmake-tools)
   - CMake Language Support (twxs.cmake)

3. Configure System PATH

Add these paths to your System Environment Variables:

```
C:\msys64\ucrt64\bin
C:\msys64\usr\bin
```

Building the Project

Option A: Using VSCode CMake Tools (Recommended)

1. Open project folder in VSCode
2. Press `Ctrl+Shift+P` and run:

   ```
   CMake: Configure
   ```

3. Select "MinGW Makefiles" as generator
4. Choose "GCC for MSYS2 UCRT64" as compiler
5. Build the project:

   ```
   CMake: Build
   ```

   or click the build button in the status bar

Option B: Command Line Build

1. Open terminal in project root
2. Execute:

   ```bash
   mkdir build
   cd build
   cmake .. -G "MinGW Makefiles"
   cmake --build .
   ```

Running the Application

The compiled executable will be located at:

```
./Debug/SDKNetApp.exe
```

To run:

```
bash

cd Debug
./SDKNetApp.exe
```

Troubleshooting

Common Issues

Error	Solution
Compiler not found	Verify PATH contains MSYS2 UCRT64 bin folder
Missing DLLs	Copy required DLLs from `C:\msys64\ucrt64\bin` to executable directory
CMake configuration fails	Check CMakeLists.txt for syntax errors
Linker errors	Verify all libraries are properly specified in CMakeLists.txt

FAQ

Q: Can I use a different compiler?

A: Yes, but you'll need to modify the CMakeLists.txt and c_cpp_properties.json files accordingly.

Q: How do I clean the build?

A: Delete the `build` directory or run `cmake --build . --target clean`

Q: Where are the project dependencies located?

A: Check the `include` and `lib64` directories in the project root.

For additional support, please contact the project maintainer.