Setup Guide

# Setting up the Graphiti

1. To start, connect your device to the Graphiti using a USB-A to Micro USB cable by connecting the USB-A to your device and the Micro USB cable to the Graphiti located on the left side of the Graphiti closest to Dot 3.
2. Next power the device on and press Space + Dot 8 to put the device into API mode. Alternatively, one can press Space + Dot 1 to put the device into Draw mode to enable the Draw API which includes Draw events.
3. Open the device manager from the Windows search bar in the taskbar by searching device manager
4. Under “Ports (COM & LPT)” identify the COM port the device has connected to which is “COM” followed by a number e.i. COM4 and note this for later

# Setting up your coding environment

1. Create a visual studio code project and install the “C/C++ Extension Pack” by Microsoft  
   A screenshot of a computer

   AI-generated content may be incorrect.
2. Next install the powershell extension “Powershell” by Microsoft

A screenshot of a computer

AI-generated content may be incorrect.

1. Now in the visual studio code terminal you can change your directory to the directory that you wish to install the Graphiti Library dependency of asio which is used for communicating with the device across the COM port (using the USB-A to Micro USB cable). You can do this by doing the command “cd C:\Users\<YourName>” and put your username as a replacement for <YourName> like “cd C:\Users\User”.
2. Now you can move the file vcpkg.ps1 found in the library to that directory and run it by doing “.\vckpg.ps1” (This script will fail if you do not have git installed)
3. After successfully running the ps1 file you can move it back to the project or delete it if you choose. If the file is not given to you, its contents are below. Which you can run as individual commands

git clone https://github.com/microsoft/vcpkg.git

.\vcpkg\bootstrap-vcpkg.bat

.\vcpkg\vcpkg integrate install

.\vcpkg\vcpkg install asio

1. Next add the directory that you installed vcpkg to your Path
   1. Do this by searching for “Edit environment variables for your account” in the Windows search bar in the taskbar
   2. Now if vcpkg was installed under a user such as “C:Users\<YourName>” edit “User variables for \_\_\_\_\_” by clicking “Path” then “Edit”
   3. Next press New and add your path to vcpkg as “C:\Users\<YourName>\vcpkg\
   4. Instead, you can also do this system wide with System variables by preference if not installed under your user
2. Now download the Graphiti Library called “graphiti” to “C:Users\<YourName>” or anywhere you wish keeping in mind its path and a following the same steps as in step 6 but instead adding the path “C:\Users\<YourName>\graphiti\bin”
3. Now add the include paths of your installations to the Include path of your C/C++ Configurations tool

C:\Users\<YourName>\graphiti\include

C:\Users\<YourName>\vcpkg\installed\x64-windows\include

${workspaceFolder}/\*\*

1. Now install a compiler and build system of your choice. I recommend using g++ from the MSYS2 GCC 15.1.0 toolchain as the compiler, and Ninja as the build system for fast and efficient builds
2. Now create a CMakeLists.txt file with the contents below:

cmake\_minimum\_required(VERSION 3.15)

set(CMAKE\_CXX\_STANDARD 23)

# Compiler g++

set(CMAKE\_CXX\_COMPILER "g++")

add\_definitions(-D\_WIN32\_WINNT=0x0601)

# Set vcpkg toolchain (adjust path as needed)

# Make sure vcpkg has asio

set(CMAKE\_TOOLCHAIN\_FILE "C:/Users/<YourName>/vcpkg/scripts/buildsystems/vcpkg.cmake")

# Set path for Graphit library

set(CMAKE\_PREFIX\_PATH "C:/Users/<YourName>//graphiti")

# Your project

project(RunGraphiti)

# Find asio for Graphiti

find\_package(asio REQUIRED)

# Find Graphiti

find\_package(Graphiti REQUIRED)

# Create Executable

add\_executable(main src/main.cpp)

# Link Executable to Graphiti Library

target\_link\_libraries(main PRIVATE Graphiti::Graphiti)

1. Be sure to replace the file paths of vcpkg and the Graphti library with the paths to your installations. Also place your cpp and hpp files in the add\_executable call to create an executable that is then linked to the Graphiti library on the following line.
2. Now you can clean, configure, build, and run your project using:

Remove-Item -Recurse -Force .\build\

cmake -G Ninja -B build -S .

ninja -C build

.\build\main.exe

# Writing a project

1. To include the graphiti library do

#include <Graphiti/Extension.hpp>

This includes functions like startUpVCP() for convenience but is not necessary

Or

#include <Graphiti/API.hpp>

#include <Graphiti/Connection/Connection\_VCP.hpp>

This includes all necessary API calls and Connection calls

1. Next call startUpVCP with your COM port identified in step four of “Setting up the Graphiti”, followed by whether you wish to enable or disable key and touch events using bool values. For example, startUpVCP(“COM4”, false, false) will disable both key and touch events
2. Next run any code you wish interacting with the graphiti api by doing

“graphiti->function” where “function” is any function call you wish.

1. Finally call shutDownVCP with the same boolean values followed by delete graphiti to delete the graphiti object.

Example code is given below:

#include <cstdio>

#include <Graphiti/Extension.hpp>

GraphitiExtension\* graphiti = new GraphitiExtension();

void sleepAndOutput(){

    graphiti->sleep(2);

    auto output = graphiti->getNextOutputEvent();

    if(output.has\_value()) {

        std::cout << output.value() << std::endl;

    } else {

        std::cout << "No value" << std::endl;

    }

}

int main() {

    bool keyEventsBool = false;

    bool touchEventsBool = false;

    std::cout << "Starting" << std::endl;

    graphiti = new GraphitiExtension();

    if(!graphiti->startUpVCP("COM4", keyEventsBool, touchEventsBool)){

        return 1;

    }

    graphiti->getDateAndTime();

    sleepAndOutput();

    graphiti->shutDownVCP(keyEventsBool, touchEventsBool);

    delete graphiti;

    std::cout << "Ending" << std::endl;

    return 0;

}