# Graphiti API Library

Daniel O’Connell

Copyright © 2025 [Orbit Research](https://www.orbitresearch.com/)

Version 1.0.1

Table of Contents

[Graphiti API Library 1](#_Toc206066229)

[Introduction 1](#_Toc206066230)

[Description 2](#_Toc206066231)

[Dependencies 2](#_Toc206066232)

[User Guide 2](#_Toc206066233)

[C++ Code Guide 3](#_Toc206066234)

[License 4](#_Toc206066235)

[Acknowledgements 4](#_Toc206066236)

### Introduction

The Graphiti API Library is an open-source library as we further our commitment to ground-breaking innovation and creative application of technology to achieve user-friendliness, affordability, and quality. Our purpose in creating this library is to improve the experience of people with disabilities by widening the application of the Graphiti. This library hopes to allow any operating system and common computer programming language to communicate with the device easily.

### Description

This library was designed and written in C++23 with the interests in mind of allowing the Graphiti to be accessible using several languages with strong usability and potential to be used in different languages and on different platforms. This library provides all API calls and capabilities that Graphiti has to offer. Currently the API supports C++, C, Python, and Java with VCP and HID communications on the windows operating system.

### Dependencies

This library uses Asio from Boost to communicate across USB VCP. This library also uses ws2\_32 to support Asio on Windows. The library also uses Hidapi to communicate across USB HID. Both asio and hidapi are added to the library using the vcpkg installation tool. This version of Asio has documentation at this link: [Asio C++ Library](https://think-async.com/Asio/). This version of hidapi is also available at this link: [hidapi: hidapi API](http://www.kubicz.engineer/hidapi/docs/group__API.html). And a link to vcpkg website is here: [vcpkg - Open source C/C++ dependency manager from Microsoft](https://vcpkg.io/en/)

### User Guide

The Graphiti Library comes with a Graphiti Setup Guide document in the documents folder to inform users on how to set up their development space. Additionally, the Graphiti Setup Guide has a guide on how to write a project for using the C++ library with the Graphiti, Ninja, and C++. In the library, there is a binding folder that contains each of the bindings for different programming languages. The folder jgraphiti contains the code to set up the Graphiti library with java and examplej shows code that can use the java code. The C folder under binding shows how to write a project using the C programming language. Finally, the python folder under binding shows how to write a project using the python programming language and the Graphiti library. Dependences can be locally added using the copy-files.ps1 scripts or be added to your include path on your IDE. If you wish to use this code as a reference, there is a C++ Code Guide below.

### C++ Code Guide

The library is divided into two major groups, being the API and Connection files. The API classes include API.cpp and Extension.cpp. The lib/src folder contains these two files.

The Connection files are Connection\_VCP and Connection\_HID which were made with the intention of separating libraries that handle different kinds of connections with the device. Connection\_VCP is intended to use Asio while Connection\_HID was made with the intention of using Hidapi.

The Extension class was made with methods to allow the library to be more easily used by using the Connection class and API class to reduce calls to set up a program while also having some extra methods that help with using the library.

The library also comes with a test harness used for testing the functions with the device. Many of the tests are functionality tests implying that their ability to pass or fail is determined by the user of the test. The passed and failed tests txt files hold all the passed and failed tests of any given run.

The API uses ThreadSafeQueue objects to store the events sent by the Graphiti which also has getNext() calls to take the latest kind of event out of that queue. To start listening to device responses call the methods startResponseThread() after setting up the connection. The Extension class and example code show this clearly.

### License

This project is licensed under the GNU General Public License v3.0 (GPL-3.0).

See the LICENSE file for full details.

This library incorporates and builds upon work by Ken Perry in the original Graphiti Python API library, which is also licensed under the GNU GPL.

### Acknowledgements

Big thanks to Ken Perry for their Graphiti API library and support.