# Audio Plugin Development with QtGrpc

A Journey Into Remote GUIs

Dennis Oberst

19 October 2023

**Abstract**

The Qt project is a powerful and versatile C++ framework widely used for developing cross-platform applications. Additionally, Qt is no stranger to the audio-industry, with a strong presence within professional audio companies. If we look into the expanding world of audio-plugins however, we would steer into the void when looking for Qt user-interfaces. With this work I will explain the reasons and problems that led to this state and show a new and innovative approach in solving the problems.

# Contents

# Chapter 1: Introduction

Throughout this work we will talk about plugins and how they affect the usabillity of an application. We will specifically explore the development of graphical user interfaces for audio plugins. We will inspect how it affects user experiences and explore their correlations with development experiences. When thinking about GUIs[1] in general it comes natural to spend some thoughts about their flexibillity and stability. There are more operating systems, graphic-backends, and platform specific details out there, than a single developer could handle to implement.

When we spent time learning and potentially mastering a skill, we often want it to be applicable to a variety of use-cases. Choosing a library that *has stood the test of time* would complement the stability, but we often don't want to re-learn a topic just because the API[2] of our skill set has changed.

Thus, it comes natural to think about Qt, a cross-platform framework for creating graphical user interfaces (GUIs) when considering the implementation of an audio plugin UI[3], that should be available on all the major platforms. The skill set we acquire in using the Qt framework is versatile and can be used to develop mobile, desktop or even embedded applications without the need to re-learn syntax or logic. One of the slogans of Qt applies here:

> Code once, deploy everywhere.

We can see the demand on this topic by simply searching for references on the forum "kvraudio.com", which is a well-known website to discuss audio related topics.

A short query of:

> "Qt" "Plugin" :site www.kvraudio.com

reveals 57.800 entries found. From which 580 are in the timeframe of the past year between 10/19/2022 - 10/19/2023. Although the meaningfullness of such numbers is questionable, it still shows the relevance and need of seeing Qt as an option for developing audio plugins.

## 1.1 Background

### 1.1.1 Plugins: Overview and Significance

**Plug-*ins*** in their most basic form extend the functionality of a plugin-loading-host dynamically. We could think of them as functionality that is made available *on-demand*. They are used all around the software and hardware world and can be found in a multitude of areas. Be it the extension of specialized filters for image processing applications like *Adobe Photoshop*, dynamically loadable drivers for operating systems like *GNU/Linux* (Vandevoorde, 2006) or operating system dependent features as used in .

- Define what plugins are and their role in software systems.
- Explain how plugins enhance the functionality and extensibility of software applications.
- Discuss the advantages of using plugins in the audio processing domain and its impact on the user experience.

[1] Graphical User Interfaces
[2] Application Programming Interface
[3] User Interface

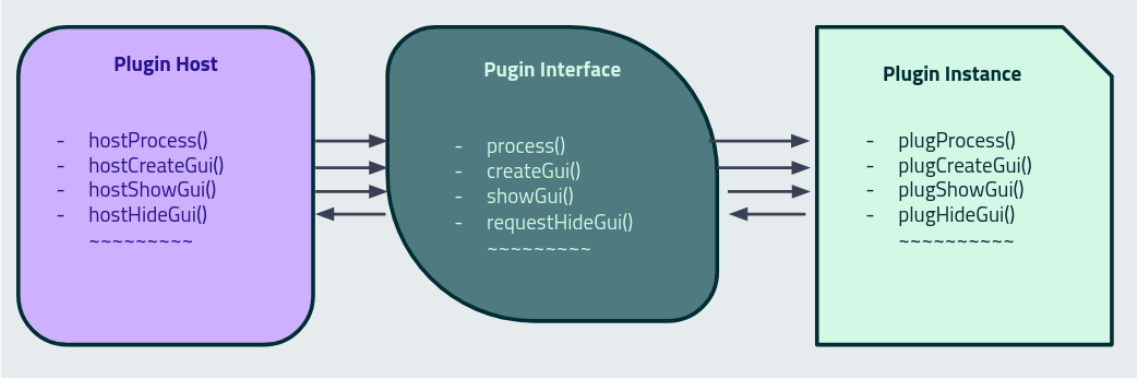


Figure 1: basic plugin architecture

## 1.2 Problem Statement

- Clearly state the research problem and the challenges associated with the integration process.
- No exec, because of problems with multuple instances
- Explain Windows, Linux and Mac behavior, Glib static presence
- QApplication, static environment, can
- Shared Library - non blocking

## 1.3 Objectives

- Outline the main objectives of the thesis, focusing on the integration of Qt and Clap.

## 1.4 Scope and Limitations

- Define the boundaries and extent of the research.
- Cross platform: Linux (wayland + xorg), windows and mac
- Identify any limitations or constraints that may affect the integration process.

# Chapter 2: The Qt Framework

- Provide a comprehensive introduction to the Qt framework, its history, and its key features.
- Discuss the advantages of using Qt for software development.

## 2.2 Qt Modules and Architecture, QtGRPC

- Explore the various modules and components of the Qt framework.
- Explain the architecture and design principles of Qt.

## 2.3 Qt APIs and Tools

- Discuss the different APIs and tools provided by Qt for application development.
- Highlight relevant APIs that will be utilized in the integration process.

# Chapter 3: The Clap Audio Plugin Format

## 3.1 Introduction to Clap

- Explain the purpose and significance of the Clap audio plugin format.
- Discuss its role in the audio processing industry.

## 3.2 Clap Plugin Structure

- Describe the structure and organization of Clap audio plugins.
- Explain the required components and their functionalities.
- Events, Realtime

## 3.3 Clap API and Specifications

- Explore the Clap API and its usage in developing audio plugins.
- Discuss the specifications and guidelines for creating Clap-compatible plugins.

## Chapter 3.5: Realtime, Events, Data Structures, Communications

## Chapter 4: Integrating Qt with Clap

### 4.1 Design Considerations

- Discuss the design principles and considerations for integrating Qt with Clap.
- Address any challenges or conflicts that may arise during the integration process.

### 4.2 Architecture and Implementation

- Propose an integration architecture that leverages the strengths of both Qt and Clap.
- Detail the implementation steps and techniques involved in the integration.

# Chapter 5: Experimental Results and Evaluation

## 5.1 Test Setup and Methodology

- Explain the experimental setup used for evaluating the integrated solution.
- Describe the methodology employed to measure the performance and effectiveness.

## 5.2 Results and Analysis

- Mention all problems: GLib event loop / no auto attach on linux
- Present the empirical results obtained from the experiments.
- Analyze and interpret the results in relation to the integration objectives.

# Chapter 6: Discussion and Conclusion

## 6.1 Summary of Findings

- Summarize the key findings and outcomes of the research.

## 6.2 Discussion of Results

- Discuss the implications and significance of the results obtained.
- Compare the integrated solution with existing approaches and discuss its advantages.

## 6.3 Contributions and Future Work

- Highlight the contributions of the thesis to the field of audio processing and software development.
- Identify potential areas for future research and improvement in the integration process.

## Bibliography

Vandevoorde, D. (2006, September 7). Plugins in c++. Wikibooks. retrieved 23.7.2023. Available at: https://www.open-std.org/JTC1/sc22/wg21/docs/papers/2006/n2074.pdf