МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

Факультет прикладной математики, информатики и механики

Кафедра математического обеспечения ЭВМ

Исследование возможности добавления палитры команд в произвольные приложения с использованием фреймворка Qt

Магистерская диссертация

Направление 01.04.02 Прикладная математика и информатика

Профиль Математическое и программное обеспечение вычислительных машин

Зав. кафедрой _____ д.т.н., проф. Г.В. Абрамов ___.___ 2020 г.

Обучающийся _____ Д.В. Польшаков

Руководитель _____ к.ф.-м.н., доц. М.К. Чернышов

Воронеж 2020

Список сокращений

БД — база данных

ПО — программное обеспечение

ГИП — графический интерфейс пользователя

API — application programming interface (программный интерфейс приложения)

Содержание

ВВЕДЕНИЕ

Из-за роста возможностей персональных компьютеров, программное обеспечение становится все сложнее и функциональнее. Такие группы программ как графические редакторы, браузеры, органайзеры и многое другое обрастают огромным числом функций. Для доступа к ним используются элементы ГИП.

Для быстрого доступа к функциям приложения, используются горячие клавиши. Они обычно создаются только для самых часто используемых команд. Остальные же действия приходится производить вручную через интерфейс.

Как бы хорошо ни был разработан интерфейс, число функций может оказаться настолько большим, что появляется проблема с поиском нужного элемента управления. Кроме того, некоторые системы поддерживают возможность добавления сторонних модулей. В таком случае место расположения элементов регулируется сторонними разработчиками, а не авторами оригинального приложения.

«Палитра команд» — технология, которую придумали для упрощения поиска элементов. Она представляет собой специальное окно в интерфейсе приложения, где показываются все доступные команды. Иногда рядом с ними отображается сочетание горячих клавиш, с помощью которых пользователь может её вызвать. Такой подход помогает пользователю легко запоминать новые сочетания, который он забыл или не знал раньше. В этом же окне есть поле для ввода поискового запроса.

Такая функциональность появилась в различных текстовых редакторах и средах разработки. Палитра команд оказалась достаточно удобной, поэтому позже энтузиасты разработали библиотеку с открытым исходным кодом Plotinus. Она позволяет добавлять такую функцию в приложения, которые используют фреймворк GTK. Данная библиотека опиралась на механизм, который предоставляет сами GTK, для расширения готовых приложений. Более подробный обзор решений, которые используются палитру команд произво-

дится в разделе 2.1.

Целью данной работы было реализовать механизм, который позволял бы добавлять палитру команд в сторонние приложения без их пересборки. Из-за того, что такой механизм для GTK уже существует, был выбран другой (не меньший по распространенности) фреймворк — Qt.

Для достижения поставленной цели, нужно было исследовать возможность добавления функций в уже собранную программу, разработать библиотеку для расширения произвольных приложений и программу для координации работы.

В первой главе данной работы рассматриваются существующие приложения, которые используют палитру команд, проводится анализ способов добавления функций в существующее приложение, формулируются задачи для реализации и описывается архитектура взаимодействия всех элементов разрабатываемой системы.

Во второй главе рассматриваются детали и средства реализации каждого из элементов системы, протокол взаимодействия их друг с другом, и вспомогательные средства, которые были разработаны для решения технических задач.

В третьей главе производится анализ полученного решения: сравнение производительности приложения с использованием решения и без него, рассматриваются возможности дальнейших улучшений.

ГЛАВА 1. ПОСТАНОВКА ЗАДАЧИ

Требуется разработать набор программ, которые в комплексе будут решать следующие задачи:

- запускать целевые приложения в специальном окружении;
- собирать информацию о существующих элементах графического приложения;
- сохранять информацию о всех запущенных приложениях;
- отображать пользователю окно для поиска и выбора элемента;
- активировать выбранный пользователем элемент.

ГЛАВА 2. АНАЛИЗ ЗАДАЧИ

2.1. Обзор существующих реализаций палитр команд

Впервые палитра команд появилась 1 июля 2011 году в редакторе Sublime Text 2 [4]. Вслед за этим подобная функция была реализована в некоторых других программах. Таких, как:

- Atom[5],
- VSCode[6],
- JupyterLab[7].

Но это были лишь единичные случаи. В апреле 2017 года появилась альфа-версия приложения Plotinus[8], которое позволяет добавлять палитру команд в любое приложение, использующее графическую библиотеку GTK.

Таким образом можно наблюдать, что частные решения начинают заменяться более универсальными. Однако на текущий момент эти решения не позволяют покрыть большинство областей т.к. ограничены лишь программами с GTK, который используется не более чем в половине прикладных приложений для ОС Linux и занимает совсем малую долю среди приложений для ОС Windows.

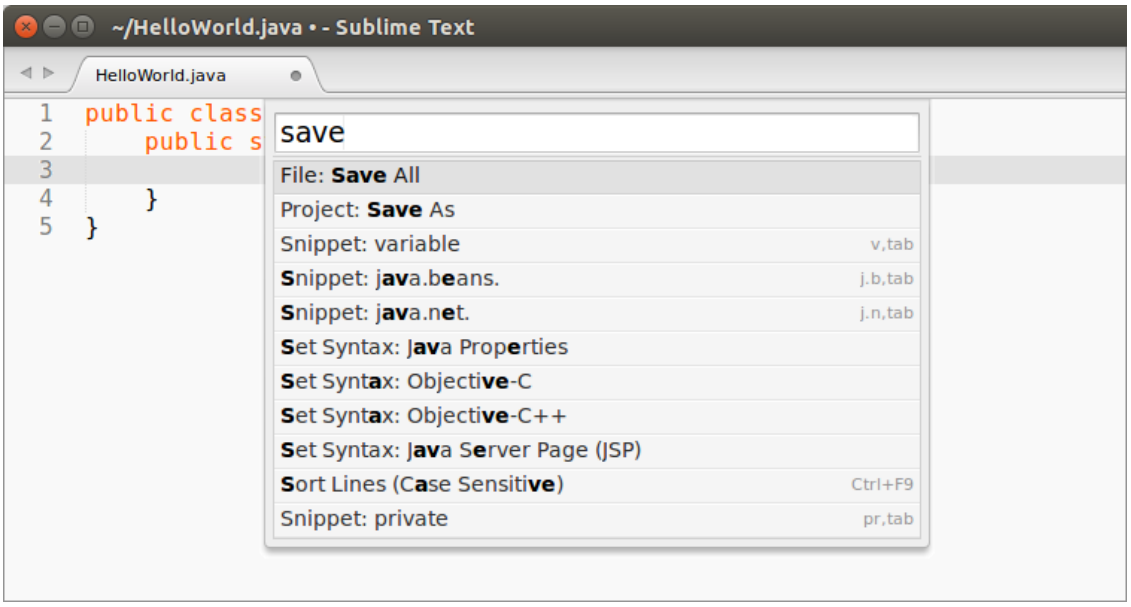

Рис. 2.1: Sublime Text

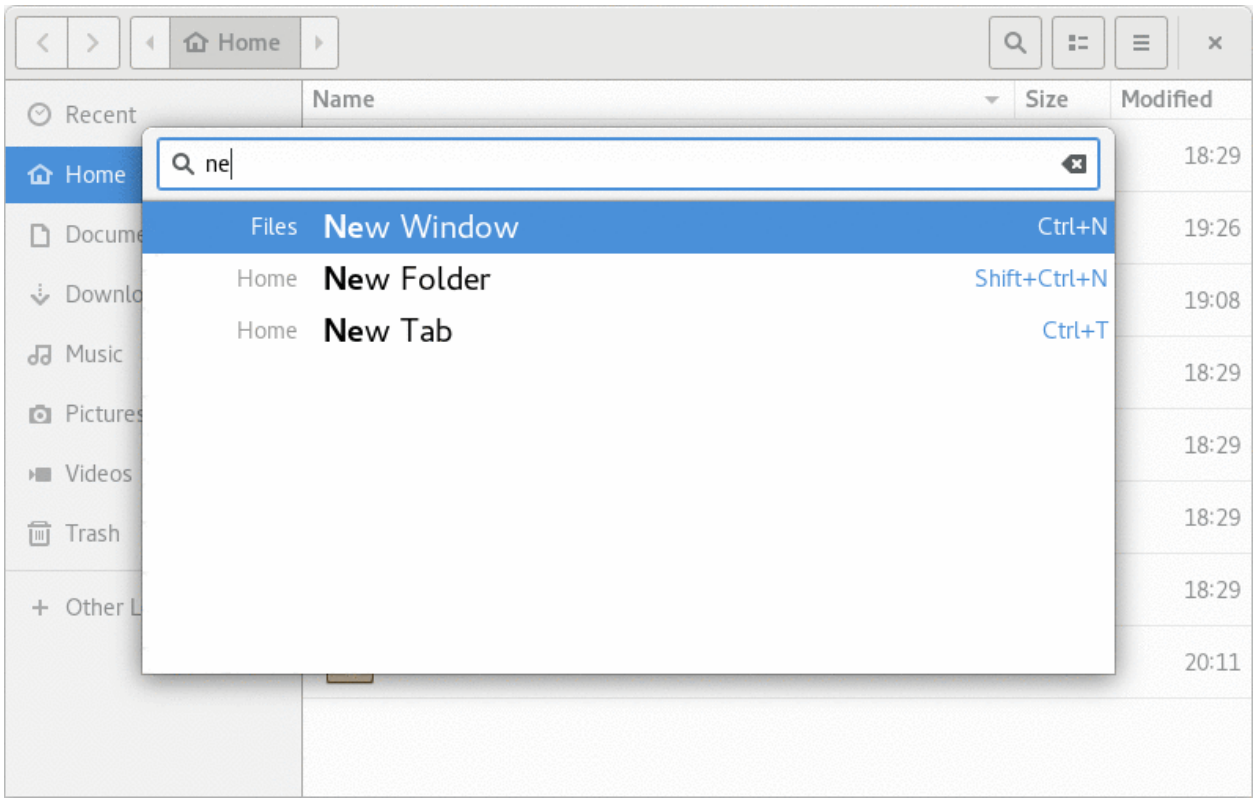

Рис. 2.2: Plotinus

2.2. О фреймворке Qt

Qt — это кроссплатформенный фреймворк для разработки программного обеспечения на языке программирования C++[3]. Он содержит множество библиотек для упрощения реализации прикладных задач. Благодаря кроссплатформенности данный фреймворк позволяет запускать написанное с его помощью ПО на многих операционных системах путем обычной сборки проекта без внесения изменений в исходный код самой программы. Отличительной особенностью Qt является наличие метаобъектного компилятора, который запускается в начале сборки и генерирует вспомогательный код. Такой подход позволил добавить частичную поддержку рефлексии.

Qt включает в себя следующие модули:

- Qt Core — базовые классы, обеспечивающие рефлексию, работу со строками, механизмы владения и т.п. Используются всеми остальными модулями;
- Qt Network — классы, позволяющие легко писать переносимый код для работы с сетью;
- Qt SQL — классы, предоставляющие удобный программный интерфейс для работы с различными реляционными БД;
- Qt Multimedia — классы, для работы с данными (аудио и видео), устройствами (камеры, микрофоны);
- Qt GUI — классы, позволяющие реализовать приложения с графическим интерфейсом.

В ОС Linux библиотеки GTK и Qt являются двумя наиболее популярными средствами для реализации приложений с графическим интерфейсом. Т.к. средство для добавления палитры команд уже есть для GTK, в данной работе будет рассмотрена такая возможность для Qt.

События в Qt — это объекты, унаследованные от абстрактного класса `QEvent`. Они представляют действия, произошедшие внутри приложения, либо созданные в результате активности пользователя, о которых приложение должно знать. События могут быть получены и обработаны любым эк-

земпляром подкласса `QObject`, но они особенно актуальны для графических элементов.

Обычно события доставляются объектам через вызов виртуальных функций. Если разработчик хочет заменить функцию базового класса, он должен реализовать всю обработку самостоятельно. Однако, если нужно только расширить функциональность то разработчик должен реализовать нужное расширение, а затем вызывать базовый класс, чтобы обеспечить поведение по умолчанию для тех случаев, которые он не хочет обрабатывать.

Не всегда в классе имеется нужная функция для события. Наиболее распространенный пример — обработка нажатия клавиш, для которой нет соответствующей виртуальной функции. Для обработки таких событий нужно переопределить метод `QObject::event()`. Он является общим обработчиком событий, и позволяет выполнить дополнительные действия до или после обработки по-умолчанию.

2.3. Организация работы приложения

Система управления должна позволять контролировать множество приложений. Для ее реализации лучше всего подходит клиент-серверная архитектура. Для каждого целевого приложения запускается отдельный клиент, который занимается сбором информации об элементах управления и передает ее на сервер.

В качестве сервера будет выступать приложение, которое запускает целевые приложения вместе с клиентами. После этого сервер принимает входящее соединение от клиента и отображающее окно поиска элемента для текущего активного окна.

Для удобной работы окно поиска должно отображаться окно поверх текущего активного приложения. Палитра команд является инструментом для помощи пользователю в поиске нужной функции. Он может не знать точное наименование команды, поэтому в приложении должна быть поддержка нечеткого поиска.

2.3.1. Добавление логики в стороннее приложение

Разработчики любого приложения не могут учесть желания и капризы всех пользователей. Благодаря этому приложения не превращаются в комбайны, которые невозможно было бы поддерживать. Разработчики могут убрать какую-то деталь, которая нужна малому проценту людей. Ведь надо тратить время на исправление ошибок в ней. А иногда эти специфичные функции могут даже замедлять работу всего приложения. В таком случае пользователи могут захотеть добавить какую-то дополнительную логику или функцию в основное приложение.

Подходы делятся на два типа:

- добавление функции на этапе сборки приложения;
- добавление функции в момент выполнения программы.

Целью данной работы является добавление функциональности в максимальную группу приложений. Первый же подход исключает такую возможность для приложений с закрытым исходным кодом. К тому же при первым