https://doi.org/10.26160/2474-5901-2023-35-65-73

КРОССПЛАТФОРМЕННОЕ ПРИЛОЖЕНИЕ ДЛЯ ОБУЧЕНИЯ РАЗРАБОТКЕ ГРАФИЧЕСКИХ ПОЛЬЗОВАТЕЛЬСКИХ ИНТЕРФЕЙСОВ С ПОМОЩЬЮ БИБЛИОТЕКИ QT

Коротков А.А., Букунов С.В.
Санкт-Петербургский государственный архитектурно-строительный университет, Санкт-Петербург, Россия

Ключевые слова: программирование, обучение, библиотека Qt, язык QML, кроссплатформенное приложение.
Аннотация. С помощью библиотеки Qt разработано кроссплатформенное приложение для обучения декларативному языку QML, широко используемому для создания приложений с графическим пользовательским интерфейсом. В приложении реализованы оригинальные авторские решения, позволяющие в удобной форме обучаться разработке графических пользовательских интерфейсов с помощью библиотеки Qt и языка QML.

CROSS-PLATFORM APPLICATION FOR LEARNING TO DEVELOP GRAPHICAL USER INTERFACES USING QT LIBRARY

Korotkov A.A., Bukunov S.V.
Saint-Petersburg State University of Architecture and Civil Engineering, Saint-Petersburg, Russia

Keywords: programming, learning, Qt library, QML language, cross-platform application.
Abstract. A cross-platform application for teaching the QML declarative language widely used to create applications with a graphical user interface has developed. The application implements original author's solutions that allow you to learn the development of graphical user interfaces using the Qt library and the QML language in a convenient way.

Введение

В настоящее время очень сложно переоценить роль графического интерфейса в различных компьютерных приложениях. Обычному пользователю уже недостаточно при запуске приложения видеть обычные серые кнопки в рамках. Ему нужны красочные компоненты с различными анимациями, в разных темах (ночной и дневной) и проч. И, что самое главное, важным свойством любого современного приложения должна быть его кроссплатформенность, т.е. возможность его использования на различных устройствах (мобильных устройствах, планшетах, персональных компьютерах и др.), управляемых различными операционными системами.

К одному из наиболее популярных средств разработки таких приложений относится кроссплатформенная библиотека Qt [1]. С помощью этой библиотеки можно создавать приложения с графическим пользовательским интерфейсом для различных прикладных задач [2, 3]. Для разработки пользовательских интерфейсов в библиотеке Qt имеется специальная среда разработки QtQuick, которая представляет собой набор

технологий для создания анимированных, динамических пользовательских интерфейсов нового поколения.

Цель данной работы заключается в создании простого и удобного приложения для обучения разработчиков приложений с графическим пользовательским интерфейсом технологии создания таких интерфейсов.

Используемые технологии

Для разработки приложения были выбраны высокопроизводительный язык программирования C++, кроссплатформенная библиотека Qt и декларативный язык программирования QML.

Язык C++ – компилируемый строго типизированный язык программирования общего назначения. Поддерживает разные парадигмы программирования: процедурную, обобщённую, функциональную; наибольшее внимание уделено поддержке объектно-ориентированного программирования [4].

Qt – фреймворк для разработки кроссплатформенного программного обеспечения на языке программирования C++ [1]. Данный фреймворк является отличным выбором для создания программного обеспечения. Основным его преимуществом является отсутствие необходимости в рутинной работе по переносу разработанного приложения на отдельную операционную систему, каждый раз учитывая нюансы платформы и внося правки в исходный код. Для этого достаточно просто скомпилировать приложение под нужную платформу, при этом не внося никаких изменений в исходный код.

QML (Qt Meta Language или Qt Modeling Language) – декларативный язык программирования, предназначенный для разработки графических пользовательских интерфейсов, базирующийся на языке программирования JavaScript и являющийся составной частью среды разработки QtQuick, входящей в состав библиотеки Qt [5].

Для работы с библиотекой Qt использовалась среда разработки QtCreator, которая предоставляет огромное количество инструментов для разработки приложений любых масштабов [6].

Проектирование приложения

Довольна большая часть успеха разработки приложения состоит в том, чтобы определиться с требованиями, которым должно соответствовать будущее приложение [7].

На этапе проектирования приложения к нему были описаны определенные требования:

– пользователь должен иметь возможность просмотреть список всех доступных для него видов обучения (теории и практики);

– пользователь должен иметь возможность отличать просмотренные уроки от остальных;

– пользователь должен иметь возможность изучать теорию и практику;

– у пользователя должна быть возможность выполнять практические задания.

Требования к практическим заданиям:

– должна быть возможность видеть результаты изменения кода «на лету», т.е. без перекомпиляции кода;

– должна быть предусмотрена возможность работы в удобном для пользователя текстовом редакторе;

– должна быть возможность сравнения результата работы и некоего образца, причем в двух вариантах: с использованием рамки (несовпадающие участки выделяются рамкой) и фона (несовпадающие участки должны остаться оригинальными, остальные - затемниться);

– должна быть возможность выбора размера сравниваемых участков;

– должна быть возможность просмотра авторского решения в удобном для пользователя редакторе.

Программная реализация приложения

Приложение реализовано на языке C++ в объектно-ориентированном стиле и состоит из совокупности классов.

Основные классы приложения:

– QmlLearnerEng – класс, в котором реализованы основные методы, необходимые для прохождения практик. В нем реализованы следующие методы:

o QVector<int> compareTwoImages(const QImage& lhs, const QImage& rhs, int level) – метод, отвечающий за сравнение созданного пользователем интерфейса и образца. Метод принимает в качестве аргументов два изображения и уровень разбиения области ошибки. Метод возвращает массив индексов областей, в которых есть ошибка;

o bool prepareToPractice(int id) – метод, отвечающий за подготовку приложения к определенной практике. Возвращает результат подготовки.

– LessonsModel – класс, в котором реализованы методы для работы со списками уроков, их сохранение и загрузка. Реализованные методы:

o bool loadLessons() – метод, отвечающий за загрузку всех доступных уроков с результатами их прохождения. Метод возвращает результат загрузки;

o void openLesson(int id) – метод, отвечающий за открытие практики или теории. Метод принимает id пройденного материала;

o void saveLessonResult(int id) – метод, отвечающий за сохранение результатов прохождения.

Результаты

Приложение представляет собой многостраничное приложение. Функциональность каждой части приложения реализована в отдельной вкладке. При запуске приложения пользователь видит главную страницу со списком доступных ему обучающих программ (теория и практика). При этом пройденные этапы обучения отмечены зелёным знаком.

Общий вид главной страницы приложения представлен на рисунке 1.

Рис. 1. Главная страница приложения

Выбор нужного пункта из общего списка заданий осуществляется с помощью двойного щелчка мышью по соответствующей вкладке. Например, для изучения теоретического материала пользователь может выбрать соответствующую вкладку в разделе «Теория». Для выполнения практического задания пользователь может выбрать соответствующую вкладку в разделе «Практика».

Вкладки с теоретическим материалом

При изучении теории пользователю предлагается теоретический материал по выбранной теме. После изучения материала пользователь может вернуться в основное меню. В том случае, если материал был прочитан полностью, он считается пройденным, о чем будет сигнализировать зеленый знак напротив соответствующего пункта меню.

Пример внешнего вида страницы с теоретическим материалом представлен на рисунке 2.

Вкладки с практическими заданиями

При открытии урока с практическим заданием пользователь видит окно, представленное на рисунке 3. Область окна разделена на две части. Левая часть предназначена для отображения результата работы кода, написанного пользователем. В правой части окна отображается образец, к которому пользователь должен стремиться.

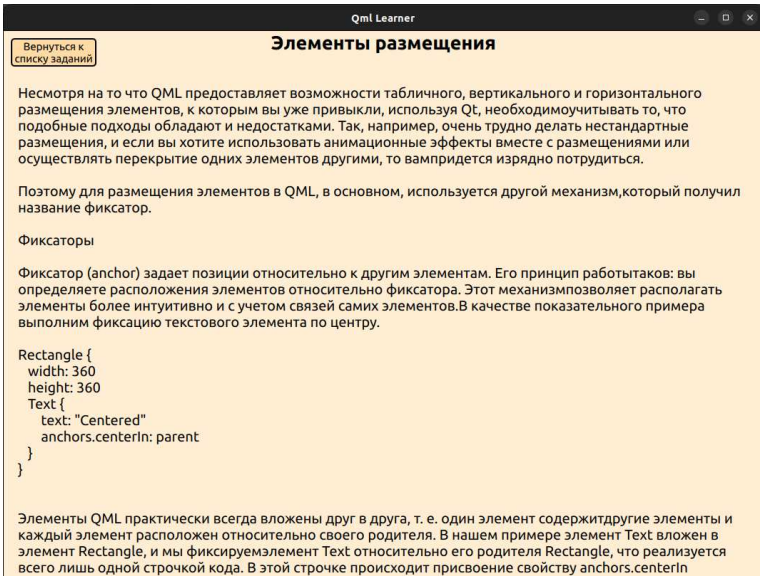

Рис. 2. Внешний вид страницы с теоретическим материалом

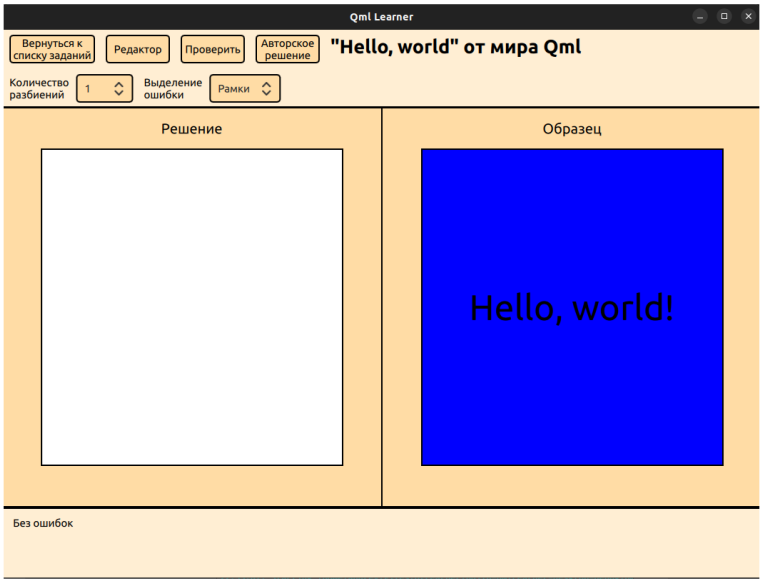

Рис. 3. Внешний вид окна для выполнения практического задания

В верхней части окна располагаются элементы управления процессом обучения. В частности,

– кнопка «Вернуться к списку заданий» возвращает пользователя к списку заданий;

– при нажатии на кнопку «Редактор», открывается редактор, который у пользователя установлен по умолчанию для файлов формата .qml;

– при нажатии на кнопку «Проверить» происходит сравнение решения пользователя и образца;

– при нажатии кнопки «Авторское решение» открывается редактор с кодом, который написал автор, чтобы сделать образец;

– количество разбиений – это список из значений, на которые необходимо разбить картинку для проверки;

– выделение ошибки – это список, который отвечает за вид отображения ошибки (рамка или фон).

В нижней части окна находится консоль для вывода ошибок, которые найдены в коде пользователя.

Описание процесса выполнения практического задания

Диаграмма процесса выполнения практического задания продемонстрирована на рисунке 4.

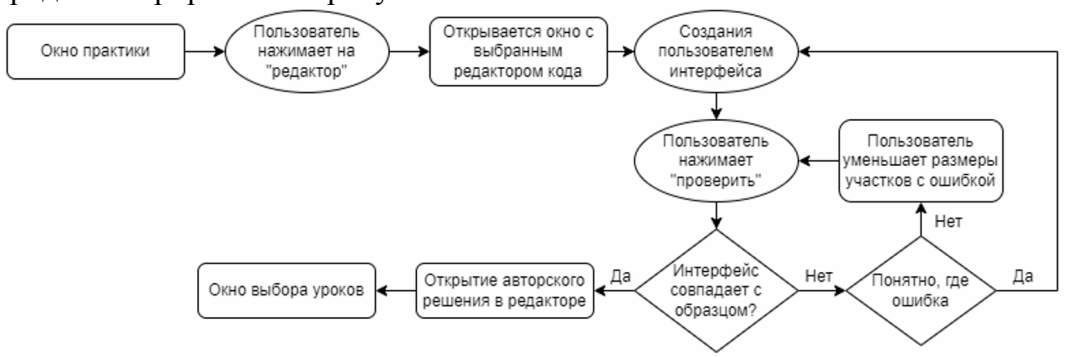

Рис. 4. Диаграмма процесса выполнения практического задания

При выполнении практического задания пользователь должен с помощью кнопки «Редактор» открыть редактор кода. При этом открывается редактор, который у пользователя назначен по умолчанию для файлов формата qml.

При этом при сохранении файла, в окне «Решение» сразу появляется результат работы созданного кода, если в нем не обнаружены ошибки. Если же код был написан с ошибками, то они будут выведены в консоль в нижней части окна. Пример вывода сообщения об ошибке представлен на рисунке 5.

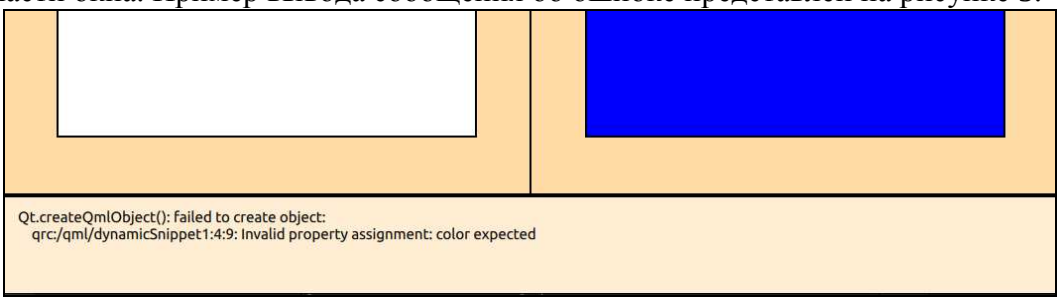

Рис. 5. Вывод ошибки в пользовательском коде

После того, как пользователь решит, что созданное окно полностью соответствует образцу, он должен проверить результат. Если образец и полученное решение полностью совпадают, то пользователю выдается соответствующее сообщение. После получения сообщения урок считается пройденным. Пример успешного выполнения практического задания представлен на рисунке 6.

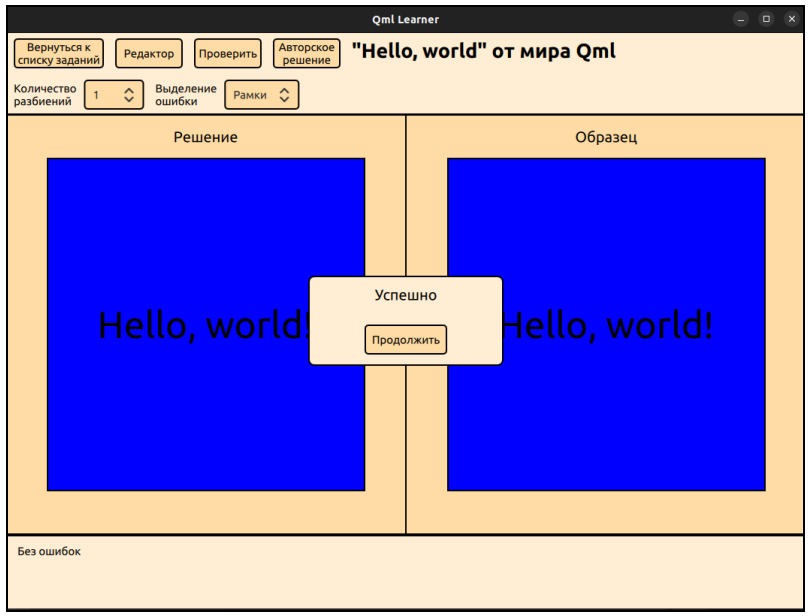

Рис. 6. Пример успешного выполнения практического задания

В случае неправильного выполнения практического задания пользователю выводится соответствующие сообщение и выделяется область, которая не совпадает с образцом. При этом всё изображение разбивается на равные части, количество которых задается пользователем в поле «Количество разбиений» в верхней части окна. Само выделение различается в зависимости от заданных настроек. В режиме «рамка» неверные части выделяются рамкой (рис. 7). В режиме «фон» совпадающие части затемняются, а фон несовпадающих частей остается неизменным (рис. 8).

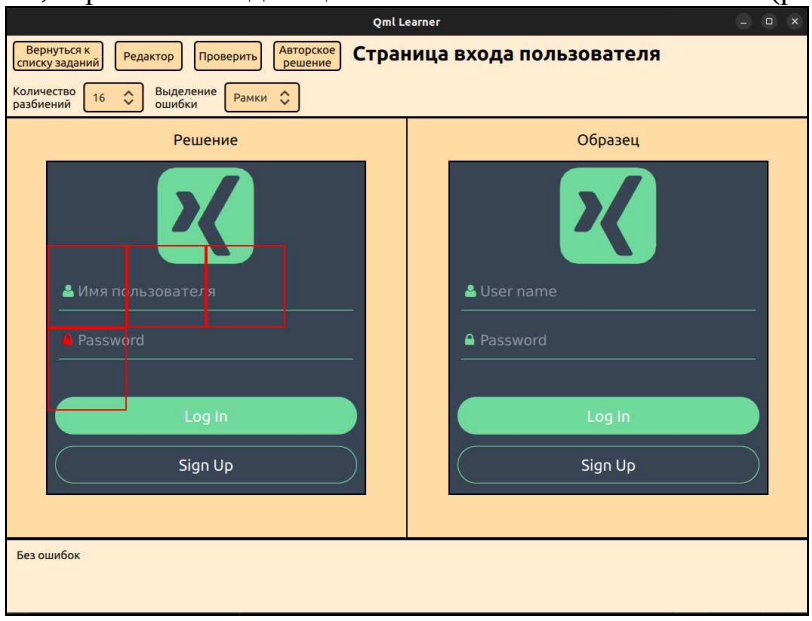

Рис. 7. Пример выделения ошибочных участков с помощью рамки

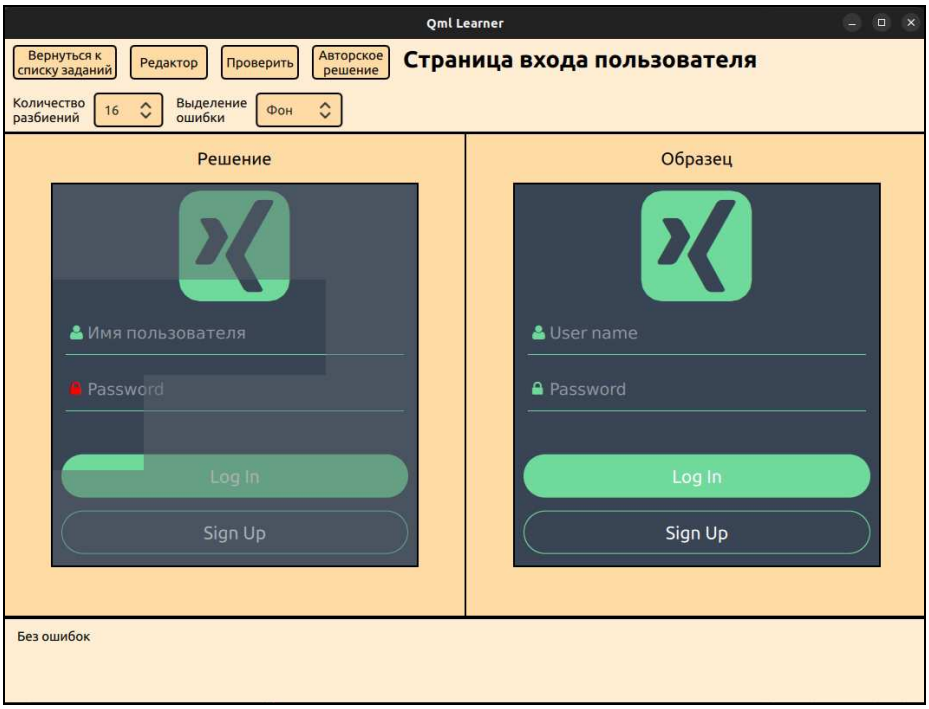

Рис. 8. Пример фонового выделения ошибочных участков

При этом важно отметить, что решение и образец сравниваются именно как две картинки, а не путем сравнения программного кода. Такой подход позволяет пользователю использовать любые известные ему инструменты и подходы к реализации пользовательского интерфейса, не думая о том, какой способ реализации данного дизайна использовал автор.

Заключение

С помощью библиотеки Qt и языка программирования C++ разработано приложение, предназначенное для обучения созданию графических пользовательских интерфейсов с использованием декларативного языка QML. Отличительной особенностью приложения является возможность визуального просмотра результата работы измененного программного кода без его перекомпиляции. Использование такого подхода позволяет существенно ускорить процесс разработки пользовательского интерфейса. Новизна используемого подхода заключается еще и в том, что для оценки правильности решения практического задания используется не сравнение программных кодов, реализующих заданный интерфейс, а сравнение самих интерфейсов, полученных с помощью этих кодов. В этом случае главная задача разработчика интерфейса заключается не в написании программного кода, максимально соответствующего коду, с помощью которого создается интерфейс-образец, а в максимальном соответствии разработанного интерфейса интерфейсу-образцу. Сам программный код (его структура, используемые алгоритмы и проч.) в данном случае является вторичным.

Список литературы

1. Шлее М. Qt 5.10 Профессиональное программирование на C++. – СПб.: БХВ-Петербург, 2020. – 1052 с.
2. Баланков Н.В., Букунов С.В. Разработка кроссплатформенного приложения с помощью библиотеки Qt на примере графического векторного редактора // Journal of Advanced Research in Technical Science. – 2020. – №5. – С. 79-89.
3. Курбанов Н.В., Букунов С.В. Разработка кроссплатформенного приложения для мониторинга финансового состояния кредитных организаций с помощью библиотеки Qt // Южно-Сибирский научный вестник. – 2022. – №5. – С. 145-151.
4. Лафоре Р. Объектно-ориентированное программирование в C++. – СПб.: Питер, 2018. – 928 с.
5. Qt 6 QML [Электронный ресурс]. – URL: https://www.qt.io/product/qt6/qml-book.
6. Алексеев Е.Р., Злобин Г.Г., Костюк Д.А., Чеснокова О.В., Чмыхало А.С. Программирование на языке C++ в среде Qt Creator. – М.: ALT Linux, 2015. – 448 с.
7. Файлер М. Шаблоны корпоративных приложений. – М.: Диалектика-Вильямс, 2018. – 544 с.

References

1. Shlee M. Qt 5.3. Professional programming in C++. – SPb.: BHV-Peterburg, 2015. – 928 p.
2. Balankov N.V., Bukunov S.V. Developing a cross-platform application using Qt library on the example of a graphical vector editor // Journal of Advanced Research in Technical Science. 2020, no. 5, pp. 79-89.
3. Kurbanov N.A., Bukunov S.V. Development of a cross-platform application for monitoring the financial state of credit institution using Qt library // South-Siberian Scientific Bulletin. 2022, no. 5, pp. 145-151.
4. Lafore R. Object-oriented programming in C++. – SPb.: Piter, 2018. – 928 p.
5. Qt 6 QML [Electronic resource]. – URL: https://www.qt.io/product/qt6/qml-book.
6. Alekseev E.R., Zlobin G.G., Kostyuk D.A., Chesnokova O.V., Chmyhalo A.S. C++ programming in the Qt Creator. – M.: ALT Linux, 2015. – 448 p.
7. Fauler M. Enterprise application templates. – M.: Dialectics-Willjams. 2018. – 544 p.

Коротков Андрей Александрович – магистрант	**Korotkov Andrey Aleksandrovich** – graduate student
Букунов Сергей Витальевич – кандидат технических наук, доцент кафедры информационных технологий	**Bukunov Sergei Vitalievich** – candidate of technical sciences, associate professor of Information Technology Department
sergeybukunov@yandex.ru	

Received 25.04.2023