МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования   
**«Национальный исследовательский   
Нижегородский государственный университет им. Н.И. Лобачевского»**

**(ННГУ)**

**Институт информационных технологий, математики и механики**

Направление подготовки: «Программная инженерия»

**ОТЧЁТ**

по лабораторной работе №2

**КЛИКЕР**

**Выполнил:** студент группы

382008-1

\_\_\_А.Д. Кандрин \_\_\_\_\_\_\_

Подпись

**Проверил:** младший научный сотрудник

\_\_\_\_Д.А. Карчков\_\_\_\_\_\_\_\_

Подпись

Нижний Новгород  
2023 г.

## Содержание

[1. Постановка задачи 3](#_Toc27873363)

[2. ХОД РАБОТЫ](#_Toc27873364) 5

[3. ЗАКЛЮЧЕНИЕ](#_Toc27873365) 9

[4. ЛИТЕРАТУРА](#_Toc27873374) 10

## Постановка задачи.

Цель: освоить процесс создания нового проекта, изучить его структуру, научиться запускать приложение. Посмотреть возможности среды Qt Creator и эмулятора.

Шаги:

1. Создать новый проект со стандартной заготовкой приложения.
2. Посмотреть содержимое вкладок Qt creator (Welcome,Edit, Debug, Projects, Sailfish OS, Help). Выяснить назначение каждой из них.
3. Изучить структуру созданного проекта (каталоги, расположение файлов).
4. Выяснить соглашения по размещению файлов для проектов Qt для Sailfish OS.
5. Изучить содержимое \*.pro файла проекта. Выяснить назначение разделов файла. Документация по файлам проекта доступна по адресу <http://doc.qt.io/qt-5/qmake-project-files.html>
6. Изучить содержимое \*.qml файлов. Выяснить назначение элементов, используемых в файле, с помощью интерактивной справки (нажать на элемент в файле, затем вызвать справку нажатием на кнопку F1 на клавиатуре).
7. Запустить эмулятор, освоить принципы навигации в Sailfish OS, посмотреть возможности и настройки эмулятора. Научиться осуществлять навигацию на устройстве Sailfish OS, узнать возможности настроек устройства (приложение Settings).
8. Собрать и запустить заготовку приложения на эмуляторе
9. Используя материал слайдов 35 и 39 из лекции, изменить приложение таким образом, чтобы оно содержало одно текстовое поле со счётчиком и одну кнопку, позволяющую увеличивать значение счётчика на 1. Размещению элементов на экране внимания можно не уделять.
10. Собрать и запустить приложение на эмуляторе. Убедиться в правильности его работы.

## Ход работы.

1. Создадим новый проект со стандартной заготовкой приложения.

Для этого нажмем Ctrl+N и в открывшемся окне выберем шаблон “Aurora Os Qt Quick Application”. В следующем окне указываем название проекта и путь, по которому он будет храниться. В следующем окне выбираем комплект сборки, название которого оканчивается на “i486”. Во всех остальных окнах нажимаем “Далее” и в последнем окне – “Завершить”. После этого откроется созданный проект.

1. На левой панели Qt Creator есть следующие вкладки:
2. Начало. Здесь находятся кнопки доступа к справочной информации Qt Creator (“Примеры”, “Учебники”, “Начать сейчас”), а также кнопки для работы с проектом – “Новый проект” и “Открыть проект”. Также здесь отображаются последние проекты и сессии.
3. Редактор. Здесь на левой панели находится дерево панели, посередине – текстовый редактор, в котором можно редактировать исходный код, а снизу – панели с информацией (“Проблемы”, “Результаты поиска”, “Вывод приложения”, “Консоль сборки” и т.д.).
4. Далее располагается заблокированная вкладка “Дизайн”, используемая для редактирования форм.
5. Отладка. Эта вкладка используется во время отладки для просмотра отладочной информации.
6. Проекты. В этой вкладке можно посмотреть и отредактировать настройки проекта, сборки и запуска.
7. Справка. В этой вкладке можно посмотреть справочную информацию по Qt Creator, Qt, QML и некоторым другим технологиям.
8. Рассмотрим структуру созданного проекта.

Файл с расширением “.pro” описывает структуру проекта для системы сборки qmake.  
Каталог “icons” содержит значки приложений для разных разрешений экрана.  
Каталог “qml” содержит исходный код QML и ресурсы пользовательского интерфейса.  
Каталог “cover” содержит реализации обложек приложений.  
Каталог “pages” содержит страницы приложения.  
Файл “.qml” обеспечивает реализацию окна приложения.  
Каталог “rpm” содержит настройки сборки пакета rpm. Файл “.spec” используется утилитой rpmbuild, генерируется из файла “.yaml”.  
Каталог “src” содержит исходный код C++. Файл main.cpp является точкой входа в приложение.  
Каталог “translations” содержит файлы перевода пользовательского интерфейса.  
Файл “.desktop” определяет отображение и параметры запуска приложения.

1. Рассмотрим содержимое \*.pro файла проекта.

Проектный файл содержит всю необходимую информацию для того, чтобы qmake смог выполнить сборку проекта.  
Рассмотрим, за что отвечают отдельные переменные в этом файле:

1. TARGET – определяет имя целевого файла (например, имя исполняемого модуля).
2. CONFIG – указывает опции и параметры, с которыми должен быть сконфигурирован проект.
3. SOURCES – указывает файлы с исходным кодом.
4. DISTFILES – указывает список файлов, которые должны быть включены в цель dist.
5. SAILFISHAPP\_ICONS – размеры иконок для приложения sailfishapp.
6. TRANSLATIONS – указывает файлы с переводами текста на элементах пользовательского интерфейса.
7. Изучим содержимое \*.qml файлов.  
   Основной .qml файл лежит в папке “qml”. В нем описывается “ApplicationWindow” – компонент верхнего приложения Sailfish. Указаны его свойства initialPage – начальная страница, cover – обложка, allowedOrientations – набор доступных ориентаций для приложения.  
   В папке “pages” лежат страницы. В них описывается “Page” – одиночная страница внутри приложения. У страницы есть “id” – идентификатор, также – “allowedOrientations” и компонент “SilicaFlickable” – перелистываемый вид, внутри которого описаны “PullDownMenu” – меню (для получения которого нужно потянуть вниз) и “Column” – содержимое страницы.  
   В “Column” указан “PageHeader” – заголовок страницы и могут быть указаны другие компоненты. В сгенерированном проекте в нём присутствует “Label” – текстовое поле со строкой.
8. Запустим эмулятор, разблокируем устройство, потянув вправо или влево. Теперь, потянув влево или вправо можно перейти на экран с уведомлениями и виджетами (например, погоды). Если теперь потянуть вниз, то откроется малая панель, где можно включить/выключить вай-фай, включить авиарежим, сделать заметку, открыть камеру или выполнить некоторые другие действия. Если потянуть вверх, то откроется панель с приложениями. Среди приложений есть приложение “Settings” – с помощью которого можно управлять настройками устройства. Свернуть приложение можно, потянув вправо в верхнем левом углу.
9. Соберем и запустим приложение, сгенерированное по умолчанию. Отобразится страница с текстом “Hello Sailors” и заголовком “UI Template”.
10. Изменим приложение таким образом, чтобы оно содержало одно текстовое поле со счётчиком и одну кнопку, позволяющую увеличивать значение счётчика на 1.  
    Для этого удалим SecondPage.qml, CoverPage.qml, раздел “PullDownMenu” из “SilicaFlickable” и свойство “cover” из “ApplicationWindow” за ненадобностью. В разделе “Column” единственной оставшейся страницы отредактируем содержимоe “Label” и добавим “Button” так, чтобы получилось следующее:

Button

{

text: qsTr("Increase value")

width: parent.width

onClicked: {

label.count++;

}

}

Label {

id: label

property int count: 0

x: Theme.horizontalPageMargin

text: count

color: Theme.secondaryHighlightColor

font.pixelSize: Theme.fontSizeExtraLarge

}

1. Соберем и запустим приложение на эмуляторе. При нажатии на кнопку счетчик увеличивает своё значение.
2. **Заключение.**

Мы рассмотрели некоторые возможности Aurora IDE, запустили эмулятор Sailfish OS и рассмотрели его интерфейс, а также исследовали шаблон приложения, идущий в комплекте с Aurora SDK, и на его основе написали своё приложение-кликер.

**7. Литература.**

1. Документация Qt: сайт. — URL: https://doc.qt.io (дата обращения: 01.03.2022). — Текст: электронный.