Министерство науки и высшего образования Российской Федерации

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

Институт информационных технологий, математики и механики

Отчет по практическому заданию №8

«Инструменты разработки мобильных приложений»

Выполнил:
студент группы 381908-4
Трофимов В. А.

Нижний Новгород
2022

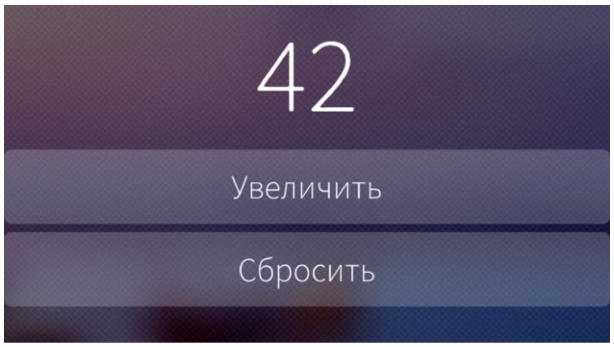
Оглавление

Цель задачи

В данном лабораторной работе требуется научиться использовать C++ классы в QML, научиться писать собственные QML компоненты на языке C++ и использовать их в приложении.

Постановка задачи

1. Создать класс-счётчик с полем для хранения текущего значения и методами для увеличения значения на единицу и сброса до нуля.
2. Использовать мета-объект класса-счётчика для создания объекта и вызова его методов (использовать функцию main, результат изменения состояния проверять выводом на консоль).
3. Создать приложение с текстовым полем и двумя кнопками. Использовать класс-счётчик в QML: текстовое поле должно отображать текущее значение счётчика, кнопки используются для увеличения значения счётчика на единицу и сброса значения до нуля.



4. Сделать поле со значением счётчика свойством и инициализировать его каким-либо значением при создании объекта в QML.
5. Создать класс, содержащий список из строк. Класс должен содержать методы для добавления строки в список и удаления последней добавленной строки.
6. Создать приложение, позволяющее добавить введённое слово и удалить последнее добавленное с использованием данного класса в QML. Слова сохраняются в нижнем регистре.
7. Реализовать свойство только для чтения, которое позволяет получить список всех строк в виде одной, перечисленных через запятую и использовать это свойство для вывода добавленных строк на экран. Свойство должно моментально реагировать на изменение содержимого списка, первое слово начинается с заглавной буквы.

Описание программной реализации

1. Класс-счётчик
2. Использование мета-объект класса-счётчика

```
Counter {
    id: counter
    count: 10;
}
```

3. Приложение с текстовым полем и двумя кнопками. Использование класс-счётчика в QML

```
Column {
    id: column
    width: parent.width
    anchors.horizontalCenter: parent.horizontalCenter
    spacing: 20

    PageHeader {
        title: qsTr("Счетчик")
    }

    Label {
        id: label;
        anchors.horizontalCenter: parent.horizontalCenter
        text: counter.getCount();
        color: Theme.secondaryHighlightColor
        font.pixelSize: Theme.fontSizeExtraLarge
    }

    Button {
        anchors.horizontalCenter: parent.horizontalCenter
        text: "Увеличить"
        onClicked: {
            counter.inc();
            label.text = counter.getCount();
        }
    }

    Button {
        anchors.horizontalCenter: parent.horizontalCenter
        text: "Обнулить"
        onClicked: {
            counter.reset();
            label.text = counter.getCount();
        }
    }
}
```

4. Поле со значением счётчика как свойство

```
Q_PROPERTY(int count READ getCount WRITE setCount NOTIFY countChanged)
```

5. Класс, содержащий список из строк

```
class StringList : public QObject
{
    Q_OBJECT
public:
    explicit StringList(QObject *parent = nullptr);
    Q_INVOKABLE void add(QString temp) { m_data << temp; };
    Q_INVOKABLE void popBack()
    {
        if (!m_data.isEmpty()) {
            m_data.pop_back();
        }
    };

    Q_INVOKABLE QString getAll()
    {
        QString temp;
        for (int i = 0; i < m_data.length(); i++)
        {
            if (i == 0) {
                QString t = m_data[i];
                t[0] = t[0].toUpper();
                temp += t;
            } else {
                temp += m_data[i].toLower();
            }

            if (i != m_data.length()-1){
                temp += ", ";
            }
        }

        return temp;
    };
private:
    QList<QString> m_data;
};
```

6. Приложение, использующее класс-список

```
StringList {
    id: stringList
}
```

7. Свойство только для чтения, которое позволяет получить список всех строк в виде одной

```
Q_INVOKABLE QString getAll()
{
    QString temp;
    for (int i = 0; i < m_data.length(); i++)
    {
        if (i == 0) {
```

```
            QString t = m_data[i];
            t[0] = t[0].toUpper();
            temp += t;
        } else {
            temp += m_data[i].toLower();
        }

        if (i != m_data.length()-1){
            temp += ", ";
        }
    }
    return temp;
};
```

Руководство пользователя

После запуска программы пользователь видит страницу с первым заданием и кнопку “Next”, которая перейдет на следующую страницу. В первом задании мы видим счетчик с кнопками для увеличения и обнуления.

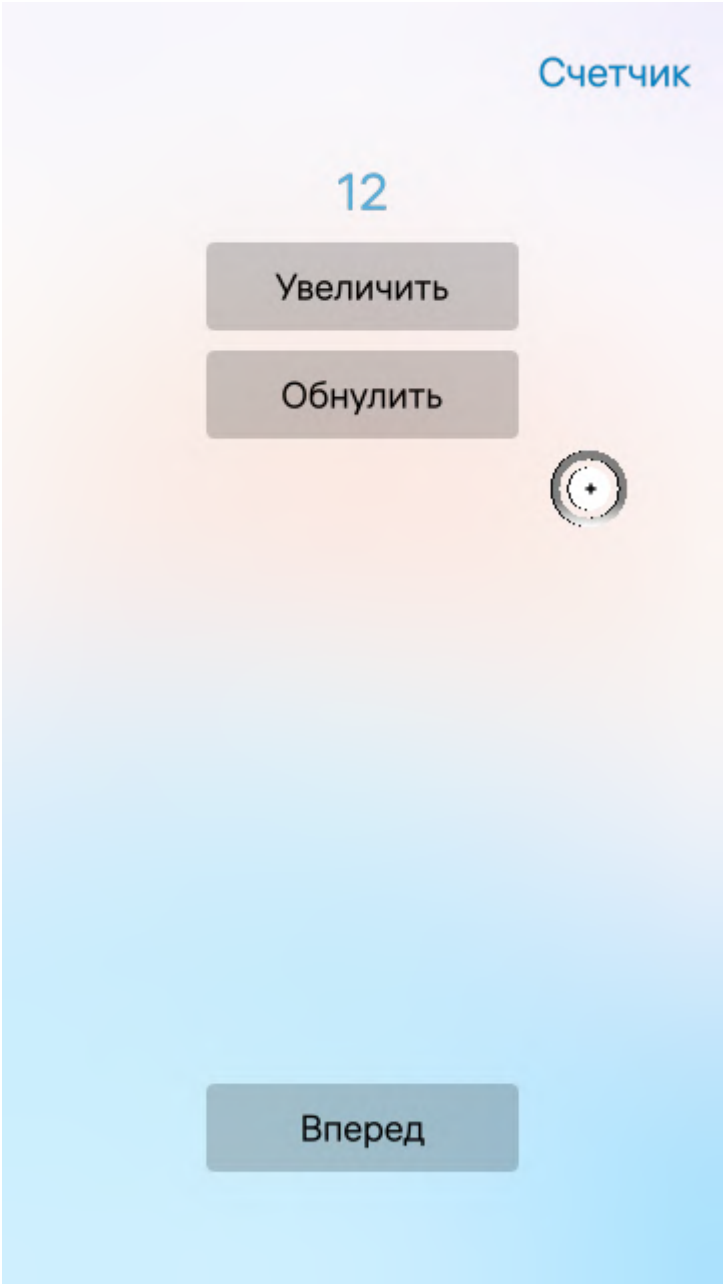


Рис. 1: Страница со счетчиком

Во втором задании находится текстовое поле и кнопки для добавления и удаления слов. Добавленные слова образуют строку слов, разделенных запятой.

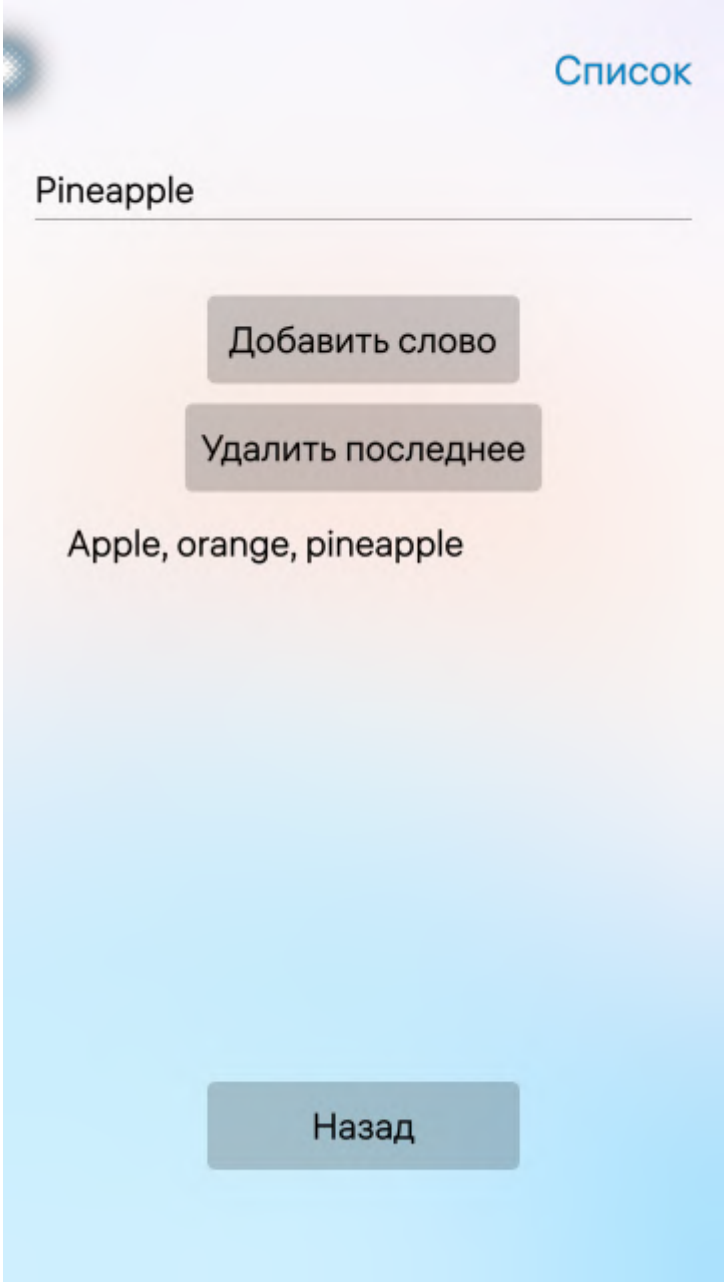


Рис. 2: Страница со списком слов

Заключение

В процессе выполнения данной лабораторной работы мы научились использовать C++ классы в QML, научиться писать собственные QML компоненты на языке C++ и использовать их в приложении.

Приложение

counter.h

```
#ifndef COUNTER_H
#define COUNTER_H

#include <QObject>
#include <QString>

class Counter : public QObject
{
    Q_OBJECT
    Q_PROPERTY(int count READ getCount WRITE setCount NOTIFY countChanged)
public:
    explicit Counter(QObject *parent = nullptr);
    Q_INVOKABLE int getCount() { return m_count; };
    Q_INVOKABLE void inc() { m_count++; };
    Q_INVOKABLE void reset() { m_count = 0; };

    void setCount(const int temp) { m_count = temp; emit countChanged(); };

signals:
    void countChanged();


private:
    int m_count = 0;
};

#endif // COUNTER_H
```

StringList.h

```
#ifndef STRINGLIST_H
#define STRINGLIST_H

#include <QObject>
#include <QString>

class StringList : public QObject
{
    Q_OBJECT
public:
    explicit StringList(QObject *parent = nullptr);
    Q_INVOKABLE void add(QString temp) { m_data << temp; };
    Q_INVOKABLE void popBack()
```

```
    {
        if (!m_data.isEmpty()) {
            m_data.pop_back();
        }
    };

    Q_INVOKABLE QString getAll()
    {
        QString temp;
        for (int i = 0; i < m_data.length(); i++)
        {
            if (i == 0) {
                QString t = m_data[i];
                t[0] = t[0].toUpper();
                temp += t;
            } else {
                temp += m_data[i].toLower();
            }

            if (i != m_data.length()-1){
                temp += ", ";
            }
        }

        return temp;
    };
private:
    QList<QString> m_data;
};

#endif // STRINGLIST_H
```

main.cpp

```
/****************************************************************************
**
** Copyright (C) 2022 Open Mobile Platform LLC.
** Contact: https://community.omprussia.ru/open-source
**
** This file is part of the Aurora OS Application Template project.
**
** Redistribution and use in source and binary forms,
** with or without modification, are permitted provided
** that the following conditions are met:
**
** * Redistributions of source code must retain the above copyright notice,
**   this list of conditions and the following disclaimer.
** * Redistributions in binary form must reproduce the above copyright notice,
```

```
**   this list of conditions and the following disclaimer
**   in the documentation and/or other materials provided with the distribution.
** * Neither the name of the copyright holder nor the names of its contributors
**   may be used to endorse or promote products derived from this software
**   without specific prior written permission.
**
** THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
** AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
** THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
** FOR A PARTICULAR PURPOSE ARE DISCLAIMED.
** IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
** FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY,
** OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
** PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
** LOSS OF USE, DATA, OR PROFITS;
** OR BUSINESS INTERRUPTION)
** HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,
** WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
** (INCLUDING NEGLIGENCE OR OTHERWISE)
** ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE,
** EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
**
****************************************************************************/

#include <QScopedPointer>
#include <QGuiApplication>
#include <QQuickView>
#include "Counter.h"
#include "StringList.h"

#include <sailfishapp.h>

int main(int argc, char *argv[])
{
    QScopedPointer<QGuiApplication> application(SailfishApp::application(argc, argv))
    application->setOrganizationName(QStringLiteral("ru.auroraos"));
    application->setApplicationName(QStringLiteral("Lab7"));

    qmlRegisterType<Counter>("com.counter", 1, 0, "Counter");
    qmlRegisterType<StringList>("com.stringlist", 1, 0, "StringList");

    QScopedPointer<QQuickView> view(SailfishApp::createView());
    view->setSource(SailfishApp::pathTo(QStringLiteral("qml/Lab7.qml")));
    view->show();

    return application->exec();
}
```

Page1.qml

```
import QtQuick 2.0
import Sailfish.Silica 1.0
import com.counter 1.0

Page {
    id: page
    allowedOrientations: Orientation.All

    SilicaFlickable {
        anchors.fill: parent

        Counter {
            id: counter
            count: 10;
        }

        Column {
            id: column
            width: parent.width
            anchors.horizontalCenter: parent.horizontalCenter
            spacing: 20

            PageHeader {
                title: qsTr("Счетчик")
            }

            Label {
                id: label;
                anchors.horizontalCenter: parent.horizontalCenter
                text: counter.getCount();
                color: Theme.secondaryHighlightColor
                font.pixelSize: Theme.fontSizeExtraLarge
            }

            Button {
                anchors.horizontalCenter: parent.horizontalCenter
                text: "Увеличить"
                onClicked: {
                    counter.inc();
                    label.text = counter.getCount();
                }
            }

            Button {
                anchors.horizontalCenter: parent.horizontalCenter
                text: "Обнулить"
                onClicked: {
                    counter.reset();
```

```
                    label.text = counter.getCount();
                }
            }
        }

        Button {
            anchors.horizontalCenter: parent.horizontalCenter
            y: parent.height - 200
            text: "Вперед"
            onClicked: pageStack.push(Qt.resolvedUrl(qsTr("Page2.qml")))
        }
    }
}
```

Page2.qml

```
import QtQuick 2.0
import Sailfish.Silica 1.0
import com.stringlist 1.0

Page {
    id: page
    allowedOrientations: Orientation.All
    SilicaFlickable {
        anchors.fill: parent

        StringList {
            id: stringList
        }

        Column {
            id: column
            width: parent.width
            anchors.horizontalCenter: parent.horizontalCenter
            spacing: 20

            PageHeader {
                title: qsTr("Список")
            }

            TextField {
                id: textField;
                placeholderText: "Введите слово"
            }
            Button {
                text: "Добавить слово"
                onClicked: {
                    stringList.add(textField.text)
```

```
                label.text = stringList.getAll();
            }
            anchors.horizontalCenter: parent.horizontalCenter
        }
        Button {
            text: "Удалить последнее"
            onClicked: {
                stringList.popBack();
                label.text = stringList.getAll();
            }
            anchors.horizontalCenter: parent.horizontalCenter
        }
        TextField {
            id: label;
            width: parent.width;
            x: Theme.horizontalPageMargin
            text: stringList.getAll()
            readOnly: true;
        }
    }

    Button {
        anchors.horizontalCenter: parent.horizontalCenter
        y: parent.height - 200
        text: "Назад"
        onClicked: pageStack.pop()
    }
  }
}
```