

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ НЕФТИ И ГАЗА
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)
ИМЕНИ И.М. ГУБКИНА»

ФАКУЛЬТЕТ КОМПЛЕКСНОЙ БЕЗОПАСНОСТИ ТЭК
КАФЕДРА БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Лабораторная работа №13

по дисциплине «Специализированные языки и технологии
программирования»
на тему «Sailfish Silica»

Выполнил студент:
группы КА-22-06
Воронин Алексей Дмитриевич

Преподаватель:
Греков Владимир Сергеевич

Москва, 2025

Оглавление	
Цель работы	3
Задание 1: Управление кнопкой	4
Задание 2: Диалог с вычислениями.....	4
Задание для самостоятельной работы.....	4
ЗАКЛЮЧЕНИЕ	9
Контрольные вопросы	10

Цель работы

- Освоить базовые принципы работы с кнопками и обработки их состояний
- Научиться создавать и использовать диалоговые окна
- Получить навыки обработки пользовательского ввода и базовых вычислений

Задание 1: Управление кнопкой

- Создать кнопку с текстом "Нажми меня"
- Реализовать текстовое поле для отображения состояния
- Настроить обновление текста при изменении состояния кнопки

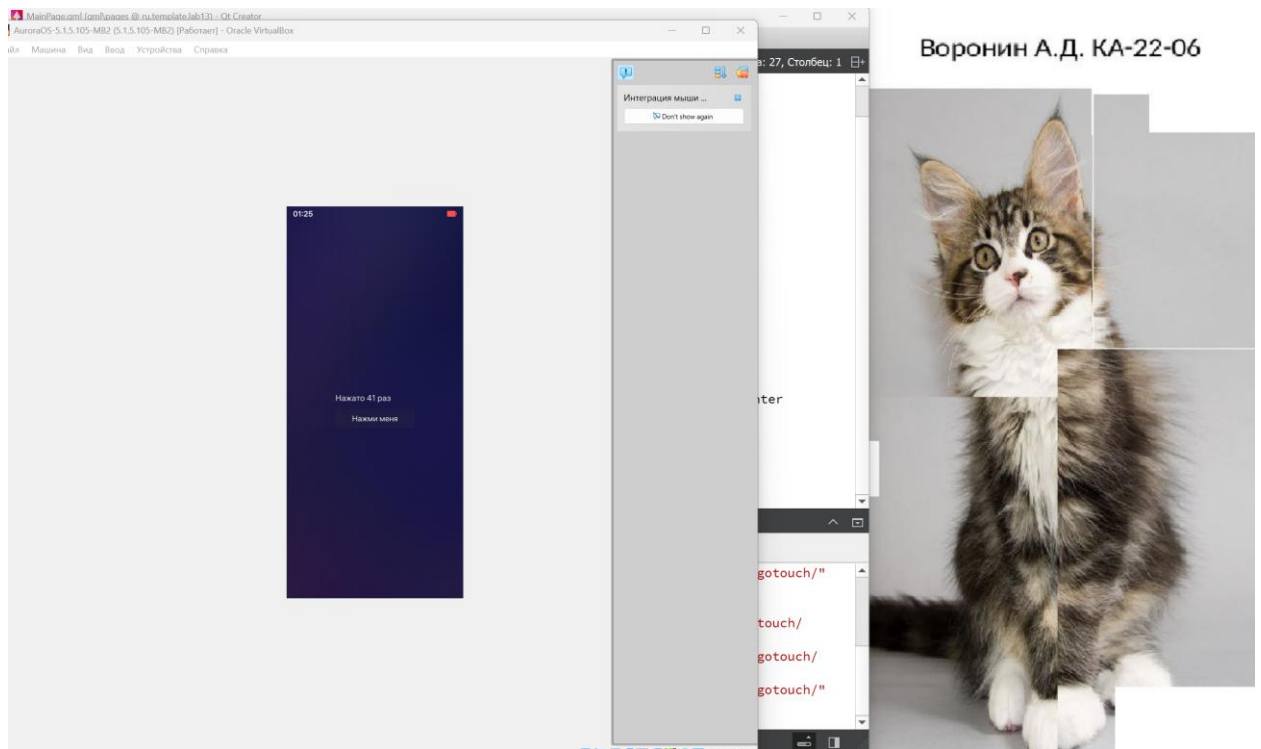


Рисунок 1 - Кнопка с текстом "Нажми меня"

Задание 2: Диалог с вычислениями

- Создать диалоговое окно с двумя полями ввода
- Добавить кнопку подтверждения "Подтвердить"
- Реализовать вычисление суммы введенных чисел

Задание для самостоятельной работы

В рамках самостоятельной работы необходимо добавить проверку на ввод только числовых значений, а также выводить историю в отдельном списке. Для этого в MainPage.qml добавить следующее:

```
import QtQuick 2.0
import Sailfish.Silica 1.0

ApplicationWindow {
    id: app
    initialPage: mainPage
    allowedOrientations: Orientation.All
    property int clickCounter: 0
```

```

// История как ListModel
ListModel {
    id: historyModel
}

Page {
    id: mainPage

    //property int clickCounter: 0

    Column {
        anchors.centerIn: parent
        spacing: Theme.paddingLarge
        width: parent.width - 2 * Theme.paddingLarge

        // Счётчик нажатий
        Label {
            id: counterLabel
            text: "Нажато " + clickCounter + " раз"
            font.pixelSize: 30
            horizontalAlignment: Text.AlignHCenter
            anchors.horizontalCenter: parent.horizontalCenter
        }

        Button {
            text: "Нажми меня"
            anchors.horizontalCenter: parent.horizontalCenter
            height: 70
            onClicked: clickCounter += 1
        }

        // Пример сложения
        Row {
            spacing: Theme.paddingLarge
            anchors.horizontalCenter: parent.horizontalCenter

            TextField {
                id: firstNumber
                width: 160
                height: 120
                placeholderText: "0"
                inputMethodHints: Qt.ImhFormattedNumbersOnly
                horizontalAlignment: Text.AlignHCenter
            }

            Label {
                text: "+"
                font.pixelSize: 30
                verticalAlignment: Text.AlignVCenter
            }

            TextField {
                id: secondNumber
                width: 160
                height: 120
                placeholderText: "0"
                inputMethodHints: Qt.ImhFormattedNumbersOnly
                horizontalAlignment: Text.AlignHCenter
            }

            Label {
                text: "="
                font.pixelSize: 30
            }
        }
    }
}

```

```

        verticalAlignment: Text.AlignVCenter
    }

    TextField {
        id: resultField
        width: 180
        height: 120
        placeholderText: "Ответ"
        readOnly: true
        horizontalAlignment: Text.AlignHCenter
    }
}

Button {
    text: "Подтвердить"
    anchors.horizontalCenter: parent.horizontalCenter
    height: 70
    onClicked: {
        var num1 = parseFloat(firstNumber.text)
        var num2 = parseFloat(secondNumber.text)
        if (isNaN(num1) || isNaN(num2)) {
            resultField.text = "Ошибка"
        } else {
            var res = (num1 + num2).toFixed(2)
            resultField.text = res
            // добавляем в историю
            historyModel.append({
                example: firstNumber.text + " + " +
secondNumber.text + " = " + res
            })
        }
    }
}

Button {
    text: "История"
    anchors.horizontalCenter: parent.horizontalCenter
    height: 70
    onClicked: pageStack.push(historyPage)
}
}

Page {
    id: historyPage

    SilicaListView {
        anchors.fill: parent
        model: historyModel
        header: PageHeader { title: "История" }

        delegate: ListItem {
            Label {
                text: example
                anchors.verticalCenter: parent.verticalCenter
            }
        }
    }
}
}

```

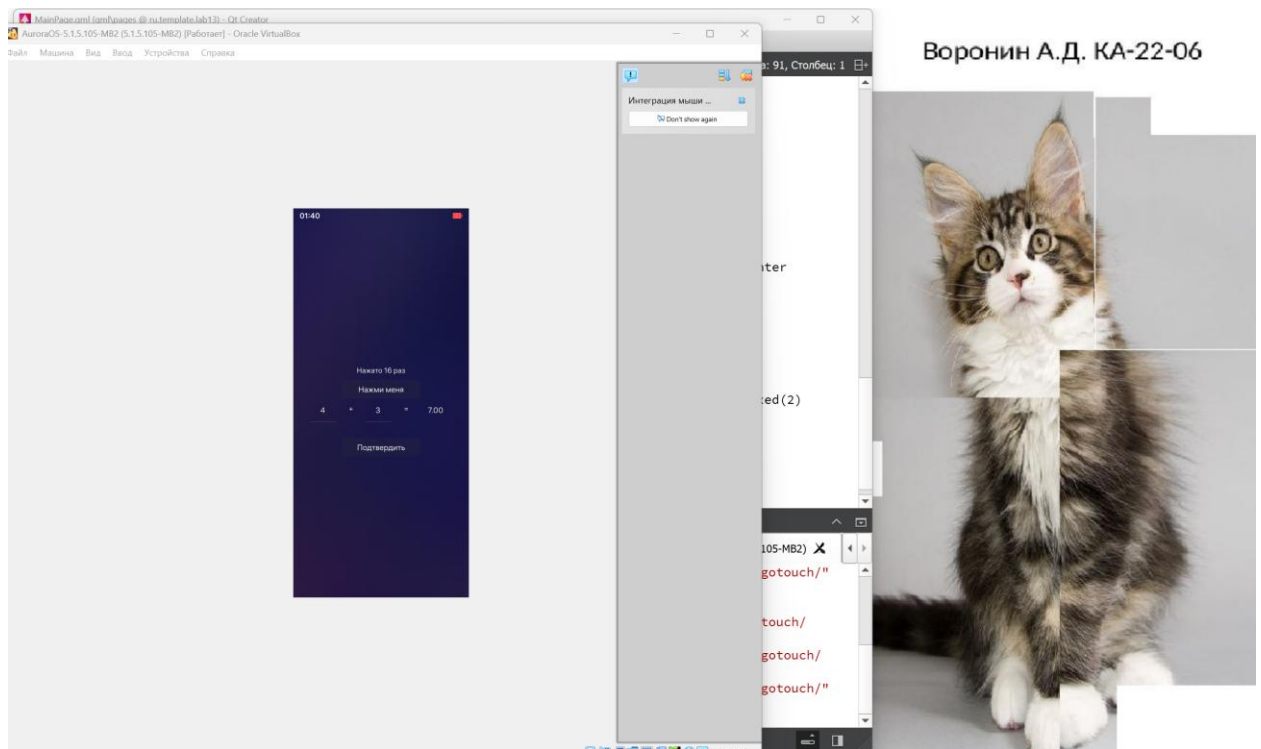


Рисунок 2 – Итоговое приложение

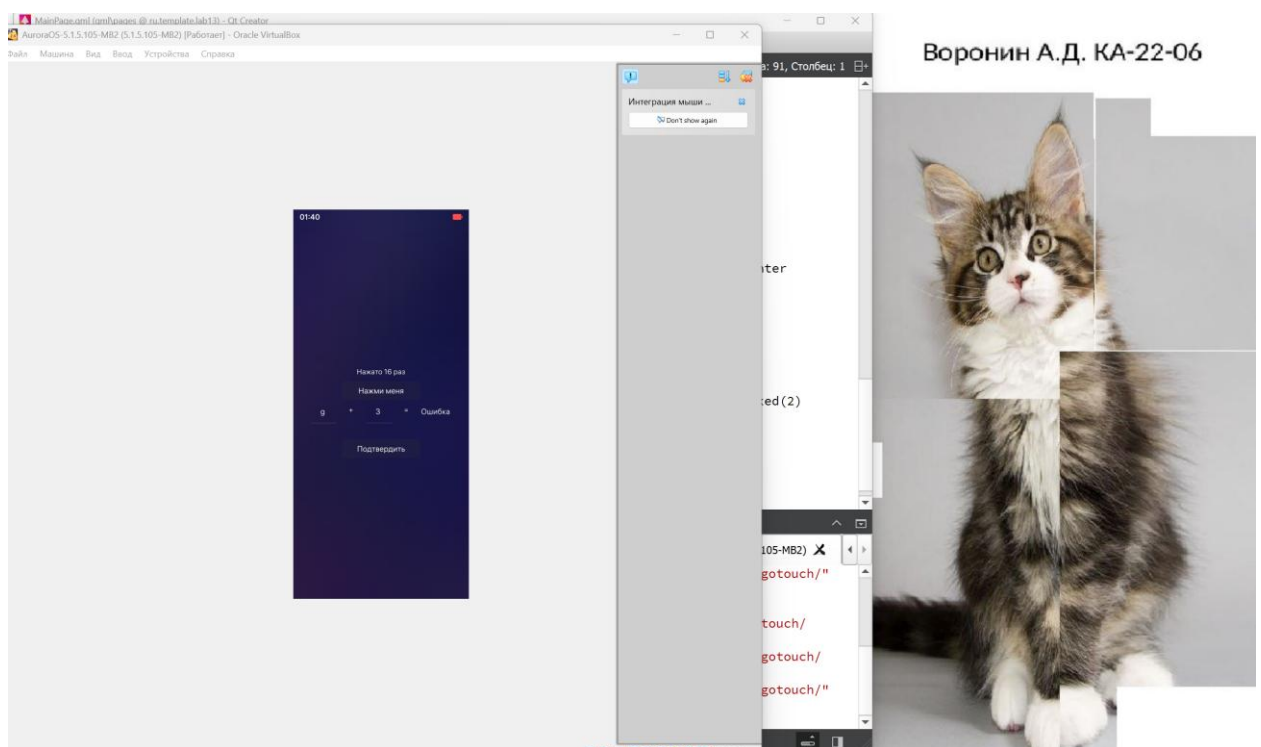
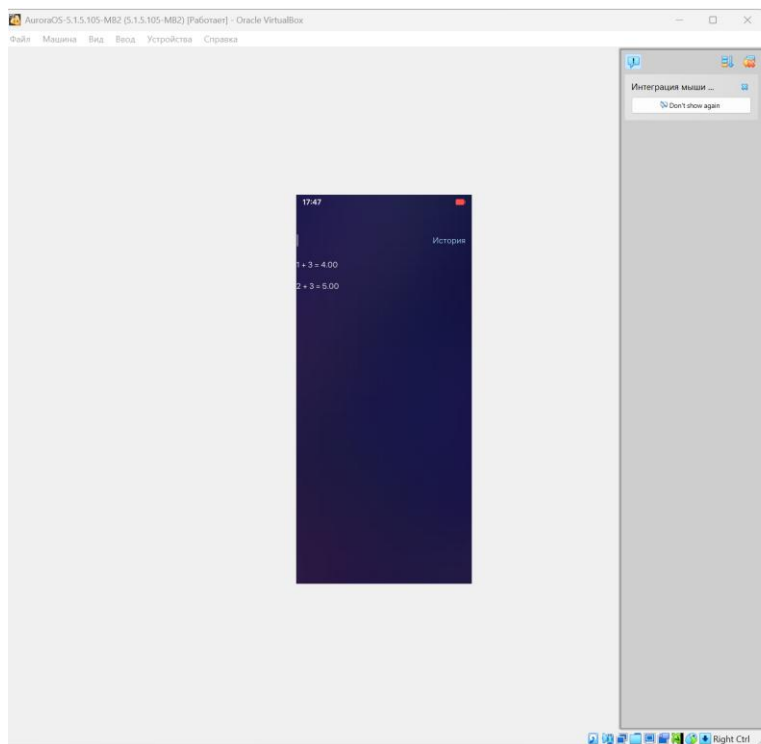


Рисунок 3 – Обработка нечисловых значений



Воронин А.Д., КА-22-06



Рисунок 4 – История

ЗАКЛЮЧЕНИЕ

В заключение, подводя итог проделанной работе, все поставленные в начале цели и задания данной лабораторной работы были достигнуты в полном объеме.

Контрольные вопросы

1. Какие основные состояния кнопки существуют?

Основные состояния кнопки в Qt Quick:

- Нормальное (normal) - стандартное состояние кнопки, когда она не нажата и не выделена.
- Нажатое (pressed) - состояние, когда кнопку удерживают нажатой.
- Фокус (focus) - когда кнопка выделена и готова к взаимодействию с клавиатуры.
- Наведение (hovered) - когда курсор мыши находится над кнопкой.
- Выбранное (checked) - для кнопок с включённым свойством checkable, состояние переключателя (включено/выключено).
- Отключённое (disabled) - когда кнопка недоступна для взаимодействия.

2. Как преобразовать строку в число в QML?

- `parseInt(string)`
- `parseFloat(string)`
- `var n = Number("42.5")`

3. Какие компоненты используются для создания диалогового окна?

`Dialog` – базовый контейнер для диалогового окна.

`DialogHeader` – стандартная шапка диалога.

4. Как организовать обработку нажатия кнопки?

В компоненте `Button` добавить `onClicked`.

nClicked: `pageStack.push(historyPage)`