

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 №2
«Инструменты разработки мобильных приложений»

Выполнила:
студентка группы 381906-1
Тырина А. К.

Нижегород
2022

Оглавление

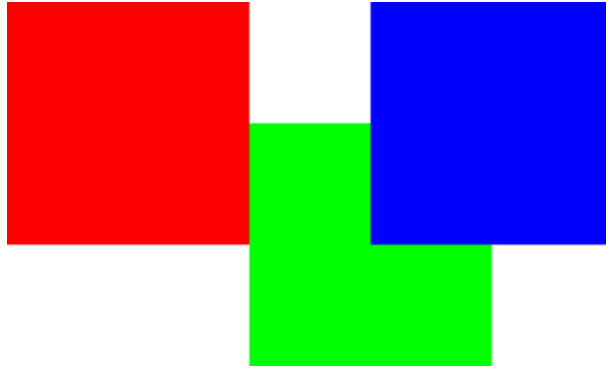
| | |
|---|----|
| Цель задачи | 3 |
| Постановка задачи | 4 |
| Описание программной реализации | 6 |
| Руководство пользователя | 8 |
| Заключение | 13 |
| Приложение | 14 |

Цель задачи

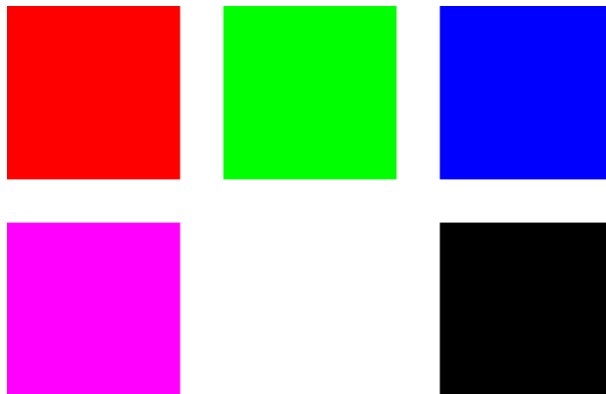
В данном лабораторной работе требуется освоить базовые навыки построения пользовательских интерфейсов, позиционирования, отрисовки и перемещения элементов. Научиться анимировать элементы. Научиться создавать диалоги и взаимодействовать с ними.

Постановка задачи

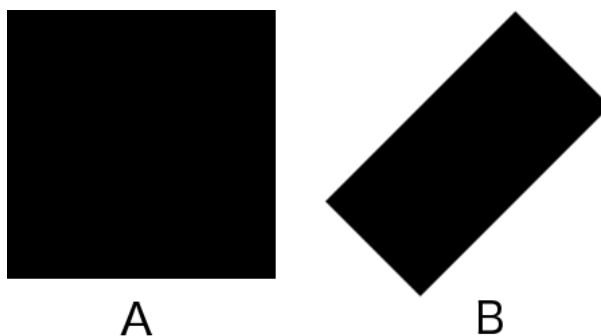
1. Создать новый проект со стандартной заготовкой приложения.
2. Нарисовать 3 квадрата красного, зелёного и синего цветов следующим образом:



3. Поместить текст "Квадрат" белого цвета по центру синего квадрата.
4. Нарисовать 5 квадратов с использованием Column и Row следующим образом:

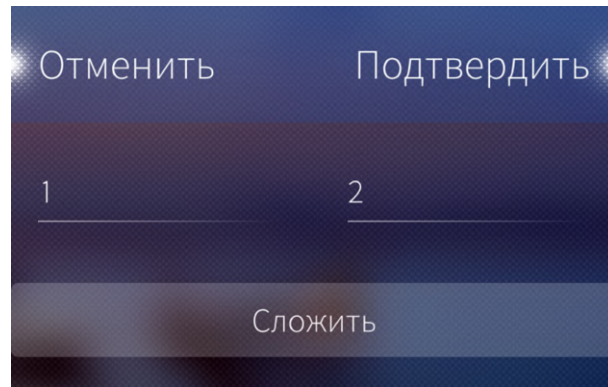


5. Нарисовать те же 5 квадратов с использованием Grid.
6. Сделать из квадрата "А" прямоугольник "В" с использованием объектов Translate, Scale и Rotation:



7. Нарисовать квадрат и анимировать его перемещение вниз с увеличением его размера. Документация по анимации доступна по адресу <http://doc.qt.io/qt-5/qml-qtquick-animation.html>

8. Реализовать диалог с двумя текстовыми полями, в которые вводятся числа. После нажатия на кнопку "Подтвердить" в консоль выводится сумма чисел. Для преобразования строк к числам использовать функцию `parseInt("42")`. Валидацией и обработкой ошибок можно пренебречь.



Отменить Подтвердить

1 2

Сложить

Описание программной реализации

Для того, чтобы нарисовать 3 квадрата красного, зелёного и синего цветов и поместить текст "Квадрат" белого цвета по центру одного из них, использовались следующие визуальные элементы:

Элемент Rectangle

- **id** – идентификатор объекта
- **width** – ширина объекта в пикселях
- **height** – высота объекта в пикселях
- **color** – цвет объекта

Для смещения квадратов использовались свойства **anchors.left**, **anchors.top**, **anchors.bottom** для размещения элементов по опорным точкам.

Элемент Text

- **text** – текст надписи
- **color** – цвет надписи
- **anchors.centerIn** – размещение надписи по опорной точке "центр"
- **font.pointSize** – размер шрифта надписи в пикселях

Контейнеры Column, Row, Grid

Для отрисовки пяти квадратов использовались следующие контейнеры:

- **Column** – компонент для размещения элементов в столбец
- **Row** – для размещения элементов в ряд
- **Grid** – для размещения элементов сеткой

Свойство transform

Для трансформирования квадрата использовалось свойство **transform** со следующими параметрами:

- **Scale** – изменение масштаба элемента

- **Rotation** – вращение элемента

Анимация **ParallelAnimation**, **NumberAnimation**

Для анимации квадрата использовался **ParallelAnimation**, который позволяет запускать несколько анимаций параллельно и **NumberAnimation**, который анимирует изменения числовых значений. **NumberAnimation** имеет следующие параметры:

- **target** – целевой элемент анимации
- **property** – свойство, которое анимация меняет
- **to** – конечное значение анимации
- **duration** – продолжительность анимации
- **loops** – заикливание анимации

Элемент **Dialog**

Для обработки данных пользователя используется элемент **Dialog**, который в котором есть два свойства, определяющие, что происходит, когда пользователь нажимает **Асепт** или **Cancel**: **onAccepted** и **onCanceled**. Для вывода суммы двух чисел в консоль свойство **onAccepted** имеет вид:

```
onAccepted: console.log(parseInt(num1.text) + parseInt(num2.text))
```

Для ввода данных используются элементы **TextField**.

Руководство пользователя

После запуска программы пользователь видит страницу с первым заданием и кнопку “Next”, которая перейдет на следующую страницу.

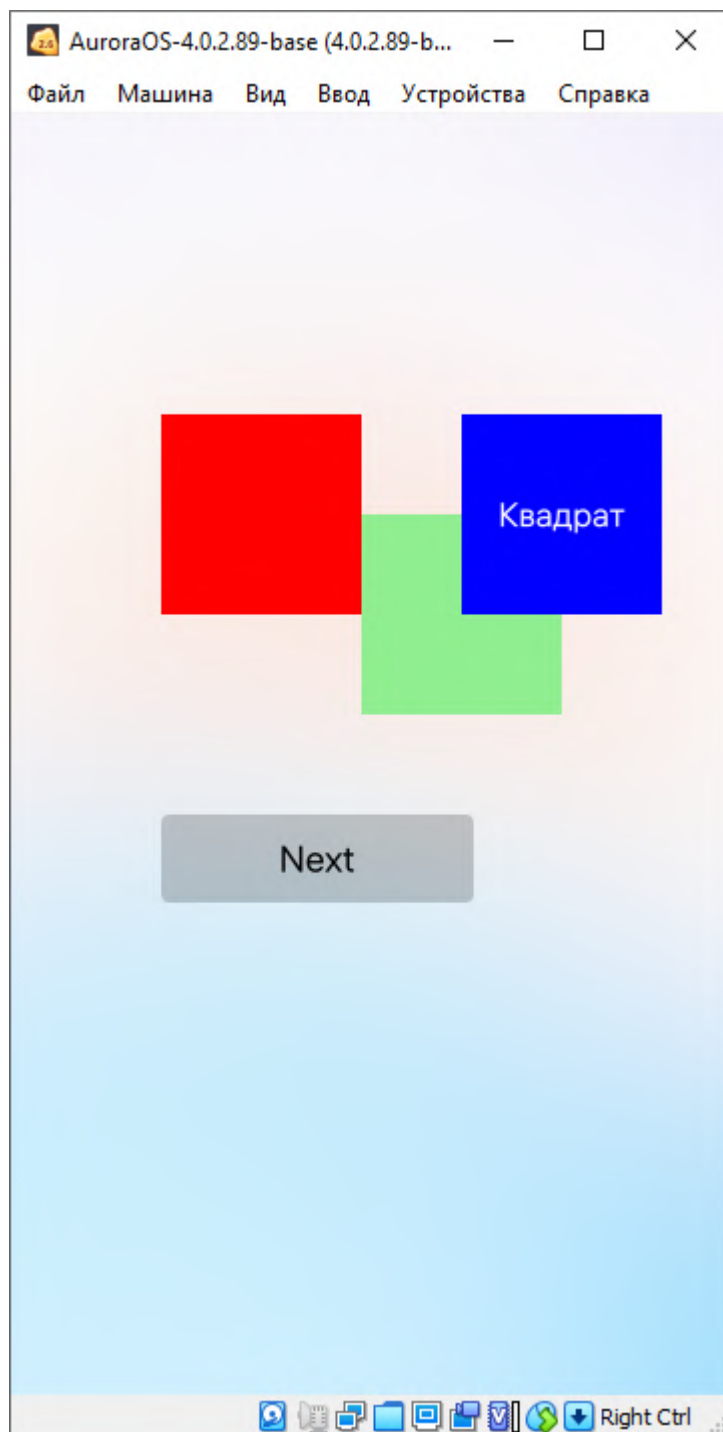


Рис. 1: Страница с первым заданием

На следующих страницах появляется кнопка “Back”, возвращающая предыдущую страницу.

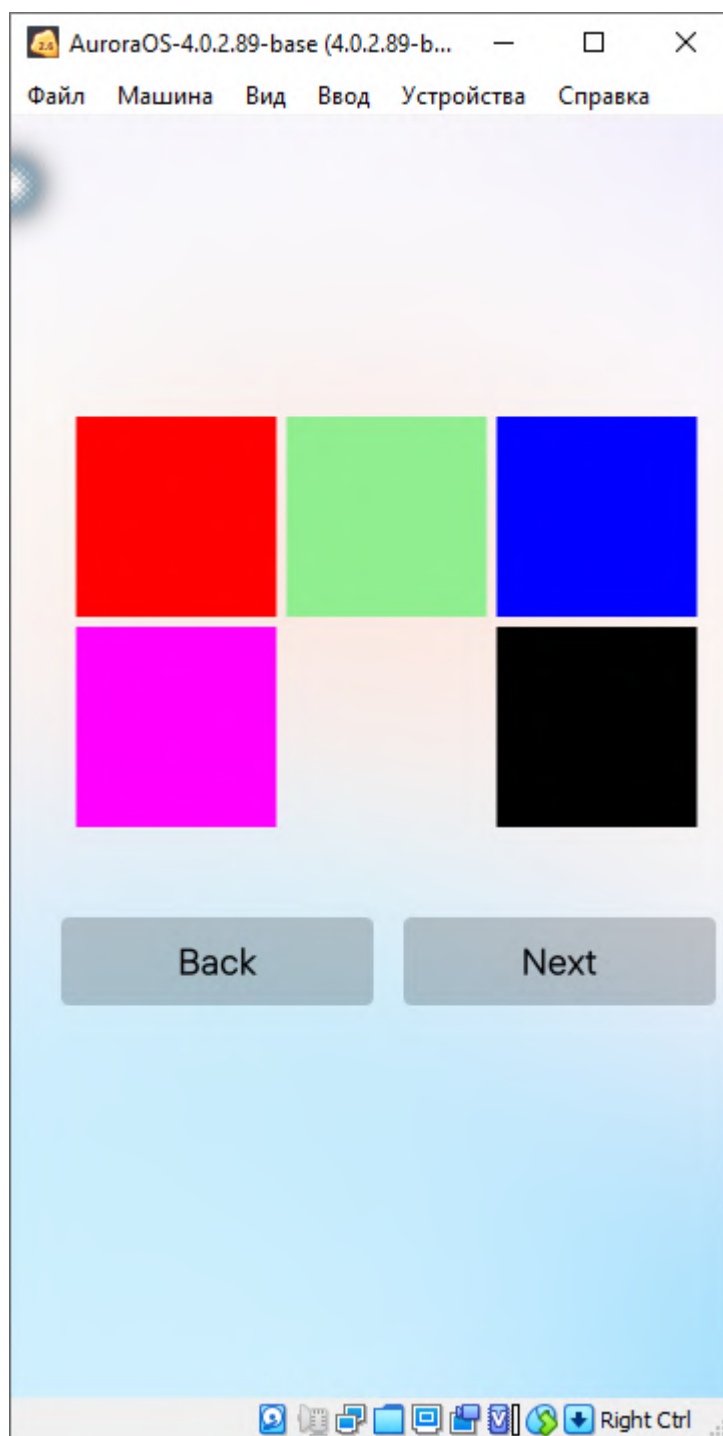


Рис. 2: Страница со вторым заданием

На 6-ом задании есть кнопка “Открыть диалог” для открытия диалога.

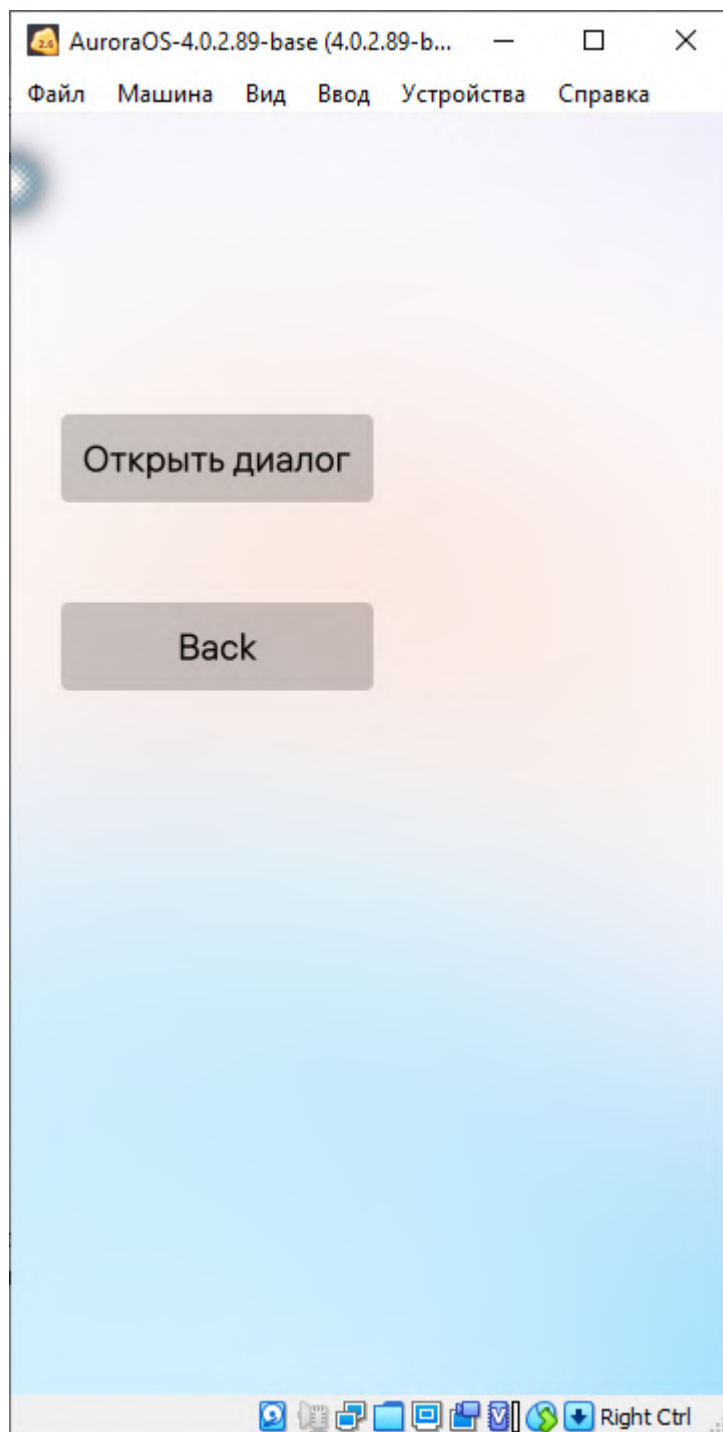


Рис. 3: Кнопка “Открыть диалог”

В интерфейсе диалога есть кнопки “Ассерп”, “Сancel” и два поля ввода текста.

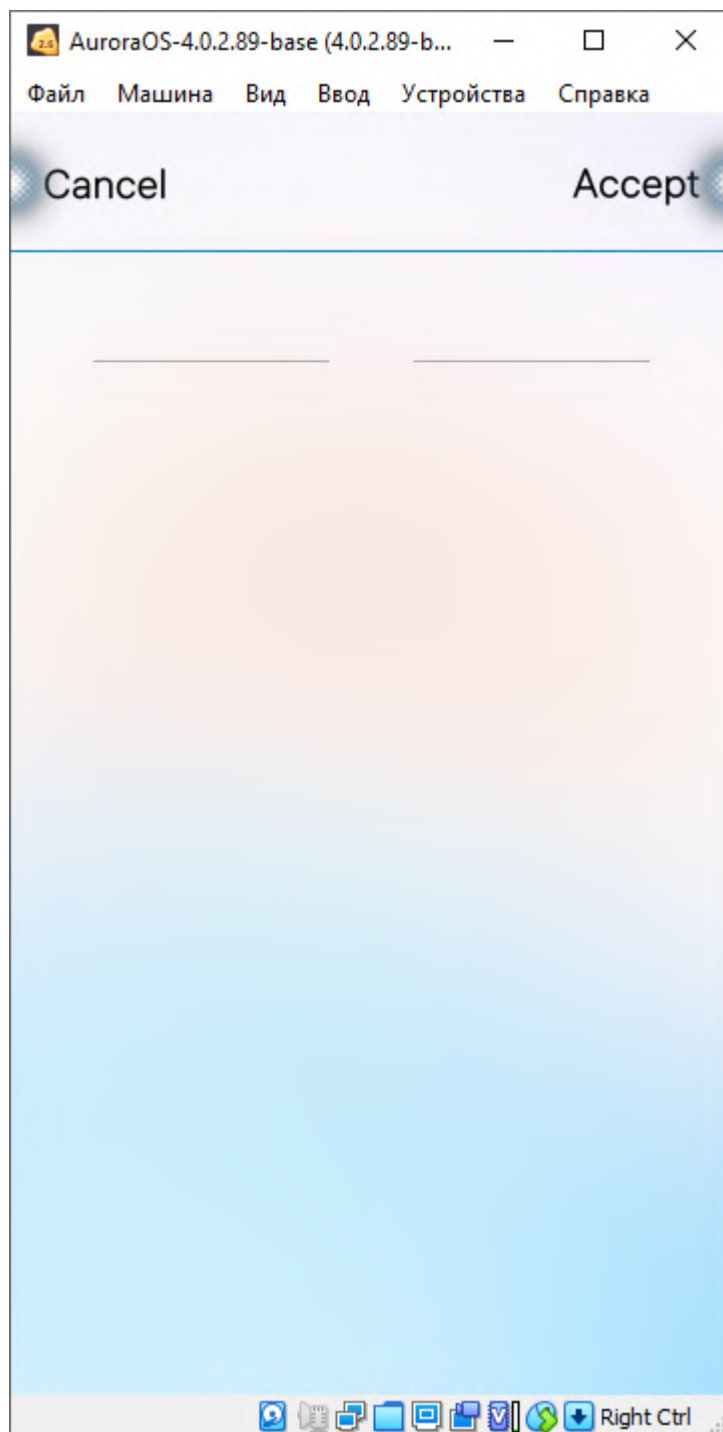


Рис. 4: Интерфейс диалога

При вводе чисел в диалог и нажатии кнопки “Асепт” сумма чисел появляется в консоли.

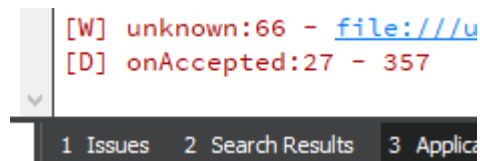


Рис. 5: Вывод в консоли

Заключение

В процессе выполнения данной лабораторной работы мы освоили базовые навыки построения пользовательских интерфейсов, позиционирования, отрисовки и перемещения элементов. Научились анимировать элементы. Научились создавать диалоги и взаимодействовать с ними, а также выполнили практическое задание.

Приложение

Page1.qml

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    Column {
        spacing: 100
        y: 300
        x: 150

        Item {
            width: 500
            height: 300

            Rectangle {
                id: red
                width: 200
                height: 200
                color: "red"
            }

            Rectangle {
                id: greed
                width: 200
                height: 200
                color: "lightgreen"
                anchors.top: red.verticalCenter
                anchors.left: red.right
            }

            Rectangle {
                id: blue
                width: 200
                height: 200
                color: "blue"
                anchors.left: greed.horizontalCenter
                anchors.bottom: greed.verticalCenter
            }

            Text {
                text: "Квадрат"
                color: "white"
                anchors.centerIn: blue
                font.pointSize: 25
            }
        }
    }
}
```

```

    }

    Row {
        Button {
            text: "Next"
            onClicked: pageStack.push(Qt.resolvedUrl("Page2.qml"))
        }
    }
}
}
}

```

Page2.qml

```

import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    Column {
        spacing: 100
        y: 300
        x: 50

        Item {
            width: 650
            height: 400

            Row {
                anchors.horizontalCenter: parent.horizontalCenter
                spacing: 10

                Column {
                    spacing: 10

                    Rectangle {
                        width: 200
                        height: 200
                        color: "red"
                    }

                    Rectangle {
                        width: 200
                        height: 200
                        color: "magenta"
                    }
                }
            }

            Column {
                spacing: 10
            }
        }
    }
}

```

```

        Rectangle {
            width: 200
            height: 200
            color: "lightgreen"
        }
    }

    Column {
        spacing: 10

        Rectangle {
            width: 200
            height: 200
            color: "blue"
        }

        Rectangle {
            width: 200
            height: 200
            color: "black"
        }
    }
}

Row {
    spacing: 30
    Button {
        text: "Back"
        onClicked: pageStack.pop()
    }

    Button {
        text: "Next"
        onClicked: pageStack.push(Qt.resolvedUrl("Page3.qml"))
    }
}
}
}

```

Page3.qml

```

import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    Column {

```



```

spacing: 100
y: 300
x: 50

Item {
    anchors.horizontalCenter: parent.horizontalCenter
    width: 650
    height: 400

    Grid {
        spacing: 10
        columns: 3

        Rectangle {
            width: 200
            height: 200
            color: "red"
        }

        Rectangle {
            width: 200
            height: 200
            color: "lightgreen"
        }

        Rectangle {
            width: 200
            height: 200
            color: "blue"
        }

        Rectangle {
            width: 200
            height: 200
            color: "magenta"
        }

        Rectangle {
            width: 200
            height: 200
            opacity: 0
        }

        Rectangle {
            width: 200
            height: 200
            color: "black"
        }
    }
}

```

```

    Row {
        spacing: 30
        Button {
            text: "Back"
            onClicked: pageStack.pop()
        }

        Button {
            text: "Next"
            onClicked: pageStack.push(Qt.resolvedUrl("Page4.qml"))
        }
    }
}
}

```

Page4.qml

```

import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    Column {
        spacing: 100
        y: 300
        x: 50

        Item {
            width: 350
            height: 200

            Row {
                spacing: 200

                Rectangle {
                    width: 200
                    height: 200
                    color: "black"
                }

                Rectangle {
                    width: 200
                    height: 200
                    color: "black"
                    transform: [
                        Scale {
                            xScale: 0.5

```

```

        },
        Rotation {
            angle: 45
        }
    ]
}
}
}
}
Row {
    spacing: 30
    Button {
        text: "Back"
        onClicked: pageStack.pop()
    }

    Button {
        text: "Next"
        onClicked: pageStack.push(Qt.resolvedUrl("Page5.qml"))
    }
}
}
}
}

```

Page5.qml

```

import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    Column {
        spacing: 100
        y: 300
        x: 50

        Item {
            anchors.horizontalCenter: parent.horizontalCenter
            width: 300
            height: 400

            Rectangle {
                id: black
                width: 200
                height: 200
                color: "black"
                anchors.horizontalCenter: parent.horizontalCenter

                ParallelAnimation {

```

```

        running: true
        NumberAnimation {
            target: black
            property: "y"
            to: 150
            duration: 800
            loops: Animation.Infinite
        }
        NumberAnimation {
            target: black
            property: "scale"
            to: 2
            duration: 800
            loops: Animation.Infinite
        }
    }
}
}
Row {
    spacing: 30
    Button {
        text: "Back"
        onClicked: pageStack.pop()
    }

    Button {
        text: "Next"
        onClicked: pageStack.push(Qt.resolvedUrl("Page6.qml"))
    }
}
}
}

```

Page6.qml

```

import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    Dialog {
        id: dialog
        Column {
            anchors.fill: parent
            spacing: Theme.paddingMedium
            DialogHeader { }
            Row {

```

```

        anchors.horizontalCenter: parent.horizontalCenter
        spacing: 20
        TextField {
            id: num1
            width: 300
            horizontalAlignment: TextInput.AlignHCenter
        }
        TextField {
            id: num2
            width: 300
            horizontalAlignment: TextInput.AlignHCenter
        }
    }
}
onAccepted: console.log(parseInt(num1.text) + parseInt(num2.text))
}

Column {
    spacing: 100
    y: 300
    x: 50

    Button {
        text: "Открыть диалог"
        onClicked: dialog.open()
    }

    Row {
        spacing: 30
        Button {
            text: "Back"
            onClicked: pageStack.pop()
        }
    }
}
}
}

```