

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 №3**

### **«Инструменты разработки мобильных приложений»**

**Выполнил:**

студент группы 381908-4  
Грищенко А. А.

Нижний Новгород  
2022

# **Оглавление**

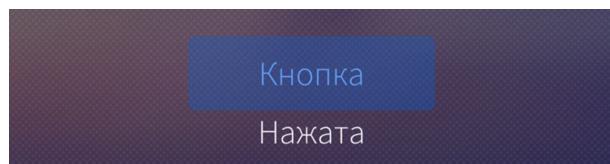
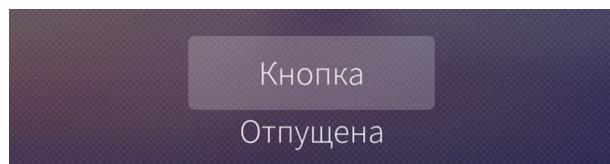
Цель задачи . . . . .	3
Постановка задачи . . . . .	4
Описание программной реализации . . . . .	6
Руководство пользователя . . . . .	7
Заключение . . . . .	10
Приложение . . . . .	11

## **Цель задачи**

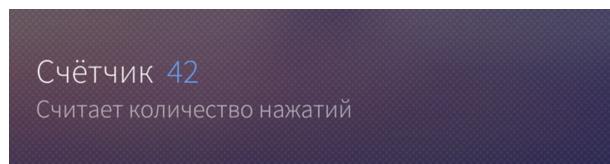
В данном лабораторной работе требуется научиться применять типовые элементы интерфейса Sailfish OS.

## Постановка задачи

1. Создать текстовое поле для ввода числа с заголовком и подсказкой.
2. Создать кнопку, которая будет сохранять визуально нажатое состояние, после того, как пользователь нажал на неё один раз.
3. Создать кнопку и поле с текстом. Поле с текстом должно отображать нажата ли кнопка или нет выводом текста “Нажата” или “Отпущена”.

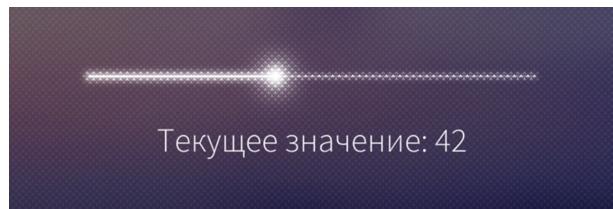


4. Создать кнопку со значением, которая будет отображать количество нажатий на неё.



5. Создать селектор даты, который будет отображать выбранную дату в консоли.
6. Создать селектор времени, который будет отображать выбранное время в консоли.
7. Создать поле с выпадающим списком, позволяющее выбрать строку из списка. Результат выбора отобразить в консоли.
8. Создать переключатель с текстом, в тексте отобразить состояние переключателя “Включен” или “Выключен”.

9. Создать ползунок и поле с текстом. Поле с текстом должно отображать текущее значение ползунка.



# Описание программной реализации

**Задание 1** – создать текстовое поле для ввода числа с заголовком и подсказкой. Для этого задействуем **TextField** с указанием подсказки в *placeholderText* и описанием в *description*.

**Задание 2** – создать кнопку, которая будет сохранять визуально нажатое состояние после того, как пользователь нажал на неё один раз. Для этого в элементе **Button** создадим пользовательское свойство *bool toggle*, укажем его в свойство *down*, а в *onClicked* будем делать *toggle = !toggle*.

**Задание 3** – создать кнопку и поле с текстом. Поле с текстом должно отображать нажата ли кнопка или нет выводом текста “Нажата” или “Отпущена”. Для этого в элемент **Label** будем передавать свойство *down* у **Button** и воспользуемся условной отрисовкой.

**Задание 4** – создать кнопку со значением, которая будет отображать количество нажатий на неё. По аналогии с первой лабораторной, будем увеличивать пользовательское свойство в *onClicked* у **Button** и передавать его в **Label**.

**Задание 5** – создать селектор даты, который будет отображать выбранную дату в консоли. Используется **DatePicker** и метод *onDateChanged*.

**Задание 6** – создать селектор времени, который будет отображать выбранное время в консоли. Используется **TimePicker** и метод *onTimeChanged*.

**Задание 7** – создать поле с выпадающим списком, позволяющее выбрать строку из списка. Результат выбора отобразить в консоли. Используется **ComboBox**. Пункты списка задаются полем *menu*, в котором мы размещаем элементы **MenuItem** со свойством *text*. Вывод результата в консоль в методе *onCurrentIndexChanged*.

**Задание 8** – создать переключатель с текстом, в тексте отобразить состояние переключателя “Включен” или “Выключен”. Используется элемент **Switch** и **Label** для вывода состояния. Состояние проверяется свойством *checked* элемента **Switch**.

**Задание 9** – создать ползунок и поле с текстом. Поле с текстом должно отображать текущее значение ползунка. Используются **Slider** со свойством *value* и **Label** для вывода результата.

# Руководство пользователя

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

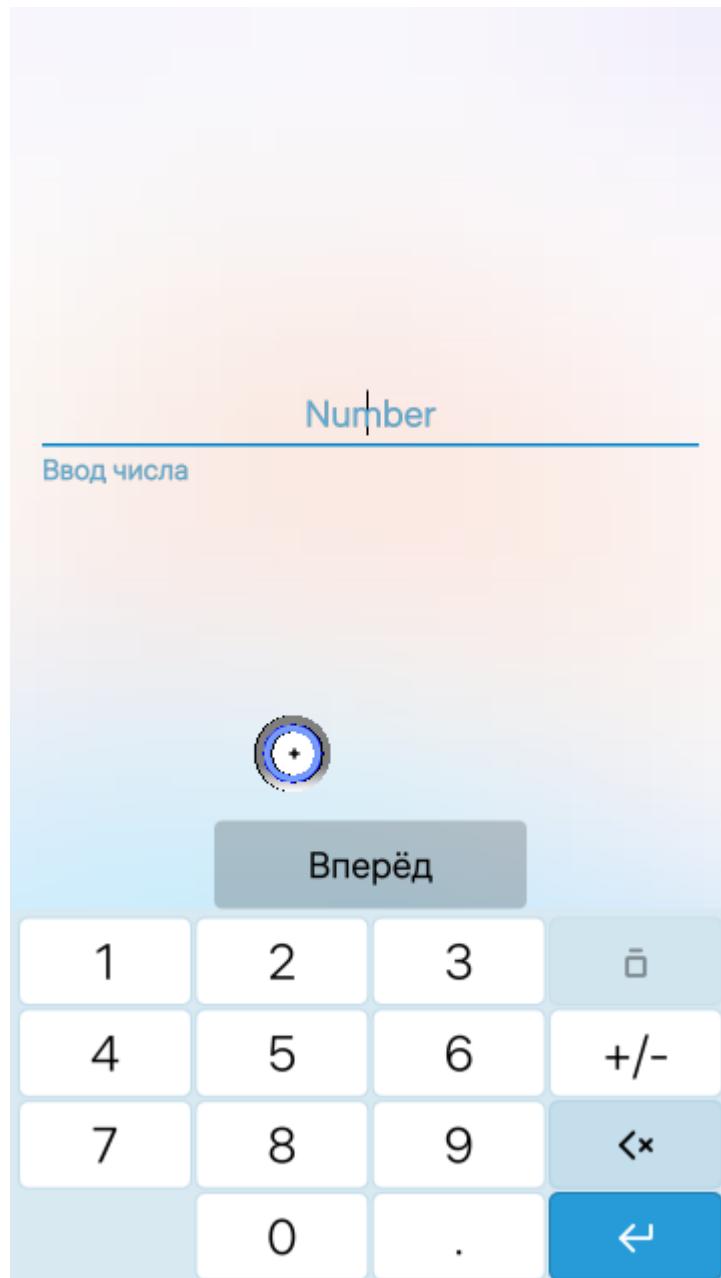


Рис. 1: Текстовое поле для ввода цифр

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

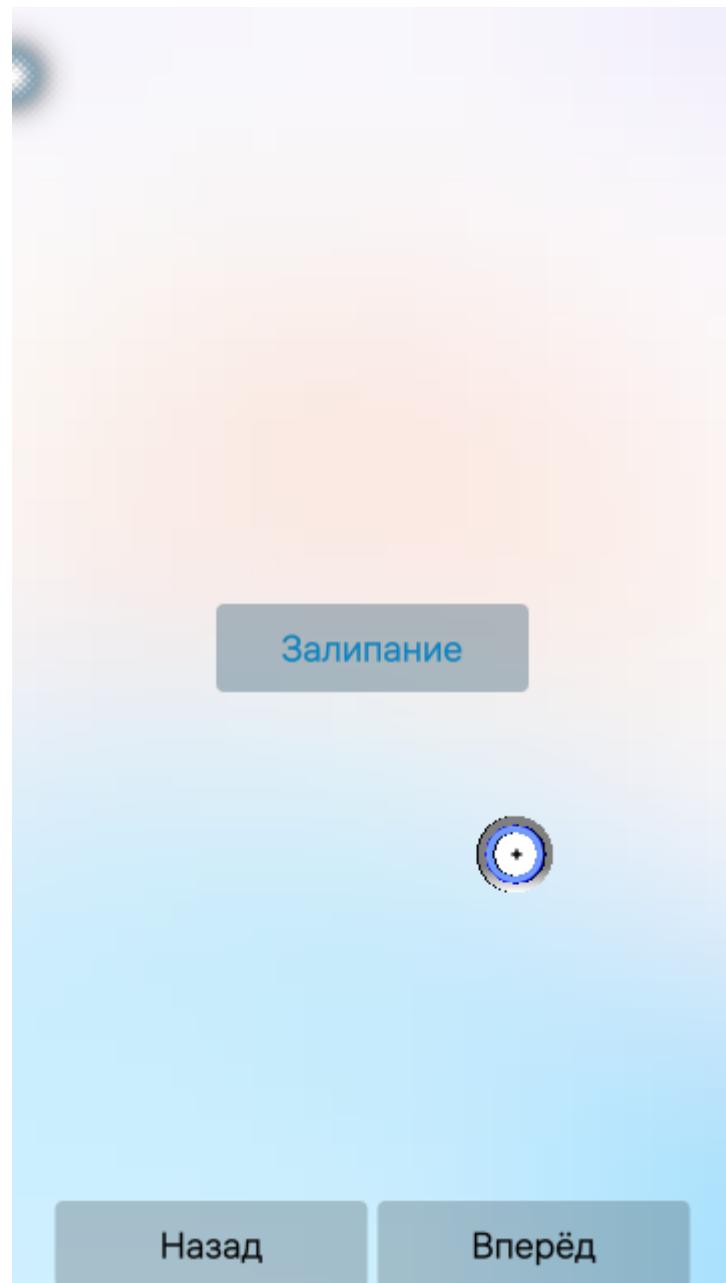
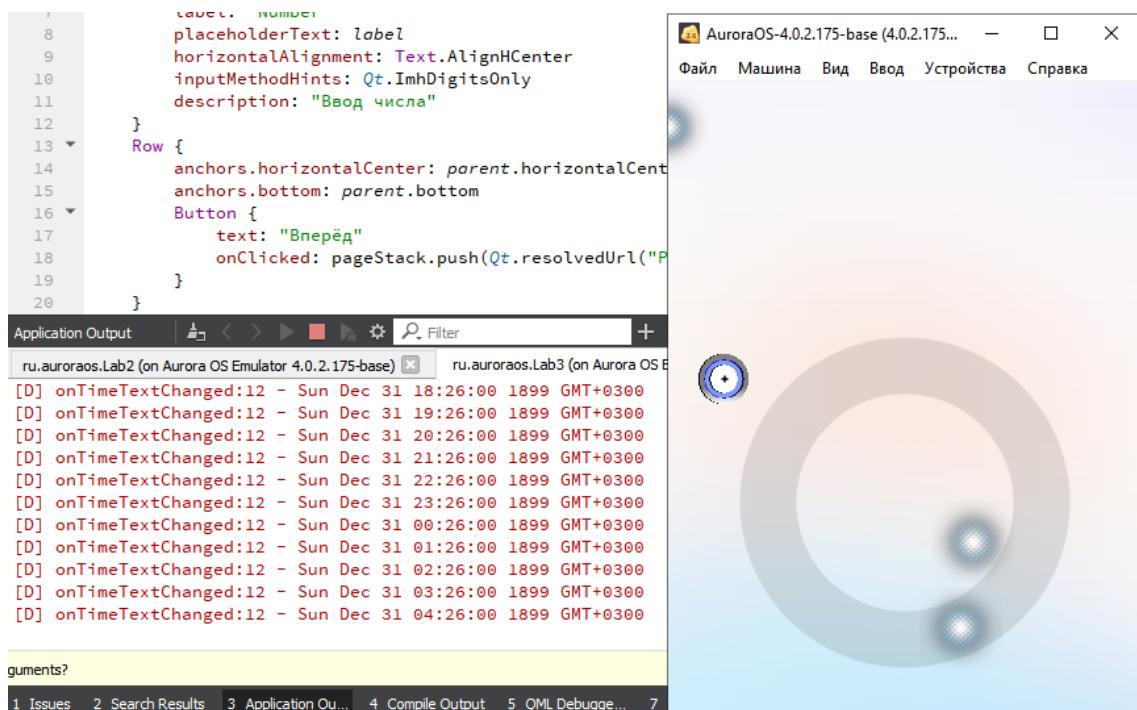


Рис. 2: Кнопка с залипанием

При изменении выбора элемента, в консоль делается вывод.



The screenshot shows the Aurora OS Emulator interface. On the left is a code editor with QML code. On the right is a window titled "AuroraOS-4.0.2.175-base (4.0.2.175...)" containing a digital clock application. The application has a large grey circle with three smaller circles inside, representing a clock face. Below the clock is a status bar with the text "ru.auroraos.Lab2 (on Aurora OS Emulator 4.0.2.175-base)". To the right of the status bar is a terminal window showing the following log entries:

```
[D] onTimeTextChanged:12 - Sun Dec 31 18:26:00 1899 GMT+0300
[D] onTimeTextChanged:12 - Sun Dec 31 19:26:00 1899 GMT+0300
[D] onTimeTextChanged:12 - Sun Dec 31 20:26:00 1899 GMT+0300
[D] onTimeTextChanged:12 - Sun Dec 31 21:26:00 1899 GMT+0300
[D] onTimeTextChanged:12 - Sun Dec 31 22:26:00 1899 GMT+0300
[D] onTimeTextChanged:12 - Sun Dec 31 23:26:00 1899 GMT+0300
[D] onTimeTextChanged:12 - Sun Dec 31 00:26:00 1899 GMT+0300
[D] onTimeTextChanged:12 - Sun Dec 31 01:26:00 1899 GMT+0300
[D] onTimeTextChanged:12 - Sun Dec 31 02:26:00 1899 GMT+0300
[D] onTimeTextChanged:12 - Sun Dec 31 03:26:00 1899 GMT+0300
[D] onTimeTextChanged:12 - Sun Dec 31 04:26:00 1899 GMT+0300
```

Below the terminal window is a navigation bar with tabs: "Application Output" (selected), "Issues", "Search Results", "Compile Output", "QML Debugge...", and "7".

Рис. 3: Выбор времени и вывод в консоли

## **Заключение**

В процессе выполнения данной лабораторной работы мы научились использовать типовые элементы интерфейса.

# Приложение

## Page1.qml

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    TextField {
        anchors.centerIn: parent
        label: "Number"
        placeholderText: label
        horizontalAlignment: Text.AlignHCenter
        inputMethodHints: Qt.ImhDigitsOnly
        description: "Ввод числа"
    }
    Row {
        anchors.horizontalCenter: parent.horizontalCenter
        anchors.bottom: parent.bottom
        Button {
            text: "Вперёд"
            onClicked: pageStack.push(Qt.resolvedUrl("Page2.qml"))
        }
    }
}
```

## Page2.qml

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    Button {
        anchors.centerIn: parent
        property bool toggle: false
        anchors.horizontalCenter: parent.horizontalCenter
        text: "Залипание"
        down: toggle
        onClicked: toggle = !toggle
    }
    Row {
        anchors.horizontalCenter: parent.horizontalCenter
        anchors.bottom: parent.bottom
        spacing: 10
        Button {
            text: "Назад"
            onClicked: pageStack.pop()
        }
    }
}
```

```

    }

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

}

```

## Page3.qml

```

import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    Column {
        anchors.centerIn: parent
        spacing: 100

        Column {
            anchors.horizontalCenter: parent.horizontalCenter
            spacing: 20

            Button {
                id: btn
                anchors.horizontalCenter: parent.horizontalCenter
                text: "Нажать!"
            }
            Label {
                anchors.horizontalCenter: parent.horizontalCenter
                text: btn.down ? "Нажато" : "Не нажато"
            }
        }
    }
    Row {
        anchors.horizontalCenter: parent.horizontalCenter
        anchors.bottom: parent.bottom
        spacing: 10
        Button {
            text: "Назад"
            onClicked: pageStack.pop()
        }

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

```

```
}
```

## Page4.qml

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    Column {
        anchors.centerIn: parent
        spacing: 100

        Column {
            anchors.horizontalCenter: parent.horizontalCenter
            spacing: 20

            Button {
                property int counter: 0

                id: btn
                anchors.horizontalCenter: parent.horizontalCenter
                text: "Счетчик"
                onClicked: counter++
            }
            Label {
                anchors.horizontalCenter: parent.horizontalCenter
                text: btn.counter
                color: "deepskyblue"
            }
        }
    }
    Row {
        anchors.horizontalCenter: parent.horizontalCenter
        anchors.bottom: parent.bottom
        spacing: 10
        Button {
            text: "Назад"
            onClicked: pageStack.pop()
        }

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

## Page5.qml

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    DatePicker {
        anchors.centerIn: parent
        date: new Date()
        daysVisible: true
        weeksVisible: true
        onDateTextChanged: console.log(this.date)
    }
    Row {
        anchors.horizontalCenter: parent.horizontalCenter
        anchors.bottom: parent.bottom
        spacing: 10
        Button {
            text: "Назад"
            onClicked: pageStack.pop()
        }

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

## Page6.qml

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    TimePicker {
        anchors.centerIn: parent
        anchors.horizontalCenter: parent.horizontalCenter
        id: timePicker
        hour: 15
        minute: 26
        hourMode: DateTime.TwelveHours
        onTimeTextChanged: console.log(this.time)
    }
    Row {
        anchors.horizontalCenter: parent.horizontalCenter
        anchors.bottom: parent.bottom
        spacing: 10
```

```

        Button {
            text: "Назад"
            onClicked: pageStack.pop()
        }

        Button {
            text: "Вперёд"
            onClicked: pageStack.push(Qt.resolvedUrl("Page7.qml"))
        }
    }
}

```

## Page7.qml

```

import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    ComboBox {
        anchors.centerIn: parent
        label: "Выпадающий список"
        description: "Описание выпадающего списка"
        menu: ContextMenu {
            MenuItem { text: "первый" }
            MenuItem { text: "второй" }
            MenuItem { text: "третий" }
        }
        onCurrentIndexChanged: console.log(value)
    }
    Row {
        anchors.horizontalCenter: parent.horizontalCenter
        anchors.bottom: parent.bottom
        spacing: 10
        Button {
            text: "Назад"
            onClicked: pageStack.pop()
        }

        Button {
            text: "Вперёд"
            onClicked: pageStack.push(Qt.resolvedUrl("Page8.qml"))
        }
    }
}

```

## Page8.qml

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    Column {
        anchors.centerIn: parent
        anchors.horizontalCenter: parent.horizontalCenter
        Switch {
            anchors.horizontalCenter: parent.horizontalCenter
            id: mute
            icon.source: "image://theme/icon-m-speaker-mute?"
                + (checked ? Theme.highlightColor
                    : Theme.primaryColor)
        }
        Label {
            text: "The sound is " + (mute.checked ? "off" : "on")
        }
    }
    Row {
        anchors.horizontalCenter: parent.horizontalCenter
        anchors.bottom: parent.bottom
        spacing: 10
        Button {
            text: "Назад"
            onClicked: pageStack.pop()
        }

        Button {
            text: "Вперёд"
            onClicked: pageStack.push(Qt.resolvedUrl("Page9.qml"))
        }
    }
}
```

## Page9.qml

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    Column {
        anchors.centerIn: parent
        Slider {
            width: 500
            label: "Ползунок"
            maximumValue: 40
        }
    }
}
```

```
        minimumValue: -10
        value: 10
        stepSize: 1
        valueText: value
        onValueChanged: console.log(value)
        id: slider
    }

    Label {
        anchors.horizontalCenter: parent.horizontalCenter
        text: "Значение: " + slider.value
    }
}

Row {
    anchors.horizontalCenter: parent.horizontalCenter
    anchors.bottom: parent.bottom
    spacing: 10
    Button {
        text: "Назад"
        onClicked: pageStack.pop()
    }
}
}
```