МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ 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**

**Кафедра Математического обеспечения и суперкомпьютерных технологий**

Направление подготовки: «Фундаментальная информатика и

информационные технологии»

Профиль подготовки: «Инженерия программного обеспечения»

**ОТЧЕТ**

**По лабораторной работе №4**

по дисциплине «Технология программирования мобильных систем»

**Выполнил(а):** студент(ка) группы

381908-4

Кургузиков Кирилл

**Руководитель:**

доцент кафедры программной

инженерии

Борисов Николай Анатольевич

Нижний Новгород

2022

Содержание

[**1.Цель** 3](#_Toc97543312)

[**2.Постановка задачи** 4](#_Toc97543313)

[**3.Решение поставленной задачи** 5](#_Toc97543314)

[**4.Руководство пользователя** 6](#_Toc97543315)

[**5.Приложение** 8](#_Toc97543316)

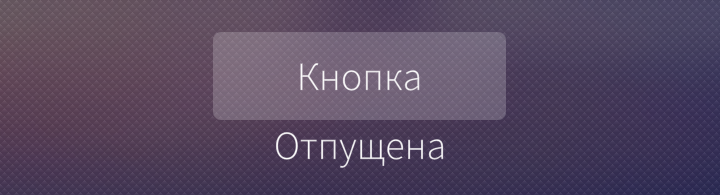
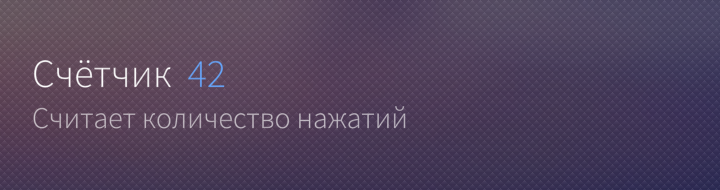
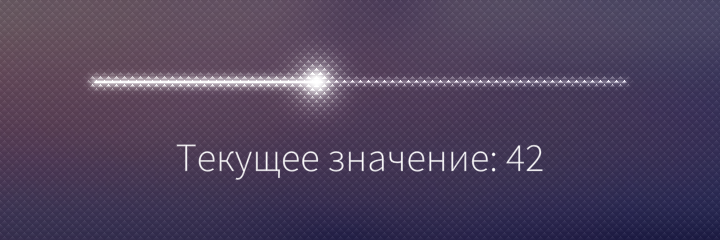
# **1.Цель**

Цель: научиться применять типовые элементы интерфейса Sailfish OS

# **2.Постановка задачи**

Цель: научиться применять типовые элементы интерфейса Sailfish OS

Шаги:

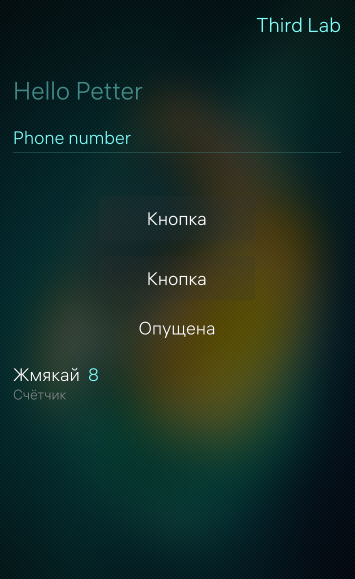
1. создать текстовое поле для ввода числа с заголовком и подсказкой
2. создать кнопку, которая будет сохранять визуально нажатое состояние, после того, как пользователь нажал на неё один раз
3. создать кнопку и поле с текстом. Поле с текстом должно отображать нажата ли кнопка или нет выводом текста “Нажата”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

# **3.Решение поставленной задачи**

Для создания текстового поля для ввода числа был создан контейнер TextField с методом inputMethodHints:*Qt*.ImhDialableCharactersOnly. Чтобы создать кнопку, которая будет сохранять визуально нажатое состояние, была добавлена переменна bool, которая сохраняет состояние кнопки, также был добавлен метод OnClicked, чтобы сохранять состояние кнопки. Для создания кнопки с надписью были созданы контейнеры Button и Label. Был сделан id для Button, с помощью которой будет передаваться значение метода down. Была создан счётчик с помощью контейнера ValueButton, в которой есть переменная a, которая увеличится при нажатии. Для создания селектора даты был создан контейнер DatePicker в которой есть метод onDateChanged: *console*.log (*datePicker*.dateText), с селектором времени были проведены точно, такиеже манипуляции. Для создания списка был создан ComboBox в которой были добавлены элементы с помощь. Метода MenuItem и был добавлен метод onCurrentIndexChanged: *console*.log(*value*), с помощью. Которого будет выводиться выбранный элемент в консоль. Для создания переключателя был создан элемент TextSwitch, в которой была добавлена переменная bool, которая меняется при нажатии на неё, также был добавлен метод OnClicked, который меняет значение переменной a. Для добавления ползунка был добавлен элемент Slider в которой есть элемент label, которая меняется со значением ползунка.

# **4.Руководство пользователя**

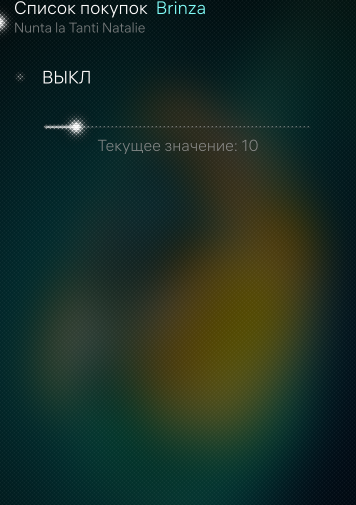
При открытии приложения на 1 странице будет поле для ввода номера телефона 2 кнопки. 1-ая кнопка сохраняет визуально нажатие на неё, 2-ая кнопка при её нажатии меняет текст ниже на “Нажата” если кнопка нажата. Внизу есть счётчик при нажатии на который будет увеличиваться значение.



На второй странице есть селектор даты и времени, при нажатии на любую дату, она будет выведена в консоль, также и селектором времени, при выбора времени оно будет выведена в консоль.



На 3 странице есть список, переключатель и ползунок. При нажатии на список будет дан выбор, что из этого списка выбрать. Переключатель при нажатии на него пишет “ВКЛ” и “ВЫКЛ”. Ползунок меняет значение с 0 до 100.



# **5.Приложение**

FirstPage.qml

import QtQuick 2.0

import Sailfish.Silica 1.0

Page {

id: *page*

// The effective value will be restricted by ApplicationWindow.allowedOrientations

allowedOrientations: Orientation.All

// To enable PullDownMenu, place our content in a SilicaFlickable

SilicaFlickable {

anchors.fill: *parent*

// PullDownMenu and PushUpMenu must be declared in SilicaFlickable, SilicaListView or SilicaGridView

PullDownMenu {

MenuItem {

text: *qsTr*("Show Page 2")

onClicked: *pageStack*.push(*Qt*.resolvedUrl("SecondPage.qml"))

}

}

// Tell SilicaFlickable the height of its content.

contentHeight: *column*.height

// Place our content in a Column. The PageHeader is always placed at the top

// of the page, followed by our content.

Column {

id: *column*

width: *page*.width

spacing: Theme.paddingLarge

PageHeader {

title: *qsTr*("Third Lab")

}

Label {

x: Theme.horizontalPageMargin

text: *qsTr*("Hello Petter")

color: Theme.secondaryHighlightColor

font.pixelSize: Theme.fontSizeExtraLarge

}

TextField {

width: *parent*.width

inputMethodHints:

*Qt*.ImhDialableCharactersOnly

label: "Phone number"

placeholderText: *label*

placeholderColor: Theme.highlightColor

color: Theme.highlightColor

}

Button {

property bool a: false

down: *a*

anchors.horizontalCenter:

*parent*.horizontalCenter

color: "white"

text: "Кнопка"

onClicked: if(*a*)

*a* = false

else

*a* = true;

}

Button{

id: *bt*

anchors.horizontalCenter:

*parent*.horizontalCenter

color: "white"

text: "Кнопка"

}

Label{

anchors.horizontalCenter:

*parent*.horizontalCenter

text: if(*bt*.down)

"Нажата"

else

"Опущена"

}

ValueButton {

property int a: 0

label: "Жмякай"

description: "Счётчик"

value: *a*

onClicked: *a*++

}

}

}

}

SecondPage.qml

import QtQuick 2.0

import Sailfish.Silica 1.0

Page {

id: *page*

// The effective value will be restricted by ApplicationWindow.allowedOrientations

allowedOrientations: Orientation.All

// To enable PullDownMenu, place our content in a SilicaFlickable

SilicaFlickable {

anchors.fill: *parent*

// PullDownMenu and PushUpMenu must be declared in SilicaFlickable, SilicaListView or SilicaGridView

PullDownMenu {

MenuItem {

text: *qsTr*("Show Page 3")

onClicked: pageStack.push(*Qt*.resolvedUrl("ThirdPage.qml"))

}

}

// Tell SilicaFlickable the height of its content.

contentHeight: *column*.height

// Place our content in a Column. The PageHeader is always placed at the top

// of the page, followed by our content.

}

Column{

id: *column*

width: *page*.width

spacing: Theme.paddingLarge

Item {

height: *timePicker*.height

width: *parent*.width

TimePicker {

anchors.horizontalCenter:

*parent*.horizontalCenter

id: *timePicker*

hour: 00

minute: 00

onHourChanged: *console*.log(*timeText*)

onMinuteChanged: *console*.log(*timeText*)

}

Label {

anchors.centerIn: *timePicker*

text: *timePicker*.timeText

font.pixelSize: Theme.fontSizeExtraLarge

}

}

DatePicker {

id: *datePicker*

daysVisible: true

weeksVisible: true

onDateChanged: *console*.log (*datePicker*.dateText)

}

}

}

ThirdPage.qml

import QtQuick 2.0

import Sailfish.Silica 1.0

Page {

id: *page*

// The effective value will be restricted by ApplicationWindow.allowedOrientations

allowedOrientations: Orientation.All

// To enable PullDownMenu, place our content in a SilicaFlickable

SilicaFlickable {

anchors.fill: *parent*

// PullDownMenu and PushUpMenu must be declared in SilicaFlickable, SilicaListView or SilicaGridView

PullDownMenu {

MenuItem {

text: *qsTr*("Show Page 3")

onClicked: pageStack.push(*Qt*.resolvedUrl("ThirdPage.qml"))

}

}

// Tell SilicaFlickable the height of its content.

contentHeight: column.height

// Place our content in a Column. The PageHeader is always placed at the top

// of the page, followed by our content.

}

// Place our content in a Column. The PageHeader is always placed at the top

// of the page, followed by our content.

Column {

width: *parent*.width

ComboBox {

label: "Список покупок"

description: "Nunta la Tanti Natalie"

menu: ContextMenu {

MenuItem { text: "Brinza" }

MenuItem { text: "Mamaliga" }

MenuItem { text: "Vin" }

MenuItem { text: "Sunca" }

MenuItem { text: "vita" }

MenuItem { text: "placinte" }

}

onCurrentIndexChanged: *console*.log(*value*)

}

TextSwitch {

property bool a: false

down: *a*

text: if(*down*)

"ВКЛ"

else

"ВЫКЛ"

onClicked: if(*a*)

*a* = false

else

*a* = true;

}

Slider {

id: *sl*

width: *parent*.width

label: "Текущее значение: " + *value*

maximumValue: 100

minimumValue: 0

value: 10

stepSize: 1

}

}

}