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**

Направление подготовки: «Фундаментальная информатика и информационные технологии»

Отчет по практическому заданию №2:

**«Инструменты разработки мобильных приложений»**

**Выполнил:**

студент группы 381906-1

Яшин Кирилл Евгеньевич

Нижний Новгород

2022г.

**Оглавление**

[Постановка задач 3](#_Toc116663234)

[Руководство программиста 4](#_Toc116663235)

[Руководства пользователя 6](#_Toc116663236)

[Вывод 8](#_Toc116663237)

[Приложение 9](#_Toc116663238)

# Постановка задач

В данном практическом задании необходимо:

1. Освоить базовые навыки построения пользовательских интерфейсов, позиционирования, отрисовки и перемещения элементов. Научиться анимировать элементы. Научиться создавать диалоги и взаимодействовать с ними.
2. Выполнить ряд практических задач: отрисовку квадратов 3 цветов (красного, зеленого и синего); поместить текст «Квадрат» белого цвета по центру синего квадрата; отрисовку 5 квадратов с использованием Column и Row, Grid; сделать из квадрата прямоугольник с использованием объектов Translate, Scale, Rotation; анимирование и трансформирование квадрата; реализовать диалог с двумя текстовыми полями, в которые вводятся числа и при нажатии кнопки они суммируются (использовать функцию parseInt).

# Руководство программиста

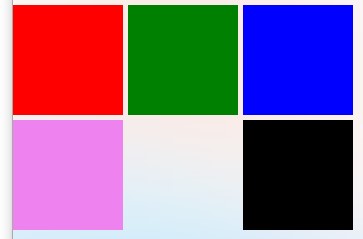
Для решения поставленной задачи был использован контейнер Column. Внутрь него поместились контейнеры Item с соответствующими отрисовками фигур, необходимых по заданию.

1. Внутри firstItem находится отрисовка первых трех квадратов данным образом:



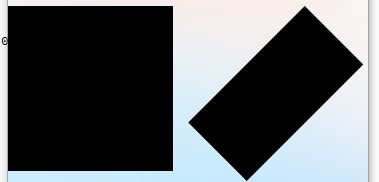
Для расположения квадратов были использованы якоря (anchors).

1. Внутри secondItem находится отрисовка первых трех квадратов данным образом:

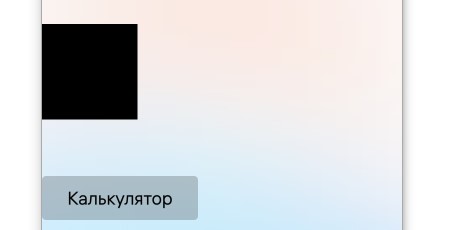
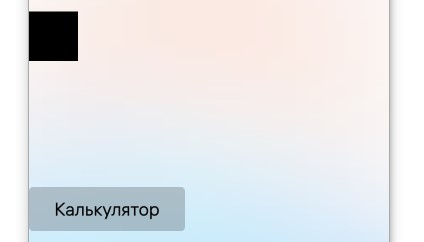


Отрисовка данных квадратов выполняется с помощью Column и Row. Для отступов используется spacing.

1. Внутри thirdItem отрисовываются такие же квадраты, как выше, но уже с использованием Grid.
2. Внутри fourthItem отрисовывается квадрат, из которого получается прямоугольник с использованием объектов Translate, Scale, Rotation:

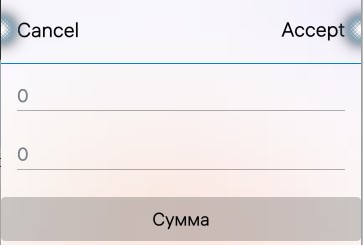


1. Далее, реализуется анимация увеличения квадрата с помощью ParallelAnimation.



1. Наконец, реализуется калькулятор. При нажатии на кнопку «Калькулятор» появляется следующее окно:

При вводе чисел в поля и последующем нажатии кнопок Accept или Сумма, в консоль выводится сумма двух введенных чисел. Данный функционал реализуется с помощью Dialog.



В качестве дополнительного задания нужно было увеличить размер квадратов до размера экрана и сделать скроллер. Размеры квадратов были зафиксированы с помощью масштабирования параметра width окна, а скроллер был реализован с помощью:

SilicaFlickable {

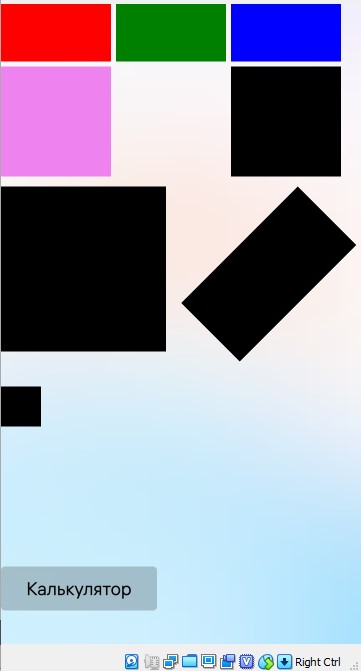
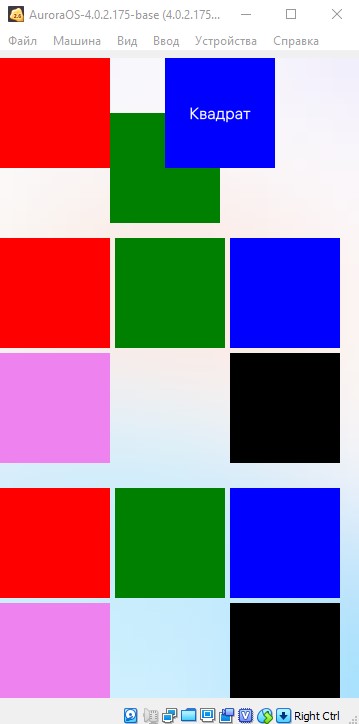
quickScroll : true

contentHeight: page.width \* 10

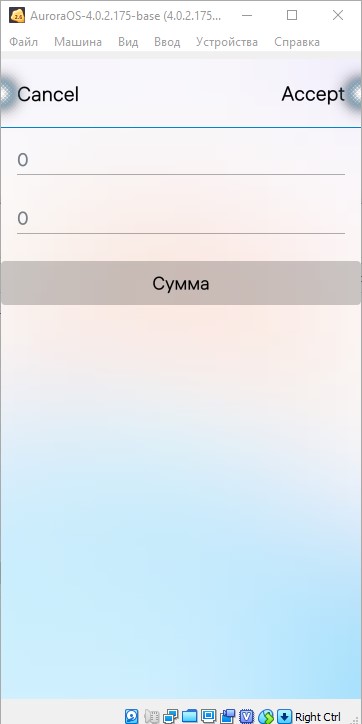
}

# Руководства пользователя

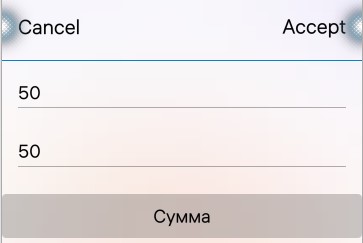
После запуска программы, открывается окно эмулятора, в котором отображается страница с данным практическим заданием.



При нажатии на кнопку «Калькулятор» происходит переход в диалог.



После ввода цифр в текстовые поля и нажатии на кнопку «Сумма» в консоль выводится их сумма.



# Вывод

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 практические задания.

Все отчеты и исходный код можно найти на моем GitHub - [ссылка](https://github.com/KirillYashin/Aurora_labs).

# Приложение

import QtQuick 2.0

import Sailfish.Silica 1.0

Page {

objectName: "mainPage"

allowedOrientations: Orientation.All

property int size: page.width / 3 - 20

SilicaFlickable {

anchors.fill: parent

quickScroll : true

contentHeight: page.width \* 10

Column {

id: column

width: page.width

spacing: Theme.paddingLarge

Item {

height: size \* 1.5

id: firstItem

Rectangle{

id: first

width: size

height: size

color: "red"

}

Rectangle{

id: second

width: size

height: size

color: "green"

anchors.verticalCenter: first.bottom

anchors.left: first.right

}

Rectangle{

id: third

width: size

height: size

color: "blue"

anchors.left: second.horizontalCenter

anchors.bottom: second.verticalCenter

Text{

text:"Квадрат"

color: "white"

anchors.centerIn: parent

}

}

}

Item {

id: secondItem

height: size \* 2

y: firstItem.height + 30

Row {

spacing: 10

Column{

spacing: 10

Rectangle{

width: size

height: size

color: "red"

}

Rectangle{

width: size

height: size

color: "violet"

}

}

Rectangle{

width: size

height: size

color: "green"

}

Column{

spacing: 10

Rectangle{

width: size

height: size

color: "blue"

}

Rectangle{

width: size

height: size

color: "black"

}

}

}

}

Item {

id: thirdItem

height: size \* 2

y: secondItem.height + firstItem.height + 90

Grid{

columns: 3

rows: 2

spacing: 10

Rectangle{

width: size

height: size

color: "red"

}

Rectangle{

width: size

height: size

color: "green"

}

Rectangle{

width: size

height: size

color: "blue"

}

Rectangle{

width: size

height: size

color: "violet"

}

Item{

width: size

height: size

}

Rectangle{

width: size

height: size

color: "black"

}

}

}

Item {

id: fourthItem

height: size \* 1.5

y: thirdItem.height + secondItem.height + firstItem.height + 120

Rectangle {

id: black

width: size \* 1.5

height: size \* 1.5

color: "black"

}

Rectangle {

width: size \* 1.5

height: size \* 1.5

color: "black"

transform: [

Scale {

xScale: 0.5

},

Rotation {

angle: 45

},

Translate {

x: size \* 2.7

}

]

}

}

Item {

id: fifthItem

height: size

y: fourthItem.height + thirdItem.height + secondItem.height + firstItem.height + 150

Rectangle{

id: animation

color:"black"

width: size

height: size

ParallelAnimation{

running: true

loops: Animation.Infinite

NumberAnimation {

target: animation

property: "width"

to:200

duration: 1000

}

NumberAnimation {

target: animation

property: "height"

to:200

duration: 1000

}

NumberAnimation {

target: animation

property: "y"

to:100

duration: 1000

}

}

}

}

Item{

y: fifthItem.height+ fourthItem.height + thirdItem.height + secondItem.height + firstItem.height + size \* 1.5

Button {

text: "Калькулятор"

onClicked:pageStack.push(dialog)

}

}

Component{

id: dialog

Dialog {

Column {

width: parent.width

DialogHeader {}

TextField {

id: num1

placeholderText: "0"

strictValidation: true

validator: IntValidator{}

}

TextField {

id: num2

placeholderText: "0"

strictValidation: true

validator: IntValidator{}

}

Button {

text: "Сумма"

width: parent.width

onClicked: console.log(parseInt(num1.text) + parseInt(num2.text))

}

}

}

}

}

}

}