МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ  
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: «Программная инженерия»

**ОТЧЁТ**

по лабораторной работе №3

**КОНТЕЙНЕРЫ И ПОЗИЦИОНИРОВАНИЕ**

**Выполнил:** студент группы

382008-1

\_\_\_А.Д. Кандрин \_\_\_\_\_\_\_

Подпись

**Проверил:** младший научный сотрудник

\_\_\_\_Д.А. Карчков\_\_\_\_\_\_\_\_

Подпись

Нижний Новгород  
2023 г.

## Содержание

[1. Постановка задачи 3](#_Toc27873363)

[2. ХОД РАБОТЫ](#_Toc27873364) 5

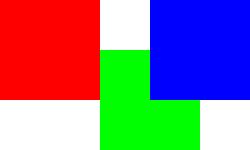
[3. ЗАКЛЮЧЕНИЕ](#_Toc27873365) 9

[4. ЛИТЕРАТУРА](#_Toc27873374) 10

## Постановка задачи.

Цель: освоить базовые навыки построения пользовательских интерфейсов, позиционирования, отрисовки и перемещения элементов. Научиться анимировать элементы. Научиться создавать диалоги и взаимодействовать с ними.

## Ход работы.

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

Для этого в раздел “Column” qml-файла страницы добавим несколько прямоугольников “Rectangle”, написав следующий код:

Item {

id: item1

width: page1.width

height: rectSize \* 1.75

property int rectSize: 200

Rectangle

{

id: rect1\_1

color: "red"

width: parent.rectSize

height : parent.rectSize

}

Rectangle

{

id: rect1\_2

anchors.left: rect1\_1.right

anchors.top: rect1\_1.verticalCenter

color: "lime"

width: parent.rectSize

height: parent.rectSize

}

Rectangle

{

id: rect1\_3

anchors.left : rect1\_2.horizontalCenter

anchors.bottom: rect1\_2.verticalCenter

color: "blue"

width: parent.rectSize

height: parent.rectSize

}

}

1. Поместим текст “Квадрат” белого цвета по центру синего квадрата. Для этого в описание последнего прямоугольника (с id “rect1\_3”) добавим раздел “Text” со следующим содержанием:

Text {

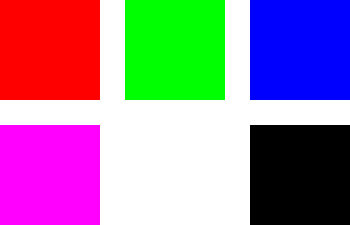
anchors.centerIn: parent

text: qsTr("Квадрат")

font.pointSize: parent.rectSize / 6

color: "white"

}

1. Нарисуем 5 квадратов с использованием “Column” и “Row” следующим образом:

Для этого добавим следующий код внутрь главного “Column”:

Item

{

id: item2

width: page1.width

height: rectSize \* 2.5

property int rectSize: 150

property int \_spacing : rectSize / 5

Row

{

spacing : item2.\_spacing

Column

{

spacing : item2.\_spacing;

Rectangle

{

width: item2.rectSize

height: item2.rectSize

color: "red"

}

Rectangle

{

width: item2.rectSize

height: item2.rectSize

color: "fuchsia"

}

}

Column

{

spacing : item2.\_spacing;

Rectangle

{

width: item2.rectSize

height: item2.rectSize

color: "lime"

}

}

Column

{

spacing : item2.\_spacing;

Rectangle

{

width: item2.rectSize

height: item2.rectSize

color: "blue"

}

Rectangle

{

width: item2.rectSize

height: item2.rectSize

color: "black"

}

}

}

}

1. Нарисуем те же пять квадратов с использованием “Grid”.

Item

{

id: item3

width: page1.width

height: rectSize \* 2.5

property int rectSize: 150

Grid

{

rows: 2

columns: 3

spacing: parent.rectSize / 5

Rectangle

{

width: item2.rectSize

height: item2.rectSize

color: "red"

}

Rectangle

{

width: item2.rectSize

height: item2.rectSize

color: "lime"

}

Rectangle

{

width: item2.rectSize

height: item2.rectSize

color: "blue"

}

Rectangle

{

width: item2.rectSize

height: item2.rectSize

color: "fuchsia"

}

// Это пустой элемент.

// Чтобы он добавился в Grid нужно ему добавить видимость, а также ненулевую длину и ширину

// (в соответствии с документацией Grid:

// If an item within a Grid is not visible, or if it has a width or height of 0,

// the item will not be laid out and it will not be visible within the column)

Item

{

visible: true

width: item2.rectSize

height: item2.rectSize

}

Rectangle

{

width: item2.rectSize

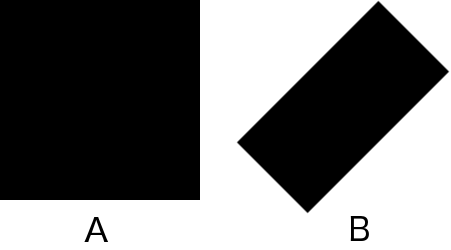
height: item2.rectSize

color: "black"

}

}

}

1. Сделаем из квадрата “A” прямоугольник “B” с использованием объектов Translate, Scale и Rotation.

Для этого добавим код, рисующий прямоугольник A, а затем – продублируем этот код с добавлением Translate, Scale, Rotation:

Item

{

id: item4

width: page1.width

height: rectSize

property int rectSize: 150

Rectangle

{

width: item4.rectSize

height: item4.rectSize

color: "black"

}

Rectangle

{

width: item4.rectSize

height: item4.rectSize

color: "black"

transform: [

Scale

{

xScale: 0.5

},

Rotation

{

// вращение происходит вокруг координаты (0; 0) прямоугольника (т.е. вокруг верхнего левого угла)

angle: 45

},

Translate

{

x: item4.rectSize \* 2.5

}

]

}

}

1. Нарисуем квадрат и анимировать его перемещение вниз с увеличением его размера. Для этого создадим вторую страницу в приложении (её id присвоим “page2”). Затем добавим следующий код внутрь её главного Column:

Item

{

id: item5;

width: page2.width

height: page2.height

Rectangle

{

id: rect5\_1

anchors.horizontalCenter: item5.horizontalCenter

color: "skyblue"

property int rectSize: 300

width: rectSize

height: rectSize

ParallelAnimation

{

id: parallelAnimation5\_1

running: true

loops: Animation.Infinite

property int animationDuration: 7500

NumberAnimation {

target: rect5\_1

property: "y"

from: -300

to: 1600

duration: parallelAnimation5\_1.animationDuration

}

NumberAnimation {

target: rect5\_1

property: "rectSize"

from: 300

to: 600

duration: parallelAnimation5\_1.animationDuration

}

}

}

}

1. Реализуем диалог с двумя текстовыми полями, в которые вводятся числа. После нажатия на кнопку “Подтвердить” в консоль выведем сумму чисел. Для преобразования строк к числам используем функцию parseInt.  
   Для этого создадим ещё один qml-файл, главным разделом которого будет теперь не “Page”, а “Dialog”:

Dialog {

id: dialog

property int currentResult: 0

Column

{

anchors.fill: parent

width: parent.width

spacing: Theme.paddingMedium

DialogHeader { } // для кнопок "подтвердить" и "отмена" наверху диалога

Row

{

width: parent.width

spacing: Theme.paddingMedium

TextField {

id: firstOperand

width: parent.width / 2.5

}

TextField {

id: secondOperand

width: parent.width / 2.5

}

}

Button

{

width: parent.width

text: qsTr("Сложить")

onClicked:

{

dialog.currentResult = parseInt(firstOperand.text) + parseInt(secondOperand.text)

}

}

}

onAccepted: console.log(dialog.currentResult)

}

1. **Заключение.**

Мы освоили базовые навыки построения пользовательских интерфейсов, позиционирования, отрисовки и перемещения элементов, научились анимировать элементы, а также научились создавать диалоги и взаимодействовать с ними.

**7. Литература.**

1. Документация Qt: сайт. — URL: https://doc.qt.io (дата обращения: 01.03.2022). — Текст: электронный.