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 №5:

«Инструменты разработки мобильных приложений»

Выполнила:

студентка группы 381906-2

Яшина Дарья Степановна

Нижний Новгород

2022

Постановка задачи

Цель: научиться организовывать многостраничное приложение, использовать контейнеры Silica, вытягиваемые меню и обложку приложения.

Шаги:

1. Создать приложение, которое будет отображать страницу с двумя кнопками “Назад” и “Вперёд”. Первая удалит текущую страницу со стека, вторая добавит новую. Также на экране нужно отображать текущую глубину стека.

2. Создать приложение из двух страниц. Первая страница содержит две кнопки “Добавить страницу” и “Убрать страницу”. Первая кнопка добавит вторую страницу как прикреплённую, вторая кнопка её удалит. На второй странице должна быть кнопка для возврата на первую страницу без закрытия второй.

3. Создать приложение с одной кнопкой и текстовом поле. После нажатия на кнопку отображается диалог для ввода текста. После согласия с результатом введённый текст отображается в текстовое поле.

4. Создать приложение с одной кнопкой и текстовым полем. После нажатия на кнопку отображается диалог для выбора даты. После согласия с результатом ввода выбранная дата отображается в текстовое поле.

5. Создать приложение с одной кнопкой и текстовым полем. После нажатия на кнопку отображается диалог для выбора времени. После согласия с результатом ввода выбранное время отображается в текстовом поле.

6. Создать приложение со списком SilicaListView, из задач на неделю. Задачи должны содержать дату и описание. В списке задачи группировать по датам.

7. Создать приложение с SilicaWebView для доступа к вашему любимому сайту.

8. Использовать SlideshowView для просмотра и перелистывания задач на неделю. На одном слайде – одна задача.

9. Создать приложение с вытягиваемыми меню сверху и снизу и текстовым полем. После выбора какого-либо элемента меню, его название отобразить в текстовом поле.

10. Создать приложение со списком и контекстным меню. После выбора элемента контекстного меню отобразить в консоли название выбранного элемента меню и индекс элемента списка.

Руководство программиста

Для реализации данной лабораторной работы нам потребовались следующие инструменты :

Task_1:

- **Button**

  text: "Назад" – название кнопки

  onClicked: pageStack.pop() – параметр, указывающий на то, что при нажатии на кнопку страница со стека удаляется

- **Button**

  text: "Вперед" – название кнопки

  onClicked: pageStack.push(Qt.resolvedUrl("FirstPage.qml")) – параметр, указывающий на то, что при нажатии на кнопку страница добавляется на стек

- **Label**

  text: "Глубина стека: " + pageStack.depth – текст лейбла, показывающий глубину стека

Task_2:

- **Button**

  text: "Вперед" – название кнопки

  onClicked: pageStack.pushAttached(Qt.resolvedUrl("MainPage.qml")) – параметр, указывающий на то, что при нажатии на кнопку к первой странице прикрепляется меню

- **Button**

  text: "Назад" – название кнопки

  onClicked: pageStack.popAttached() – параметр, указывающий на то, что при нажатии на кнопку вторая страница откреплyется

Task_3:

- **Button**

text: " Назад " – название кнопки

onClicked: pageStack.navigateBack() – параметр, указывающий на то, что при нажатии на кнопку происходит возвращение на первую страницу

- **PageHeader**

  title: "Страница" – название страницы

- **Button**

  text: " Открыть диалог" – название кнопки

  onClicked: dialog.open() – параметр, указывающий на то, что при нажатии на кнопку открывается диалог

- **Dialog**

  id: dialog – идентификатор диалога

  DialogHeader{} – параметр создающий диалог

  TextField { id: inText} – поле для ввода текста в диалоге

  onAccepted: outText.text = inText.text – при подтверждении текста диалога он выводится в текстовом поле

- **TextField**

  id: outText – идентификатор текствого поля

Task_4:

- **Button**

  text: " Открыть диалог" – название кнопки

  onClicked: dialog.open() – параметр, указывающий на то, что при нажатии на кнопку открывается диалог

- **DatePickerDialog**

  id: dialog – идентификатор диалога

  onAccepted: outText.text = inText.text – при подтверждении даты в диалоге она выводится в текстовом поле

- **TextField**

  id: outText – идентификатор текствого поле

Task_5:

- **Button**

  text: "Открыть диалог" – название кнопки

  onClicked: dialog.open() – параметр, указывающий на то, что при нажатии на кнопку открывается диалог

- **TimePickerDialog**

  id: dialog – идентификатор диалога

  onAccepted: outText.text = inText.text – при подтверждении времени в диалоге он выводится в текстовом поле

- **TextField**·

  id: outText – идентификатор текствого поле

Task_6:

- **ListModel**

  id: task – идентификатор модели

  ListElement { name: "Сходить в универ на пару"; date: "26.10.2022" } – элемент списка модели

- **SilicaListView**

  model: task – модель, входные данные

  header: PageHeader { title: "Задачи"} - заголовок

  section: - определяет выражение, которое надо оценить и внешний вид меток

  property – параметр определяющий свойство

  delegate - предоставляет шаблон, определяющий каждый элемент

Task_7:

- **SilicaWebView**

  id: webView - идентификатор

  url:"https://habr.com/ru/all/" - адрес сайта

  header: PageHeader – заголовок страницы

  title: "https://habr.com/ru/all/"– текст заголовка

Task_8:

- **ListModel**

  id: task – идентификатор модели

  ListElement { name: "Сходить в универ на пару"; date: "26.10.2022" } – элемент списка модели

- **SlideshowView**

  id: view – идентификатор

  model: task – модель, входные данные

  delegate - предоставляет шаблон, определяющий каждый элемент

Task_9:

- **SilicaFlickable**

  TextField

  id: tf – идентификатор текстового поле

- **PullDownMenu -** вытягиваемое меню сверху

  MenuItem – элемент меню

  text: "1" – название элемента

  onClicked: tf.text = text – при выборе элемента он появляется в текстовом поле

- **PushUpMenu -** вытягиваемое меню снизу

  MenuItem – элемент меню

  text: "Задача 3" – название элемента

  onClicked: tf.text = text – при выборе элемента он появляется в текстовом поле

Task_10**:**

- **ListModel**

  id: task – идентификатор модели

  ListElement { name: "Меню 1" } – элемент списка модели

- **SilicaListView**

model: task – модель, входные данные

delegate - предоставляет шаблон, определяющий каждый элемент

menu: ContextMenu – контекстное меню

MenuItem – элемент меню

text: "Элемент 1" – название элемента

onClicked: console.log(model + " " + text) – при выборе элемента он отображается в консоль

10. в Laba5.qml:

- **Cover**

id: cover - идентификатор

property int counter: 0 – переменная счетчик

CoverActionList – список действий обложки

CoverAction - действие

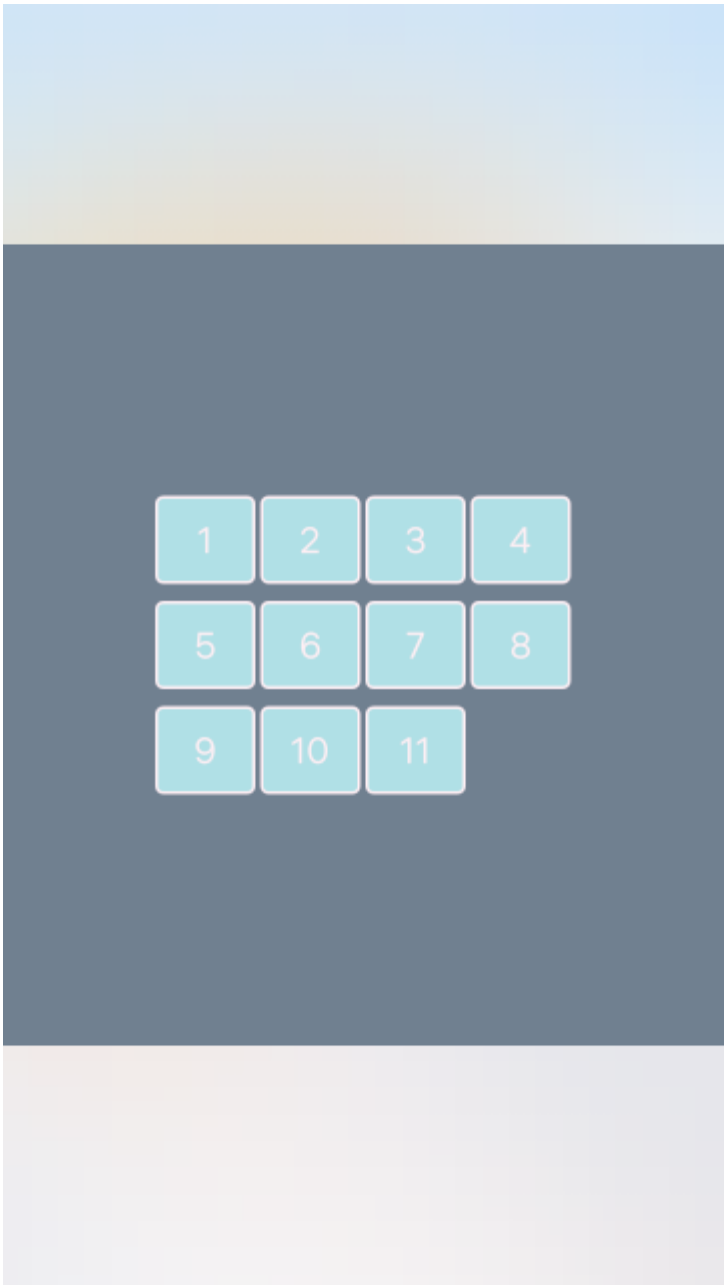
iconSource: "image://theme/icon-cover-cancel" - URL-адрес значка, отображаемого для этого действия обложки

onTriggered: cover.counter = 0 – обработчик сигнала

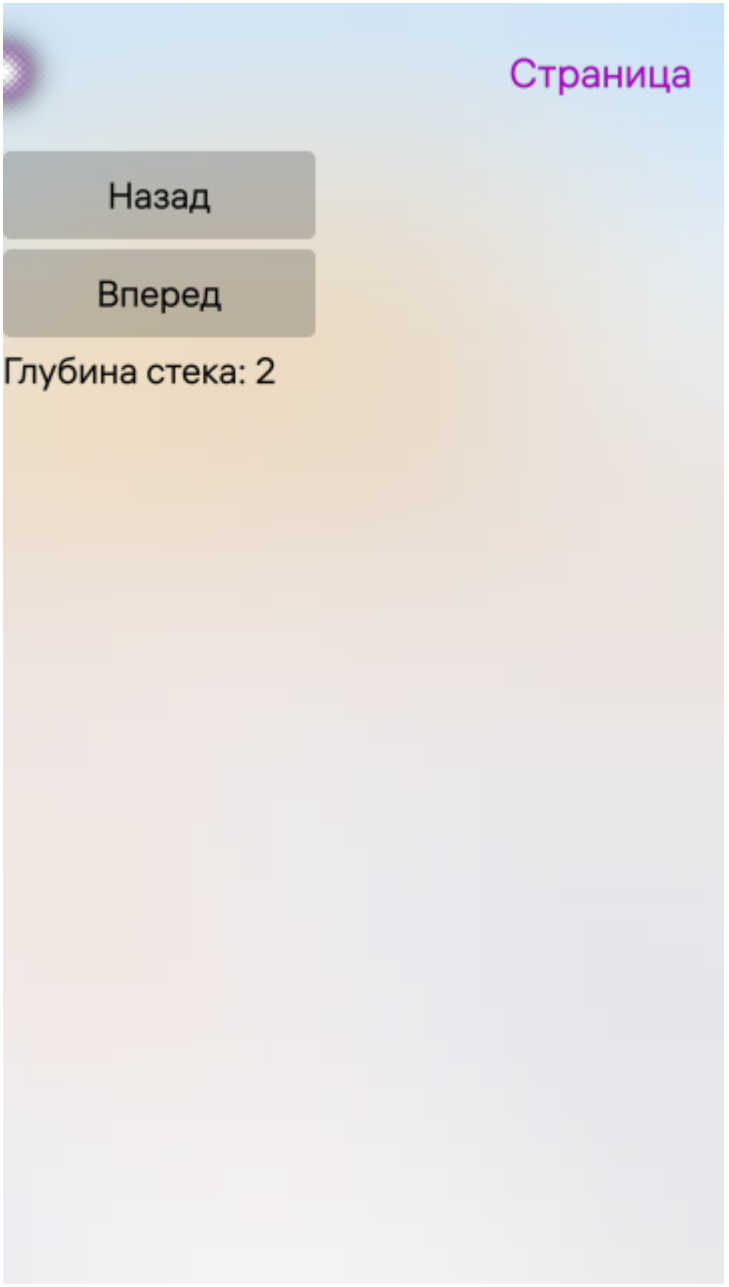
Руководство пользователя

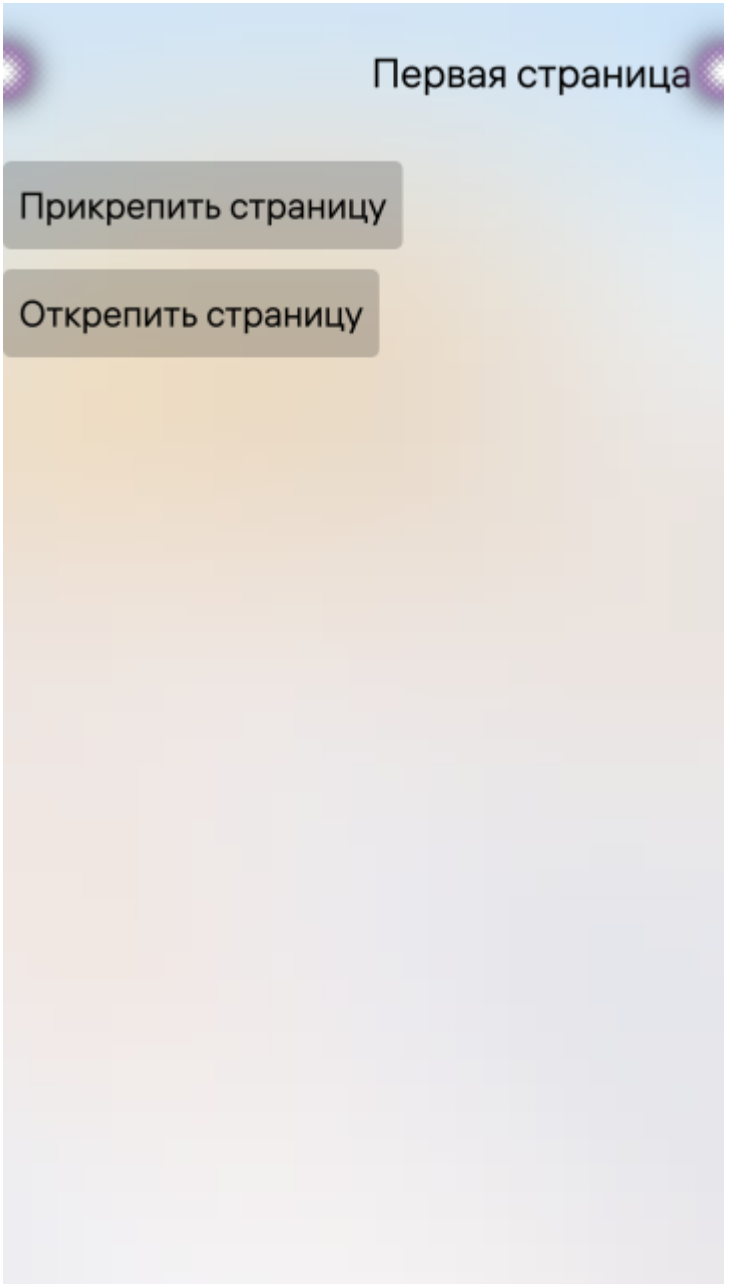
После запуска программы, открывается окно эмулятора, в котором отображается страница с кнопками. Каждая кнопка соответствует определенному номеру задания.

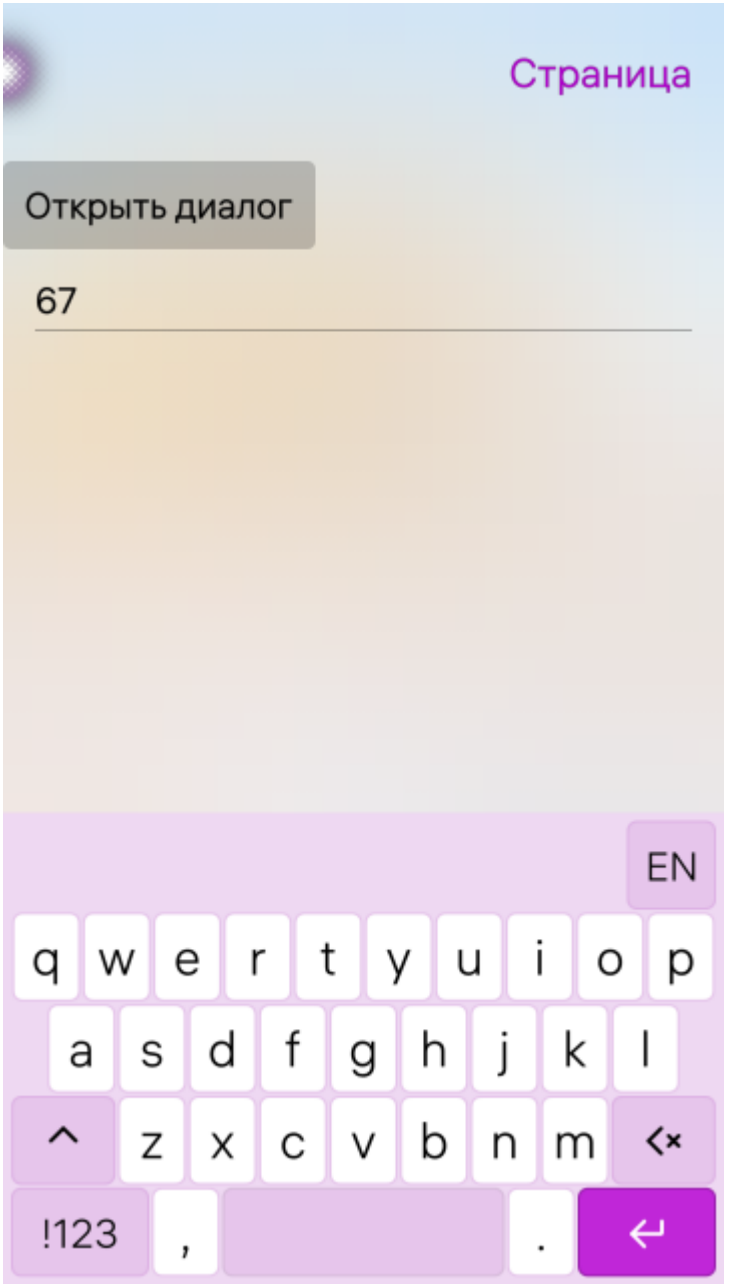
- Главная страница



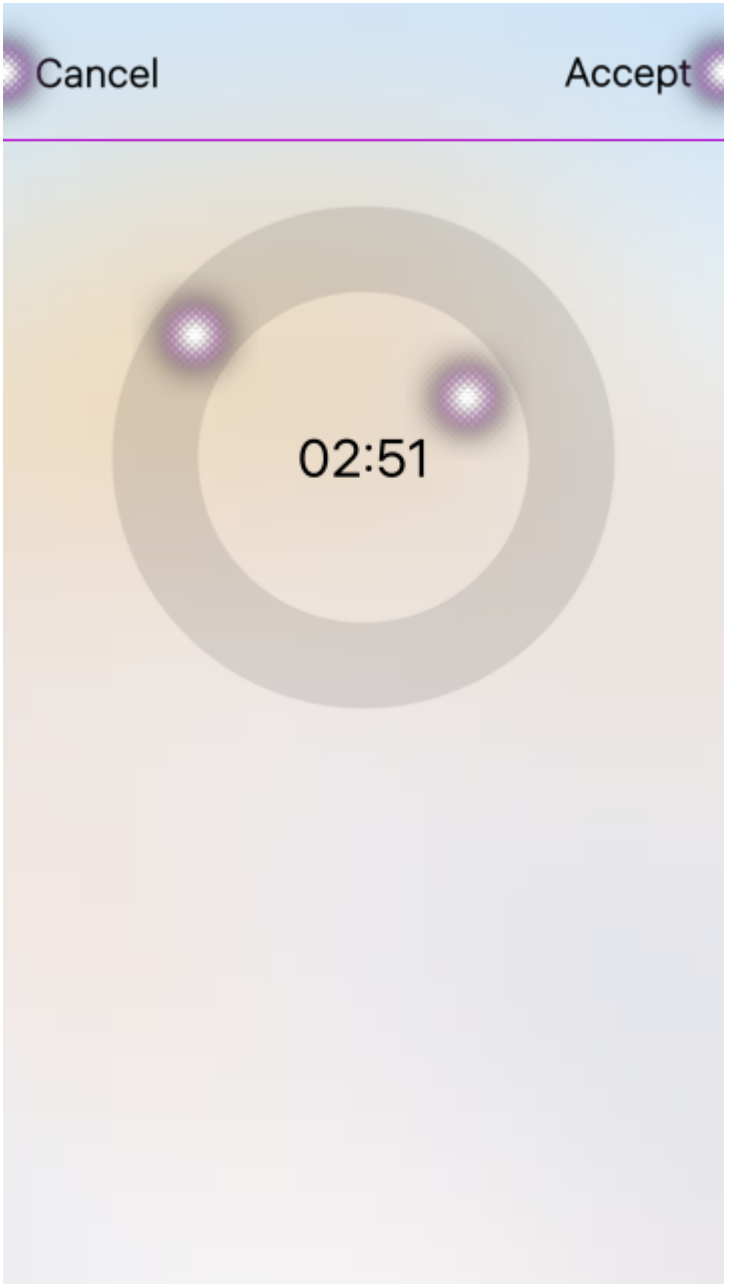
- Задания по порядку

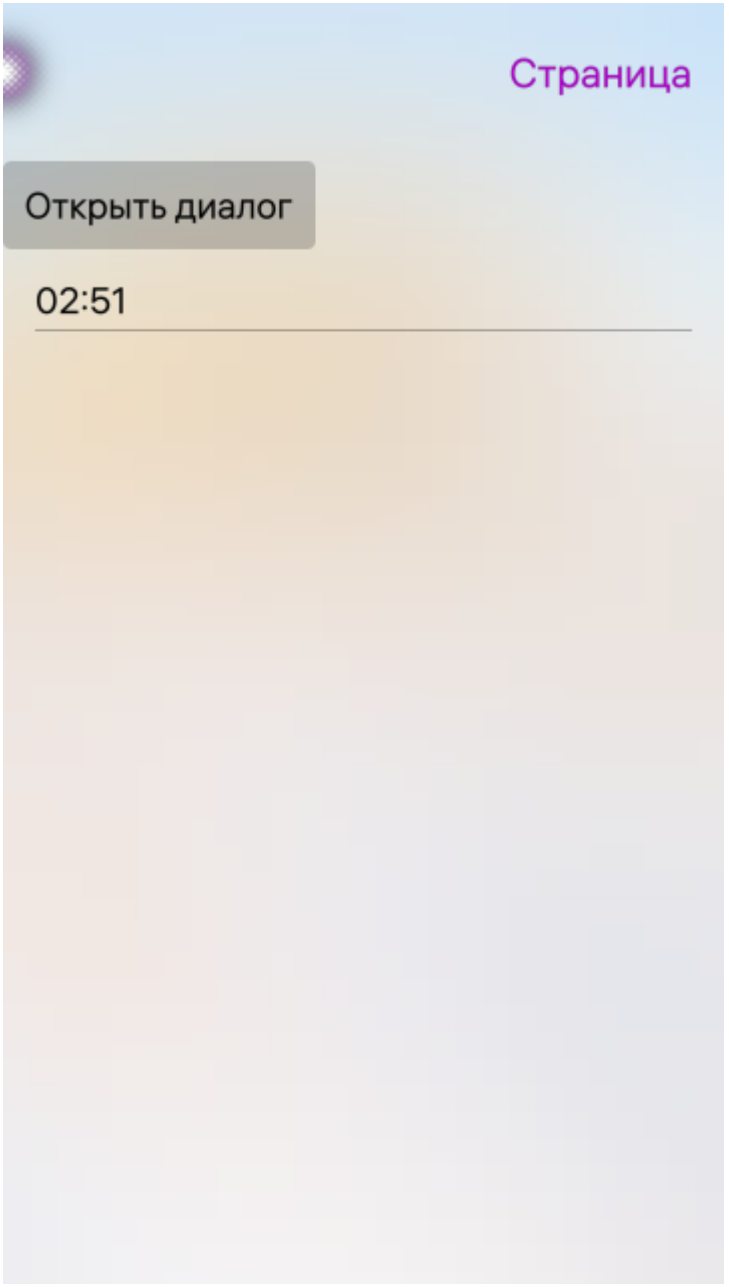

Страница
Назад
Вперед
Глубина стека: 2


Первая страница
Прикрепить страницу
Открепить страницу


Страница
Открыть диалог
67
EN
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
!123
,
.


2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048


Cancel
Accept
02:51


Страница
Открыть диалог
02:51

Расписание на неделю

26.10.2022

Сходить на пары в универ

25.10.2022

Поездка в Москву

24.10.2022

Сдать проект

23.10.2022

Посетить конференцию

Открыть список

Все потоки

СТАТЬИ +58 НОВОСТИ +54 ХАБЫ АВТОРЫ КОМПАНИИ

Все подряд Открыть список

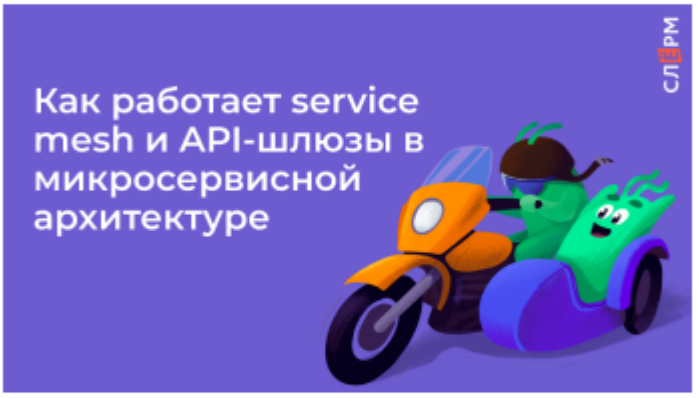
Anna_sokol22 сегодня в 18:33

Скоп
ссыл

Как работает service mesh и API-шлюзы в микросервисной архитектуре

Блог компании Southbridge , IT-инфраструктура *

Перевод



Вы наверняка много раз слышали о *service mesh* и *API-шлюзе* применительно к микросервисам. Их часто путают. В этой статье мы подробно поговорим о двух этих инструментах, а также разберемся, когда их лучше использовать и что будет, если их объединить.

Читать далее

Всего голосов 6: ↑6 и ↓0 +6

Просмотры 106 Доб.

Комментарии 0

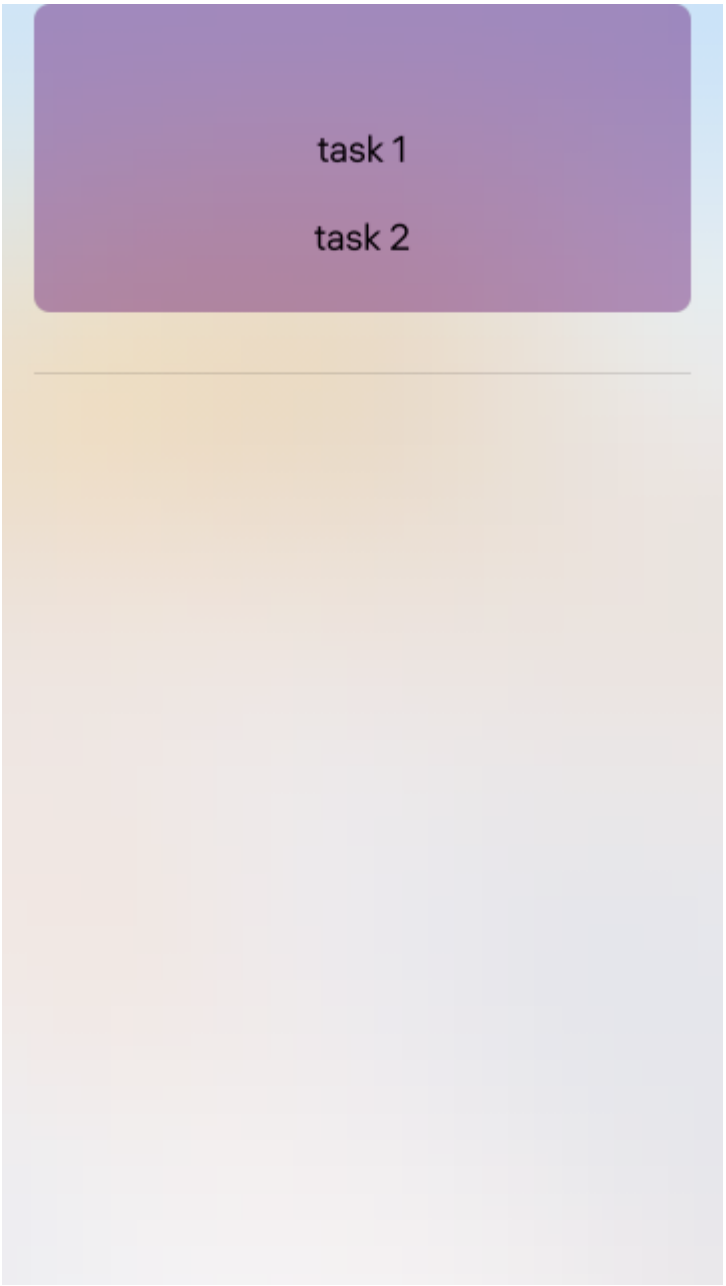
новости

все подряд лучшие

https://habr.com/ru/all/

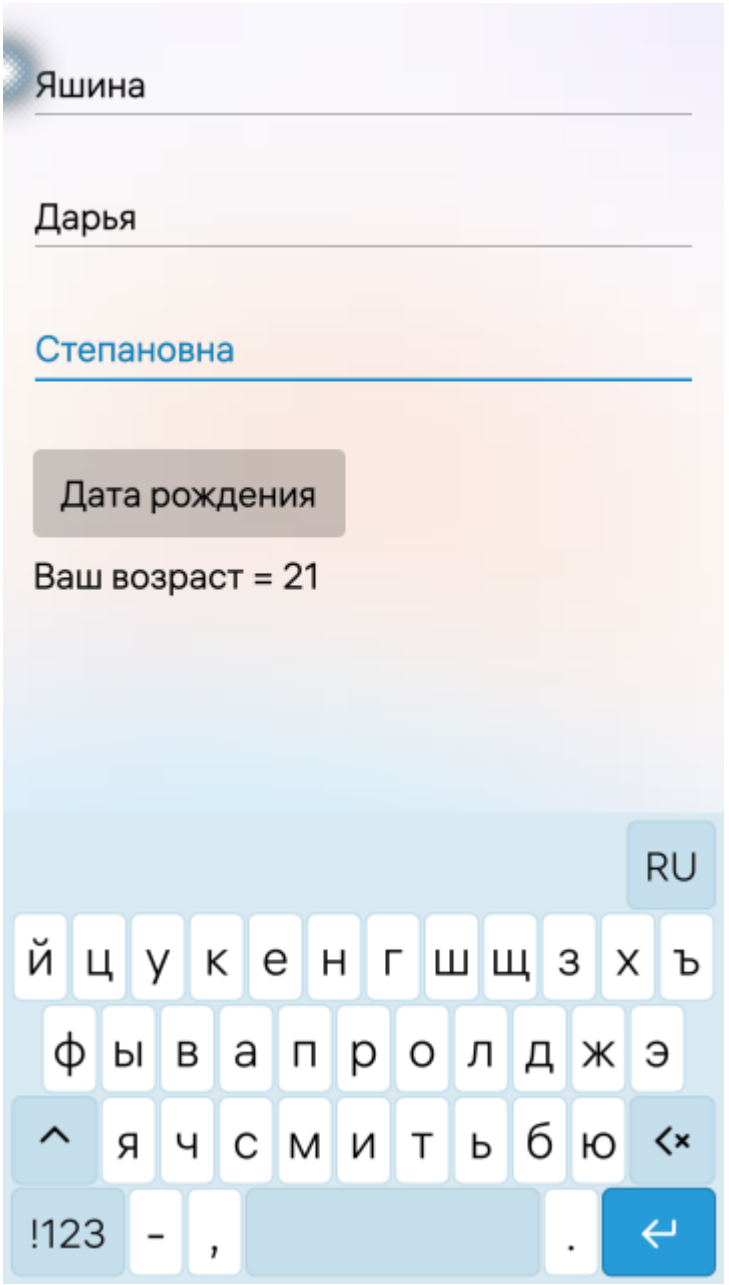
Все публикации подряд / Хабр

Сходить на пары в универ - 26.10.2022


task 1
task 2


task 2

Доп.задание:


Яшина
Дарья
Степановна
Дата рождения
Ваш возраст = 21
RU
й ц у к е н г ш щ з х ъ
ф ы в а п р о л д ж э
я ч с м и т ь б ю
!123 - ,
.

Приложение

Task_1

```
import QtQuick 2.0

import Sailfish.Silica 1.0

Page {
  id: page
  Column {
     spacing: 10
     anchors.fill: parent
     PageHeader {
        title: "Страница"
     }
     Button {
        text: "Назад"
        onClicked: pageStack.pop()
     }
     Button {
        text: "Вперед"
        onClicked: pageStack.push(Qt.resolvedUrl("MainPage.qml"))
     }
     Label {
        text: "Глубина стека: " + pageStack.depth
     }
  }
}
```

Task_2

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
  id: page
  Column {
     spacing: 20
     anchors.fill: parent
     PageHeader {
        title: "Первая страница"
     }
     Button {
        text: "Прикрепить страницу"
        onClicked: pageStack.pushAttached(Qt.resolvedUrl("Task_1.qml"))
     }
     Button {
        text: "Открепить страницу"
```

```
            onClicked: pageStack.popAttached()
        }
    }
}
```

Task_3

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    id: page
    Column {
        spacing: 20
        anchors.fill: parent
        PageHeader {
            title: "Страница"
        }
        Button {
            text: "Открыть диалог"
            onClicked: dialog.open()
        }
        Dialog {
            id: dialog
            Column {
                width: page.width
                DialogHeader{
                title: "Диалог"}
                TextField {
                    id: inText
                }
            }
            onAccepted: outText.text = inText.text
        }
        TextField {
            id: outText
        }
    }
}
```

Task_4

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    id: page
    Column {
        spacing: 20
```

```
        anchors.fill: parent
        PageHeader {
            title: "Страница"
        }
        Button {
            text: "Открыть диалог"
            onClicked: dialog.open()
        }
        //Диалог для выбора даты
        DatePickerDialog {
            id: dialog
            onAccepted: outText.text = dateText
        }

         TextField {
             id: outText
         }
    }
}
```

Task_5

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    id: page
    Column {
        spacing: 20
        anchors.fill: parent
        PageHeader {
            title: "Страница"
        }
        Button {
            text: "Открыть диалог"
            onClicked: dialog.open()
        }
        //Диалог для выбора времени
        TimePickerDialog {
            id: dialog
            onAccepted: outText.text = timeText
        }

         TextField {
             id: outText
         }
    }
}
```

Task_6

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
   id: page
   ListModel{
   id:list
   ListElement{name:"Сходить на пары в универ";date:"26.10.2022"}
   ListElement{name:"Поездка в Москву";date:"25.10.2022"}
   ListElement{name:"Сдать проект";date:"24.10.2022"}
   ListElement{name:"Посетить конференцию";date:"23.10.2022"}
   }
   SilicaListView{
   anchors.fill: parent
   model:list
   spacing: 10
   header:PageHeader{title: "Расписание на неделю"}
   section{
      property:"date"
      delegate: BackgroundItem {
         PageHeader{title: section}
       }

   }
   delegate: BackgroundItem {
      Label {
         text: name
          }
         }

   }
}
```

Task_7

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
   id: page

   SilicaWebView{
   id: web
   anchors{
      top:parent.top;bottom:urlField.top;
```

```
        left:parent.left;right:parent.right;
    }
    url:"https://habr.com/ru/all/"
    }

    TextField{
    id:urlField
    anchors{
    left:parent.left;right:parent.right;
    bottom:parent.bottom
    }
    text:"https://habr.com/ru/all/"
    label:web.title
    EnterKey.onClicked: web.url = text
    }

}
```

Taak_8

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    id: page
    ListModel{
    id:list
    ListElement{name:"Сходить на пары в универ";date:"26.10.2022"}
    ListElement{name:"Поездка в Москву";date:"25.10.2022"}
    ListElement{name:"Сдать проект";date:"24.10.2022"}
    ListElement{name:"Посетить конференцию";date:"23.10.2022"}
    }
    SlideshowView{
    id:view
    anchors.centerIn: parent
    anchors.fill:parent
    model:list
    delegate: Rectangle {
        width: view.itemWidth;
        height: view.itemHeight;
        Text {
            anchors.centerIn: parent
            text: name + " - " + date
        }
    }

    }
}
```

Task_9

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
  id: page
  SilicaFlickable {
     anchors.fill: parent
     TextField {
        id: t
     }
     PullDownMenu {
        MenuItem {
           text: "task 1"
           onClicked: t.text = text
        }
        MenuItem {
           text: "task 2"
           onClicked: t.text = text
        }
     }
     PushUpMenu {
        MenuItem {
           text: "task 3"
           onClicked: t.text = text
        }
        MenuItem {
           text: "task 4"
           onClicked: t.text = text
        }

     }
  }
}
```

Task_10

```
import QtQuick 2.2
import Sailfish.Silica 1.0

Page {
  id: page

  ListModel {
     id: task
     ListElement { name: "Menu 1" }
     ListElement { name: "Menu 2" }
```

```
        ListElement { name: "Menu 3" }
        ListElement { name: "Menu 4" }
        }

    SilicaListView {
        anchors.fill: parent
        model: task
        delegate: ListItem {
            Label {
                text: model.name
                anchors.centerIn: parent
            }
        menu: ContextMenu {
            MenuItem {
                text: "1"
                onClicked: console.log(model + " " + text)
            }
            MenuItem {
                text: "2"
                onClicked: console.log(model + " " + text)
            }
        }
    }
  }
}
```

Task_11

```
import QtQuick 2.0
import Sailfish.Silica 1.0

Page {
    id: page
    allowedOrientations: Orientation.All
    Column {
        anchors.fill: parent
        anchors.topMargin: 50
        spacing: 15
        TextField {
            id: surname
            placeholderText: "Введите фамилию"
        }
        TextField {
            id: name
            placeholderText: "Введите имя"
        }
        TextField {
```

```
            id: patronym
            placeholderText: "Введите отчество"
        }
        Button {
            anchors.left: parent.left
            anchors.leftMargin: 30
            text: "Дата рождения"
            onClicked: dialog.open()
        }
        Label {
            id: ageLabel
            property int value: 0
            anchors.left: parent.left
            anchors.leftMargin: 30
            text: "Ваш возраст = " + value
        }
    }

    DatePickerDialog {
        id: dialog
        onAccepted: {
            var currentDate = new Date();
            var dateBorn = date;
            var age = currentDate.getFullYear() - dateBorn.getFullYear();
            if (currentDate.getMonth() <= dateBorn.getMonth() &&
                    currentDate.getDate() < dateBorn.getDate()) {
              age -= 1;
            }
            ageLabel.value = parseInt(age)
        }
    }

}
```