Министерство образования и науки Российской Федерации

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

«Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Кафедра информационных систем

и программной инженерии

**Лабораторная работа № 6-7**

**по дисциплине**

**«Технологии разработки моильных приложений»**

**Тема: Работа с Canvas. Обработка касаний экрана**

Выполнила:

ст. гр. ПРИ-117

А.И. Рощина

Принял:

Козаев А.Т.

Владимир, 2020

1. ЦЕЛЬ РАБОТЫ

Разработать мобильное приложение, которое будет давать возможность рисовать простейшие геометрические фигуры. Разработать мобильное приложение, которое будет обрабатывать нажатия на экран.

1. ВЫПОЛНЕНИЕ РАБОТЫ

Репозиторий проекта:

[https://github.com/flamebrier/tdma/tree/main/lab6](https://github.com/flamebrier/tdma/tree/main/lab11)

[https://github.com/flamebrier/tdma/tree/main/lab](https://github.com/flamebrier/tdma/tree/main/lab11)7

***Задания 6***

1. Выполнить пример из лабораторной работы.

2.1. Нарисовать пятиконечную звезду

***Задания 7***

1. Выполнить пример из лабораторной работы.

2.1. Используя предыдущую лабораторную работу реализовать

рисование.

***Ход работы***

Выполнение примера по рисованию.

1. Заполнение цветом (рисунок 1).

Код:

import 'package:flutter/material.dart';

class LabPainter extends CustomPainter {

@override

void paint(Canvas canvas, Size size) {

canvas.drawPaint(Paint()..color = Colors.green);

}

@override

bool shouldRepaint(covariant CustomPainter oldDelegate) => true;

}

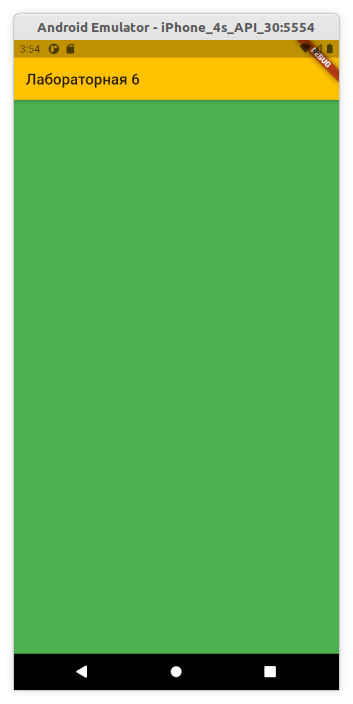


Рисунок 1 - Заполнение цветом

1. Рисование примитивов (рисунок 2).

Код:

import 'dart:ui';

import 'package:flutter/material.dart';

class LabPainter extends CustomPainter {

// настройки рисования

final p = Paint()

..color = Colors.blue.shade900

..strokeWidth = 10

..strokeCap = StrokeCap.round;

Rect rect = Rect.fromLTRB(200, 150, 400, 200);

@override

void paint(Canvas canvas, Size size) {

//фон

canvas.drawPaint(Paint()..color = Color.fromARGB(80, 102, 204, 255));

// точка

canvas.drawPoints(PointMode.points, [Offset(50, 50)], p);

// линия

canvas.drawLine(Offset(100, 100), Offset(500, 50), p);

// круг

canvas.drawCircle(Offset(100, 200), 50, p);

// прямоугольник

canvas.drawRect(rect, p);

// изменение прямоугольника

rect = Rect.fromLTRB(250, 300, 350, 500);

canvas.drawRect(rect, p);

}

@override

bool shouldRepaint(covariant CustomPainter oldDelegate) => true;

}

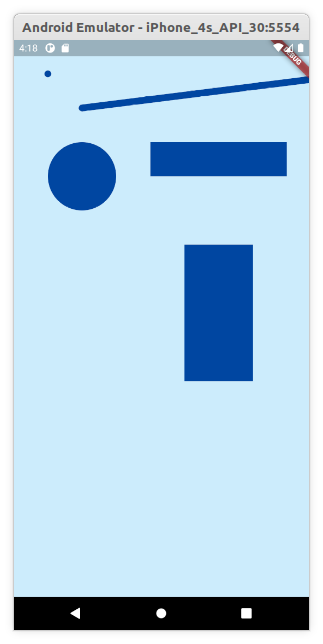


Рисунок 2 - Рисование примитивов

1. Рисование других объектов (рисунок 3).

Код:

import 'dart:math';

import 'dart:ui';

import 'package:flutter/material.dart';

class LabPainter extends CustomPainter {

// настройки рисования

Paint p = Paint()

..color = Colors.blue.shade900

..strokeWidth = 10

..strokeCap = StrokeCap.round;

Rect rect = Rect.fromLTRB(700, 30, 800, 80);

List<Offset> points = [

Offset(100, 50),

Offset(150, 100),

Offset(150, 200),

Offset(50, 200),

Offset(50, 100)

];

List<Offset> points1 = [

Offset(300, 200),

Offset(600, 200),

Offset(300, 300),

Offset(600, 300),

Offset(400, 100),

Offset(400, 400),

Offset(500, 100),

Offset(500, 400)

];

@override

void paint(Canvas canvas, Size size) {

canvas.drawPaint(Paint()..color = Color.fromARGB(80, 102, 204, 255));

canvas.drawPoints(PointMode.points, points, p);

for (int i = 0; i < points1.length; i += 2)

canvas.drawLine(points1[i], points1[i + 1], p);

p = p..color = Colors.green;

canvas.drawRRect(RRect.fromRectAndRadius(rect, Radius.circular(20)), p);

rect = rect.translate(0, 100);

canvas.drawOval(rect, p);

rect = rect.translate(0, 100);

canvas.drawArc(rect, -pi / 2, -3 \* pi / 2, true, p);

rect = rect.translate(0, 100);

canvas.drawArc(rect, -pi / 2, -3 \* pi / 2, false, p);

canvas.drawLine(Offset(150, 450), Offset(150, 600), p);

var tp1 = TextPainter(

text: TextSpan(text: "text start", style: TextStyle(color: Colors.blue.shade900)), textAlign: TextAlign.start, textDirection: TextDirection.ltr);

var tp2 = TextPainter(

text: TextSpan(text: "text center", style: TextStyle(color: Colors.blue.shade900)), textAlign: TextAlign.center, textDirection: TextDirection.ltr);

var tp3 =

TextPainter(text: TextSpan(text: "text end", style: TextStyle(color: Colors.blue.shade900)), textAlign: TextAlign.end, textDirection: TextDirection.ltr);

tp1.layout(minWidth: 150);

tp2.layout(minWidth: 150);

tp3.layout(minWidth: 150);

tp1.paint(canvas, Offset(50, 300));

tp2.paint(canvas, Offset(50, 320));

tp3.paint(canvas, Offset(50, 340));

}

@override

bool shouldRepaint(covariant CustomPainter oldDelegate) => true;

}

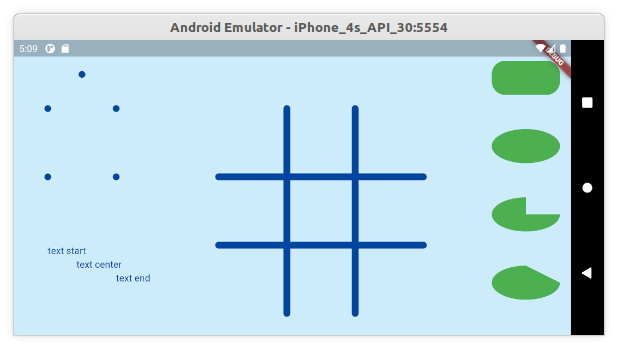


Рисунок 3 - Рисование других объектов

1. Рисование звезды (рисунок 4).

Код:

import 'dart:ui';

import 'package:flutter/material.dart';

class LabPainter4 extends CustomPainter {

// настройки рисования

Paint p = Paint()

..color = Colors.blue.shade900

..strokeWidth = 10

..strokeCap = StrokeCap.round;

Path star = Path();

@override

void paint(Canvas canvas, Size size) {

canvas.drawPaint(Paint()..color = Color.fromARGB(80, 102, 204, 255));

star.moveTo(300, 100);

star.relativeLineTo(-25, 50);

star.relativeLineTo(-50, 0);

star.relativeLineTo(50, 25);

star.relativeLineTo(-25, 50);

star.relativeLineTo(-25, -50);

star.relativeLineTo(25, 50);

star.relativeLineTo(50, -35);

star.relativeLineTo(50, 35);

star.relativeLineTo(-25, -50);

star.relativeLineTo(50, -25);

star.relativeLineTo(-50, 0);

canvas.drawPath(star, p);

}

@override

bool shouldRepaint(covariant CustomPainter oldDelegate) => true;

}

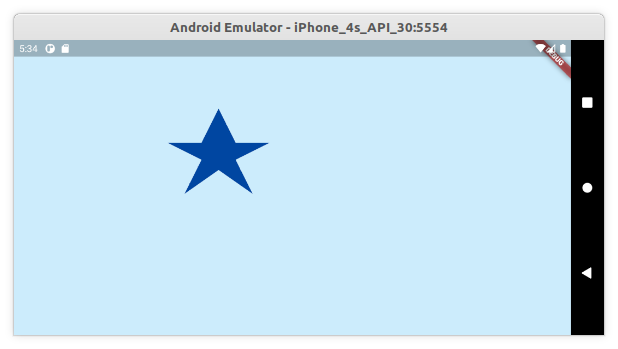


Рисунок 4 - Рисование звезды

Задания 7.

1. Пример

Код:

class MyHomePage extends StatefulWidget {

MyHomePage({Key key, this.title}) : super(key: key);

final String title;

@override

\_MyHomePageState createState() => \_MyHomePageState();

}

class \_MyHomePageState extends State<MyHomePage> {

int \_downCounter = 0;

int \_upCounter = 0;

int \_pointerCount = 0;

List<String> \_pointers = [];

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text(widget.title),

),

body: Listener(

onPointerDown: (details) {

if (mounted)

setState(() {

\_downCounter++;

});

},

onPointerUp: (details) {

if (mounted)

setState(() {

\_upCounter++;

});

},

onPointerMove: (details) {

if (mounted)

setState(() {

\_pointers.add(

"index = ${details.pointer}, ID = ${details.embedderId}, X = ${details.localPosition.dx}, Y = ${details.localPosition.dy}");

});

},

child: Center(

child: Container(

width: double.infinity,

height: double.infinity,

child: Column(

mainAxisSize: MainAxisSize.max,

children: <Widget>[

ElevatedButton(

onPressed: () => Navigator.push(context,

MaterialPageRoute(builder: (context) => PaintPart())),

child: Text("Paint Something")),

Text(

'down: $\_downCounter',

),

Text(

'up: $\_upCounter',

),

Text(

'pointerCount: $\_pointerCount',

),

Expanded(

child: ListView.builder(

itemCount: \_pointers.length,

shrinkWrap: true,

itemBuilder: (context, index) {

return Text(\_pointers[index]);

}))

],

),

),

)),

);

}

}

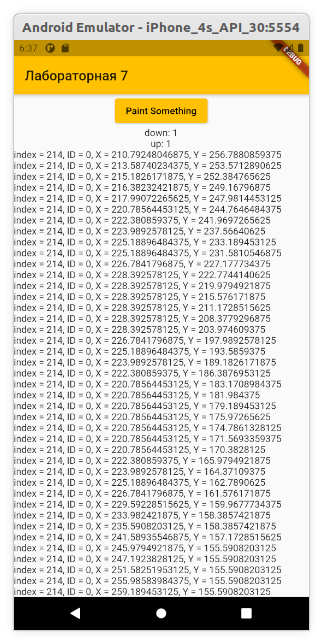


Рисунок 5 - Вывод данных о касаниях

1. Рисование

Код:

import 'dart:ui';

import 'package:flutter/material.dart';

class PaintPart extends StatefulWidget {

@override

\_PaintPartState createState() => \_PaintPartState();

}

class \_PaintPartState extends State<PaintPart> {

final List<Offset> points = List.empty(growable: true);

@override

Widget build(BuildContext context) {

return Listener(

onPointerMove: (details) {

if (mounted)

setState(() {

points.add(details.localPosition);

});

},

child: CustomPaint(painter: PaintPainter(points)),

);

}

}

class PaintPainter extends CustomPainter {

final List<Offset> points;

PaintPainter(this.points);

@override

void paint(Canvas canvas, Size size) {

canvas.drawPaint(Paint()..color = Colors.grey.shade900);

if ((points ?? []).isNotEmpty) {

canvas.drawPoints(

PointMode.points,

points,

Paint()

..color = Colors.amber

..strokeWidth = 10

..strokeCap = StrokeCap.round

..strokeJoin = StrokeJoin.round);

}

}

@override

bool shouldRepaint(covariant CustomPainter oldDelegate) => true;

}



Рисунок 6 - Рисование

1. ВЫВОД

В процессе выполнения лабораторной работы было разработано приложение, позволяющее рисовать простейшие геометрические фигуры и приложение, обрабатывающее нажатия на экран.