Министерство науки и высшего образования Республики Казахстана

Некоммерческое акционерное общество

«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ имени Гумарбека Даукеева»

Институт автоматизации и информационных технологий

Кафедра IT-инженерии и искусственного интеллекта

# **ОТЧЕТ**

# **по лабораторной работе №4**

# По дисциплине: Разработка программных приложений на базе Android

На тему: Анимация

Образовательная программа: 6B06103 – Вычислительная техника и программное обеспечение

Выполнил: Кенес А. Ж. Группа ВТ(ПИ)у-23-4

Проверила: Старший преподаватель Ахсутова А.А.

\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ «\_\_\_\_» \_\_\_\_\_\_\_\_2024 г.

Алматы 2024

# **Скриншоты приложение**



Рисунок 1 – Скриншот приложение на базе Android

# **Листинг кода:**

@Composable

fun AnimationsScreen(

modifier: Modifier = Modifier

) {

var colorClicked by remember { mutableStateOf(false) }

val color by animateColorAsState(

targetValue = if (colorClicked) Color.Red else Color.Blue,

animationSpec = tween(durationMillis = 1000)

)

var visible by remember { mutableStateOf(true) }

var expanded by remember { mutableStateOf(false) }

val size by animateDpAsState(

targetValue = if (expanded) 40.sp.value.dp else 20.sp.value.dp

)

var transformIsToggle by remember { mutableStateOf(false) }

val transform by animateDpAsState(

targetValue = if (transformIsToggle) 200.dp else 0.dp,

animationSpec = if (transformIsToggle) spring(dampingRatio = 0.3f) else spring(dampingRatio = 1f)

)

var rotationYIsToggle by remember { mutableStateOf(false) }

val rotationY by animateFloatAsState(

targetValue = if (rotationYIsToggle) 180f else 0f,

animationSpec = tween(durationMillis = 1000, easing = LinearEasing),

)

var rotationZIsToggle by remember { mutableStateOf(false) }

val rotationZ by animateFloatAsState(

targetValue = if (rotationZIsToggle) 360f else 0f,

animationSpec = tween(durationMillis = 1000, easing = LinearEasing),

)

Column(

horizontalAlignment = Alignment.CenterHorizontally,

verticalArrangement = Arrangement.Center,

modifier = modifier.fillMaxSize()

) {

Box(

modifier = Modifier

.fillMaxWidth()

.fillMaxHeight(0.6f),

) {

Column(

modifier = Modifier.fillMaxSize(),

horizontalAlignment = Alignment.CenterHorizontally,

verticalArrangement = Arrangement.Center

) {

AnimatedVisibility(

visible = visible,

enter = fadeIn(),

exit = fadeOut()

) {

Text(

text = "Lab 4 Kenes Aset",

color = color,

fontSize = size.value.sp,

modifier = Modifier

.padding(top = transform)

.graphicsLayer {

if (rotationZIsToggle) {

transformOrigin = TransformOrigin(0f, 0.5f)

this.rotationZ = rotationZ

} else {

TransformOrigin(0.5f, 0.5f)

}

}

)

}

}

}

Box(

modifier = Modifier

.fillMaxWidth()

.fillMaxHeight(0.4f)

) {

Column {

Row(modifier = Modifier.fillMaxWidth()) {

Button(onClick = { expanded = !expanded }) {

Text("Size")

}

Button(onClick = { visible = !visible }) {

Text("visibility")

}

}

Row(modifier = Modifier.fillMaxWidth()) {

Button(onClick = { transformIsToggle = !transformIsToggle }) {

Text("Transform")

}

Button(onClick = { colorClicked = !colorClicked }) {

Text("Color")

}

Button(onClick = {

rotationZIsToggle = !rotationZIsToggle

rotationYIsToggle = false

}) {

Text("Rotation Z")

}

}

}

}

}

}

Этот код написан на языке Jetpack Compose и представляет собой экран с несколькими анимациями. Он включает различные кнопки для изменения цвета текста, его видимости, размера, трансформации и вращения.

Основные компоненты:

1. AnimationsScreen: это главная функция, которая определяет пользовательский интерфейс (UI) с анимациями и кнопками.

2. Анимации:

- Цвет текста: переменная colorClicked управляет изменением цвета текста с помощью функции animateColorAsState, которая анимирует переход между красным и синим цветами.

- Размер текста: переменная expanded управляет размером текста с помощью функции animateDpAsState. Размер изменяется от 20.sp до 40.sp в зависимости от состояния.

- Трансформация по вертикали: переменная transformIsToggle управляет отступом текста сверху, анимация проходит с помощью spring (пружинная анимация).

- Вращение вокруг оси Z: переменная rotationZIsToggle управляет вращением текста вокруг оси Z (по часовой стрелке), используя функцию animateFloatAsState.

- Видимость текста: переменная visible управляет отображением текста с помощью AnimatedVisibility, которая включает эффекты появления и исчезновения (fadeIn и fadeOut).

3. UI компоненты:

- Внутри Column (вертикальный контейнер) есть несколько кнопок, которые меняют состояние переменных. Каждая кнопка вызывает анимацию: изменение размера, изменение видимости, трансформация по вертикали, изменение цвета, вращение.

Как это работает:

- Цвет: нажимая на кнопку "Color", вы переключаете цвет текста между красным и синим.

- Размер: нажатие на "Size" увеличивает или уменьшает размер текста.

- Трансформация: нажатие на "Transform" изменяет отступ сверху текста, создавая эффект сдвига текста вниз.

- Вращение: нажатие на "Rotation Z" вращает текст вокруг оси Z на 360 градусов.

- Видимость: нажатие на "Visibility" скрывает или показывает текст.

Это упражнение позволяет понять, как использовать анимации в Jetpack Compose для создания интерактивных и динамичных пользовательских интерфейсов.

# **Заключение**

В этом проекте на Jetpack Compose были исследованы возможности создания анимаций и управления различными аспектами пользовательского интерфейса, такими как цвет, размер, видимость и трансформация элементов. Пример демонстрирует, как эффективно применять функции animate\*AsState для создания плавных анимаций и динамичных взаимодействий с пользователем. Использование различных анимационных спецификаций, таких как tween и spring, показало, как можно контролировать скорость и физику анимаций.

Основной вывод заключается в том, что Jetpack Compose предоставляет гибкий и мощный инструментарий для создания анимаций, которые легко интегрируются с состоянием приложения, делая интерфейсы более отзывчивыми и интерактивными.

**Список литературы**

1. Динамически меняем параметры элементов экрана в андроид-приложении | Android Studio. URL: <https://www.fandroid.info/urok-18-dinamicheski-menyaem-parametry-elementov-ekrana-v-android-prilozhenii-android-studio/>

2. Делаем приложение: калькулятор для андроид | Android Studio. URL: <https://www.fandroid.info/urok-19-delaem-prilozhenie-kalkulyator-dlya-android-android-studio/>