Петрозаводский государственный университет
Институт математики и информационных технологий
Кафедра информатики и математического обеспечения

09.03.04 – Программная инженерия

Профиль направления подготовки бакалавриата

Системное и прикладное программное обеспечение

Отчет о проекте по курсу 'Разработка приложений для мобильных ОС'

Разработка мобильной игры

Выполнил:
студент 2 курса группы 22207

Н.Н. Анисимов

Петрозаводск — 2022

Содержание

Введение

Моей задачей было разработать мобильную игру - "ферму где необходимо покупать игровые объекты, приносящие деньги, а так же поддерживать их жизнедеятельность периодическими заходами в игру.

Разработка велась на Android Studio.

1 Требования

В игре должна быть реализована основная механика 'покупка - поддержка - доход' , данные должны запоминаться при выходе из приложения, чтобы пользователь не был обязан беспрерывно в нем находится, должно быть реализовано музыкальное сопровождение, должен быть сделан некоторый дизайн.

2 Проектирование

Для хранения данных будем использовать SharedPreferences

Для воспроизведения музыки MediaPlayer

Создадим класс Farm который будет отвечать за генерацию на поле купленных персонажей(класс Guy), их характеристики, накопленные деньги, сохранение и загрузку данных. Для обновления характеристик во времени будут использоваться потоки.

Так же в основном цикле программы будет происходить сохранение данных, в том числе и текущего времени. При загрузке игры будет вычисляться сколько прошло времени и в соотвествии с этим обновляться характеристики.

В классе MainActivity будут реализованы обработка кнопок, воспроизведение музыки и запуск функций Farm при загрузке игры.

3 Реализация

Листинг 1 – Основной цикл программы

```
var money: Float = 20F;
var guys:MutableList<Guy> = mutableListOf();
var food: Int = 0
fun go(context: Context)
```

```
{
    while(true)
    {
        val toDel:MutableList<Guy> = mutableListOf()
        for(curGuy in guys)
        {
            curGuy.hungry++;
            if(money < 9999F)
            money += when(curGuy.type) {
                "PredatorGuy" -> 0.04F
                "MaleGuy" -> 0.02F
                "FemaleGuy" -> 0.03F
                else -> 0.01F
            }
            money = min(money, 9999F)
            if(curGuy.hungry >= 5000)
                toDel.add(curGuy)
            else if(curGuy.hungry >= 2000)
                curGuy.showCloud(context)
        }
        for(delGuy in toDel)
        {
            delGuy.delete(context)
            guys.remove(delGuy)
        }
        save(context)
        Thread.sleep(1800);
    }
}
```

Листинг 2 – Сохранение и загрузка данных

```
private fun save(context: Context){
    try {
        val guysCopy = guys
        val sv = context.getSharedPreferences(
            "DATA", Context.MODE_PRIVATE)
        val ed = sv.edit()
        ed.putInt("FOOD", food)
        ed.putFloat("MNY", money)
        ed.putInt("NUM", guysCopy.size)
        ed.putLong("TIME", System.currentTimeMillis())
        for (i in 0 until guysCopy.size) {
            ed.putLong("food$i", guysCopy[i].hungry)
            ed.putInt("x$i", guysCopy[i].X)
            ed.putInt("y$i", guysCopy[i].Y)
            ed.putString("type$i", guysCopy[i].type)
        }
        ed.apply()
    }
    finally{}
}
fun load(context: Context) {
    try {
        val sv = context.getSharedPreferences(
            "DATA", Context.MODE_PRIVATE)
        food = sv.getInt("FOOD", 0)
        money = sv.getFloat("MNY", 20F)
        val num = sv.getInt("NUM", 0)
        val time = sv.getLong("TIME", System.currentTimeMillis())
        for(guy in guys) guy.delete(context)
        guys.clear()
        for (i in 0 until num) {
            val guy = Guy(
```

```
                    sv.getString("type$i", "").toString(), context,
                    sv.getInt("x$i", 0), sv.getInt("y$i", 0)
                )
                val dop = (System.currentTimeMillis() - time) / 1800
                val dop2 = min(max(5000 - sv.getLong("food$i", 5000), 0), dop)
                money += when (guy.type) {
                    "PredatorGuy" ->
                        0.04F * dop2
                    "MaleGuy" -> 0.02F * dop2
                    "FemaleGuy" -> 0.03F * dop2
                    else -> 0.01F * dop2
                }
                guy.hungry = sv.getLong("food$i", 0) + dop
                guys.add(guy)
            }
        }
        finally{}
    }
```

4 Заключение

У меня получилось разработать необходимое приложение, отвечающее всем требованиям.