### 

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC VĂN HIẾN**

**KHOA CÔNG NGHỆ - THÔNG TIN**

****

**Lập Trình Andoird**

**ĐỀ TÀI:**

**Lập trình game 2D Nutty Acorn**

**GVHD: Nguyễn Thị Phương Trang**

**SVTH: Lê Nguyễn Hoàn Huy**

**MSSV: 211A290034 LỚP: 211A2901**

## 

## 

**HỒ CHÍ MINH - 2021**

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

**NHẬN XÉT CỦA GIẢNG VIÊN**

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………..…………………………………………………………..………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

**Mục lục**

[I. Giới thiệu 4](#_Toc184573844)

[1.1. Giới thiệu về sản phẩm và ý tưởng thiết kế 4](#_Toc184573845)

[1.2. Lý do chọn đề tài 4](#_Toc184573846)

[1.3. Thiết kế 4](#_Toc184573847)

[II. Sử dụng môi trường và công nghệ 4](#_Toc184573848)

[2.1. Công cụ và môi trường phát triển 4](#_Toc184573849)

[2.2. Ngôn ngữ và framework 4](#_Toc184573850)

[2.3. Thư viện và công cụ hỗ trợ 5](#_Toc184573851)

[III. Chức năng chính của trò chơi 5](#_Toc184573852)

[3.1. Điều khiển nhân vật 5](#_Toc184573853)

[3.2. Chướng ngại vật 5](#_Toc184573854)

[3.3. Tính điểm 6](#_Toc184573855)

[3.4. Âm thanh 6](#_Toc184573856)

[IV. Triển khai chi tiết 6](#_Toc184573857)

[4.1. File chứa các thông số cơ bản của game (constants) 6](#_Toc184573858)

[4.2. Điều khiển nhân vật 7](#_Toc184573859)

[4.3. Chướng ngại vật 8](#_Toc184573860)

[4.3.1 Hàng chứa chương ngại vật 8](#_Toc184573861)

[4.3.2 Cấp độ LevelTest 10](#_Toc184573862)

[4.3.2 Cấp độ LevelMain 11](#_Toc184573863)

[4.3.2 Hàm randomRowObstacles 12](#_Toc184573864)

[4.5. Tính điểm 15](#_Toc184573865)

[V Hướng phát triển 15](#_Toc184573866)

[5.1 Cải thiện phần nhìn: 15](#_Toc184573867)

[5.2 Tăng độ khó của game: 16](#_Toc184573868)

[5.3 Âm nhạc và hiệu ứng âm thanh: 16](#_Toc184573869)

[5.4 Cải thiện phần menu và tính năng: 16](#_Toc184573870)

[5.5 Tính năng mở khóa: 16](#_Toc184573871)

[VI. Tài liệu tham khảo 17](#_Toc184573872)

# I. ****Giới thiệu****

## ****1.1. Giới thiệu về sản phẩm và ý tưởng thiết kế****

Game nay được lấy ý tưởng từ tựa game Flappy bird một trờ khá nỗi tiếng vừa đơn giản lại dễ gây nghiện của Flappy bird. Tuy nhiên thay vì bay ngan qua đới với game này là 1 chú sóc bay xuống vá tranh các chướng ngại vật để lấy được hạt thóc ở cuối màn chơi

## ****1.2. Lý do chọn đề tài****

Ngành game là một trông lĩnh vực khá phố biến hiện này vã cũng hay được nghe về nhiều thách thức trong ngành phát triển game cũng như các loại thuật toán và sữ dụng toán học tính toán toán học khá nhiều mà trước này chưa từng thử sức vì vậy dữ án lần này mình muốn thử sức với 1 thứ mới lạ vừa tầm và có tính thư thách nhất định

## 1.3. Thiết kế

Vì chỉ muốn thử thách bản thân trong môi trường mới nên tâm trung vào lập trình và logic game vì vậy phần thiết kế được đơn giản hóa nên có rất nhiều thiếu sót phần lớn những thiết kế điều lấy từ những nguồn khác nhau trên internet

* **Người chới**: Một chú sóc bay ta có thể di chuyển trái phải để tranh chướng ngại
* **Vật cản:** có thể là lá cạy và những cành cay sẽ xuất hiện ngẫu nhiên theo từng hàng
* **Đích đến**: là hạt thốc ở điểm cuối trò chơi
* **Nền đất:** chỉ là những khối hình chữ nhật có đồ họa nền đất đơn giản để làm điểm cuối cho trò chơi

# II. Sử dụng môi trường và công nghệ

## 2.1. Công cụ và môi trường phát triển

* **Android Studio**: Sử dụng làm môi trường giả lập để chạy thử và kiểm tra game trên các thiết bị Android ảo
* **Visual Studio Code**: Là công cụ chính để viết mã nguồn cho ứng dụng, khá được ưa chuộng nhờ sự tối ưu và linh hoạt trong việc lập trình. VS Code được tích hợp nhiều tiện ích hỗ trợ lập trình Flutter

## 2.2. Ngôn ngữ và framework

* **Flutter**: Được chọn làm framework chính nhờ khả năng phát triển giao diện người dùng đa nền tảng (cross-platform) mượt mà. Flutter cho phép lập trình nhanh chóng, dễ dàng tối ưu hóa hiệu suất và giảm thiểu thời gian phát triển
* **Dart: Là ngôn ngữ lập trình chính của Flutter, cung cấp cú pháp gọn nhẹ và dễ học. Dart đặc biệt mạnh mẽ trong việc xử lý logic trò chơi và cập nhật giao diện người dùng một cách đồng bộ**

## 2.3. Thư viện và công cụ hỗ trợ

* **Flame**: Là thư viện game engine dành cho Flutter, cung cấp một bộ giải pháp mạnh mẽ cho việc phát triển trò chơi 2D. Flame đơn giản hóa việc triển khai vòng lặp trò chơi và cung cấp các tính năng cần thiết cho việc phát triện game đơn giản:
  + **Quản lý đầu vào** (input) như cám ứng màn hình, bản phím, chuột cảm biến
  + **Hiển thị hình ảnh và hoạt ảnh** (animation) của nhân vật chướng ngại vật âm thanh....
  + **Xử lý va chạm và các hệ thống thành phần** (Flame Component System - FCS) như khi nhân vật cảm phải 1 đối tượng nào đó và thành phần là như nhân vật, vật cản, đích đến....
* **go\_router**: Thư viện điều hướng được sử dụng trong Flutter, giúp quản lý các luồng điều hướng trong ứng dụng như màng hình menu chính của game khi nhấn nút thì nó sẽ điều hướng đến các trang khác nhau như trang chơi game trang setting thành tưu
* **Provider:** Sử dụng để quản lý và chia sẻ trạng thái điểm số của người chơi trong toàn bộ ứng dụng. Dù Provider có thể xử lý nhiều trạng thái phức tạp, trong dự án này, nó chỉ được sử dụng đơn giản để hiển thị điểm số trên màn hình trò chơi

# III. Chức năng **chính của** trò chơi

## 3.1. Điều khiển nhân vật

* Người chơi điều khiển chú sóc di chuyển sang trái hoặc phải để tránh các chướng ngại vật.

## 3.2. Chướng ngại vật

* **Chuyển động**: Các chướng ngại vật tự động di chuyển từ dưới lên trên, tạo cảm giác nhân vật đang rơi xuống.
* **Chế độ chơi**:
  + **LevelTest**: Các chướng ngại vật được tạo thủ công bằng cách lập trình từng đoạn mã level này để thực hiện test các chương ngại vật và va chậm
  + **LevelMain**: Các chướng ngại vật được tạo tự động trong quá trình chơi, nhưng có thể thêm các đoạn tạo thủ công để tăng độ thử thách và dự định là game sẽ có khoản 1000 dồng nhưng phần level này vẫn còn cần phát triển thêm
* **Tăng độ khó**:
  + Tốc độ chướng ngại vật bắt đầu ở mức **400**.
  + Sau mỗi 5 giây sống sót, tốc độ tăng thêm **100**, tối đa đạt **1000**.

## 3.3. Tính điểm

* Điểm số được tính dựa trên thời gian sống sót.
  + Ban đầu: Người chơi nhận **1 điểm mỗi 2 giây**.
  + Sau mỗi 5 giây, thời gian để nhận điểm giảm dần (từ 2 giây xuống 1.75 giây, 1.5 giây...), thấp nhất là **0.5 giây để nhận 1 điểm**.

## 3.4. Âm thanh

* Mặc dù game dự tính bản đầu game sẽ có âm thanh nền và hiệu ứng âm thanh khi vượt chướng ngại vật, nhưng do không thể tích hợp thư viện flame\_audio, phần âm thanh chưa được thực hiện

Phần **"Triển khai chi tiết"** thường dùng để mô tả cách bạn thực hiện các chức năng chính của game. Bạn có thể chia nhỏ từng chức năng và giải thích cách triển khai bằng mã hoặc mô tả thuật toán. Dưới đây là một gợi ý:

# IV. Triển khai chi tiết

## 4.1. File chứa các thông số cơ bản của game (constants)

//Độ phân giải HD 9x16

//chiều rộng màng hinh

const gameWidth = 1080.0;

//chiều cao màng hinh

const gameHeight = 1920.0;

//kiet thuoc vat can

const obstancleSize = 210.0;

//size cho player

const playerWidth = 200.0;

const playerHeight = 150.0;

double elapsedTime = 0.0; //biến lưu thời gian mà chương trình chạy đã qua

bool isDelayOver = false; //gắn cờ đánh dấu trì hoãn thời gian

double pointTime = 0.0; //biến này để tính điểm

double timeSpeed =

    0.0; //biến này để tăng hoạt động của game để tăng tốc độ chơi

double timeToIncPoint = 2.0;

**Giải thích code**

* + - **gameWidth, gameHeight:** 2 biến này để định hình chiều rộng và chiều dài cho game để game ko bị bớp méo hình ảnh khi chuyển qua các thiết bị có size khác nhau đồng thời biến **gameWidth** cũng giới hạn di chuyển cho người chơi và giúp xác định vị trị đặt các vật cản
    - **obstancleSize:** đặt kích thước chung cho các vật cản
    - **playerWidth, playerHeight:** đặt kích thước cho player
    - **elapsedTime, isDelayOver: elapsedTime** dùng để ghi lại thời gian trôi qua khi chương trình khơi động biến này dùng để xác định **isDelayOver** được chuyển đội khi nào biến **isDelayOver** dùng để gắn cờ khi nào vật cản bắt đầu di chuyên tuy nhiên khi dùng cách này thì game không được mượt mà khi bắt đầu game nên đã có cách khác thay thế cho 2 biến này ở trong **levelMain** vì vậy không cần sữ dụng nữa
    - **pointTime, timeToIncPoint: pointTime** biến này để tính điểm cho người chơi **timeToIncPoint** là thời gian sẽ tính điểm tức là **pointTime = timeToIncPoint** cứ mỗi 2 giây thì **pointTime** sẽ được reset về 0 và cộng 1 điểm **timeToIncPoint** sẽ giảm dần dần để điểm có thể tăng nhanh hơn

## 4.2. Điều khiển nhân vật

Nhân vật (chú sóc) được điều khiển bằng cách nhận đầu vào từ người chơi thông qua cảm ứng màn hình. Khi người chơi vuốt sang

* **Trái**: Nhân vật di chuyển sang trái.
* **Phải**: Nhân vật di chuyển sang phải.

**Code mẫu**

  void move(double deltaX) {

    double newX = position.x + deltaX;

    double minX = -(gameRef.size.x / 2) + size.x / 2;

    double maxX = (gameRef.size.x / 2) - size.x / 2;

    newX = newX.clamp(minX, maxX);

    position.x = newX;

  }

## 4.3. Chướng ngại vật

  final obstacleSpacing = obstancleSize + (playerHeight \* 2);

  final leftSide = -(gameWidth / 2) + (obstancleSize / 2);

  final rightSide = (gameWidth / 2) - (obstancleSize / 2);

**obstacleSpacing**: tạo khoản cách giữa các hàng

**leftSide** và **rightSide:** được tính dựa vào gameWidth để đặt các vật cản bên trái hoặc bên phải màn hình

### 4.3.1 Hàng chứa chương ngại vật

Khi tạo chướng ngại vật sẽ chia chương ngại vật thành 5 chướng ngại vật trong 1 hàng tương ứng với Item1 Item2.. Item5

Dưới đây là code mẫu cho hàm tạo chương ngại vật

  List<ObstacleData> obstacleRow({

    required int row,

    ObstacleType? item1,

    ObstacleType? item2,

    ObstacleType? item3,

    ObstacleType? item4,

    ObstacleType? item5,

  }) {

    List<ObstacleData> content = [];

    final yPosition = obstacleSpacing \* row;

    if (item1 != null) {content.add(ObstacleData(position: Vector2(leftSide, yPosition),

type: item1,),);}

    if (item2 != null) {content.add(ObstacleData(

          position: Vector2(leftSide + (gameWidth / 5), yPosition),

          type: item2,),);}

…

}

**Biến yPosition** để tính vị trị mà vật cản này sẽ được đặt ở đâu trong đoạn code này thì **yPosition = obstacleSpacing \* row** tức là vị trí của vật cản chắc chắn sẽ nằm ở sau khoản cách của vật cản trước và khoản cách của các vật cản

Ví dụ thực tế:

**Item1** được đặt ở vị trí **leftSide** tức **Item1** sẽ nằm sắt ở bên trái màng hình

****

**Item2** thì sẽ nằm ở vị trí bên trái và chưa ra 1 khoản vừa đủ cho **Item1 leftSide** **+ (gameWidth / 5)**

****

**Khi cả 2 item xuất hiện cùng lút**

****

**Ví dụ về yPosition** khi vật thể đầu tiên xuất hiện thì sẽ cách 1 khoản để vật thể tiếp theo có thể xuất hiện

****

### 4.3.2 ****Cấp độ LevelTest****

Trong hàm **LevelTest**, mục tiêu là kiểm tra các vật cản được đặt ở các vị trí cố định trên màn hình, giúp việc thử nghiệm dễ dàng hơn. Ở chế độ này chỉ định các vật cản bằng cách thêm từng item vào mỗi hàng

**Code minh họa**

  //level test chỉ có vài cái để test

  List<ObstacleData> levelTest() {

    List<ObstacleData> level = [];

    //tạo danh sách chưa vật cản từng hàm là tượng chứng một đoạn vật cản

    //đã chia size vừa đủ để chứa 5 item cho 1 hàng vì vậy tối đa là 5 vật cản 1 hàng

    level.addAll(obstacleRow(

      row: 0,

      item1: ObstacleType.leaf,

      item2: ObstacleType.mapleleaf,

      item4: ObstacleType.branch,

      item5: ObstacleType.leaf,

    ));

    level.addAll(obstacleRow(

      row: 1,

      item1: ObstacleType.leaf,

      item2: ObstacleType.leaf,

      item3: ObstacleType.mapleleaf,

      item5: ObstacleType.branch,

    ));

……

}

### 4.3.2 ****Cấp độ LevelMain****

Trong hàm **LevelMain**, mục tiêu là tạo ra một cấp độ với các vật cản được sinh ngẫu nhiên để làm trò chơi thêm phần thử thách và không dễ đoán. Ở chế độ này, các vật cản sẽ được tạo ra ngẫu nhiên cho mỗi hàng **Code minh họa levelMain**

  List<ObstacleData> levelMain() {

    List<ObstacleData> level = [];

    int maxrow = 20;

    // tạo ngẩu nhiên theo dồng

    for (int row = 2; row <= maxrow; row++) {

      final generatedItems = randomRowObstacles();

      level.addAll(obstacleRow(

        row: row,

        item1: generatedItems[0],

        item2: generatedItems[1],

        item3: generatedItems[2],

        item4: generatedItems[3],

        item5: generatedItems[4],

      ));

    }

}

* **LevelMain** tạo ra các vật cản ngẫu nhiên cho mỗi hàng trong một cấp độ. Ở đây, các vật cản trong mỗi hàng được sinh ra từ hàm **randomRowObstacles**, cho phép thay đổi mỗi lần gọi
* **Maxrow** xác định số hàng tối đa mà hàm này tạo
* Vị trí của các vật cản trong mỗi hàng được tính toán giống như trong **LevelTest**, nhưng thay vì chỉ định cố định các vật cản, hệ thống sẽ tạo ra chúng một cách ngẫu nhiên

### 4.3.2 Hàm randomRowObstacles

Hàm này sẽ tạo ra một danh sách vật cản ngẫu nhiên để điền vào các hàng trong **LevelMain.**

**Code minh họa randomRowObstacles**

  List<ObstacleType?> randomRowObstacles() {

    final availableObjects = [

      ObstacleType.leaf,

      ObstacleType.branch,

      ObstacleType.mapleleaf,

    ];

    // Tạo danh sách vật cản với 5 phần tử, khởi tạo tất cả là null

    List<ObstacleType?> result = [null, null, null, null, null];

    // Điền các vật cản khác vào các vị trí còn lại

    for (int i = 0; i < result.length; i++) {

      if (result[i] == null) {

        // danh sách vật cản không bao gồm branch nếu là một trong các vị trí không mong muốn

        List<ObstacleType> filteredObjects = List.from(availableObjects);

        //hàm này muốn là những cành cay giống như là mộc ra từ những cái cây 2

        //không cho branch xuất hiện ở vị trí 1 2 3

        if (i == 1 || i == 2 || i == 3) {

          filteredObjects

              .remove(ObstacleType.branch); //xóa branch khỏi danh sách

          // Đặt branch ở item1 hoặc item5 và các item tiếp theo

          if (result[0] == ObstacleType.branch) {

            result[1] = ObstacleType.branch;

            if (result[1] == ObstacleType.branch) {

              result[2] = ObstacleType.branch;

            }

          } else if (result[4] == ObstacleType.branch) {

            result[3] = ObstacleType.branch;

            if (result[3] == ObstacleType.branch) {

              result[2] = ObstacleType.branch;

            }

          }

        }

        // Chọn ngẫu nhiên một vật cản khác

        int randomIndex = random.nextInt(filteredObjects.length);

        result[i] = filteredObjects[randomIndex];

      }

    }

    //chắc chắn người chơi phải có ít nhất 1 ô trống để có thể vượt qua

    if (!result.contains(null)) {

      bool hasReplaced = false;

      while (!hasReplaced) {

        int replaceIndex = random.nextInt(result.length);

        //không thay thế vật cản không phải branch

        if (result[replaceIndex] != ObstacleType.branch) {

          result[replaceIndex] = null;

          hasReplaced = true; // Đã thay thế thành công

        }

      }

    }

    return result;

  }

**hàm**

    final availableObjects = [

      ObstacleType.leaf,

      ObstacleType.branch,

      ObstacleType.mapleleaf,

    ];

danh sách vật cản có thể xuất hiện trong mỗi hàng. Gồm leaf, branch, và mapleleaf

    List<ObstacleType?> result = [null, null, null, null, null];

Danh sách result sẽ chứa 5 phần tử, mỗi phần tử ban đầu được gán là null. Sau đó, các vị trí sẽ được điền các vật cản một cách ngẫu nhiên

Giải thích tốm gọn ở những đoạn code còn lại múc địch là cho các nhánh cay xuất hiện ở những điểm 1 2 3 hoặc 5 4 3 tức là như những cành cay mọc dài ra như 1 cành cây mọc ra từ 1 cái cây là muốn có cành cay xuất hiện ở 2 thì phải có cành xuất hiện ở 1 chứ không phải ngẫu nhiên xuất hiên trên không chung

Hàm if ở cuối

    if (!result.contains(null)) {

      bool hasReplaced = false;

      while (!hasReplaced) {

        int replaceIndex = random.nextInt(result.length);

        //không thay thế vật cản không phải branch

        if (result[replaceIndex] != ObstacleType.branch) {

          result[replaceIndex] = null;

          hasReplaced = true; // Đã thay thế thành công

        }

      }

    }

Hàm này sẽ kiếm tra xem hàng vật cản tức là cả hàng đó item 1 2 3 4 5 phải có 1 item trống để người chơi có thể vượt qua

#### 4.4. Tăng độ khó

* Tốc độ di chuyển của chướng ngại vật được điều chỉnh bằng một bộ đếm thời gian.
* Mỗi 5 giây, tốc độ tăng thêm 100 đơn vị cho đến khi đạt tối đa 1000.

**Mã minh họa**:

            timeSpeed += dt;

if (gameRef.obstacleSpeed != 1000) {

          //cữ mỗi 5 giay thi tốc độ vật cản chạy lên tăng thêm 100

          if (timeSpeed >= 5) {

            gameRef.obstacleSpeed += 100;

            //chuyen timeSpeecd ve 0 để tiếp t ực thực hiện dồng if này

            timeSpeed = 0;

          }

        }

## 4.5. Tính điểm

* Điểm số tăng dựa trên thời gian sống sót.
* Khoảng thời gian nhận điểm giảm dần mỗi 5 giây để tăng độ thử thách.

**Mã minh họa**:

        if (gameRef.obstacleSpeed != 1000) {

          //cữ mỗi 5 giay thi tốc độ vật cản chạy lên tăng thêm 100

          if (timeSpeed >= 5) {

            //cữ mỗi 5 giay và thời gian cộng điểm lớn hơn 0.5s thì sẽ giảm 0.2 (cở bản là 2s)

            //cái này dùng để tăng tốc nhận thêm điểm khi tốc độ game tăng lên

            if (timeToIncPoint > 0.5) {

              timeToIncPoint -= 0.25;

            }

            //chuyen timeSpeecd ve 0 để tiếp t ực thực hiện dồng if này

            timeSpeed = 0;

          }

        }

      }

#### 4.6. Âm thanh

* Dự kiến sử dụng thư viện flame\_audio để thêm nhạc nền và hiệu ứng âm thanh khi qua màng. Tuy nhiên, hiện tại không thể tích hợp thư viên nên chức năng âm thanh chưa hoạt động.

# ****V Hướng phát triển****

Mặc dù game đã được xây dựng cơ bản và đáp ứng được các chức năng chính, tuy nhiên vẫn có nhiều điểm có thể cải thiện để mang đến trải nghiệm chơi game tốt hơn cho người dùng

## 5.1 Cải thiện phần nhìn:

* + **Animation nhân vật**: Hiện tại, nhân vật trong game di chuyển khá đơn giản. Việc thêm các hiệu ứng chuyển động mượt mà hơn, như animation khi nhân vật di chuyển, có thể giúp tăng tính sinh động cho game hoặc là animation đống mở màng bay lượng của sóc tăng tốc độ game khi lướt xuống và trơ lại tốc độ bình thương khi thả tay ra hoặc lướt lên
  + **Cái thiện model**: Có thể tự thiết kế model riêng cho game
  + **Hiệu ứng vật cản**: Tạo thêm các hiệu ứng cho vật cản, như các vật cản sẽ có hoạt động thay đổi trạng thái khi bị người chơi va phải

## 5.2 Tăng độ khó của game:

* + **Main Level**: Để tăng tính thích nghi với thư thách, các vật cản xuất hiện ngẫu nhiên sẽ dần dẫn trợ nên nhiều hơn theo thời gian thay vì quá nhiều như hiện tại và sẽ tăng tần suất ngày càng dày đặc. Mức độ khó có thể được điều chỉnh dần dần trong các cấp độ tiếp theo, với các vật cản có kích thước và độ phức tạp cao hơn
  + **Vòng gacha**: Sẽ có nhưng vòng ẩn để người chơi có chọn đi qua nó hay không khi đi qua nó sẽ có cơ hội thêm 1 mạng hoặc buff giảm tốc độ vật cản đi xuyên vật cản hoặc 1 số hiệu ứng gây khó chiệu

## 5.3 Âm nhạc và hiệu ứng âm thanh:

* + **Nhạc nền**: Thêm nhạc nền phù hợp với từng cảnh vật trong game để tạo không khí, giúp người chơi cảm thấy hứng thú và tập trung hơn.
  + **Âm thanh khi vượt qua vật cản**: Cung cấp các hiệu ứng âm thanh như tiếng động khi nhân vật vượt qua vật cản hoặc khi chạm vào vật cản, giúp tăng sự sống động cho trò chơi.

## 5.4 Cải thiện phần menu và tính năng:

* + **Menu tùy chỉnh**: Cho phép người chơi điều chỉnh các yếu tố trong game như tốc độ, số lượng vật cản, số lượng row. Người chơi có thể điều chỉnh độ khó và lựa chọn mức độ thách thức riêng biệt cho mình.
  + **Chế độ chơi và bảng xếp hạng**: Cung cấp thêm chế độ chơi mới, bảng xếp hạng để người chơi có thể so tài với nhau, tạo sự cạnh tranh và động lực để người chơi cố gắng hoàn thành game tốt hơn.

## 5.5 Tính năng mở khóa:

* + **Tùy chỉnh nhân vật**: Cung cấp các tính năng mở khóa mới cho người chơi như thay đổi skin cho nhân vật, chọn lựa vật phẩm hỗ trợ để game thêm phần thú vị và giúp người chơi cảm thấy tự do hơn khi chơi

# VI. Tài liệu tham khảo

1. **Flame Engine Documentation**: Flame Engine. (2024). Flame Engine Documentation. Truy cập từ <https://docs.flame-engine.org/latest/flame/flame.html>
2. **YouTube Tutorial**: YouTube. (2024). Flame Engine Tutorial Series. Truy cập từ <https://www.youtube.com/watch?v=nKDPT47unDo&list=PL_D-RntzgLvYIxI_Kuwy1f7HedxTF2GPK>
3. **Flaticon**: Flaticon. (2024). Icons from Flaticon. Truy cập từ <https://www.flaticon.com/>
4. **Flutter Documentation**: Flutter Dev. (2024). Flutter Documentation. Truy cập từ <https://docs.flutter.dev/>