BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC NHA TRANG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO THỰC TẬP CƠ SỞ**

**CÀI ĐẶT TRÒ CHƠI FLAPPY BIRD**

**GV hướng dẫn: GV. Phạm Thị Kim Ngoan**

**Họ và Tên sinh viên: Ngô Trọng Tín**

**Mã số sinh viên: 63131452**

**Lớp: 63-CNTT2**

**Nha Trang – 2023**

Mục lục

[CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI 3](#_Toc155704291)

[1.1. LÝ DO CHỌN ĐỀ TÀI 3](#_Toc155704292)

[1.2. MỤC TIÊU ĐỀ TÀI 3](#_Toc155704293)

[1.3. YÊU CẦU ĐỀ TÀI 3](#_Toc155704294)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT 5](#_Toc155704295)

[2.1. NGÔN NGỮ LẬP TRÌNH RUBY 5](#_Toc155704297)

[2.2. THƯ VIỆN GOSU 5](#_Toc155704298)

[2.3. CHỨC NĂNG 5](#_Toc155704299)

[CHƯƠNG 3: KẾT QUẢ CÀI ĐẶT 8](#_Toc155704300)

[3.1. CÁC GIAI ĐOẠN TRONG TRÒ CHƠI 8](#_Toc155704301)

[3.2. CẤU TRÚC DỮ LIỆU 8](#_Toc155704302)

[TỔNG KẾT 11](#_Toc155704303)

Phần mở đầu

Trong thời đại công nghệ ngày nay, việc thực hiện các đề tài thực tế là một phần quan trọng của quá trình học tập và phát triển kỹ năng. Trong khuôn khổ thực tập của mình, tôi đã chọn dự án thú vị và ý nghĩa - xây dựng một phiên bản của trò chơi Flappy Bird. Flappy Bird không chỉ là một trò chơi giải trí đơn giản mà còn là một thách thức lớn đối với người phát triển, yêu cầu sự sáng tạo và kỹ năng lập trình để tạo ra một trải nghiệm chơi game thú vị và gây nghiện.

Dự án này không chỉ giới thiệu tôi vào thế giới của phát triển game, mà còn đưa ra cơ hội để áp dụng các kiến thức học được từ trường đại học vào thực tế. Bằng cách này, tôi có cơ hội nắm bắt và thử nghiệm nhiều khía cạnh khác nhau của quá trình phát triển phần mềm, từ việc lên kế hoạch và thiết kế đến lập trình và kiểm thử.

Mục tiêu của dự án này không chỉ đơn thuần là tạo ra một phiên bản mới của Flappy Bird, mà còn là hiểu rõ sâu sắc về cách các yếu tố khác nhau, như vật lý chuyển động, đồ họa, âm thanh và giao diện người dùng, tác động đến trải nghiệm chơi game. Tôi hy vọng rằng dự án này sẽ mang lại những kỹ năng và hiểu biết mới, đồng thời mở ra những cơ hội để phát triển sự sáng tạo của mình trong lĩnh vực phát triển phần mềm và game.

Chương 1: tổng quan về đề tài

## LÝ DO CHỌN ĐỀ TÀI

Việc lựa chọn đề tài thực tập là một quá trình quan trọng, đòi hỏi sự cân nhắc kỹ lưỡng để đảm bảo rằng nó không chỉ mang lại kiến thức chuyên sâu mà còn đáp ứng đúng mong muốn và mục tiêu cá nhân. Dưới đây là những lý do mà tôi đã chọn dự án Flappy Bird làm đề tài thực tập của mình:

Thách Thức Kỹ Thuật: Flappy Bird, mặc dù có vẻ đơn giản, thực sự đặt ra những thách thức kỹ thuật đáng kể. Từ việc xử lý vật lý chuyển động cho đến tối ưu hóa hiệu suất và tương tác người dùng, dự án này cung cấp một cơ hội lý tưởng để phát triển và kiểm tra kỹ năng lập trình và thiết kế.

Áp Dụng Kiến Thức Học Được: Việc áp dụng những kiến thức đã học trong quá trình đào tạo là quan trọng. Dự án Flappy Bird cho phép tôi áp dụng các khái niệm về lập trình game, xử lý sự kiện, và quản lý tài nguyên trong một môi trường thực tế.

## MỤC TIÊU ĐỀ TÀI

Áp dụng những kiến thức đã được học như câu lệnh if else, switch case, vòng lặp, hướng đối tượng, cùng với việc tự tìm hiểu thêm các kiến thức về đồ hoạ, thư viện cũng như cách cài đặt trò chơi từ đó tạo nên một trò chơi hấp dẫn.

## YÊU CẦU ĐỀ TÀI

Đồ họa và Âm thanh Hấp dẫn: Tạo ra giao diện đồ họa thân thiện với người chơi, với đồ họa đơn giản nhưng sáng tạo, phản ánh chủ đề của Flappy Bird. Âm thanh sẽ được chọn kỹ lưỡng để tạo ra trải nghiệm âm thanh hấp dẫn và tương thích với sự kiện trong trò chơi.

Điều khiển Mượt mà: Phát triển hệ thống điều khiển linh hoạt và mượt mà cho người chơi. Sử dụng các sự kiện của bàn phím hoặc chuột để điều khiển nhân vật chính (vịt) một cách dễ dàng và tự nhiên.

Khó khăn Tăng dần: Tăng độ khó dần theo thời gian hoặc theo cách nhất định để thách thức người chơi. Điều này có thể thể hiện qua việc tăng tốc độ di chuyển của các chướng ngại vật hoặc làm cho khoảng trống giữa chúng trở nên hẹp hơn.

Hiển thị Điểm số và Điểm cao: Tích hợp tính năng hiển thị điểm số của người chơi và theo dõi điểm cao nhất. Điều này tạo động lực cho người chơi cố gắng đạt được điểm cao mới.

Xử lý Các Sự kiện Đặc biệt: Xử lý các sự kiện đặc biệt như va chạm với chướng ngại vật, điểm số mới, và thậm chí là kết thúc trò chơi. Cung cấp thông báo rõ ràng và thú vị khi xảy ra các sự kiện này.

Mã nguồn Rõ ràng và Dễ hiểu: Viết mã nguồn theo quy tắc clean code, giúp dễ dàng bảo trì và mở rộng trong tương lai. Chú trọng vào cách tổ chức mã nguồn để nó trở nên dễ đọc và hiểu, đồng thời áp dụng các nguyên tắc lập trình hướng đối tượng.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT



## NGÔN NGỮ LẬP TRÌNH RUBY

Ruby, một trong những ngôn ngữ lập trình linh hoạt và mạnh mẽ, đã thu hút sự chú ý của cộng đồng phát triển từ khi được giới thiệu. Được tạo ra bởi Yukihiro "Matz" Matsumoto, Ruby nổi tiếng với cú pháp đơn giản, dễ đọc và linh hoạt, tạo điều kiện thuận lợi cho sự sáng tạo và phát triển nhanh chóng. [1]

Ruby là một ngôn ngữ lập trình hướng đối tượng đầy đủ, hỗ trợ nhiều kiểu lập trình khác nhau. Nó chủ yếu được sử dụng cho việc phát triển ứng dụng web thông qua framework như Ruby on Rails, nhưng cũng rất thích hợp cho phát triển trò chơi và ứng dụng đa dạng khác nhau. [1]

## THƯ VIỆN GOSU

Trong lĩnh vực phát triển game, thư viện Gosu nổi bật như một công cụ mạnh mẽ cho việc tạo ra trải nghiệm game đơn giản và gọn nhẹ bằng Ruby. Gosu cung cấp một bộ công cụ đồ họa và âm thanh hiệu suất cao, giúp nhà phát triển dễ dàng thực hiện các tính năng như xử lý sự kiện, quản lý vật lý, và tương tác với người chơi.

Với cộng đồng Ruby ngày càng phát triển, việc sử dụng Ruby và Gosu để phát triển trò chơi không chỉ là một cách tiếp cận thú vị mà còn là một lựa chọn linh hoạt cho những người muốn khám phá và sáng tạo trong thế giới đa dạng của lập trình game và phát triển ứng dụng.

## CHỨC NĂNG

Chức năng chính của thư viện Gosu bao gồm quản lý cửa sổ đồ họa, xử lý sự kiện đầu vào từ bàn phím và chuột, cũng như tạo và quản lý hình ảnh và âm thanh. Gosu hỗ trợ việc vẽ đồ họa 2D với fps mặc định là 60 giúp tối ưu hiệu suất, tối ưu hóa trải nghiệm người dùng trong các ứng dụng đồ họa động và trò chơi.

Mở một cửa sổ window:

require 'gosu'

class MyWindow < Gosu::Window

def initialize

super(640, 480, false)

self.caption = 'Hello World!'

end

end

window = MyWindow.new

window.show

Ngoài ra, Gosu cũng cung cấp các tính năng khác như quản lý thời gian, hiệu ứng chuyển động, và tích hợp thuận lợi với các thư viện khác trong cộng đồng Ruby. Điều này giúp Gosu trở thành một lựa chọn phổ biến cho những người phát triển muốn tạo ra các ứng dụng đa phương tiện đẹp mắt và chất lượng cao bằng ngôn ngữ Ruby.

Quản lý thời gian trong Gosu:

class GameWindow < Gosu::Window

def initialize

super(640, 480, false)

self.caption = "Quản lý thời gian trong Gosu"

@start\_time = Gosu.milliseconds

end

def update

elapsed\_time = Gosu.milliseconds - @start\_time

seconds = (elapsed\_time / 1000).to\_i

# Thực hiện các công việc cập nhật dựa trên thời gian, ví dụ:

puts "Đã trôi qua #{seconds} giây."

end

def draw

# Thực hiện các công việc vẽ tại đây

end

end

chương 3: kết quả cài đặt

## CÁC GIAI ĐOẠN TRONG TRÒ CHƠI

Game sẽ có 3 giai đoạn:

Giai đoạn 1: Trước khi chơi, người chơi sẽ được đưa tới một menu với các mục lựa chọn như chọn độ khó, thoát game, xem điểm xếp hạng, chơi mới,…

Giai đoạn 2: Bắt đầu trò chơi, người chơi sẽ nhấn nút space để điều khiển chim

+ Yêu cầu: chim không được bay quá cao (chạm đỉnh) cũng như không được bay quá thấp (chạm đáy). Sau một khoảng thời gian bắt đầu chơi thì sẽ có các cặp ống di chuyển về phía chú chim, nhiệm vụ của người chơi chỉ đơn giản là né các cặp ống này, né được mỗi cặp sẽ thêm được một điểm.

Giai đoạn 3: Khi người chơi không thực hiện được các yêu cầu ở giai đoạn 2 (tức thua), trò chơi sẽ được tạm ngừng và người chơi sẽ được lựa chọn là thoát game (quay trở lại giai đoạn 1) hay tiếp tục chơi.

Độ khó của trò chơi được chia làm hai mức: khó và dễ, nếu người chơi chọn mức khó thì các ống sẽ di chuyển nhanh hơn và khoảng cách giữa mỗi cặp ống sẽ được thu hẹp.

## Cấu trúc dữ liệu

Vì Ruby là một ngôn ngữ lập trình hướng đối tượng nên dự án game Flappy Bird cũng được xây dựng dựa trên việc hươngs đối tượng, ví dụ về các đối tượng như các cửa sổ, cửa sổ menu,bảng điểm, gameplay,…. Các vật như chim hay ống cũng được viết theo phong cách hướng đối tượng, chỉ một vài chức năng nhỏ như kiểm tra chạm ống mới được viết theo hướng chức năng.

Lớp vẽ ống:

class Pipe

attr\_accessor :x\_position, :y\_position, :image, :in

def initialize() # khởi tạo một ống

def self.initialize\_pipe(num\_of\_pipe) # khởi tạo nhiều ống

end

//

Lớp vẽ chim:

class Bird

attr\_accessor :x\_position, :y\_position, :image, :above\_corner, :below\_corner

def initialize(x = -1, y = -1) # khởi tạo chim

def update\_info # cập nhật vị trí của chim cho mỗi khung hình

end

//

Lớp vẽ menu ban đầu:

class SelectMenu < Gosu::Window

end

//

Lớp vẽ bảng điểm:

class ScoreBoard < Gosu::Window

private

def score\_taken # hàm lấy dữ liệu từ file json

end

//

Lớp gameplay sử dụng các lớp khác:

class Game < Gosu::Window

attr\_accessor :pipes, :game\_over

def initialize(dificulty = "Dễ") # khởi tạo các thông số đầu game

def draw # vẽ cửa sổ của một khung hình

def update # cập nhật các thông số của game

def button\_down(ascii\_key) # đón sự kiện từ người dùng

private

def score\_font\_draw # vẽ điểm

def init\_pipe\_pos(pipe) # khởi tạo vị trí cặp ống

def pipe\_draw # vẽ ống

def pipe\_move(pipe) # di chuyển một cặp ống

def pipe\_init\_move # di chuyển các cặp ống

def not\_init?(pipe) # kiểm tra cặp ống đã được khởi tạo hay chưa

def init\_all? # kiểm tra các cặp ống đã được khởi tạo hay chưa

end

//

Các hàm riêng biệt viết theo hướng cấu trúc:

def create\_background(option) # vẽ nền tuỳ theo màn

def set\_of\_in\_background\_pipes(pipes, background) # hàm trả về các ống bên trong nền

def did\_bird\_touch?(bird, set\_pipes, background) # kiểm tra xem chim đã chạm ống hay chưa

def num\_of\_score(pipes, bird) # số điểm người chơi đạt được

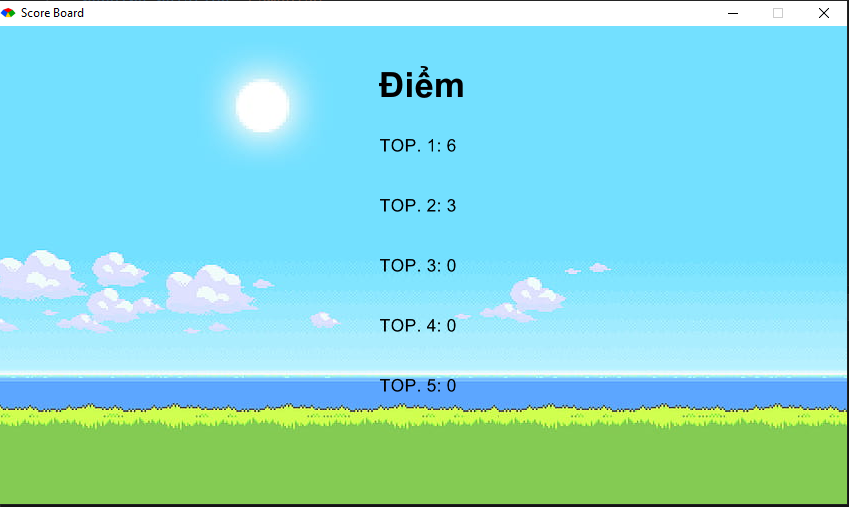
def save\_score(score) # lưu điểm số vào file json

Việc lưu điểm số của người chơi vào file **json** giúp nhà phát triển có thể dễ dàng mở rộng thêm dữ liệu thông qua cấu trúc **Hash.**

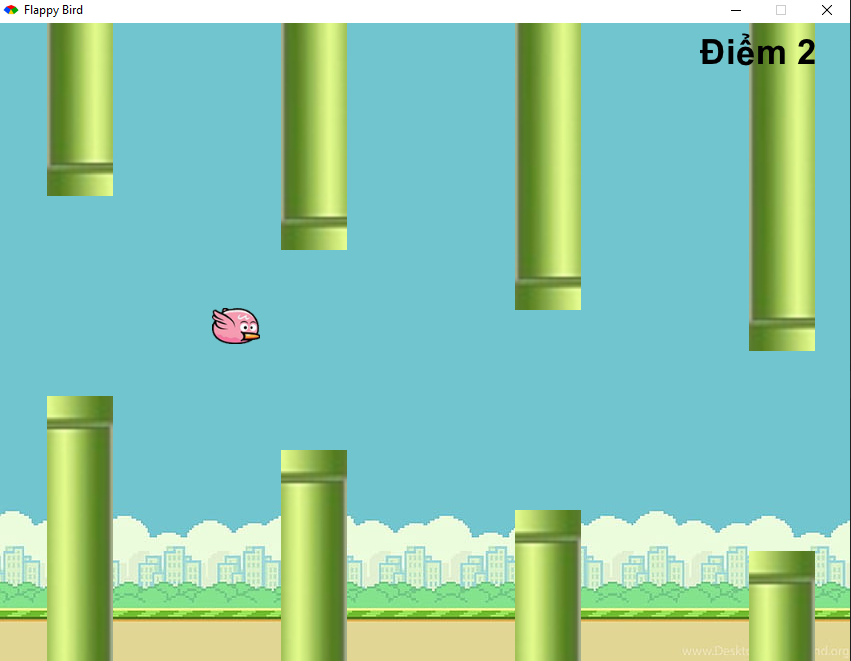
Và kết quả là một game khá giống với phiên phản Flappy Bird tiền nhiệm được hoàn thành.



Hình 3.2 1: Màn hình menu



Hình 3.2 2 Bảng điểm



Hình 3.2 3 Màn hình chơi game

tổng kết

Trong suốt thời gian thực tập, tôi đã có cơ hội học hỏi và trải nghiệm nhiều về việc phát triển trò chơi thông qua việc viết game Flappy Bird bằng ngôn ngữ Gosu Ruby. Việc này đã giúp tôi làm quen với nền tảng lập trình game và củng cố kỹ năng lập trình của mình.

Trong quá trình thực tập, tôi đã tiếp cận và làm việc với thư viện Gosu, một công cụ mạnh mẽ hỗ trợ phát triển game trên ngôn ngữ lập trình Ruby. Tôi đã hiểu rõ về cách Gosu giúp tạo ra các đối tượng trong game, quản lý sự kiện và vẽ đồ họa. Điều này giúp tôi xây dựng một phiên bản Flappy Bird đơn giản, từ việc tạo ra nhân vật chính cho đến việc xử lý va chạm và điểm số.

Việc tìm tòi và phát triển những thứ tôi chưa từng thử trước đây giúp tôi hiểu hơn về ngôn ngữ lập trình và sức mạnh mà nó mang lại. Tôi mong đây là một dự án cũng như cơ hội để mình trau dồi kiến thức, làm đa dạng những kỹ năng của mình, không chỉ bó hẹp trong không gian web mà còn trên các nền tảng khác nhau.

tài liệu tham khảo

# Tài liệu liên quan tới ngôn ngữ ruby và thư viện Gosu

|  |  |
| --- | --- |
|  | [*https://www.rubydoc.info/gems/gosu/*](https://www.rubydoc.info/gems/gosu/)  [*https://www.ruby-lang.org/vi/*](https://www.ruby-lang.org/vi/) |

[1] *https://tuhoclaptrinh.edu.vn/ung-dung-cua-ngon-ngu-lap-trinh-ruby-1011.html*