**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**PHÂN HIỆU TẠI TP. HỒ CHÍ MINH**

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO ĐỒ ÁN TỐT NGHIỆP**

**ĐỀ TÀI:**

Giảng viên hướng dẫn: TRẦN PHONG NHÃ

Sinh viên thực hiện: NGUYỄN VĂN LONG

Lớp : CQ.59.CNTT

Khoá : 59

Tp. Hồ Chí Minh, năm 2024

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**PHÂN HIỆU TẠI TP. HỒ CHÍ MINH**

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO ĐỒ ÁN TỐT NGHIỆP**

**ĐỀ TÀI:**

Giảng viên hướng dẫn: TRẦN PHONG NHÃ

Sinh viên thực hiện: NGUYỄN VĂN LONG

Lớp : CQ.59.CNTT

Khoá : 59

Tp. Hồ Chí Minh, năm 2024

TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**PHÂN HIỆU TẠI THÀNH PHỐ HỒ CHÍ MINH** Độc lập – Tự do – Hạnh phúc

# NHIỆM VỤ THIẾT KẾ TỐT NGHIỆP

BỘ MÔN: **CÔNG NGHỆ THÔNG TIN**

-------\*\*\*-------

**Mã sinh viên:**5951071051 **Họ tên SV:** Nguyễn Văn Long

**Khóa:** 59 **Lớp:** CQ.CNTT

1. **Tên đề tài**

Xây dựng trò chơi 2D trên Unity:

1. **Mục đích, yêu cầu**

- Mục đích:

+ Phát triển một trò chơi 2D trên nề tảng Unity nhằm mang lại trải nghiệm thú vị cho người chơi.

+ Hiểu được quá trình phát triển trò chơi trên nền tảng Unity.

- Yêu cầu:

+ Người chơi có thể điều khiển nhân vật trò chơi theo các hướng (trái, phải, lên, xuống). Nhân vật có thể tấn công kẻ thù và tương tác với các vật phẩm trong game.

+ Hệ thống chiến đấu: Nhân vật có thể dùng vũ khí tấn công kẻ thù và kẻ thù cũng có thể phản công gây sát thương cho nhân vật.

+ Thu thập vật phẩm: Khi tiêu diệt kẻ thù có thể rơi ra một số vật phẩm như tiền xu, bình máu,... nhân vật có thể thu thập lại.

+ Hệ thống kĩ năng: Trò chơi cung cấp một số kĩ năng để tấn công kẻ thù.

+ Âm thanh và nhạc nền: Tích hợp âm thanh và nhạc nền phù hợp với bối cảnh trò chơi nhằm tăng trải nghiệm cho người chơi.

+ Giao diện người dùng: Thiết kế giao diện thân thiện, dễ sử dụng.

+ Lưu trữ và tải lại tiến độ: Trò chơi có chức năng lưu lại dữ liệu người chơi và cho phép người chơi tải lại dữ liệu.

+ Mã hóa dữ liệu: Dữ liệu người chơi lưu về máy sẽ được mã hóa tránh việc bị truy cập và sửa chữa dữ liệu.

1. **Nội dung và phạm vi đề tài**

**-** Nội dung:

+ Nghiên cứu về phát triển trò chơi 2D, các công cụ và thư viện liên quan. Tìm hiểu các yếu tố thiết kế trò chơi.

+ Lên ý tưởng và thiết kế nhân vật, kẻ thù, giao diện người dùng và các tương tác.

+ Thiết lập môi trường phát triển và công cụ cần thiết. Lập trình các chức năng chính như điều khiển nhân vật, hệ thống chiến đấu,.... Tích hợp âm thanh và nhạc nền.

- Phạm vi:Tập trung vào phát triển một trò chơi 2D trên máy tính không bao gồm tính năng phức tạp như chơi trực tuyến hay tích hợp mạng xã hội.

1. **Công nghệ, công cụ và ngôn ngữ lập trình**

**-** Các công nghệ và công cụ: Unity Engine, Visual Studio Code, Git và GitHub.

- Ngôn ngữ lập trình C# là ngôn ngữ chính để lập trình các chức năng.

1. **Các kết quả chính dự kiến sẽ đạt được và ứng dụng**

**-** Kết quả dự kiến: Xây dựng được một trò chơi 2D đáp ứng được các yêu cầu được đề ra.

- Ứng dụng: Cung cấp một trò chơi nhằm đáp ứng nhu cầu giải trí và là nền tảng để phát triển các trò chơi phức tạp hơn trong tương lai.

1. **Giáo viên và cán bộ hướng dẫn**

Họ tên: Trần Phong Nhã

Đơn vị công tác:Trường ĐH Giao thông vận tải phân hiệu Tp.HCM

Điện thoại:0906761014 Email:tpnha@utc2.edu.vn

|  |  |
| --- | --- |
| **Ngày tháng 03 năm 2024**  **Trưởng BM Công nghệ Thông tin** | **Đã giao nhiệm vụ TKTN**  **Giáo viên hướng dẫn** |
| **ThS. Trần Phong Nhã** |  |

Đã nhận nhiệm vụ TKTN

Sinh viên:Nguyễn Văn Long Ký tên:

Điện thoại:0817157873 Email:5951071051@st.utc2.edu.vn

**LỜI CẢM ƠN**

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

|  |
| --- |
| ***Tp. Hồ Chí Minh, ngày ….… tháng ….… năm ….…***  **Giáo viên hướng dẫn**  **Trần Phong Nhã** |

**MỤC LỤC**

[NHIỆM VỤ THIẾT KẾ TỐT NGHIỆP 3](#_Toc7393)

[DANH MỤC HÌNH ẢNH 8](#_Toc18229)

[CHƯƠNG 1. MỞ ĐẦU 9](#_Toc32383)

[1.1 Tổng quan 9](#_Toc25977)

[1.1.1 Thể loại Top-Down. 9](#_Toc15094)

[1.1.2 Tổng quan về Unity. 11](#_Toc18547)

[1.1.3 Tổng quan về Visual Studio Code. 12](#_Toc11922)

[1.1.4 Ngôn ngữ lập trình C#. 13](#_Toc26403)

[1.2 Mục tiêu nghiên cứu 15](#_Toc30869)

[1.3 Phạm vi 15](#_Toc24319)

[CHƯƠNG 2. CÔNG CỤ UNITY ENGINE 16](#_Toc14065)

[2.1 Giới thiệu về Unity. 16](#_Toc15163)

[2.2 Giao diện Unity Editor. 17](#_Toc16618)

[2.2.1 Cửa sổ Scene. 17](#_Toc4968)

[2.2.2 Cửa sổ Game. 18](#_Toc24514)

[2.2.3 Cửa sổ Hierachy. 18](#_Toc9186)

[2.2.4 Cửa sổ Inspector 20](#_Toc15844)

[2.2.5 Cửa sổ Project. 20](#_Toc22252)

[2.2.6 Cửa sổ Animator. 21](#_Toc18459)

[2.2.7 Cửa sổ Animation. 22](#_Toc11113)

[2.3 Các thành phần trong Unity. 23](#_Toc23006)

[2.3.1 GameObject. 23](#_Toc3727)

[2.3.2 Component. 23](#_Toc5314)

[2.3.3 Assets 24](#_Toc10249)

[2.3.4 Scenes. 24](#_Toc1151)

[2.3.5 Sprite. 25](#_Toc21610)

[2.3.6 Animation. 26](#_Toc31799)

[2.3.7 Prefab. 26](#_Toc10197)

[2.3.8 Script. 26](#_Toc16094)

[2.3.9 Camera. 27](#_Toc25414)

[CHƯƠNG 3. MÔ TẢ THIẾT KẾ TRÒ CHƠI 28](#_Toc21492)

[3.1 Xây dựng kịch bản trò chơi. 28](#_Toc22764)

[3.2 Chuẩn bị tài nguyên cho trò chơi. 28](#_Toc13034)

[3.3 Tạo những đối tượng và thêm thành phần vật lý cho tất cả đối tượng. 29](#_Toc24681)

[3.4 Tạo Animation cho đối tượng. 32](#_Toc755)

[3.5 Thiết kế giao diện. 33](#_Toc16883)

[3.6 Thiết kế âm thanh. 35](#_Toc23050)

[3.7 Xử lý hành vi cho đối tượng. 35](#_Toc30640)

[3.8 Tạo lớp lưu trữ dữ liệu. 35](#_Toc32113)

[CHƯƠNG 4. KẾT QUẢ VÀ KIẾN NGHỊ 36](#_Toc24259)

[4.1 Kết quả đạt được. 36](#_Toc30464)

[4.2 Hạn chế. 36](#_Toc32222)

[4.3 Hướng phát triển. 36](#_Toc11762)

# **DANH MỤC HÌNH ẢNH**

# CHƯƠNG 1. MỞ ĐẦU

## Tổng quan

Đề tài này tập trung vào việc phát triển một game 2D thể loại Top-Down sử dụng công cụ Unity. Game sẽ có bối cảnh phiêu lưu hành động, nơi người chơi điều khiển một nhân vật chính để chiến đấu với kẻ thù và thu thập vật phẩm.

### 1.1.1 Thể loại Top-Down.

- Định nghĩa: Top-down là loại trò chơi mà góc nhìn của người chơi được đặt từ trên cao nhìn xuống, thường với một góc nhìn trực quan trực tiếp xuống môi trường trò chơi. Hình ảnh và các yếu tố trong game được hiển thị theo dạng 2D.



Hình 1.1 Game 2D thể loại Top-Down.

- Lịch sử phát triển:

+Thời kỳ đầu: Các trò chơi top-down đã xuất hiện từ những năm 1980, với những tựa game kinh điển như "Gauntlet" và "Ultima". Những trò chơi này đã đặt nền móng cho thể loại này với cách điều khiển và cơ chế chơi đơn giản.

+Phát triển và đổi mới: Trong những năm 1990 và 2000, thể loại này tiếp tục phát triển với sự ra đời của nhiều tựa game đình đám và sự tiến bộ trong công nghệ đồ họa, giúp nâng cao trải nghiệm người chơi.

+Hiện đại: Ngày nay, thể loại top-down vẫn rất phổ biến và được yêu thích, đặc biệt trong các trò chơi indie. Các công cụ phát triển hiện đại như Unity và Godot cho phép các nhà phát triển dễ dàng tạo ra các trò chơi chất lượng cao với đồ họa và cơ chế phong phú.

- Đặc điểm chính:

+Góc nhìn từ trên xuống: Người chơi nhìn thấy nhân vật và môi trường từ một góc nhìn thẳng đứng, cho phép họ quan sát toàn bộ khu vực xung quanh nhân vật.

+ Điều khiển đơn giản: Thường sử dụng các phím mũi tên hoặc joystick để di chuyển nhân vật theo các hướng (lên, xuống, trái, phải) và các nút khác để thực hiện các hành động như tấn công, nhảy, hoặc sử dụng vật phẩm.

+ Đồ họa 2D: Đồ họa thường được thiết kế dưới dạng 2D sprite, giúp giảm thiểu yêu cầu về phần cứng và dễ dàng hơn trong việc phát triển trò chơi.

- Các thể loại con phổ biến:

+Action-adventure (Hành động-phưu lưu): Ví dụ điển hình là loạt game "The Legend of Zelda", nơi người chơi điều khiển nhân vật chính khám phá các hầm ngục, giải đố và chiến đấu với kẻ thù.

+RPG (Nhập vai): Ví dụ như "Pokémon" hoặc "Final Fantasy" những phiên bản đầu, nơi người chơi điều khiển nhân vật hoặc nhóm nhân vật, tham gia vào các trận chiến theo lượt hoặc thời gian thực, và phát triển nhân vật thông qua hệ thống cấp độ và kỹ năng.

+Shooter (Bắn súng): Ví dụ như "Hotline Miami", nơi người chơi điều khiển nhân vật di chuyển qua các màn chơi và bắn hạ kẻ thù trong thời gian thực.

+Puzzle (Giải đố): Các trò chơi như "Sokoban" hoặc "Baba Is You", nơi người chơi cần giải quyết các câu đố để tiến xa hơn.

- Ưu điểm:

+Dễ tiếp cận: Thể loại này thường có cơ chế điều khiển đơn giản, dễ tiếp cận cho mọi đối tượng người chơi.

+Phong cách nghệ thuật đa dạng: Đồ họa 2D cho phép nhiều phong cách nghệ thuật khác nhau, từ pixel art cổ điển đến các hình ảnh vẽ tay đẹp mắt.

+Yêu cầu phần cứng thấp: Không đòi hỏi phần cứng mạnh mẽ, phù hợp với nhiều nền tảng và thiết bị.

- Hạn chế:

+Giới hạn về không gian: Góc nhìn từ trên xuống có thể hạn chế cảm giác về không gian và chiều sâu của môi trường trò chơi.

+Thiếu chi tiết: Đồ họa 2D có thể thiếu chi tiết và độ phức tạp so với đồ họa 3D.

### 1.1.2 Tổng quan về Unity.

- Unity là một công cụ phát triển game và phần mềm ứng dụng, cho phép các nhà phát triển tạo ra các sản phẩm tương tác 2D, 3D, thực tế ảo (VR), và thực tế tăng cường (AR).



Hình 1.2 Logo của Unity.

- Các lĩnh vực ứng dụng

+Phát triển trò chơi: Unity được sử dụng rộng rãi để phát triển các trò chơi 2D và 3D trên nhiều nền tảng khác nhau.

+Ứng dụng thực tế ảo và tăng cường: Unity hỗ trợ mạnh mẽ cho VR và AR, giúp các nhà phát triển tạo ra các ứng dụng thực tế ảo và thực tế tăng cường chất lượng cao.

+Ứng dụng tương tác: Unity còn được sử dụng để phát triển các ứng dụng tương tác khác như mô phỏng, giáo dục, đào tạo, và các ứng dụng doanh nghiệp.

- Ưu điểm:

+Đa nền tảng: Khả năng phát triển cho nhiều nền tảng khác nhau từ một mã nguồn duy nhất.

+Cộng đồng lớn: Một cộng đồng phát triển mạnh mẽ với nhiều tài liệu, hướng dẫn và hỗ trợ từ cộng đồng.

+Giao diện trực quan: Dễ sử dụng và trực quan, phù hợp cho cả người mới bắt đầu và những nhà phát triển có kinh nghiệm.

+Asset Store: Một kho tài nguyên phong phú giúp tiết kiệm thời gian phát triển.

- Hạn chế:

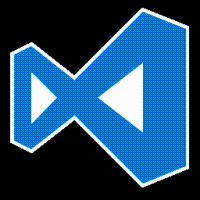
+Hiệu suất: Đôi khi có thể gặp vấn đề về hiệu suất, đặc biệt là khi làm việc với các dự án phức tạp hoặc lớn.

+Chi phí: Một số tính năng nâng cao và giấy phép Unity Pro có chi phí cao.

+Cấu trúc phức tạp: Mặc dù giao diện thân thiện, nhưng việc quản lý các dự án lớn có thể trở nên phức tạp.

### 1.1.3 Tổng quan về Visual Studio Code.

- Visual Studio Code (VS Code) là một trình biên soạn mã nguồn mở, dễ sử dụng và mạnh mẽ, được phát triển bởi Microsoft.



Hình 1.3 Logo Visual Studio Code.

- Lịch sử phát triển: Visual Studio Code được giới thiệu lần đầu vào tháng 4 năm 2015 bởi Microsoft. Từ đó đến nay, VS Code đã trở thành một trong những trình soạn thảo mã nguồn phổ biến nhất trên thế giới, với một cộng đồng người dùng đông đảo và nhiều tính năng mạnh mẽ.

- Đặc điểm chính:

+Hỗ trợ nhiều ngôn ngữ lập trình: VS Code hỗ trợ nhiều ngôn ngữ lập trình phổ biến như JavaScript, TypeScript, Python, C#, Java, HTML/CSS, và nhiều ngôn ngữ khác.

+Mở rộng và tích hợp: VS Code cho phép cài đặt các extension từ cộng đồng hoặc từ Microsoft để mở rộng chức năng của trình soạn thảo.

+Giao diện người dùng tùy biến: VS Code cung cấp một giao diện người dùng linh hoạt và có thể tùy chỉnh để phản ánh sở thích cá nhân của người dùng.

+Hỗ trợ hệ thống kiểm soát phiên bản: VS Code tích hợp sẵn với Git và hỗ trợ nhiều hệ thống kiểm soát phiên bản khác nhau như GitHub, Bitbucket, và Azure DevOps.

+Tích hợp terminal: VS Code có một terminal tích hợp sẵn, cho phép người dùng chạy các lệnh và quản lý dự án mà không cần chuyển sang ứng dụng terminal riêng biệt.

+Hỗ trợ debug: VS Code cung cấp các công cụ hỗ trợ debug phong phú cho nhiều ngôn ngữ lập trình, giúp người dùng tìm và sửa lỗi dễ dàng.

- Ưu điểm:

+ Giao diện người dùng đơn giản và linh hoạt.

+ Hỗ trợ nhiều ngôn ngữ lập trình và tích hợp nhiều tính năng mạnh mẽ.

+ Tích hợp với Git và các hệ thống kiểm soát phiên bản khác.

+ Hỗ trợ debug và terminal tích hợp.

+ Phong phú với các extension và cộng đồng lớn hỗ trợ.

- Hạn chế:

+Hiệu suất có thể không tốt với các dự án lớn và phức tạp.

+Khả năng tùy chỉnh giao diện có thể không linh hoạt như một số trình soạn thảo mã khác.

### 1.1.4 Ngôn ngữ lập trình C#.

- Ngôn ngữ lập trình C# là một ngôn ngữ lập trình hướng đối tượng (OOP), đa nền tảng (cross-platform), phát triển bởi Microsoft. Nó được thiết kế để kết hợp tính linh hoạt của C++ và tiện ích của C# để phát triển các ứng dụng đa dạng trên nền tảng .NET.



Hình 1.4 Logo ngôn ngữ C#.

- Lịch sử phát triển

+ C#: C# được phát triển vào những năm 2000 bởi Microsoft, và phiên bản đầu tiên của nó (C# 1.0) được phát hành cùng với .NET Framework vào năm 2002.

+ .NET Core: .NET Core là một phiên bản mới của .NET, ra mắt vào năm 2016, đưa C# trở thành ngôn ngữ lập trình đa nền tảng.

+ .NET 5: .NET 5 là sự kết hợp của .NET Framework và .NET Core, ra mắt vào năm 2020, đánh dấu sự thống nhất của hệ sinh thái .NET.

- Đặc điểm chính:

+ Hướng đối tượng: C# hỗ trợ lập trình hướng đối tượng với các tính năng như kế thừa, đa hình, đóng gói và trừu tượng hóa.

+ Đa nền tảng: Với .NET Core và .NET 5 (và các phiên bản mới hơn), C# trở thành ngôn ngữ lập trình đa nền tảng, cho phép viết mã một lần và chạy trên nhiều hệ điều hành khác nhau như Windows, Linux và macOS.

+ Cú pháp dễ đọc: Cú pháp của C# rất gần với ngôn ngữ lập trình tự nhiên, dễ hiểu và dễ đọc, giúp giảm bớt thời gian và công sức trong việc phát triển và bảo trì mã nguồn.

+ Kiểm soát bộ nhớ: C# có một hệ thống quản lý bộ nhớ tự động (garbage collection), giúp giảm thiểu rủi ro lỗi như rò rỉ bộ nhớ và thải bộ nhớ không đúng.

+ Tích hợp mạnh mẽ: C# tích hợp tốt với nền tảng .NET, cho phép truy cập vào một loạt các thư viện chuẩn và bên thứ ba để phát triển các ứng dụng đa dạng từ ứng dụng desktop đến ứng dụng web và di động.

+ Hỗ trợ mạnh mẽ từ Visual Studio: Visual Studio là một IDE (Integrated Development Environment) mạnh mẽ từ Microsoft, được thiết kế đặc biệt cho phát triển ứng dụng C#. Nó cung cấp các công cụ hỗ trợ phong phú như IntelliSense, Debugging, và các công cụ kiểm tra mã nguồn.

- Ưu điểm:

+ Cú pháp dễ đọc và dễ hiểu.

+ Đa nền tảng và hỗ trợ đa môi trường phát triển.

+ Hỗ trợ mạnh mẽ cho lập trình hướng đối tượng.

+Sự tích hợp tốt với các công nghệ Microsoft khác như Azure và Visual Studio.

+Cộng đồng phát triển lớn và nhiều tài liệu học tập.

- Hạn chế:

+Đòi hỏi học một số khái niệm phức tạp của lập trình hướng đối tượng.

+ Không hoàn toàn mã nguồn mở nhưng có sự hỗ trợ từ cộng đồng mã nguồn mở thông qua các dự án như .NET Core.

+ Cần sử dụng các công cụ và hệ thống môi trường phát triển phức tạp hơn so với một số ngôn ngữ lập trình khác.

## Mục tiêu nghiên cứu

- Tạo ra một trò chơi 2D chơi trên máy tính.

- Phát triển một hệ thống gameplay mạch lạc và đầy thách thức.

- Tạo ra một thế giới game top-down có đồ họa hấp dẫn và môi trường tương tác.

## Phạm vi

- Thiết kế và xây dựng các cấp độ hoặc màn chơi với độ khó tăng dần.

- Tạo ra một hệ thống điều khiển linh hoạt và dễ dàng cho người chơi.

- Phát triển các tính năng như hệ thống vũ khí, vật phẩm, quái vật, và nhiệm vụ.

- Xây dựng một hệ thống AI cho các quái vật hoặc kẻ thù trong trò chơi.

- Thực hiện các tính năng nâng cấp và tiến hóa cho nhân vật hoặc vũ khí của người chơi.

- Tích hợp các yếu tố của trò chơi như âm nhạc, hiệu ứng âm thanh và đồ họa để tạo ra trải nghiệm tuyệt vời cho người chơi.

# CHƯƠNG 2. CÔNG CỤ UNITY ENGINE

## 2.1 Giới thiệu về Unity.

- Unity là một hệ thống phát triển game đa nền tảng nổi tiếng nhất hiện nay. Nó bao gồm công cụ phát triển game (game engine) và tích hợp môi trường phát triển (IDE).

- Ra mắt đầu tiên vào năm 2005 tại sự kiện Apple’s Worldwide Developer Conference bởi nhà sáng lập David Helgason.Tháng 5-2012 theo cuộc khảo sát Game Developer Megazine được công nhận là Game engine tốt nhất cho mobile. Năm 2014 Unity thắng giải “Best Engine” tại giải UK’s annual Develop Industry Exellence.

- Unity cung cấp công cụ dựng hình (kết xuất đồ họa) cho các hình ảnh 2D hoặc 3D, công cụ vật lý (tính toán và phát hiện va chạm), âm thanh, mã nguồn, hình ảnh động, trí tuệ nhân tạo, phân luồng, tạo dòng dữ liệu xử lý, quản lý bộ nhớ, dựng ảnh đồ thị và kết nối mạng. Nhờ có các engine mà công việc làm game trở nên ít tốn kém và đơn giản hơn.

- Một trong các thế mạnh của Unity chính là khả năng hỗ trợ gần như toàn bộ các nền tảng hiện có bao gồm: PlayStation 3, Xbox 360, Wii U, iOS, Android, Windows, Blackberry 10, OS X, Linux, trình duyệt Web và cả Flash. Nói cách khác, chỉ với một gói engine, các studio có thể làm game cho bất kỳ hệ điều hành nào và dễ dàng convert chúng sang những hệ điều hành khác nhau. Đồng thời, đây cũng là giải pháp cho các game online đa nền tảng – có thể chơi đồng thời trên nhiều hệ điều hành, phần cứng khác nhau như Web, PC, Mobile, Tablet….



Hình 2.1 Các nền tảng Unity xuất bản.

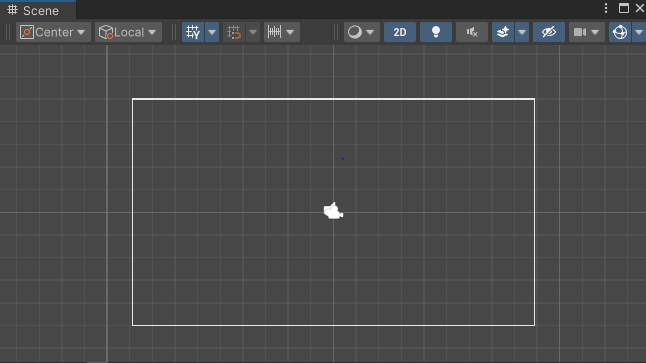
- Unity được built trong một môi trường phát triển tích hợp, cung cấp một hệ thống toàn diện cho các lập trình viên, từ soạn thảo mã nguồn, xây dựng công cụ tự động hóa đến trình sửa lỗi. Do được hướng đến đồng thời cả lập trình viên không chuyên và studio chuyên nghiệp, nên Unity khá dễ sử dụng.

## 2.2 Giao diện Unity Editor.

### 2.2.1 Cửa sổ Scene.

- Đây là cửa sổ của thế giới game trong scene hiện tại, nơi lưu trữ toàn bộ các elements (có thể nhìn thấy hoặc không) của scene. Tại đây có thể theo dõi được vị trí tương đối của chúng cũng như thực hiện một số chỉnh sửa đơn giản.

- Đây là nơi để thiết lập một số thông số như hiển thị ánh sáng, âm anh, cách nhìn 2D hay 3D ... Cửa sổ Scene là nơi bố trí các GameObject như cây cối, cảnh quan, enemy, player, camera, … trong game. Sự bố trí hoạt cảnh là một trong những chức năng quan trọng nhất của Unity.



Hình 2.2 Cửa sổ Scene trong Unity.

- Cửa sổ Scene còn đi kèm với các nút chức năng. Các nút này có thể được lựa chọn để quản lý và chỉnh sửa cảnh trò chơi.



Hình 2.3 Các nút chức năng.

+ Nút Hand Tool: Có biểu tượng hình bàn tay. Nút này cho phép di chuyển đến vị trí nào đó trong Scene bằng cách kéo thả chuột trái.

+ Nút Move Tool: Có biểu tượng hình mũi tên bốn hướng. Cho phép di chuyển các đối tượng theo các trục x, y, và z. Khi công cụ này được chọn, có thể kéo và thả các đối tượng để di chuyển chúng trong không gian.

+ Nút Rotate Tool: Có biểu tượng hình mũi tên cong. Cho phép xoay các đối tượng quanh các trục x, y, và z. Khi công cụ này được chọn, có thể kéo và quay các tay cầm trên các trục để xoay các đối tượng.

+ Nút Scale Tool: Có biểu tượng hình một khối hộp với các mũi tên kéo dài từ các cạnh. Cho phép thay đổi kích thước của các đối tượng bằng cách kéo và thay đổi tỷ lệ trên các trục x, y, và z.

+ Nút Rect Tool: Có biểu tượng hình hộp với các nút điều khiển nhỏ ở các góc. Cho phép thay đổi kích thước, di chuyển đối tượng.

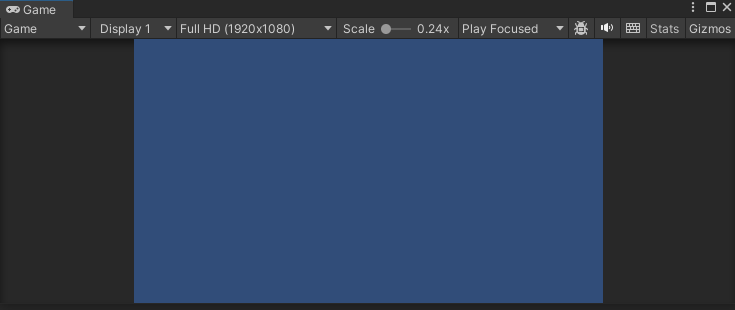
+ Nút Transform Tool: Có biểu tượng hai đầu mũi tên kéo theo hai hướng kết hợp với bốn đường cong và mũi tên bốn hướng ở trung tâm. Cho phép bạn di chuyển, xoay và thay đổi kích thước đối tượng với các handle phù hợp cho từng chức năng.

+ Nút 2D/3D Mode: Có biểu tượng 2D/3D. Cho phép chuyển đổi qua lại giữa chế độ 2D hoặc 3D.

### 2.2.2 Cửa sổ Game.

- Đây là màn hình demo Game, là góc nhìn từ camera trong game.

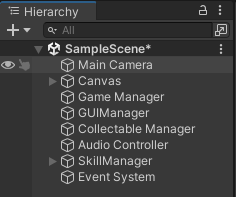
- Thanh công cụ trong cửa sổ game cung cấp các tùy chỉnh về độ phân giải man hình, thông số (stats), gizmos, tùy chọn bật tắt các component...



Hình 2.4 Cửa sổ Game.

### 2.2.3 Cửa sổ Hierachy.

- Đây là nơi lưu trữ toàn bộ các gameObjects có trong scene của bạn khi các đối tượng được thêm hoặc xóa trong Sences, tương ứng với các đối tượng đó trong cửa sổ Hierarchy, đồng thời có một đặc điểm về GameObjects cần lưu ý đó là tính phân cấp (hay phân bậc) thành cây.



Hình 2.5 Cửa sổ Hierachy

- Khi di chuyển hay xoay GameObject cha, các children của nó cũng sẽ di chuyển và xoay tương ứng. Giả sử:

+ Trong GameObject Building chứa nhiều GameObjects Room, mỗi GameObject Room lại có nhiều GameObject Furnitures như vậy khi di chuyển Building bạn sẽ có thể di chuyển toàn bộ trong một lần thay vì di chuyển từng cái.

+ GameObject Player nhặt được 2 GameObjects giày, như vậy đặt 2 GameObjects giày làm con (child) của Player. Như vậy khi player di chuyển, giày sẽ luôn dính vào chân của Player.

- Bên cạnh đó của sổ Hierachy có một số tính năng sau:

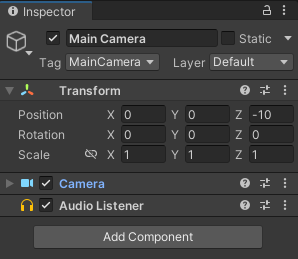
+ Tìm kiếm: Có một hộp tìm kiếm trên đầu cửa sổ Hierarchy cho phép tìm kiếm GameObject theo tên.

+ Lọc: Có một số nút lọc bên dưới hộp tìm kiếm, cho phép lọc các GameObject theo loại, ví dụ: chỉ hiển thị các GameObject có sử dụng Script, chỉ hiển thị các GameObject được chọn, v.v.

+ Kéo và thả: Có thể kéo và thả các GameObject trong cửa sổ Hierarchy để thay đổi cấu trúc cây và mối quan hệ cha con giữa các GameObject.

### 2.2.4 Cửa sổ Inspector

- Cửa sổ Inspector hiển thị chi tiết các thông tin về GameObject đang làm việc, kể cả những component được đính kèm và thuộc tính của nó. Có thể điều chỉnh, thiết lập mọi thông số và chức năng của GameObject thông qua cửa sổ Inspector.



Hình 2.6 Cửa sổ Inspector.

- Ví dụ một gameObject gọi là Player, khi click vào nó trên Hierachy, sẽ theo dõi được vị trí x, y và z trong không gian, tốc độ, hình ảnh hiển thị,… trên cửa sổ Inspector.

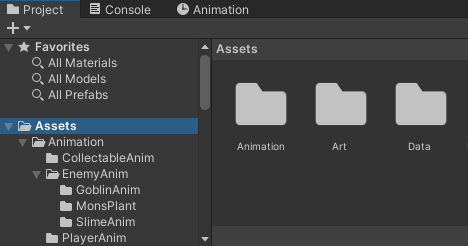
- Tại phần đầu của Inspector, có thể thay đổi tên, gắn nhãn (tag) và chỉ định tầng (layer) cho GameObject hiện đang được chọn.

- Có thể thêm hoặc xóa các thành phần bằng cách nhấn vào nút “***Add Component”*** hoặc dấu ba chấm bên phải ngoài cùng của Component > “***Remove Component***”.

### 2.2.5 Cửa sổ Project.

- Cửa sổ Project trong Unity là nơi quản lý tất cả các tài nguyên (asset) được sử dụng trong dự án, bao gồm tệp tin, hình ảnh, âm thanh, video, script, prefab, và nhiều loại tài nguyên khác.

- Cột bên trái hiển thị assets và các mục yêu thích dưới dạng cây thư mục tương tự như Windows Explorer. Khi click vào một nhánh trên cây thư mục thì toàn bộ nội dung của nhánh đó sẽ được hiển thị ở khung bên phải.



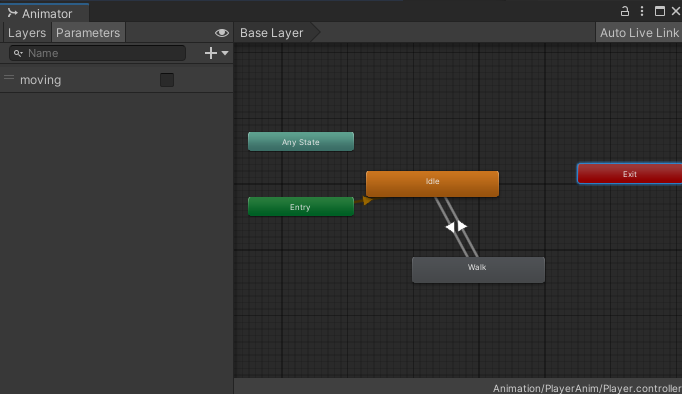
Hình 2.7 Cửa sổ Project.

- Tại cửa sổ Project có thể sử dụng hộp tìm kiếm để nhanh chóng tìm kiếm các tài nguyên trong dự án bằng cách nhập tên tài nguyên hoặc từ khóa tìm kiếm.

- Có thể kéo và thả các tệp tin từ bên ngoài vào cửa sổ Project để thêm chúng vào dự án.

### 2.2.6 Cửa sổ Animator.

- Cửa sổ Animator trong Unity là nơi tạo và chỉnh sửa các trạng thái (state) và chuyển tiếp (transition) của một Animator Controller, điều khiển hành vi hoạt động của các đối tượng trong trò chơi.



Hình 2.8 Cửa sổ Animator.

- Tại cửa sổ Animator hiển thị cấu trúc của Animator Controller bao gồm:

+ Trạng thái (State): Mỗi trạng thái đại diện cho một hành động hoặc trạng thái cụ thể của đối tượng, ví dụ như trạng thái đi, trạng thái đứng yên, hoặc trạng thái nhảy.

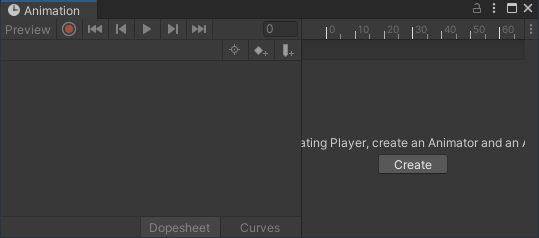
+ Chuyển tiếp (Transition): Các chuyển tiếp xác định cách mà trạng thái chuyển từ một trạng thái này sang trạng thái khác khi xảy ra điều kiện cụ thể.

- Để chuyển đổi các trạng thái của Animator Controller cần tạo ra các parameter. Các loại tham số bao gồm Boolean, Integer, Float, và Trigger.

- Ngoài ra để kết hợp các Animation một cách mượt mà bằng cách tạo Blend Tree.

### 2.2.7 Cửa sổ Animation.

- Cửa sổ Animation trong Unity là nơi tạo, chỉnh sửa và quản lý các hoạt ảnh (animation) cho các GameObject. Đây là một công cụ cho phép thiết kế và điều chỉnh các hoạt động của nhân vật hoặc đối tượng trong trò chơi.

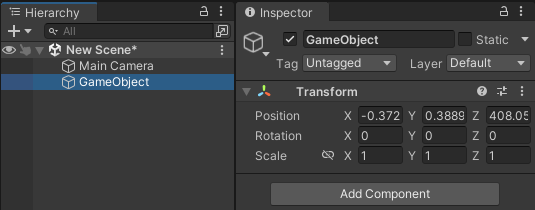


Hình 2.9 Cửa sổ Animation.

## 2.3 Các thành phần trong Unity.

### 2.3.1 GameObject.

- GameObject là thành phần cơ bản nhất trong Unity, được sử dụng để xây dựng mọi thứ trong trò chơi, từ nhân vật, đối tượng môi trường, đến giao diện người dùng.



Hình 2.10 Một Empty GameObject.

- Cấu trúc cơ bản của GameObject bao gồm:

+ Tên: Mỗi GameObject có một tên duy nhất giúp dễ dàng nhận diện nó trong cảnh và trong cửa sổ Hierarchy.

+ Tag: GameObject có thể được gán một hoặc nhiều tag để phân loại và quản lý chúng dễ dàng hơn. Tag thường được sử dụng để nhận diện các nhóm GameObject như "Player", "Enemy", hoặc "Collectible".

+ Layer: GameObject có thể được gán vào một layer để quản lý và phân lớp hiển thị hoặc các tương tác vật lý.

- Các GameObject được tạo đều sẽ được hiển thị ở cửa sổ Hierachy và tất cả GameObject đều có vị trí, xoay và tỉ lệ trong thế giới game.

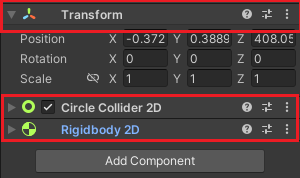
### 2.3.2 Component.

- Component là các thành phần riêng lẻ gắn vào GameObject để xác định hành vi và thuộc tính của chúng. Có thể thêm nhiều Component khác để mở rộng chức năng của GameObject đó, ví dụ như:

+ Vị trí, độ xoay và scale của một GameObject được định nghĩa bởi component có tên là Transform.

+ Rigidbody component làm nhiệm vụ định nghĩa cách mà một object của thể di chuyển một cách “vật lý” trong Game như cân nặng, sức cản không khí, trọng lượng,…

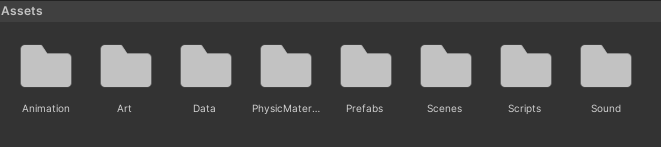
+ Collider component định nghĩa một khối bao quanh GameObject để xử lý va chạm với các GameObject cũng có Collider khác trong scene.



Hình 2.11 Các Component của một GameObject.

### 2.3.3 Assets

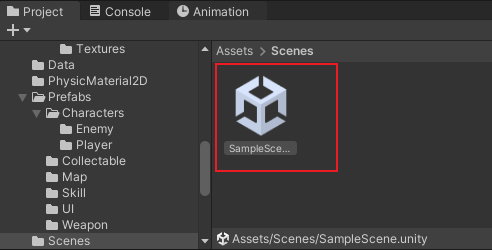
Assets là tất cả các tài nguyên và tệp tin được sử dụng để xây dựng trò chơi, bao gồm mọi thứ từ mô hình 3D, âm thanh, hình ảnh, script, vật liệu, đến các tài liệu hoạt ảnh. Chúng được quản lý trong cửa sổ Project của Unity.



Hình 2.12 Assets của một trò chơi.

### 2.3.4 Scenes.

- Scene là một không gian 2D hoặc 3D chứa tất cả các GameObject và các thành phần liên quan để tạo ra một phần của trò chơi hoặc ứng dụng. Mỗi Scene có thể được coi là một cấp độ, một màn chơi hoặc một màn hình giao diện khác nhau trong trò chơi.

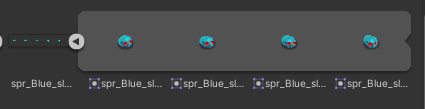


Hình 2.13 Một Scene của Unity.

### 2.3.5 Sprite.

- Sprite là một loại tài nguyên quan trọng trong Unity, được sử dụng chủ yếu trong phát triển các trò chơi 2D. Sprites là các hình ảnh tĩnh hoặc động được hiển thị trong trò chơi và có thể đại diện cho các nhân vật, đối tượng, nền, và nhiều thành phần khác.

- Để thêm sprite vào trong Unity có thể kéo và thả tệp tin hình ảnh vào cửa sổ Project. Sau khi thêm, cần thiết lập các thuộc tính của Sprite trong cửa sổ Inspector.



Hình 2.14 Sprite của trò chơi 2D.

- Cài đặt sprite trong cửa sổ Inspector bao gồm các thuộc tính sau:

+ Texture Type: Đặt loại texture thành Sprite (2D and UI) để Unity nhận diện hình ảnh là một Sprite.

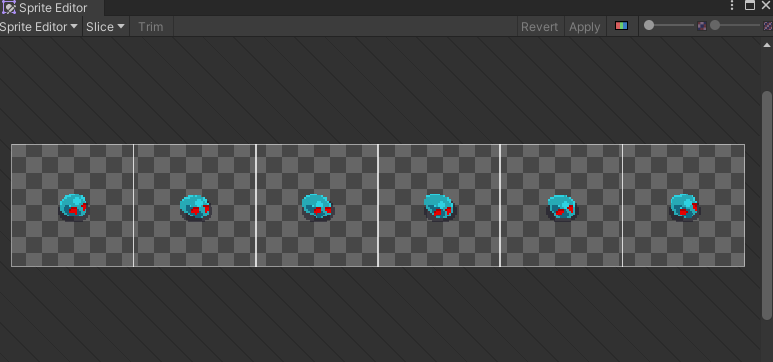
+ Sprite Mode: Dùng Single khi hình ảnh là một Sprite duy nhất hoặc Multiple khi hình ảnh chứa nhiều Sprite (spritesheet).

+ Pixels Per Unit: Xác định tỷ lệ giữa kích thước của Sprite và đơn vị thế giới. Ví dụ, 100 pixels per unit có nghĩa là 100 pixel của hình ảnh sẽ tương ứng với 1 đơn vị thế giới trong Unity.

+ Filter Mode: Sử dụng Point (no filter) cho các hình ảnh pixel để giữ độ sắc nét. Bilinear hoặc Trilinear cho các hình ảnh cần làm mịn.

+ **Compression:** Giảm kích thước tệp tin hình ảnh để tối ưu hóa hiệu suất nhưng có thể ảnh hưởng đến chất lượng hình ảnh.

- Trong Unity cung cấp Sprite Editor cho phép cắt và chỉnh sửa các Sprite từ một spritesheet hoặc từ các hình ảnh nhiều Sprite.

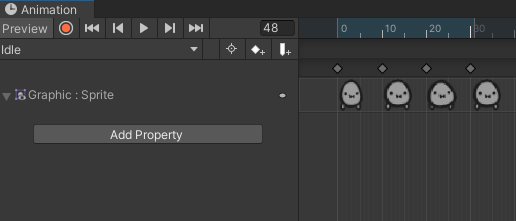


Hình 2.15 Cắt hình ảnh nhiều sprite thành sprite riêng lẻ.

### 2.3.6 Animation.

- Animation là một quá trình tạo ra các chuyển động phức tạp và tương tác cho các đối tượng trong trò chơi. Các đối tượng có thể là nhân vật, môi trường, giao diện người dùng, hoặc bất kỳ GameObject nào.

- Có thể tạo một Animation Clip mới bằng cách nhấp vào *“****Create*** *>* ***Animation****”* trong cửa sổ Project hoặc bằng cách chọn “***Create New Clip”*** trong cửa sổ Animation.



Hình 2.16 Một trạng thái Animation của Player.

### 2.3.7 Prefab.

Là một khái niệm trong Unity, dùng để sử dụng lại các đối tượng giống nhau có trong game mà chỉ cần khởi tạo lại các giá trị vị trí, tỉ lệ biến dạng và góc quay từ môt đối tượng ban đầu. Ví dụ: Các đối tượng là đồng tiền trong game Mario đều có xử lý giống nhau, nên ta chỉ việc tạo ra một đối tượng ban đầu, các đồng tiền còn lại sẽ sử dụng prefabs. Hoặc khi ta lát gạch cho một cái nền nhà, các viên gạch cũng được sử dụng là prefabs.

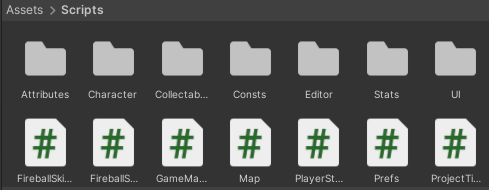


Hình 2.17 Một vài Prefabs trong Unity.

### 2.3.8 Script.

- Script là tập tin mã lệnh được viết để tạo ra các hành vi, logic, và tương tác trong trò chơi. Scripts được viết bằng ngôn ngữ lập trình như C#, và được gắn vào các GameObject để điều khiển chúng.

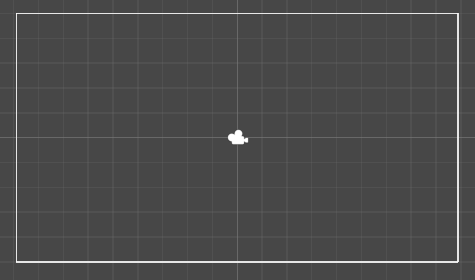
- Để tạo script mới và gắn vào GameObject nhấp chuột phải vào thư mục Project, chọn “***Create > C# Script”***, đặt tên cho script mới. Kéo thả Script từ cửa sổ Project vào GameObject trong cửa sổ Hierarchy hoặc Inspector.



Hình 2.18 Một vài Scrips trong Unity

### 2.3.9 Camera.

Là một GameObject đặc biệt trong scene, dùng để xác định tầm nhìn, quan sát các đối tượng khác trong game.



Hình 2.19 Main Camera của Unity.

# CHƯƠNG 3. MÔ TẢ THIẾT KẾ TRÒ CHƠI

## 3.1 Xây dựng kịch bản trò chơi.

Thế giới yên bình, xinh đẹp của bạn đang bị thế lực quái vật xâm chiếm và phá hủy. Với vai trò là anh hùng bạn phải dũng cảm đứng lên anh dũng chiến đấu với thế lực quái vật và vượt qua nhiều thử thách để đem lại hòa bình cho thế giới.

- Anh hùng: Một nhân vật dũng cảm được trang bị vũ khí để chiến đấu với quái vật.

- Kẻ thù: Hàng loạt quái vật, người chơi phải đánh bại tất cả để khôi phục hòa bình.

- Lối chơi:

+ Người chơi sẽ di chuyển qua các khu vực khác nhau.

+ Mỗi khu vực tồn tại các loại quái vật khác nhau.

+ Hệ thống chiến đấu đơn giản nhưng cần người chơi di chuyển một cách khéo léo để tránh né đòn tấn công của kẻ thù nhằm đánh bại chúng một cách dễ dàng.

- Phát triển nhân vật:

+ Người chơi có thể nâng cấp vũ khí thông qua việc thu thập tiền xu.

+ Trong quá trình tiêu diệt quái vật người chơi có thể nhận được một lượng giá trị kinh nghiệm nhất định để tăng lên đẳng cấp.

+ Bên cạnh đó người chơi cũng có thể thu thập kĩ năng để giúp việc đánh bại quái vật một cách dễ dàng hơn.

+ Tuy nhiên theo đẳng cấp của người chơi tăng lên năng lực của quái vật cũng tăng lên tương ứng.

## 3.2 Chuẩn bị tài nguyên cho trò chơi.

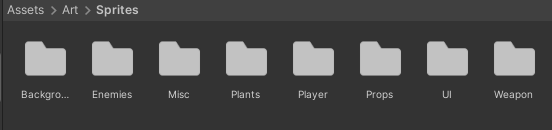
- Để thêm tài nguyên cần thiết cho trò chơi thường được dùng theo nhiều cách khác nhau:

+ Cách 1: Chọn các tệp thư mục tài nguyên cần dùng trong máy tính sau đó kéo và thả các tệp vào cửa sổ “***Project***” của Unity.

+ Cách 2: Trong giao diện của Unity Editor vào menu “***Assets*** > ***Import new Asset...***” và chọn tệp cần thêm vào.

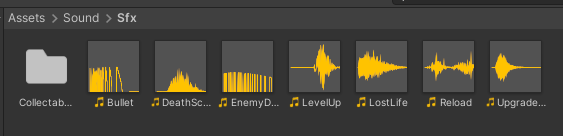
- Dưới đây là một số tài nguyên cần dùng cho trò chơi:

+ Sprites:



Hình 3.1 Sprites cần dùng trò chơi.

+ Âm thanh:

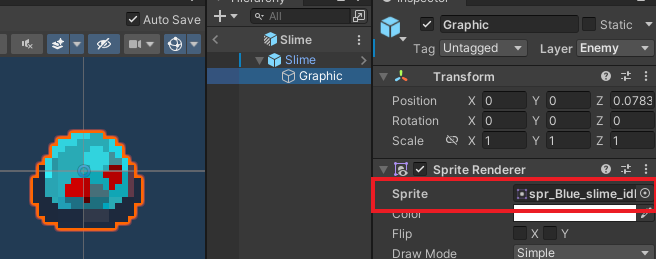


Hình 3.2 Âm thanh cho trò chơi.

## 3.3 Tạo những đối tượng và thêm thành phần vật lý cho tất cả đối tượng.

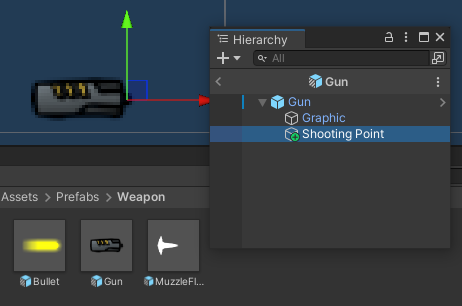
- Trong cửa sổ Hierachy tạo một Empty GameObject đặt tên là BaseObject. Tạo một GameObject Graphic con cho BaseObject có **Component Sprite Renderer** với mục đích chứa hình ảnh cho các đối tượng. Sau đó kéo BaseObject vào của sổ Project để tạo Prefabs có thể tái sử dụng. Dựa vào Prefab BaseObject tạo những đối tượng sau:

+ Đối tượng Enemy: Tại GameObject Graphic kéo sprite cần dùng vào thuộc tính **Sprite** của **Sprite Renderer** để tải hình ảnh cho Enemy.



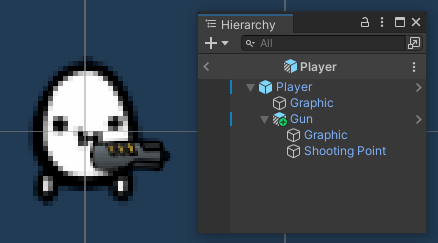
Hình 3.3 Đối tượng Enemy Slime.

+ Đối tượng Weapon: Tương tự như Enemy nhưng với đối tượng súng cần tạo thêm một GameObject con là Shooting point đặt ở đầu súng để khi bắn sẽ tạo ra đạn và tia lửa ở vị trí này.



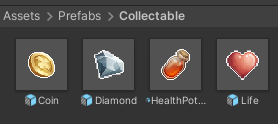
Hình 3.4 Đối tượng Weapon.

+ Đối tượng Player: Tương tự như những đối tượng khác và kéo đối tượng súng vào Player để làm vũ khí cho Player.



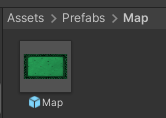
Hình 3.5 Đối tượng Player.

+ Đối tượng Collectable:



Hình 3.6 Đối tượng Collectable.

+ Đối tượng bản đồ:



Hình 3.7 Đối tượng bản đồ.

+ Đối tượng kĩ năng Fireball:

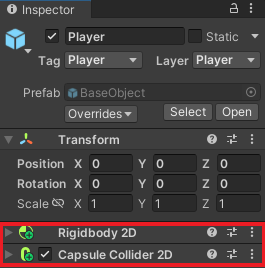


Hình 3.8 Đối tượng Fireball.

- Sau khi tạo thành công các đối tượng ta tiến hành thêm các thành phần vật lý cho các đối tượng cần thiết:

+ Rigidbody 2D: Giúp mô phỏng các hiện tượng vật lý như rơi tự do, va chạm, hoặc trượt trên bề mặt.

+ Collider 2D: Dùng để có thể phát hiện va chạm với các đối tượng khác.



Hình 3.9 Thêm thành phần vật lý cho Player.

## 3.4 Tạo Animation cho đối tượng.

- Để tăng sự cuốn hút của trò chơi việc tạo Animation giúp các đối tượng có thể chuyển động là sự cần thiết. Và để tạo các Animation thì các đối tượng cần phải có một thành phần quản lý gọi là Animator nó giúp tạo ra Animation Controller giúp quản lý các hành vi hoạt động của đối tượng.

+ Thêm thành phần Animator: Chọn các đối tượng cần tạo Animation sau đó nhấn vào nút “***Add Component***” > “***Miscellaneous***”> “***Animator***”.

+ Tạo Animation cho đối tượng: Sau khi thêm thành phần Animator, chọn đối tượng cần tạo Animation. Tại cửa sổ Animation nhấn vào nút “***Create***” để tạo và đặt tên cho Animation Clip.

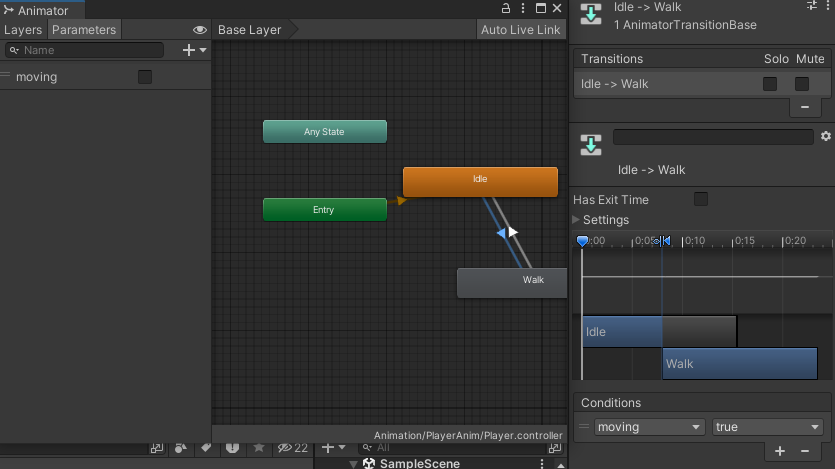
+ Từ Animation Clip vừa tạo có thể nhấn vào dấu ba chấm ở góc phải trên cửa sổ Animation để chọn chế độ chạy (Seconds, Frames) cho Animation Clip. Sau đó thêm và chỉnh sửa các KeyFrames để Animation chạy theo ý muốn.

+ Với một đối tượng có nhiều Animation Clip, vào cửa sổ Animator tạo các Parameters và các Transition để chuyển đổi các Animation.

+ Chọn vào Transition vừa tạo và tại cửa sổ Inspector nhấn dấu + của thuộc tính “***Conditions***”để thêm và thiết lập giá trị của Parameter.



Hình 3.10 Trạng thái đi của Player.



Hình 3.11 Animator quản lý trạng thái của Player.

## 3.5 Thiết kế giao diện.

Thiết kế giao diện trong Unity là một phần quan trọng để tạo ra trải nghiệm người dùng hấp dẫn và trực quan. Dưới đây là các bước để thiết kế giao diện người dùng (UI) cơ bản trong Unity:

- Tạo Canvas: Canvas là thành phần chính của UI trong Unity. Nó chứa tất cả các yếu tố giao diện như nút, hình ảnh, văn bản, v.v.

+ Trong cửa sổ Hierarchy, nhấp chuột phải và chọn “***UI > Canvas***”. Unity sẽ tạo một đối tượng Canvas cùng với một Event System cần thiết cho việc xử lý các sự kiện UI.

+ Thiết lập Canvas: Trong cửa sổ Inspector, có thể điều chỉnh các thuộc tính của Canvas. Hãy chắc chắn rằng “***Render Mode***” được đặt là “***Screen Space - Overlay***” để hiển thị UI trên toàn màn hình.

- Thêm các thành phần UI:

+ Panel: Panel giúp tạo bố cục và nhóm các thành phần UI để dễ dàng quản lý và thiết kế giao diện. Nhấp chuột phải vào Canvas chọn “***UI > Panel***” Unity sẽ tạo một Panel mới là con của Canvas. Trong cửa sổ Inspector, có thể thay đổi các thuộc tính của Panel như Color, Image, Rect Transform, v.v.

+ Button: Nhấp chuột phải vào Canvas trong cửa sổ Hierarchy và chọn “***UI > Button***”. Một đối tượng Button sẽ được tạo ra. Có thể di chuyển, thay đổi kích thước và chỉnh sửa văn bản trên nút bằng cách điều chỉnh thuộc tính của nó trong cửa sổ Inspector.

+ Text: Nhấp chuột phải vào Canvas trong Hierarchy và chọn “***UI > Legacy > Text***”. Sử dụng Inspector để chỉnh sửa văn bản, kích thước, phông chữ, màu sắc và căn chỉnh của nó.

+ Image: Nhấp chuột phải vào Canvas trong Hierarchy và chọn “***UI > Image***”. Có thể thay đổi hình ảnh hiển thị bằng cách kéo thả sprite vào thuộc tính “***Source Image***” trong Inspector.



Hình 3.12 Giao diện bắt đầu trò chơi.



Hình 3.13 Giao diện chơi trò chơi.



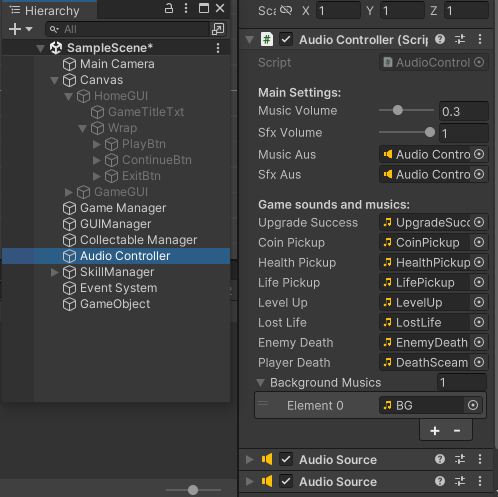
Hình 3.14 giao diện nâng cấp súng.

## 3.6 Thiết kế âm thanh.

Âm thanh là yếu tố không kém phần quan trọng trong trò chợi. Hãy tưởng tượng một cảnh đánh nhau hay các hiệu ứng đẹp mắt mà không có âm thanh. Vì thế âm thanh là yếu tố cần thiết để một trò chơi thực tế và sống động hơn.

- Tại cửa sổ Hierachy, nhấn chuột phải và chọn “***Audio > Audio Source***” để tạo đối tượng âm thanh. Sau đó kéo tệp âm thanh từ Project vào thuộc tính “***Audio Clip***” của “***Audio Source***”.

- Điều chỉnh các thuộc tính khác như “***Volume***”, “***Pitch***”, “***Loop***”,... để được đối tượng âm thanh ưng ý nhất.



Hình 3.15 Âm thanh trong trò chơi.

## 3.7 Xử lý hành vi cho đối tượng.

## 3.8 Tạo lớp lưu trữ dữ liệu.

# CHƯƠNG 4. KẾT QUẢ VÀ KIẾN NGHỊ

## 4.1 Kết quả đạt được.

## 4.2 Hạn chế.

## 4.3 Hướng phát triển.