TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN TỐT NGHIỆP

ĐỀ TÀI

XÂY DỰNG GAME SHADOW ADVENTURE

Giảng viên hướng dẫn : Th.s Đỗ Văn Đức

Sinh viên thực hiện : Phạm Văn Lương

Lớp : CNTT6

Mã sinh viên : 201200222

Hà Nội - 2024

LỜI CẢM ƠN

Trong thời gian học tập tại trường Đại học Giao thông vận tải, em đã được các thầy cô giáo giảng dạy tận tình, truyền đạt cho em những kiến thức rất bổ ích để cho em có được những vốn kiến thức rất quan trọng cho chuyên ngành của em sau này. Trên thực tế không có sự thành công nào mà không gắn liền với những sự hỗ trợ, giúp đỡ dù ít hay nhiều, dù trực tiếp hay gián tiếp của người khác. Em xin chân thành cảm ơn các thầy cô của khoa Công nghệ thông tin đã tận tâm giảng dạy cho em để giúp em trang bị kiến thức cơ bản trong 4 năm đại học để em có thể thực hiện và hoàn thành đồ án tốt nghiệp này.

Trong suốt quá trình thực hiền đề tài đồ án tốt nghiệp, ngoài những cố gắng bản thân thì em sẽ không thể hoàn thành tốt được nếu không có sự giúp đỡ và chỉ bảo tận tình của thầy Đỗ Văn Đức. Em xin chân thành cảm ơn thầy đã giúp đỡ em trong suốt quá trình thực hiện đề tài.

Mặc dù em đã cố gắng hoàn thành đề tài đồ án này trong phạm vi và khả năng cho phép, nhưng chắc chắn không tránh khỏi những thiếu sót. Em rất mong nhận được những lời góp ý của thầy cô để đề tài của em được hoàn thiện hơn. Đồng thời giúp em có thêm kiến thức áp dụng những kinh nghiệm cho các dự án tiếp theo trong tương lai.

Em xin chân thành cảm ơn!

Hà Nội, ngày …. tháng …. năm 2024

Sinh viên thực hiện

Phạm Văn Lương

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

Hà Nội, ngày…..tháng…. năm 2024

Giảng viên hướng dẫn

ThS. Đỗ Văn Đức

MỤC LỤC

DANH MỤC TỪ VIẾT TẮT

STT	Từ viết tắt	Tên đầy đủ	Giải thích
1	API	Application Programming Interface	phương thức trung gian kết nối các ứng dụng và thư viện khác nhau.
2	IDE	Integrated Development Environment	môi trường phát triển tích hợp
3	GC pauses	Garbage Collection Pauses	hiện tượng tạm dừng không mong muốn khi chương trình giải phóng bộ nhớ.

DANH MỤC HÌNH ẢNH

MỞ ĐẦU

1. Lý do chọn đề tài

Trong kỷ nguyên số hóa hiện nay, ngành công nghiệp trò chơi điện tử đã và đang phát triển mạnh mẽ, trở thành một phần quan trọng của giải trí và văn hóa đại chúng. Đặc biệt, game 2D vẫn giữ được vị thế quan trọng nhờ vào tính sáng tạo, phong cách nghệ thuật độc đáo, và khả năng tiếp cận rộng rãi đến mọi lứa tuổi. Sự phát triển của các công cụ và nền tảng hỗ trợ, như Unity, đã mở ra cơ hội cho nhiều nhà phát triển độc lập và các nhóm nhỏ tham gia vào việc tạo ra các tựa game chất lượng cao mà không cần đầu tư quá nhiều vào công nghệ phức tạp.

Đồ án phát triển game 2D thể loại platform với tên gọi "Shadow Adventure" được thực hiện với mục đích khám phá và ứng dụng các kỹ thuật lập trình và thiết kế game hiện đại, từ đó tạo ra một sản phẩm game ưng ý. Unity, với ưu điểm là một công cụ phát triển game mạnh mẽ và dễ sử dụng, đã được chọn làm nền tảng chính cho dự án này. Thông qua quá trình phát triển, tôi không chỉ học hỏi về các khía cạnh kỹ thuật mà còn về quản lý dự án, và quan trọng nhất là cách tạo ra trải nghiệm người chơi hấp dẫn.

"Shadow Adventure" là một trò chơi platform 2D, nơi người chơi sẽ nhập vai vào một nhân vật bóng tối, vượt qua các chướng ngại vật, chiến đấu với kẻ thù và khám phá các bí ẩn của thế giới bóng tối. Trò chơi được thiết kế để mang lại những thử thách độc đáo, kết hợp giữa kỹ năng di chuyển, nhảy, leo trèo và tấn công, tạo nên một trải nghiệm hấp dẫn và lôi cuốn.

2. Phương pháp thực hiện

Để thực hiện đồ án này, các bước sau sẽ được tiến hành:

- ❖ Nghiên cứu và thu thập tài liệu: Tìm hiểu các tài liệu liên quan đến phát triển game 2D bằng Unity, và cách xây dựng cốt truyện, nhân vật.
- ❖ Lên ý tưởng và thiết kế: Xây dựng ý tưởng chi tiết cho trò chơi, bao gồm cốt truyện, nhân vật, môi trường và các cơ chế chơi
- ❖ Phát triển game: Sử dụng Unity để lập trình và tích hợp các tài sản đồ họa vào trò chơi. Tập trung vào việc phát triển các cơ chế chính của trò chơi như di chuyển nhân vật, tương tác với môi trường, và các hiệu ứng đánh quái, hiệu ứng khi chết.
- ❖ Kiểm thử và tinh chỉnh: Thực hiện kiểm thử trò chơi để phát hiện và sửa chữa lỗi. Tinh chỉnh các yếu tố của trò chơi để đảm bảo trải nghiệm người chơi tốt nhất.

- ❖ Hoàn thiện và trình bày: Hoàn thiện trò chơi và chuẩn bị các tài liệu cần thiết để trình bày đồ án, bao gồm báo cáo chi tiết và hướng dẫn sử dụng.

3. Bố cục đồ án

Đồ án "Game Shadow Adventure" sẽ được trình bày theo cấu trúc sau:

- ❖ Mở Đầu: Giới thiệu về lý do chọn đề tài, phương pháp thực hiện và bố cục đồ án.
- ❖ Tổng Quan Về Đề Tài: Giới thiệu tổng quan về thể loại game, các nghiên cứu và công nghệ liên quan
- ❖ Phân Tích Thiết Kế Game: Trình bày các kiến thức nền tảng về phát triển game 2D, các kỹ thuật và công cụ của Unity, và các yếu tố thiết kế game như cốt truyện, nhân vật, môi trường.
- ❖ Xây Dựng Chương Trình: Mô tả chi tiết về các chức năng trong game, các level hiện tại của game.
- ❖ Tổng Kết Và Hướng Phát Triển: Trình bày những gì đã đạt được, điểm yếu, điểm mạnh, hướng phát triển của game.
- ❖ Tài Liệu Tham Khảo: Liệt kê danh sách các tài liệu tham khảo dùng trong báo cáo

CHƯƠNG 1. TỔNG QUAN VỀ ĐỀ TÀI

1.1. Giới thiệu Game

1.1.1. Tổng quan về thể loại game Adventure [1]

1.1.1.1. Đặc điểm của game Adventure:

- ❖ Cốt truyện phong phú: Thể loại game Adventure thường có cốt truyện sâu sắc và phức tạp, tập trung vào việc khám phá và phiêu lưu.
- ❖ Tương tác với môi trường: Người chơi có thể tương tác với các đối tượng trong game như đồ vật, nhân vật phụ (NPC) để tiến hành nhiệm vụ.
- ❖ Giải đố: Các câu đố và thử thách logic là phần quan trọng của gameplay.
- ❖ Phát triển nhân vật: Người chơi có thể phát triển kỹ năng và trang bị của nhân vật theo thời gian.

1.1.1.2. Lịch sử phát triển của thể loại game Adventure:

- ❖ Những năm 1980: Các game phiêu lưu văn bản như "Zork" mở đường cho thể loại này.
- ❖ Những năm 1990: Sự xuất hiện của đồ họa 2D và 3D đã tạo nên các tựa game nổi bật như "The Legend of Zelda" và "Monkey Island".
- ❖ Những năm 2000 và sau đó: Các game như "The Elder Scrolls", "Assassin's Creed", và "Hollow Knight" đã đưa thể loại này lên tầm cao mới với đồ họa tiên tiến và gameplay phức tạp.

1.1.1.3. Các game nổi bật trong thể loại Adventure:

- ❖ The Legend of Zelda: Một trong những series game Adventure nổi tiếng nhất, nổi bật với thế giới mở và cốt truyện hấp dẫn.
- ❖ Hollow Knight: Một game 2D Adventure được đánh giá cao với lối chơi thử thách và thế giới ngầm phức tạp.
- ❖ Monkey Island: Được biết đến với lối chơi point-and-click và các câu đố hài hước.

1.1.2. Tổng quan về sản phẩm

Trong Shadow Adventure người chơi điều khiển nhân vật chính, một thợ săn quái vật, đi qua các màn chơi, tiêu diệt quái vật và vượt qua chướng ngại vật để đến đích. Mỗi màn chơi kết thúc khi người chơi đạt đến điểm đích hoặc tiêu diệt boss. Game có tất cả là 3 level, độ khó cứ tăng dần theo mỗi level.

1.2. Tổng quan về Unity [2]

Hình 1.1. Logo của Unity

1.2.1. Khái niệm Unity

Unity là một nền tảng phát triển trò chơi và ứng dụng đa nền tảng hàng đầu trên thị trường hiện nay. Được phát triển bởi Unity Technologies, Unity cung cấp một môi trường mạnh mẽ và linh hoạt cho các nhà phát triển để tạo ra các trò chơi 2D, 3D, các ứng dụng thực tế ảo (AR), thực tế ảo (VR), và các ứng dụng tương tác 3D khác.

1.2.2. Quá trình phát triển của Unity

Ra mắt đầu tiên vào năm 2005 tại sự kiện Apple's Worldwide Developer Conference bởi nhà sáng lập David Helgason, trải qua hơn 12 năm phát triển, nay Unity đã có version 5.5 hoàn thiện hơn về rất nhiều mặt. 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.

1.2.3. Những tính năng nổi bật của Unity

1.2.3.1. Đa nền tảng

Unity hỗ trợ xuất bản game trên nhiều nền tảng, bao gồm Windows, macOS, Linux, iOS, Android, WebGL, các console như PlayStation và Xbox, và thậm chí cả VR và AR.

Hình 1.2. Unity hỗ trợ phát triển đa nền tảng

1.2.3.2. Công cụ đồ họa mạnh mẽ

- ❖ Scriptable Render Pipeline (SRP): Cho phép tùy chỉnh các quy trình render để tối ưu hóa hiệu suất.
- ❖ High Definition Render Pipeline (HDRP): Mang lại chất lượng đồ họa cao cho các game PC và console.
- ❖ Universal Render Pipeline (URP): Cung cấp đồ họa chất lượng cao cho các thiết bị di động và VR/AR.

1.2.3.3. Hệ thống animation

- ❖ Mecanim: Hệ thống animation mạnh mẽ hỗ trợ blend trees, state machines và inverse kinematics.

1.2.3.4. Công cụ phát triển

- ❖ Visual Scripting: Cho phép tạo các logic game mà không cần viết code.
- ❖ Công cụ Editor: Môi trường phát triển trực quan với nhiều công cụ hỗ trợ thiết kế, debug và tối ưu hóa game.

1.2.4. Ứng dụng và cộng đồng

1.2.4.1. Các dự án nổi bật

- ❖ Unity đã được sử dụng để phát triển nhiều tựa game nổi tiếng như Hearthstone, Monument Valley, Cuphead, và Ori and the Blind Forest.
- ❖ Ngoài game, Unity còn được sử dụng trong các lĩnh vực khác như kiến trúc, phim ảnh, mô phỏng và giáo dục.

1.2.4.2. Cộng đồng và hỗ trợ

- ❖ Unity có một cộng đồng lớn và năng động, với nhiều tài liệu hướng dẫn, diễn đàn, và khóa học trực tuyến

❖ Unity Technologies cung cấp các dịch vụ hỗ trợ, bao gồm Unity Learn, Asset Store và Unity Connect.

1.2.5. Ưu và nhược điểm của Unity

❖ Ưu điểm

➢ Đa nền tảng

- Hỗ trợ nhiều nền tảng: Unity cho phép phát triển game trên nhiều nền tảng khác nhau, bao gồm Windows, macOS, Linux, iOS, Android, WebGL, các console như PlayStation và Xbox, và VR/AR.
- Quy trình build dễ dàng: Công cụ build tích hợp giúp chuyển đổi giữa các nền tảng một cách dễ dàng mà không cần thay đổi nhiều mã nguồn.

➢ Cộng đồng và tài liệu phong phú

- Cộng đồng lớn: Có một cộng đồng nhà phát triển đông đảo và năng động, cung cấp sự hỗ trợ và chia sẻ kiến thức thông qua các diễn đàn, nhóm, và các kênh truyền thông xã hội.
- Tài liệu và khóa học: Unity có nhiều tài liệu hướng dẫn, khóa học trực tuyến và tài liệu tham khảo chi tiết giúp người dùng dễ dàng học và làm quen.

➢ Công cụ mạnh mẽ và dễ sử dụng

- Unity Editor: Giao diện người dùng trực quan và dễ sử dụng giúp nhà phát triển dễ dàng tạo và chỉnh sửa các dự án.
- Visual Scripting: Hỗ trợ tạo các logic game mà không cần viết code, giúp những người không chuyên về lập trình cũng có thể phát triển game.

➢ Công nghệ đồ họa tiên tiến

- Scriptable Render Pipeline (SRP): Cho phép tùy chỉnh các quy trình render để tối ưu hóa hiệu suất và chất lượng đồ họa.
- High Definition Render Pipeline (HDRP) và Universal Render Pipeline (URP): Đem lại chất lượng đồ họa cao cho các thiết bị mạnh mẽ và tối ưu hóa cho các thiết bị di động.

➢ Asset Store phong phú

- Asset Store: Cung cấp hàng nghìn asset, plugin, script và các công cụ hỗ trợ khác giúp tiết kiệm thời gian và công sức phát triển.

➢ Khả năng tích hợp và mở rộng

- Plugin và API: Unity hỗ trợ tích hợp với nhiều plugin và API của bên thứ ba, giúp mở rộng tính năng và khả năng của công cụ.

❖ Nhược điểm

➢ Chi phí

- Giá thành cao: Mặc dù có phiên bản miễn phí, nhưng các phiên bản Unity Plus, Pro có giá thành khá cao, đặc biệt đối với các studio nhỏ hoặc nhà phát triển độc lập.

➢ Hiệu suất
- Tối ưu hóa: Các game phát triển trên Unity đôi khi gặp vấn đề về hiệu suất, đặc biệt trên các thiết bị di động hoặc khi phát triển các game đồ họa nặng. Đòi hỏi nhà phát triển phải có kiến thức sâu rộng về tối ưu hóa.
- Dung lượng build: Dung lượng của file build trên Unity thường lớn hơn so với các engine khác.

➢ Phụ thuộc vào plugin
- Phụ thuộc vào Asset Store: Nhiều nhà phát triển phụ thuộc vào các asset và plugin từ Asset Store, có thể dẫn đến các vấn đề về tương thích và bảo trì khi các plugin không được cập nhật thường xuyên.

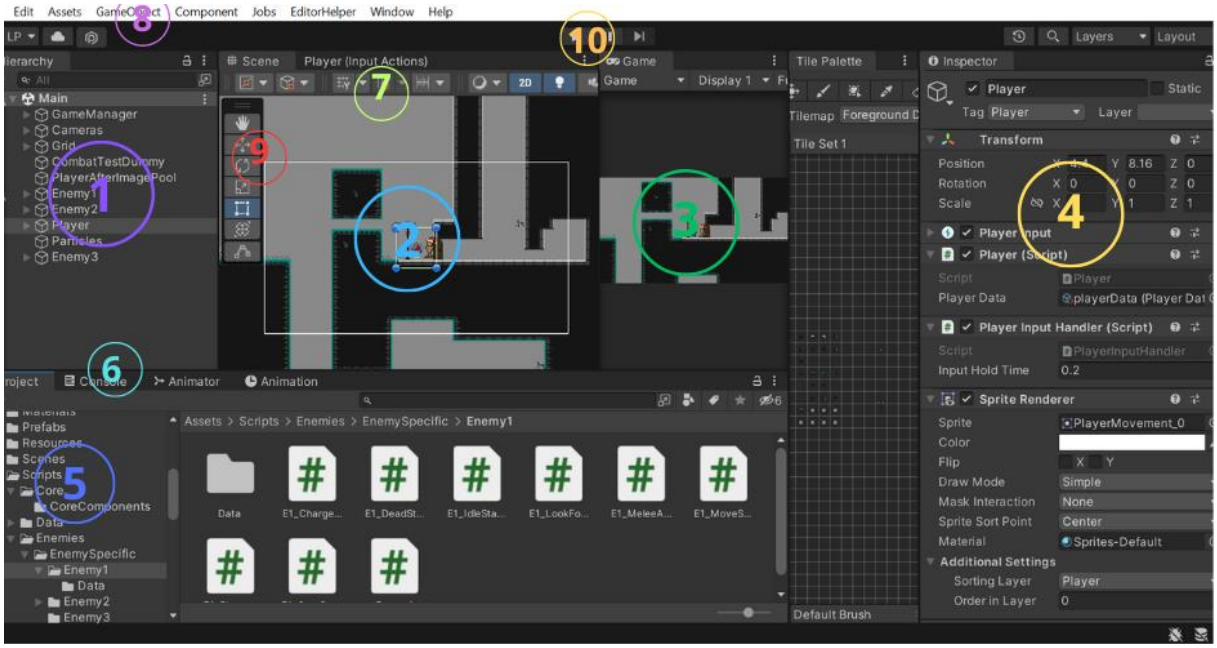
➢ Hạn chế trong phát triển 2D
- Khả năng 2D: Mặc dù Unity đã cải thiện đáng kể khả năng phát triển game 2D, nhưng vẫn còn một số hạn chế so với các engine chuyên về 2D như Godot hoặc GameMaker.

➢ Yêu cầu học tập cao
- Độ phức tạp: Mặc dù Unity dễ sử dụng đối với các dự án đơn giản, nhưng việc làm chủ toàn bộ công cụ và tối ưu hóa dự án yêu cầu một lượng lớn thời gian và công sức học tập.

➢ Vấn đề về sự nhất quán
- Cập nhật và thay đổi: Unity thường xuyên cập nhật và thay đổi các tính năng, có thể gây khó khăn trong việc duy trì sự nhất quán và ổn định cho các dự án dài hạn.

1.3. Tìm hiểu về Unity Engine

1.3.1. Các thành phần quan trọng trong Unity Editor



Hình 1.3. Các thành phần quan trọng trong Unity Editor

1.3.1.1. Hierarchy Window

- ❖ Hiển thị cấu trúc của tất cả các GameObject trong Scene.
- ❖ Cho phép bạn tổ chức và thao tác với các đối tượng trong Scene như thêm, xóa, sắp xếp và nhóm chúng.

1.3.1.2. Scene View

- ❖ Cho phép bạn xem và chỉnh sửa các đối tượng trong Scene.
- ❖ Đây là nơi bạn tạo, di chuyển và cấu hình vị trí, góc quay và tỷ lệ của các đối tượng.

1.3.1.3. Game View

- ❖ Hiển thị trực tiếp màn hình trò chơi trong quá trình chạy.
- ❖ Cho phép bạn kiểm tra và kiểm tra trải nghiệm người chơi mà không cần chạy lại trò chơi từ đầu.

1.3.1.4. Inspector Window

- ❖ Hiển thị thông tin chi tiết và thuộc tính của các GameObject hoặc Component đang được chọn.
- ❖ Cho phép bạn chỉnh sửa các thuộc tính và cấu hình của các đối tượng.

1.3.1.5. Project Window

- ❖ Hiển thị tất cả các tài nguyên trong dự án của bạn như các file tài nguyên, script, prefab, và các folder tổ chức.
- ❖ Cho phép bạn quản lý và tổ chức các tài nguyên trong dự án.

1.3.1.6. Console Window

- ❖ Hiển thị thông tin debug, cảnh báo và lỗi khi chạy trò chơi.
- ❖ Rất hữu ích để gỡ lỗi và phát hiện các vấn đề trong mã nguồn của bạn.

1.3.1.7. Toolbar

- ❖ Cung cấp các công cụ và chức năng phổ biến giúp bạn tạo, chỉnh sửa và làm việc với dự án của mình.
- ❖ Bao gồm các công cụ như Move Tool, Rotate Tool, Scale Tool, và nút Play để chạy trò chơi.

1.3.1.8. Menu Bar

- ❖ Chứa các menu và các lựa chọn lớn như File, Edit, GameObject, Component, Window, và Help.
- ❖ Cung cấp quyền truy cập đến các chức năng và công cụ khác nhau trong Unity Editor.

1.3.1.9. Toolbar (Scene View)

- ❖ Cung cấp các công cụ và chức năng cụ thể cho việc chỉnh sửa Scene View.
- ❖ Bao gồm các công cụ như Move Tool, Rotate Tool, Scale Tool, và các công cụ khác để tạo và chỉnh sửa đối tượng.

1.3.1.10. Toolbar (Game View)

- ❖ Cung cấp các công cụ và chức năng cụ thể cho việc xem và kiểm tra Game View.
- ❖ Bao gồm các nút để chạy trò chơi, tạm dừng, dừng lại, và chế độ xem khác.

1.3.2. Các khái niệm cơ bản trong Unity

1.3.2.1. GameObject

- ❖ Khái niệm: GameObject là đối tượng cơ bản nhất trong Unity. Mọi thứ trong Scene đều là GameObject, từ các nhân vật, môi trường, đến các đối tượng vô hình như đèn và camera.
- ❖ Chức năng: GameObject chỉ là một container rỗng cho các Components. Tính năng của GameObject được xác định bởi các Components gắn vào nó.

Hình 1.4. GameObject trong Unity

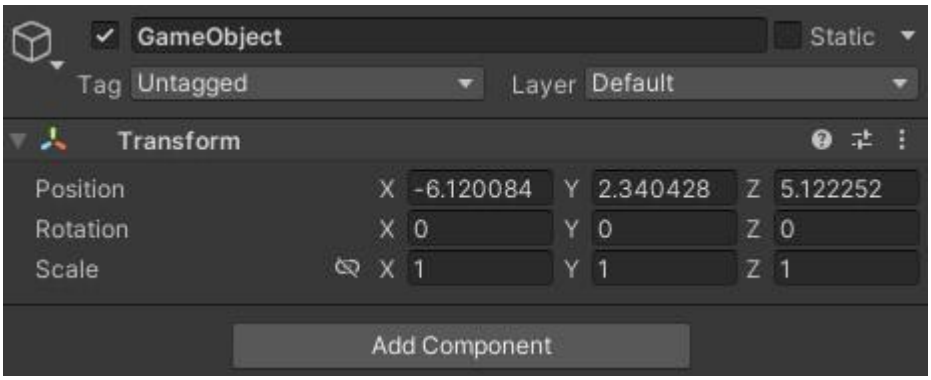
1.3.2.2. Component

- ❖ Khái niệm: Component là các mô-đun chức năng có thể được gắn vào GameObject. Mỗi Component thêm một chức năng cụ thể cho GameObject.
- ❖ Chức năng: Các Components phổ biến bao gồm Transform, Rigidbody, Collider, Script, Audio Source, và nhiều hơn nữa.

Hình 1.5. Component trong Unity

1.3.2.3. Transform

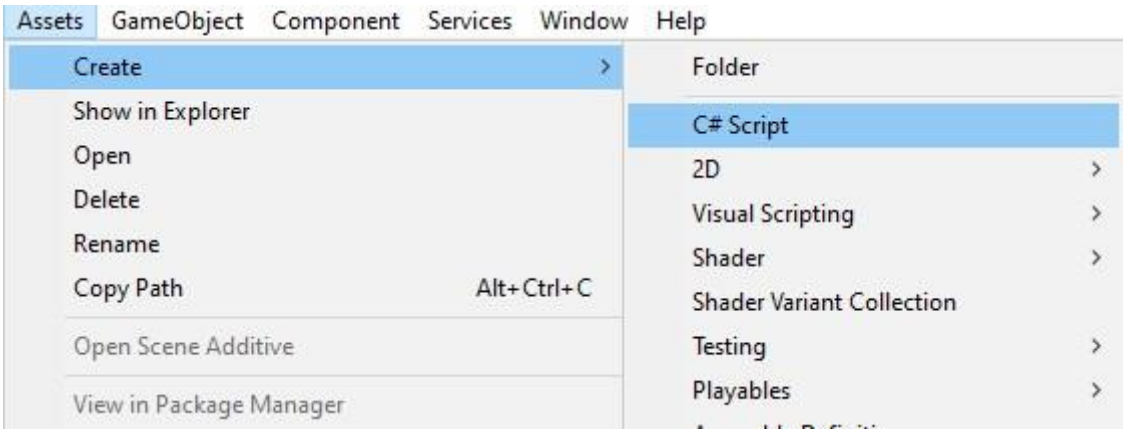
- ❖ Khái niệm: Transform là một Component mặc định của tất cả GameObject.
- ❖ Chức năng: Xác định vị trí (Position), góc quay (Rotation), và tỷ lệ (Scale) của GameObject trong không gian ba chiều.



Hình 1.6. Transform trong Unity

1.3.2.4. Script

- ❖ Khái niệm: Script là các đoạn mã được viết để điều khiển hành vi của GameObject.
- ❖ Chức năng: Script trong Unity thường được viết bằng C# và gắn vào GameObject dưới dạng Component.



Hình 1.7. Cách tạo Script mới

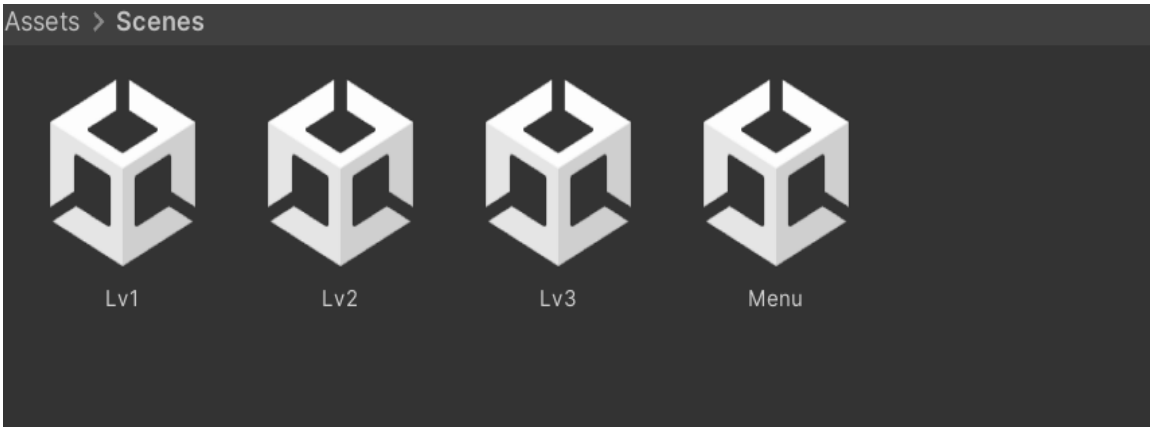
1.3.2.5. Prefab

- ❖ Khái niệm: Prefab là các mẫu GameObject hoặc nhóm GameObject đã được lưu trữ lại để tái sử dụng.
- ❖ Chức năng: Giúp tiết kiệm thời gian và công sức khi tạo nhiều bản sao của một đối tượng có cấu trúc phức tạp hoặc đã được cấu hình sẵn.

Hình 1.8. Prefab trong Unity

1.3.2.6. Scene

- ❖ Khái niệm: Scene là một không gian hoặc màn chơi trong game của bạn.
- ❖ Chức năng: Mỗi Scene chứa các GameObject và có thể lưu trữ môi trường, cấp độ hoặc các thành phần cụ thể của game. Bạn có thể có nhiều Scene và chuyển đổi giữa chúng.



Hình 1.9. Scene trong Unity

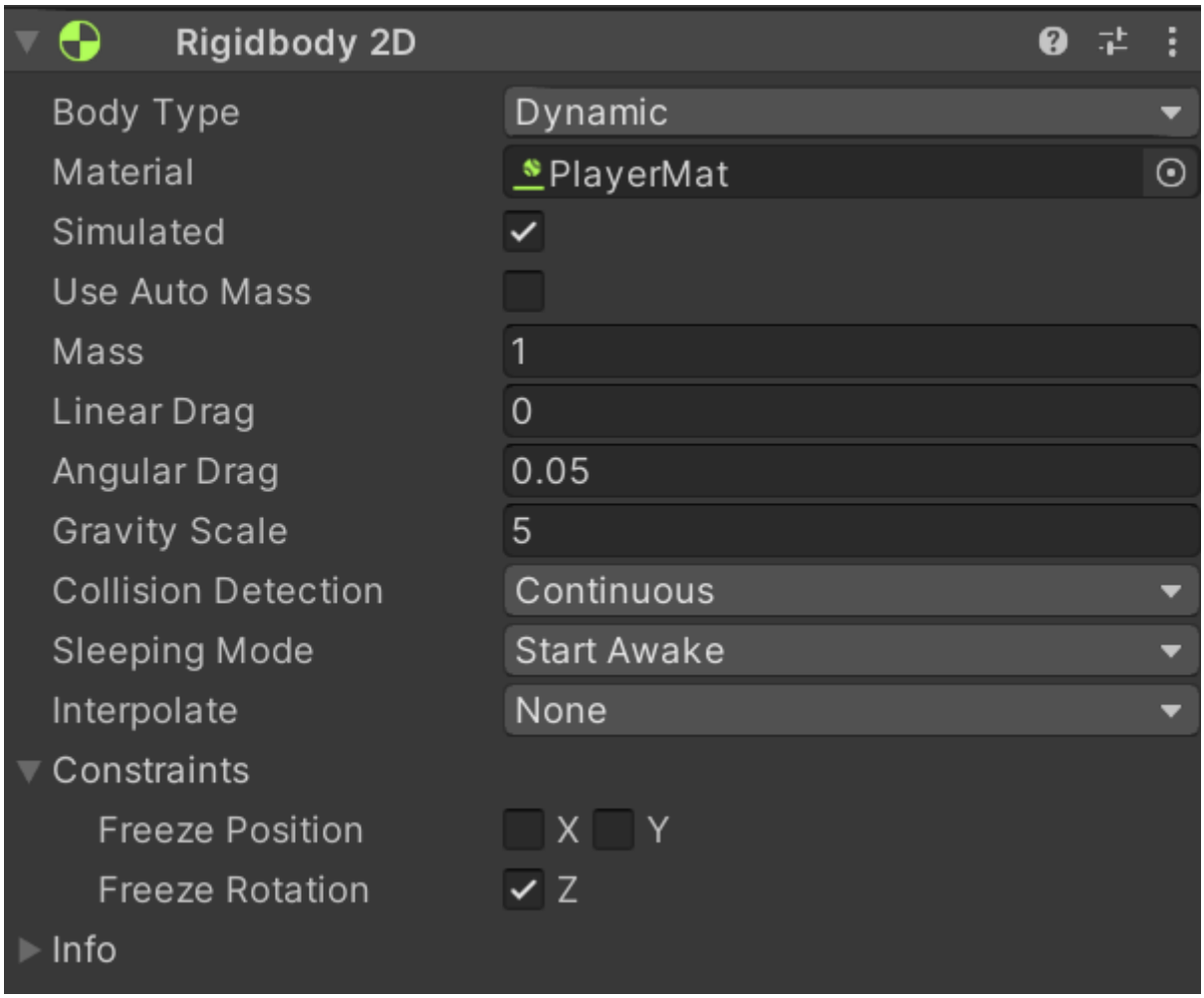
1.3.2.7. Asset

- ❖ Khái niệm: Asset là các tài nguyên được sử dụng trong dự án của bạn, bao gồm mô hình 3D, âm thanh, hình ảnh, texture, và script.
- ❖ Chức năng: Được lưu trữ trong Project Window và có thể kéo vào Scene hoặc Prefab.

Hình 1.10. Asset trong Unity

1.3.2.8. Rigidbody

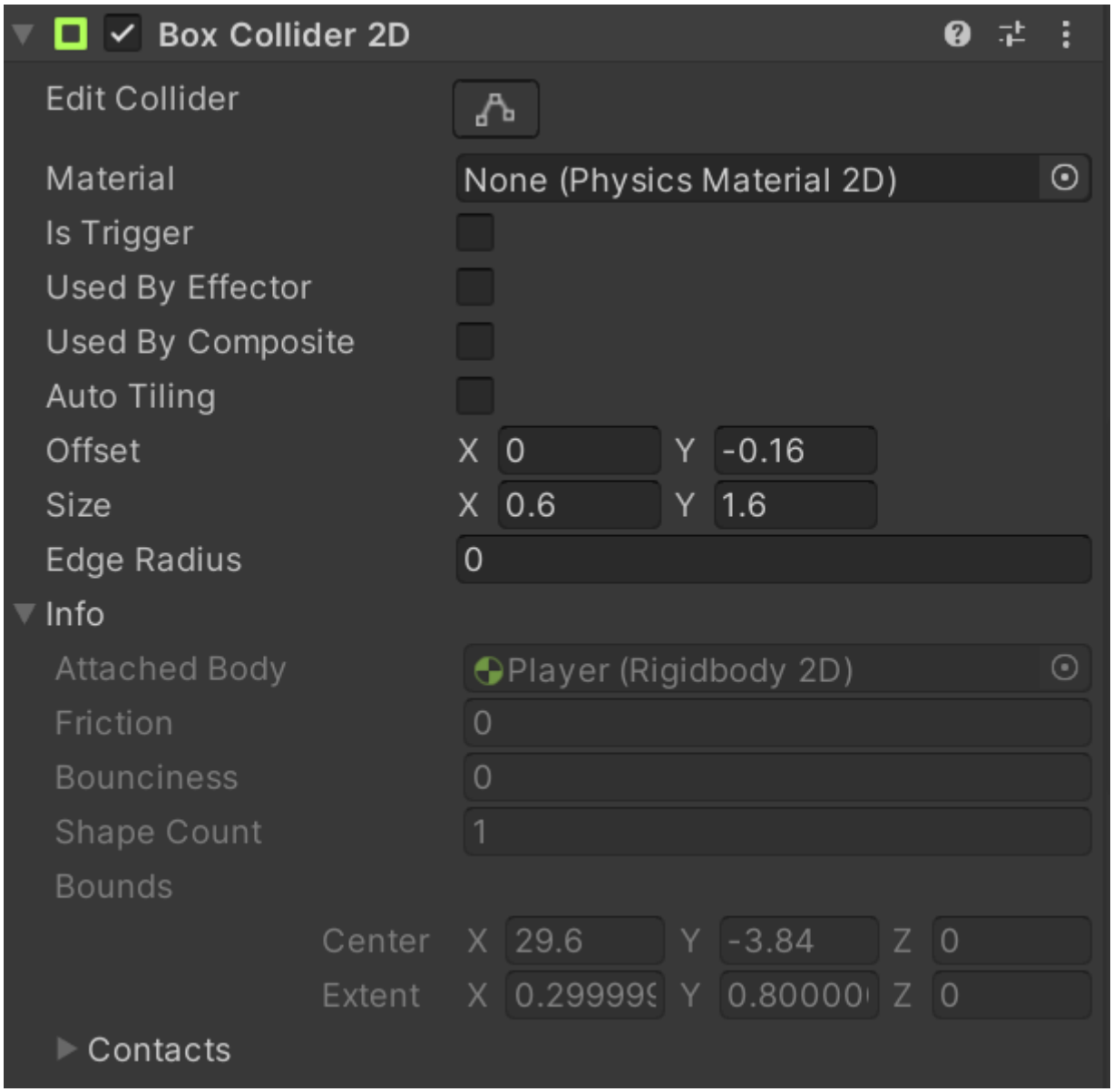
- ❖ Khái niệm: Rigidbody là một Component thêm tính vật lý cho GameObject.
- ❖ Chức năng: Giúp GameObject chịu tác động của các lực vật lý như trọng lực, va chạm và ma sát.



Hình 1.11. Rigidbody trong Unity

1.3.2.9. Collider

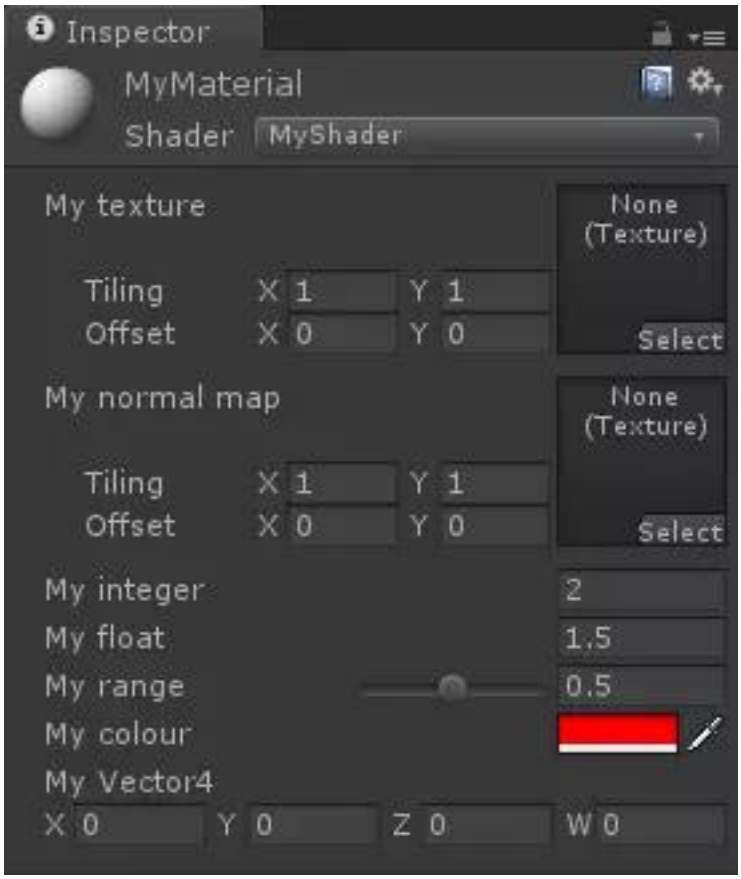
- ❖ Khái niệm: Collider là một Component định nghĩa vùng không gian mà GameObject có thể va chạm.
- ❖ Chức năng: Collider kết hợp với Rigidbody để xử lý va chạm vật lý.



Hình 1.12. Collider trong Unity

1.3.2.10. Material và Shader

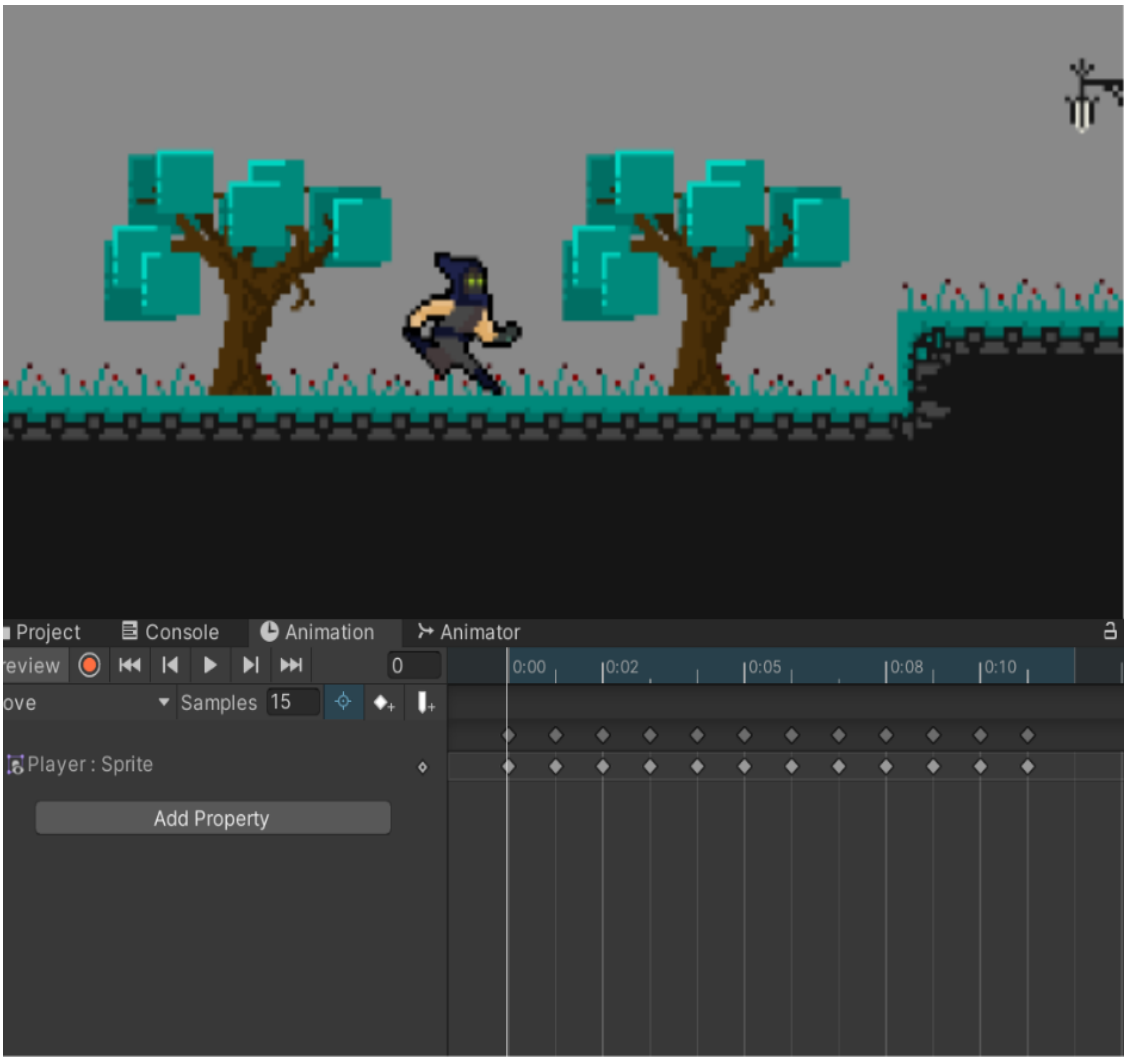
- ❖ Material: Là một Asset xác định cách một bề mặt của GameObject sẽ được hiển thị.
- ❖ Shader: Là một chương trình nhỏ chạy trên GPU, xác định cách Material sẽ được render.



Hình 1.13. Material và Shader

1.3.2.11. Animation

- ❖ Khái niệm: Animation trong Unity là quá trình làm cho các GameObject di chuyển hoặc thay đổi trạng thái theo thời gian.
- ❖ Chức năng: Unity sử dụng các công cụ như Animator và Animation để tạo và quản lý các chuyển động.



Hình 1.14. Animation

1.3.2.12. Physics

- ❖ Khái niệm: Unity có một hệ thống vật lý tích hợp giúp mô phỏng các tương tác vật lý giữa các đối tượng.
- ❖ Chức năng: Sử dụng Rigidbody, Collider, và các thành phần khác để mô phỏng trọng lực, va chạm, và các lực khác.

1.3.2.13. UI (User Interface)

- ❖ Khái niệm: UI trong Unity là hệ thống giao diện người dùng, bao gồm các phần tử như nút, hình ảnh, văn bản, và menu.
- ❖ Chức năng: Unity cung cấp công cụ UI để tạo và quản lý giao diện người dùng cho game.

1.3.2.14. NavMesh và AI

- ❖ NavMesh: Là một công cụ để tạo ra các lưới điều hướng cho nhân vật AI di chuyển.

- ❖ AI (Artificial Intelligence): Unity cung cấp các công cụ và API để tạo ra các hành vi thông minh cho nhân vật trong game.

1.3.2.15. Scripting API

- ❖ Khái niệm: Scripting API của Unity cung cấp các lớp và phương thức để điều khiển và tương tác với các thành phần của Unity từ code.
- ❖ Chức năng: Các lớp phổ biến bao gồm GameObject, Transform, Rigidbody, và nhiều hơn nữa.

1.4. Tổng quan về C# [3]

Hình 1.15. Logo của C#

1.4.1. C# là gì ?

C# (C-Sharp) là một ngôn ngữ lập trình hướng đối tượng được phát triển bởi Microsoft, chủ yếu sử dụng trong các ứng dụng .NET. Đây là một ngôn ngữ mạnh mẽ và linh hoạt, kết hợp các tính năng của C++ và Java, được thiết kế để đơn giản hóa quá trình phát triển phần mềm.

1.4.2. Điểm mạnh của C#

1.4.2.1. Tích hợp chặt chẽ với Unity

- ❖ Sự tương thích tốt: C# là ngôn ngữ chính thức được hỗ trợ bởi Unity, đảm bảo sự tương thích và hiệu suất cao.
- ❖ API phong phú: Unity cung cấp một bộ API phong phú cho C# giúp lập trình viên dễ dàng truy cập và điều khiển các chức năng của Unity như vật lý, đồ họa, âm thanh và hệ thống input.

1.4.2.2. Dễ học và sử dụng

- ❖ Ngôn ngữ hiện đại: C# là một ngôn ngữ lập trình hiện đại, có cú pháp rõ ràng, dễ đọc và dễ hiểu.
- ❖ Hỗ trợ OOP: C# hỗ trợ lập trình hướng đối tượng, giúp tổ chức mã nguồn một cách logic và dễ quản lý.
- ❖ Cộng đồng lớn: Có một cộng đồng lập trình viên lớn sử dụng C#, cung cấp nhiều tài liệu, hướng dẫn và hỗ trợ trực tuyến.

1.4.2.3. Công cụ phát triển mạnh mẽ

- ❖ Visual Studio: Unity tích hợp tốt với Visual Studio, một môi trường phát triển tích hợp (IDE) mạnh mẽ với các tính năng như gỡ lỗi, tự động hoàn thành mã và quản lý dự án.
- ❖ Visual Studio Code: Một lựa chọn khác nhẹ hơn cho các nhà phát triển với các plugin hỗ trợ Unity.

1.4.2.4. Hiệu suất và khả năng mở rộng

- ❖ Hiệu suất tốt: C# cung cấp hiệu suất tốt nhờ vào sự tối ưu hóa của .NET runtime.
- ❖ Thư viện phong phú: Hàng ngàn thư viện và gói mở rộng có sẵn để tăng cường khả năng của C# trong Unity.

1.4.3. Điểm yếu của C#

1.4.3.1. Hiệu suất so với C++

Chậm hơn C++: Mặc dù C# có hiệu suất tốt, nhưng vẫn chậm hơn so với C++ khi xử lý các tác vụ yêu cầu hiệu suất cao như xử lý đồ họa hoặc tính toán vật lý phức tạp.

1.4.3.2. Quản lý bộ nhớ

Garbage Collection: C# sử dụng cơ chế quản lý bộ nhớ tự động (garbage collection), có thể gây ra hiện tượng tạm dừng không mong muốn (GC pauses) khi chương trình giải phóng bộ nhớ.

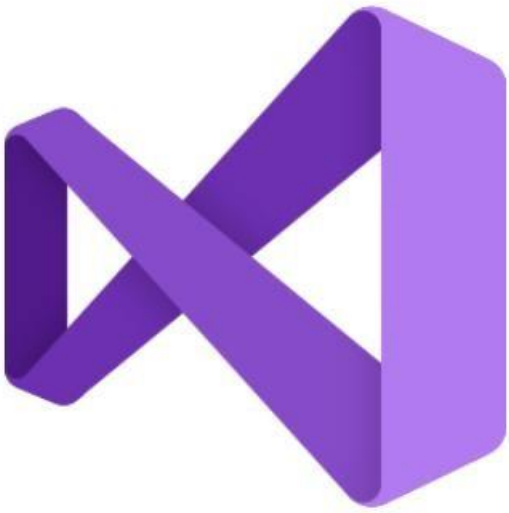
1.4.3.3. Hạn chế nền tảng

Cross-platform limitations: Mặc dù Unity hỗ trợ nhiều nền tảng, nhưng việc sử dụng C# có thể gặp một số hạn chế hoặc cần thêm cấu hình đặc biệt trên các nền tảng không phải Windows.

1.4.3.4. Hạn chế về ngôn ngữ

Tính năng hạn chế: So với C++ hay các ngôn ngữ kịch bản khác, C# có thể thiếu một số tính năng đặc biệt hoặc khả năng tùy chỉnh sâu hơn mà các lập trình viên có thể yêu cầu.

1.5. Tìm hiểu công cụ hỗ Visual Studio [4]



Hình 1.16. Logo của Visual Studio

1.5.1. Giới thiệu

Visual Studio là một môi trường phát triển tích hợp (Integrated Development Environment - IDE) được Microsoft phát triển. Đây là một trong những công cụ phát triển phần mềm phổ biến và mạnh mẽ nhất, hỗ trợ lập trình viên trong việc viết mã, gỡ lỗi, kiểm thử và triển khai các ứng dụng phần mềm. Visual Studio có thể được sử dụng để phát triển các ứng dụng cho nhiều nền tảng khác nhau như Windows, web, di động và đám mây.

1.5.2. Lý do nên dùng Visual Studio

Visual Studio cung cấp nhiều hỗ trợ mạnh mẽ cho việc lập trình game với Unity, giúp các nhà phát triển dễ dàng hơn trong việc viết, gỡ lỗi và quản lý mã nguồn. Dưới đây là một số tính năng nổi bật mà Visual Studio mang lại cho Unity:

- ❖ IntelliSense: Visual Studio cung cấp IntelliSense, một tính năng tự động hoàn thành mã, giúp bạn viết mã nhanh hơn và chính xác hơn. Nó gợi ý các biến, phương thức, lớp, và các thành phần khác dựa trên ngữ cảnh hiện tại.
- ❖ Debugging (Gỡ lỗi): Bạn có thể gỡ lỗi mã của mình trực tiếp trong Visual Studio. Điều này bao gồm việc thiết lập breakpoint (điểm dừng), xem các giá trị biến, và từng bước qua mã để kiểm tra logic của bạn.
- ❖ Unity Integration: Visual Studio có tích hợp tốt với Unity, bao gồm khả năng mở script từ Unity trong Visual Studio, và quay lại Unity sau khi hoàn thành việc chỉnh sửa. Khi bạn cài đặt Visual Studio Tools for Unity, nó cung cấp nhiều tính năng bổ sung để hỗ trợ quy trình làm việc.

- Code Navigation (Điều hướng mã): Bạn có thể dễ dàng điều hướng qua mã nguồn của mình bằng cách sử dụng các tính năng như "Go to Definition" (Đi đến định nghĩa), "Find All References" (Tìm tất cả tham chiếu), và "Peek Definition" (Xem trước định nghĩa).
- Code Snippets (Đoạn mã mẫu): Visual Studio hỗ trợ nhiều đoạn mã mẫu giúp bạn nhanh chóng viết các đoạn mã thường dùng mà không cần phải nhớ cú pháp chi tiết.
- Refactoring (Tái cấu trúc mã): Tính năng tái cấu trúc mã giúp bạn cải thiện chất lượng mã bằng cách thực hiện các thay đổi như đổi tên biến, tạo phương thức, và tách mã mà không làm hỏng chức năng hiện tại.
- Version Control (Quản lý phiên bản): Visual Studio tích hợp với nhiều hệ thống quản lý phiên bản như Git, giúp bạn dễ dàng theo dõi, quản lý và hợp tác trên mã nguồn của mình.
- Performance Profiling (Phân tích hiệu suất): Công cụ phân tích hiệu suất giúp bạn xác định các phần của mã gây ra chậm trễ hoặc hiệu suất kém, từ đó tối ưu hóa trò chơi của mình.
- Package Manager (Trình quản lý gói): Visual Studio có tích hợp với NuGet, cho phép bạn dễ dàng quản lý các thư viện và gói bổ sung mà bạn sử dụng trong dự án của mình.

CHƯƠNG 2: PHÂN TÍCH THIẾT KẾ GAME

2.1. Giới thiệu

- ❖ Tên game: Shadow Adventure
- ❖ Tóm tắt: "Shadow Adventure" là một trò chơi platform 2D, nơi người chơi sẽ nhập vai vào một nhân vật bóng tối, vượt qua các chướng ngại vật, chiến đấu với kẻ thù và khám phá các bí ẩn của thế giới bóng tối. Trò chơi được thiết kế để mang lại những thử thách độc đáo, kết hợp giữa kỹ năng di chuyển, nhảy, leo trèo, tránh né và tấn công, tạo nên một trải nghiệm hấp dẫn và lôi cuốn.
- ❖ Nền tảng: PC, Console
- ❖ Thể loại: Platformer 2d, Action

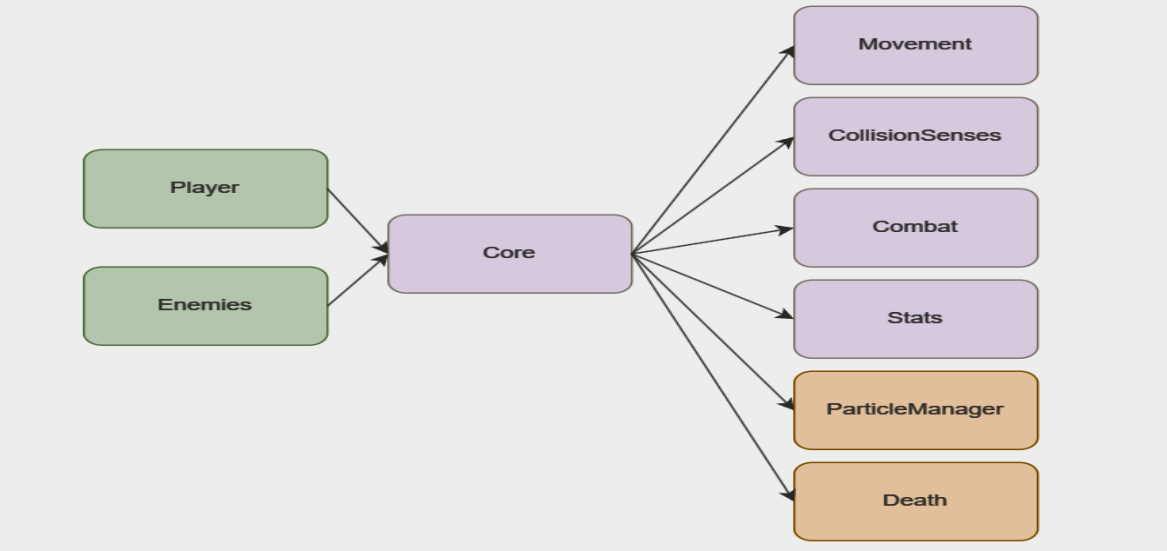
2.2. Tổng quan về Gameplay

Mục tiêu của game: Tiêu diệt quái vật và hoàn thành các màn chơi.

- ❖ Cơ chế đi cảnh:
  - ➢ Nhảy qua chướng ngại vật: Người chơi cần phải nhảy qua các vật cản trên đường.
  - ➢ Leo trèo: Người chơi có thể leo lên các bề mặt nhất định.
  - ➢ Tránh bẫy: Bẫy có thể gây sát thương hoặc làm giảm tốc độ di chuyển của người chơi.
  - ➢ Tấn công quái vật: Quái vật có thể gây sát thương cho người chơi

2.3. Core Game Machine

Description: Sơ đồ mô tả những điểm chung mà cả player và enemy đều phải sử dụng



Hình 2.1. Sơ đồ Core Game

Core Game State Machine của game "Shadow Adventure" bao gồm các trạng thái chính như Idle, Movement, CollisionSenses, Combat, Damage, Stats, ParticleManager, và Death. Mỗi trạng thái đại diện cho một phần quan trọng trong hành vi và tương tác của các nhân vật trong game.

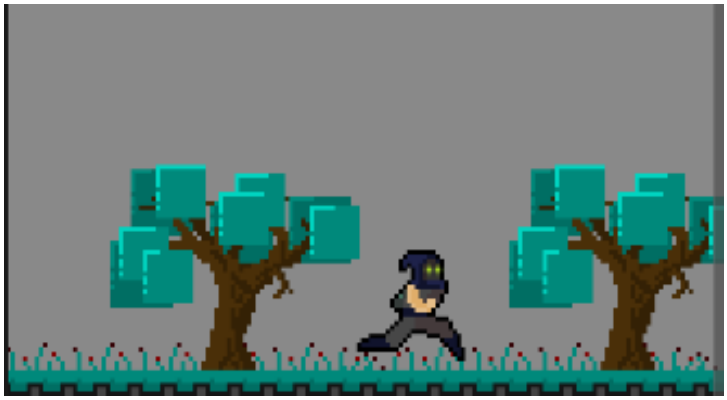
- Movement: Quản lý di chuyển của nhân vật.
- CollisionSenses: Kiểm tra và xử lý va chạm.
- Combat: Quản lý các hành động chiến đấu.
- Damage: Xử lý khi nhân vật nhận sát thương.
- Stats: Cập nhật và quản lý các chỉ số của nhân vật.
- ParticleManager: Quản lý và hiển thị hiệu ứng hạt.
- Death: Trạng thái cuối cùng khi nhân vật chết.

2.4. Thiết kế nhân vật

2.4.1. Character Design

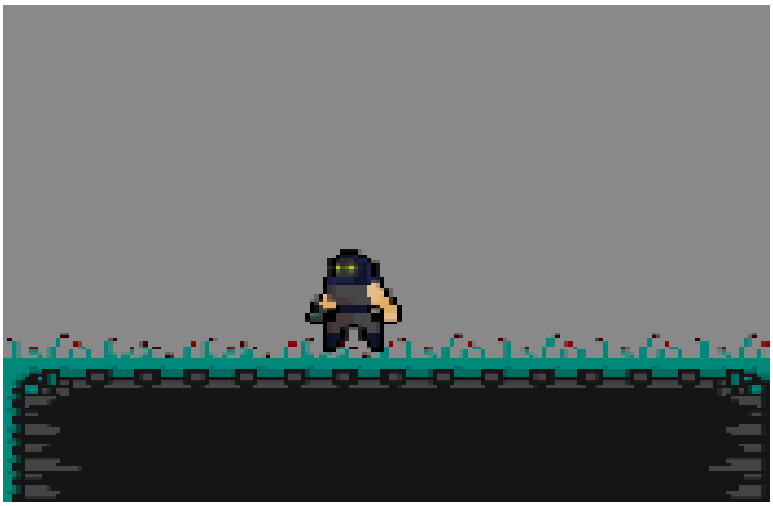
Mô tả: Mô tả chi tiết về các khả năng và hành động của nhân vật người chơi

- Move (Di Chuyển)
  - Mô tả: Nhân vật sẽ di chuyển mượt mà và linh hoạt theo mọi hướng.
  - Chuyển đổi: Chuyển từ trạng thái đứng yên hoặc cúi sang trạng thái di chuyển khi nhận được lệnh di chuyển.



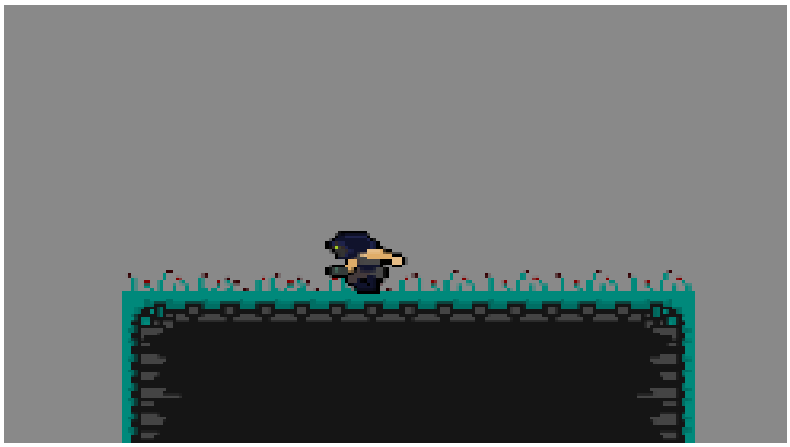
Hình 2.2. Player Movement

- Idle (Đứng Yên)
  - Mô tả: Nhân vật có các hoạt ảnh nhẹ nhàng khi đứng yên.
  - Chuyển đổi: Chuyển từ bất kỳ trạng thái nào khác về trạng thái đứng yên khi không có lệnh di chuyển.



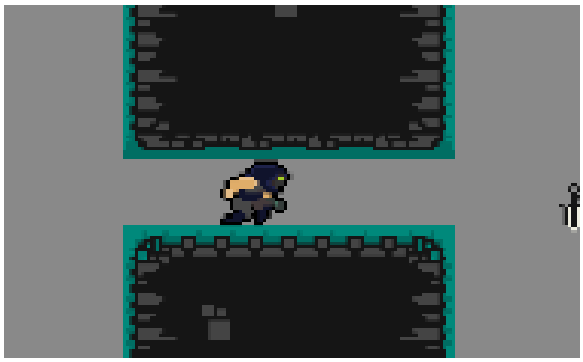
Hình 2.3. Player Idle

- Crouch Move (Di Chuyển Khi Cúi)
  - Mô tả: Nhân vật di chuyển trong tư thế cúi.
  - Chuyển đổi: Chuyển từ trạng thái cúi yên hoặc đứng yên sang trạng thái cúi di chuyển khi nhận được lệnh di chuyển và cúi.



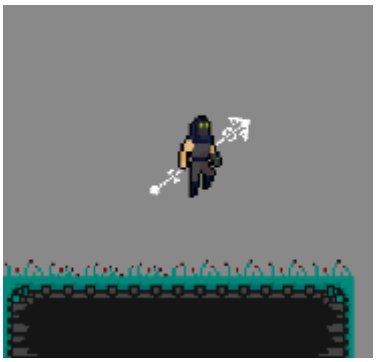
Hình 2.4. Player Crouch Move

- Crouch Idle (Đứng Yên Khi Cúi)
  - Mô tả: Nhân vật duy trì tư thế cúi người khi đứng yên.
  - Chuyển đổi: Chuyển từ trạng thái cúi di chuyển sang trạng thái cúi yên khi ngừng di chuyển nhưng vẫn giữ tư thế cúi.



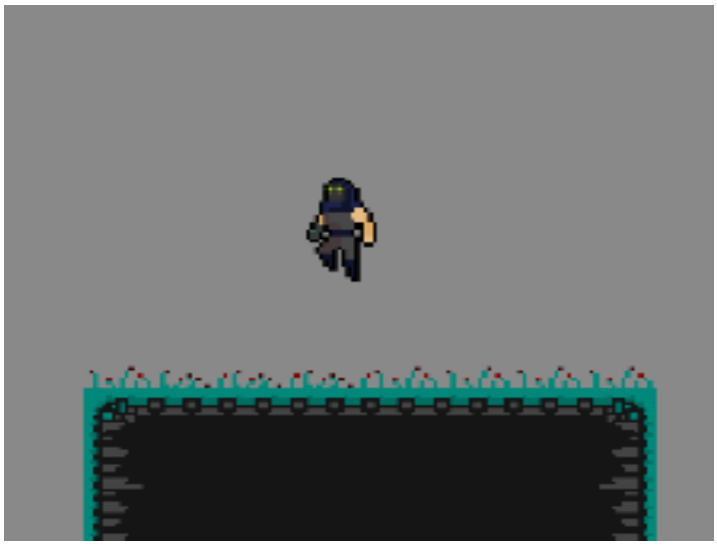
Hình 2.5. Player Crouch Idle

- Dash (Lao Nhanh)
  - Mô tả: Nhân vật thực hiện một cú tăng tốc nhanh chóng.
  - Chuyển đổi: Chuyển từ trạng thái di chuyển hoặc đứng yên sang trạng thái lao nhanh khi nhận được lệnh dash.



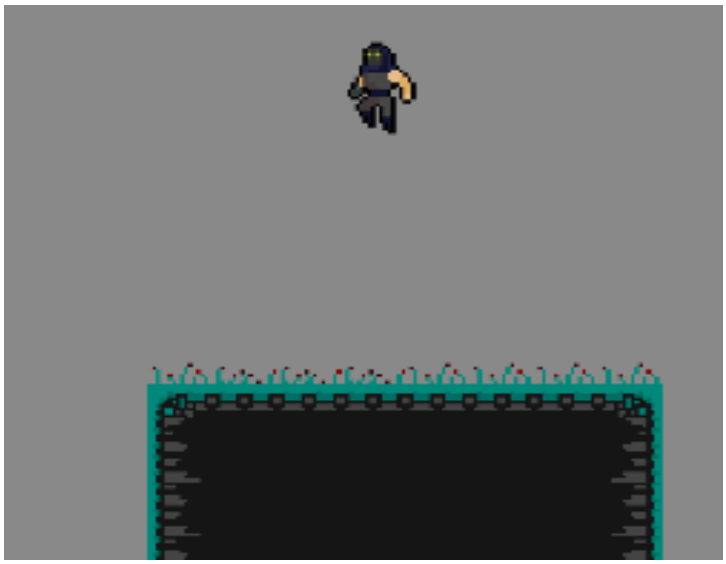
Hình 2.6. Player Dash

- Jump (Nhảy)
  - Mô tả: Nhân vật nhảy khỏi mặt đất.
  - Chuyển đổi: Chuyển từ trạng thái di chuyển, đứng yên, hoặc cúi sang trạng thái nhảy khi nhận được lệnh nhảy.



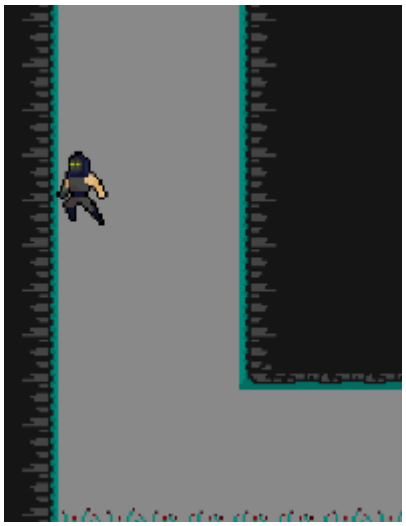
Hình 2.7. Player Jump

- Double Jump (Nhảy Kép)
  - Mô tả: Nhân vật thực hiện thêm một cú nhảy nữa trên không.
  - Chuyển đổi: Chuyển từ trạng thái nhảy sang trạng thái nhảy kép khi nhận được lệnh nhảy lần thứ hai trong không trung.



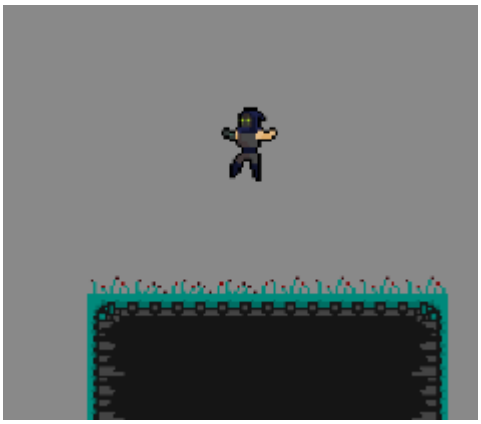
Hình 2.8. Player Double Jump

- Wall Jump (Nhảy Tường)
  - Mô tả: Nhân vật đẩy ra và nhảy theo hướng ngược lại khi tiếp xúc với tường.
  - Chuyển đổi: Chuyển từ trạng thái bám tường hoặc trượt tường sang trạng thái nhảy tường khi nhận được lệnh nhảy khi tiếp xúc với tường.



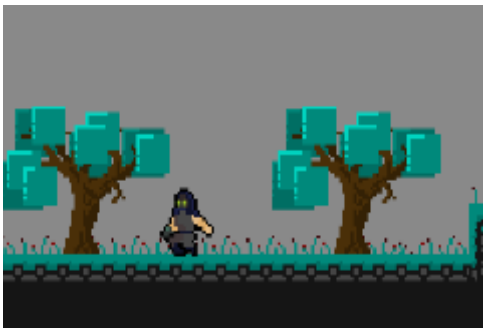
Hình 2.9. Player Wall Jump

- InAir (Trên Không)
  - Mô tả: Nhân vật ở trạng thái trên không sau khi nhảy hoặc bị hất tung.
  - Chuyển đổi: Chuyển từ trạng thái nhảy, nhảy kép, nhảy tường hoặc bị hất tung sang trạng thái trên không khi nhân vật không tiếp xúc với mặt đất.



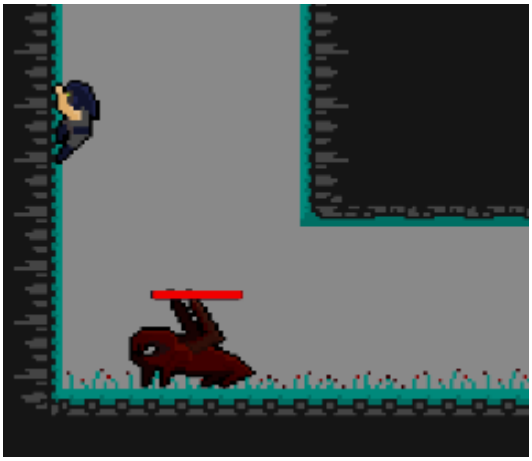
Hình 2.10. Player In Air

- Land (Tiếp Đất)
  - Mô tả: Nhân vật tiếp đất một cách mượt mà.
  - Chuyển đổi: Chuyển từ trạng thái nhảy, nhảy kép, nhảy tường hoặc trạng thái trên không sang trạng thái tiếp đất khi chạm đất.



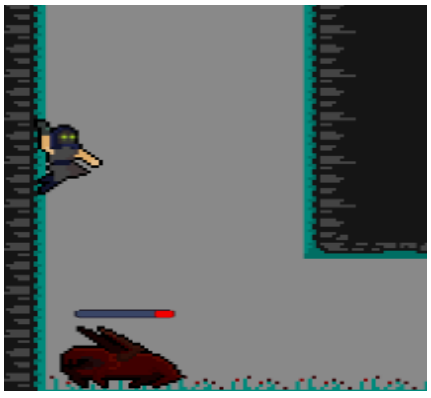
Hình 2.11. Player Land

- Wall Grab (Bám Tường)
  - Mô tả: Nhân vật bám vào các bề mặt thẳng đứng.
  - Chuyển đổi: Chuyển từ trạng thái nhảy hoặc nhảy tường sang trạng thái bám tường khi tiếp xúc với tường.



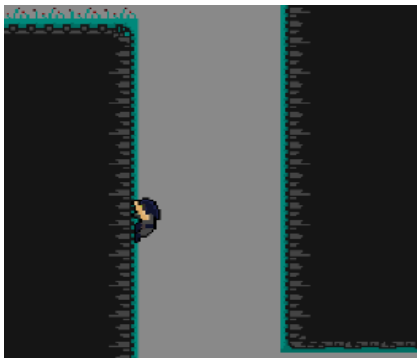
Hình 2.12. Player Wall Grab

- Wall Slide (Trượt Tường)
  - Mô tả: Nhân vật trượt xuống chậm khi bám vào tường.
  - Chuyển đổi: Chuyển từ trạng thái bám tường sang trạng thái trượt tường khi không có lệnh leo tường hoặc nhảy.



Hình 2.13. Player Wall Slide

❖ Wall Climb (Leo Tường)
- ➢ Mô tả: Nhân vật leo lên các bề mặt thẳng đứng.
- ➢ Chuyển đổi: Chuyển từ trạng thái bám tường hoặc trượt tường sang trạng thái leo tường khi nhận được lệnh leo.

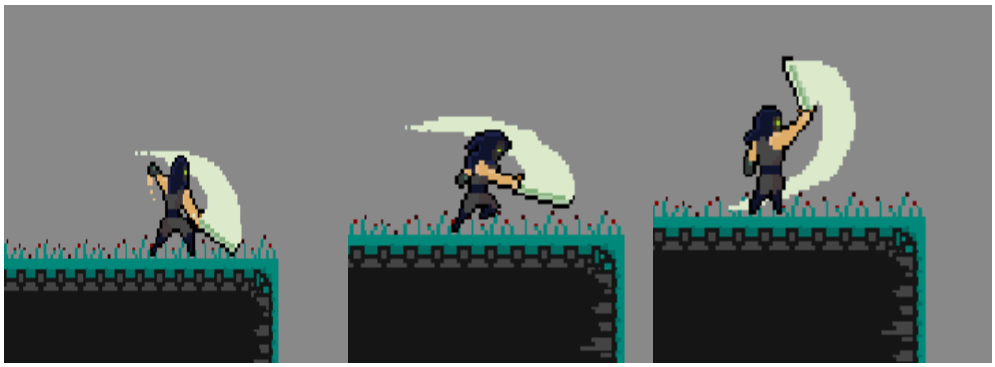


Hình 2.14. Player Wall Climb

❖ Ledge Climb (Leo Mép)
- ➢ Mô tả: Nhân vật tự kéo lên khi chạm đến một mép.
- ➢ Chuyển đổi: Chuyển từ trạng thái bám mép hoặc trượt tường sang trạng thái leo mép khi nhận được lệnh leo.

Hình 2.15. Player Ledge Climb

❖ Attack (Tấn Công)
- ➢ Mô tả: Nhân vật thực hiện các hoạt ảnh tấn công khác nhau.
- ➢ Chuyển đổi: Chuyển từ bất kỳ trạng thái nào sang trạng thái tấn công khi nhận được lệnh tấn công.

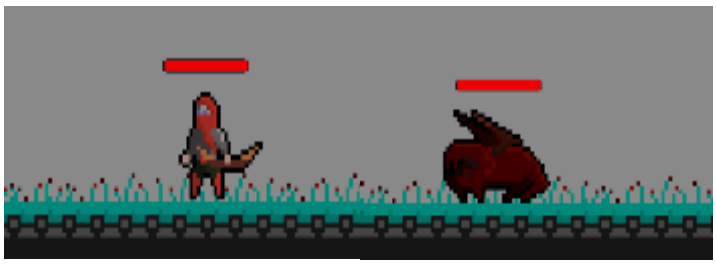


Hình 2.16. Player Attack

2.4.2. Design Enemy

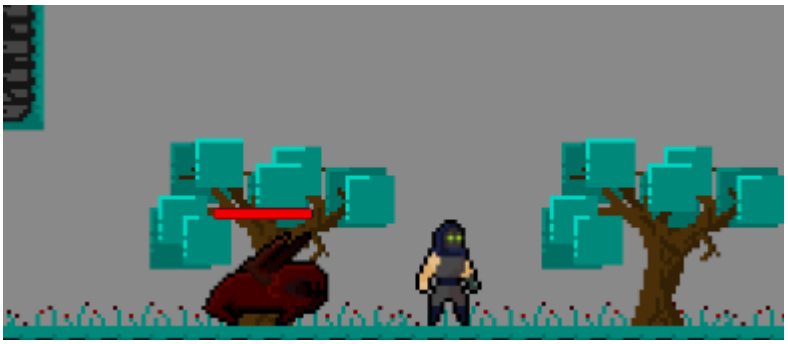
Mô tả: Tổng quan về các loại enemy trong game, cách chúng tương tác với người chơi và vai trò của chúng trong gameplay.

- Idle
  - Mô tả: Trạng thái khi kẻ địch không thực hiện bất kỳ hành động nào, chờ đợi hoặc không có thông tin về người chơi.
  - Chuyển đổi: Kẻ địch sẽ chuyển sang trạng thái Look For Player khi người chơi vào phạm vi kích hoạt tìm kiếm



Hình 2.17. Enemy Idle

- Look For Player
  - Mô tả: Trạng thái khi kẻ địch bắt đầu tìm kiếm người chơi, quan sát xung quanh hoặc di chuyển ngắn để tìm kiếm.
  - Chuyển đổi: Chuyển sang trạng thái PlayerDetected khi phát hiện người chơi.



Hình 2.18. Enemy Look For Player

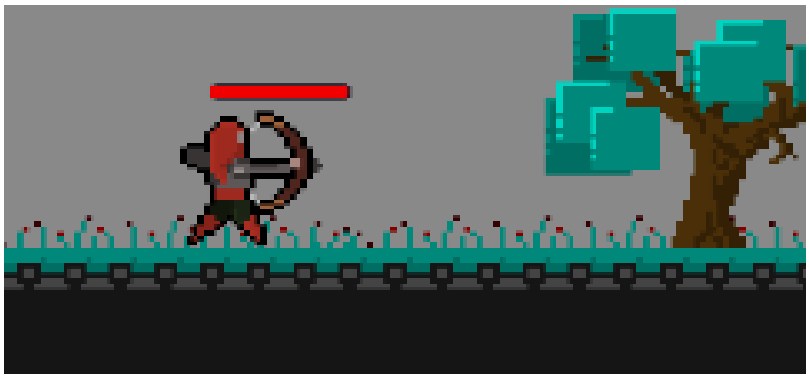
- PlayerDetected
  - Mô tả: Trạng thái khi kẻ địch đã phát hiện ra người chơi và chuẩn bị thực hiện hành động.
  - Chuyển đổi: Chuyển sang trạng thái Move để tiếp cận người chơi hoặc chọn tấn công (Ranged Attack) nếu người chơi ở gần hoặc (Melee Attack) nếu người chơi ở xa.

Hình 2.19. PlayerDetected

❖ Move
- ➢ Mô tả: Kẻ địch di chuyển đến vị trí của người chơi hoặc tới một điểm cụ thể.
- ➢ Chuyển đổi: Khi đến gần người chơi, kẻ địch sẽ chuyển sang trạng thái tấn công (Ranged Attack hoặc Melee Attack).

Hình 2.20. Enemy Move

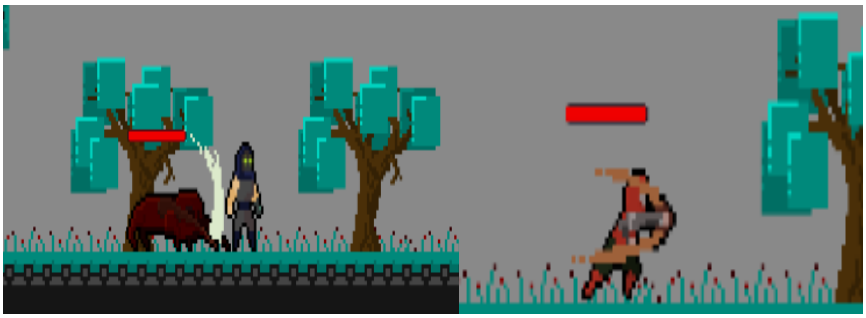
❖ Ranged Attack
- ➢ Mô tả: Kẻ địch tấn công người chơi từ xa bằng vũ khí hoặc kỹ năng.
- ➢ Chuyển đổi: Sau khi thực hiện tấn công, kẻ địch sẽ trở lại trạng thái Move hoặc Dodge để né tránh người chơi nếu người chơi đến quá gần



Hình 2.21. Enemt Ranged Attack

❖ Melee Attack
- ➢ Mô tả: Kẻ địch tấn công người chơi bằng vũ khí cận chiến hoặc đòn tấn công vật lý.

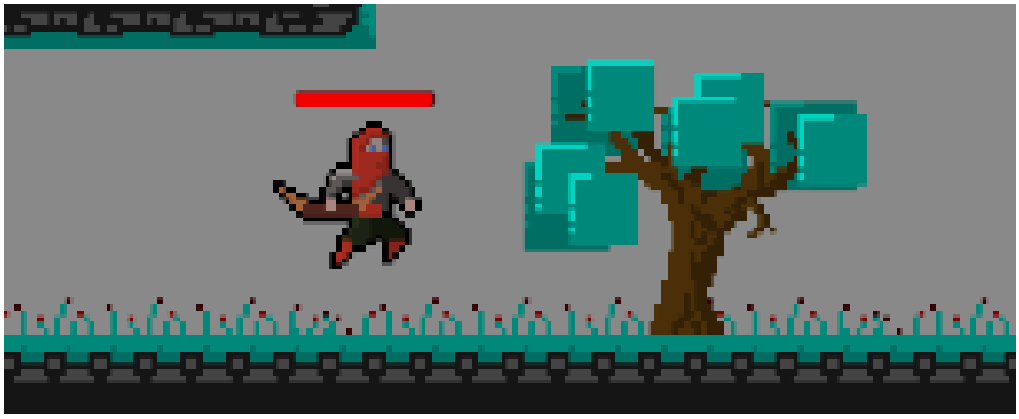
- Chuyển đổi: Sau khi thực hiện tấn công, kẻ địch có thể trở lại trạng thái Move hoặc tiếp tục tấn côngtránh đòn tấn công của người chơi nếu người chơi đến quá gần.



Hình 2.22. Enemy Melee Attack

❖ Dodge

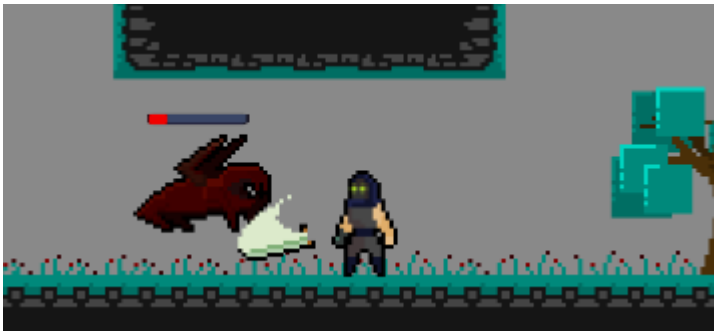
- Mô tả: Kẻ địch tấn công tầm xa né tránh lùi ra sau khi người chơi tới gần
- Chuyển đổi: Sau khi né tránh, kẻ địch có thể trở lại trạng thái Move hoặc tấn công lại người chơi.



Hình 2.24. Enemy Dodge

❖ Damage

- Mô tả: Kẻ địch nhận sát thương từ người chơi.
- Chuyển đổi: Nếu kẻ địch nhận quá nhiều sát thương hoặc bị tấn công mạnh

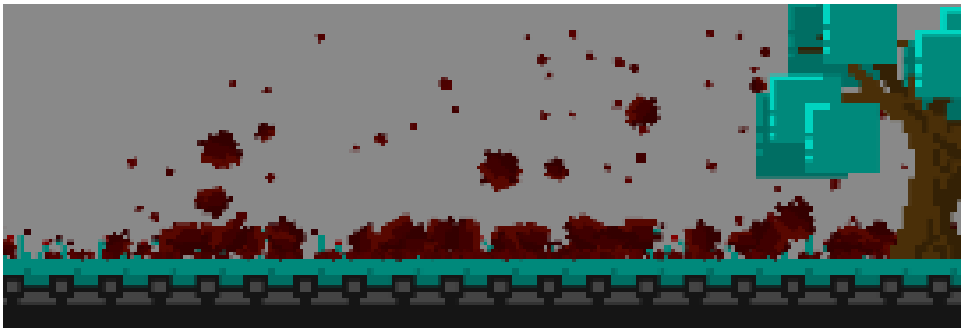


Hình 2.25. Enemy Damage

❖ Death

➢ Mô tả: Kẻ địch bị tiêu diệt và không thể thực hiện bất kỳ hành động nào nữa.

➢ Chuyển đổi: Trạng thái cuối cùng của kẻ địch, không có chuyển đổi tiếp theo.



Hình 2.27. Enemy Death

2.5. Design Patterns [5]

Trong quá trình phát triển game "Shadow Adventure", chúng tôi sử dụng các Design Patterns để tối ưu hóa cấu trúc và hiệu suất của game. Dưới đây là chi tiết về ba mẫu thiết kế chính được áp dụng: Singleton Pattern, State Pattern và Object Pooling.

2.5.1. Singleton Pattern

❖ Mô tả: Singleton Pattern đảm bảo rằng một class chỉ có một instance duy nhất và cung cấp một điểm truy cập toàn cục cho instance này. Pattern này được sử dụng để quản lý các tài nguyên dùng chung như âm thanh, dữ liệu game, hoặc hệ thống quản lý scene.

❖ Triển khai: Trong game "Shadow Adventure", chúng tôi sử dụng Singleton Pattern cho lớp Scene Controller và Sound Manager. Việc sử dụng Singleton Pattern giúp đảm bảo rằng chỉ có một instance của Scene Controller và Sound Manager tồn tại trong toàn bộ thời gian chạy của game. Điều này giúp quản lý trạng thái của game, xử lý sự kiện và giữ các thuộc tính quan trọng của game.

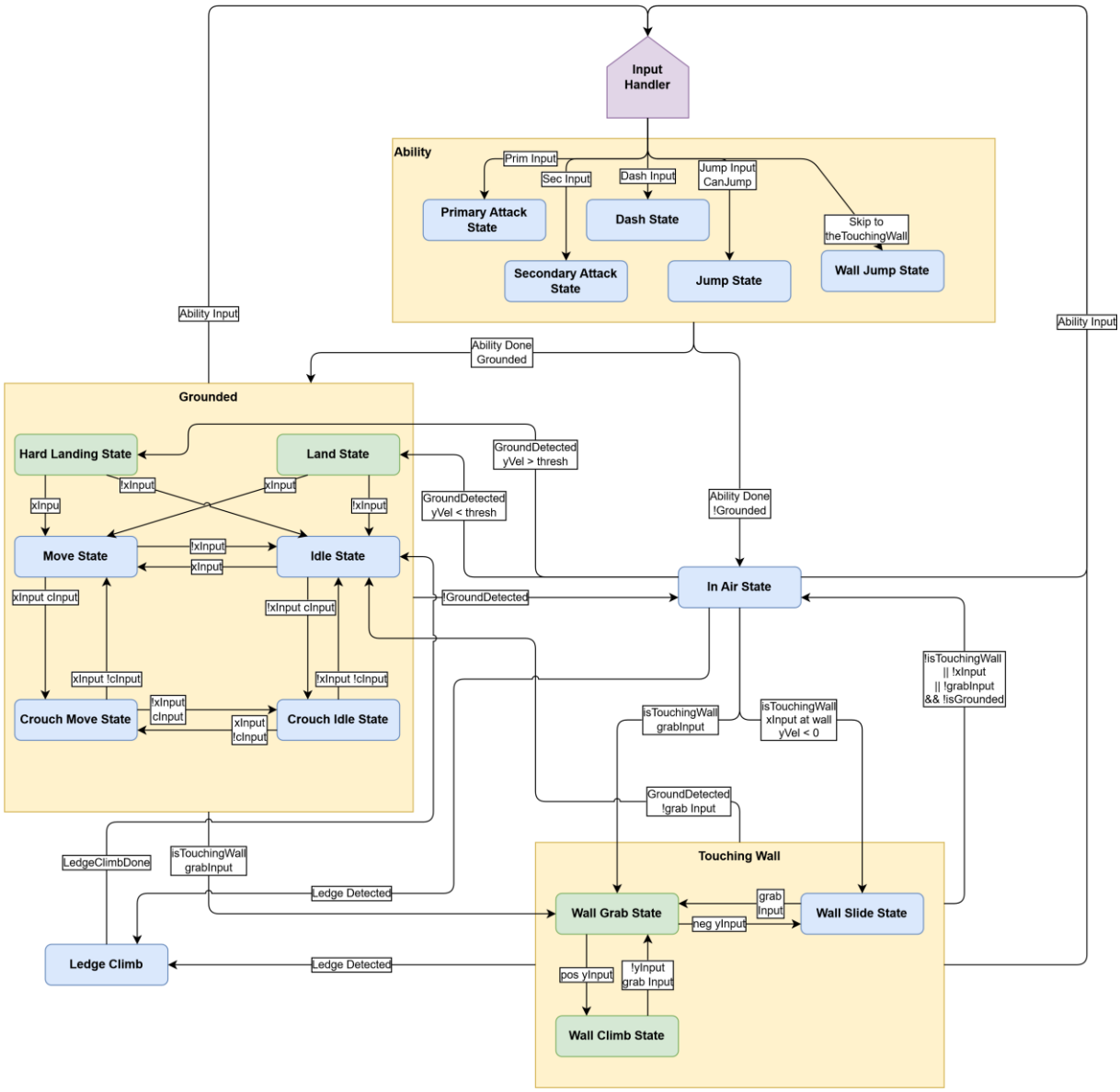
2.5.2. State Pattern

State Pattern cho phép một đối tượng thay đổi hành vi của nó khi trạng thái nội tại thay đổi. Pattern này được sử dụng để quản lý các trạng thái khác nhau của một nhân vật trong game, như đi, chạy, nhảy, và tấn công.

Sơ đồ State Machine

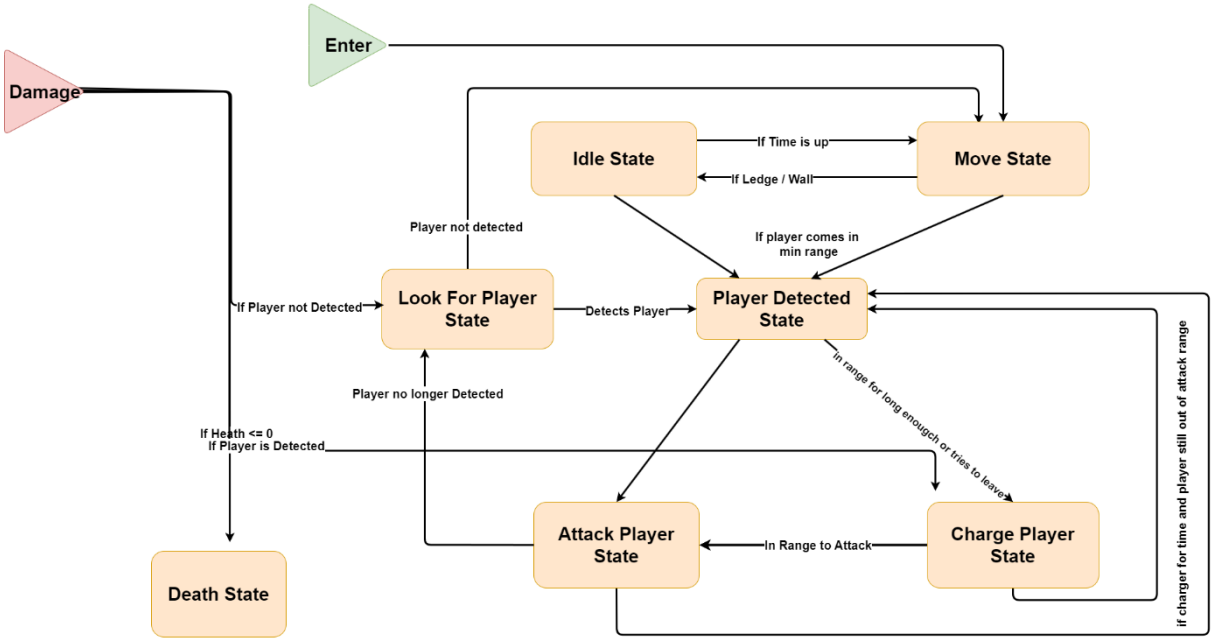
2.5.2.1. Player State Machine

Mô tả: Sơ đồ trạng thái mô tả các trạng thái khác nhau của người chơi và cách chuyển đổi giữa chúng.



Hình 2.28. Player State Machine

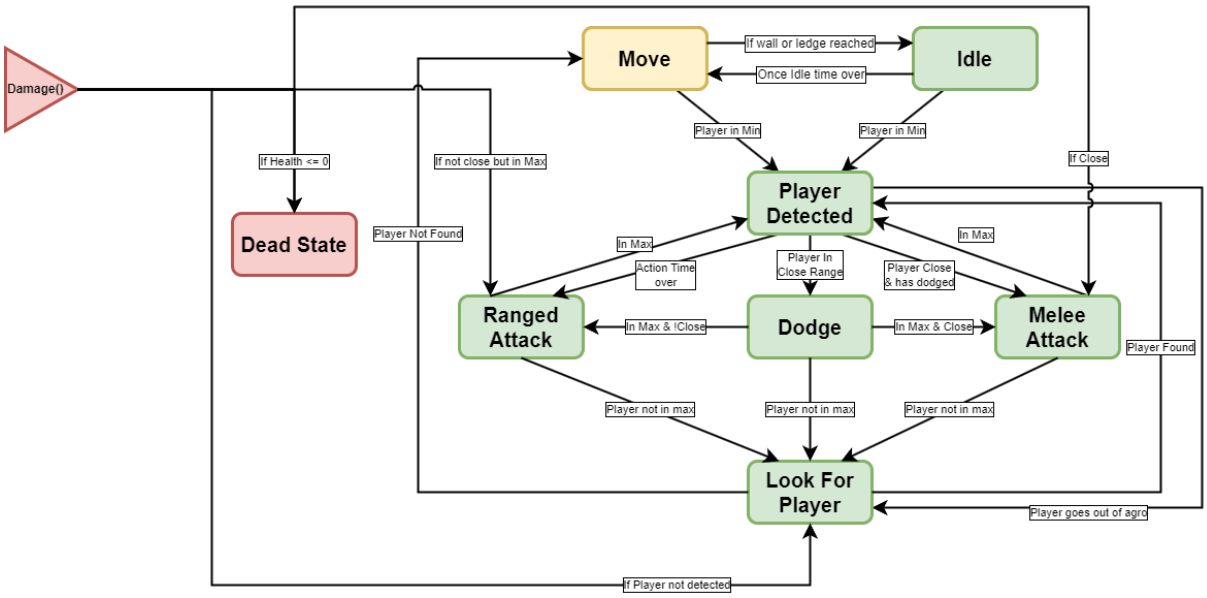
2.5.2.2. Sơ đồ State Machine chung cho tất cả các Enemy



Hình 2.29. Enemy State Machine

2.5.2.3. Ranged And Melee Enemy State Machine

Mô tả: Sơ đồ trạng thái mô tả các trạng thái khác nhau của Ranged Enemy Attack và Melee Enemy Attack và cách chuyển đổi giữa chúng.



Hình 2.30. Enemy Ranged And Melee State Machine

2.5.3. Object Pooling

- Mô tả: Object Pooling là một kỹ thuật tối ưu hóa nhằm giảm chi phí khởi tạo đối tượng bằng cách tái sử dụng các đối tượng đã được tạo thay vì tạo mới và hủy liên tục. Pattern này rất hữu ích cho các đối tượng được sử dụng thường xuyên và có vòng đời ngắn trong game.
- Triển khai: Trong game "Shadow Adventure", tôi sử dụng Object Pooling cho việc tạo dư ảnh khi Dash của người chơi. Điều này giúp giảm tải cho bộ nhớ và tăng hiệu suất của game.

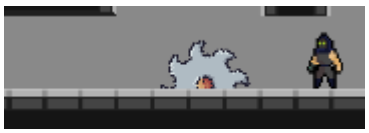
2.6. Hệ thống bẫy (Trap System)

Hệ thống bẫy là một phần quan trọng trong thiết kế game, giúp tăng thêm tính thử thách và kịch tính cho người chơi. Dưới đây là các loại bẫy chính trong game:

- Bẫy chông (Spike Trap)
  - Mô tả: Bẫy chông sẽ được đặt dưới đất, khi người chơi bước vào, các chông sẽ gây sát thương.
  - Cơ chế hoạt động: Kích hoạt khi người chơi bước vào vùng kích hoạt.
  - Hậu quả: Gây sát thương cho người chơi và có thể khiến người chơi bị mất máu trong một khoảng thời gian ngắn.

Hình 2.31. Spike Trap

- Bẫy cưa (Saw Trap)
  - Mô tả: Bẫy cưa là một lưỡi cưa lớn di chuyển qua lại, có thể gây sát thương nặng cho người chơi.
  - Cơ chế hoạt động: Di chuyển liên tục theo một đường nhất
  - Hậu quả: Gây sát thương nặng cho người chơi ngay lập tức nếu bị trúng.



Hình 2.32. Saw Trap

- Bẫy chông rơi (Spike Fall Trap)
  - Mô tả: Bẫy chông rơi là các chông từ trên cao rơi xuống khi người chơi kích hoạt.
  - Cơ chế hoạt động: Kích hoạt khi người chơi bước vào vùng kích hoạt.

- Hậu quả: Gây sát thương nặng hoặc tiêu diệt người chơi ngay lập tức nếu bị trúng.

Hình 2.33. Spike Fall Trap

- Bẫy Spiked Head
  - Mô tả: Bẫy Spiked Head là một cỗ đầu lớn có chứa nhiều chiếc chông sắc nhọn trên bề mặt.
  - Cơ chế hoạt động: Bẫy này sẽ lao vào người chơi khi người chơi đi vào khoảng tấn công
  - Hậu quả: Khi người chơi tiếp cận quá gần gây sát thương cho người chơi. Đối với người chơi, nếu không tránh kịp, có thể dẫn đến việc mất mạng hoặc bị thương nặng.

Hình 2.34. Spiked Head Trap

2.7. Âm Thanh (Sound Design)

Âm thanh đóng vai trò quan trọng trong việc tạo ra trải nghiệm chơi game phong phú và sống động. Dưới đây là cách tôi thiết kế và triển khai hệ thống âm thanh trong trò chơi bằng cách sử dụng các packet âm thanh có sẵn.

2.7.1. Âm thanh nền (Background Music)

- Nhạc nền các màn chơi là như nhau

2.7.2. Âm thanh hiệu ứng (Sound Effects)

- Hành động của nhân vật chính:
  - Bước chân: Sử dụng âm thanh bước chân từ packet âm thanh. Các tệp âm thanh này sẽ được đặt vào các AudioSource và kích hoạt khi nhân vật di chuyển.

- Nhảy: Sử dụng âm thanh ngắn và nhẹ khi nhân vật nhảy lên và âm thanh mạnh hơn khi nhân vật tiếp đất từ packet âm thanh. Các âm thanh này sẽ được kích hoạt bằng cách sử dụng các sự kiện trong mã lệnh Unity.
- Tấn công: Sử dụng âm thanh của vũ khí khi vung và âm thanh khi trúng mục tiêu từ packet âm thanh. Các âm thanh này sẽ được tích hợp vào các hành động tấn công của nhân vật chính và kẻ thù.

2.8. Công cụ và Công Nghệ Sử Dụng

Trong quá trình phát triển dự án, chúng tôi sử dụng các công cụ và công nghệ sau:

- Unity Engine: Sử dụng Unity Engine làm nền tảng phát triển chính cho game 2D.
- C#: Sử dụng ngôn ngữ lập trình C# để lập trình các tính năng và hành vi của game trong Unity.
- Tiled Map Editor: Sử dụng Tiled Map Editor để thiết kế và xây dựng các màn chơi và môi trường game.
- Unity Animation: Sử dụng Spine hoặc tính năng Animation của Unity để tạo và quản lý các animation cho nhân vật và đối tượng trong game.
- Unity Input System: Sử dụng Unity Input System để quản lý việc xử lý input từ bàn phím, chuột và các thiết bị đầu vào khác.
- TextMeshPro: Sử dụng TextMeshPro để tạo và định dạng văn bản trong game với hiệu suất cao hơn so với Text component mặc định của Unity.
- Unity Particle System: Sử dụng Unity Particle System để tạo và điều chỉnh các hiệu ứng hạt (particle effects) như lửa, nước, khói, ánh sáng, và các hiệu ứng khác.
- Visual Studio Sử dụng Visual Studio làm môi trường lập trình cho việc phát triển mã nguồn game bằng ngôn ngữ C#

CHƯƠNG 3. XÂY DỰNG CHƯƠNG TRÌNH [6]

3.1. Môi Trường Cài Đặt

3.1.1. Phần cứng:

- Bộ vi xử lý: AMD Ryzen 5
- RAM: 8GB
- Card đồ họa: AMD RADEON Graphics

3.1.2. Phần mềm:

- Hệ điều hành: Windows 11
- Unity Engine: Phiên bản 2021.3.17
- Visual Studio: Phiên bản 2022

3.2. Giao diện chương trình

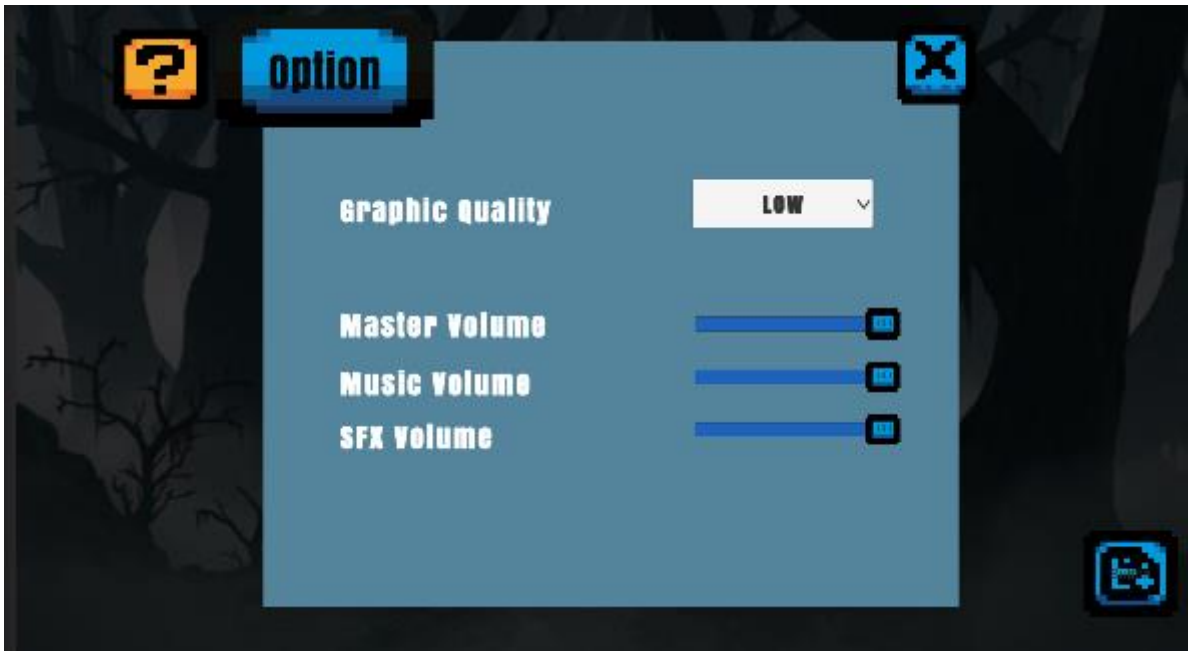
3.2.1. Màn hình chính

Mô tả: Màn hình chính bao gồm logo của trò chơi, các nút bắt đầu, tùy chọn, và thoát.

- Thành phần:
  - Nút "Play" để chọn màn chơi.
  - Nút "Options" để điều chỉnh cài đặt âm thanh, đồ họa.
  - Nút "Exit" để thoát khỏi trò chơi.
  - Nút “Question” để xem hướng dẫn chơi

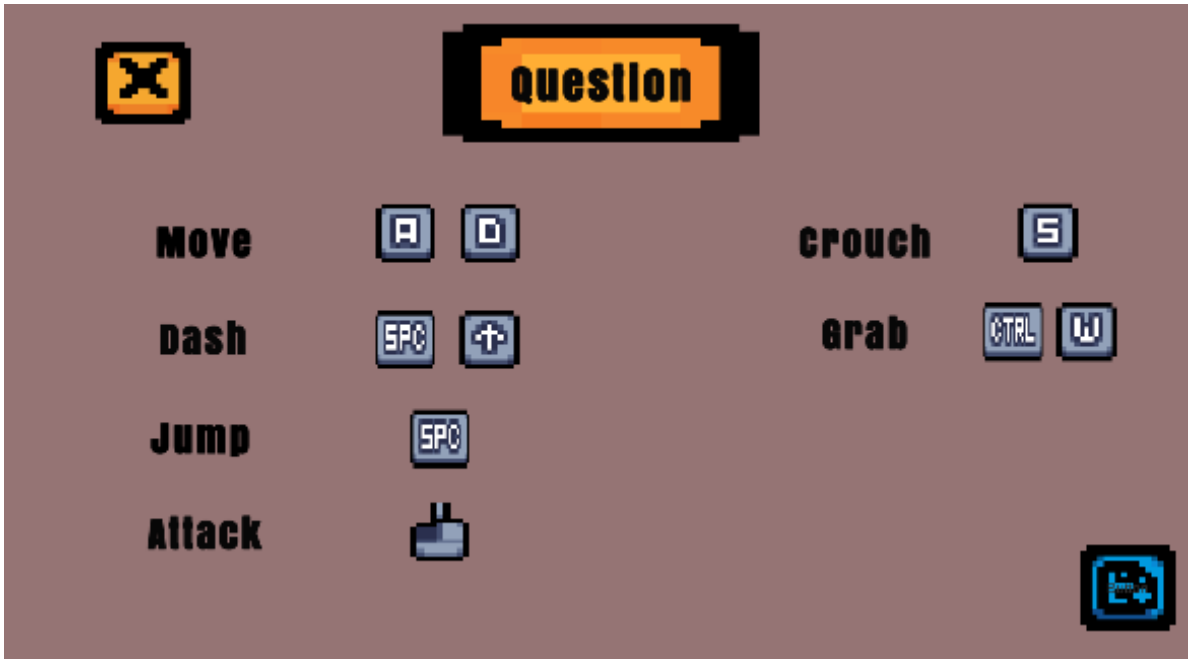
Hình 3.1. Giao diện Menu chính

Trong giao diện Option người chơi có thể điều chỉnh đồ họa và điều chỉnh âm thanh cho game



Hình 3.2. Giao diện Option

Giao diện Question hướng dẫn các nút cần thiết để điều khiển Player



Hình 3.3. Giao diện Question

Giao diện Select Level cho phép người chơi chọn Level chơi, bắt đầu game Level mặc định được mở khóa là 1. Khi người chơi hoàn thành Level hiện tại thì Level tiếp theo sẽ được mở khóa

Hình 3.4. Giao diện Select Level

3.2.2. Khi vào game

Giao diện khi vào game gồm có:

- ❖ Thanh Hp: Hiển thị lượng Hp hiện tại của người chơi
- ❖ Nút Pause: Hiển thị giao diện Pause Game

Hình 3.5. Giao diện khi vào Game

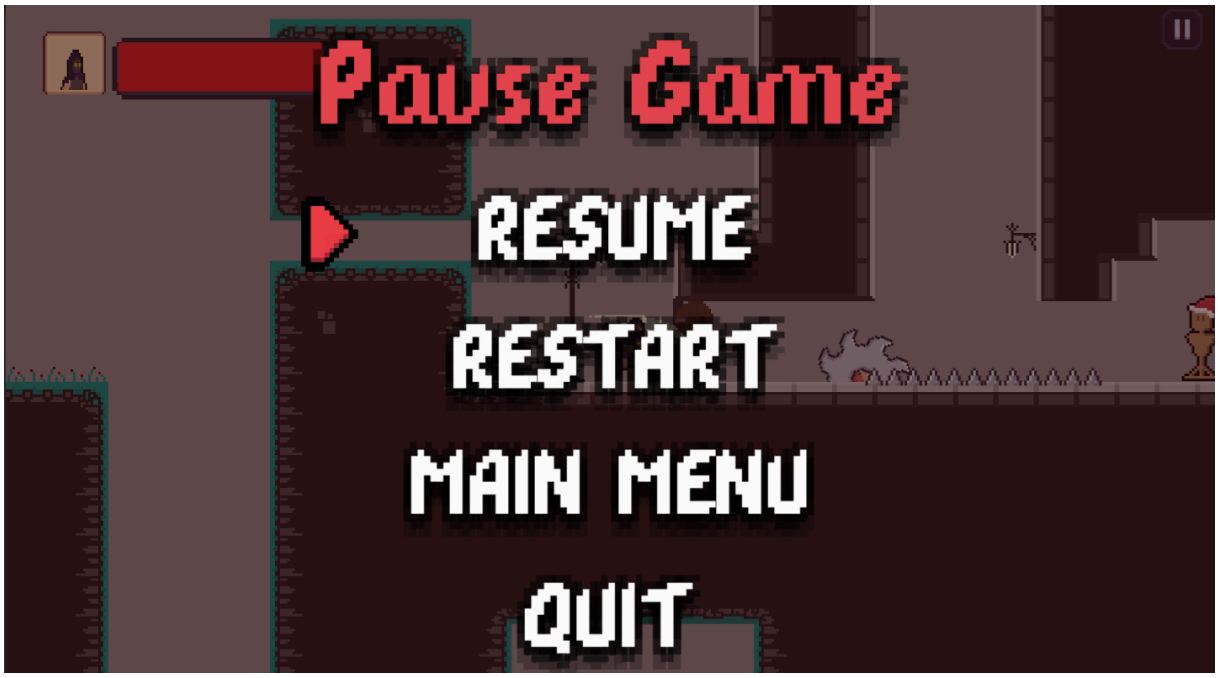
Khi Hp của người chơi giảm về 0, giao diện thông báo người chơi đã thua được hiện lên

- Trong giao diện khi thua gồm có:
  - Restart: Reset lại màn chơi , giúp người chơi chơi lại màn hiện tại
  - Main Menu: Chuyển về giao diện menu
  - Quit: Thoát Game

Hình 3.6. Giao diện khi thua

Khi người chơi ấn nút Pause trên màn hình, giao diện Pause Game sẽ được hiện ra

- Trong giao diện Pause Game gồm có:
  - Resume: tiếp tục game trước khi ấn nút Pause
  - Restart: Reset lại màn chơi, chơi lại màn chơi từ đầu
  - MainMenu: Chuyển sang giao diện Menu
  - Quit: Thoát Game



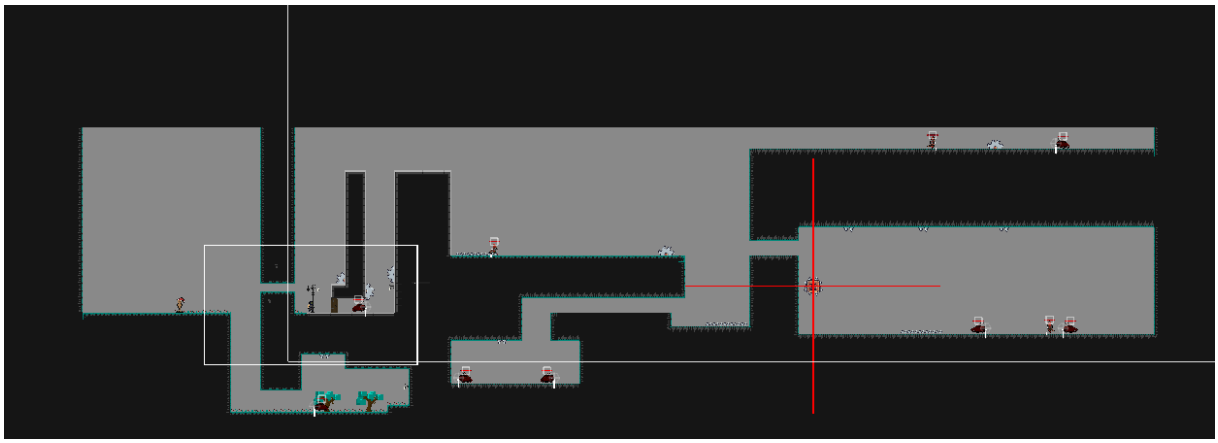
Hình 3.7. Giao diện khi Pause Game

3.2.3. Các Level

Hiện tại game có 3 level với độ khó tăng dần. Độ khó dựa vào số lượng quái , số lượng bẫy và địa hình của map. Người chơi cần vượt qua các chướng ngại vật, quái vật đi đến cuối của bản đồ để hoàn thành màn chơi

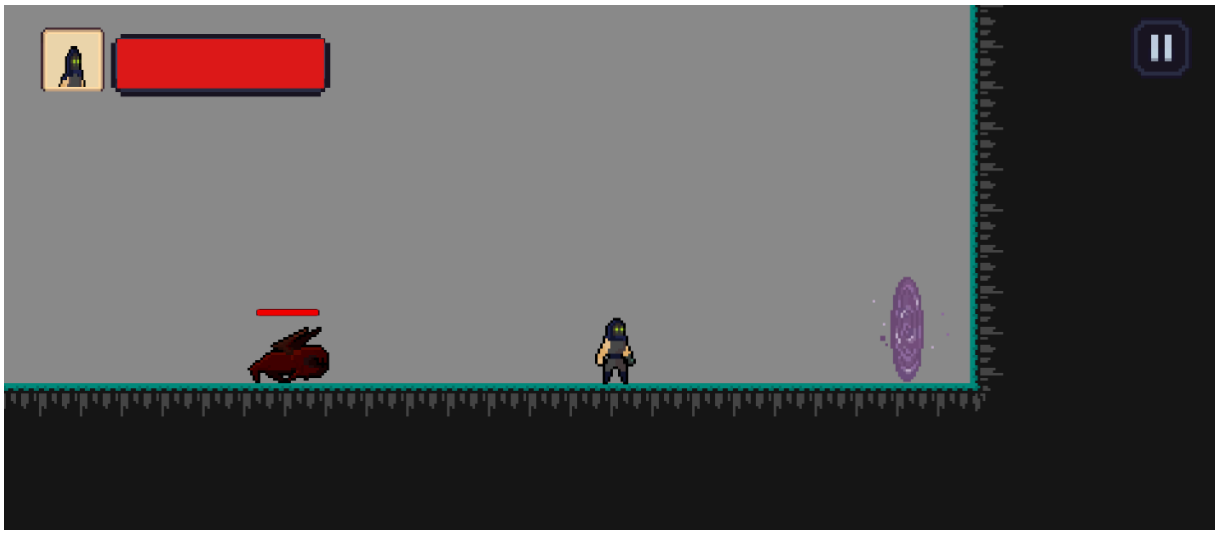
- ❖ Sử dụng Tilemap trong Unity để tạo nền và các yếu tố môi trường.
- ❖ Tạo các Tile Palette từ các hình ảnh đã import và sử dụng chúng để xây dựng level trong cửa sổ Scene.

Level 1:



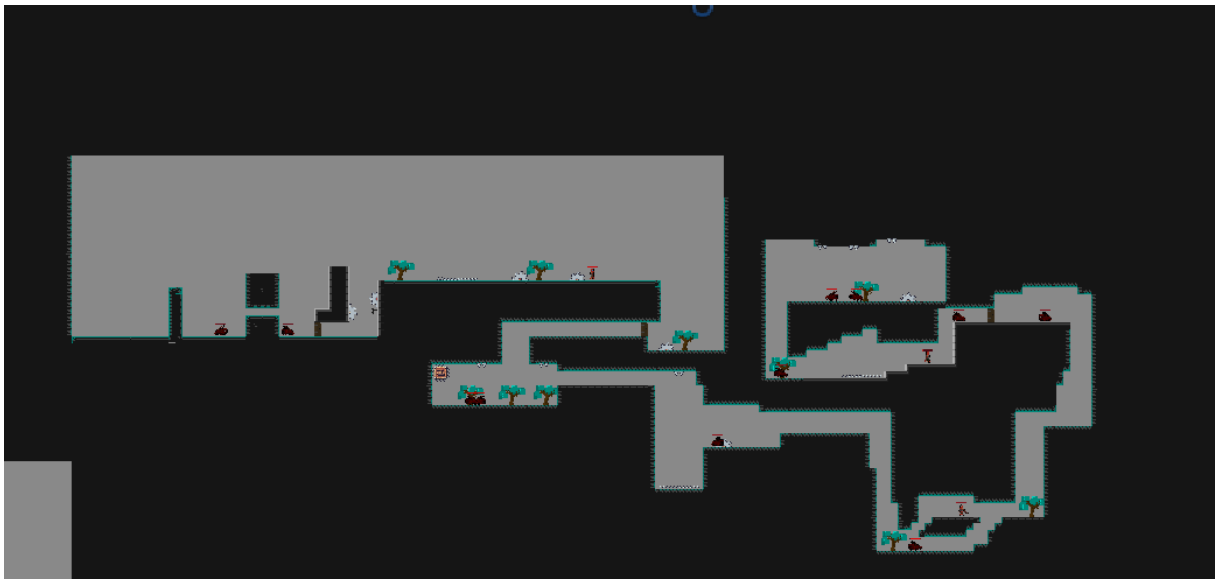
Hình 3.8. Sơ đồ map Level 1

Khi người chơi đi đến cuối map, cổng dịch chuyển sẽ tự động mở ra và đưa người chơi sang màn tiếp theo



Hình 3.9. Hình ảnh khi đến cuối mỗi màn

Level 2:



Hình 3.10. Sơ đồ map level 2

Level 3:

Hình 3.11. Sơ đồ map level 3

TỔNG KẾT VÀ HƯỚNG PHÁT TRIỂN

1. Tổng kết

1.1. Kết quả đạt được

Trong quá trình thực hiện dự án phát triển game 2D Shadow Adventure trên nền tảng Unity, tôi đã đạt được những kết quả sau:

- ❖ Hoàn thành thiết kế màn chơi: Đã xây dựng và thiết kế các màn chơi đa dạng, mang lại trải nghiệm phong phú cho người chơi.
- ❖ Phát triển nhân vật và kẻ địch: Tạo và lập trình các hành vi phức tạp cho nhân vật chính cũng như các kẻ địch khác nhau, bao gồm cả kẻ địch
- ❖ Xây dựng được máy trình phát hoạt ảnh cho nhân vật và ememy giúp dễ dàng thêm mới các nhân vật và enemy khác
- ❖ Tích hợp các công nghệ hiện đại: Sử dụng các công cụ và công nghệ như Unity Input System, TextMeshPro, và Unity Particle System để nâng cao chất lượng đồ họa và trải nghiệm người chơi.
- ❖ Trải nghiệm điều khiển nhân vật mượt mà

1.2. Nhược điểm, hạn chế.

- ❖ Chưa xây dựng được game hoàn chỉnh như mong muốn
- ❖ Giao diện chưa thực sự bắt mắt
- ❖ Game thiếu các yếu tố giữ chân người chơi như hệ thống nhiệm vụ

2. Hướng Phát Triển

2.1. Mở rộng số lượng nhân vật

- ❖ Thêm nhân vật mới:
  - ➢ Phát triển các nhân vật mới với các kỹ năng và đặc điểm riêng biệt.
  - ➢ Thiết kế cốt truyện và nhiệm vụ riêng cho từng nhân vật để tạo sự phong phú và hấp dẫn.
- ❖ Hệ thống chuyển đổi nhân vật:
  - ➢ Cho phép người chơi chuyển đổi giữa các nhân vật trong các màn chơi để tận dụng tối đa các kỹ năng của từng nhân vật.

2.2. Xây dựng túi đồ cho nhân vật

- ❖ Hệ thống túi đồ (inventory):
  - ➢ Phát triển hệ thống túi đồ cho phép người chơi lưu trữ và quản lý các vật phẩm thu thập được trong game.
  - ➢ Thiết kế giao diện túi đồ trực quan và dễ sử dụng.

- Tích hợp vật phẩm:
  - Thêm các loại vật phẩm như vũ khí, trang bị, vật liệu và vật phẩm hồi máu.

2.5. Xây dựng hệ thống đa vũ khí

- Hệ thống vũ khí đa dạng:
  - Thiết kế nhiều loại vũ khí khác nhau, từ kiếm, cung tên đến các vũ khí phép thuật.
  - Mỗi vũ khí có đặc điểm và hiệu ứng riêng, tạo nên chiến thuật đa dạng cho người chơi.

TÀI LIỆU THAM KHẢO

[1] [Online]. Available: https://mindovermetal.org/game-adv-la-gi-lich-su-phat-trien-cua-the-loai-game-adv/.

[2] [Online]. Available: https://viblo.asia/p/gioi-thieu-ve-unity-engine-game-engine-pho-bien-nhat-hien-nay-V3m5WBj8lO7.

[3] [Online]. Available: https://vi.wikipedia.org/wiki/C_Sharp_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh).

[4] [Online]. Available: https://vi.wikipedia.org/wiki/Visual_Studio.

[5] [Online]. Available: https://www.pancakellc.com/p/game-programming-patterns-p2.

[6] [Online]. Available: https://unity.com/learn.