|  |  |
| --- | --- |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO** | **BỘ NÔNG NGHIỆP VÀ PTNT** |

**TRƯỜNG ĐẠI HỌC THỦY LỢI**



NGUYỄN KHƯƠNG DUY

**XÂY DỰNG GAME 2D “DUNGEON AND WITCHER”**

ĐỒ ÁN TỐT NGHIỆP

HÀ NỘI, 2024

|  |  |
| --- | --- |
| **BỘ GIÁO DỤC VÀ ĐÀO TẠO** | **BỘ NÔNG NGHIỆP VÀ PTNT** |

**TRƯỜNG ĐẠI HỌC THỦY LỢI**

NGUYỄN KHƯƠNG DUY

**XÂY DỰNG GAME 2D “DUNGEON AND WITCHER”**

Ngành: Công nghệ thông tin

Mã số:

NGƯỜI HƯỚNG DẪN: **1.** **ThS. Trương Xuân Nam**

**2. TS. Lê Nguyễn Tuấn Thành**

HÀ NỘI, 2024

**GÁY BÌA ĐỒ ÁN TỐT NGHIỆP**

**NGUYỄN KHƯƠNG DUY ĐỒ ÁN TỐT NGHIỆP HÀ NỘI, NĂM 2024**

|  |  |
| --- | --- |
| Logo-WRU | CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM  **Độc lập - Tự do - Hạnh phúc**  ----------★----------  **NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP** |

Họ tên sinh viên: Nguyễn Khương Duy Hệ đào tạo : Đại học chính quy

Lớp: 61TH1 Mã sinh viên: 1951060661

Ngành: Công nghệ thông tin

Khoa: Công nghệ thông tin

1. TÊN ĐỀ TÀI:

XÂY DỰNG GAME 2D “DUNGEON AND WITCHER”

1. CÁC TÀI LIỆU CƠ BẢN:

* Quy trình giao đồ án tốt nghiệp khoa Công nghệ thông tin Đại học Thủy Lợi (2023)
* C#: <https://learn.microsoft.com/en-us/dotnet/csharp/>
* Unity: <https://docs.unity3d.com/Manual/UsingTheEditor.html>
* Assets: <https://itch.io/game-assets/free/tag-2d>
* Trợ giúp : <https://stackoverflow.com/>
* Lập trình Unity căn bản: Nhiều nguồn.

1. NỘI DUNG CÁC PHẦN THUYẾT MINH VÀ TÍNH TOÁN:

|  |  |
| --- | --- |
| **Nội dung cần thuyết minh** | **Tỷ lệ** |
| 1. Chương I. Tổng quan về ngành game |  |
| 2. Chương II. Tổng quan về Unity |  |
| 3. Chương III. Mô tả thiết kế game |  |
| 4. Chương IV: Phân tích và thiết kế hệ thống |  |
| 5. Chương V: Xây dựng và demo |  |
| 6. Chương VI: Kết luận và hướng phát triển |  |

1. GIÁO VIÊN HƯỚNG DẪN TỪNG PHẦN

|  |  |
| --- | --- |
| **Các phần** | **GVHD** |
| 1. Chương I. Tổng quan về ngành game |  |
| 2. Chương II. Tổng quan về Unity |  |
| 3. Chương III. Mô tả thiết kế game |  |
| 4. Chương IV: Phân tích và thiết kế hệ thống |  |
| 5. Chương V: Xây dựng và demo |  |
| 6. Chương VI: Kết luận |  |

5. NGÀY GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

Ngày ..........tháng ......... năm 202..

|  |  |
| --- | --- |
| **Trưởng Bộ môn**  *(Ký và ghi rõ Họ tên)* | **Giáo viên hướng dẫn chính**  *(Ký và ghi rõ Họ tên)* |

Nhiệm vụ Đồ án tốt nghiệp đã được Hội đồng thi tốt nghiệp của Khoa thông qua.

Ngày. . . . .tháng. . . . .năm 202...

**Chủ tịch Hội đồng**

*(Ký và ghi rõ Họ tên)*

Sinh viên đã hoàn thành và nộp bản Đồ án tốt nghiệp cho Hội đồng thi ngày ... tháng ... năm 202...

**Sinh viên làm Đồ án tốt nghiệp**

*(Ký và ghi rõ Họ tên)*

Nguyễn Khương Duy

TRƯỜNG ĐẠI HỌC THUỶ LỢI 

**KHOA CÔNG NGHỆ THÔNG TIN**

BẢN TÓM TẮT ĐỀ CƯƠNG ĐỒ ÁN TỐT NGHIỆP

TÊN ĐỀ TÀI: **Xây dựng game 2D “Dungeons And Witcher”**

*Sinh viên thực hiện*: Nguyễn Khương Duy

*Lớp*: 61TH1

*Mã sinh viên:* 1951060661

*Số điện thoại:* 0355321023

*Email:* duytlu2001@gmail.com

*Giáo viên hướng dẫn*: ThS. Trương Xuân Nam

TS. Lê Nguyễn Tuấn Thành

*Thời gian thực hiện: 14 tuần (từ ngày: 16/10/2023 đến ngày: 21/01/2024)*

**TÓM TẮT ĐỀ TÀI**

"Dungeons And Witcher" là một trò chơi dựa trên cơ chế của “The Binding of Isaac” và “Noita”, tập trung vào cuộc phiêu lưu hấp dẫn của một phù thủy trong một thế giới dưới lòng đất đầy nguy hiểm. Trong game, người chơi sẽ vào vai một phù thủy và bắt đầu hành trình khám phá hầm ngục bí ẩn.

Nhiệm vụ chính của người chơi là đối mặt với những thách thức khó khăn, đánh quái vật và tiêu diệt các boss mạnh mẽ để thu thập những cây đũa phép có sức mạnh đặc biệt. Mỗi cây đũa phép sẽ mang theo một loạt các kỹ năng phép thuật độc đáo, giúp người chơi thích nghi với các tình huống khác nhau. Giữa các tầng của hàm ngục sẽ có những căn phòng trao đổi giúp người chơi nâng cấp những cây đũa phép của mình và tạo ra những phép mới lạ.

Trong suốt cuộc hành trình, người chơi sẽ phải tìm kiếm và sử dụng thông minh các vật phẩm, thuốc phép, và trang bị để cải thiện sức mạnh của họ và tăng khả năng sinh tồn. Các hầm ngục sẽ luôn thay đổi, đảm bảo sự đa dạng và tính chất ngẫu nhiên trong mỗi lần chơi. "Dungeons And Witcher" hứa hẹn mang đến cho người chơi trải nghiệm thú vị với đồ họa độc đáo, lối chơi đa dạng và một thế giới đầy bí ẩn và kỳ quái để khám phá.

**CÁC MỤC TIÊU CHÍNH**

Chức năng :

1. Tìm hiểu về ngôn ngữ C#, cách sử dụng unity trong làm game.
2. Xây dựng nhân vật, tìm hiểu về cơ chế điều khiển nhân vật.
3. Xây dựng map, Tìm hiểu về các thuật toán tạo ra hầm ngục ngẫu nhiên.
4. Xây dựng quái, tìm hiểu cơ chế của quái.
5. Xây dựng kho vật phẩm, tỉ lệ rơi vật phẩm, cấp bậc vật phẩm.
6. Xây dựng các loại phép thuật, cơ chế kết hợp phép thuật.
7. Xây dựng môi trường, cách mà các đối tượng tương tác với môi trường.
8. Thiết kế UI.

Công nghệ sử dụng :

● C#

● Unity, Visual Studio

**KẾT QUẢ DỰ KIẾN**

Sản phẩm:

* Trò chơi hoàn thiện, đáp ứng được các mục tiêu đề ra.

Kiến thức:

* Thành thạo C#.
* Kiến thức về công cụ unity.
* Kiếm thức về lập trình game, quy trình để tạo nên game 2D.
* Kiến thức về phân tích và thiết kế.

Về kĩ năng:

* Sử dụng tốt công cụ unity.
* Khả năng thiết kế giao diện UI, thiết kế nhân vật.
* Nâng cao khả năng code, tư duy logic.
* Kĩ năng làm việc, tìm kiếm thông tin.

**TIẾN ĐỘ THỰC HIỆN**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Thời gian** | **Nội dung công việc** | **Kết quả dự kiến đạt được** |
| 1 | 16/10/2023 –05/11/2023 | Đọc tài liệu, phân tích thiết kế xây dựng lược đồ quan hệ | Phân tích thiết kế ra mô hình đầy đủ nhất của dự án |
| 2 | 06/11/2023 –10/01/2024 | Lập trình xây dựng trò chơi | hoàn chỉnh trò chơi với các chức năng |
| 3 | 11/01/2024 –16/01/2024 | Kiểm thử và fix lỗi hệ thống | đảm bảo hệ thống hoạt động bình thường |
| 4 | 17/01/2024 –20/01/2024 | Hoàn thiện và chỉnh sửa báo cáo đồ án | Báo cáo đồ án hoàn chỉnh |

# LỜI CAM ĐOAN

Tác giả xin cam đoan đây là Đồ án tốt nghiệp của bản thân tác giả. Các kết quả trong Đồ án tốt nghiệp này là trung thực, và không sao chép từ bất kỳ một nguồn nào và dưới bất kỳ hình thức nào.Việc tham khảo các nguồn tài liệu (nếu có) đã được thực hiện trích dẫn và ghi nguồn tài liệu tham khảo đúng quy định.

**Tác giả ĐATN**

**Nguyễn Khương Duy**

# LỜI CẢM ƠN

Trong suốt thời gian học tập rèn luyện tại trường Đại học Thủy Lợi, em xin gửi lời cảm ơn đến tập thể các thầy cô trong trường nói chung và các thầy cô khoa Công Nghệ Thông Tin nói riêng. Nhờ sự dìu dắt tận tình của các thầy cô mà em mới có đủ khả năng, kiến thức để làm Đồ án tốt nghiệp lần này cũng như là kinh nghiệm để bước ra ngoài làm việc.

Em xin chân thành cảm ơn thầy giáo, Thạc sĩ Trương Xuân Nam đã hướng dẫn em hoàn thành tốt đồ án tốt nghiệp này trong thời gian vừa qua. Trong quá trình làm đồ án tốt nghiệp đã đưa cho em những lời khuyên, khuyến khích, góp ý tận tình để em hoàn thành đồ án của mình một cách tốt nhất.

Em xin chân thành cảm ơn thầy giáo, Tiến sĩ Lê nguyễn Tuấn Thành đã cùng thầy Nam đồng hướng dẫn em hoàn thành đồ án tốt nghiệp này trong thời gian vừa qua. Thầy luôn nhiệt tình và quan tâm đến sinh viên, mỗi lần nhận được góp ý của thầy đều giúp em nhìn nhận lại lỗi sai mà em đã mắc phải trong quá trình phân tích và thiết kế trò chơi.

Em xin bày tỏ lòng biết ơn đến tất cả các bạn và thầy cô trường Đại học Thủy lợi đã tạo điều kiện hỗ trợ, giúp đỡ em trong suốt quá trình học tập và nghiên cứu đề tài này. Trong suốt thời gian từ khi bắt đầu học tập tại trường đến nay, em đã nhận được rất nhiều sự quan tâm, giúp đỡ của thầy cô và bạn bè.

Do kiến thức còn hạn chế và thời gian hoàn thành không nhiều nên có thể còn thiếu sót, em rất mong nhận được sự góp ý của thầy cô và bạn bè để đồ án này được hoàn thiện hơn.

Em xin chân thành cảm ơn!

# MỤC LỤC

[LỜI CAM ĐOAN i](#_Toc154783255)

[LỜI CẢM ƠN ii](#_Toc154783256)

[MỤC LỤC iii](#_Toc154783257)

[DANH MỤC HÌNH ẢNH vi](#_Toc154783258)

[DANH MỤC BẢNG BIỂU vi](#_Toc154783259)

[DANH MỤC TỪ VIẾT TẮT VÀ GIẢI THÍCH CÁC THUẬT NGỮ vi](#_Toc154783260)

[CHƯƠNG 1: TỔNG QUAN VỀ NGÀNH GAME 1](#_Toc154783261)

[1.1. Lịch sử và hướng phát triển 1](#_Toc154783263)

[1.2. Các công cụ làm game 2](#_Toc154783264)

[1.3. Các ngôn ngữ làm game 5](#_Toc154783265)

[CHƯƠNG 2: TỔNG QUAN VỀ CÔNG CỤ LÀM GAME UNITY 7](#_Toc154783267)

[2.1. Unity là gì? 7](#_Toc154783269)

[2.2. Ưu điểm của Unity 7](#_Toc154783270)

[2.3. Các thành phần trong Unity 8](#_Toc154783271)

[2.3.1. Cửa sổ Sence 8](#_Toc154783272)

[2.3.2. Cửa sổ Game view 9](#_Toc154783273)

[2.3.3. Cửa sổ Hierarchy 10](#_Toc154783274)

[2.3.4. Cửa sổ Inspector 10](#_Toc154783275)

[2.3.5. Cửa sổ Project 11](#_Toc154783276)

[2.3.6. Cửa sổ Toolbar 12](#_Toc154783277)

[2.3.7. Cửa sổ Console 13](#_Toc154783278)

[2.3.8. Cửa sổ Animator 14](#_Toc154783279)

[2.4. Các khái niệm trong Unity 14](#_Toc154783280)

[2.4.1. GameObject 14](#_Toc154783281)

[2.4.2. Compoment 14](#_Toc154783282)

[2.4.3. Sprite 15](#_Toc154783283)

[2.4.4. Animation 15](#_Toc154783284)

[2.4.5. Prefab 15](#_Toc154783285)

[2.4.6. Script 15](#_Toc154783286)

[2.4.7. Assets 15](#_Toc154783287)

[2.4.8. Camera 15](#_Toc154783288)

[2.4.9. Sound 15](#_Toc154783289)

[2.4.10. Transform 16](#_Toc154783290)

[2.4.11. Material và Shader 16](#_Toc154783291)

[2.4.12. Unity Flow 16](#_Toc154783292)

[2.4.13. Finite State Machine: 17](#_Toc154783293)

[2.5. Các lớp Unity API 18](#_Toc154783294)

[2.5.1. MonoBehaviour Class 18](#_Toc154783295)

[2.5.2. ScriptableObject Class 18](#_Toc154783296)

[2.5.3. GameObject Class 18](#_Toc154783297)

[2.5.4. Component Class 19](#_Toc154783298)

[2.5.5. Transform Class 19](#_Toc154783299)

[2.5.6. Animator Class 19](#_Toc154783300)

[2.5.7. SpriteRenderer Class 19](#_Toc154783301)

[2.5.8. Collider Class 19](#_Toc154783302)

[2.5.9. Camera Class 19](#_Toc154783303)

[2.5.10. AudioSource Class 19](#_Toc154783304)

[2.5.11. Physics Class 19](#_Toc154783305)

[2.5.12. SceneManager Class 19](#_Toc154783306)

[2.5.13. Input Class 20](#_Toc154783307)

[2.5.14. Time Class 20](#_Toc154783308)

[2.5.15. Rigidbody Class 20](#_Toc154783309)

[Chương III. Mô tả thiết kế game 20](#_Toc154783310)

[3.1. Ý tưởng 20](#_Toc154783312)

[3.2. Thiết kế game 21](#_Toc154783313)

[3.2.1. Thiết kế 21](#_Toc154783314)

[Chương IV: Phân tích và thiết kế hệ thống 21](#_Toc154783315)

[4.1. Xác định các tác nhân hệ thống 21](#_Toc154783316)

[4.2. Xây dựng biểu đồ Usecase 22](#_Toc154783317)

[4.2.1. Mô hình Usecase tổng quát 22](#_Toc154783318)

[4.3. Đặc tả và phân tích chức năng của hệ thống 23](#_Toc154783319)

[4.3.1. Usercase Chuyển đổi trạng thái 23](#_Toc154783320)

[4.3.2. Usercase Tương tác với kẻ địch 26](#_Toc154783321)

[4.3.3. Usercase Lựa chọn chúc phúc 29](#_Toc154783322)

[4.3.4. Usercase Quản lý vật phẩm 32](#_Toc154783323)

[4.3.5. Usercase Quản lý thống kê 35](#_Toc154783324)

[4.3.6. Usercase Lưu trữ 38](#_Toc154783325)

[4.4. Xác định lớp 41](#_Toc154783326)

[Chương V: Xây dựng và demo 42](#_Toc154783327)

[Chương VI: Kết luận và hướng phát triển 42](#_Toc154783328)

# DANH MỤC HÌNH ẢNH

# DANH MỤC BẢNG BIỂU

# DANH MỤC TỪ VIẾT TẮT VÀ GIẢI THÍCH CÁC THUẬT NGỮ

|  |  |  |
| --- | --- | --- |
| **Từ viết tắt** | **Từ đầy đủ** | **Nghĩa tiếng việt** |
|  |  |  |

# CHƯƠNG 1: TỔNG QUAN VỀ NGÀNH GAME



## Lịch sử và hướng phát triển

Ngành công nghiệp game đã trải qua một hành trình dài và phát triển đáng kể trong suốt hơn nửa thế kỷ qua. Sau đây là một số sự kiện quan trọng trong lịch sử ngành game:

* Năm 1947: Thomas T. Goldsmith Jr. và Estle Ray Mann phát minh ra một máy chơi game đầu tiên gọi là “Cathode Ray Tube Amusement Device”, có thể coi là bước đầu tiên trong lịch sử game.
* Năm 1958: Physicist William Higinbotham tạo ra trò chơi tennis đầu tiên trên máy tính tại Brookhaven National Laboratory.
* Năm 1972: Nolan Bushnell và Ted Dabney thành lập công ty Atari và phát hành trò chơi đầu tiên của họ là Pong, một trong những trò chơi arcade đầu tiên.
* Năm 1980: Nintendo phát hành game đầu tiên của mình, Donkey Kong, và đưa ra thị trường máy console đầu tiên của họ, Nintendo Entertainment System (NES).
* Năm 1990: Sony phát hành PlayStation, máy console đầu tiên của họ và một trong những máy console phổ biến nhất của thập niên 1990.
* Năm 1993: id Software phát hành game Doom, đưa thể loại game bắn súng trở nên phổ biến hơn bao giờ hết.
* Năm 1994: The Entertainment Software Rating Board (ESRB) được thành lập để đánh giá nội dung của các trò chơi video.
* Năm 2001: Microsoft phát hành máy console Xbox đầu tiên của họ.
* Năm 2004: Blizzard Entertainment phát hành game trực tuyến đa nền tảng đầu tiên của họ, World of Warcraft, và trở thành một trong những game trực tuyến phổ biến nhất trong lịch sử.
* Năm 2007: Apple phát hành iPhone, mở ra một thị trường mới cho game di động.
* Năm 2017: Nintendo phát hành máy console di động Nintendo Switch, được coi là một trong những thiết bị chơi game phổ biến nhất của thế hệ mới.

Từ đó đến nay, ngành công nghiệp game đã trở thành một ngành công nghiệp lớn, có ảnh hưởng đến nhiều lĩnh vực khác nhau và tạo ra hàng tỉ đô la doanh thu hàng năm.

Dự đoán xu hướng ngành game đến năm 2030 là khá khó khăn, tuy nhiên có một số xu hướng chung có thể xảy ra trong tương lai gần:

* Chơi game trực tuyến sẽ tiếp tục tăng: Với sự phát triển của mạng internet và công nghệ kết nối, chơi game trực tuyến sẽ tiếp tục tăng. Các trò chơi đa người chơi trực tuyến và các game có tính tương tác cao sẽ trở nên phổ biến hơn.
* Game di động sẽ tiếp tục phát triển: Đã có xu hướng tăng trưởng trong việc chơi game trên điện thoại di động, và điều này sẽ tiếp tục phát triển trong tương lai. Các game di động sẽ ngày càng được phát triển để đáp ứng nhu cầu của người chơi trên toàn cầu.
* Các trò chơi sử dụng công nghệ VR và AR sẽ trở nên phổ biến: Công nghệ thực tế ảo (VR) và thực tế tăng cường (AR) đang được sử dụng trong các trò chơi và có thể trở thành xu hướng chính trong ngành game trong tương lai.
* Sự phát triển của game trên đám mây: Với sự phát triển của công nghệ đám mây, các game có thể được truy cập trên nhiều thiết bị và người chơi có thể chơi các trò chơi yêu thích của mình mọi lúc, mọi nơi.
* Game chạy trên blockchain và sử dụng NFT sẽ phát triển: Game chạy trên blockchain và sử dụng NFT đang trở nên phổ biến hơn, và điều này có thể tiếp tục phát triển trong tương lai khi người chơi muốn sở hữu các tài sản trong game một cách độc đáo và minh bạch.
* Sự xuất hiện của game AI: Sự phát triển của trí tuệ nhân tạo (AI) sẽ mở ra cơ hội để phát triển các game có tính tự động cao hơn, hỗ trợ cho việc tạo ra các nhân vật và tạo nên các nội dung trong game.

## Các công cụ làm game

Các công cụ làm game là các phần mềm hỗ trợ cho nhà phát triển game dễ dàng hơn trong quá trình thiết kế và phát triển game. Một số lợi ích có thể dễ nhận thấy khi sử dụng các công cụ làm game như: dễ học, dễ sử dụng, tiếp kiệm thời gian phát triển, dễ dàng nâng cấp và bảo trì.

Một số công cụ phát triển game phổ biến hiện nay:

* Unity: Được phát triển bởi Unity Technologies, Unity là một ứng dụng lập trình game được các nhà lập trình game dùng để phát các game và mô phỏng cho PC, Mobile và bảng điều khiển. Unity được các nhà lập trình game ưu ái sử dụng bởi vì tính năng đa nền tảng và thư viện tài nguyên khổng lồ. Bên cạnh đó, phần mềm tạo game này còn sở hữu kho tàng vô vàn video hướng dẫn cơ bản cho người dùng mới bắt đầu sử dụng cũng như nâng cao cho người đã có kinh nghiệm.
* Hỗ trợ định dạng 2D và 3D, các nhà phát triển game có thể thiết kế game dựa trên cả 3 ngôn ngữ lập trình là C#, Boo và UnityScript. Tuy nhiên, hiện nay C# vẫn là ngôn ngữ được ưu ái sử dụng nhiều nhất cho phần mềm làm game này.

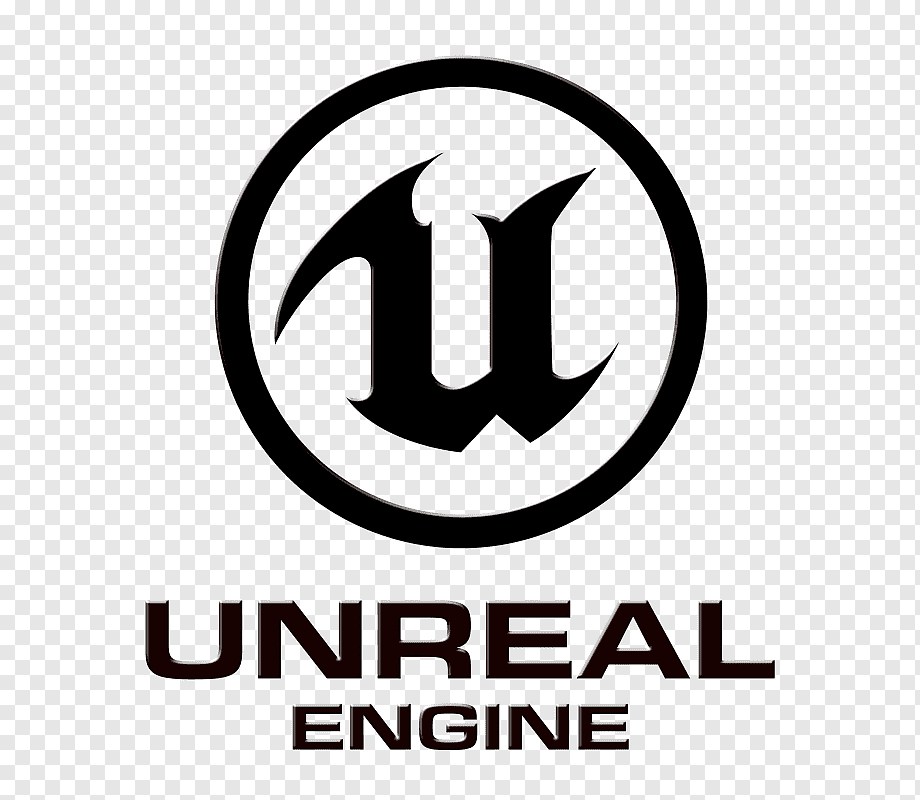
A black background with a white object in the middle

Description automatically generated

Hình 1.1.1: Logo Unity

* Unreal Engine: là phần mềm lập trình game cho phép các nhà lập trình game tạo ra môi trường 3D theo thời gian thực. Cụ thể, phần mềm làm game này cho phép tạo ra các mô hình 3D cũng như các hiệu ứng tương tác với hình ảnh chất lượng cao và chuyển động mượt mà. Bạn có thể sử dụng phần mềm lập trình game này để phát triển game trên đa dạng nền tảng từ Mobile, Pc cho đến PS4, Xbox hay Nintendo Switch.

Ra đời năm 1998, Unreal Engine tính đến nay đã có 5 phiên bản với phiên bản mới nhất là Unreal Engine cùng rất nhiều tính năng cải tiến hỗ trợ tối đa cho các nhà phát triển game. Trước đó, khi sử dụng ứng dụng làm game này, người dùng cần trả một khoản phí. Tuy nhiên, từ khi Unreal Engine 4 ra đời, thì ứng dụng tạo game này được cho phép sử dụng miễn phí nếu bạn đồng ý trả cho nhà phát hành – Epic games 5% doanh số từ trò chơi của bạn.



Hình 1.1.2: Logo UnrealEngine

* Cocos2d-x: Hỗ trợ cả hai định dạng 2D và 3D, ứng dụng lập trình game Cocos2d-x hỗ trợ phát triển trò chơi đa nền tảng như PC, Mobile, … với mã nguồn mở. Phần mềm lập trình game này sở hữu các chức năng phong phú giúp người dùng thoải mái thực hiện các ý tưởng game của mình như kết xuất đồ họa, GUI, âm thanh, hình ảnh vật lý, …

Sử dụng các ngôn ngữ lập trình C#, Lua hoặc JavaScript. Cocos2d-x cung cấp cho các nhà phát triển game những tài nguyên cơ bản để thiết kế một tựa game chỉ bằng các động tác kéo và thả đơn giản. Hiện tại, phần mềm làm game này hỗ trợ triển khai phát triển game cho các hệ điều hành IOS, Android, HTML5, Windows và Mac



Hình 1.1.1: Logo Cocos2d-x

* Gotdot: Giống như Unity, ứng dụng lập trình game Godot hỗ trợ tạo cả trò chơi cho cả hai định dạng 2D và 3D. Với hiệu suất tốt hơn, game 2D của phần mềm lập trình game này đã được thiết kế ít lỗi hơn và quy trình làm việc tổng thể gọn gàng hơn. Godot có thể xuất bản game đến nhiều nền tảng ngay lập tức, bao gồm Windows, Mac, Linux, Android, iOS và HTML5. Các nhà phát triển game không cần mua thêm giấy phép, mặc dù có một số hạn chế như cần phải sử dụng hệ thống Mac để triển khai hệ nhị phân Mac.



Hình 1.1.1: Logo Gotdot

## Các ngôn ngữ làm game

Các ngôn ngữ làm game là những ngôn ngữ lập trình được sử dụng để viết mã nguồn cho game. Các ngôn ngữ làm game thường có các đặc điểm như:

Dễ học và sử dụng: Các ngôn ngữ làm game thường có cú pháp đơn giản và rõ ràng, giúp cho người lập trình dễ hiểu và viết mã nguồn.

Nhanh và hiệu quả: Các ngôn ngữ làm game thường có khả năng biên dịch hoặc thông dịch nhanh chóng, giúp cho quá trình kiểm thử và chạy game được thuận tiện hơn.

Linh hoạt và mở rộng: Các ngôn ngữ làm game thường có khả năng tương thích với nhiều công cụ và thư viện khác nhau, giúp cho quá trình thiết kế và phát triển game được đa dạng hơn.

Một số ngôn ngữ làm game phổ biến hiện nay bao gồm:

C#: Một ngôn ngữ lập trình hướng đối tượng, được sử dụng trong công cụ Unity để viết mã nguồn cho game. C# có cú pháp giống với Java và C++, có khả năng hỗ trợ lập trình đa luồng và xử lý sự kiện.

JavaScript: Một ngôn ngữ lập trình thông dịch, được sử dụng trong công cụ Unity hoặc các công cụ khác để viết mã nguồn cho game. JavaScript có cú pháp giống với Java và C#, có khả năng hỗ trợ lập trình hướng mẫu (prototype) và xử lý sự kiện.

C++: Một ngôn ngữ lập trình hướng đối tượng, được sử dụng trong công cụ Unreal Engine hoặc các công cụ khác để viết mã nguồn cho game. C++ có cú pháp giống với C#, có khả năng hỗ trợ lập trình thấp cấp (low-level) và xử lý bộ nhớ.

GML: Một ngôn ngữ lập trình thông dịch, được sử dụng trong công cụ GameMaker Studio để viết mã nguồn cho game. GML có cú pháp đơn giản và dễ học, có khả năng hỗ trợ các hàm và biến toàn cục và cục bộ.

Python: Một ngôn ngữ lập trình thông dịch, được sử dụng trong các công cụ như Pygame, Panda3D hoặc Blender để viết mã nguồn cho game. Python có cú pháp rõ ràng và sạch sẽ, có khả năng hỗ trợ lập trình hướng đối tượng và hàm.



# CHƯƠNG 2: TỔNG QUAN VỀ CÔNG CỤ LÀM GAME UNITY



## Unity là gì?

Unity là một công cụ phát triển game đa nền tảng được phát triển bởi Unity Technologies từ những năm 2005, hiện nay nó đã dần trở thành một game engine phổ biến nhất trên thế giới. Các nền tảng được hỗ trợ Unity hiện nay là Android, IOS, Linux, macOS, Windows, Windows Phone, ...

Unity 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 nên cũng khá dễ sử dụng. Ngôn ngữ lập trình chính của Unity là C#, ngoài ra còn có hỗ trợ cho Javascript.

Unity cũng tận dụng chức năng của các thư viện phần mềm như engine mô phỏng vật lý PhysicX của Nvidia, OpenGL và Direct3D để kết xuất hình ảnh 3D, OpenAL cho âm thanh, … nên nó hỗ trợ rất mạnh cho công việc lập trình game.

A black background with a white object in the middle

Description automatically generated

Hình 2.1: Logo Unity

## Ưu điểm của Unity

* Dễ học và sử dụng: Unity có một cộng đồng lớn và phong phú, cung cấp nhiều tài nguyên giáo trình và hỗ trợ, làm cho nó trở thành một lựa chọn lý tưởng cho cả người mới học lập trình và những nhà phát triển kinh nghiệm.
* Đồng nhất đa nền tảng: Unity cho phép phát triển một lần và triển khai trên nhiều nền tảng, giảm thiểu công sức cần thiết cho việc đa nền tảng hóa.
* Cộng đồng lớn và hỗ trợ: Cộng đồng Unity rất tích cực và hỗ trợ lẫn nhau thông qua diễn đàn, blog, video hướng dẫn, và các sự kiện cộng đồng.
* Thư viện tài nguyên phong phú: Unity Asset Store cung cấp một nguồn lực khổng lồ các tài nguyên, công cụ, và asset có sẵn để giúp nhà phát triển tăng tốc quá trình phát triển.

## Các thành phần trong Unity

Dưới đây là 1 số cửa sổ cơ bản mà chúng ta thường dùng khi sử dụng Unity.

A screenshot of a computer

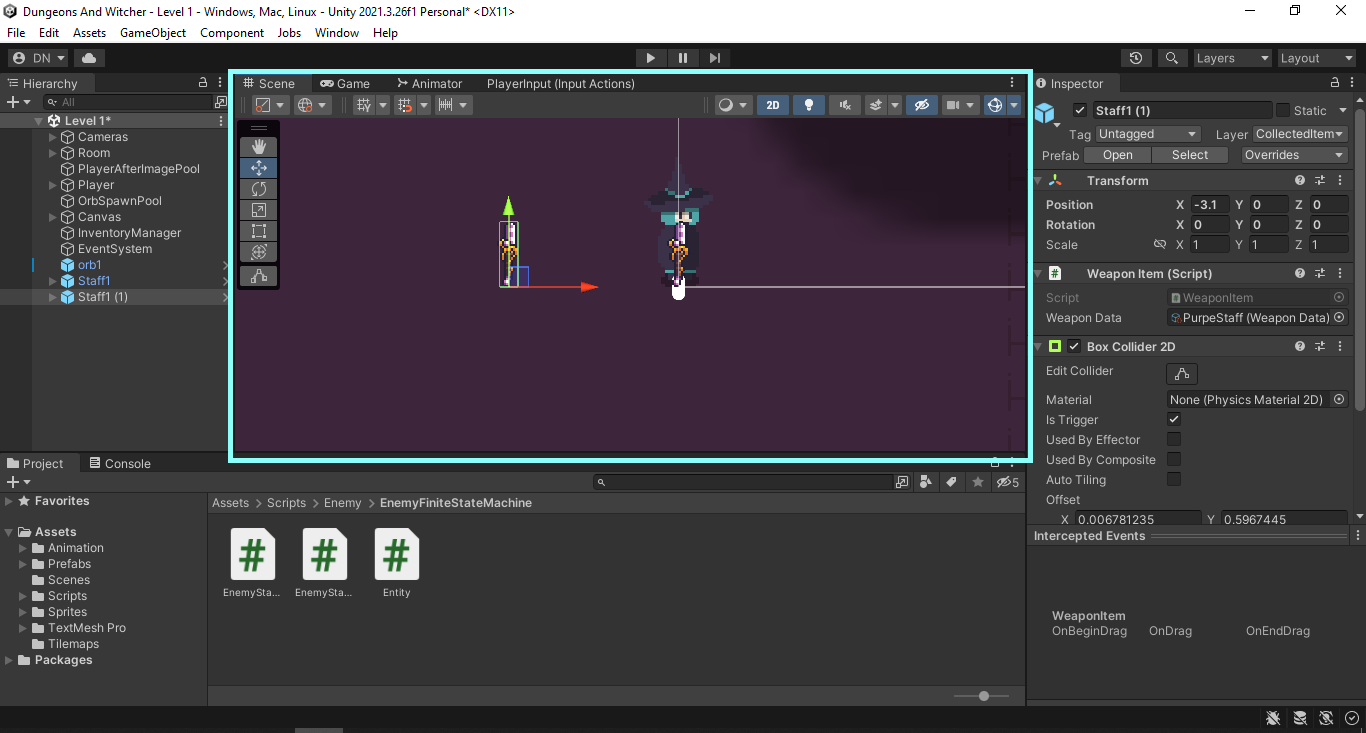
Description automatically generated

Hình 2.3. Cửa sổ làm việc của unity

### Cửa sổ Sence

Trong Unity, chúng ta có thể xem Scenes là các màn chơi, cấp độ chơi riêng lẻ, hoặc các vùng của nội dung trò chơi. Ví dụ như Main menu, Options, About …

Bằng cách xây dựng trò chơi với nhiều cảnh, chúng ta sẽ có thể phân phối thời gian tải và thử nghiệm các phần khác nhau của trò chơi riêng lẻ một cách nhanh chóng và chính xác. Trong cửa sổ này, bạn có thể thêm, xóa và sắp xếp các đối tượng, ánh sáng, camera, và các thành phần khác để tạo nên môi trường trò chơi.

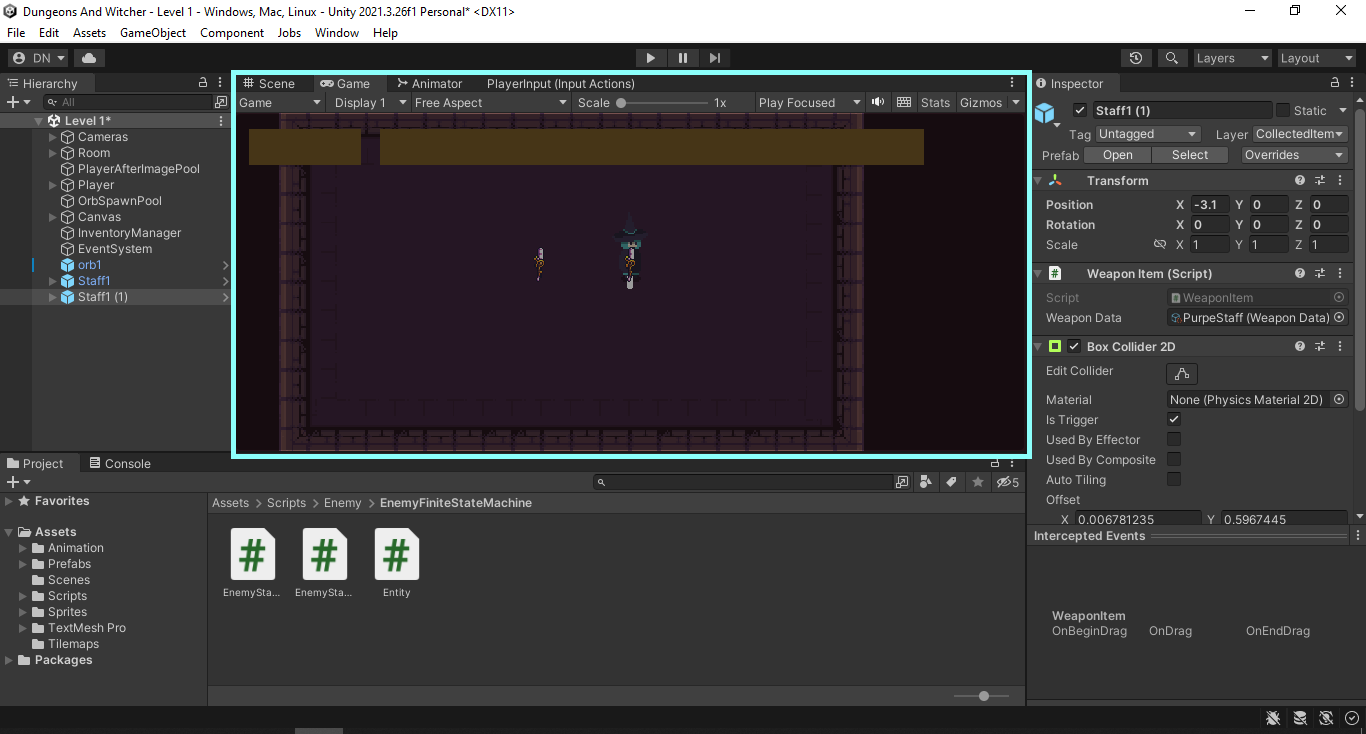


Hình 2.3.1. Cửa sổ Sence

### Cửa sổ Game view

Game view là góc nhìn từ những Camera trong Game. Đó là những gì được nhìn thấy khi hoàn tất, khi Game được xuất bản. Chúng ta sẽ cần ít nhất là một hoặc nhiều hơn số lượng các Camera để quyết định những gì mà người chơi sẽ nhìn thấy khi họ chơi Game.

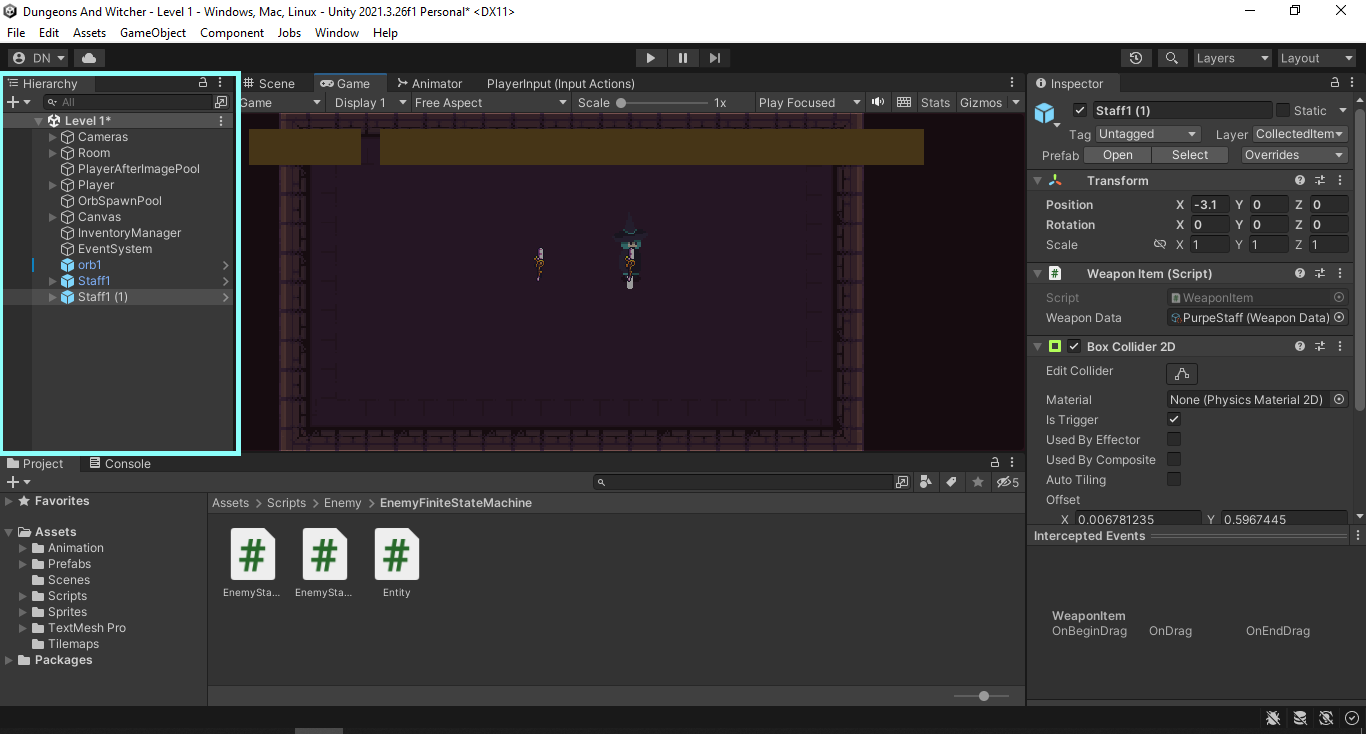
Cửa sổ Game view cho phép bạn xem trò chơi trong khi đang phát triển. Điều này giúp bạn kiểm tra và thử nghiệm trải nghiệm người chơi trực tiếp từ góc độ của người chơi.



Hình 2.3.2. Cửa sổ Game view

### Cửa sổ Hierarchy

Cửa sổ Hierarchy liệt kê tất cả các đối tượng trong scene hiện thời. Đối tượng nào được hiển thị ở đây sẽ xuất hiện trong Sence. Quản lý các đối tượng bằng cách kéo và thả chúng, thay đổi thứ tự, và tạo các đối tượng (prefab).

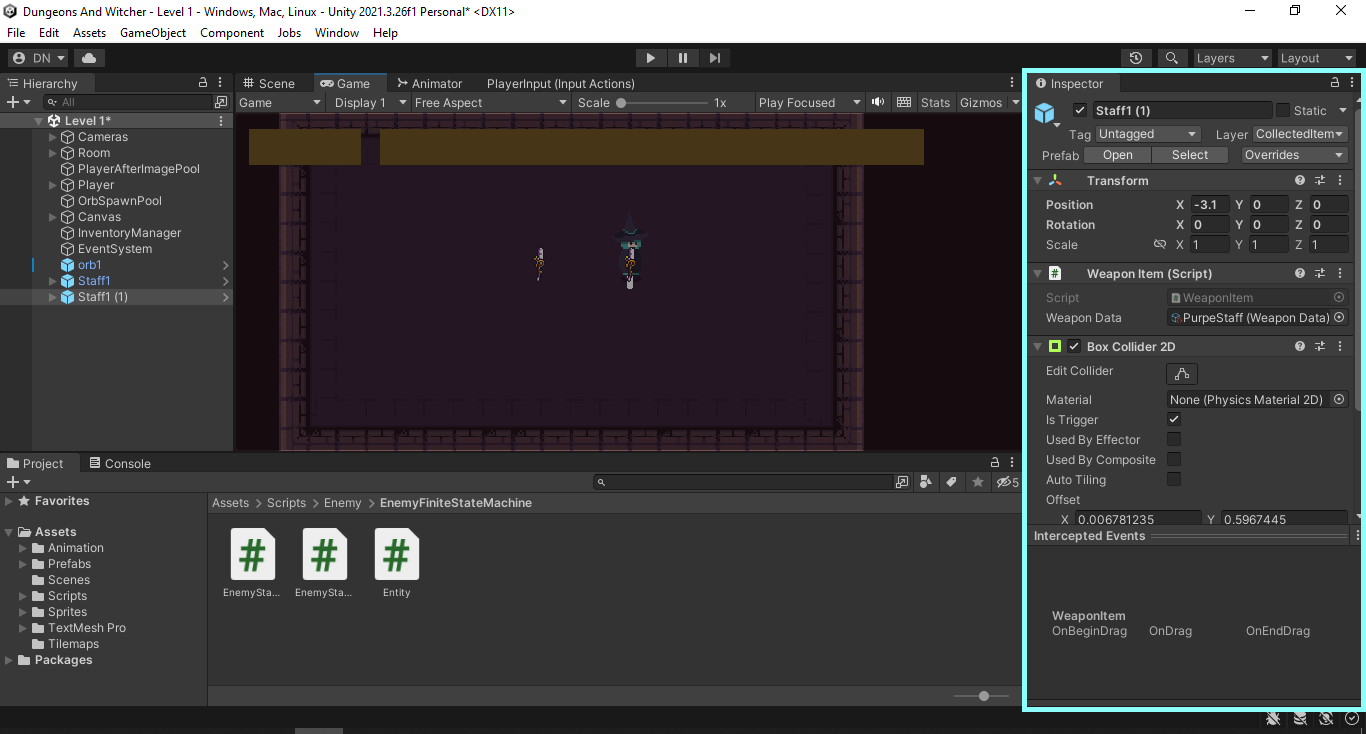


Hình 2.3.3. Cửa sổ Hierarchy

### Cửa sổ Inspector

Cửa sổ Inspector hiển thị thông tin chi tiết và các thuộc tính của đối tượng được chọn trong Scene hoặc Hierarchy. Bạn có thể điều chỉnh các giá trị, thêm các thành phần, và tùy chỉnh các thuộc tính của đối tượng ở đây.

Mọi thuộc tính thể hiện trong Inspector đều có thể dễ dàng tuỳ chỉnh trực tiếp mà không cần thông qua một kịch bản định trước. Tuy nhiên Scripting API cung cấp một số lượng nhiều và đầy đủ hơn do giao diện Inspector là có giới hạn.



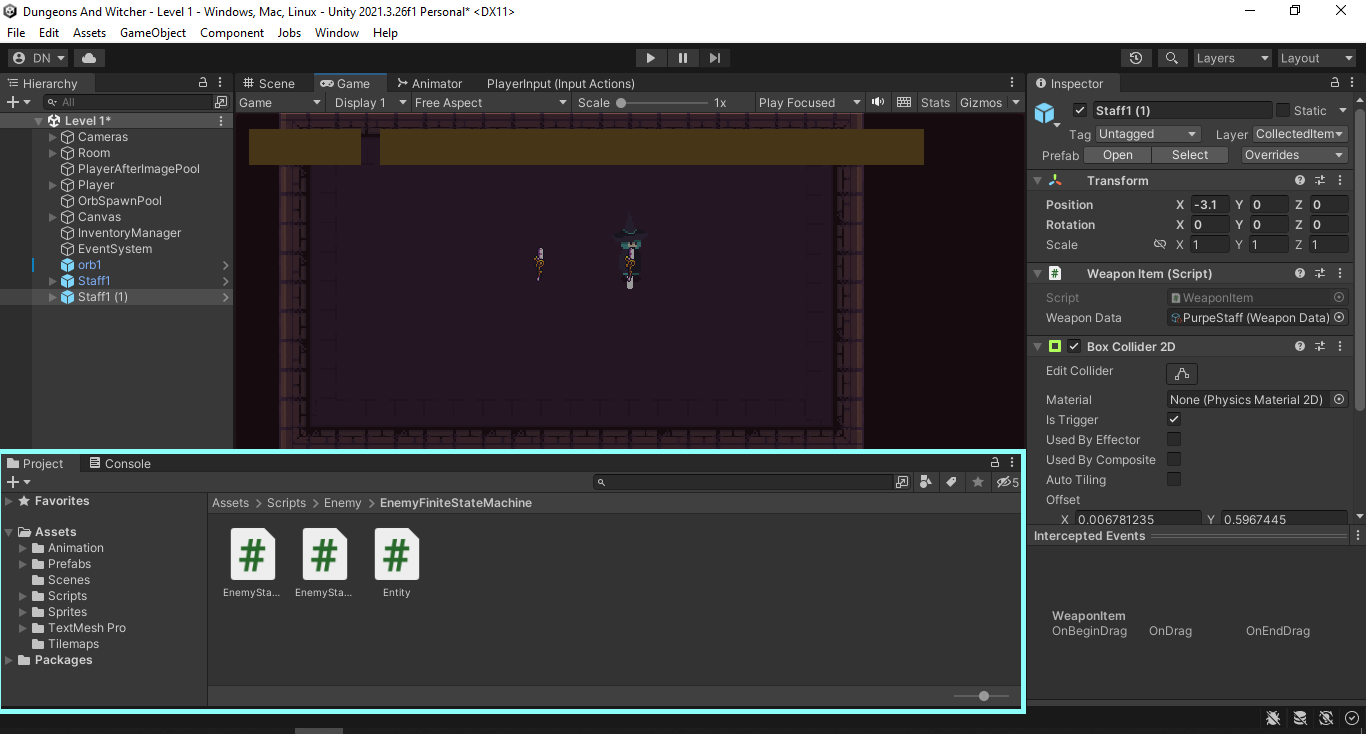
Hình 2.3.4. Cửa sổ Inspector

### Cửa sổ Project

Cửa sổ Project hiển thị cấu trúc thư mục của dự án. Nó chứa tất cả các tài nguyên như hình ảnh, âm thanh, văn bản, mã nguồn và các prefab. Bạn có thể quản lý và tổ chức tài nguyên của mình từ đây.

Mỗi Project của Unity đều chứa một thư mục Assets. Nội dung của thư mục này được hiển thị trong Project View. Nó chứa tất cả các tài nguyên như hình ảnh, âm thanh, văn bản, mã nguồn và các prefab. Chúng ta có thể quản lý và tổ chức tài nguyên của mình tại đây.

Để thêm assets vào project, chúng ta có thể kéo thả bất kỳ file nào vào trong project view hoặc vào công cụ Assets – Import new Assets (Click chuột phải vào Project View). Scenes cũng được lưu trữ trong Project view, và đây là một level độc lập mang tính cá nhân. Chúng ta dễ dàng tạo một assets game trong Unity bằng cách chọn hình tam giác nhỏ nằm bên phải Create trong cửa sổ Project hoặc click chuột phải trong Project View, sau đó chọn assets tương ứng.

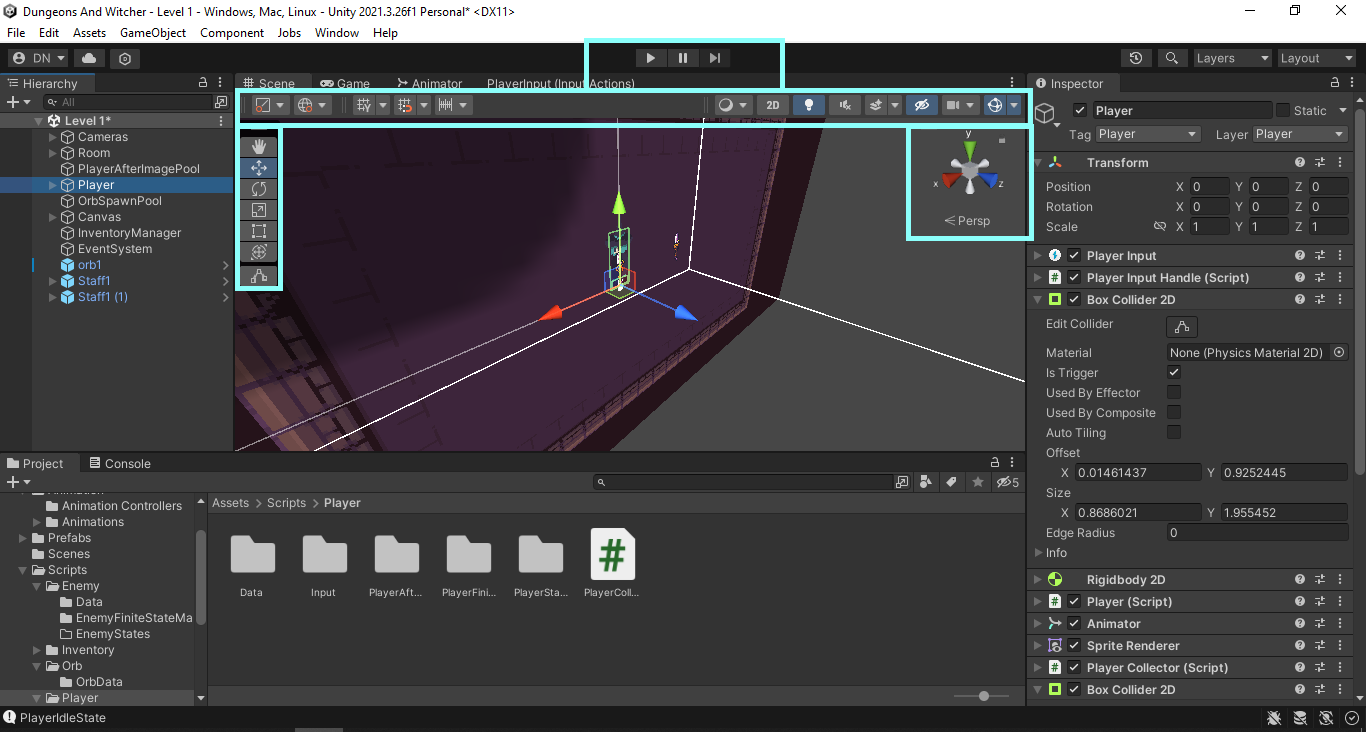


Hình 2.3.5. Cửa sổ Project

### Cửa sổ Toolbar

Toolbar chứa các loại điều khiển cơ bản, mỗi loại giữ một vai trò quan trọng trong Editor.

* Transform Tool: được dung với Scene view, như quay trái, phải, lên trên, xuống dưới, phóng to thu nhỏ đối tượng.
* View Options: Dùng cho việc thể hiện Scene view như 2D, 3D, audio, light,....
* Play/Pause/Step Buttons: dùng cho view game, chạy game ngay trong Editor để kiểm tra.

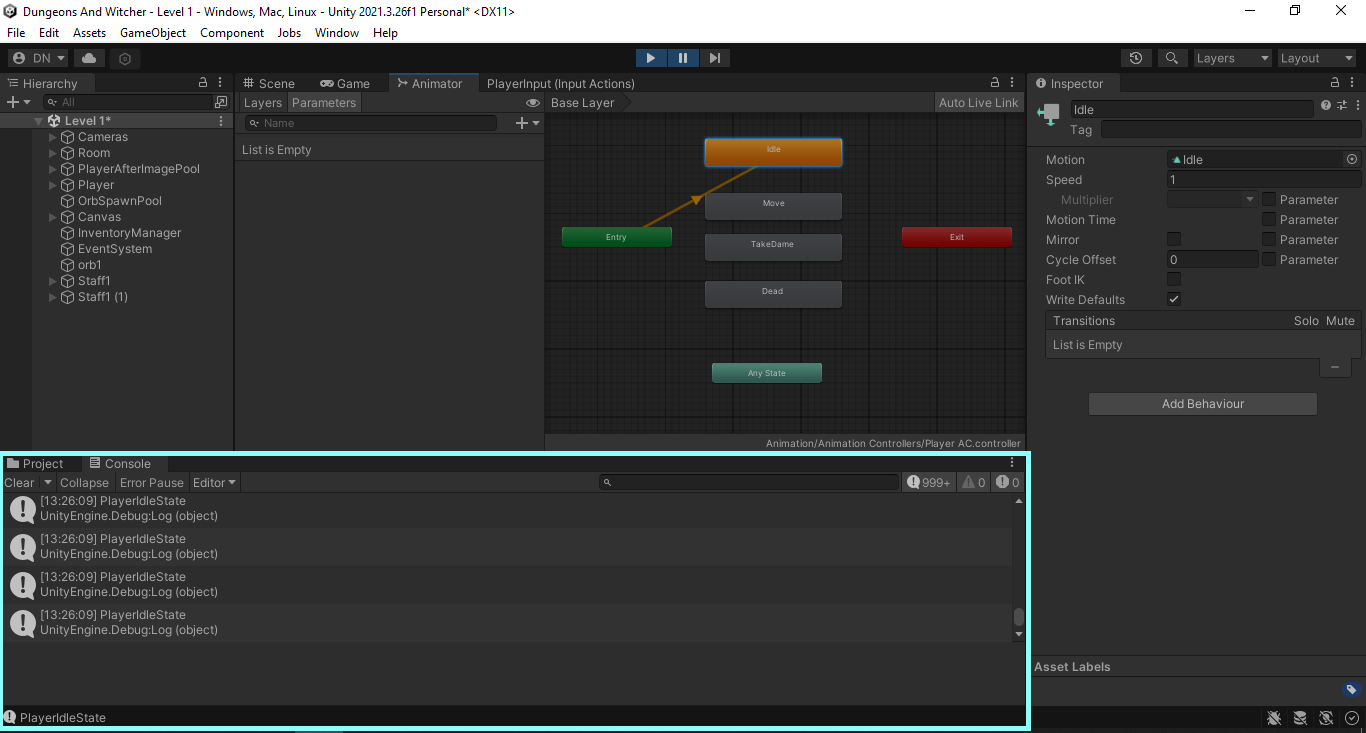


Hình 2.3.6. Cửa sổ Toolbar

### Cửa sổ Console

Cửa sổ Console hiển thị các lỗi, cảnh báo và thông báo khác mà Editor tìm thấy. Những lỗi hoặc cảnh báo này giúp chúng ta tìm ra vấn đè trong dự án để sửa lại lỗi.

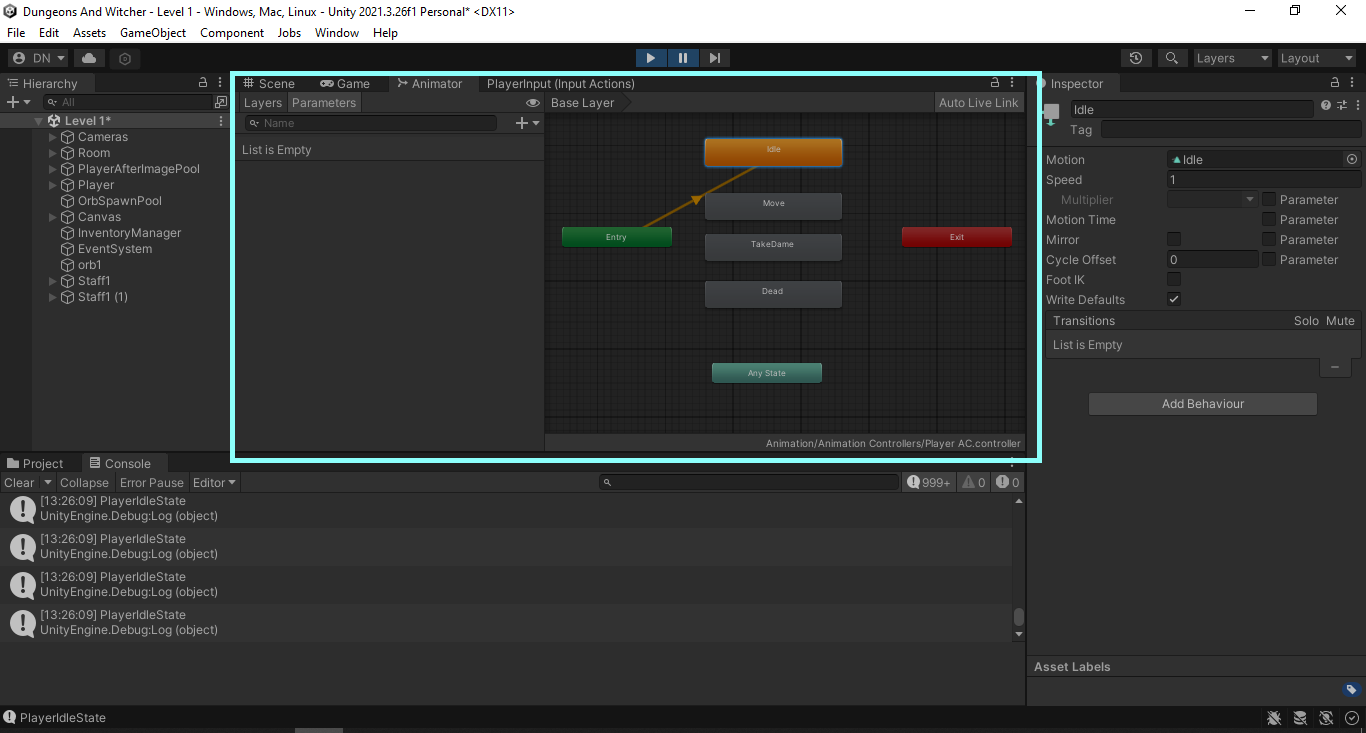
Ngoài ra ta có thể sử dụng Debug để hiện những test case cần thiết để xem code của mình hoạt động như nào.



Hình 2.3.7. Cửa sổ Console

### Cửa sổ Animator

Cửa sổ Animator được sử dụng để tạo và quản lý các trạng thái và chuyển động của các đối tượng trong trò chơi. Bạn có thể tạo và chỉnh sửa các animation clips và điều khiển cách chúng được kích hoạt trong trò chơi.



Hình 2.3.8. Cửa sổ Animator

## Các khái niệm trong Unity

Dưới đây là mô tả chi tiết về một số thành phần quan trọng có trong Unity: GameObject, Component, Sprite, Animation, Prefab, Script, Assets, Camera, Sound, Transform, Material và Shader.

### GameObject

GameObject là thành phần cơ bản trong Unity, đại diện cho mọi đối tượng trong cảnh (scene). Đối tượng này có thể là nhân vật, vật phẩm, ánh sáng,…. GameObject không thể tồn tại một cách độc lập mà phải chứa các Component để có thể thực hiện các chức năng cụ thể.

### Compoment

Component là các thành phần gắn liền với GameObject, dùng để cung cấp các chức năng và tính năng cho GameObject. Các Component bao gồm như Renderer, Collider, và Script,…. Mỗi GameObject có thể có nhiều Component khác nhau để thực hiện các nhiệm vụ và hành vi cụ thể.

### Sprite

Sprite là một hình ảnh 2D được sử dụng để hiển thị trên màn hình. Trong Unity, Sprite thường được sử dụng cho các đối tượng như nhân vật, vật phẩm, và nền của trò chơi 2D.

### Animation

Animation là quá trình tạo ra chuyển động cho GameObject trong trò chơi. Bạn có thể tạo Animation bằng cách ghi lại các keyframe, xác định sự thay đổi trong thuộc tính Transform trong khoảng thời gian, bạn cũng có thể thêm các sự kiện kích hoạt ở các keyframe cụ thể.

### Prefab

Prefab là một mô hình hoặc đối tượng có thể được tái sử dụng trong nhiều địa điểm khác nhau trong dự án. Prefab giúp tiết kiệm thời gian và duy trì tính thống nhất trong trò chơi.

### Script

Script là các đoạn mã nguồn viết bằng ngôn ngữ lập trình như C# hoặc JavaScript, được gắn kết với GameObject để thực hiện các chức năng và tương tác trong trò chơi. Script là một phần quan trọng của việc lập trình trong Unity.

### Assets

Assets là tất cả các tài nguyên (resources) như hình ảnh, âm thanh, văn bản, và Prefab có thể được sử dụng trong dự án Unity. Cửa sổ Project chứa toàn bộ các Assets.

### Camera

Camera quyết định góc nhìn và cách mà người chơi nhìn thấy cảnh trong trò chơi. Nó đóng vai trò quan trọng trong việc xác định cách mà trò chơi được hiển thị.

### Sound

Sound là âm thanh trong game như nhạc nên, hiệu ứng âm thanh.

### Transform

Transform xác định vị trí, quay, và kích thước của một GameObject trong không gian 3D hoặc 2D.

### Material và Shader

Shader là đoạn script quy định cách thức render của chất liệu trên bề mặt vật thể. Material sử dụng shader để làm chất liệu cho mô hình. Giữa materials và shaders có mối liên hệ với nhau. Shaders quy định các thuộc tính cần để shader làm việc. Còn material cho phép gán hình ảnh vào các thuộc tính đó từ Asset.

### Unity Flow

Trong Unity, có một số event được thực thi theo thứ tự xác định, vì hầu hết các hàm sử dụng trong Unity đều kế thừa từ class MonoBehaviour. Thứ tự thực hiện này được mô tả như sau:

A screenshot of a computer

Description automatically generated

Hình ..12. Thứ tự thực thi trong Unity

### Finite State Machine:

Finite State Machine hay còn gọi là một cỗ máy trạng thái hữu hạn, là một mô hình tính toán dựa trên một cỗ máy giả định được tạo thành từ một hoặc nhiều trạng thái. Chỉ một trạng thái duy nhất có thể hoạt động cùng lúc, do đó cỗ máy phải chuyển từ trạng thái này sang trạng thái khác để thực hiện các tác vụ khác nhau.

Finite State Machine thường được sử dụng để tổ chức và đại diện cho luồng thực thi, rất hữu ích để triển khai AI trong trò chơi. Ví dụ, hành vi của kẻ thù có thể được thực hiện bằng cách sử dụng FSM: mỗi trạng thái đại diện cho một hoạt động, chẳng hạn như tấn công hoặc né tránh.

A diagram of a game

Description automatically generated

Hình 2.4.13. Finite State Machine

## Các lớp Unity API

Dưới đây là liệt kê các lớp Unity API thường được sử dụng.

### MonoBehaviour Class

Là lớp cơ bản của hầu hết các script trong Unity, nó làm cho script có thể được gắn liền với GameObject và thực hiện các hàm số giống như vòng đời của đối tượng.

* VD: public class MyScript : MonoBehaviour { /\*...\*/ }

### ScriptableObject Class

Cho phép lưu trữ dữ liệu không phải là prefab và có thể chia sẻ giữa các đối tượng.

* VD: *public class MyScriptableObject : ScriptableObject { //code here… }*

### GameObject Class

Đại diện cho đối tượng trong trò chơi và là cơ sở cho hầu hết các thành phần.

* VD: GameObject myObject = new GameObject("ObjectName");

### Component Class

Lớp cơ bản cho tất cả các thành phần gắn liền với GameObject.

* VD: Rigidbody2D rb = myObject.GetComponent<Rigidbody2D>();

### Transform Class

Định nghĩa vị trí, quay, và kích thước của GameObject.

* VD: *Transform myTransform = myObject.transform;*

### Animator Class

Quản lý và phát các Animation cho GameObject.

* VD: *Animator myAnimator = myObject.GetComponent<Animator>();*

### SpriteRenderer Class

Hiển thị Sprite cho đối tượng trong môi trường 2D.

* VD: *SpriteRenderer SpriteR = myObject.GetComponent<SpriteRenderer>();*

### Collider Class

Xác định vùng va chạm của GameObject để xử lý các sự kiện va chạm.

* VD: *BoxCollider2D myCollider = myObject.GetComponent<BoxCollider2D>();*

### Camera Class

Quản lý cách mà trò chơi được hiển thị trên màn hình.

* VD: *Camera mainCamera = Camera.main;*

### AudioSource Class

Quản lý và phát âm thanh trong trò chơi.

* VD: *AudioSource audioSource = myObject.GetComponent<AudioSource>();*

### Physics Class

Cung cấp các hàm để tương tác với hệ thống vật lý trong trò chơi.

* VD: *bool isGrounded = Physics.Raycast(transform.position, Vector3.down, 0.1f);*

### SceneManager Class

Quản lý các scene trong dự án Unity.

* VD: *SceneManager.LoadScene("NextScene");*

### Input Class

Xử lý đầu vào từ bàn phím, chuột, và các thiết bị khác.

* VD: *if (Input.GetKeyDown(KeyCode.Space)) { //Jump }*

### Time Class

Cung cấp thông tin về thời gian trong trò chơi, chẳng hạn như deltaTime, TimeScale.

* VD: *float moveSpeed = 10f \* Time.deltaTime;*

### Rigidbody Class

Điều khiển vật lý của GameObject, cho phép áp dụng lực và làm cho đối tượng phản ứng với lực tự do. Đây là lớp cần thiết đối với tất cả đối tượng vật lý.

* VD: Rigidbody2D myRigidbody = myObject.GetComponent<Rigidbody2D>();

# Chương III. Mô tả thiết kế game



## Ý tưởng

Ý tưởng trò chơi "Dungeons And Witcher" dựa trên cơ chế của “The Binding of Isaac” và “Noita”, game tập trung vào cuộc phiêu lưu hấp dẫn của một phù thủy trong một thế giới dưới lòng đất đầy nguy hiểm. Trong game, người chơi sẽ vào vai một phù thủy và bắt đầu hành trình khám phá hầm ngục bí ẩn.

Nhiệm vụ chính của người chơi là đối mặt với những thách thức khó khăn, đánh quái vật và tiêu diệt các boss mạnh mẽ để thu thập những cây đũa phép có sức mạnh đặc biệt. Mỗi cây đũa phép sẽ mang theo một loạt các kỹ năng phép thuật độc đáo, giúp người chơi thích nghi với các tình huống khác nhau. Giữa các tầng của hàm ngục sẽ có những căn phòng trao đổi giúp người chơi nâng cấp những cây đũa phép của mình và tạo ra những phép mới lạ.

Trong suốt cuộc hành trình, người chơi sẽ phải tìm kiếm và sử dụng thông minh các vật phẩm, thuốc phép, và trang bị để cải thiện sức mạnh của họ và tăng khả năng sinh tồn. Các hầm ngục sẽ luôn thay đổi, đảm bảo sự đa dạng và tính chất ngẫu nhiên trong mỗi lần chơi.

## Thiết kế game

### Thiết kế người chơi

Trạng thái người chơi: Idle, Move, Takendame, Dead.

Người chơi có chức năng:

* Di chuyển tránh chướng ngại vật, tránh kẻ địch và đạn.
* Sử dụng vũ khí để tấn công kẻ địch.
* Thu thập vật phẩm, gậy phép và ngọc phép.



Hình 3.2.1. Sprite người chơi

### Thiết kế kẻ địch

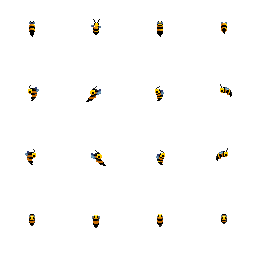
Kẻ địch có trạng thái: Idle, Move, Attack, TakenDame, Dead

Kẻ địch bao gồm: Red Monster, Bee.

* Kẻ địch sẽ di chuyển ngẫu nhiên trong phạm vi phòng.
* Kẻ địch sẽ tấn công người chơi nếu người chơi trong tầm.



Hình 3.2.2a. Sprite quái vật



Hình 3.2.2b. Sprite quái vật

### Thiết kế map

Map gồm hành lang và phòng, quái vật sẽ xuất hiện trong phòng.

Map được tạo ngẫu nhiên khi hầm ngục được tạo.



Hình 3.2.3. Sprite map

### Thiết kế vũ khí

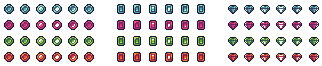
Có nhiều loại gậy phép, mỗi gậy phép lại có thông số khác nhau.



Hình 3.2.4. Sprite vũ khí

### Thiết kế ngọc phép

Có nhiều loại ngọc phép, mỗi loại ngọc phép có thông số khác nhau.



Hình 3.2.5. Sprite ngọc phép

## Các quy luật chơi

### Di chuyển

Người chơi di chuyển bằng bàn phím, di chuyển trái, phải, lên, xuống, lướt để né tấn công của kẻ địch. Người chơi chỉ có thể di chuyển trong phạm vi camera.

### Tấn công

Người chơi có thể bắn đạn để tiêu diệt kẻ địch, mỗi kẻ địch có một lượng máu nhất định, có quái vật máu thấp và có quái vật máu cao vậy nên người chơi cần phải bắn liên tục để tiêu diệt kẻ địch

### Thu thập

Người chơi khi vượt qua các phòng sẽ xuật hiệt vật phẩm ở giữa phòng, người chơi có thể thu thập vật phẩm để nâng cấp sức mạnh của bản thân.

### Thanh trạng thái

Người chơi có các thông số liên quan đến máu, năng lượng, chỉ số. Các thông số này thay đổi khi người chơi tấn công, nhận sát thương hoặc nhặt vật phẩm. Người chơi cần đảm bảo không làm giảm hết số máu để sống sót.

# Chương IV: Phân tích và thiết kế hệ thống

## Xác định các tác nhân hệ thống

|  |  |  |
| --- | --- | --- |
| STT | Tác nhân | Nhiệm vụ |
| 1 | Người chơi | Điều khiển, tương tác với trò chơi thông qua dữ liệu đầu vào là các phím bấm.  Các chức năng chính:   * Chuyển đổi trạng thái * Tương tác với kẻ địch * Lựa chọn chúc phúc * Quản lý vật phẩm * Quản lý thống kê * Lưu trữ |

Bảng 4.1. Xác định các tác nhân hệ thống

## Xây dựng biểu đồ Usecase

### Mô hình Usecase tổng quát

A diagram of a person with text

Description automatically generated

Hình 4.2.1. Mô hình Usecase tổng quát

## Đặc tả và phân tích chức năng của hệ thống

### Usercase Chuyển đổi trạng thái

Đặc tả Usecase

|  |  |  |  |
| --- | --- | --- | --- |
| Mã use-case | 01 | | |
| Tên use-case | Chuyển đổi trạng thái | | |
| Tác nhân | Người chơi | | |
| Mô tả | Chức năng cho phép tác nhân chuyển đổi trạng thái nhân vật như đứng yên, di chuyển, tấn công, tương tác,… dựa trên đầu vào tác nhân truyền đến cho hệ thống. | | |
| Điều kiện kích hoạt | Tác nhân thực hiện truyền đầu vào là các phím điều khiển chức năng di chuyển, tấn công, tương tác. | | |
| Điều kiện thành công | Trạng thái nhân vật được thay đổi. | | |
| Luồng sự kiện |  | Tác nhân | Hệ thống phản hồi |
| 1 | Tác nhân truyền trạng thái mới đến hệ thống. |  |
| 2 |  | Hệ thống kiểm tra trạng thái hợp lệ, cập nhật lại thông tin và thực hiện chuyển đổi trạng thái. |
| Luồng thay thế |  |  |  |
| Luồng ngoại lệ | 2a.1 |  | Nếu trạng thái không hợp lệ, hệ thống không thực hiện chức năng. |

Bảng 4.3.1 Đặc tả Usercase Chuyển đổi trạng thái

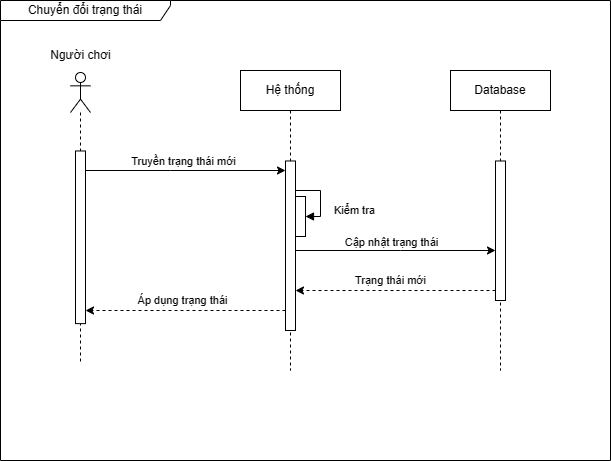
Biểu đồ luồng hoạt động

A diagram of a diagram

Description automatically generated

Hình 4.3.1 Biểu đồ luồng hoạt động Usercase Chuyển đổi trạng thái

Biểu đồ trình tự



Hình 4.3.1.2 Biểu đồ luồng hoạt động Usercase Chuyển đổi trạng thái

### Usercase Tương tác với kẻ địch

Đặc tả Usercase

|  |  |  |  |
| --- | --- | --- | --- |
| Mã use-case | 02 | | |
| Tên use-case | Tương tác với kẻ địch | | |
| Tác nhân | Người chơi | | |
| Mô tả | Chức năng này cho phép tác nhân điều khiển nhân vật để truyền tương tác và chiến đấu với kẻ thù. | | |
| Điều kiện kích hoạt | Tác nhân thực hiện truyền đầu vào là các phím điều khiển chức năng di chuyển, tấn công, tương tác. | | |
| Điều kiện thành công | Thông tin lưu trữ của nhân vật hoặc quái vật đã được thay đổi. | | |
| Luồng sự kiện |  | Tác nhân | Hệ thống phản hồi |
| 1 | Tác nhân truyền tương tác từ nhân vật đến kẻ thù. |  |
| 2 |  | Hệ thống lấy thông tin tương tác, cập nhật vào nhân vật và kẻ thù, hiển thị tương tác lên màn hình. |
| Luồng thay thế |  |  |  |
| Luồng ngoại lệ |  |  |  |

Bảng 4.3.2 Đặc tả Usercase Tương tác với kẻ địch

Biểu đồ luồng hoạt động

A diagram of a diagram

Description automatically generated

Hình 4.3.2 Biểu đồ luồng hoạt động Usercase Tương tác với kẻ địch

Biểu đồ trình tự

A diagram of a diagram

Description automatically generated

Hình 4.3.2.2 Biểu đồ luồng hoạt động Usercase Tương tác với kẻ địch

### Usercase Lựa chọn chúc phúc

Đặc tả Usercase

|  |  |  |  |
| --- | --- | --- | --- |
| Mã use-case | 03 | | |
| Tên use-case | Lựa chọn chúc phúc | | |
| Tác nhân | Người chơi | | |
| Mô tả | Khi kết thúc 1 phòng, có xác xuất hệ thống sẽ đưa ra các chúc phúc để tác nhân lựa chọn, tăng cường đặc biệt dành cho nhân vật mà tác nhân điều khiển. | | |
| Điều kiện kích hoạt | Tác nhân điều khiển nhân vật hoàn thành thử thách của trò chơi. | | |
| Điều kiện thành công | Thông tin lưu trữ của nhân vật hoặc quái vật đã được thay đổi. | | |
| Luồng sự kiện |  | Tác nhân | Hệ thống phản hồi |
| 1 | Tác nhân lựu chọn chúc phúc. |  |
| 2 |  | Hệ thống cập nhật thông tin chúc phúc vào nhân vật và hiển thị lên màn hình. |
| Luồng thay thế |  |  |  |
| Luồng ngoại lệ |  |  |  |

Bảng 4.3.3 Đặc tả Usercase Lựa chọn chúc phúc

Biểu đồ luồng hoạt động

A diagram of a diagram

Description automatically generated

Hình 4.3.3 Biểu đồ luồng hoạt động Usercase Lựa chọn chúc phúc

Biểu đồ trình tự

A diagram of a flowchart

Description automatically generated

Hình 4.3.3.2 Biểu đồ luồng hoạt động Usercase Lựa chọn chúc phúc

### Usercase Quản lý vật phẩm

Đặc tả Usercase

|  |  |  |  |
| --- | --- | --- | --- |
| Mã use-case | 04 | | |
| Tên use-case | Quản lý vật phẩm | | |
| Tác nhân | Người chơi | | |
| Mô tả | Chức năng này cho phép người chơi điều khiển nhân vật nhặt và quản lý các vật phẩm trong trò chơi. | | |
| Điều kiện kích hoạt | Tác nhân truyền tương tác nhặt hoặc quản lý vật phẩm. | | |
| Điều kiện thành công | Thông tin vật phẩm được cập nhật ở kho lưu trữ vật phẩm của nhân vật. | | |
| Luồng sự kiện |  | Tác nhân | Hệ thống phản hồi |
| 1 | Tác nhân nhặt vật phẩm. |  |
| 2 |  | Kho đồ của nhân vật được cập nhật. |
| 3 | Tác nhân mở kho đồ. |  |
| 4 |  | Hiển thị danh sách vật phẩm. |
|  | 5 | Tác nhân chỉnh sửa kho đồ. |  |
|  | 6 |  | Cập nhật thông tin kho đồ và hiển thị lên màn hình. |
| Luồng thay thế |  |  |  |
| Luồng ngoại lệ |  |  |  |

Bảng 4.3.4 Đặc tả Usercase Quản lý vật phẩm

Biểu đồ luồng hoạt động

A diagram of a diagram

Description automatically generated

Hình 4.3.4 Biểu đồ luồng hoạt động Usercase Quản lý vật phẩm

Biểu đồ trình tự

A diagram of a diagram

Description automatically generated

Hình 4.3.4.2 Biểu đồ luồng hoạt động Usercase Quản lý vật phẩm

### Usercase Quản lý thống kê

Đặc tả Usercase

|  |  |  |  |
| --- | --- | --- | --- |
| Mã use-case | 05 | | |
| Tên use-case | Quản lý thống kê | | |
| Tác nhân | Người chơi | | |
| Mô tả | Chức năng này cho phép tác xem thông tin của vật phẩm, thông tin của nhân vật mà tác nhân điều khiển. | | |
| Điều kiện kích hoạt | Tác nhân mở bảng thông tin của vật phẩm hoặc của nhân vật. | | |
| Điều kiện thành công | Thông tin của vật phẩm hoặc nhân vật được hiển thị. | | |
| Luồng sự kiện |  | Tác nhân | Hệ thống phản hồi |
| 1 | Tác nhân mở bảng thông tin của vật phẩm hoặc nhân vật. |  |
| 2 |  | Hệ thống kiểm tra xem tác nhân cần xem thông tin của đối tượng nào rồi hiển thị thông tin lên màn hình. |
| Luồng thay thế |  |  |  |
| Luồng ngoại lệ |  |  |  |

Bảng 4.3.5 Đặc tả Usercase Quản lý thống kê

Biểu đồ luồng hoạt động

A diagram of a diagram

Description automatically generated

Hình 4.3.5 Biểu đồ luồng hoạt động Usercase Quản lý thống kê

Biểu đồ trình tự

A diagram of a diagram

Description automatically generated

Hình 4.3.5.2 Biểu đồ luồng hoạt động Usercase Quản lý thống kê

### Usercase Lưu trữ

Đặc tả Usercase

|  |  |  |  |
| --- | --- | --- | --- |
| Mã use-case | 06 | | |
| Tên use-case | Lưu trữ | | |
| Tác nhân | Người chơi | | |
| Mô tả | Chức năng này cho phép người chơi lưu trữ trạng thái game và tải lại nó. | | |
| Điều kiện kích hoạt | Tác nhân lựa chọn lưu trữ dữ liệu khi thoát khỏi trạng thái game hiện tại. | | |
| Điều kiện thành công | Trạng thái game được lưu trữ và sử dụng lại được. | | |
| Luồng sự kiện |  | Tác nhân | Hệ thống phản hồi |
| 1 | Tác nhân lựa chọn lưu trữ trạng thái game. |  |
| 2 |  | Hệ thống lưu trữ trạng thái game. |
| Luồng thay thế |  |  |  |

Bảng 4.3.6 Đặc tả Usercase Lưu trữ

Biểu đồ luồng hoạt động

A diagram of a diagram

Description automatically generated

Hình 4.3.6 Biểu đồ luồng hoạt động Usercase Lưu trữ

Biểu đồ trình tự

A diagram of a diagram

Description automatically generated

Hình 4.3.6.2 Biểu đồ luồng hoạt động Usercase Lưu trữ

## Xác định lớp

|  |  |  |
| --- | --- | --- |
| STT | Tên lớp | Mô tả |
| 1 | Player | Lớp người chơi |
| 2 | Enemy | Lớp kẻ địch |
| 4 | Item | Lớp vật phẩm |
| 5 | Relic | Lớp chúc phúc |
| 6 | Weapon | Lớp vũ khí |
| 7 | Room | Lớp phòng |
| 8 | System | Lớp hệ thống |

Bảng 4.4. Xác định các lớp

# Chương V: Xây dựng và demo

# Chương VI: Kết luận và hướng phát triển