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

TRƯỜNG ĐẠI HỌC THỦY LỢI



DƯƠNG GIÁP ĐỨC

**PHÁT TRIỂN TỰA GAME SINH TỒN 2D QUA CÔNG CỤ UNITY ENGINE**

ĐỒ ÁN TỐT NGHIỆP

HÀ NỘI, NĂM 2023

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

TRƯỜNG ĐẠI HỌC THỦY LỢI

DƯƠNG GIÁP ĐứC

**PHÁT TRIỂN TỰA GAME SINH TỒN 2D QUA CÔNG CỤ UNITY ENGINE**

|  |  |
| --- | --- |
| Ngành: | Công nghệ thông tin |
| Mã số: | 1951060614 |

|  |  |
| --- | --- |
| NGƯỜI HƯỚNG DẪN | 1. TS. Trương Xuân Nam |

HÀ NỘI, NĂM 2023

**GÁY BÌA ĐỒ ÁN TỐT NGHIỆP, KHÓA LUẬN TỐT NGHIỆP**

**DƯƠNG GIÁP ĐỨC ĐỒ ÁN/KL TỐT NGHIỆP HÀ NỘI, NĂM 2023**

**LỜI CAM ĐOAN**

Tác giả xin cam đoan rằng Đồ án tốt nghiệp là kết quả nghiên cứu của bản thân em và được sự hướng dẫn của TS. Trương Xuân Nam. Các kết quả trong đồ án tốt nghiệp đều là trung thực không qua bất kỳ nguồn nào và sao chép dưới bất kỳ hình thức nào. Việc tham khảo (nếu có) đã được thực hiện trích dẫn và ghi nguồn tài liệu tham khảo đúng quy định.

TÁC GIẢ ĐATN

Dương Giáp Đức

LỜI CÁM ƠN

Trong quá trình hoàn thiện đề tài “Phát triển game sinh tồn 2D qua công cụ Unity Engine”, em đã nhận được rất nhiều sự giúp đỡ của thầy cô và bạn bè.

Đầu tiên em xin chân thành cảm ơn thầy cô tại trường Đại học Thủy Lợi nói chung và các thầy cô trong khoa Công nghệ thông tin nói riêng đã chỉ dạy nên em có thể đúc kết lại những kiến thức học được để hoàn thiện đồ án này.

Tiếp đó em xin gửi lời cảm ơn sâu sắc tới TS. Trương Xuân Nam, cảm ơn thầy đã hướng dẫn tận tình và góp ý cho em trong quá trình hoàn thiện đồ án này.

Với khả năng còn hạn chế trong đề tài được chọn không thể thiếu đi sự sai sót nhưng em tự tin rằng với những kĩ năng mình tích lũy được em đã có thể hoàn thiện đồ án này tốt nhất có thể.

Em xin chân thành cảm ơn!

# MỤC LỤC

[MỤC LỤC 3](#_Toc156000438)

[DANH MỤC HÌNH ẢNH 6](#_Toc156000439)

[DANH MỤC BẢNG BIỂU 8](#_Toc156000440)

[TÓM TẮT ĐỀ TÀI 9](#_Toc156000441)

[CHƯƠNG 1 : TỔNG QUÁT VỀ NGÀNH CÔNG NGHIỆP GAME 12](#_Toc156000442)

[1.1 Tổng quát 12](#_Toc156000443)

[1.2 Công cụ làm game 13](#_Toc156000444)

[1.2.1 Công cụ làm game là gì? 13](#_Toc156000445)

[1.2.2 Một số công cụ làm game 13](#_Toc156000446)

[CHƯƠNG 2 : CƠ SỞ LÝ THUYẾT 15](#_Toc156000447)

[2.1 Tổng quát về Unity 15](#_Toc156000448)

[2.1.1 Unity là gì? 15](#_Toc156000452)

[2.1.2 Lịch sử phát triển 15](#_Toc156000453)

[2.1.3 Lý do sử dụng Unity Engine 16](#_Toc156000454)

[2.1.4 Màn hình làm việc chính của Unity và các thuộc tính cơ bản của Unity 17](#_Toc156000455)

[2.1.5 Các khái niệm trong Unity 20](#_Toc156000456)

[2.2 Các công cụ hỗ trợ khác 29](#_Toc156000457)

[2.2.1 Visual Studio 29](#_Toc156000458)

[2.2.2 GIMP 29](#_Toc156000459)

[2.2.3 Github 29](#_Toc156000460)

[CHƯƠNG 3: LÊN Ý TƯỞNG VÀ THIẾT KẾ GAME 30](#_Toc156000461)

[3.1 Lên ý tưởng 30](#_Toc156000465)

[3.2 Thiết kế game 30](#_Toc156000466)

[3.2.1 Bản đồ 30](#_Toc156000467)

[3.2.2 Nhân vật chính 31](#_Toc156000468)

[3.2.3 Quái vật 31](#_Toc156000469)

[2.2.4 Hệ thống vũ khí 32](#_Toc156000470)

[3.2.4 Hệ thống cấp độ và nâng cấp 33](#_Toc156000471)

[3.2.5 Điều khiển 34](#_Toc156000472)

[3.2.6 Các hệ thống khác 34](#_Toc156000473)

[CHƯƠNG 4: PHÂN TÍCH HỆ THỐNG 35](#_Toc156000474)

[4.1 Phân tích các actor chính 35](#_Toc156000475)

[4.2 Xác định usercase và đặc tả 35](#_Toc156000476)

[4.2.1 Xác định usercase 35](#_Toc156000481)

[4.2.2. Đặc tả usercase 36](#_Toc156000482)

[4.2 Biểu đồ chức năng 42](#_Toc156000483)

[4.2.1 Biểu đồ usercase tổng quát 42](#_Toc156000484)

[4.2.2 Chức năng di chuyển 43](#_Toc156000485)

[4.2.3 Chức năng AI của quái vật 44](#_Toc156000486)

[4.2.4 Chức năng hệ thống vũ khí 46](#_Toc156000487)

[4.2.5 Chức năng lên cấp 47](#_Toc156000488)

[4.2.6 Chức năng nâng cấp 49](#_Toc156000489)

[4.2.7 Chức năng xử lý va chạm của người chơi 51](#_Toc156000490)

[4.2.8 Chức năng xử lý va chạm của quái vật 54](#_Toc156000491)

[CHƯƠNG 5: XÂY DỰNG GAME VÀ DEMO 56](#_Toc156000492)

[5.1 Xây dựng game 56](#_Toc156000493)

[5.2 Demo 56](#_Toc156000494)

[CHƯƠNG 6: KẾT LUẬN 60](#_Toc156000495)

[6.1 Kết luận 60](#_Toc156000496)

[6.2 Định hướng phát triển 60](#_Toc156000497)

[TÀI LIỆU THAM KHẢO 61](#_Toc156000498)

[CÁC ASSET SỬ DỤNG TRONG ĐỀ TÀI 62](#_Toc156000499)

# DANH MỤC HÌNH ẢNH

[Hình 1 Unity là gì? 14](#_Toc156000552)

[Hình 2 Màn hình làm việc chính của Unity 16](#_Toc156000553)

[Hình 3 Cửa sổ Hierachy 17](#_Toc156000554)

[Hình 4 Cửa số Scene 17](#_Toc156000555)

[Hình 5 Cửa sổ Game 18](#_Toc156000556)

[Hình 6 Cửa sổ Project 18](#_Toc156000557)

[Hình 7 Cửa sổ Inspector 19](#_Toc156000558)

[Hình 8 Ví dụ về một Scenes 20](#_Toc156000559)

[Hình 9 Ví dụ về một GameObject 21](#_Toc156000560)

[Hình 10 Ví dụ về Transform 22](#_Toc156000561)

[Hình 11 Minh họa Unity Flow 25](#_Toc156000562)

[Hình 12 Hình ảnh cấu trúc của bản đồ 29](#_Toc156000563)

[Hình 13 Sprite nhân vật chính 30](#_Toc156000564)

[Hình 14 Hình ảnh quái vật Dơi 31](#_Toc156000565)

[Hình 15 Ví dụ về vũ khí cận chiến 32](#_Toc156000566)

[Hình 16 Ví dụ về vũ khí tầm xa 32](#_Toc156000567)

[Hình 17 Màn hình lên cấp và hệ thống nâng cấp 33](#_Toc156000568)

[Hình 18 Biểu đồ usercase tổng quát 41](#_Toc156000569)

[Hình 19 Biểu đồ usercase di chuyển 42](#_Toc156000570)

[Hình 20 Biểu đồ hoạt động di chuyển 42](#_Toc156000571)

[Hình 21 Biểu đồ trình tự di chuyển 43](#_Toc156000572)

[Hình 22 Biểu đồ usercase AI của quái vật 43](#_Toc156000573)

[Hình 23 Biểu đồ hoạt động AI của quái vật 44](#_Toc156000574)

[Hình 24 Biểu đồ trình tự AI của quái vật 44](#_Toc156000575)

[Hình 25 Biểu đồ usercase hệ thống vũ khí 45](#_Toc156000576)

[Hình 26 Biểu đồ hoạt động của hệ thống vũ khí 45](#_Toc156000577)

[Hình 27 Biểu đồ trình tự hệ thống vũ khí 46](#_Toc156000578)

[Hình 28 Biểu đồ usercase lên cấp 46](#_Toc156000579)

[Hình 29 Biểu đồ hoạt động lên cấp 47](#_Toc156000580)

[Hình 30 Biểu đồ trình tự lên cấp 48](#_Toc156000581)

[Hình 31 Biểu đồ usercase nâng cấp 48](#_Toc156000582)

[Hình 32 Biểu đồ hoạt động nâng cấp 49](#_Toc156000583)

[Hình 33 Biểu đồ trình tự nâng cấp 50](#_Toc156000584)

[Hình 34 Biểu đồ usercase xử lý va chạm người chơi 50](#_Toc156000585)

[Hình 35 Biểu đồ hoạt động xử lý va chạm của người chơi 51](#_Toc156000586)

[Hình 36 Biểu đồ trình tự xử lý va chạm của người chơi 52](#_Toc156000587)

[Hình 37 Biểu đồ usercase xử lý va chạm của quái vật 53](#_Toc156000588)

[Hình 38 Biểu đồ hoạt động xử lý va chạm của quái vật 53](#_Toc156000589)

[Hình 39 Biểu đồ trình tự xử lý va chạm của quái vật 54](#_Toc156000590)

[Hình 40 Màn hình bắt đầu game 55](#_Toc156000591)

[Hình 41 Menu chọn trang bị khởi đầu 56](#_Toc156000592)

[Hình 42 Màn hình chính trong game 56](#_Toc156000593)

[Hình 43 Màn hình dừng game 57](#_Toc156000594)

[Hình 44 Màn hình lên cấp và nâng cấp 57](#_Toc156000595)

[Hình 45 Màn hình kết quả màn chơi 58](#_Toc156000596)

# DANH MỤC BẢNG BIỂU

[Bảng 1 Sơ đồ usercase di chuyển của người chơi 36](#_Toc156000545)

[Bảng 2 Usercase AI của quái vật 36](#_Toc156000546)

[Bảng 3 Usercase hệ thống vũ khí và nội tại 37](#_Toc156000547)

[Bảng 4 Usercase lên cấp 38](#_Toc156000548)

[Bảng 5 Usercase nâng cấp 39](#_Toc156000549)

[Bảng 6 Usercase xử lý va chạm của người chơi 39](#_Toc156000550)

[Bảng 7 Usercase xử lý va chạm của quái vật 40](#_Toc156000551)

|  |  |
| --- | --- |
| logo | TRƯỜNG ĐẠI HỌC THUỶ LỢI  **KHOA CÔNG NGHỆ THÔNG TIN**  BẢN TÓM TẮT ĐỀ CƯƠNG ĐỒ ÁN TỐT NGHIỆP |

TÊN ĐỀ TÀI: **Phát triển tựa game sinh tồn 2D qua công cụ Unity Engine**

Sinh viên thực hiện: Dương Giáp Đức

*Lớp:* 61TH1

*Mã sinh viên:* 1951060614

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

# TÓM TẮT ĐỀ TÀI

Vào thời kì phát triển của công nghệ, đi kèm với đó là những loại hình giải trí như trò chơi điện tử cũng qua đó càng ngày càng phát triển hơn. Nhắm vào thị trường đang trên đà phát triển đó để mang lại một tựa game có tính thử thách, có khả năng chơi lại cao và hơn hết là mang tính chất giải trí cho người chơi. Qua đó tựa game sinh tồn 2D được ra mắt để đáp ứng những tiêu chí trên đánh vào thị yếu những game thủ thích game sinh tồn.

Game sinh tồn 2D này có sự tương đồng và lấy ý tưởng từ tựa game cùng thể loại cực kỳ nổi tiếng qua một vài năm trở lại đây là Vampire Survivor, thay vì phát triển các lớp nhân vật có khởi đầu vũ khí khác nhau, tựa game sẽ chỉ có một nhân vật và khi bắt đầu được người chơi sẽ lựa chọn một trong các vũ khí khởi đầu, sinh tồn qua một khoảng thời gian nhất định, hệ thống giết kẻ địch để thăng cấp và nâng cấp các vũ khí, kĩ năng.

Về gameplay chính của tựa game, game sinh tồn 2D sẽ cho phép người chơi di chuyển theo 4 hướng, camera sẽ chạy theo nhân vật. Với một số vũ khí bắn thì sẽ được thiết kế tự động bắn theo hướng người chơi đang di chuyển mà không phải thông qua chuột, giết địch lên cấp và nâng cấp các vũ khí để tăng hiệu quả trong quá trình giết quái. Quái vật sẽ được sinh ra với số lượng nhất định, có thể giết hết chúng rồi hoàn thiện màn chơi hoặc sinh tồn trong một khoảng thời gian nhất định.

Hướng mà trò chơi hướng đến là các nâng cấp sẽ giúp người chơi mang lại sự thỏa mãn khi tiêu diệt các quái vật, mang đến cho người chơi một khoảng thời gian giải trí nhẹ nhàng và ngắn hạn.

Với những đặc điểm trên và muốn thử sức với một đề tài trong ngành công nghiệp mà bản thân cũng tiêu thụ rất nhiều, em đã chọn đề tài trên để nắm rõ hơn về ngành công nghiệp làm game nói chung và quy trình phát triển một tựa game.

**Công nghệ sử dụng**

* Unity Engine
* Visual Studio
* Github
* GIMP

**Các mục tiêu chính**

Mục tiêu:

* Nhân vật di chuyển theo 4 hướng, camera sẽ chạy nhân vật.
* Các vũ khí bắn sẽ được bắn theo hướng di chuyển của nhân vật.
* Xây dựng cơ chế tự sinh bản đồ khi người chơi di chuyển.
* Tự động ngẫu nhiên các đồ vật trên khung cảnh của một tấm map sinh ra để tránh sự trùng lặp.
* Tạo ra các đồ vật ở môi trường có thể tương tác và rơi ra vật phẩm.
* Tạo ra AI đơn giản cho các quái vật.
* Tạo ra hệ thống chỉ số cho người chơi và quái vật.
* Tạo ra hệ thống cấp độ.
* Tạo ra hệ thống vũ khí và nội tại.
* Tạo ra hệ thống nâng cấp vũ khí, nội tại khi lên cấp.
* Tạo ra các luồng dừng, thoát game, màn hình hiển thị các chỉ số và nâng cấp trong và sau khi hoàn thiện màn chơi (hoặc chết).

Kỹ năng:

* Sinh viên tìm hiểu về kiến thức C#, các chức năng Unity Engine.
* Sinh viên thành công xây dựng một demo với các tính năng trên.

**Kết quả dự kiến**

Kiến thức:

* Sinh viên tìm hiểu về các kiến thức của C#, Unity Engine.
* Sinh viên nắm được cơ bản quy trình phát triển một tựa game.

Kỹ năng:

* Báo cáo tổng hợp về công cụ Unity Engine.
* Sinh viên lập trình thành công demo dựa trên các bài toán đề ra bằng công Unity Engine với ngôn ngữ chủ đạo là C#.

**Bố cục của báo cáo**

* Chương 1: Tổng quát về ngành công nghiệp game
* Chương 2: Cơ sở lý thuyết
* Chương 3: Ý tưởng và hướng thiết kế game
* Chương 4: Phân tích hệ thống
* Chương 5: Xây dựng và demo các chức năng
* Chương 6: Kết luận.

# : TỔNG QUÁT VỀ NGÀNH CÔNG NGHIỆP GAME

## Tổng quát

Ngành công nghiệp game là một ngành lớn trong việc sản xuất, phát triển, phân phối và bán các tựa game trên nhiều nền tảng khác nhau, từ máy tính cá nhân, điện thoại di đông và các tựa máy chơi game điển hình như PlayStation, Xbox và Nintendo. Khởi nguồn của trò chơi điện tử đầu tiên xuất hiện vào năm 1947 khi các nhà khoa học và kỹ sư sử dụng các ống phóng tia âm cực để tạo ra các trò chơi đơn giản trên máy tính lớn [1]. Đến nay khi thời kỳ bùng nổ của internet và sự phát triển của các thiết bị chơi game tại gia khiến người chơi dần chuyển sang các thế loại game trực tuyến kết nối qua mạng để tương tác với những người chơi khác.

Hiện nay ngành công nghiệp game là một ngành công nghiệp có tốc độ tăng trưởng rất nhanh với giá trị doanh thu khoảng 300 tỷ USD tính đến năm 2023 và dự kiến sẽ còn tăng mạnh vào những năm kế tiếp. Thị trường game rất phát triển tại các nước như Hoa Kỳ, Nhật Bản, Trung Quốc, Hàn Quốc điểm qua là những công ty đứng đầu trong ngành phát triển game như Tencent (Trung Quốc), Nexon (Hàn Quốc), Nintendo (Nhật Bản), Ubisoft (Hoa Kỳ) và rất nhiều những tên tuổi lớn khác trong ngành.

Tuy nhiên, do dư luận xã hội còn giữ định kiến với trò chơi điện tử nói riêng nên sự phát triển của lĩnh vực này vẫn còn hạn chế ở Việt Nam. Chúng ta nên có cái nhìn khác về ngành game là một ngành sáng tạo, cần được phát triển phù hợp với xu hướng của thế giới và cũng rất phù hợp với thời kỳ chuyển đổi số hiện nay ở Việt Nam. Có thể thấy ngành game là một ngành công nghiệp mang lại lợi nhuận rất lớn không những vậy nó còn là một công cụ giúp bạn bè trên thế giới trải nghiệm được thông điệp, hình ảnh, văn hóa mà những nhà làm game gửi gắm vào trong tựa game của họ.

## Công cụ làm game



#### Công cụ làm game là gì?

Công cụ làm game là một phần mềm cho phép chúng ta tạo ra các trò chơi điện tử mà không cần phải viết code từ đầu. Công cụ làm game thường cung cấp cho các tính năng như:

* Một môi trường lập trình trực quan, cho phép kéo thả các đối tượng, hành động, sự kiện, … để xây dựng logic của trò chơi.
* Một thư viện các tài nguyên có sẵn, như hình ảnh, âm thanh, hiệu ứng, … có thể sử dụng trong trò chơi của mình.
* Một hệ thống xuất game, cho phép chuyển đổi trò chơi thành các định dạng khác nhau như web, di động, máy tính, … để có thể chia sẻ và chơi game trên các nền tảng khác nhau.

### Một số công cụ làm game

#### Godot

Là một phần mềm mã nguồn mở miễn phí viết bằng ngôn ngữ C+ ra mắt vào năm 2014, hỗ trợ làm game trên mọi nền tảng PC, Android, Web. Điều thú vị của Godot là việc viết code cho game sử dụng một ngôn ngữ tích hợp riêng của Godot được gọi là GDScript, có sự tương tự với ngôn ngữ lập trình python và là một ngôn ngữ bậc cao.

#### GameMaker Studio

GameMaker tiền thân là Animo, là một phần mềm đa nền tảng viết bằng ngôn ngữ C++ được tạo ra từ năm 1999. GameMaker là một công cụ chủ yếu để thiết kế ra các tựa game 2D, tuy nó có thể thiết kế các tựa game 3D nhưng còn rất nhiều hạn chế về mặt công nghệ. Là một phần mềm rất lâu đời nên nó có hỗ trợ làm game trên mọi nền tảng PC, Mobile, Web, Consoles. Ngôn ngữ lập trình của GameMaker là GameMaker Language (GML) có sự tương đồng với ngôn ngữ JavaScript và C.

#### RPG Maker.

RPG Maker là một phần mềm chuyên làm về game 2D Pixel Art xuất phát từ Nhật Bản vào năm 1992 bằng ngôn ngữ JavaScript. Như tên gọi của nó RPG viết tắt của Role-Playing Game hay còn gọi là game nhập vai phưu lưu giả tưởng có cốt truyện một thể loại game đến từ Nhật Bản. Hỗ trợ làm game trên các nền tảng chính PC, Console.

#### Unreal Engine:

Unreal Engine là một công cụ làm game được phát triển bởi Epic Game viết bằng ngôn ngữ C++, được giới thiệu lần đầu tiên vào năm 1998. Mục đích ban đầu của phần mềm là tạo ra các tựa game bắn súng góc nhìn thứ nhất. Một trong những công cụ cho ra đồ họa cực kỳ chất lượng, mã nguồn mở có thể truy cập được trên GitHub và sử dụng một mô hình tài chính người dùng phải trả tiền cho công cụ khi doanh thu game đạt được hơn 1 triệu đô la ngoài việc làm game Unreal Engine cũng được sử dụng trong phim ảnh ví dụ như The Mandalorian của Disney.

# : CƠ SỞ LÝ THUYẾT

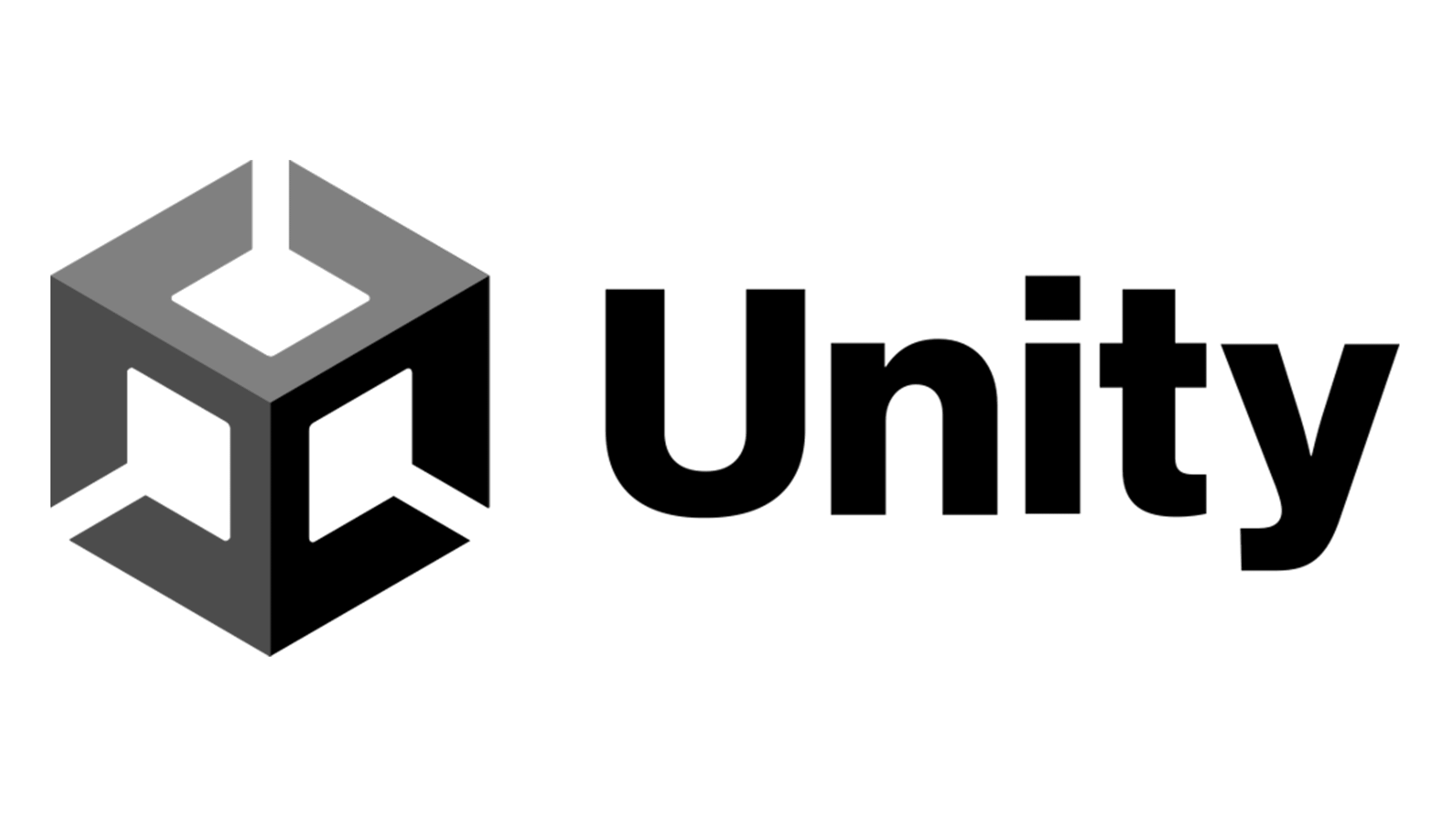
## Tổng quát về Unity



### Unity là gì?

Unity là một công cụ phát triển game đa nền tảng được phát triển bởi Unity Technologies. Game Engine này được sử dụng để phát triển game trên PC, consoles, thiết bị di động và trên website [2].

Người ta ước tính có hơn 50% số lượng game trên thị trường được tạo ra bởi Unity. Một vài tựa game vô cùng nổi tiếng được tạo ra bởi Unity có thể kể tên như: Among Us, Pokemon Go, HeartStone. Độ “phủ sóng” của Unity rất rộng, có thể được áp dụng phổ biến trong nhiều dòng game khác nhau từ những game đồ họa khủng cho đến những game giáo dục cho trẻ con [3].



Hình 1 Unity là gì?

### Lịch sử phát triển

Unity Technologies là công ty đứng phía sau Unity được thành lập vào năm 2004 bởi ba người sáng lập là Helgason, Joachim Ante và Nicholas Fancis. Ban đầu, họ tập trung vào phát triển giúp các nhà phát triển tạo ra trò chơi trên nền tảng MacOS.

Phiên bản đầu tiên của Unity là Unity 1.0 được phát hành vào năm 2005. Tuy nhiên, Unity không nhận được sự quan tâm lớn đến tư công đồng lập trình game thời điểm đó.

Sự thay đổi lớn nhất và cũng như quyết định lịch sử đến từ Unity khi họ quyết định chuyển đổi sang một mô hình kinh doanh miễn phí. Họ cung cấp Unity 3 miễn phí cho người dùng, điều nay tạo ra cơ hội cho nhiều nhà phát triển mới tham gia và đóng góp vào hệ sinh thái của Unity.

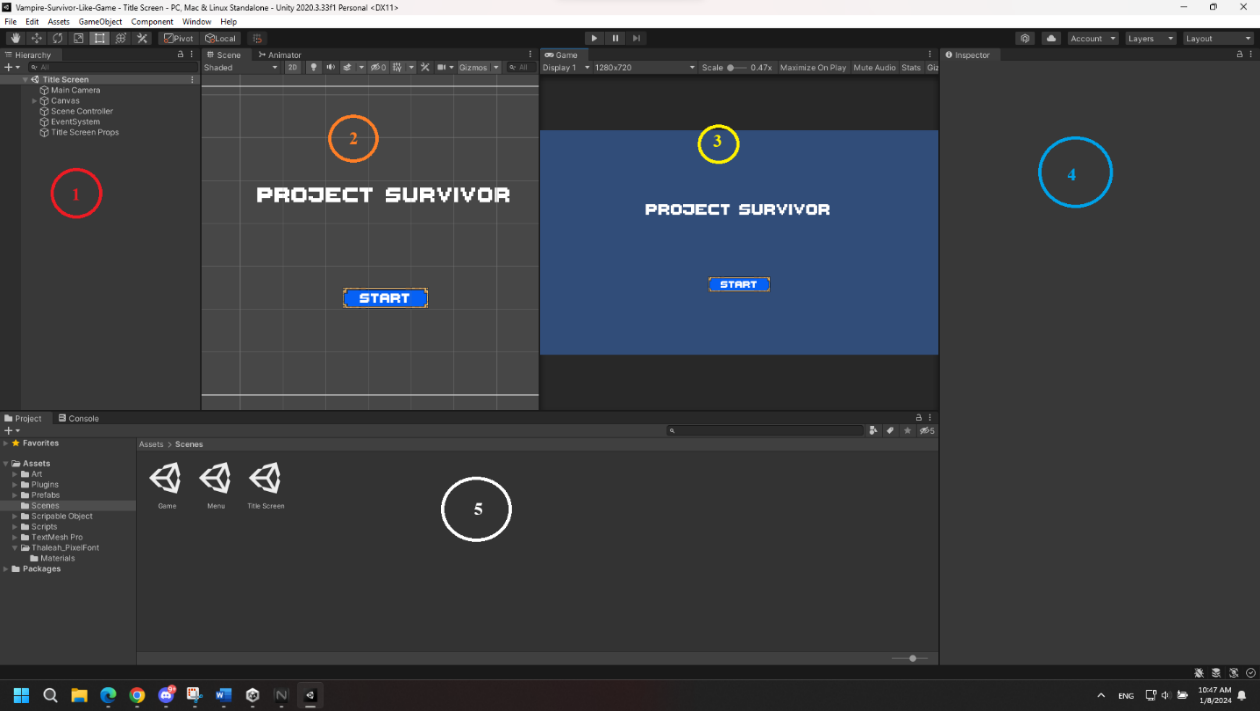
Sau khi thay đổi chiến lược, Unity phát triển nhanh chóng và liên tục ra mắt các phiên bản cải thiện hiệu suất và ra mắt các tính năng mới. Điều này giúp cho Unity trở thành một công cụ phát triển trò chơi hàng đầu thế giới.

### Lý do sử dụng Unity Engine

Sau khoảng thời gian dài phát triển Unity hiện nay là một công cụ ưu tiên hàng đầu của những nhà phát triển game và sau đây là những lý do chúng ta nên sử dụng nó:

* Có phiên bản miễn phí: Unity cung cấp miễn phí cho người dùng cá nhân và các doanh nghiệp có doanh thu dưới 100.000 USD/năm.
* Có thể tạo ra một tựa game nhiều người chơi: Tính năng nổi trội nhất của Unity chính là cho phép các lập trình viên có thể tạo ra một tựa game hỗ trợ nhiều người chơi cùng lúc. Ở Unity, bạn có thể tạo ra nhứng tựa game 3D với giao diện trực quan, và có bảng điều khiển dễ dàng. Do đó, công cụ Unity sẽ vô cùng phù hợp nếu bạn lên ý tưởng xây dựng mọto trò chơi có nhiều nút điều khiển phức tạp. Unity sẽ hỗ trợ lập trình game, biến những ý tưởng đó thành hiện thực theo cách đơn giản nhất [4].
* Cộng đồng lớn mạnh và các khóa học cho người mới bắt đầu: Unity có một cộng đồng người sử dụng đông đảo và số lượng những khóa học thông qua Unity Learn cũng rất nhiều giúp cho những nhà phát triển game dễ dàng bắt đầu và học hỏi những kiến thức liên quan về việc phát triển game.
* Hỗ trợ đa nền tảng: Unity là một công cụ đa nền tảng giúp tạo ra các trò chơi trên các nền tảng như PC, iOS, macOS, consoles, Android,…
* Dễ dàng sử dụng: Unity được xây dựng trong một môi trường phát triển tích hợp, cung cấp một hệ thống toàn diện cho lập trình viên, từ soạn thảo mã nguồn, xây dựng công cụ tự động hóa đến trình sửa lỗi. Do hướng đến mọi người dùng kể cả những người không chuyên về lập trình nên unity rất dễ để sử dụng.

### Màn hình làm việc chính của Unity



Hình 2 Màn hình làm việc chính của Unity

#### Cửa sổ Hierarchy

Hierarchy là nơi chứa mọi GameObject trong Scene hiện tại. Khi chúng ta thực hiện các tác vụ như thêm, sửa hoặc xóa trong Hierachy, thì các đối tượng trong Scene cũng sẽ thay đổi theo.

A screenshot of a computer menu

Description automatically generated

Hình 3 Cửa sổ Hierachy

#### Cửa sổ Scene

Trong cửa sổ Scene chúng ta có thể thay đổi góc nhìn hay bố trí các Game Object trên Scene. Cũng có thể sử dụng tool bar để thực hiện kéo thả, phóng to, thu nhỏ, xoay các đối tượng.

A screen shot of a video game

Description automatically generated

Hình 4 Cửa số Scene

#### Cửa sổ Game

Cửa sổ Game là màn hình game khi bắt đầu chạy, những thứ chúng ta thấy trong cửa sổ Game đều được lấy từ góc nhìn của Camera.

Thanh công cụ của cửa sổ game có thể thay đổi độ phân giải, hiển thị các thông số chi tiết liên quan đến hiệu năng trong quá trình chạy game.

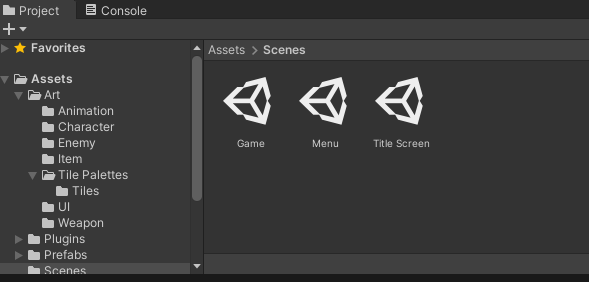
A screenshot of a video game

Description automatically generated

Hình 5 Cửa sổ Game

#### Cửa sổ Project

Đây là cửa sổ quản lý những tài nguyên hiện tại của project, giúp chúng ta có thể quản lý tài nguyên dễ dàng hơn hay tiện dụng trong việc thêm những tài nguyên mới bằng việc kéo thả thẳng vào trong những thư mục mong muốn.



Hình 6 Cửa sổ Project

#### Cửa sổ Inspector

Cửa sổ Inspector là nơi chứa mọi thông tin của một đối tượng mà hiện tại đang được lựa chọn. Chúng ta cũng có thể thay đổi nhưng thông tin hiện có của những đối tượng

A screenshot of a computer program

Description automatically generated

Hình 7 Cửa sổ Inspector

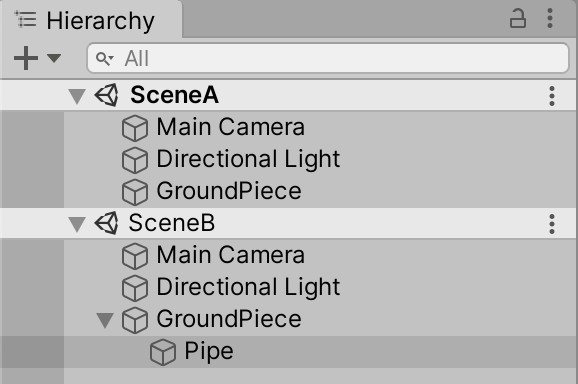
### Các khái niệm trong Unity

#### Asset

Asset là mọi tài nguyên phục vụ cho project bao gồm sprite, animaton, sound, script, scenes, …

#### Scene

Scene có thể được tính như một không gian làm việc, chúng ta có thể tạo một lúc nhiều scene để xây dựng và quản lý các tác vụ khác nhau hay mỗi một scene có thể tượng trưng một màn chơi riêng biệt.



Hình 8 Ví dụ về một Scenes

#### GameObject

GameObject là đơn vị cơ bản nhất trong Unity, đại diện cho mọi thứ trong trò chơi. Mọi đối tượng như nhân vật, hiệu ứng, ánh sáng,... đều được biểu diễn bằng GameObject

A screenshot of a computer

Description automatically generated

Hình 9 Ví dụ về một GameObject

#### Script

Script là một tập tin chứa các đoạn mã nguồn, dùng để khởi tạo và xử lý các đối tượng trong game. Trong Unity chúng ta có thể sử dụng C#, JavaScript, BOO để lập trình script.

#### Transform

Là lớp biểu hiện vị trí của 1 GameObject trên màn hình, nó chứa các thông số như Position (vị trí), Rotation (xoay), Scale (tỷ lệ) giúp chúng ta có thể xoay, phóng to thu nhỏ, di chuyển một đối tượng trên Scene.

A screenshot of a computer

Description automatically generated

Hình 10 Ví dụ về Transform

#### Camera

Là một GameObject đặc biệt khi tạo một project Unity. Camera có tác dụng như một máy quay chiết suất hình ảnh trên cửa sổ Scene và đưa nó qua cửa sổ Game dưới dạng góc nhìn 2D. Camera có các khái niệm cơ bản sau:

* Projection: Có 2 loại phép chiếu của camera Orthographic – phép chiếu song song, được sử dụng trong các game 2D, các đối tượng sẽ được vẽ đúng với tỉ lệ dù ở bất kỳ khoảng cách chiều sâu nào. Perspective – phép chiếu phối cảnh, được ứng dụng trong game 3D, mô phỏng lại thế giới thực của con người. Các vật thể càng xa thì tỷ lệ hình ảnh càng nhỏ và ngược lại [5].
* Field of view (fov): là góc mở của camera Perspective. Đối với Orthographic camera, tham số này được bỏ qua. Với cùng một đối tượng và vị trí, nếu fov càng lớn, đối tượng sẽ càng nhỏ và ngược lại [5].
* Near – Far: Near và Far là hai mặt phẳng cắt, quy định khoảng nhìn thấy của Camera tính theo chiều sâu. Một vật thể muốn được hiển thị phải nằm trong phạm vi nhìn thấy của Camera, bao gồm cả khoảng cách từ Camera đến vật đó. Nếu quá gần và quá xa, Camera sẽ không thu được hình ảnh để hiển thị trên màn hình [5].

#### Collider

Là thành phần của một GameObject để xác định những va chạm có tác dụng trong việc xử lý va chạm giữa các đối tượng khác với nhau.

#### Component

Component là những thành phần được gắn vào một GameObject để thêm những tính năng. Ví dụ như Transform: di chuyển, quay, thay đổi tỉ lệ, Rigidbody: giả lập vật lý, Sprite Renderer: quản lý và chỉnh sửa sprite.

#### Prefab

Prefab là một tính năng tạo GameObject tái sử dụng bằng cách kéo thả những GameObject trên Scene vào trong một thư mục asset của game. Mọi thay đổi ở prefab gốc sẽ ảnh hưởng đến mọi bản sao của nó trên Scene.

#### Animation

Là một công cụ giúp chúng ta có thể sử dụng những hình ảnh, mô hình nhân vật có sẵn thực hiện lại một tập hợp những động tác, tư thế khác nhau để mô phỏng lại những hành động chạy, nhảy, ngồi,...

#### MonoBehaviour

MonoBehaviour là một lớp cơ bản và mặc định của mọi script được tạo ra từ Unity, luôn được tạo với một template nhất định bao gồm lớp Start() và lớp Update(). MonoBehaviour có tác dụng thực thi những logic được lập trình bằng Script thông qua việc gắn vào những GameObject để tạo ra những chức năng khác nhau.

#### ScriptableObject

SriptableObject trong Unity là một loại đối tượng dùng để lưu trữ dữ liệu lớn, độc lập với các thể hiện của Class khác. Một trong những lý do chính trong việc sử dụng ScriptableObject là để giảm thiểu bộ nhớ của một project khi chúng ta có những Prefab lưu trữ dữ liệu không thay đổi và gắn trong MonoBehaviour script [6].

Khi chúng ta tạo ra Prefab chúng sẽ có những bản sao sở hữu y nguyên những giá trị khi nó được tạo ra. Thay vì thế chúng ta sử dụng ScriptableObject để lưu trữ những giá trị không thay đổi đó và dùng nó để tham chiếu đến mọi Prefab để chỉ duy nhất tồn tại một giá trị dữ liệu trong bộ nhớ [6].

Về cơ bản nó hoạt đông như MonoBehaviour là một đối tượng của Unity, nhưng không thể gắn những ScriptableObject vào thẳng một GameObject giống như một Script mà chúng sẽ được tạo ra dưới dạng những asset và được thêm vào dưới một dạng GameObject.

#### Singleton Pattern

Singleton Pattern là một mẫu thiết kế (Design Pattern), rất phổ biến trong việc thiết kế Game. Singleton Pattern cho phép bạn viết một lớp đối tượng duy nhất và có thể truy cập tất cả mọi nơi trong Game. Vì trong game có rất nhiều trạng thái cơ bản như: Start, Play, Pause, Stop, GameOver [7]. Để quản lý những trạng thái ấy chúng ta cần một cơ chế quản lý thích hợp để có thể cho chúng biết chúng ở trạng thái nào, chuyển trạng thái, cũng như những chức năng khác như quản lý điểm, quản lý những vật phẩm mua được trong game… [7].

Lý do lớn nhất để sử dụng singleton là để cho code trở lên rõ ràng. Ngoài hỗ trợ khái niệm một singleton cũng có thể là một phương tiện hiệu quả của việc cho phép truy cập global của một đối tượng. Các mẫu thiết kế singleton tạo điều kiện truy cập global đến một đối tượng trong khi đảm bảo rằng chỉ có một thể hiện của đối tượng tồn tại trên cùng một thời điểm. Nếu một thể hiện của Singleton không tồn tại khi nó được tham chiếu, nó sẽ được khởi tạo [7].

#### Unity Flow

Trong Unity, có thứ tự của những câu lệnh để xem ai được thực hiện trước và sau. Trình tự của chúng được biểu hiện qua biểu đồ sau:

A screenshot of a computer

Description automatically generated

Hình 11 Minh họa Unity Flow

Có thể thấy ở biểu đồ những hàm trên có quy trình thực hiện từ trên xuống dưới nhưng bình thường chúng ta cũng không sử dụng hầu hết chúng mà chỉ sử dụng một số những hàm sau đây:

Đầu tiên là những sự kiện khởi tạo (Initialization):

* Hàm Awake được gọi trước hàm Start khi một GameObject được sinh ra.
* Hàm OnEnable được sử dụng để tương tác khi một GameObject ở trạng thái hoạt động, tương tự cũng có OnDisable khi GameObject đang trong trạng thái không hoạt động.
* Hàm Start được gọi ngay sau khi game bắt đầu chạy.

Hàm Awake sẽ kết thúc khi hàm Start đầu tiên được gọi thế nên trong hàm Start chúng ta có thể lấy những giá trị đã được Awake trước đó.

Tiếp theo đó những sự kiện cập nhật thường xuyên: Gồm những hàm như FixedUpdate, Update, LateUpdate.

* Hàm FixedUpdate thường được gọi thường xuyên hơn Update và có thể được gọi nhiều lần trong một khung hình. Tất cả những cập nhật vật lý hay những tính năng tính toán sẽ được thực thi ngay sau khi câu lệnh được chạy. Thường được sử dụng để làm chuyển động cho người chơi [8].
* Hàm Update: Được gọi trên mỗi khung hình trong game để cập nhật những chức năng của nó [8].
* Hàm LateUpdate: Cũng như update nó được gọi trên mỗi khung hình ở game và được gọi ngay sau khi update hoàn thành. Vậy nên những tính toán ở hàm Update có thể sử dụng trong LateUpdate. Thường LateUpdate được sử dụng để thay đổi góc nhìn của Camera bám sát theo người chơi vì ngay sau khi người chơi di chuyển thì tọa độ đó sẽ được trực tiếp chuyển qua Camera [8].

Ngoài ra còn có Couroutines trong Unity là một hàm cho phép tạm dừng thực thi và tiếp tục tại điểm dừng đó sau khi một điều kiện được thỏa mãn. Chúng ta có thể bắt đầu một coroutine và chờ kết quả của nó trước khi mã chuyển sang dòng tiếp theo. Để sử dụng coroutine chúng ta sẽ khai báo một hàm với kiểu trả về là IEnumerator và có một câu lệnh yield return nằm trong phần thân hàm. Câu lệnh yield return là điểm mà thực thi tạm dừng và tiếp tục trong khung hình kế tiếp hoặc ngay sau khi một điều kiện được đáp ứng. Để bắt đầu một couroutine, chúng ta cần sử dụng hàm StartCouroutine [9].

Cuối cùng là những hàm sự kiện dùng để tạo ra những tính năng khi va chạm giữa các GameObject với nhau, được sử dụng phổ biến là OnCollision, OnTrigger.

* OnCollision: Sau khi những GameObject được gắn component collider tương ứng. Trong quá trình thực hiện những chuyển động chúng sẽ không thể đi xuyên qua nhau mà sẽ chặn nhau khi những vòng bao collider tiếp xúc với nhau. Và trong những quá trình tiếp xúc đó chúng ta cũng có những hàm để biểu hiện những quá trình đó như: OnCollisionEnter – được gọi khi 2 collider chạm vào nhau lần đầu tiên, OnCollisionStay- được gọi sau khi 2 collider chạm lần đầu tiên và giữ nguyên vị trị đó, OnCollisionExit – được gọi ngay sau khi 2 collider dừng chạm vào nhau [10].
* OnTrigger: Tương tự như OnCollision những GameObject được gắn component collider và bật tính năng isTrigger trong inspector thì mới sử dụng được những câu lệnh OnTrigger. Khác với OnCollision những collider tương tác sẽ chặn nhau trong quá trình di chuyển thì OnTrigger sẽ đi xuyên qua nhau. Tương tự như OnCollider, OnTrigger cũng có những có những lệnh để gọi khi chúng tiếp xúc với nó như: OnTriggerEnter – được gọi khi 2 collider chạm vào nhau lần đầu tiên, OnTriggerStay- được gọi sau khi 2 collider chạm lần đầu tiên và giữ nguyên vị trí đó, OnTriggerExit – được gọi ngay sau khi 2 collider dừng chạm vào nhau [11].

## Các công cụ hỗ trợ khác

### Visual Studio

Là một chương trình biên dịch code đến từ Microsoft bao gồm tất cả các chức năng chính như viết code, build, debug code, test, version control, collaborate, deploy. Với số lượng lớn ngôn ngữ tích hợp bao gồm C, C++, .NET, C#,… và các ngôn ngữ khác như python, ruby, node js dưới dạng các plugin hỗ trợ bên ngoài.

### GIMP

Là một chương trình chỉnh sửa hình ảnh miễn phí mã nguồn mở có thể truy cập trên github, thực hiện được các chức năng chỉnh sửa ảnh, vẽ tự do, chuyển đổi các định dạng hình ảnh khác nhau và các tác vụ chuyên biệt hơn. Có thể dễ dàng cài đặt thông qua microsoft store ở trên window.

### Github

Github là một dịch vụ cung cấp kho lưu trữ code trên web cho các dự án phần mềm và cũng như theo dõi sự thay đổi, quản lý code. Github cung cấp dịch vụ bao gồm miễn phí và trả phí. Với hơn 100 triệu người dùng khẳng định được vị thế của bản thân trong giới công nghệ là một trong những công cụ lưu trữ, quản lý code hàng đầu thế giới và dễ dàng sử dụng thông qua phiên bản Github Desktop khiến việc thao tác không cần phải thông qua command prompt.

# CHƯƠNG 3: LÊN Ý TƯỞNG VÀ THIẾT KẾ GAME



## Lên ý tưởng

Trong thể loại game sinh tồn 2D hiện nay nổi lên một tựa game rất nổi tiếng là Vampire Survivor với lối chơi rất đơn giản và không hề phức tạp, mục tiêu chính của người chơi là sống sót và tiêu diệt con trùm của bản đồ đó để hoàn thành màn chơi. Trò chơi mang đến hệ thống vũ khí và nội tại, lên cấp khi tiêu diệt kẻ thù, tiêu diệt quái lớn sẽ rơi các rương vật phẩm ngẫu nhiên và tiền để nâng cấp các kỹ năng ngoài màn chơi để tăng sức mạnh ở các lần chơi tiếp theo. Vì vậy bản thân trò chơi có giá trị chơi lại rất cao vì sở hữu số lượng lớn các vũ khí thú vị, các con quái trông có vẻ dễ dàng tiêu diệt nhưng lại rất thử thách khả năng sinh tồn của người chơi vì càng sinh tồn lâu số lượng chúng càng tăng lên [12].

## Thiết kế game

### Bản đồ

A screenshot of a video game

Description automatically generated

Hình 12 Hình ảnh cấu trúc của bản đồ

Bản đồ trong game được cấu tạo từ 1 tấm pixel với chiều dài và rộng là 20x20. Tấm bản đồ trong game có 8 hướng trên, dưới, trái, phải, trên trái, dưới trái, trên phải, dưới phải. Ý tưởng của việc tạo ra 8 hướng như trên là để tạo ra các tấm bản đồ tiếp theo theo hướng di chuyển sắp tới của người chơi. Và bản đồ cũng tích hợp khả năng có thể bật tắt khi người chơi cách xa một khoảng cách nhất định để tăng hiệu năng chơi.

### Nhân vật chính

Nhân vật chính của trò chơi của chúng ta sẽ khởi đầu với 1 trong 2 vũ khí được cho sẵn, nhân vật chính được thiết kế bao gồm có vũ khí khởi đầu, tổng máu tối đa, hồi máu trên giây, tốc độ di chuyển, hệ số sát thương và bán kính hút vật phẩm.

A screenshot of a game

Description automatically generated

Hình 13 Sprite nhân vật chính

### Quái vật

Quái vật trong game được thiết kế với ý tưởng đơn giản sẽ tự khóa mục tiêu vào người chơi sau khi được sinh ra và liên tục di chuyển tới người chơi, gây sát thương khi chạm vào người chơi, mỗi một quái thì lại có chỉ số khác nhau. Khi bị tiêu diệt quái sẽ rơi ra kim cương kinh nghiệm để giúp người chơi lên cấp.

A screenshot of a video game

Description automatically generated

Hình 14 Hình ảnh quái vật Dơi

### Hệ thống vũ khí

Hệ thống vũ khí trong game hoàn toàn tự động và không cần người chơi phải thao tác bất cứ gì ngoài di chuyển nhân vật. Ý tưởng của hệ thống vũ khí bao gồm 3 loại: vũ khí cận chiến, vũ khí tầm xa và nội tại.

* Vũ khí cận chiến là những vũ khí có tầm thi triển gần thường sẽ chỉ bao quanh nhân vật chính một khoảng nhất định.

A video game screen with a cartoon character

Description automatically generated

Hình 15 Ví dụ về vũ khí cận chiến

* Vũ khí tầm xa là những vũ khi có khả năng phóng từ vị trí người chơi đang di chuyển đến.

A video game screen with a green background

Description automatically generated

Hình 16 Ví dụ về vũ khí tầm xa

* Nội tại là những nâng cấp có thể có được sau khi lên cấp qua menu nâng cấp. Tác dụng chính của nội tại là thay đổi những chỉ số hiện có của người chơi để tăng tiến sức mạnh sau mỗi lần lên cấp.

### Hệ thống cấp độ và nâng cấp

Hệ thống cấp độ sẽ giúp tăng tiến sức mạnh của người chơi trong quá trình chơi game, khi lên cấp người chơi sẽ được lựa chọn một trong số những vũ khí hoặc nội tại chưa sở hữu hoặc nâng cấp các vật phẩm hiện có của bản thân. Khi đang trong tình trạng lên cấp toàn bộ trò chơi sẽ ngưng lại cho đến khi người chơi chọn cho mình nâng cấp.



Hình 17 Màn hình lên cấp và hệ thống nâng cấp

### Điều khiển

Điều khiển nhân vật cơ bản di chuyển với 4 hướng thông qua các nút w, a, s, d. Các vũ khí tầm xa sẽ tự động bắn trong 1 khoảng thời gian nhất định tại hướng mà người chơi đang di chuyển. Và sẽ có những hoạt ảnh di chuyển ứng với vị trị hiện tại mà người chơi đang hướng đến.

### Các hệ thống khác

Ngoài các hệ thống trên còn các hệ thống khác cho phép người chơi dừng lại trong quá trình chơi game, thống kê các chỉ số và vật phẩm hiện có của nhân vật, thoát game ra màn hình chính, bắt đầu một lần chơi mới, hệ thống tự sinh quái, tự hồi máu, hiển thị sát thương khi đánh quái và các hiệu ứng khi nhận sát thương của người chơi và quái vật.

# CHƯƠNG 4: PHÂN TÍCH HỆ THỐNG

## Phân tích các actor chính

Các actor chính của game bao gồm:

* Người chơi (player): là nhân vật chính do người chơi điều khiển thực hiện các thao tác với môi trường của game và với quái vật
* Quái vật: là các nhân vật có AI đơn giản điều khiển di chuyển luôn luôn hướng tới người chơi và gây sát thương cho người chơi.

## Xác định usercase và đặc tả



### Xác định usercase

Hệ thống có các usercase sau:

* Di chuyển của người chơi: Di chuyển bằng 4 hướng theo các phím w, a, s, d.
* AI của quái vật: Luôn luôn di chuyển đến vị trí người chơi.
* Hệ thống vũ khí và nội tại: Người chơi có một hệ thống vũ khí có chức năng khác nhau và các nội tại để gia tăng các chỉ số.
* Lên cấp: Lên cấp khi thu thập đủ lượng kinh nghiệm từ đá quý.
* Nâng cấp: Khi người chơi lên cấp sẽ xuất hiện bảng nâng cấp các vũ khi, nội tại những vũ khí và nội tại chưa sở hữu sẽ được thêm vào hành trang sau khi lựa chọn.
* Xử lý va chạm của người chơi: Người chơi sẽ bị gây sát thương khi va chạm phải quái vật
* Xử lý va chạm của quái vật: Quái vật sẽ bị gây sát thương khi chạm phải vũ khí từ người chơi.

### Đặc tả usercase

Bảng 1 Sơ đồ usercase di chuyển của người chơi

|  |  |
| --- | --- |
| **Mô tả** | Di chuyển của người chơi |
| **Tác nhân** | Người chơi |
| **Tiền điều kiện** | Màn chơi bắt đầu |
| **Hậu điều kiện** | Nhân vật bắt đầu di chuyển theo các hướng ấn ở bàn phím |
| **Luồng sự kiện** | 1. Người chơi ấn các phím w, a, s, d để di chuyển nhân vật. 2. Xử lý di chuyển của nhân vật 3. Hệ thống bắt đầu thực hiện hoạt ảnh di chuyển của nhân vật. |
| **Luồng ngoại lệ** | 2a. Hệ thống sẽ thay đổi hoạt ảnh tương ứng với việc đứng im và di chuyển.  2b. Khi người chơi chết thì sẽ không thể di chuyển được nữa. |

Bảng 2 Usercase AI của quái vật

|  |  |
| --- | --- |
| **Mô tả** | AI của quái vật |
| **Tác nhân** | Quái vật |
| **Tiền điều kiện** | Khi quái vật được sinh ra |
| **Hậu điều kiện** | Quái vật di chuyển đến vị trí của người chơi |
| **Luồng sự kiện** | 1. Quái vật được sinh ra. 2. Xử lý di chuyển của quái vật. 3. Hệ thống bắt đầu thực hiện hoạt ảnh di chuyển của quái vật. |
| **Luồng ngoại lệ** | Không có |

Bảng 3 Usercase hệ thống vũ khí và nội tại

|  |  |
| --- | --- |
| **Mô tả** | Hệ thống vũ khí và nội tại |
| **Tác nhân** | Người chơi |
| **Tiền điều kiện** | Khi vũ khí và nội tại được người chơi sở hữu. |
| **Hậu điều kiện** | Thực hiện các chức năng của vũ khí và nội tại. |
| **Luồng sự kiện** | 1. Khi vũ khí và nội tại được sở hữu bởi người chơi. 2. Hệ thống thực hiện các chức năng của vũ khí và nội tại. |
| **Luồng ngoại lệ** | 2a. Nếu vũ khí được sở hữu bởi người chơi sẽ thực hiện các chức năng của các vũ khí đó.  2b. Nếu nội tại được sở hữu bởi người chơi sẽ thay đổi các chỉ số của người chơi. |

Bảng 4 Usercase lên cấp

|  |  |
| --- | --- |
| **Mô tả** | Lên cấp |
| **Tác nhân** | Người chơi |
| **Tiền điều kiện** | Thu thập các viên đá quý kinh nghiệm |
| **Hậu điều kiện** | Lên cấp |
| **Luồng sự kiện** | 1. Thu thập đá quý nhận kinh nghiệm. 2. Hệ thống kiểm tra lượng kinh nghiệm nhận được. 3. Lên cấp. |
| **Luồng ngoại lệ** | Không có |

Bảng 5 Usercase nâng cấp

|  |  |
| --- | --- |
| **Mô tả** | Nâng cấp |
| **Tác nhân** | Người chơi |
| **Tiền điều kiện** | Lên cấp |
| **Hậu điều kiện** | Chọn nâng cấp |
| **Luồng sự kiện** | 1. Người chơi lên cấp. 2. Bảng nâng cấp xuất hiện. 3. Người chơi chọn nâng cấp. |
| **Luồng ngoại lệ** | 3a. Nâng cấp chưa được sở hữu sẽ được thêm vào hành trang còn nâng cấp đã sở hữu sẽ được thay thế cấp độ tiếp theo. |

Bảng 6 Usercase xử lý va chạm của người chơi

|  |  |
| --- | --- |
| **Mô tả** | Xử lý va chạm của người chơi |
| **Tác nhân** | Người chơi |
| **Tiền điều kiện** | Khi quái vật, vật phẩm chạm vào người chơi |
| **Hậu điều kiện** | Thực hiện các tương tác giữa người chơi và các vật thể va chạm |
| **Luồng sự kiện** | 1. Người chơi chạm phải vật thể. 2. Hệ thống kiểm tra các tương tác giữa người chơi và vật thể va chạm. 3. Thực hiện các tương tác với nhau. |
| **Luồng ngoại lệ** | 1a. Người chơi có thể va chạm được với quái vật, vật phẩm.  3a. Khi người chơi va chạm phải quái vật sẽ bị trừ máu, khi máu tụt về 0 thì màn chơi kết thúc.  3b. Sẽ có một khoảng dừng nhỏ giữa các lần bị khi người chơi bị gây sát thương.  3c. Người chơi nhận được kinh nghiệm khi tiếp xúc với đá quý kinh nghiệm.  4d. Người chơi hồi lượng máu hiện tại khi tiếp xúc với bình máu. |

Bảng 7 Usercase xử lý va chạm của quái vật

|  |  |
| --- | --- |
| **Mô tả** | Xử lý va chạm của quái vật |
| **Tác nhân** | Quái vật, vũ khí |
| **Tiền điều kiện** | Quái vật chạm phải các vũ khí |
| **Hậu điều kiện** | Quái vật bị gây sát thương |
| **Luồng sự kiện** | 1. Quái vật chạm phải vũ khí. 2. Hệ thống kiểm tra sát thương của vũ khí chạm phải quái vật. 3. Quái vật bị trừ máu. |
| **Luồng ngoại lệ** | 3a. Quái vật sẽ có một khoảng thời gian ngắn không nhận sát thương khi vừa bị gây sát thương.  3b. Nếu quái vật bị trừ hết máu thì sẽ chết và rơi ra đá quý kinh nghiệm. |

## Biểu đồ chức năng

### Biểu đồ usercase tổng quát

A diagram of a person with text

Description automatically generated

Hình 18 Biểu đồ usercase tổng quát

### Chức năng di chuyển

A diagram of a person with a diagram

Description automatically generated

Hình 19 Biểu đồ usercase di chuyển

A screenshot of a diagram

Description automatically generated

Hình 20 Biểu đồ hoạt động di chuyển

A diagram of a diagram

Description automatically generated

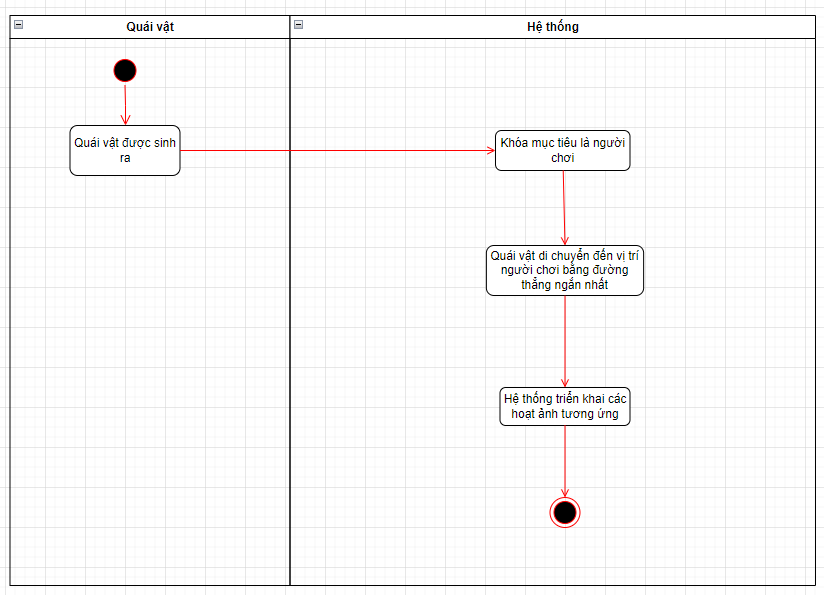
Hình 21 Biểu đồ trình tự di chuyển

### Chức năng AI của quái vật

A diagram of a person with a figure and a person with a person with a person with a person with a person with a person with a person with a person with a person with a person with

Description automatically generated

Hình 22 Biểu đồ usercase AI của quái vật



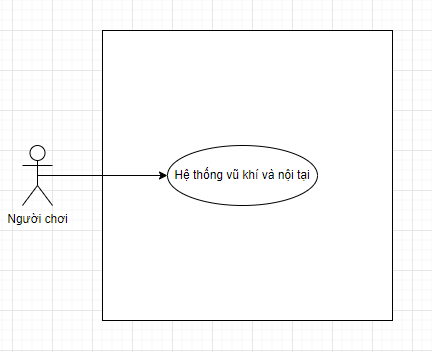
Hình 23 Biểu đồ hoạt động AI của quái vật

A diagram of a diagram

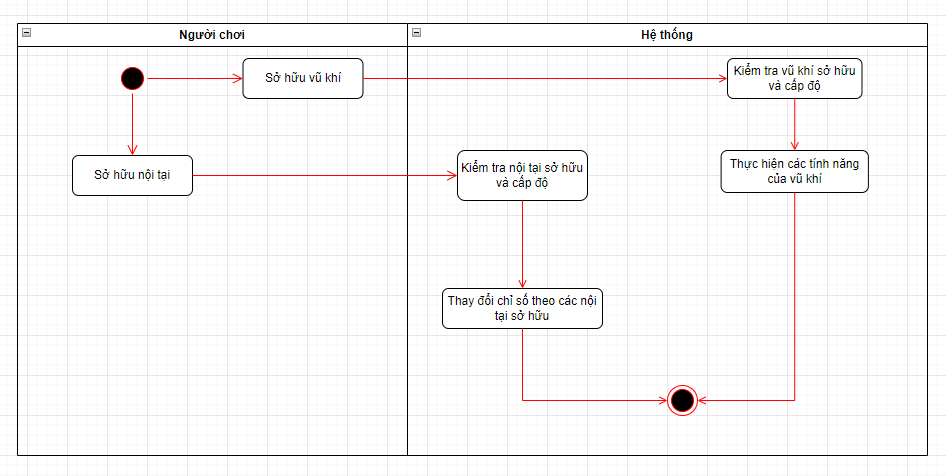
Description automatically generated

Hình 24 Biểu đồ trình tự AI của quái vật

### Chức năng hệ thống vũ khí



Hình 25 Biểu đồ usercase hệ thống vũ khí



Hình 26 Biểu đồ hoạt động của hệ thống vũ khí

A diagram of a diagram

Description automatically generated with medium confidence

Hình 27 Biểu đồ trình tự hệ thống vũ khí

### Chức năng lên cấp

A diagram of a person with a person in the middle

Description automatically generated

Hình 28 Biểu đồ usercase lên cấp

A diagram of a diagram

Description automatically generated with medium confidence

Hình 29 Biểu đồ hoạt động lên cấp

A graph with text and a line

Description automatically generated with medium confidence

Hình 30 Biểu đồ trình tự lên cấp

### Chức năng nâng cấp

A diagram of a person with a stick figure

Description automatically generated

Hình 31 Biểu đồ usercase nâng cấp

A diagram with red lines and black dots

Description automatically generated

Hình 32 Biểu đồ hoạt động nâng cấp

A screenshot of a graph

Description automatically generated

Hình 33 Biểu đồ trình tự nâng cấp

### Chức năng xử lý va chạm của người chơi

A diagram of a person with a figure

Description automatically generated

Hình 34 Biểu đồ usercase xử lý va chạm người chơi

A diagram of a flowchart

Description automatically generated

Hình 35 Biểu đồ hoạt động xử lý va chạm của người chơi

A screenshot of a graph

Description automatically generated

Hình 36 Biểu đồ trình tự xử lý va chạm của người chơi

### Chức năng xử lý va chạm của quái vật

A diagram of a person with a diagram

Description automatically generated

Hình 37 Biểu đồ usercase xử lý va chạm của quái vật

A screenshot of a diagram

Description automatically generated

Hình 38 Biểu đồ hoạt động xử lý va chạm của quái vật

A grid with lines and words

Description automatically generated with medium confidence

Hình 39 Biểu đồ trình tự xử lý va chạm của quái vật

# CHƯƠNG 5: XÂY DỰNG GAME VÀ DEMO

## Xây dựng game

Link demo: <https://github.com/Duk1321/DATN_1951060614_DuongGiapDuc>

## Demo



Hình 40 Màn hình bắt đầu game

A screenshot of a video game

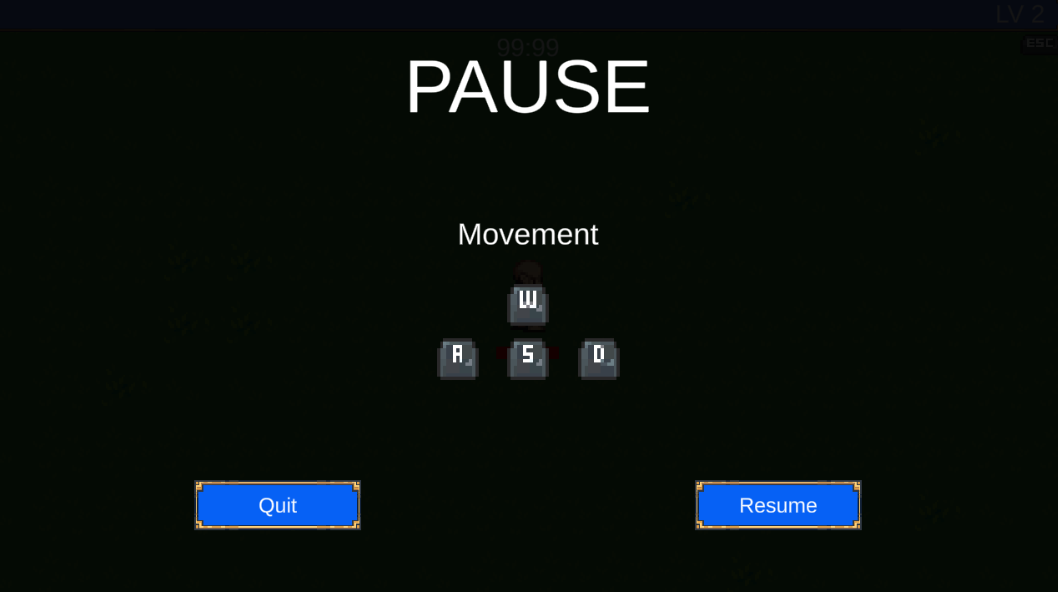
Description automatically generated

Hình 41 Menu chọn trang bị khởi đầu

A screenshot of a video game

Description automatically generated

Hình 42 Màn hình chính trong game



Hình 43 Màn hình dừng game

A screenshot of a video game

Description automatically generated

Hình 44 Màn hình lên cấp và nâng cấp

A screenshot of a video game

Description automatically generated

Hình 45 Màn hình kết quả màn chơi

# CHƯƠNG 6: KẾT LUẬN

## Kết luận

Qua việc thực hiện báo cáo này và làm một bản demo em đã được tiếp xúc một phần gần hơn với việc làm game, học cách viết những dòng lệnh unity đầu tiên, làm quen với những vật thể trong game, học những phương pháp tiếp cận để giải quyết bài toán được đặt ra. Và em cũng nhận ra rằng việc làm game là một việc đòi hỏi rất cao về khả năng sáng tạo, kiên trì. Dù chỉ những chức năng kể trên đã hoạt động theo mô tả nhưng khả năng tái sử dụng khá thấp và em nghĩ em cần thêm nhiều thời gian hơn để có thể thực sự tìm ra một lối đi rõ ràng hơn, qua đó cũng thấy rằng để hoàn thành một game hoàn chỉnh cũng cần phải có một đội ngũ những người đảm nhận những vai trò riêng trong quy trình làm một tựa game để mang đến một sản phẩm thực thụ và chỉnh chu đến người chơi.

## Định hướng phát triển

Dù demo đã hoàn thiện nhưng em cũng đã bóc tách ra được những thứ sau đây để khiến game hoàn chỉnh hơn:

* Tạo những con quái vật mới để trò chơi thêm đa dạng.
* Thêm những nâng cấp và nội tại để khiến game có giá trị chơi lại nhiều hơn.
* Thiết kế thêm những màn chơi mới, bản đồ mới.
* Thêm hiệu ứng âm thanh cho trỏ chơi.
* Chạy tựa game trên hệ điều hành điện thoại.
* Hoàn thiện hệ thống vũ khí và nội tại để dễ dàng hơn cho việc phát triển thêm những vũ khí và nội tại mới.
* Hoàn thiện giao diện người dùng.

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | Wikipedia, "Lịch sử video game," 21 4 2023. [Online]. Available: https://vi.wikipedia.org/wiki/L%E1%BB%8Bch\_s%E1%BB%AD\_video\_game. |
| [2] | T. X. Chu, "Giới thiệu về Unity Engine - Game Engine phổ biến nhất hiện nay.," 17 8 2018. [Online]. |
| [3] | T. D. Huy, "Unity là gì? Unity có quan trọng với sự nghiệp của Game Developer không?," [Online]. Available: https://itviec.com/blog/unity-la-gi/. |
| [4] | Đ. T. K. Ngân, "Có nên lập trình game bằng phần mềm Unity," 4 3 2023. [Online]. Available: https://funix.edu.vn/chia-se-kien-thuc/co-nen-lap-trinh-game/. |
| [5] | R. Nguyen, "Cơ Bản về Camera trong Unity," 24 9 2020. [Online]. Available: https://www.iostream.co/article/co-ban-ve-camera-trong-unity-p10g3U. |
| [6] | Unity, "ScriptableObject," 10 15 2018. [Online]. Available: https://docs.unity3d.com/2020.3/Documentation/Manual/class-ScriptableObject.html. |
| [7] | D. Nguyen, "Hướng dẫn sử dụng Singleton để quản lý game trong Unity3D," 24 3 2016. [Online]. Available: https://viblo.asia/p/huong-dan-su-dung-singleton-de-quan-ly-game-trong-unity3d-l0rvmxwxGyqA. |
| [8] | Unity, "Order of execution for event functions," 5 1 2024. [Online]. Available: https://docs.unity3d.com/Manual/ExecutionOrder.html. |
| [9] | Unity, "Couroutines," 28 4 2023. [Online]. Available: https://docs.unity3d.com/Manual/Coroutines.html. |
| [10] | Unity, "OnCollision Events," 5 1 2024. [Online]. Available: https://docs.unity3d.com/Manual/collider-interactions-oncollision.html. |
| [11] | Unity, "OnTrigger Event," 5 1 2024. [Online]. Available: https://docs.unity3d.com/Manual/collider-interactions-ontrigger.html. |
| [12] | Wikipedia, "Vampire Survivor," 17 1 2022. [Online]. Available: https://vampire-survivors.fandom.com/wiki/Vampire\_Survivors. |

# CÁC ASSET SỬ DỤNG TRONG ĐỀ TÀI

<https://seliel-the-shaper.itch.io/character-base>

<https://opengameart.org/content/bat-sprite>

<https://opengameart.org/content/lpc-terrain-repack>

<https://opengameart.org/content/admurins-armory>

<https://opengameart.org/content/admurins-potions>

<https://opengameart.org/content/zelda-like-tilesets-and-sprites>

<https://opengameart.org/content/gem-icons>