**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

**KHOA CÔNG NGHỆ THÔNG TIN**

**======\*\*\*======**



**BÁO CÁO THÍ NGHIỆM THỰC NGHIỆM**

**PHÁT TRIỂN ỨNG DỤNG GAME**

**ĐỀ TÀI: XÂY DỰNG GAME ROBOT SURVIVAL**

|  |  |
| --- | --- |
| **Giáo viên hướng dẫn** | **: ThS. Nguyễn Văn Tỉnh** |
| **Nhóm – Lớp** | **: Nhóm 1 – 20241IT6028001** |
| **Sinh viên thực hiện** | **: Dương Đình Trung – 2021602817**  **Nguyễn Việt Anh – 2021600938** |
|  |
|  |
|  |

*Hà Nội – Năm 2024*

# MỤC LỤC

[MỤC LỤC 2](#_Toc185877018)

[DANH MỤC HÌNH ẢNH 5](#_Toc185877019)

[CHƯƠNG 1 GIỚI THIỆU PHẦN MỀM UNITY 7](#_Toc185877020)

[1.1 Giới thiệu chung 7](#_Toc185877021)

[1.1.1 Khái niệm cơ bản về Unity 7](#_Toc185877022)

[1.1.2 Các khái niệm cơ bản trong Unity 7](#_Toc185877023)

[1.2 Giao diện làm việc và các chức năng chính 11](#_Toc185877024)

[1.2.1 Cửa sổ Scenes 12](#_Toc185877025)

[1.2.2 Cửa sổ Hierarchy 12](#_Toc185877026)

[1.2.3 Cửa sổ Game 13](#_Toc185877027)

[1.2.4 Cửa sổ Project 13](#_Toc185877028)

[1.2.5 Cửa sổ Inspector 14](#_Toc185877029)

[1.2.6 Top bar 15](#_Toc185877030)

[CHƯƠNG 2. XÂY DỰNG GAME 16](#_Toc185877031)

[2.1 Giới thiệu tổng quan 16](#_Toc185877032)

[2.1.1 Cách chơi chính 16](#_Toc185877033)

[2.1.2 Cốt truyện 17](#_Toc185877034)

[2.1.3 Các phần tử trong game 17](#_Toc185877035)

[2.1.4 Các cơ chế của game 18](#_Toc185877036)

[Di chuyển 18](#_Toc185877037)

[2.2 Thiết kế giao diện 19](#_Toc185877038)

[2.2.1 Biểu đồ 19](#_Toc185877039)

[2.2.2 Mô tả 20](#_Toc185877040)

[2.2.3 Giao diện các màn hình chính 20](#_Toc185877041)

[2.3 Thiết kế hình ảnh 23](#_Toc185877042)

[2.3.1. 2D Art PC and NPC 23](#_Toc185877043)

[2.3.2 2D Art Game Objects 24](#_Toc185877044)

[2.3.3 2D Art GUI Elements 24](#_Toc185877045)

[2.3.3 Game Actions Animations 25](#_Toc185877046)

[2.4 Thiết kế âm thanh 25](#_Toc185877047)

[2.5 Luật chơi 26](#_Toc185877048)

[CHƯƠNG 3: CÁC KỸ THUẬT XÂY DỰNG GAME VÀ KẾT QUẢ THỰC NGHIỆM 26](#_Toc185877049)

[3.1 Các kỹ thuật để xây dựng game 26](#_Toc185877050)

[3.1.1 General 26](#_Toc185877051)

[3.1.2. Level 32](#_Toc185877052)

[3.1.3. Điều khiển cho nhân vật chính (Main) 34](#_Toc185877053)

[3.1.4.Robot A 40](#_Toc185877054)

[3.1.5. Robot C 44](#_Toc185877055)

[3.1.6. Xây dựng giao diện người dùng cho game (UI) 46](#_Toc185877056)

[3.1.7. Kỹ thuật thiết kế ảnh (Tilemap) 49](#_Toc185877057)

[3.2 Hướng dẫn cài đặt 49](#_Toc185877058)

[3.2 Đánh giá kết quả 52](#_Toc185877059)

[3.3 Hướng phát triển 52](#_Toc185877060)

[KẾT LUẬN 53](#_Toc185877061)

[TÀI LIỆU THAM KHẢO 54](#_Toc185877062)

# DANH MỤC HÌNH ẢNH

[Hình 1.1: Các nền tảng thông dụng 5](#_Toc123564460)

[Hình 1.2: Sprite 6](#_Toc123564467)

[Hình 1.3: Các sprite tạo thành animation 7](#_Toc123564475)

[Hình 1.4: Các Scenes 8](#_Toc123564484)

[Hình 1.5: Bố trí mặc định của Unity 9](#_Toc123564492)

[Hình 1.6: Cửa sổ Scenes 10](#_Toc123564498)

[Hình 1.7: Giao diện cửa sổ Hierarchy 11](#_Toc123564530)

[Hình 1.8: Giao diện cửa sổ Game 11](#_Toc123564536)

[Hình 1.9: Giao diện cửa sổ Project 12](#_Toc123564541)

[Hình 1.10: Giao diện cửa sổ Inspector 12](#_Toc123564552)

[Hình 2.1: Biểu đồ thiết kế game 17](#_Toc123564561)

[Hình 2.2: Giao diện Unity 17](#_Toc123564591)

[Hình 2.3: Giao diện khi bắt đầu game 18](#_Toc123564600)

[Hình 2.4: Giao diện trong game 18](#_Toc123564605)

[Hình 2.5: Giao diện khi dừng game 19](#_Toc123564612)

[Hình 2.6: Giao diện cài đặt 21](#_Toc123564619)

[Hình 2.7: Z01 22](#_Toc123564635)

[Hình 2.8: Robot loại A 23](#_Toc123564643)

[Hình 2.9: Robot loại B 23](#_Toc123564663)

[Hình 2.10: Robot loại C 24](#_Toc123564672)

[Hình 3.1: Các tilemap trong game 48](#_Toc123564707)

[Hình 3.2: Cửa sổ đăng nhập 49](#_Toc123564714)

[Hình 3.3: Kích hoạt License 49](#_Toc123564719)

[Hình 3.4: Chọn phiên bản Unity 50](#_Toc123564723)

[Hình 3.5: Cài đặt các conpnent 50](#_Toc123564723)

**LỜI NÓI ĐẦU**

Game là một khái niệm không còn xa lạ với chúng ta. Với tốc độ phát triển, bùng nổ đến chóng mặt của ngành công nghệ game thì đã có rất nhiều game hay và hấp dẫn được ra đời nhằm thỏa mãn nhu cầu giải trí của con người sau thời gian học tập, làm việc mệt mỏi. Bên cạnh đó, game còn giúp chúng ta tăng khả năng tư duy, logic. Vì thế để có thể tạo ra game hay và thu hút người chơi thì quá trình phát triển và xây dựng game rất được chú trọng.

Hiện nay, có rất nhiều thể loại game như: casual, platformer, survival horror,... và nền tảng phát triển đang rất nổi tiếng đó là Unity. Unity là một công cụ miễn phí, chất lượng cực cao, hỗ trợ nền tảng tuyệt vời và có một cộng đồng khổng lồ và tích cực tham gia đóng góp gồm các chuyên gia và những người có cùng sở thích đóng góp vào cơ sở kiến ​​thức và tài nguyên học tập cũng như kho dữ liệu. Hệ thống của Unity để lập trình các nội dung và công cụ mới là một trong những hệ thống linh hoạt nhất và có thể mở rộng nhất hiện có, cho phép cả trò chơi 2D và 3D và cho phép bạn xây dựng quy trình làm việc của riêng mình. Trong bài tập lớn dưới đây chúng em sẽ “**Xây dựng game hành động RobotSurvival**”.

Trong quá trình thực hiện đề tài chúng em xin gửi lời cảm ơn chân thành đến thầy Nguyễn Văn Tỉnh giảng viên hướng dẫn nhóm em thực hiện đề tài này. Trong quá trình nghiên cứu và thực hiện đề tài được sự chỉ bảo tận tình của thầy, chúng em đã cố gắng hết sức để hoàn thiện đề tài. Tuy nhiên chúng em rất mong nhận được sự góp ý của thầy và các bạn.

Chúng em xin trân trọng cảm ơn!

Nhóm 1

# GIỚI THIỆU PHẦN MỀM UNITY

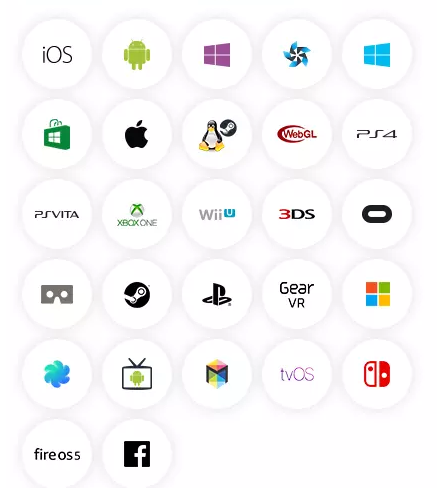
## Giới thiệu chung

### Khái niệm cơ bản về Unity

Unity là một trong những phần mềm có khả năng làm game trên đa nền tảng nhờ vào Unity Technologies. Nó có khả năng phát triển các video game cho máy tính, consoles và điện thoại di động.

Unity là một môi trường phát triển tích hợp, mạnh mẽ, hỗ trợ thao tác kéo thả tùy biến giao diện nhanh chóng, trực quan. Nó cung cấp các công cụ xử lý đồ họa, tích hợp sẵn thư viện vật lý, tính toán va chạm.

Hỗ trợ nền tảng thông dụng như OSX, Linux, Window, Web, iOS, Android, PS3, BB, Xbox 360…



Hình 1.1: Các nền tảng thông dụng

### 1.1.2 Các khái niệm cơ bản trong Unity

* **Game Object**

Một đối tượng cụ thể trong game gọi là một game object, có thể là nhân vật, đồ vật nào đó.

Ví dụ: cây cối, xe cộ, nhà cửa…

* **Component**

Một GameObject sẽ có nhiều thành phần cấu tạo nên nó như là hình ảnh (render), tập hợp các hành động (animator), thành phần xử lý va chạm (collision), tính toán vật lý (physical), mã điều khiển (script), các thành phần khác… mỗi thứ như vậy gọi là một component của GameObject.

* **Sprite**

Là một hình ảnh 2D của một GameObject có thể là hình ảnh đầy đủ, hoặc có thể là một bộ phận nào đó.



Hình 1.2: Sprite

* **Animation**

Là tập hợp một hình ảnh động dựa trên sự thay đổi liên tục của nhiều sprite khác nhau.



Hình 1.3: Các sprite tạo thành animation

* **Key Frame**

Key Frame hay Frame là một trạng thái của một animation. Có thể được tạo nên từ 1 sprite hay nhiều sprite khác nhau.

* **Prefabs**

Là một khái niệm trong Unity, dùng để sử dụng lại các đối tượng giống nhau có trong game mà chỉ cần khởi tạo lại các giá trị vị trí, tỉ lệ biến dạng và góc quay từ một đối tượng ban đầu.

Ví dụ: Các đối tượng là đồng tiền trong game Mario đều có xử lý giống nhau nên ta chỉ việc tạo ra một đối tượng ban đầu, các đồng tiền còn lại sẽ sử dụng prefabs. Hoặc khi ta lát gạch cho một cái nền nhà, các viên gạch cũng được sử dụng là prefabs.

* **Sounds**

Âm thanh trong game.

* **Script**

Script là tập tin chứa các đoạn mã nguồn, dùng để khởi tạo và xử lý các đối tượng trong game.

Trong Unity có thể dùng ngôn ngữ lập trình C#, Java Script, BOO để lập trình Script.

Unity có hỗ trợ sẵn phần mềm MonoDevelop dùng để lập trình.

* **Scenes**

Quản lý tất cả các đối tượng trong một màn chơi của game.



Hình 1.4: Các Scenes

* **Assets**

Bao gồm tất cả những gì phục vụ cho dự án game như sprite, animation, sound, script, scenes…

* **Camera**

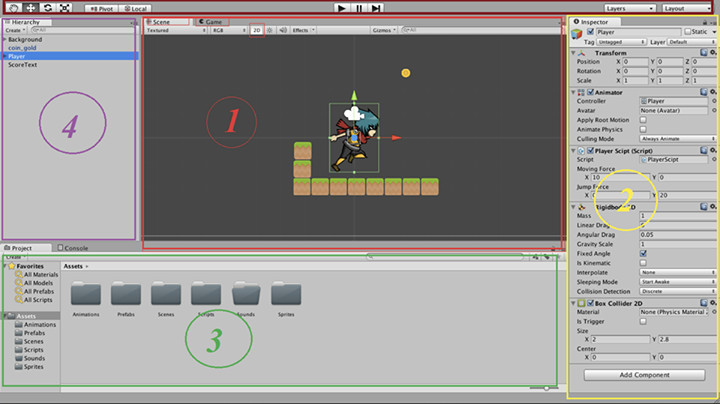
Là một game object đặc biệt trong scene, dùng để xác định tầm nhìn, quan sát các đối tượng.

* **Transform**

Là ba phép biến đổi tịnh tiến, quay theo các trục và phóng to thu nhỏ một đối tượng.

## 1.2 Giao diện làm việc và các chức năng chính

Giao diện Unity giống như nhiều môi trường làm việc khác, layout có thể tùy chỉnh. Layout của Unity bao gồm nhiều tab khác nhau và có thể bật tắt. Và bố trí giao diện điển hình như sau:



Hình 1.5: Bố trí mặc định của Unity

Như hình trên có 5 khung hình khác nhau:

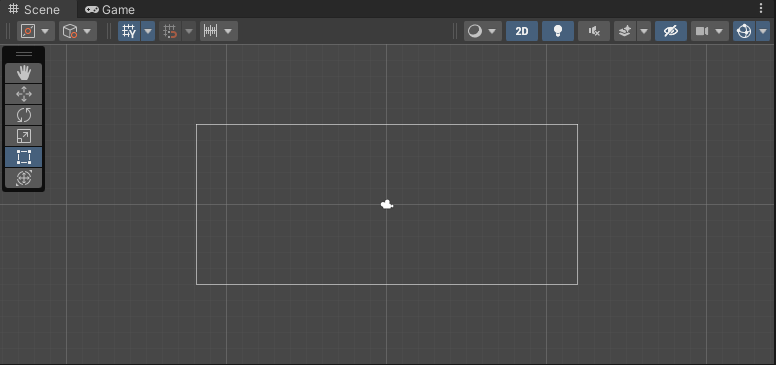
* Scene [1.1] – nơi xây dựng trò chơi.
* Game [1.2] – cửa sổ xem trước game, chỉ hoạt động ở chế độ chơi (khi nhấn Play).
* Inspector [2] – màn hình cài đặt cho tài nguyên/đối tượng đang được chọn.
* Project [3] – danh sách các tài nguyên trong project, đóng vai trò như một thư viện.
* Hierarchy [4] – danh sách các GameObject trong một cảnh game.

### 1.2.1 Cửa sổ Scenes

Phần màu đỏ số 1, phần này hiển thị các đối tượng trong scenes một cách trực quan, có thể lựa chọn các đối tượng, kéo thả, phóng to, thu nhỏ, xoay các đối tượng...

Phần này có để thiết lập một số thông số như hiển thị ánh sáng, âm thanh, cách nhìn 2D hay 3D...

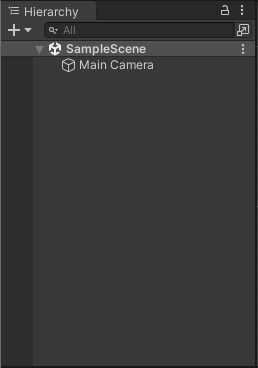
Khung nhìn Scene là nơi bố trí các GameObject như cây cối, cảnh quan, enemy, player, camera,... trong game. Sự bố trí hoạt cảnh là một trong những chức năng quan trọng nhất của Unity.



Hình 1.6: Cửa sổ Scenes

### 1.2.2 Cửa sổ Hierarchy

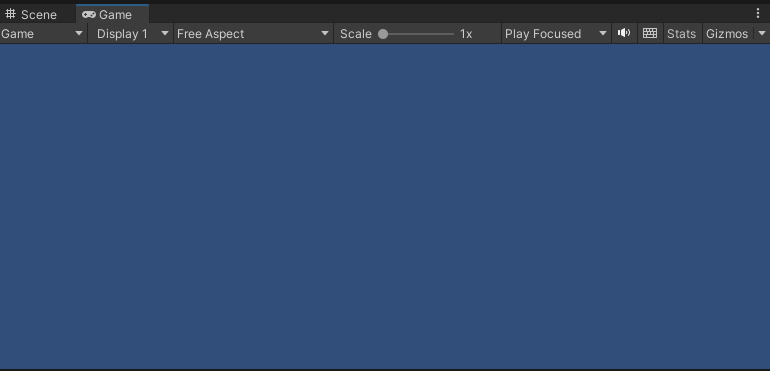
Phần màu tím số 4, phần này là nơi hiển thị các GameObject trong Scenes hiện hành. Khi các đối tượng được thêm hoặc xóa trong Scenes, tương ứng với các đối tượng đó trong cửa sổ Hierarchy.



Hình 1.7: Giao diện cửa sổ Hierarchy

### 1.2.3 Cửa sổ Game

Phần màu đỏ số 1, phần này hiển thị game khi thực thi, một tab bên cạnh của tab Scenes. Thanh công cụ trong cửa sổ game cung cấp các tùy chỉnh về độ phân giải màn hình, thông số (stats), gizmos, tùy chọn bật tắt các component.



Hình 1.8: Giao diện cửa sổ Game

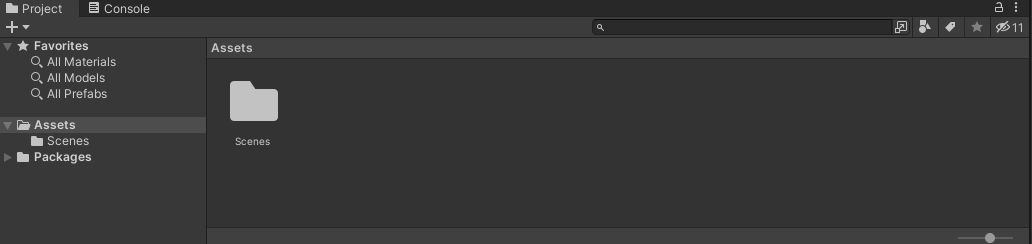
### 1.2.4 Cửa sổ Project

Phần màu xanh lá số 3, đây là cửa sổ explorer của Unity, hiển thị thông tin của tất cả các tài nguyên (Assets) trong game.

Cột bên trái hiển thị assets và các mục yêu thích dưới dạng cây thư mục tương tự như Windows Explorer. Khi kích vào một nhánh trên cây thư mục thì toàn bộ nội dung của nhanh đó sẽ được hiển thị ở khung bên phải. Và có thể tạo ra các thư mục mới bằng cách vào Right click Create chọn Folder hoặc nhấn vào nút Create ở góc trên bên trái cửa sổ Project và chọn Folder. Các tài nguyên trong game cũng có thể được tạo ra bằng cách này.

Phía trên cây thư mục và mục Favorites, giúp truy cập nhanh vào những tài nguyên thường được sử dụng. Và có thể đưa các tài nguyên vào Favorites bằng thao tác kéo thả.

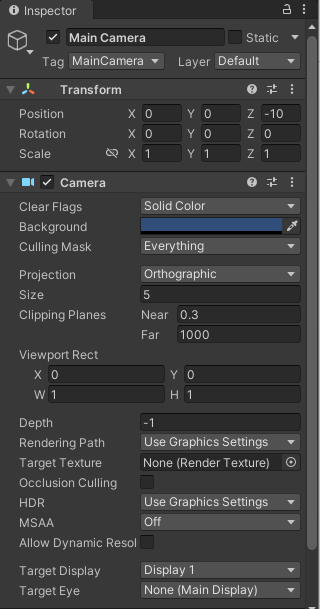
Đường dẫn của thư mục tài nguyên hiện tại có thể dễ dàng tiếp cận các thư mục con hoặc thư mục gốc bằng cách kích chuột vào mũi tên hoặc tên thư mục.



Hình 1.9: Giao diện cửa sổ Project

### 1.2.5 Cửa sổ Inspector

Phần màu vàng số 2, cửa sổ Inspector hiển thị chi tiết các thông tin về GameObject đang làm việc, kể cả những component được đính kèm và thuộc tính của GameObject. Và có thể điều chỉnh, thiết lập mọi thông số và chức năng của GameObject.



Hình 1.10: Giao diện cửa sổ Inspector

### 1.2.6 Top bar

Phần đóng khung màu đỏ, phần này chứa các nút chuyên dụng:

* Bốn nút bên trái:
* Tùy chọn cho phép dùng chuột kéo toàn bộ scenes, phóng to, thu nhỏ, lựa chọn các đối tượng trong scenes...
* Cho phép dùng chuột di chuyển các đối tượng trong scenes,.
* Cho phép dùng chuột quay các đối tượng.
* Cho phép dùng chuột phóng to, thu nhỏ các đối tượng.
* Ba mút ở giữa:
* Cho phép chạy demo game.
* Cho phép dừng game tại một frame nào đó.
* Cho phép chạy từng frame.
* Hai drop list bên phải:

Cho phép tùy chọn hiển thị các layer.

* Cho phép chọn và lưu các bố trí do người dùng thiết lập.

Ngoài ra còn có các cửa sổ khác của Unity, như Sprite Editor, Animation, Animator...

# CHƯƠNG 2. XÂY DỰNG GAME

## 2.1 Giới thiệu tổng quan

### 2.1.1 Cách chơi chính

RobotSurvival là game nhập vai đơn người chơi. Bối cảnh được lấy tại một căn cứ thí nghiệm của robot để cải tạo con người. Căn cứ được chia thành nhiều gian phòng với các chức năng cải tạo và phòng thủ khác nhau. Trò chơi được bắt đầu tại một trong những phòng thí nghiệm của căn cứ.

Người chơi có thể di chuyển tự do theo 8 hướng, phòng thủ và tấn công bằng vũ khí được gắn trên cơ thể sau khi cải tạo. Một số thuộc tính của người chơi là máu (máu giảm khi nhân vật chịu sự tấn công, giá trị này càng nhỏ thì một số hoạt động của nhân vật càng khó khăn, khi giá trị này bằng 0 thì trò chơi kết thúc), năng lượng (được sử dụng để kích hoạt các khả năng đặc biệt của nhân vật).

Nhân vật được đặt tại phòng thí nghiệm sâu nhất (ở bên trái) trong căn cứ. Nhấn các phím đơn lẻ hoặc tổ hợp của các phím W, A, S, D để di chuyển nhân vật đi lên, sang trái, đi xuống, sang phải và đi chéo. Sử dụng các phím mũi tên để di chuyển nòng súng cũng như hướng bắn của nhân vật theo 8 hướng, nhấn Space để bắn. Nhấn R để phòng thủ. Ngoài ra, người chơi cũng có thể nhấn phím E để nhặt các vật phẩm rơi ra từ robot và tương tác với các NPC ở trong game.

Trong quá trình người chơi tìm hiểu bí mật và cách thoát khỏi căn cứ. Đội quân robot sẽ tổ chức các cuộc tấn công để tiêu diệt người chơi. Lúc này chúng ta phải điều khiển nhân vật tiêu diệt chúng và cố gắng để máu không giảm về 0. Trong khi chiến đấu, người chơi có thể sử dụng các linh kiện (rơi ra từ robot bị tiêu diệt) để sửa chữa, lõi năng lượng còn sót lại của robot để cung cấp năng lượng và một số thiết bị đặc biệt để nâng cấp.

Các màn chơi khác nhau đòi hỏi người chơi phải xử lí khác nhau để có thể vượt qua chứ không đơn thuần là tiêu diệt hết kẻ thù.

Trò chơi kết thúc khi nhân vật thoát khỏi căn cứ hoặc bị tiêu diệt.

Trò chơi là sự tự trải nghiệm thế giới trong game của người chơi. Vì vậy nó cũng không có một chiến lược cụ thể nào để thắng. Nó phụ thuộc vào sự tò mò tìm hiểu, khả năng chơi game của mỗi người. Người chơi có thể kết thúc game với sự hiểu biết thêm về cốt truyện của game thông qua các NPC như đang xem một bộ phim khoa học viễn tưởng. Hoặc người chơi có thể không quan tâm đến mà chỉ cảm nhận sự hưng phấn khi tiêu diệt kẻ thù và chiến thắng trò chơi.

### 2.1.2 Cốt truyện

Trò chơi được lấy ý tưởng dựa trên bộ phim mang tên “The Matrix”. Bộ phim lấy bối cảnh khoảng năm 2199, nhân loại phải chiến đấu với những robot nổi loạn do khoa học kỹ thuật phát triển cao. Trong cuộc chiến này, robot đã dành chiến thắng nhưng chúng cũng không thể sử dụng nguồn năng lượng dồi dào là năng lượng mặt trời vì con người đã tạo ra các đám mây dày đặc để ngăn chặn. Tuy nhiên, vì vậy mà con người trở thành nguồn năng lượng mới của chúng và được “nuôi” trong các phòng thí nghiệm.

RobotSurvival cũng lấy bối cảnh sau cuộc chiến giữa con người và robot. Sự thất bại khiến cho con người lâm vào hoàn cảnh vô cùng khó khăn. Một số người cố gắng sống sót ở những nơi hẻo lánh, một số bị bắt đi. Khi robot phát triển hơn, chúng cũng rơi vào các cuộc chiến tranh vì tài nguyên. Những người bị bắt đi lúc này bị biến thành robot tiên phong trong các cuộc chiến tranh của chúng. Nhân vật chính (D01 – robot đầu tiên loại D) là một trong số đó, tuy nhiên vì một số lý do nên không chịu sự khống chế của robot và bắt đầu trên hành trình nổi dậy của nhân loại.

### 2.1.3 Các phần tử trong game

**Nhân vật chính**

|  |  |
| --- | --- |
| D01 | Máu: 290/300  Năng lượng: 100 |
| Actions | Description | Condition |
| Di chuyển | Đi theo một trong 8 hướng với tốc độ 3 đơn vị/giây |  |
| Phòng thủ | Tạo một màn chắn bằng năng lượng ngăn lại các cuộc tấn công trong 1 giây | 50 năng lượng/lần |
| Tấn công | Bắn ra một đường đạn theo hướng chỉ định gây 30ST | 1 năng lượng/lần |
| Nhặt vật phẩm | Thu thập vật phẩm rơi trên mặt đất | Khoảng cách với vật phẩm nhỏ hơn 1.5 đơn vị |
| Một số khả năng đặc biệt khác | Sau khi nâng cấp, nhân vật được thêm các khả năng riêng | Thu thập đủ một số vật phẩm yêu cầu |

**Lõi năng lượng**: vật phẩm được rơi ra từ robot. Mỗi loại robot riêng sẽ có lõi năng lượng với năng lượng dư thừa lại khác nhau sau khi bị tiêu diệt. Người chơi thu thập để tăng năng lượng.

**Xác robot**: sau khi bị tiêu diệt, một số robot khi bị tiêu diệt vẫn có độ hoàn thiện cao. Nhân vật có thể sử dụng một số linh kiện trong đó để sửa chữa cho mình.

**Vật phẩm nâng cấp**: loại robot khác nhau rơi ra vật phẩm khác nhau.

**Robot**: các robot có khả năng tấn công được điều khiển tấn công người chơi để ngăn chặn mối họa cho căn cứ. Các loại robot khác nhau có khả năng di chuyển và tấn công không giống nhau. Bảng sau hiển thị sự khác biệt đó:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Type | Health | Move | Attack Damage | Items |
| A | 100 | 2 đơn vị/giây | 10 | **Giáp**: giảm 10% sát thương nhận vào |
| B | 50 | 3.5 đơn vị/giây | 20 | **Bộ phản lực**: tăng 10% tốc độ di chuyển |
| C | 30 | 1.5 đơn vị/giây | 50 | **Vi xử lý nổ**: tăng 10% sát thương lên kẻ địch |

**Robot điều khiển**: loại robot đặc biệt phụ trách điều khiển các robot khác tấn công nhân vật. Chứa một số thông tin quan trọng trong cốt truyện của game.

### 2.1.4 Các cơ chế của game

### Di chuyển

Nhân vật di chuyển theo hướng được chỉ định với các phím mà người chơi nhấn xuống (xem lại phần *cách chơi chính*).

NPC (có thể di chuyển) di chuyển về vị trí có thể tấn công nhân vật theo các tốc độ riêng biệt (xem lại phần *các phần tử trong game*). Khi bị tiêu diệt, các vật phẩm rơi ra (các vật phẩm này không di chuyển).

**Chiến đấu**

Các robot bắn đạn (A và B) theo hướng nhân vật. Khi va chạm vào nhân vật, nhân vật bị đẩy lùi theo hướng đạn và chịu sát thương nhất định tùy theo loại robot tấn công.

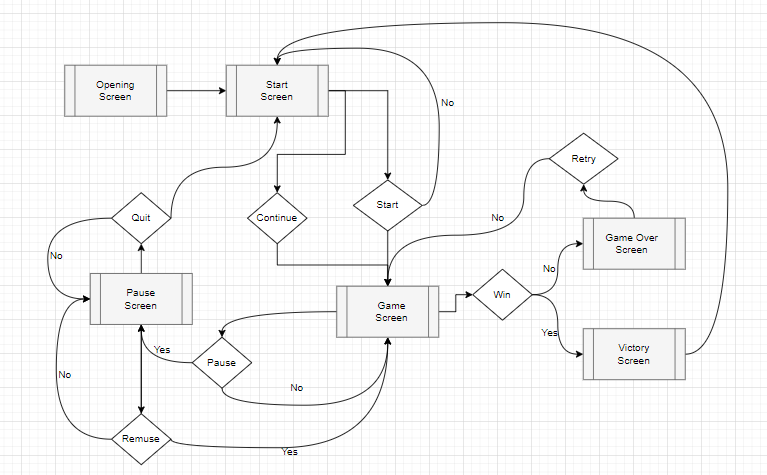
Robot C khi phát nổ mà nhân vật trong tầm ảnh hưởng cũng bị đẩy lùi và chịu sát thương, đồng thời bị đình chỉ khả năng di chuyển trong 0.1 giây.

**Artificial Intelligence**

Các NPC được lập trình để thực hiện các hành động theo trình tự phù hợp. Sau khi xuất hiện trong thế giới game, hành động đầu tiên là xác định vị trí của nhân vật. NPC sau đó sẽ di chuyển đến vị trí gần nhất mà ở đó nhân vật nằm trong tầm tấn công của chúng. Việc di chuyển đến vị trí có thể tấn công của NPC được thực hiện liên tục mỗi khung hình. Khi đến được vị trí này, NPC sẽ tấn công theo các cách riêng của mỗi loại.

## 2.2 Thiết kế giao diện

### 2.2.1 Biểu đồ



*Hinh 2.1: Biểu đồ thiết kế game*

### 2.2.2 Mô tả

**Opening Screen**: Đây là màn hình đầu tiên xuất hiện khi ứng dụng đang tải.

**Start Screen**: Màn hình mở đầu. Nó chứa hình nền của game, nút bắt đầu, nút cài đặt, nút thoát. Chọn bắt đầu để chơi game, chọn cài đặt để có thể thay đổi một số cài đặt trong game, chọn thoát để tắt ứng dụng game.

**Game screen**: Đây là màn hình trò chơi, bao gồm: bản đồ trò chơi, PC, NPC, cũng một số GUI(các nâng cấp đã chọn, máu và năng lượng của nhân vật, nút tạm dừng).

**Pause screen**: Màn hình này chứa các nút để: thoát khỏi trò chơi, tiếp tục và thay đổi cài đặt game. Thoát khỏi trò chơi sẽ chuyển sang Start Screen. Quyết định tiếp tục sẽ chuyển đến Game Screen.

**Victory screen**: Màn hình này xuất hiện khi trò chơi được hoàn thành thành công sau đó người chơi được tự động đưa về màn hình Start Screen.

**Game over screen**: Màn hình này xuất hiện khi trò chơi hoàn thành không thành công sau đó người chơi được tự động đưa về màn hình Start Screen.

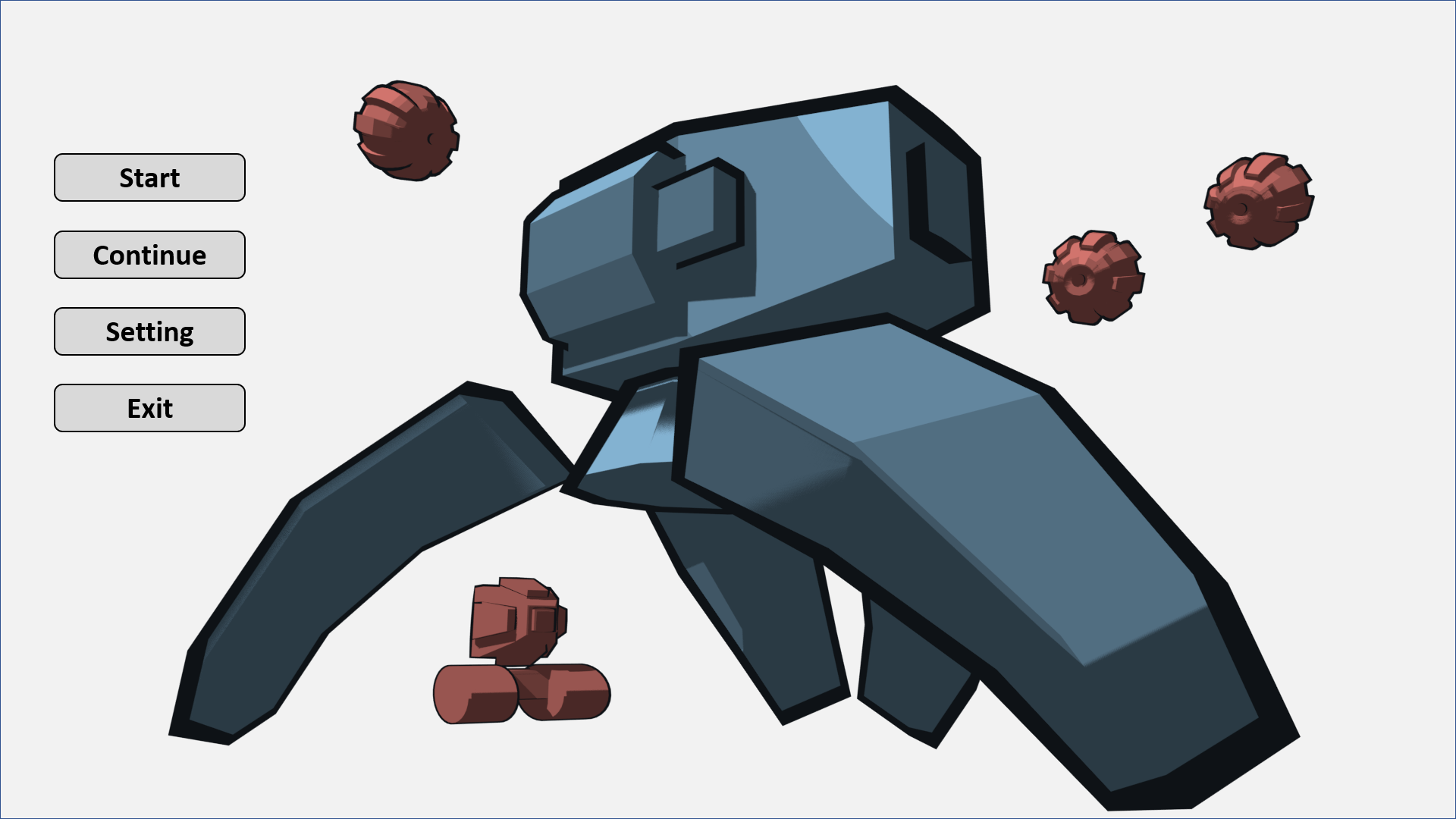
### 2.2.3 Giao diện các màn hình chính

**Opening Screen**



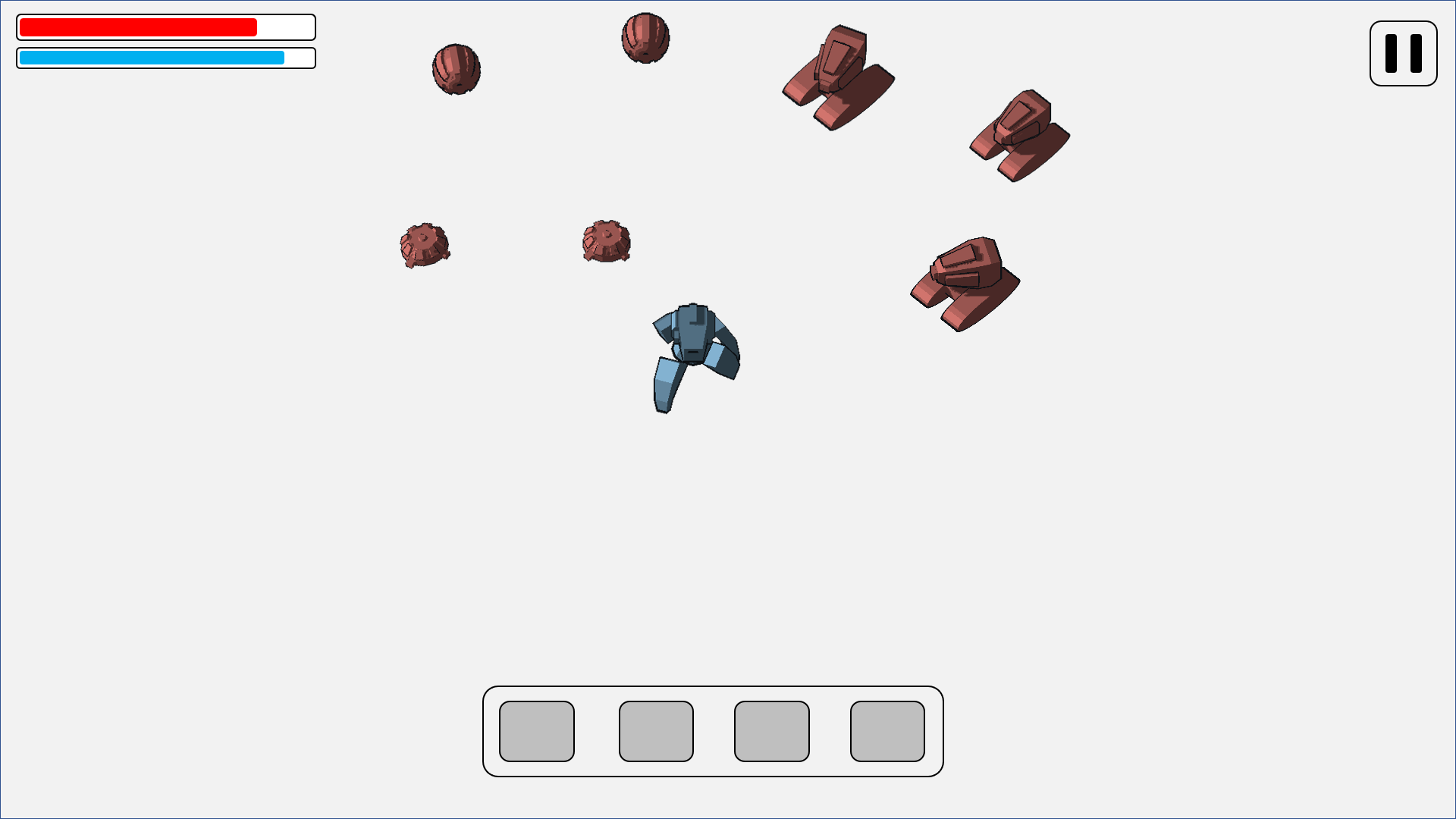
*Hình 2.2: Giao diện Unity*

**Start Screen**



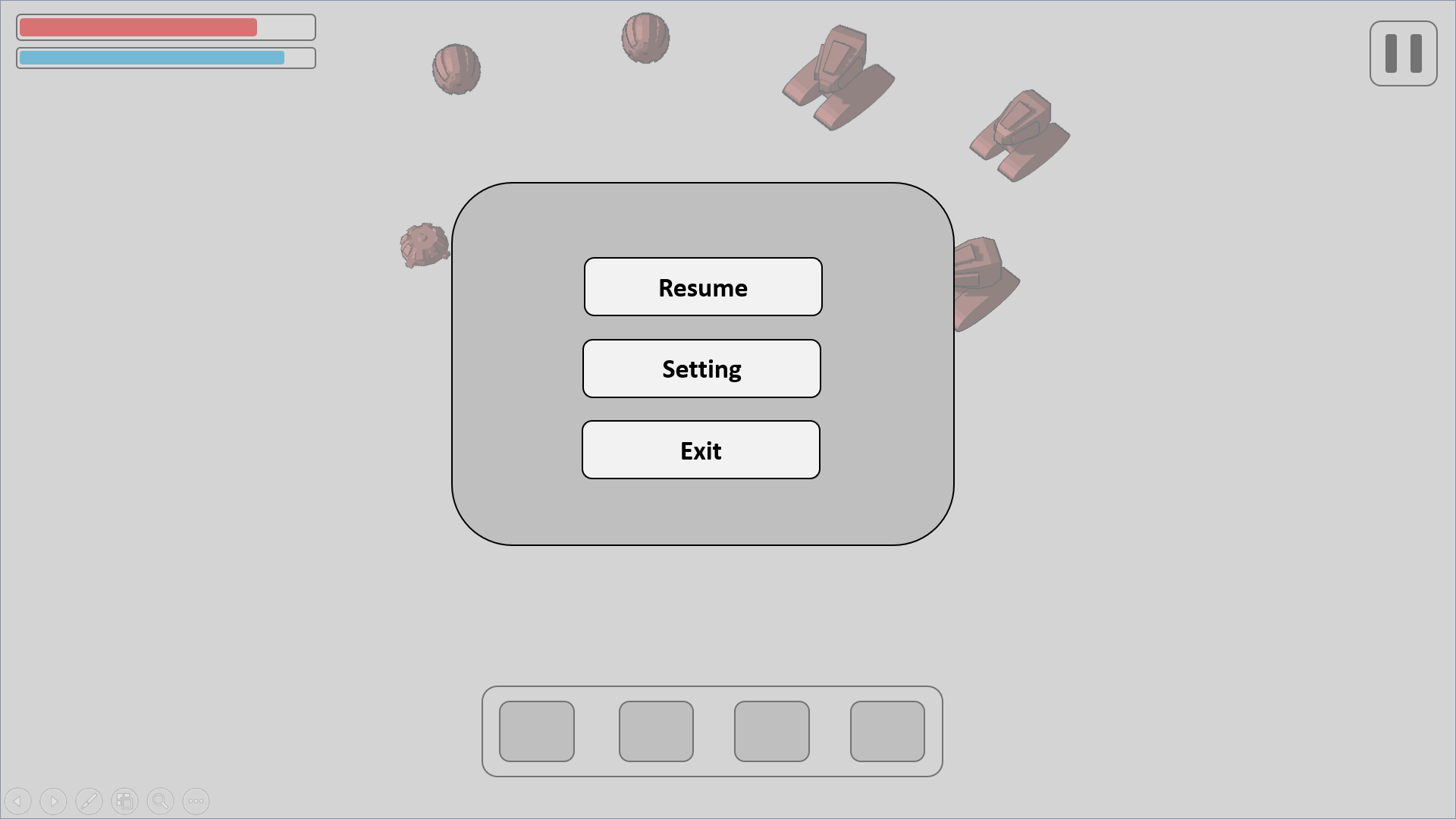
*Hình 2.3: Giao diện khi bắt đầu game*

**Game Screen**



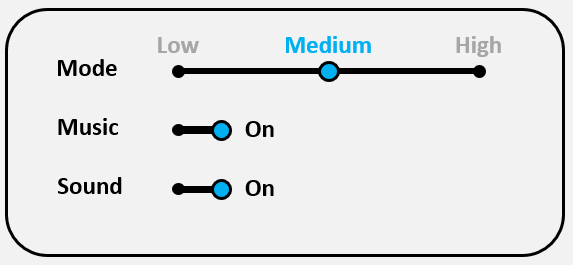
*Hình 2.4: Giao diện trong game*

**Pause Screen**



*Hình 2.5: Giao diện khi dừng game*

**Setting Screen**



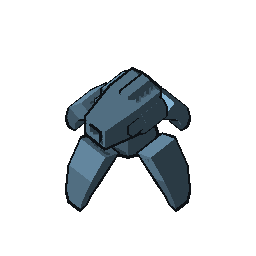
*Hình 2.6: Giao diện cài đặt*

**Buttons**

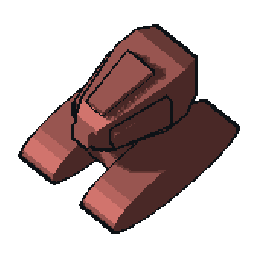
* **Start**: Trò chơi bắt đầu và sẽ chuyển tới Game Screen.
* **Continue**: Trò chơi bắt đầu với thời điểm tạm dừng cuối cùng và sẽ chuyển tới Game Screen.
* **Setting:** Mở cài đặt trong trò chơi.
* **Pause**: Tạm dừng trò chơi và chuyển đến Pause screen.
* **Resume**: Khi trò chơi bị tạm dừng. Quay lại Game Screen ở thời điểm tạm dừng cuối cùng.
* **Exit (Start Screen)**: Thoát khỏi ứng dụng trò chơi.
* **Exit (Pause Screen):** Kết thúc trò chơi và hướng tới Start Screen.

## 2.3 Thiết kế hình ảnh

### 2.3.1. 2D Art PC and NPC



*Hình 2.7: Z01*



*Hình 2.8: Robot loại A*



*Hình 2.9: Robot loại B*



*Hình 2.10: Robot loại C*

### 2.3.2 2D Art Game Objects

|  |  |
| --- | --- |
| Level | Sàn, tường, cửa, phòng điều khiển |
| Items | Lõi năng lượng, xác robot, giáp, bộ phản lực, vi xử lý nổ |

### 2.3.3 2D Art GUI Elements

|  |  |
| --- | --- |
| Buttons | Start, Continue, Setting, Exit, Pause, Resume |

### Game Actions Animations

|  |  |
| --- | --- |
| PC | Di chuyển, phòng thủ, tấn công, nhặt vật phẩm, tương tác với NPC |
| NPC | Di chuyển, tấn công |

## Thiết kế âm thanh

**Overall Goals**

Ý tưởng là tạo ra một bầu không khí u ám khi nhân vật di chuyển một mình, không khí căng thẳng khi chiến đấu.

**Units/Characters**

|  |  |
| --- | --- |
| PC movement | PC\_walk |
| PC attack | PC\_shoot |
| PC sub-actions | PC\_collect, PC\_access, PC\_destroy |
| NPC movement | A\_walk, B\_walk, C\_walk |
| NPC attack | A\_attack, B\_attack, C\_attack |
| NPC sub-actions | NPC\_destroy |
| NPC Controller action | Controller\_start |

**GUI**

|  |  |
| --- | --- |
| Buttons | Click, Select |

**Music**

|  |  |
| --- | --- |
| Music\_GUI | Start game, Menu game |
| Music\_Game | Game screen |
| Music\_Victory | Victory screen |
| Music\_Lose | Game over screen |

## Luật chơi

* + **Cách di chuyển của nhân vật:** Khi bắt đầu người chơi chỉ có thể di chuyển lên trên hoặc sang phải bằng các phím A, W, S, D.
  + **Cách dùng đạn để bắn kẻ địch:** Người chơi sử dụng con trỏ trái chuột để bắn.
  + **Khi nào người chơi thua:** Khi người chơi bị hết sạch máu.
  + **Điều khiển chiến thắng:** Người chơi tiêu diệt được hết các Robot A và B để di chuyển đến địa điểm sang màn mới.
  + **Mức độ khó của các màn:** Càng lên cao thì quái vật cùng các chướng ngại vật xuất hiện ngày càng nhiều.

# CHƯƠNG 3: CÁC KỸ THUẬT XÂY DỰNG GAME VÀ KẾT QUẢ THỰC NGHIỆM

## 3.1 Các kỹ thuật để xây dựng game

### 3.1.1 General

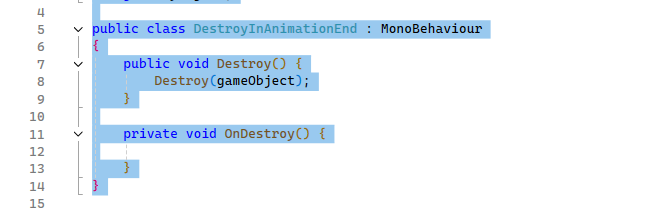
#### 3.1.1.1 Thu thập vật phẩm (Collectible )

Collectible xử lý việc người chơi thu thập các vật phẩm (sức khỏe, sát thương) trong trò chơi, bao gồm hiệu ứng biến mất khi vật phẩm được thu thập.

**Chức năng chính:**

1. **Kiểm tra va chạm (OnTriggerEnter2D):**
   * Khi người chơi (có thành phần Main\_Infor) va chạm với vật phẩm, kiểm tra xem người chơi có thể thu thập vật phẩm đó không.
   * Nếu người chơi có thể thu thập, thay đổi trạng thái vật phẩm (isCollecting = true).
2. **Hiệu ứng thu thập (Update):**
   * Khi vật phẩm đang được thu thập, hiển thị hiệu ứng khói và giảm dần độ mờ của vật phẩm cho đến khi nó biến mất.
   * Sau khi vật phẩm biến mất (sau thời gian timeHide), hủy đối tượng.

#### 3.1.1.2 Kĩ thuật hủy đối tượng (DestroyInAnimationEnd )

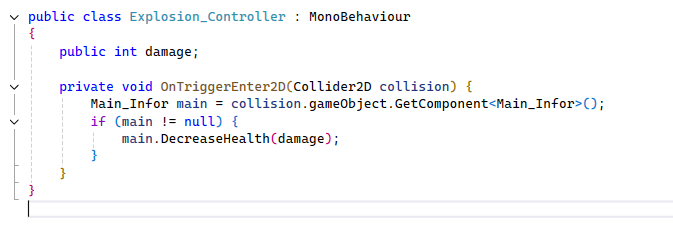


**DestroyInAnimationEnd** giúp tự động hủy đối tượng sau khi kết thúc một hoạt ảnh (animation).

**Chức năng chính:**

1. Hủy đối tượng sau khi kết thúc hoạt ảnh (Destroy):Phương thức Destroy() sẽ hủy đối tượng ngay lập tức khi được gọi, thường là từ một sự kiện trong hoạt ảnh.
2. Hàm OnDestroy**:** Hàm này được gọi khi đối tượng bị hủy, có thể dùng để thực hiện các thao tác dọn dẹp (nếu cần).

#### 3.1.1.3 Quản lý vụ nổ (Explosion Controller )



**Explosion Controller** quản lý vụ nổ và tác động của nó đến đối tượng va chạm, đặc biệt là giảm sức khỏe của đối tượng nếu là kẻ thù (hoặc nhân vật chính).

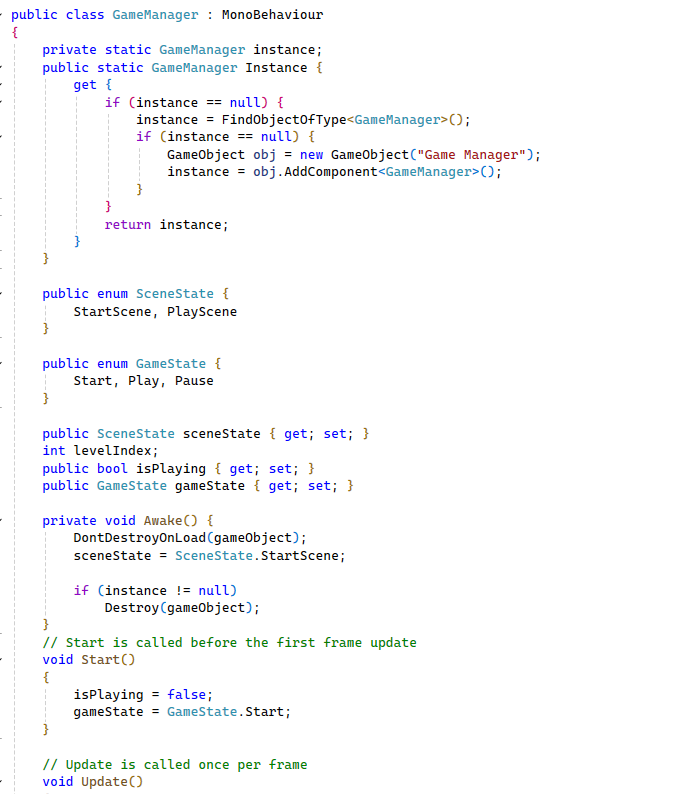
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

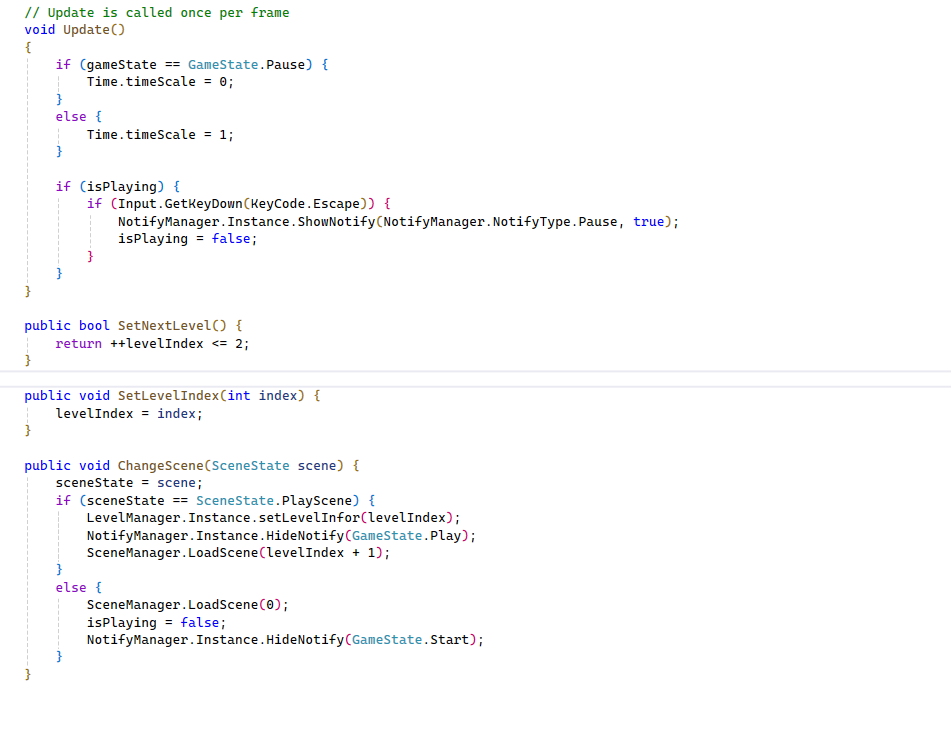
**Chức năng chính:**

1. Tác động khi va chạm (OnTriggerEnter2D): Khi vụ nổ va chạm với một đối tượng có thành phần Main\_Infor (như kẻ thù hoặc nhân vật chính), sẽ gọi hàm DecreaseHealth() trên đối tượng đó để giảm sức khỏe của nó.

2. Hàm damage: Damage chứa giá trị sát thương của vụ nổ, có thể được gán từ các đối tượng khác như Explosion hoặc nhân vật gây ra vụ nổ.

#### 3.1.1.4 Trung tâm điều khiển game (GameManager )





**GameManager:** Điều khiển trạng thái game, quản lý các cảnh (scene), xử lý việc chuyển cảnh và các trạng thái của trò chơi như bắt đầu, chơi, tạm dừng.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Chức năng chính của GameManager:

1. Quản lý các cảnh (Scenes):

* SceneState xác định trạng thái cảnh hiện tại, ví dụ: StartScene (cảnh bắt đầu) hoặc PlayScene (cảnh chơi).
* ChangeScene(): Chuyển giữa các cảnh trò chơi (bắt đầu, chơi) và cập nhật thông tin về cấp độ, ví dụ: chuyển đến màn chơi tiếp theo hoặc trở về màn hình bắt đầu.

2. Quản lý trạng thái trò chơi (Game State):

* GameState xác định trạng thái của trò chơi: Start (bắt đầu), Play (đang chơi), Pause (tạm dừng).
* gameState: Được sử dụng trong Update() để điều chỉnh tốc độ thời gian (Time.timeScale), tạm dừng hoặc tiếp tục trò chơi.

3. Xử lý cấp độ (Level Management):

* SetLevelIndex(): Cài đặt cấp độ hiện tại khi người chơi chọn cấp độ.
* SetNextLevel(): Kiểm tra và cập nhật chỉ số cấp độ cho trò chơi. Trả về true nếu có cấp độ tiếp theo.

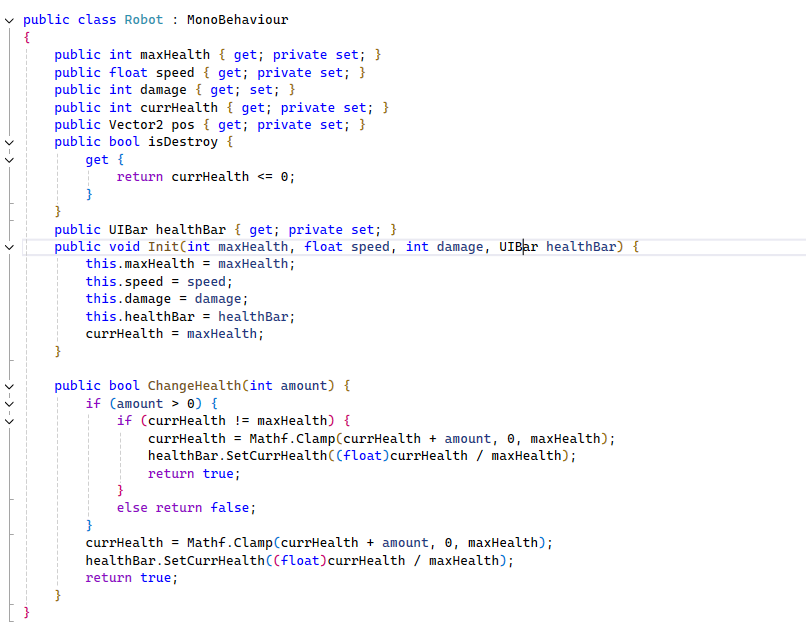
4. Tạm dừng và tiếp tục trò chơi:

Update(): Kiểm tra khi người chơi nhấn phím Escape (ESC) để tạm dừng trò chơi. Khi game ở trạng thái Pause, Time.timeScale được đặt về 0 để dừng mọi hoạt động thời gian.

5. Singleton Pattern:

Instance: Đảm bảo chỉ có một đối tượng GameManager trong game và có thể truy cập từ bất kỳ nơi nào trong mã nguồn.

#### 3.1.1.5 Xây dựng Robot (Robot)



**Robot** quản lý thông tin và trạng thái của nhân vật trong game, bao gồm sức khỏe, tốc độ, sát thương và vị trí.

**Chức năng chính:**

1. Quản lý sức khỏe: Cập nhật và thay đổi sức khỏe, hiển thị trên thanh sức khỏe (UIBar).
2. Tình trạng hủy diệt: Kiểm tra nếu robot đã bị hủy (sức khỏe <= 0).
3. Khởi tạo robot: Gán giá trị ban đầu cho sức khỏe, tốc độ, sát thương.
4. Cập nhật thanh sức khỏe: Đồng bộ sức khỏe hiện tại với thanh UI.

### 3.1.2. Level

#### 3.1.2.1. Xử lí va chạm với cửa (Doorcheck)



DoorCheck: xử lý sự kiện khi người chơi đi qua cửa, điều khiển trạng thái trò chơi và kích hoạt kẻ thù hoặc hiệu ứng.

Chức năng chính:

1. **Kiểm tra va chạm (OnTriggerEnter2D):** Khi người chơi (có thành phần Main\_Infor) va chạm với cửa, kiểm tra trạng thái trò chơi.
2. **Kích hoạt sự kiện khi qua cửa:**

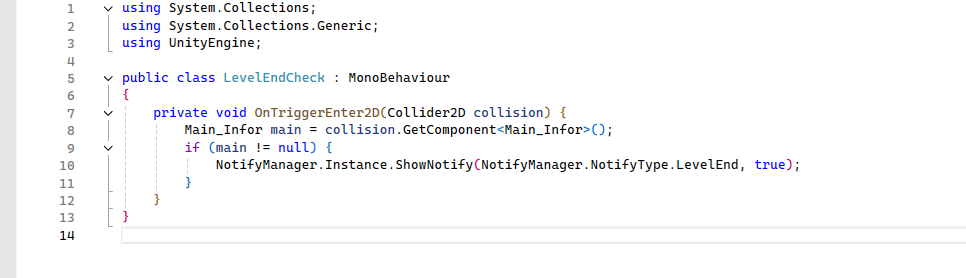
Nếu trò chơi chưa bắt đầu và cửa là loại Start, thì:

* Bắt đầu trò chơi (GameManager.Instance.isPlaying = true).
* Kích hoạt tất cả kẻ thù trong mảng enemy.
* Hiển thị thông báo bắt đầu cấp độ.

1. **Điều khiển cửa:**

* Bật hoạt ảnh và phát âm thanh cho cửa.
* Xóa đối tượng DoorCheck sau khi xử lý.

#### 3.1.2.2. Kiểm tra và thông báo khi kết thúc màn chơi (LevelEndCheck)



**LevelEndCheck** xử lý sự kiện khi người chơi chạm vào điểm kết thúc của cấp độ, hiển thị thông báo kết thúc cấp độ.

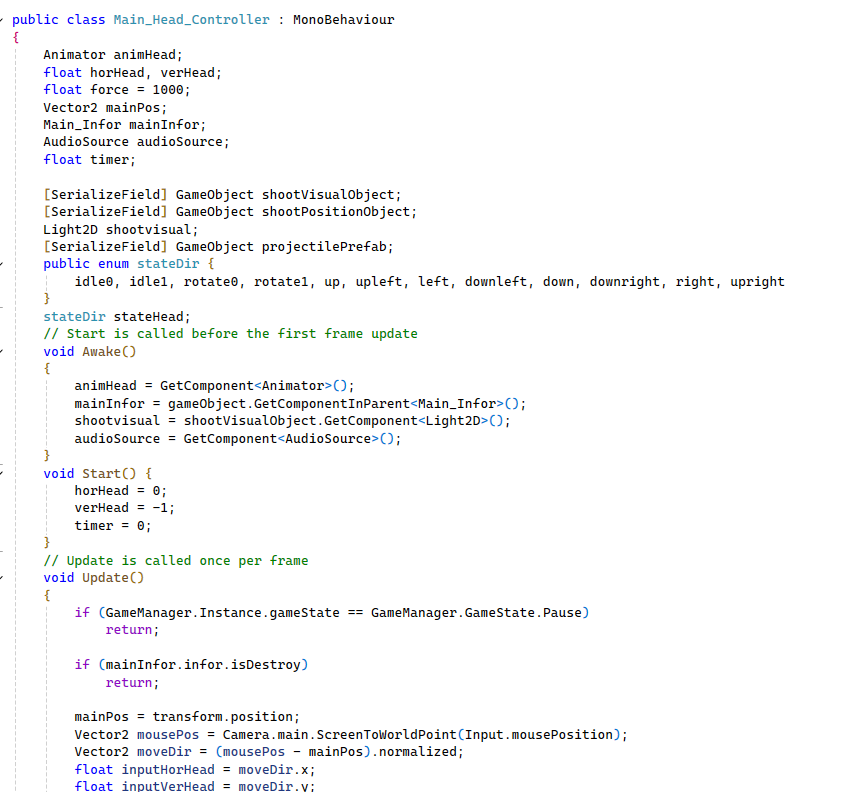
Chức năng chính:

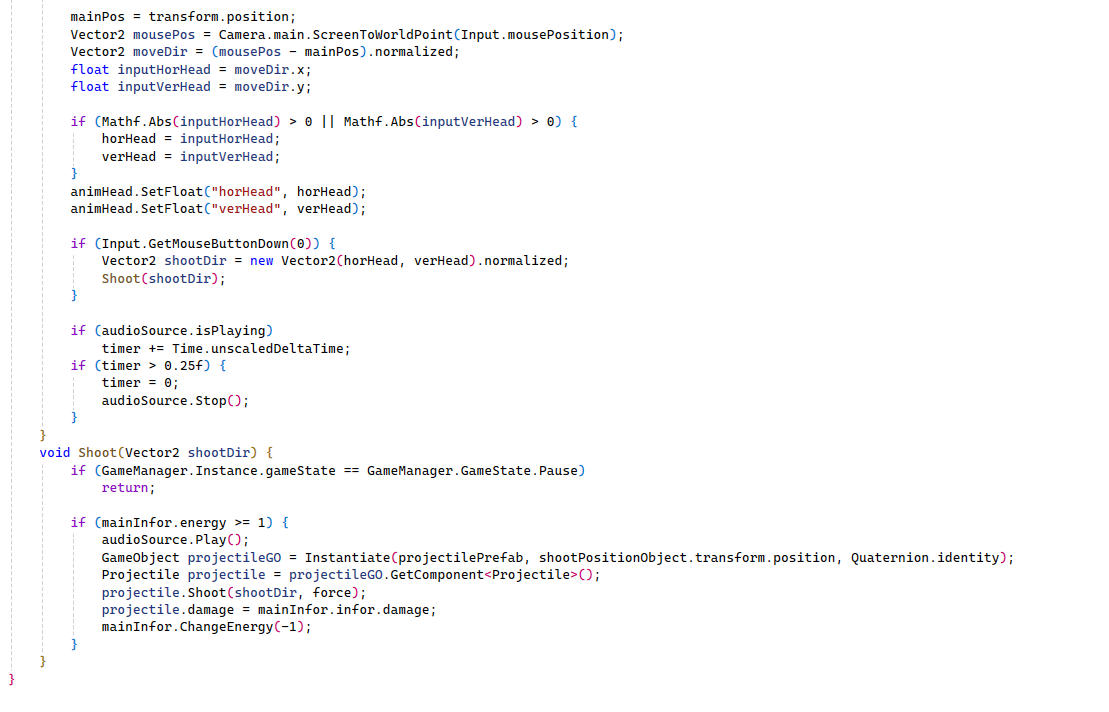
1. Kiểm tra va chạm (OnTriggerEnter2D): Khi người chơi (có thành phần Main\_Infor) va chạm với đối tượng, xác định điểm kết thúc cấp độ.

2. Hiển thị thông báo kết thúc cấp độ: Khi người chơi chạm vào điểm kết thúc, hiển thị thông báo kết thúc cấp độ thông qua NotifyManager.

### 3.1.3. Điều khiển cho nhân vật chính (Main)

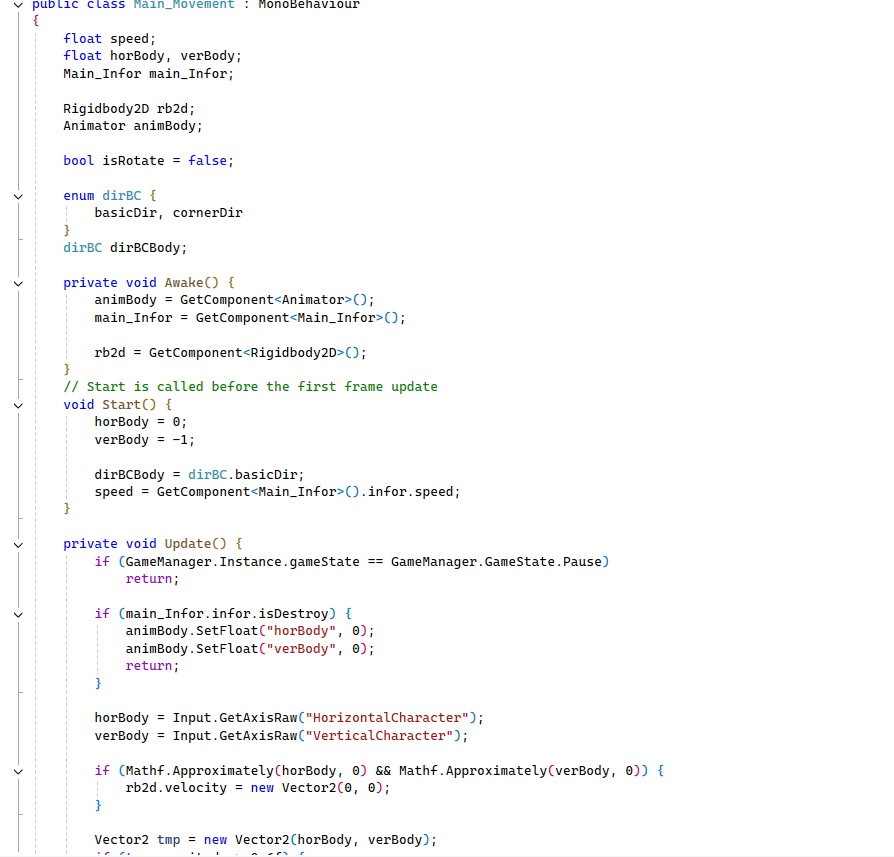
#### 3.1.3.1. Thiết kế chuyển động cho đầu của nhân vật (MainHeadController)

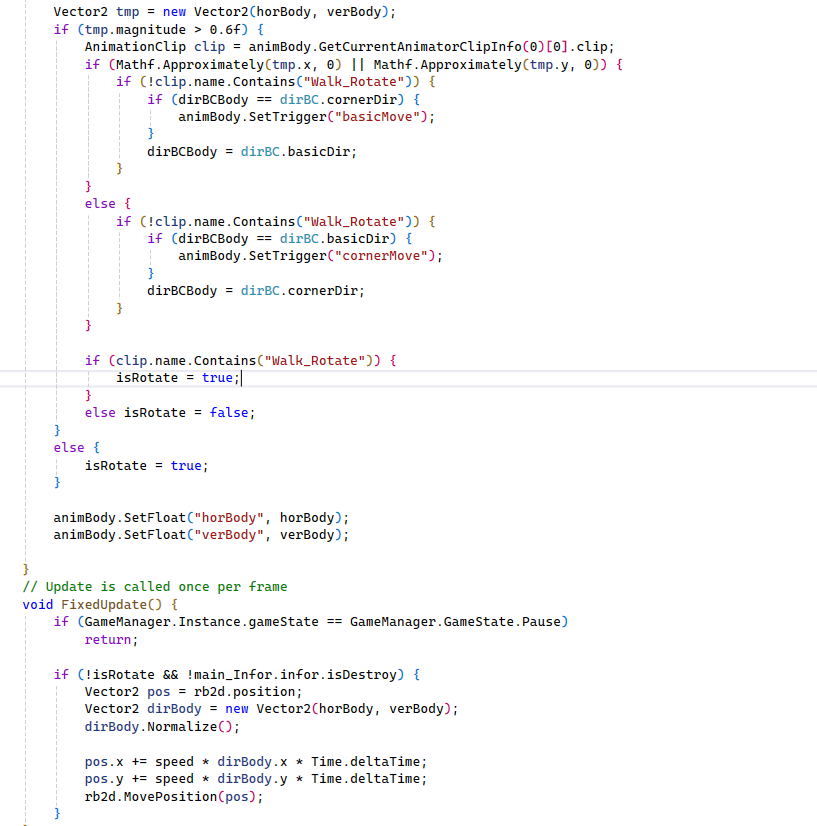


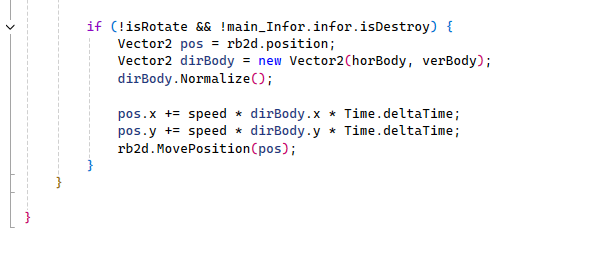


**Main\_Head\_Controller** quản lý chuyển động và hành động của đầu nhân vật. Nó lấy vị trí chuột và xoay đầu nhân vật (với Animator) để hướng về chuột. Khi nhấn chuột trái, nếu nhân vật có đủ năng lượng, nó tạo ra một viên đạn (dùng prefab projectilePrefab) và bắn về hướng chuột. Âm thanh bắn sẽ phát ra và ngừng sau 0.25 giây.

#### 3.1.3.2.Di chuyển của nhân vật(Main\_Movement )







Lớp **Main\_Movement** quản lý việc di chuyển của nhân vật, kết hợp với các đầu vào từ người chơi và các hiệu ứng hoạt hình. Dưới đây là các tính năng chính của lớp này:

1. Khởi tạo:

* speed: Lấy tốc độ di chuyển của nhân vật từ Main\_Infor.
* animBody: Truyền tham chiếu đến Animator để điều khiển các hoạt ảnh của cơ thể.
* rb2d: Truyền tham chiếu đến Rigidbody2D để xử lý di chuyển vật lý.

2. Di chuyển và hoạt ảnh:

* Sử dụng các phím di chuyển (HorizontalCharacter, VerticalCharacter) để nhận đầu vào từ người chơi.
* Dựa trên đầu vào, nhân vật di chuyển và điều chỉnh hoạt ảnh:
* Di chuyển cơ bản: Khi chỉ có một trục (X hoặc Y) có giá trị khác 0.
* Di chuyển chéo: Khi cả hai trục X và Y có giá trị khác 0.
* Điều khiển hướng quay: Nếu nhân vật quay hoặc di chuyển theo góc, các hoạt ảnh thích hợp được gọi (ví dụ: "Walk\_Rotate").

3. Cập nhật và di chuyển vật lý:

* FixedUpdate điều khiển việc di chuyển vật lý của nhân vật bằng cách sử dụng Rigidbody2D.MovePosition. Nếu không phải trạng thái quay (isRotate là false), nhân vật sẽ di chuyển về hướng đã chọn.
* Tốc độ di chuyển được tính theo speed và thời gian

#### 3.1.3.4.Thiết kế hành vi cho viên đạn(Projectile )



**Lớp Projectile** này quản lý hành vi của một viên đạn hoặc vật thể bay trong trò chơi. Các tính năng chính của lớp này bao gồm:

Khởi tạo:

* Sử dụng Rigidbody2D để điều khiển vật lý.
* Kiểm tra khoảng cách bay tối đa (maxDistanceLife).

Chức năng chính:

* Update(): Kiểm tra và hủy viên đạn nếu bay vượt quá khoảng cách tối đa.
* Shoot(): Quay viên đạn theo hướng shootDir và bắn với lực force.

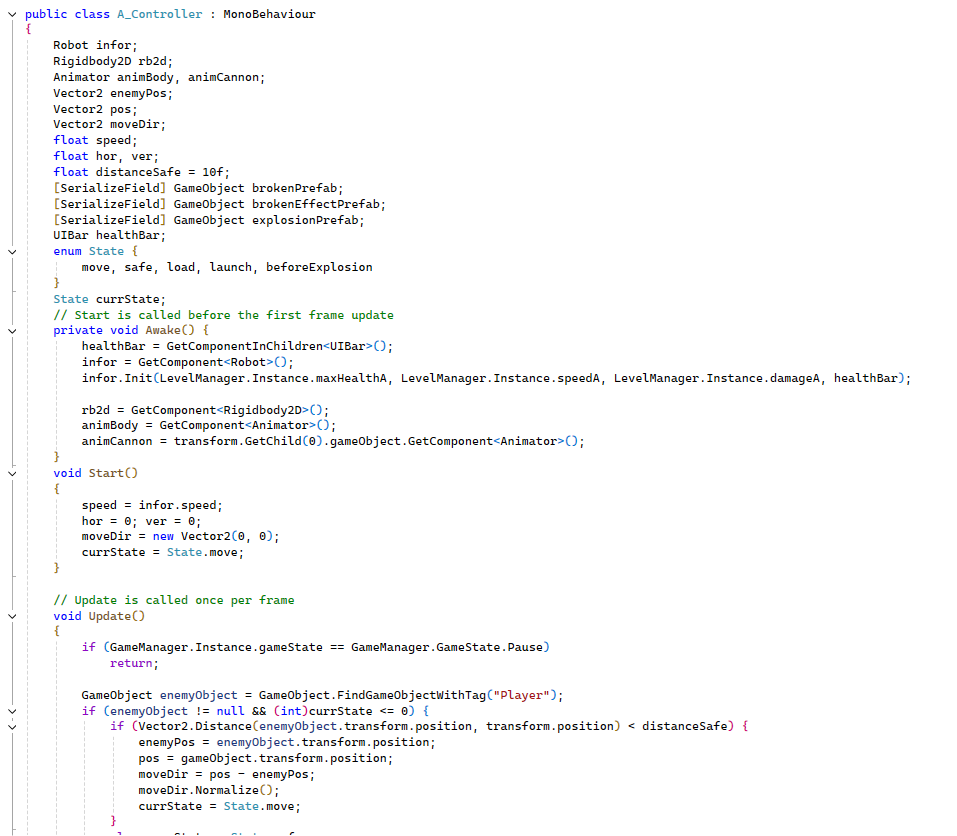
Va chạm:

* OnTriggerEnter2D(): Gây sát thương cho đối tượng Robot khi va chạm và tự hủy viên đạn.

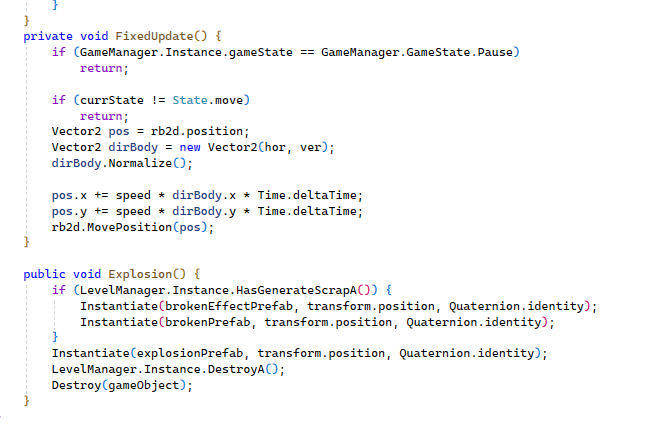
Tóm lại: Viên đạn bay theo hướng xác định, gây sát thương cho Robot và tự hủy sau khi va chạm hoặc bay quá xa.

### 3.1.4.Robot A

#### 3.1.4.1.Điều khiển robot A(A\_Controller )







**Lớp A\_Controller** điều khiển hành vi của robot A trong trò chơi, bao gồm các trạng thái di chuyển, tấn công và phát nổ:

Các thành phần chính:

* Robot: Chứa thông tin sức khỏe, tốc độ và sát thương.
* Rigidbody2D và Animator: Quản lý chuyển động vật lý và hoạt ảnh.
* Các trạng thái: Bao gồm move, safe, load, launch, và beforeExplosion để điều khiển các hành động của robot.

Hoạt động chính:

1. Khởi tạo: Lớp Robot được khởi tạo với các giá trị từ LevelManager. Các thành phần Rigidbody2D, Animator, và UIBar được thiết lập.

2. Cập nhật trạng thái:

* State.move: Robot di chuyển tránh xa người chơi khi gần.
* State.safe: Trạng thái an toàn khi không gần người chơi.
* State.load: Đợi nạp.
* State.launch: Robot thực hiện hành động tấn công.
* State.beforeExplosion: Khi robot bị hủy, bắt đầu hoạt ảnh phát nổ.

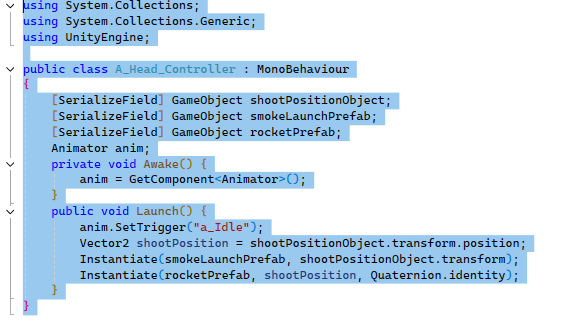
3. Di chuyển: Cập nhật vị trí robot trong FixedUpdate() nếu trạng thái là move.

4. Phát nổ: Khi robot bị hủy, gọi Explosion(), tạo ra hiệu ứng và phá hủy robot.

Tóm tắt:

Lớp này quản lý hành vi của robot đối thủ, điều khiển di chuyển, tấn công và phát nổ tùy vào tình huống trong trò chơi.

#### 3.1.4.2.Điều khiển đầu robot A(A\_Head\_Controller )



Lớp A\_Head\_Controller điều khiển hoạt động của đầu robot đối thủ, đặc biệt là hành động phóng tên lửa.

Các thành phần chính:

* shootPositionObject: Vị trí từ đó tên lửa sẽ được phóng.
* smokeLaunchPrefab: Hiệu ứng khói khi tên lửa được phóng.
* rocketPrefab: Tên lửa được phóng.
* Animator: Quản lý các hoạt ảnh liên quan đến đầu robot.

Hoạt động:

* Awake(): Thiết lập các thành phần cần thiết, bao gồm Animator.
* Launch(): Khi được gọi, phương thức này:

1. Kích hoạt hoạt ảnh a\_Idle để làm cho đầu robot trở lại trạng thái sẵn sàng.

2. Tạo hiệu ứng khói tại vị trí phóng.

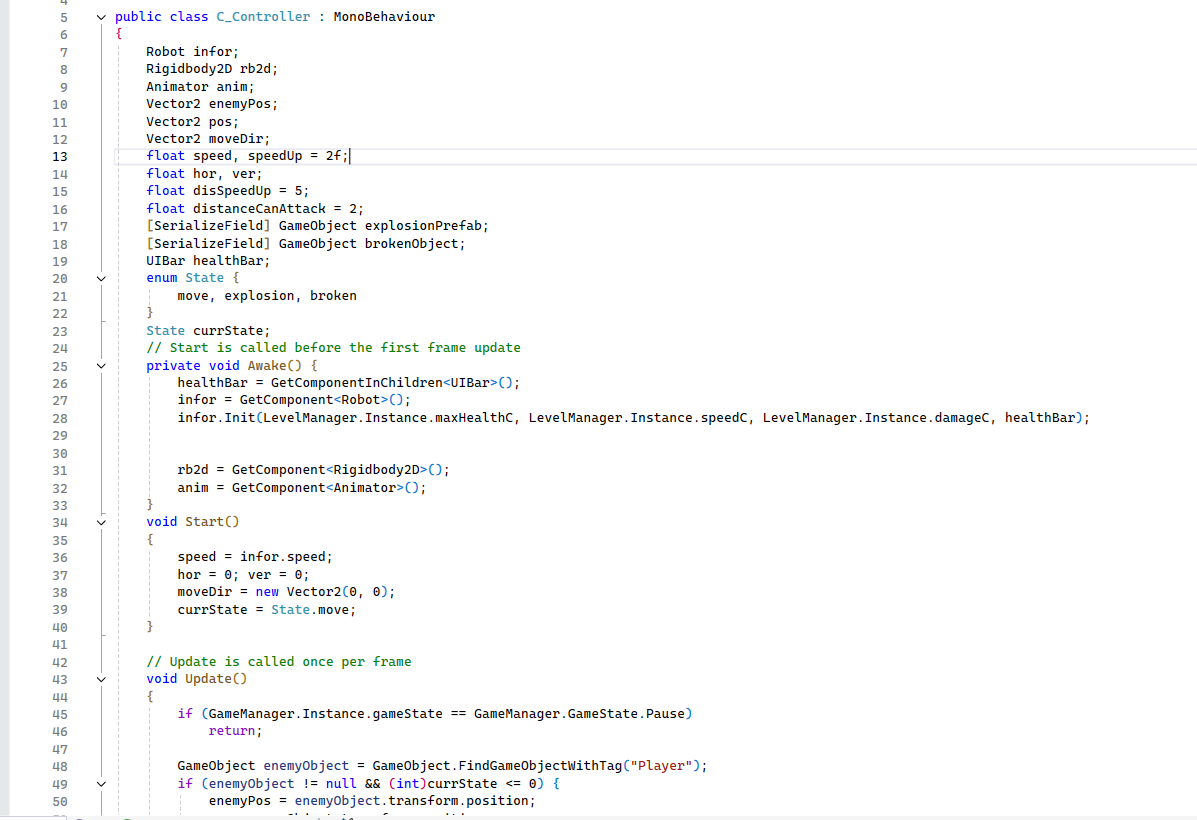
3. Tạo tên lửa tại vị trí phóng và bắn tên lửa.

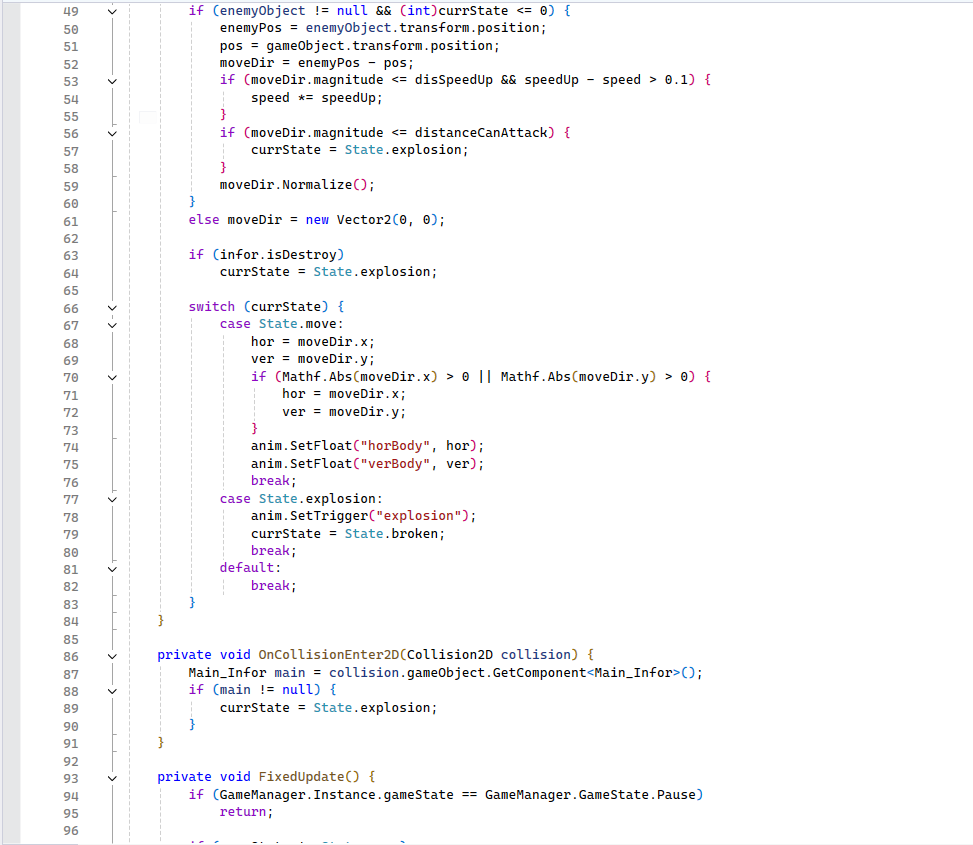
Tóm tắt:

Lớp này quản lý hành động phóng tên lửa của robot đối thủ, bao gồm việc tạo hiệu ứng khói và phóng tên lửa từ vị trí xác định.

### 3.1.5. Robot C

#### 3.1.5.1. Điều khiển cho robot C (C Controller)





Điều khiển di chuyển:

* Robot di chuyển về phía người chơi, tăng tốc nếu ở gần.
* Cập nhật hướng di chuyển và hoạt ảnh tương ứng.
* Phát hiện và tấn công:
* Phát nổ khi đến gần người chơi hoặc va chạm trực tiếp.

Quản lý trạng thái:

* move: Di chuyển.
* explosion: Phát nổ và chuyển sang trạng thái hỏng.
* broken: Robot bị hủy, ngừng hoạt động.

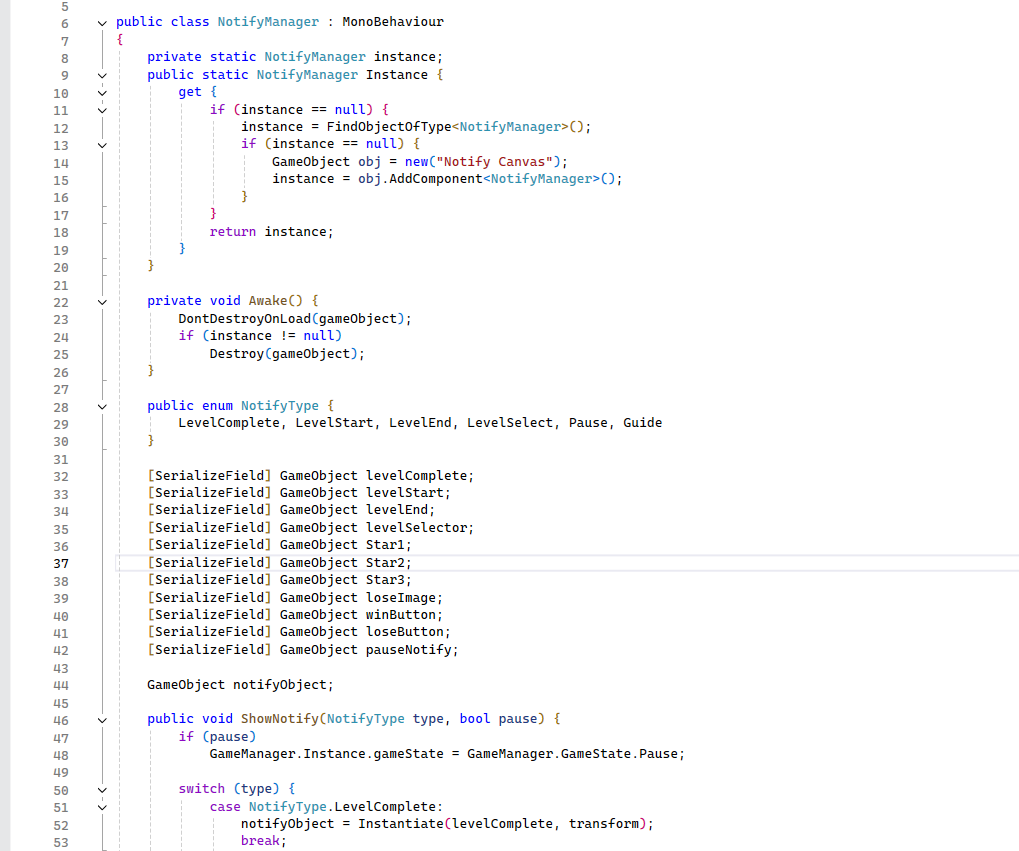
Xử lý khi nổ:

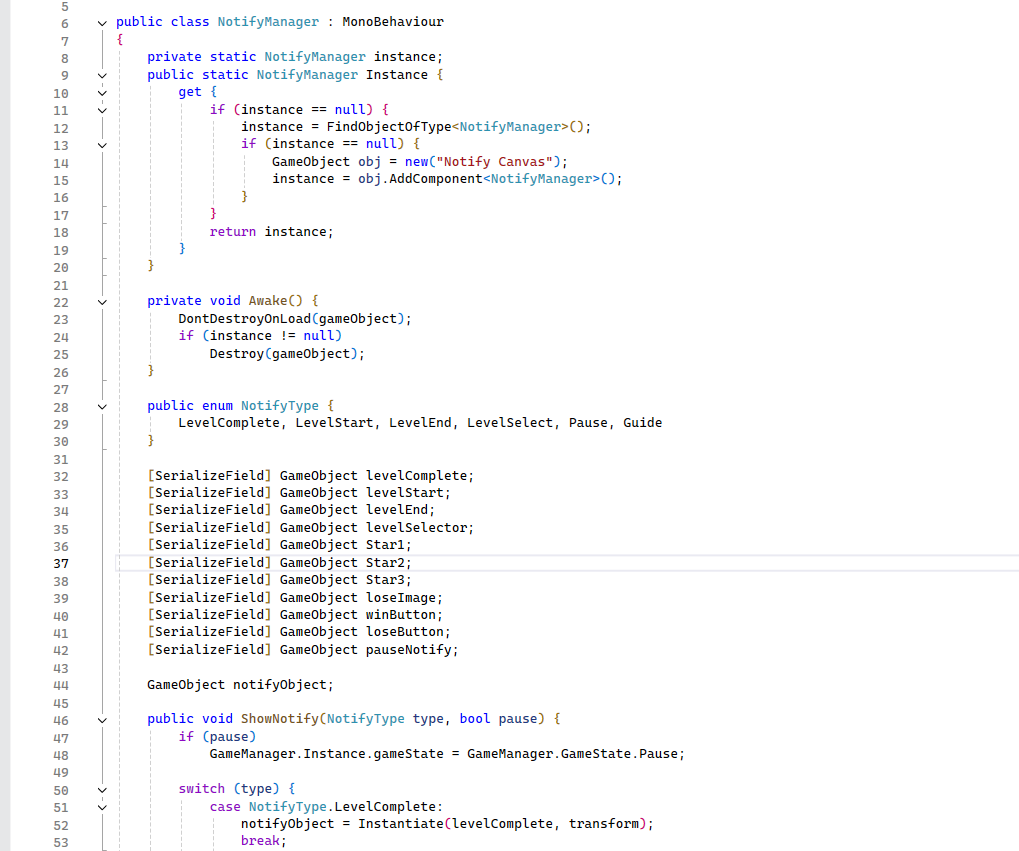
* Tạo hiệu ứng nổ, gây sát thương, sinh phế liệu nếu đủ điều kiện, và giảm số lượng robot còn lại.

Tích hợp hệ thống trò chơi: Tương tác với **GameManager** và **LevelManager** để đồng bộ thông tin cấp độ, sát thương, và trạng thái trò chơi.

### 3.1.6. Xây dựng giao diện người dùng cho game (UI)

#### 3.1.6.1. Quản lý thông báo trò chơi(NotifyManager)





**NotifyManager** quản lý việc hiển thị và ẩn các thông báo quan trọng trong trò chơi như bắt đầu, kết thúc, tạm dừng và hoàn thành cấp độ.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Chức năng chính:

1. Khởi tạo Singleton: Đảm bảo chỉ có một thể hiện **NotifyManager** tồn tại trong suốt trò chơi.

2. Hiển thị thông báo (ShowNotify):

* Hiển thị thông báo phù hợp với tình huống trò chơi.
* Cập nhật trạng thái trò chơi khi cần thiết (tạm dừng, kết thúc).
* Hiển thị số sao nhận được khi hoàn thành cấp độ.

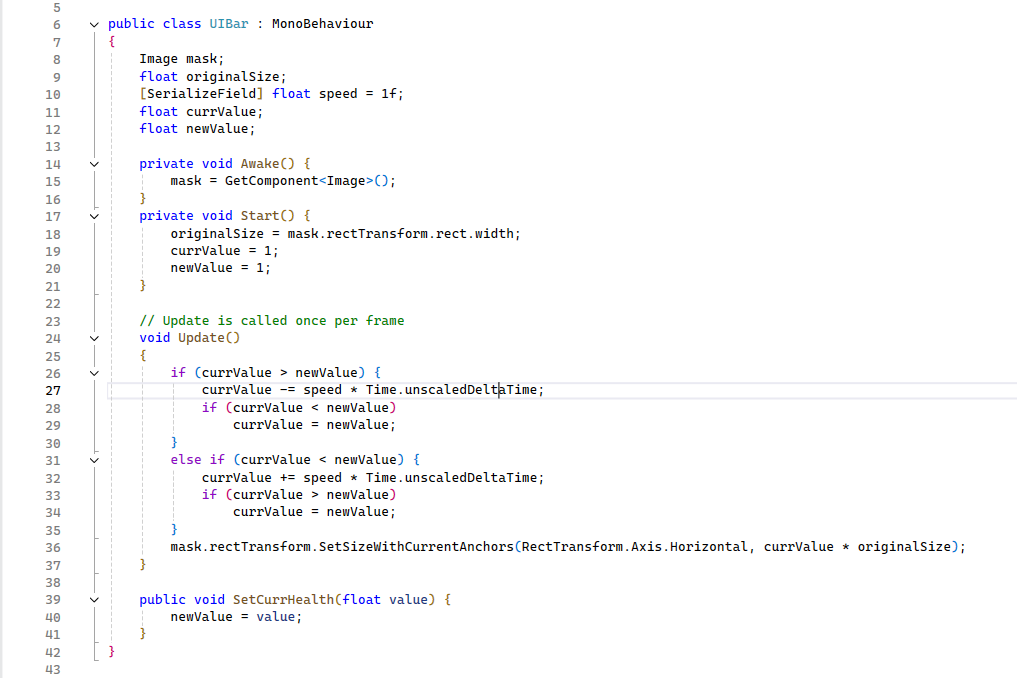
3. Ẩn thông báo (HideNotify): Ẩn thông báo và cập nhật trạng thái trò chơi.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tích hợp:

* Với **GameManager**: Cập nhật trạng thái trò chơi.
* Với **LevelManager**: Lấy số sao để hiển thị khi hoàn thành cấp độ.

#### 3.1.6.2.Quản lý trạng thái thanh hiển thị (UIBar)



UIBar quản lý thanh hiển thị trạng thái như máu, năng lượng,... bằng cách điều chỉnh kích thước thanh theo giá trị hiện tại.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Chức năng chính:

1. Khởi tạo:

* Lưu kích thước ban đầu của thanh (originalSize).
* Đặt giá trị khởi đầu cho thanh là đầy (currValue = 1).

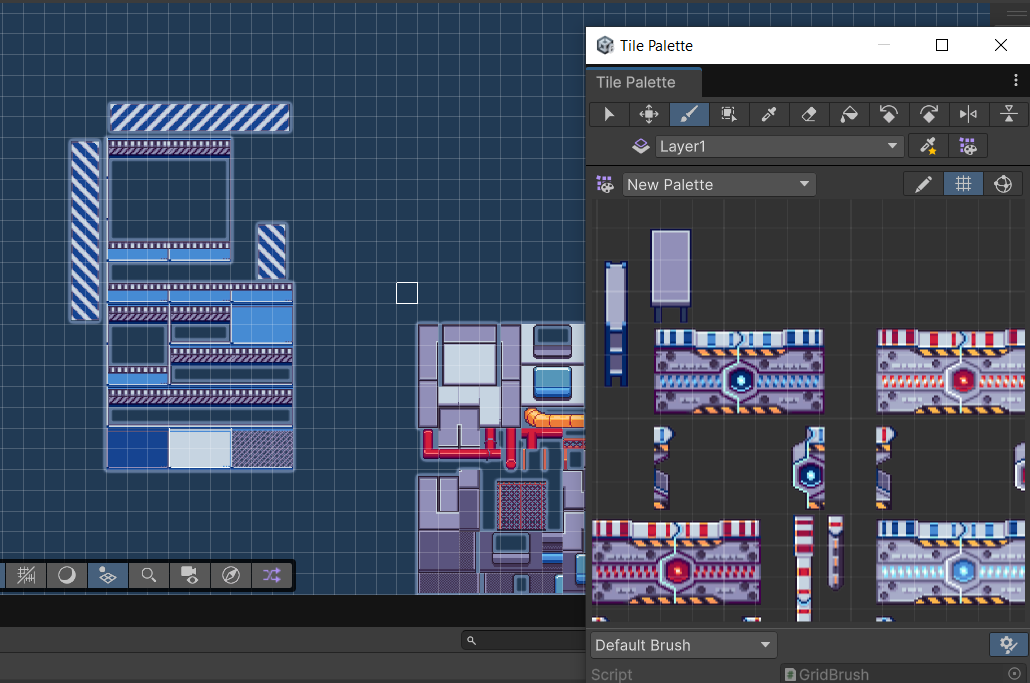
2. Cập nhật thanh (Update):

* Điều chỉnh dần kích thước thanh để khớp với giá trị mục tiêu (newValue).
* Tăng hoặc giảm giá trị hiện tại (currValue) dựa trên tốc độ cập nhật (speed).
* Cập nhật kích thước thanh bằng SetSizeWithCurrentAnchors.

1. Cài đặt giá trị mới (SetCurrHealth): Đặt giá trị mục tiêu mới (newValue) cho thanh trạng thái.

### 3.1.7. Kỹ thuật thiết kế ảnh (Tilemap)

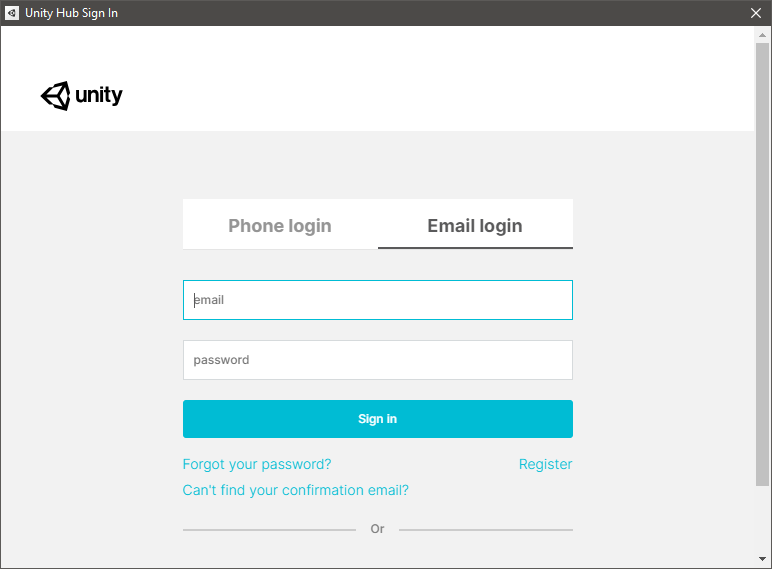
Trong phần mềm Unity: Chọn cửa sổ Window > 2D > Tile palette > Create New Palette > chọn vị trí lưu > Add các tilemap đã tải hoặc tự tạo vào Tile Palette đã tạo > Chọn tilemap phù hợp rồi vẽ lên map.



*Hình 3.1: Các tilemap trong game*

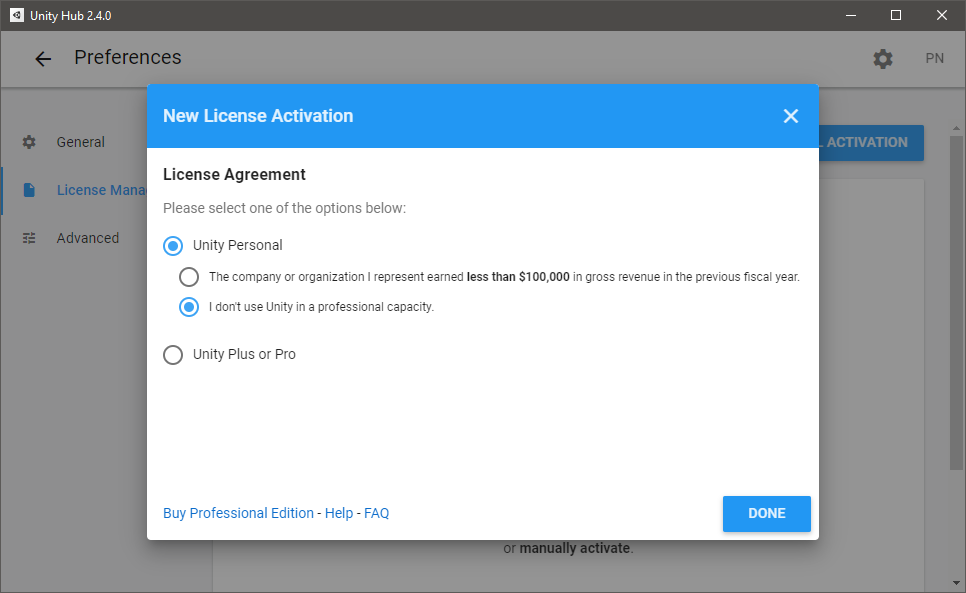
## 3.2 Hướng dẫn cài đặt

Vào trang Download chọn Download Unity Hub, sau khi tải xuống hoàn tất double click vào file đã tải. Việc cài đặt Unity Hub diễn ra bình thường và tương tự như cài đặt các chương trình khác. Sau khi cài đặt, khởi động Unity Hub rồi đăng nhập vào tài khoản Unity, tài khoản Unity có thể được tạo miễn phí tại trang chủ.



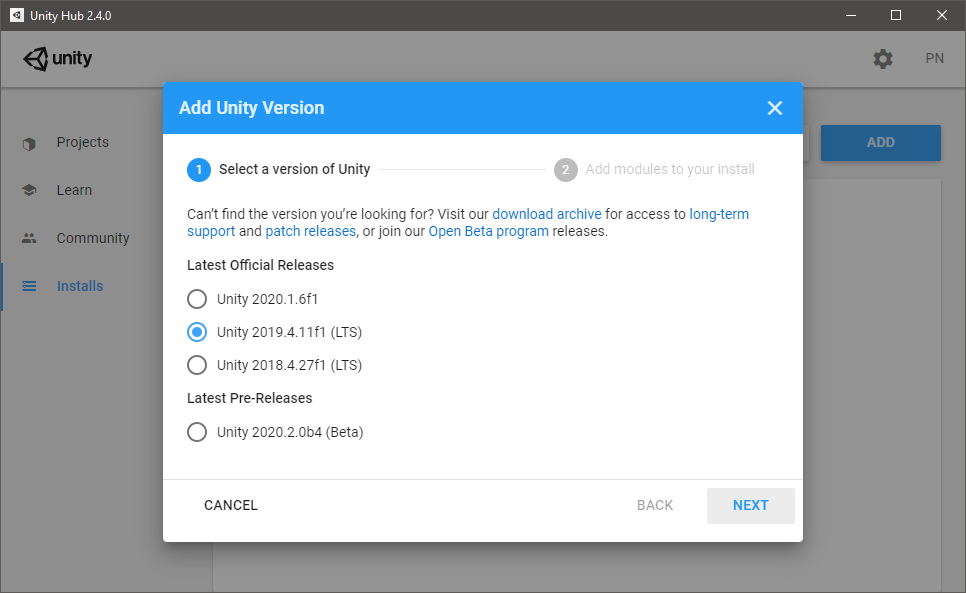
Hình 3.2: Cửa sổ đăng nhập

Sau khi đăng nhập trong trường hợp chưa có License thì phải tạo một License, vào Preferences bên cạnh profile góc trên phải và chọn License Management và chọn như hình để kích hoạt 1 License miễn phí.



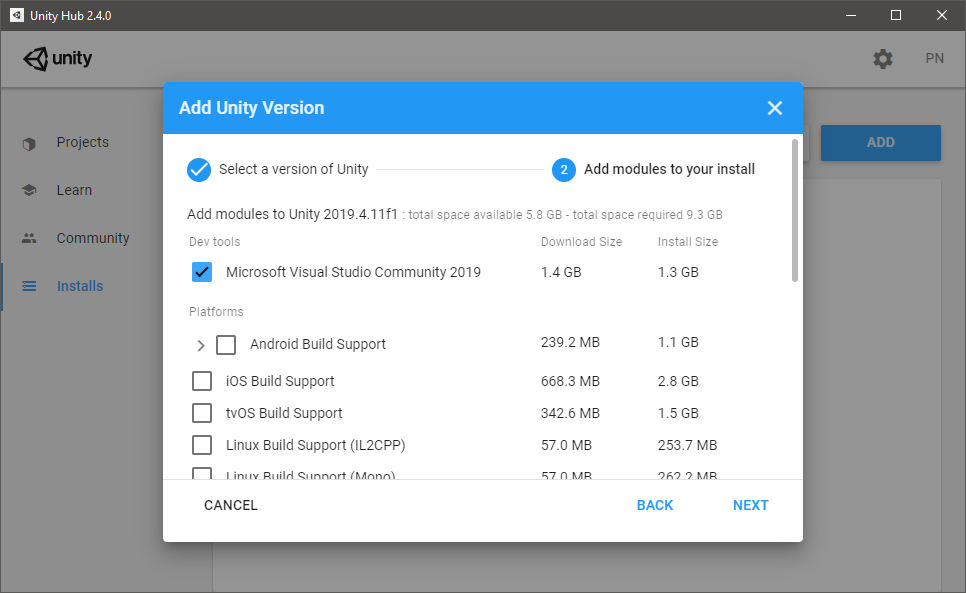
Hình 3.3: Kích hoạt License

Quay ra màn hình chính, chọn Installs - > ADD để cài đặt Unity.



Hình 3.4: Chọn phiên bản Unity

Chọn phiên bản Unity và cài đặt các component dựa vào nền tảng muốn phát triển



Hình 3.5: Cài đặt các conpnent

Việc cài đặt đã xong.

## 3.2 Đánh giá kết quả

Game Robot Survival được nhóm phát triển khá thành công. Tuy đây là một trò chơi khá đơn giản nhưng nhóm cũng đã hoàn thiện game ở phiên bản cuối cùng.

## 3.3 Hướng phát triển

Theo nhóm thì game vẫn còn khá đơn giản, cần cải thiện thêm một số cái như:

* Thêm nhiều chọn lựa nhân vật để chơi game hơn.
* Thêm tính năng tính điểm online để tăng sự ganh đua giữa những người chơi.

# KẾT LUẬN

Tóm lại qua nghiên cứu cùng với sự giúp đỡ tận tình của thầy ***Nguyễn Văn Tỉnh*** nhóm đã hoàn thành bài tập lớn: “***Xây dựng game Robot Survival****”.* Qua đây bản các thành viên trong nhóm đã học hỏi được nhiều kinh nghiệm như: làm việc nhóm một cách khoa học, hiệu quả, biết được thêm về xây dựng game và tạo ra game Robot Survival góp phần vào ngành công nghiệp game nước nhà. Ngoài ra chúng em còn biết thêm về cách viết kịch bản game, thiết kế game, unity...

Bên cạnh đó, với những kiến thức nhận được nhóm chúng em đã hoàn thành đề tài này. Tuy nhiên trong quá trình làm có rất nhiều điều hạn chế mà chúng em chưa nhận biết được. Chúng em rất mong có được những nhận xét, đánh giá từ phía các bạn và thầy để có cái nhìn nhận chính xác hơn về những hạn chế của nhóm chúng em.

Chúng em xin chân thành cảm ơn!

# TÀI LIỆU THAM KHẢO

1. <https://codelearn.io/sharing/lap-trinh-game-voi-unity-nhu-the-nao-phan-1>
2. <https://viblo.asia/p/gioi-thieu-ve-unity-engine-game-engine-pho-bien-nhat-hien-nay-V3m5WBj8lO7>
3. https://learn.unity.com/