LỜI NHẬN XÉT

Nhận xét và đánh giá điểm của người hướng dẫn:

- Ý thức thực hiện:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

- Nội dung thực hiện:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

- Hình thức trình bày:

.....

.....

.....

.....

.....

.....

.....

.....

- Tổng hợp kết quả:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Nha Trang, ngày…tháng…năm 2019

Người hướng dẫn

(Kí và ghi rõ họ tên)

LỜI NÓI ĐẦU

Chúng em xin chân thành cảm ơn Khoa Công Nghệ Thông Tin, trường Đại học Thông tin liên lạc đã tạo điều kiện tốt cho chúng em thực hiện đề tài này!

Cùng với sự giúp đỡ, hướng dẫn tận tính của Giảng viên – thầy Đỗ Văn Tuấn, chúng em đã phát triển nên Game Just Run lấy bối cảnh (MAP) một người lạc trong rừng bị nhiều động vật nguy hiểm truy đuổi phải tìm cách chạy trốn và tránh các vật cản. Hứa hẹn sẽ mang đến những trải nghiệm mới lạ, giúp người chơi thư giãn, giải trí sau những giờ học tập lao động căng thẳng. Game còn là công cụ rèn luyện kỹ năng phản xạ, tính tập trung, khả năng tính toán suy nghĩ để có thể hoàn thành thử thách khó nhất của trò chơi.

Do thời gian còn hạn hẹp cũng như chưa có kinh nghiệm làm một game lớn, nền tảng Framework mới xuất hiện trong những năm gần đây, chương trình thành phẩm của chúng em có thể còn tồn tại lỗi không mong muốn. Rất mong nhận được góp ý của thầy để chúng em hoàn thiện hơn và rút kinh nghiệm cho những đồ án, dự án lớn hơn trong tương lai.

Một lần nữa, chúng em xin chân thành cảm ơn thầy và chúc thầy nhiều sức khỏe!

DANH MỤC CÁC BẢNG, HÌNH

DANH MỤC CÁC BẢNG

DANH MỤC CÁC HÌNH

MỤC LỤC

CHƯƠNG 1: TỔNG QUAN

1.1. Lý do chọn đề tài

Xã hội ngày càng phát triển, nhu cầu giải trí của con người ngày càng tăng cao. Cùng với sự phát triển của công nghệ thông tin, điện tử, các thiết bị chơi game ngày càng phổ biến. Trong những năm gần đây ngành công nghệ game nổi lên và đem lại doanh thu khổng lồ. Có thể nói game không chỉ là một phương tiện giải trí cơ bản, nó còn là một lĩnh vực đáng quan tâm và theo đuổi. Thiết kế game là một công việc không dễ dàng đòi hỏi rất nhiều thời gian và công sức, tuy nhiên qua đó ta có thể biến những dòng code khô khan thành những trải nghiệm thú vị cho người chơi, đồng thời có thể học hỏi những kiến thức mới về nền tảng Unity Framework và củng cố ngôn ngữ C#

1.2. Đối tượng nghiên cứu

Nền tảng Unity Engine.

Cách thiết kế nhân vật, tạo bản đồ, khung cảnh, character animator, xử lý va chạm, healthy player và enemy, điểm số màn chơi, tạo AI cho enemy, xử lý hiệu ứng hình ảnh, âm thanh.

1.3. Giới thiệu cốt truyện

Just Run là một trò chơi "chạy không ngừng nghỉ". Ở đó người chơi chỉ di chuyển lên xuống xuyên suốt trò chơi. Trò chơi lấy bối cảnh một chàng trai phiêu lưu lạc trong rừng phải tìm cách thoát ra khỏi khu rừng ấy. Tuy nhiên mọi thứ không hề dễ dàng như chàng trai đã nghĩ mà ở đó anh ấy gặp không ít trở ngại phải vượt qua rất nhiều thú ăn thịt nguy hiểm như hổ, bò cạp, đại bàng … Cùng với đó là các chướng ngại vật cần vượt qua. Vậy chàng trai có thoát ra khỏi khu rừng quỷ dị này được không? Người chơi hãy vào vai chàng trai để viết tiếp cốt truyện nhé!

1.4. Ý nghĩa của trò chơi

1.4.1 Ý nghĩa dành cho người chơi

Trò chơi hứa hẹn sẽ mang đến trải nghiệm mới lạ, giúp người chơi thư giãn, giải trí sau những giờ học tập lao động căng thẳng. Game còn là công cụ rèn luyện kỹ năng phản xạ, tính tập trung, khả năng tính toán suy nghĩ để có thể hoàn thành thử thách khó nhất của trò chơi.

Những màn rượt đuổi gay cấn, những con thú với hình dạng đáng sợ cùng với tiết tấu game nhanh đã khiến cho Just Run trở thành một món ăn tinh thần có thể gây nghiện với những tín đồ game nhập vai. Hãy cùng nhau chia sẽ thành tích cao nhất của bạn nhé!

1.4.2 Ý nghĩa dành cho sinh viên nói chung

Hiện nay, sinh viên còn rất hạn chế trong việc tiếp cận các nền tảng, ngôn ngữ lập trình game. Vì thế, đề tài này sẽ giúp sinh viên có thêm hứng thú cũng như đam mê với lập trình game để có thể tự mình thiết kế một trò chơi hoàn chỉnh.

1.4.3 Ý nghĩa dành cho nhóm chúng em nói riêng

+ Giúp cho bản thân tiếp cận được với nền tảng lập trình mới (Unity).

+ Bản thân nâng cao trình độ lập trình C#, thiết kế.

+ Biết được các cách tạo Art, Audio, Animation, Particle.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Tổng quan về Unity engine

2.1.1. Unity là gì?

Unity là một "cross-flatform game engine" tạm hiểu là công cụ phát triển game đa nền tảng được phát triển bởi Unity Technologies. Game engine này được sử dụng để phát trển game trên PC, consoles, thiết bị di động (Android, IOS) và trên websites.

Hình 1: Phần mềm Unity

Unity3D là phần mềm làm games trực tiếp theo thời gian thực, mà không cần render, cho phép người design game có thể thiết kế InterfaceGraphic, map hay character, … từ một phần mềm thứ 2 (thường là các phần mềm thiết kế đồ họa chuyên nghiệp như Adobe Photoshop 3Dsmax, Blender Maya, XSL, Cinema4D, Cheetah3D, Modo, Autodesk FBX, LightWave…) sau đó chỉ việc import nó vào trong Unity với định dạng của tập tin là *.FBX hay *.dae, *.3DS, *.dxf và *.obj…

2.1.3. Quá trình phát triển của Unity

Ra mắt đầu tiên vào năm 2005 tại sự kiện Apple's Worldwide Developer Conference bởi nhà sáng lập David Helgason, trải qua nhiều năm phát triển, Unity đã có version 5.5 hoàn thiện hơn về rất nhiều mặt. Tháng 5-2012 theo cuộc khảo sát Game Developer Megazine được công nhận là Game engine tốt nhất cho mobile. Năm 2014 Unity thắng giải "Best Engine" tại giải UK's annual Develop Industry Exellence. Phiên bản mới nhất hiện nay là 2018.2 với nhiều tính năng nổi trội.

2.1.4. Một số thống kê về Unity

Tính đến quý 3 năm 2016 đã có 5 tỉ lượt download game và ứng dụng được phát triển bởi Unity

2,4 tỉ thiết bị di động đã từng tải ít nhất 1 ứng dụng bởi unity.

Trong top 1000 game Mobile miễn phí thì số lượng game tạo ra bởi Unity chiếm tới 34%

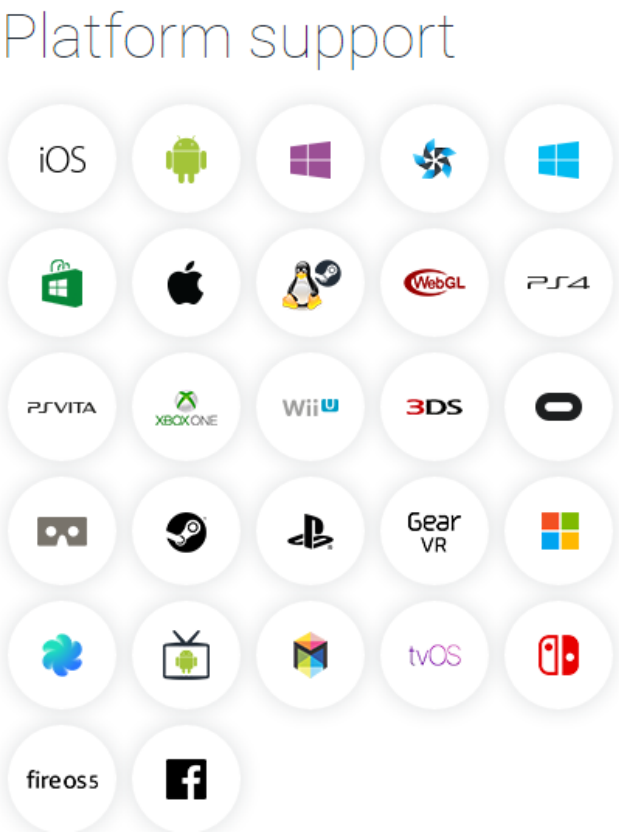
Số lượng người dùng (gamer) của Unity đạt tới con số 770 triệu, trong khi đó số người thường xuyên sử dụng Twitter là 310 triệu người.

(nguồn: thống kê từ internet)

2.1.5. Ưu điểm của Unity

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

Hỗ trợ đa nền tảng: Một trong các thế mạnh của Unity3D chính là khả năng hỗ trợ gần như toàn bộ các nền tảng hiện có bao gồm: PlayStation 3, Xbox 360, Wii U, iOS, Android, Windows, Blackberry 10, OS X, Linux, trình duyệt Web và cả Flash. Nói cách khác, chỉ với một gói engine, các studio có thể làm game cho bất kỳ hệ điều hành nào và dễ dàng convert chúng sang những hệ điều hành khác nhau. Đồng thời, đây cũng là giải pháp cho các game online đa nền tảng – có thể chơi đồng thời trên nhiều hệ điều hành, phần cứng khác nhau như Web, PC, Mobile, Tablet….



Hình 2: Các nền tảng Unity hỗ trợ

Dễ sử dụng: Unity3D được xây dựng trong một môi trường phát triển tích hợp, cung cấp một hệ thống toàn diện cho các lập trình viên, từ soạn thảo mã nguồn, xây dựng công cụ tự động hóa đến trình sửa lỗi. Do được hướng đến đồng thời cả lập trình viên không chuyên và studio chuyên nghiệp, nên Unity3D khá dễ sử dụng. Hơn nữa, đây là một trong những engine phổ biến nhất trên thế giới, người dùng có thể dễ dàng tìm kiếm kinh nghiệm sử dụng của "tiền bối" trên các forum công nghệ.

Tính kinh tế cao: Unity Technologies hiện cung cấp bản miễn phí engine Unity3D cho người dùng cá nhân và các doanh nghiệp có doanh thu dưới 100.000 USD/năm. Với bản Pro, người dùng phải trả 1.500 USD/năm – một con số rất khiêm tốn so với những gì engine này mang lại.

2.2. Các khái niệm trong Unity

2.2.1. Asset

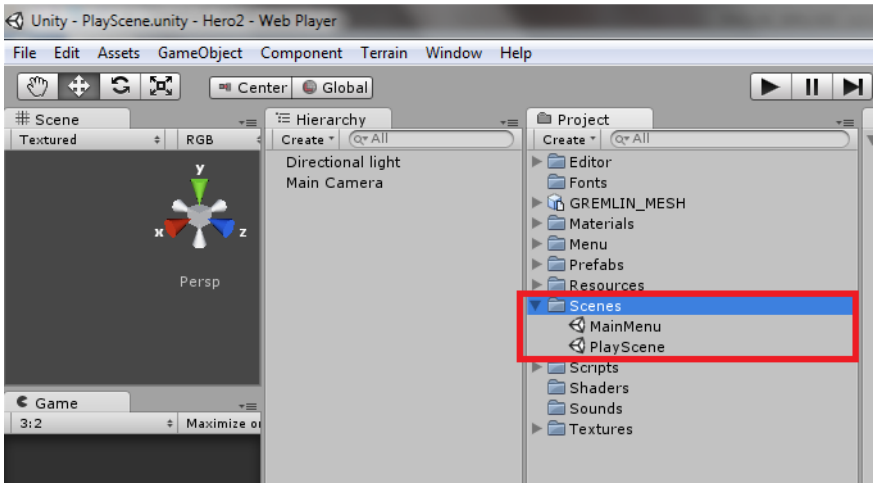
Đây là kho tài nguyên cho việc xây dựng game trong một project của Unity. Các tài nguyên này có thể là hình ảnh, âm thanh, hoặc một mô hình 3D có sẵn. Unity sẽ tham chiếu đến các tập tin chúng ta sẽ sử dụng để tạo ra các tài nguyên cho trò chơi. Đây là lý do tại sao trong bất

kỳ thư mục chứa project sử dụng Unity thì tất cả các tập tin tài nguyên phải được lưu trữ trong một thư mục con tên là Assets.

2.2.2. Scene

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.



Hình 3: Giao diện Project

2.2.3. Game Object

Khi một tài nguyên được sử dụng trong một scene, khi đó chúng ta có thể coi tài nguyên này là một "Game Object" mới. Mỗi GameObject phải chứa ít nhất một thành phần, đó là thành phần "Transform". Transform chứa các phép để biến đổi góc quay, tỷ lệ hay tịnh tiến của đối tượng.

2.2.4. Component

Component là các thành phần trong một Game Object của Unity. Bằng cách đính kèm các thành phần vào cho một đối tượng, chúng ta có thể áp dụng ngay các phần mới của game engine vào đối tượng. Thông thường các thành phần này được Unity xây dựng sẵn như ánh sáng, camera, particle, hiệu ứng vật lý, …

2.2.5. Script

Script là thành phần quan trọng nhất trong Unity, có thể xem scripts như là linh hồn của game. Chúng ta có thể viết kịch bản cho game bằng C#, Java Scripts hoặc Boo (một dẫn xuất

của ngôn ngữ Python). Theo nhiều người đã sử dụng Unity thì code bằng C# sẽ giúp game chạy nhanh hơn và giúp kiểm soát code tốt hơn do tất cả các biến phải được khai báo rõ ràng. Mặt khác ngôn ngữ C# rất tiện dụng để lập trình, nên trong thành phẩm Game Just Run 2D ở chương 4, chúng em dùng ngôn ngữ C# để viết kịch bản cho game. Mỗi file script C# là một class bắt buộc kế thừa từ lớp MonoBehaviour, có tên class phải trùng với tên file script.

Một đoạn script muốn thực thi được thì nó phải được gắn vào một đối tượng.

2.2.6. Prefab

Prefab là một bản sao lưu các vật thể chúng ta đã tạo, bao gồm các kịch bản cho hành động (khởi tạo, di chuyển, hay hủy đối tượng). Ta có thể sử dụng đối tượng này nhiều lần trong trò chơi, và cũng có thể sử dụng lại cho project khác. Prefab cho phép chúng ta lưu trữ các đối tượng, toàn bộ thành phần bên trong và cấu hình hiện tại.

2.2.7. UnityAPI

UnityAPI chứa rất nhiều lớp hỗ trợ lập trình game, trong đó có một số lớp quan trọng như:

MonoBehaviour: tất cả các script muốn gắn vào một đối tượng game bắt buộc phải kế thừa từ lớp này.

GameObject: lớp cha của tất cả các thực thể trong scene.

Component: lớp cha của tất cả các thành phần có thể gắn vào đối tượng.

Transform: giúp thay đổi vị trí, xoay, biến đổi tỉ lệ mô hình.

Input: hỗ trợ lập trình với chuột, cảm ứng đa điểm, cảm biến gia tốc.

Camera: giúp lập trình camera.

Light: giúp tạo ánh sáng trong game.

Projector: giúp chiếu texture lên bề mặt vật thể.

ParticleEmitter: hỗ trợ tạo các hiệu ứng particle đẹp mắt.

Audio: hỗ trợ lập trình với âm thanh.

Animation: chạy chuyển động của mô hình nhân vật.

Rigidbody: giúp tạo hiệu ứng vật lý liên quan đến trọng lực như bóng nẩy, lăn, ...

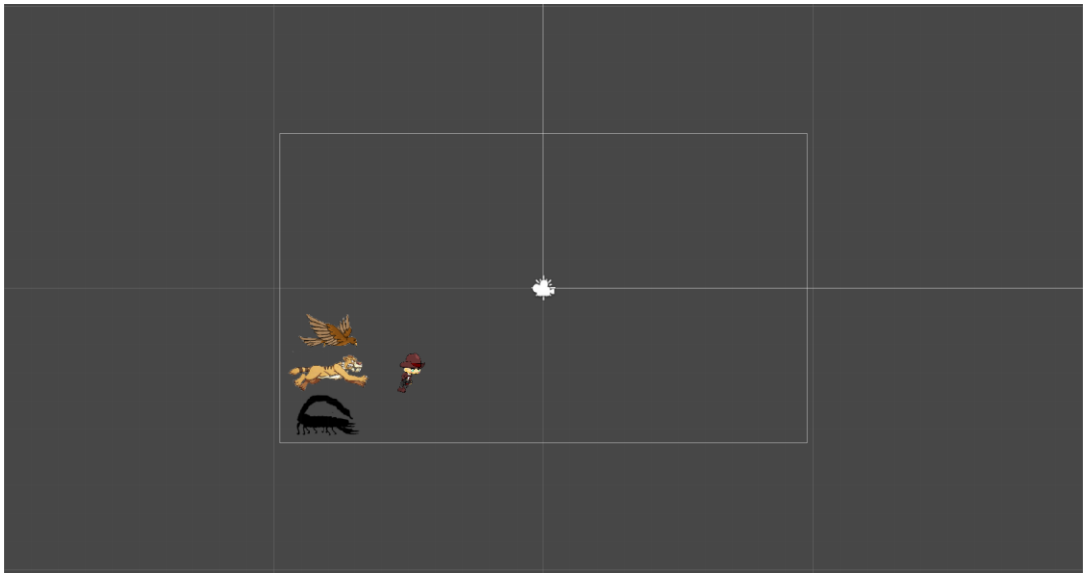
Collider: hỗ trợ lập trình va chạm giữa các vật thể.

GUI: giúp lập trình giao diện người dùng trên Unity.

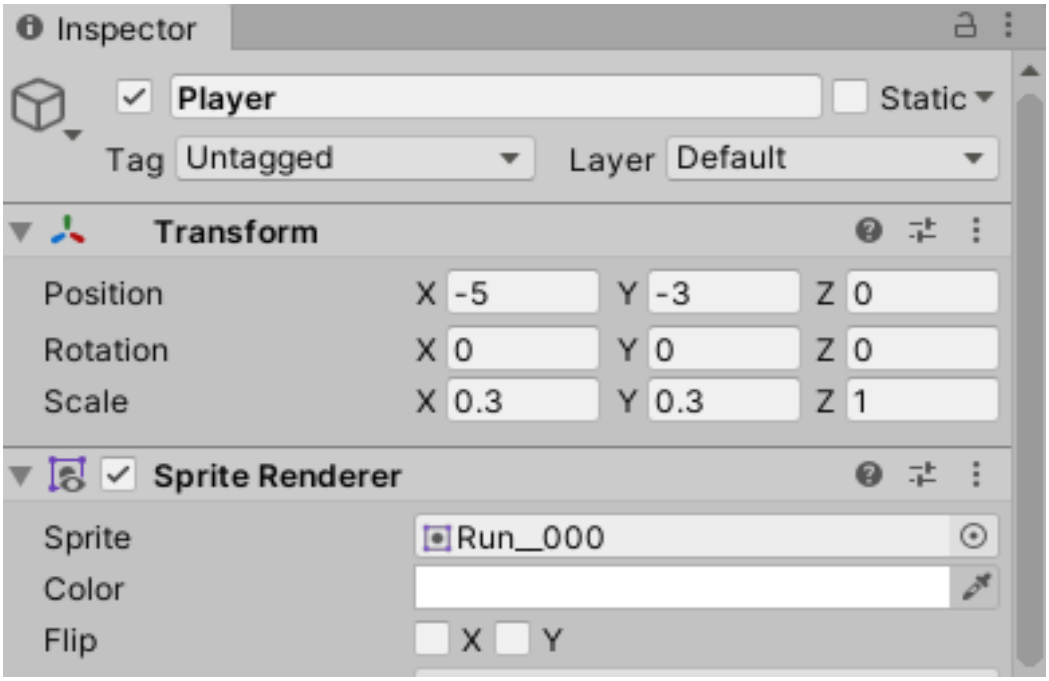
2.2.8. Không gian 2D và vector trong không gian

Nhắc về hệ tọa độ không gian 2 chiều chắc hẳn chúng ta đều nhớ môn hình học phẳng hoặc đã từng sử dụng một công cụ dựng hình 2D nào đó. Hệ tọa độ không gian 3 chiều bao gồm 3 trục X, Y, Z hay hiểu đơn giản là chiều ngang (X) chiều cao (Y) và chiều sâu (Z). Chúng được kí hiệu theo cú pháp: (X, Y, Z) nhưng trong 2D chúng ta chỉ dùng 2 trục (X, Y).

Trong không gian 3 chiều, tồn tại 1 điểm gọi là Origin hoặc Word Zero. Đây là điểm có tọa độ (0,0,0). Tất cả tọa độ của các đối tượng tồn tại trong không gian đều có mối liên hệ với điểm này, cách tính này là theo Word Space. Tuy nhiên, để việc tính toán đơn giản, người ta sử dụng thêm một khái niệm gọi là Local Space nhằm xác định tọa độ của đối tượng này so với đối tượng khác. Mối quan hệ này gọi là parent – child.

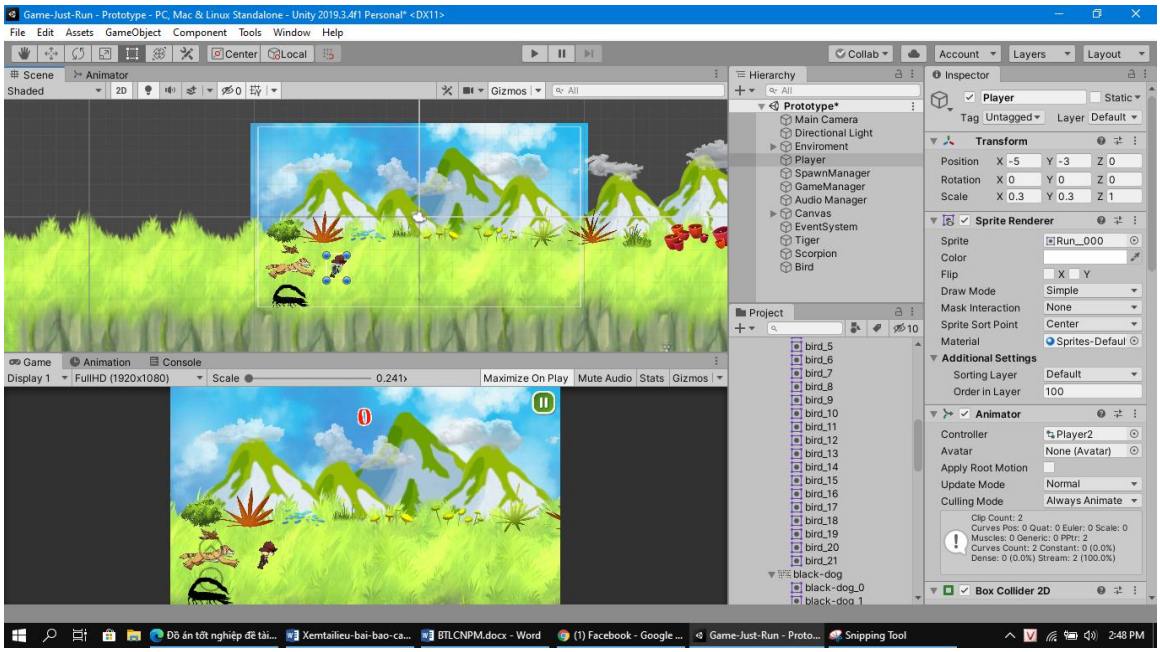


Hình 4: Animal and Player trong chiều không gian 2D



Hình 5: Tương ứng với Rotation X, Y, Z (mặc định)

2.3. Cửa sổ làm việc



Hình 6: Cửa sổ làm việc

2.3.1. Project

Mỗi Project của Unity đều chứa một thư mục Assets. Nội dung của thư mục này được hiển thị trong Project View. Đây là nơi chứa tất cả các assets để tạo Game như Scenes, Script, 2D models, Textures, Audio, Prefabs. Chúng ta không nên di chuyển các assets bằng sử

dụng Window Explorer (hay Finder trong MAC), vì nó sẽ phá vỡ cấu trúc siêu dữ liệu của assets với nhau, và luôn sử dụng Project View để cấu trúc assets

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.

2.3.2. Hierarchy

Trong Hierarchy chứa các GameObject hiện thời, một số có thể trỏ trực tiếp tới những file assets như 3D models, một số khác đại diện cho Prefabs – những đối tượng đã được tùy biến, dùng làm các công việc khác nhau sau này trong Game. Chúng ta có thể chọn và parenting Object trong Hierarchy. Một Object có thể được thêm vào hay loại bỏ trong scene và có thể thấy nó mất đi hay xuất hiện trong Hierarchy.

2.3.3. Parenting

Tức là thư mục chứa hay thư mục gốc, bất kỳ một game object nào muốn là đối tượng con (child) thì ta chỉ việc kéo thả đối tượng đó vào trong đối tượng dự tính làm Parenting trong Hierarchy và nó sẽ kế thừa chuyển động và quay của parenting.

2.3.4. Toolbar

Toolbar chứa 5 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.

- Transform Gizmo Toggles: dung cho việc thể hiện Scene view.

- Play/Pause/Step Buttons: dung cho view game, chạy game ngay trong Editor để kiểm tra.

- Layer Drop-down: kiểm soát đối tượng nào đang được thục hiện trong Scene view

- Layout Drop-down: kiểm soát sự sắp xếp cảu các Views.

2.3.5. Scene View

Là nơi thiết kế Game, đối tượng Maneuvering và Importanting trong Scene view (chuyển động và điều khiển) là hai trong số các chức năng quan trọng của Unity, ở góc bên phải của Scene là Scene Gizmo, nó thể hiện hướng nhìn trong không gian của camera trong Scene View hiện thời, cho phép thay đổi góc nhìn trực quan và nhanh chóng.

Click lên các nhánh hình nón để chuyển qua các góc nhìn khác nhau có thể xem ở chế độ Isometric Model (tức ở dạng mặt cắt hai chiều), để chuyển qua chế độ 3D, click vào hình vuông ở giữa hay giữ phím Shift + Click để chuyển đổi chế độ nhìn.

Khi xây dựng một Game, chúng ta sẽ đặt rất nhiều đối tượng vào trong Game, khi đó ta có thể dử dụng các công cụ Transform Tools ở trong Toolbar để di chuyển, xoay, phóng to thu nhỏ từng đối tượng. Nếu chọn một đối tượng trong Scene View, xung quanh đối tượng được chọn sẽ có những thay đổi tương ứng với từng chế độ trong Transform Tools. Ta có thể thay đổi đối tượng tùy ý, nếu muốn chính xác, có thể chỉnh chi tiết ở bảng Inspector.

2.3.6. Game View.

Game View được rendered 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 quết định những gì mà người chơi sẽ nhìn thấy khi họ chơi Game.

2.3.7. Play Mode

Sử dụng những nút trên Toolbar để điều khiển Editor Play Mode, và xem trước Game sẽ như thế nào khi chơi. Trong chế Play, mọi giá trị thay đổi sẽ được lưu tạm, và bị xóa khi thoát khỏi chế độ play.

2.3.8. Inspector

Games trong Unity được tạo ra bởi tập hợp rất nhiều GameObject, trong đó bao gồm meshes, scripts, âm thanh, hay những đối tượng Graphic như nguồn sáng v..v... Inspector sẽ hiển thị mọi thông tin về đối tượng đang làm việc một cách chi tiết, kể cả những Components được đính kèm và những thuộc tính của nó. Tại đây ta có thể điều chỉnh, thiết lập mọi thông số chức năng của những mối liên kết GameObject-Component.

Mọi thuộc tính thể hiện trong Inspector đều có thể được tùy biến một cách trực tiếp. Ngay cả với những biến trong script cũng có thể được hiệu chỉnh mà không cần xem mã.

Trong script, nếu chúng ta định nghĩa một giá trị là public cho một kiểu đối tượng (như GameObject hay Transform), ta có thể drag-drop một GameObject hay một Prefab vào trong Inspector để gán giá trị cho nó.

Chúng ta có thể click lên icon hình bánh răng nhỏ bên phải hay click chuột phải lên tên cảu Component để xuất hện context menu dành cho những thiết lập của Component.

Inspector cũng sẽ thể hiện mọi thông số Import Setting của assets đang làm việc.

2.4. Các phần mềm hỗ trợ khác

2.4.1. Photoshop

Adobe Photoshop là phần mềm đồ họa có tác dụng chính là chỉnh sửa ảnh. Được phát triển và phát hành bởi Adobe năm 1988. Hiện nay PS được xem là phần mềm đồ họa bitmap mạnh nhất trên thị trường, và được hầu hết những người làm thiết kế đồ họa cũng như các photography sử dụng. Photoshop cũng được đưa vào giáo trình giảng dạy chính thức của các trường đào tạo nghề cũng như các trường cao đẳng, đại học.

Photoshop giúp ta tạo ra các bản vẽ nhân vật, cắt ghép các chi tiết, tạo chữ, tạo nền trong suốt .

2.4.2. Visual Studio

Visual Studio là (*IDE – Integrated Development Environment*) một bộ công cụ phát triển phần mềm do Microsoft phát triển. Visual Studio cũng là một tool được sử dụng bởi các lập trình viên để xây dựng nên các sản phẩm phần mềm.

Hình 7: Visual studio

2.5. Khảo sát hiện trạng

2.5.1. Thực tiễn

Trong vài năm trở lại đây, endless runner trở thành một trong những thể loại game phổ biến và được yêu thích nhất trên di động. Chạy, chạy và chạy là tất cả những gì mà người chơi sẽ phải thực hiện trong các tựa game này.

Bằng khả năng sáng tạo không ngừng nghỉ, nhiều nhà phát triển ứng dụng cho mobile luôn tìm cách khai thác triệt để thể loại endless runner để cho ra đời hàng trăm tựa game chạy không ngừng nghỉ này.

Vì thế chúng em đã tạo nên 1 trò chơi mang tên Just Run dựa theo thể loại endless runner rất phổ biến hiện nay.

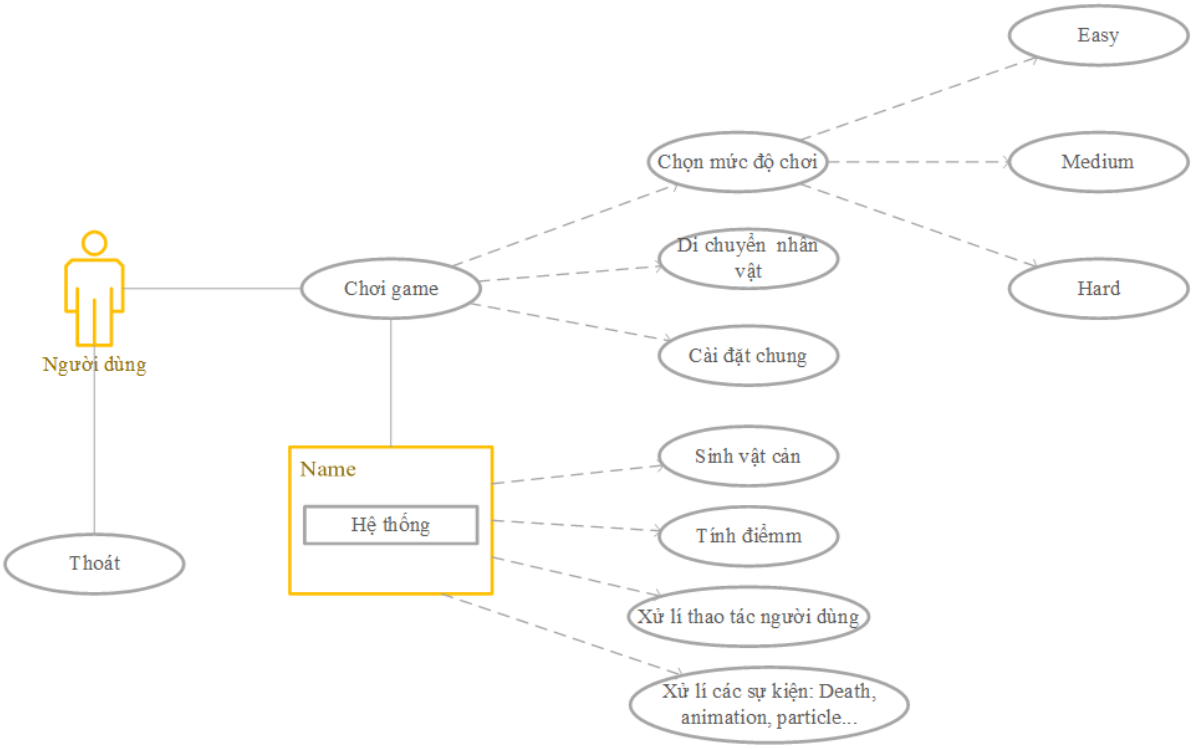
2.5.2. Lợi ích của trò chơi

Game hứa hẹn sẽ mang đến trải nghiệm mới lạ, giúp người chơi thư giãn, giải trí sau những giờ học tập lao động căng thẳng. Game còn là công cụ rèn luyện kỹ năng phản xạ, tính tập trung, khả năng tính toán suy nghĩ để có thể hoàn thành thử thách khó nhất của trò chơi.

2.6. Sơ đồ phân tích chức năng hệ thống

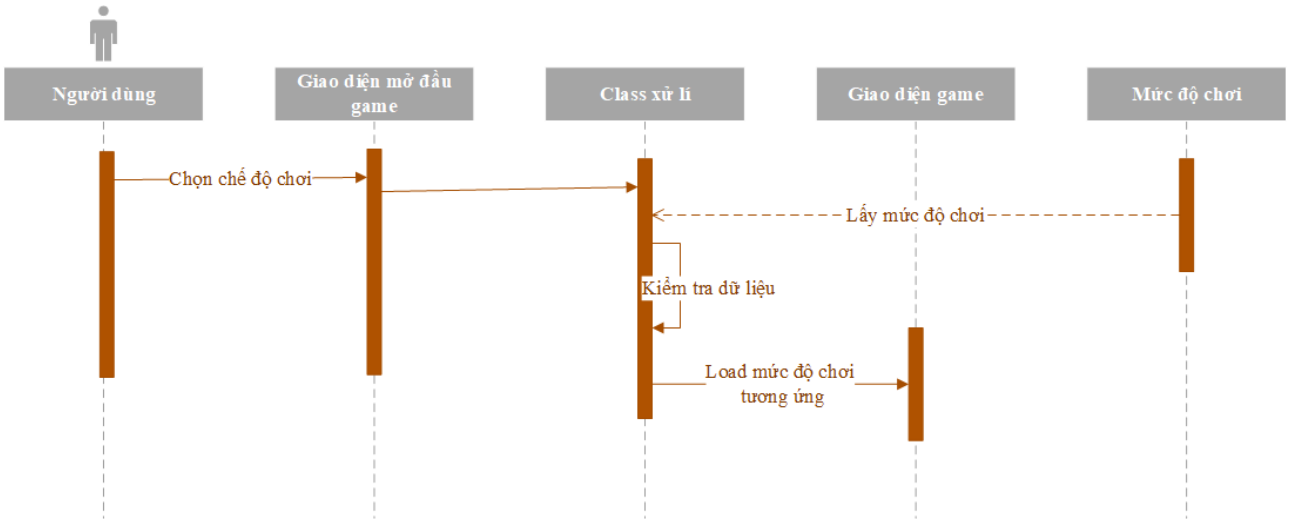
Chương trình được xây dựng trên ngôn ngữ lập trình C# dùng để thiết lập chương trình và xử lý các chức năng. Với sự trợ giúp của các phần mềm Microsoft Visual Studio 2019 để xây dựng chương trình, Unity3D hệ thống gồm các sơ đồ chức năng sau:

2.6.1. Sơ đồ Use case



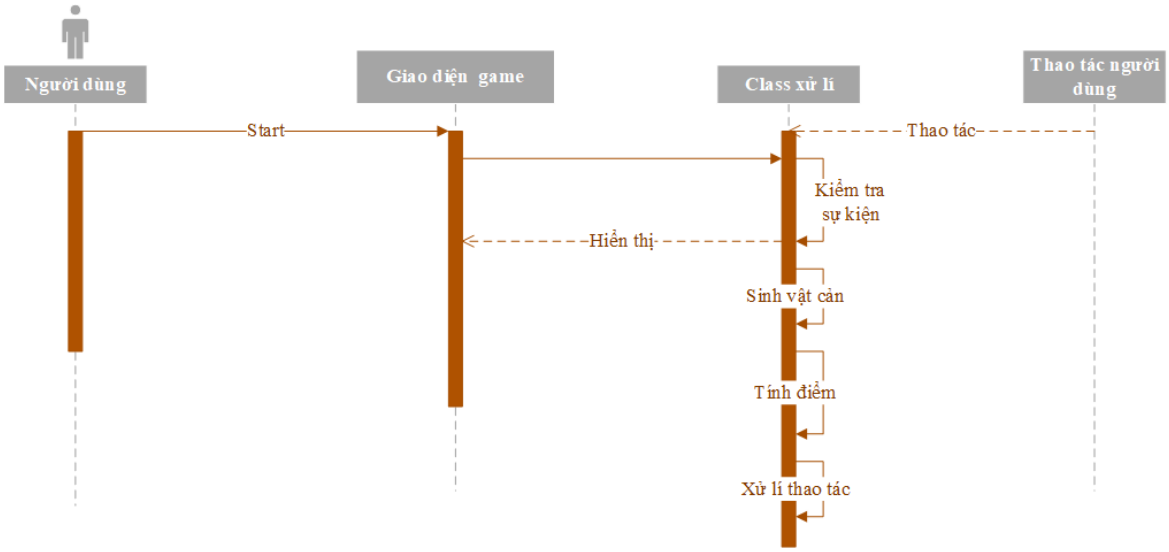
Hình 8: Sơ đồ use case

2.6.2 Sơ đồ Sequence chọn mức độ chơi



Hình 9: Sơ đồ Sequence chọn mức độ chơi

2.6.2. Sơ đồ Sequence chơi game



Hình 10: Sơ đồ Sequence chơi game

2.6.3. Sơ đồ Sequence chỉnh âm lượng

Hình 11: Sơ đồ Sequence chỉnh âm lượng

2.7. Kế hoạch quản lí dự án

2.7.1. Bảng phân rã công việc theo kế hoạch

STT	Công việc
1	Lập kế hoạch dự án
	Tài liệu kế hoạch quản lý dự án
	Bản kế hoạch đảm bảo chất lượng
	Bản kế hoạch quản lý rủi ro
2	Xác định yêu cầu
	Tài liệu yêu cầu người dùng
	Tài liệu yêu cầu chung cho hệ thống
	Tài liệu yêu cầu cho mỗi chức năng của hệ thống
	Tài liệu yêu cầu hệ thống
	Biểu đồ use case cho hệ thống
	Biểu đồ Sequence cho hệ thống
	Mô tả giao diện hệ thống
	Các tài liệu khác
3	Phân tích thiết kế
4	Phân công công việc cụ thể
	Player PC move
	Obstacle move
	Art: Background, Enviroment
	Art: Player
	Spawn random obstacle (Basic)
	Player animation
	Spawn random obstacle (Pool Object)
	Art: Animal

		Sound effect (SFX)
		Art: Obstacle
		Audio manager
		Animation manager
		User interface
		Background music
		Animals animation
		Particle: Sand, Smog
		Particle: Sparks
		Merge all code
		Refactor code
		Fix bug
5	Kiểm thử	
6	Vận hành	
7	Kết thúc dự án	

2.7.2. Kế hoạch quản lí thời gian

STT	Công việc		Thời gian	Sinh Viên
1	Lập kế hoạch dự án		01/05	Thịnh
2	Xác định yêu cầu		02/05	Thịnh
3	Phân tích thiết kế		03/05- 04/05	Thịnh, Bắc, Đức, Đạt
4	Phân công công việc cụ thể		05-10/06	
		Player PC move	05/05	Bắc
		Obstacle move	05/05	Bắc
		Art: Background, Enviroment	05/05	Thịnh
		Art: Player	05/05	Đức

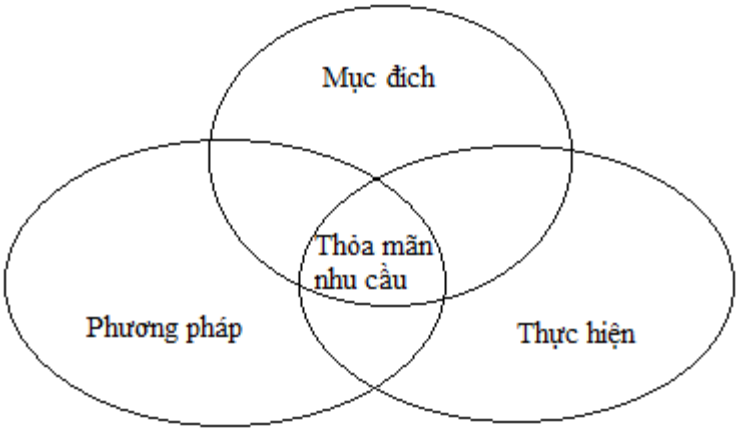
		Spawn random obstacle (Basic)	10/05	Bắc
		Player animation	10/05	Đức
		Spawn random obstacle (Pool Object)	15/05	Thịnh
		Art: Animal	15/05	Đức
		Sound effect (SFX)	15/05	Đạt
		Art: Obstacle	15/05	Đạt
		Audio manager	20/05	Đạt
		Animation manager	20/05	Đức
		User interface	20/05	Thịnh
		Background music	20/05	Đạt
		Animals animation	25/05	Đức
		Particle: Sand, Smog	25/05	Đức
		Particle: Sparks	25/05	Thịnh
		Merge all code	30/05	Thịnh
		Refactor code	10/06	Thịnh
		Fix bug	10/06	Thịnh,Bắc
5	Kiểm thử		15/06	Thịnh, Bắc, Đạt, Đức
6	Vận hành		18/06	Thịnh, Bắc, Đạt, Đức
7	Kết thúc dự án		20/06	

2.7.3. Kế hoạch quản lý chi phí

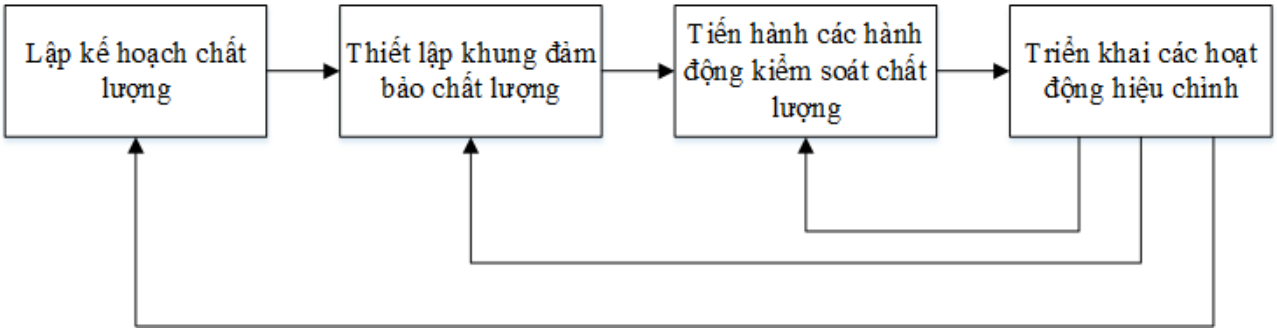
- Lập kế hoạch quản lý chi phí.
- Ước lượng chi phí.
- Xác định ngân sách dự án.
- Kiểm soát chi phí.

2.7.4. Kế hoạch quản lý chất lượng

- Thích hợp với mục đích
- Giảm tối đa sự lãng phí bằng cách thực hiện đúng ngay từ lần đầu.
- Cân bằng chất lượng.



Hình 12: Cân bằng chất lượng.



Hình 13: Quy trình quản lý chất lượng.

2.7.5. Kế hoạch quản lý rủi ro

Khi lập kế hoạch quản lý

- Khi dự án sẵn sàng thực thi

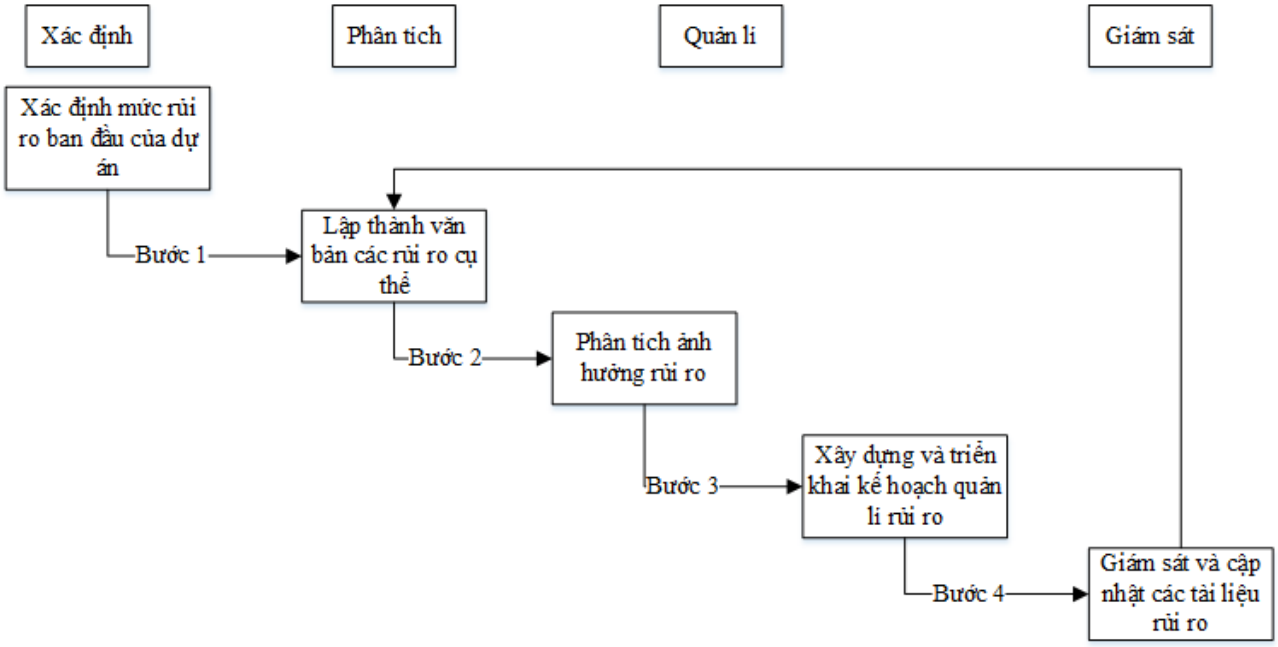
- Khi khôi phục một dự án đã bỏ dở

- Khi rà xét dự án

- Khi có sự sai lệch lớn so với kế hoạch xảy ra à

Giảm thiểu ảnh hưởng của các sự cố không biết trước cho dự án. Nâng cao xác suất thực hiện thành công dự án. Tạo ra ý thức kiểm soát. Có được các giải pháp hiệu quả và kịp thời.

Giảm tối thiểu ảnh hưởng của những sự cố không biết trước cho dự án bằng cách xác định và đưa ra những giải pháp tình huống trước khi có những hậu quả xấu xảy ra.



Hình 14: Kế hoạch quản lý rủi ro.

CHƯƠNG 3: MÔ TẢ THIẾT KẾ GAME

- Phần mềm: Unity

- Nền tảng: Windows

- Ngôn ngữ lập trình: C#

- Công cụ hỗ trợ: Visual Studio Professional 2019, Adobe Photoshop 2020, Audacity.

3.1. Ý tưởng

Dựa theo những game mang thể loại ‘Endless Runner’ như: Temple Run, Despicable Me, … Chúng em đã tạo ra một trò chơi mới hướng theo thể loại ‘Endless Runner’ này, có tên là Game Just Run.Thứ nhất, bối cảnh trong Game Just Run khác biệt ở chỗ góc nhìn của trò chơi là từ trái sang phải (Temple Run là góc nhìn từ gần ra xa). Thứ hai là cách di chuyển của Game Just Run cũng khác biệt là di chuyển 3 làn lên và xuống. Thứ ba là cách chơi game đơn giản, chỉ cần thao tác với 2 nút trên màn hình (đối với điện thoại), 2 phím lên xuống (đối với máy tính). Thứ tư nói về mức độ chơi cũng có các mức độ khác nhau từ dễ, trung bình đến khó.

3.2. Tạo bản đồ và nhân vật

3.2.1. Bản đồ (Map)

Bản đồ có sự kết hợp núi rừng, cây cối … tạo cho người chơi cảm giác như đang lạc vào 1 khu rừng thật sự

Hình 15: Môi trường

3.2.2. Nhân vật

- Player:

+ Di chuyển bằng các phím để tránh các vật cản và trốn khỏi các con thú hung dữ.

+ Chạy đến vùng có khiên để nhận được bảo vệ.

Hình 16: Hình ảnh nhân vật

- Animal

+ Bao gồm các loài vật trong tự nhiên như: Hổ, Chim, Bò Cạp

Hình 17: Hổ

Hình 18: Chim

Hình 19: Bò cạp

+ Các con vật này sẽ chạy theo sau người chơi đuổi theo người chơi đến khi người chơi va phải vật cản.

3.3. Xử lý di chuyển, điểm số, kết thúc màn chơi

- Di chuyển: Người chơi sử dụng phím ↑ và ↓ hoặc (W và S) để di chuyển người chơi né các vật cản

- Điểm Số: Khi người chơi bắt đầu trò chơi điểm số sẽ tăng dần theo các các cập độ cấp độ càng nhanh thì điểm càng nhiều đồng thời sẽ khó hơn các cấp độ trước đó. Điểm sẽ được hiển thị ở giữa bên trên của màn hình game.

- Vật phẩm hỗ trợ: Khi người chơi di chuyển đến 1 mức nào đó sẽ hiện thị 1 vật bảo vệ khi người chơi nhặt được vật phẩm đó player sẽ được bảo vệ.

- Khi người chơi va phải vật cản trò chơi sẽ kết thúc (Game Over).

3.4. Tiến trình của Game

- Người chơi sẽ phải khéo léo vượt qua toàn bộ các vật cản cùng với đó là các con vật nguy hiểm đang đuổi theo sau.

- Khi người chơi vượt qua càng nhiều vật cản các mốc trò chơi sẽ tăng lên khiến cho người chơi gặp khó khăn hơn khi vượt qua các vật cản.

- Trò chơi sẽ kết thúc khi người chơi va phải vào vật cản.

Từ giao diện chính người chơi có những lựa chọn sau:

- New Game – Bắt đầu trò chơi .
- Options – Điều chỉnh thông số của các chức năng.
- Quit – Thoát game.

3.5. Hoạt họa và cảm nhận

- Trò chơi mang đậm tính giải trí giúp cho người chơi giải tỏa áp lược sau những giờ học căng thẳng nhân vật 3D đầy ngộ nghĩnh phù hợp với lứa tuổi học sinh.

- Hình ảnh bị thú vật rượt đuổi giúp người chơi dễ bị lôi cuốn hòa mình vào game hơn .

- Hiệu ứng vật lý, hiệu ứng ánh sáng kết hợp nhạc nền dồn dập tạo cảm giác thôi thúc, kịch tính, thú vị.

3.6 Một số giao diện trong game

3.6.1 Giao diện bắt đầu game

Hình 20: Giao diện bắt đầu game

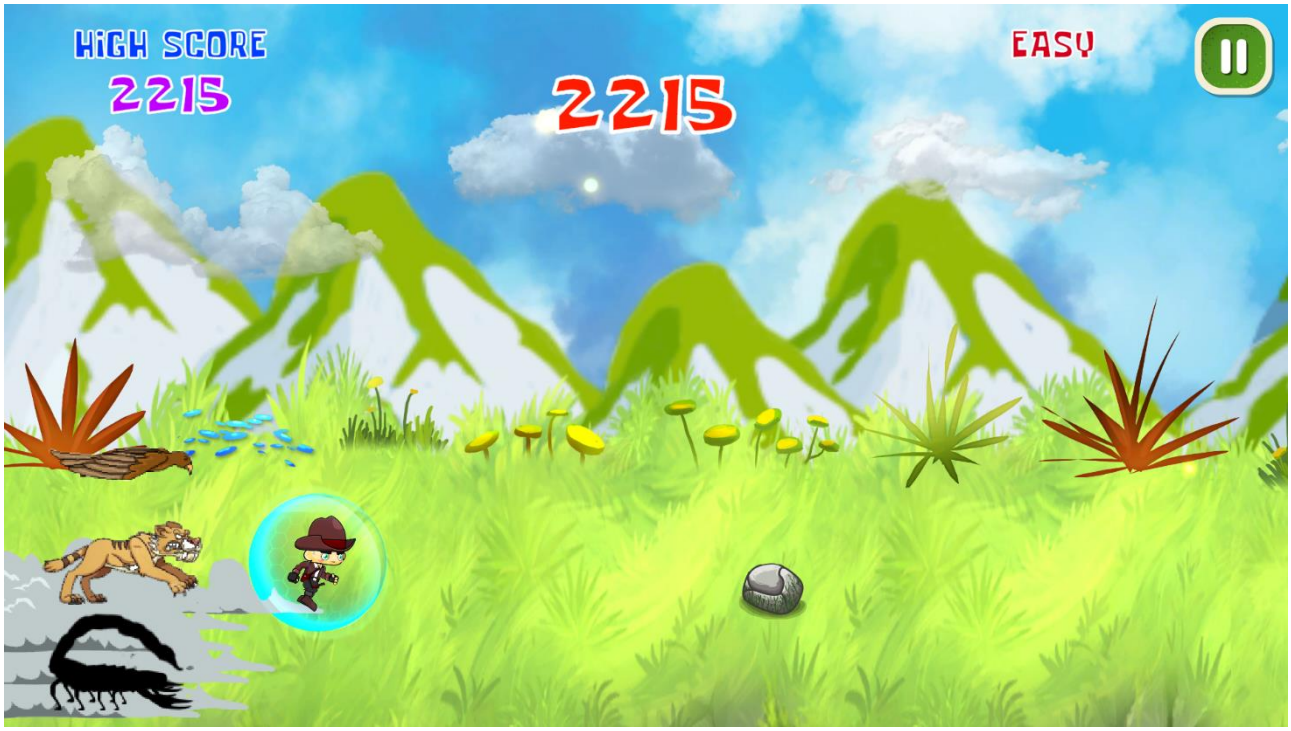
3.6.2 Giao diện chọn mức độ chơi

Hình 21: Giao diện chọn mức độ chơi

3.6.3 Giao diện thiết lập

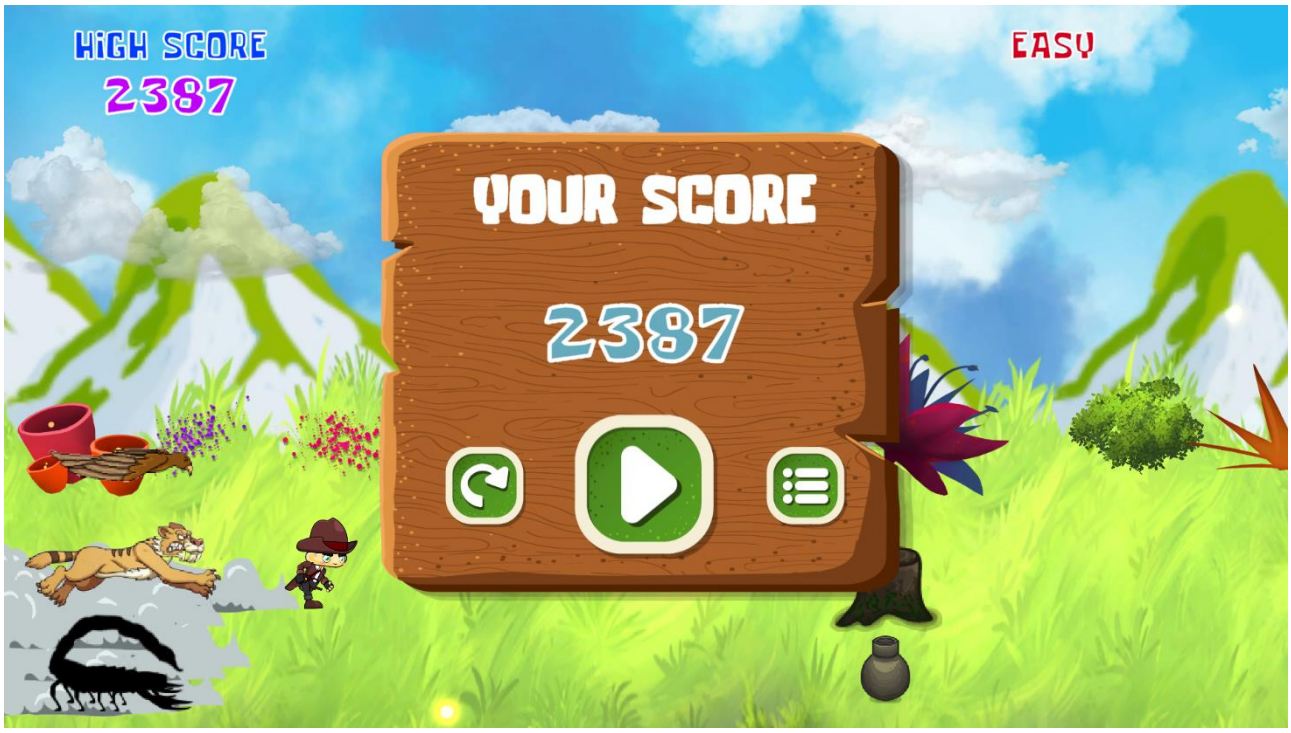
Hình 22: Giao thiết lập

3.6.4 Giao diện chính



Hình 23: Giao diện chính

3.6.5 Giao diện khi tạm dừng game



Hình 24: Giao diện tạm dừng game

3.6.6 Giao diện khi kết thúc game

Hình 25: Giao diện kết thúc game

3.7. Một số hàm quan trọng trong game

3.7.1. Player Move

```
void PCInput()
    {
        if(Input.GetButtonDown("UpArrow"))
        {
            transform.position = new Vector2(transform.position.x,
Mathf.Min(transform.position.y + Distance,ySpawnPos + Distance));
        }
        if (Input.GetButtonDown("DownArrow"))
        {
            transform.position = new Vector2(transform.position.x,
Mathf.Max(transform.position.y - Distance, ySpawnPos - Distance));
        }

    }



public void MobileInput(string direction)
    {
        if (!dead)
        {
            if (direction.Equals("UpArrow"))
            {
                transform.position = new Vector2(transform.position.x,
Mathf.Min(transform.position.y + Distance, ySpawnPos + Distance));
            }
            if (direction.Equals("DownArrow"))
            {
                transform.position = new Vector2(transform.position.x,
Mathf.Max(transform.position.y - Distance, ySpawnPos - Distance));
            }
        }

    }
```

3.7.2. Nhận biết va chạm

```
void OnTriggerEnter2D(Collider2D col)
    {
        if (col.gameObject.CompareTag("Shield"))
        {
            AudioManager.Instance.PlaySound("shield");
            if (!hasShield)
            {
                hasShield = true;
                shield.SetActive(true);
            }
            else
            {
                GameManager.Instance.Plus1000(transform.position);
            }
            SpawnManager.Instance.ReturnToPool("shield", col.gameObject);
        }
        else if (col.gameObject.CompareTag("Obstacle"))
        {
            if (!hasShield)
            {
                dead = true;
```

```
                AudioManager.Instance.PlaySound("ugh");
                GameManager.Instance.StopGame();
            }
            else
            {
                AudioManager.Instance.PlaySound("collision");
                hasShield = false;
                shield.SetActive(false);
            }
        }

    }
```

3.7.3. Sinh vật cản ngẫu nhiên

```
public void SpawnObstacleToScene()
    {
        StartCoroutine(_SpawnObstacleToScene());
    }

    IEnumerator _SpawnObstacleToScene()
    {
        while (!GameManager.Instance.GameOver)
        {
            yield return new WaitForSeconds(Random.Range(spawnObstaclesDelayMin,
spawnObstaclesDelayMax) / GameManager.Instance.Speed);
            if (poolObjects[strObstacles].Count > 1)
            {
                ySpawned = new bool[3];
                for (int i=0; i<Random.Range(1, 3); i++) // Spawn 1 or 2 obstacle
                {
                    GameObject ob = poolObjects[strObstacles][0];

                    // Spawn random position
                    int index;
                    do
                    {
                        index = Random.Range(0, 3);
                    } while (ySpawned[index] == true);
                    ySpawned[index] = true;

                    ob.transform.position = new Vector2(ScreenBounds.Right,
ySpawnPos[index]);
                    ob.SetActive(true);
                    poolObjects[strObstacles].Remove(ob);
                }
            }
        }

    }
```

3.7.4. Âm thanh

```
public class Sound
{
    public string name;
    public AudioClip clip;
    public bool playOnAwake = false;
    public bool loop = false;
    [Range(0f, 1f)] public float volume = 1f;
    [Range(0f, 3f)] public float pitch = 1f;
    [HideInInspector] public AudioSource source;
    public void SetAudioSource(AudioSource src)
```

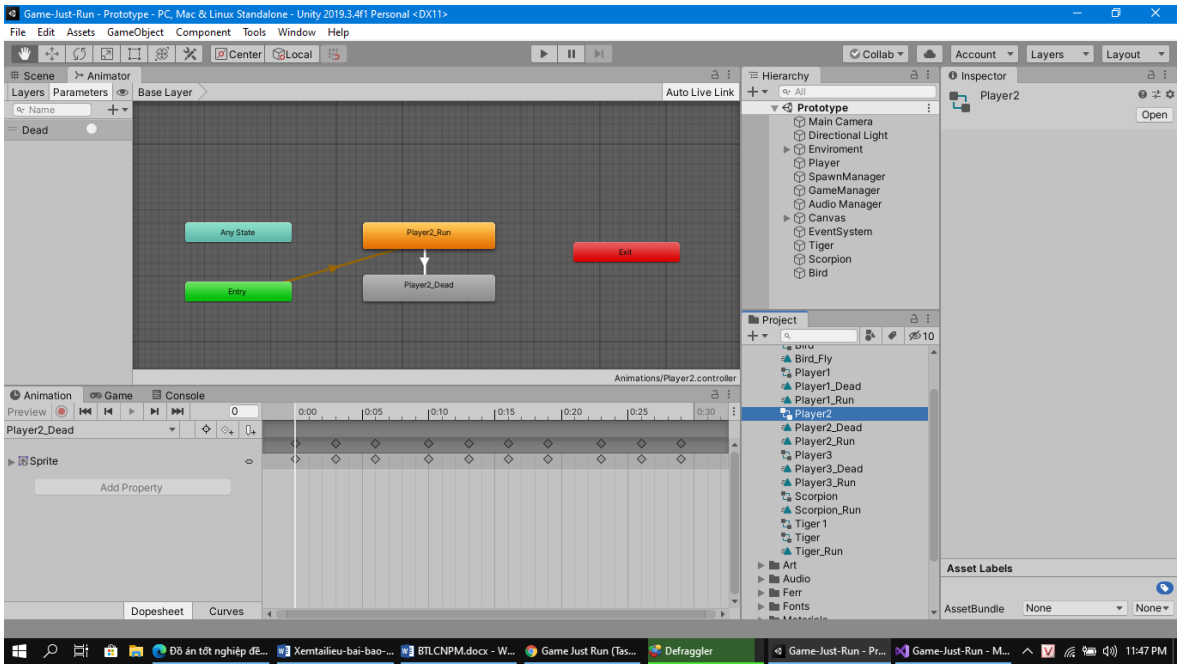
```
    {
        source = src;
        source.clip = clip;
        source.playOnAwake = playOnAwake;
        source.loop = loop;
        source.volume = volume;
        source.pitch = pitch;
    }
    public void Play()
    {
        source.Play();
    }
 }



public void PlaySound(string name)
    {
        if(sound.ContainsKey(name))
        {
            sound[name].Play();
        }

    }
```

3.7.5. Player Animation

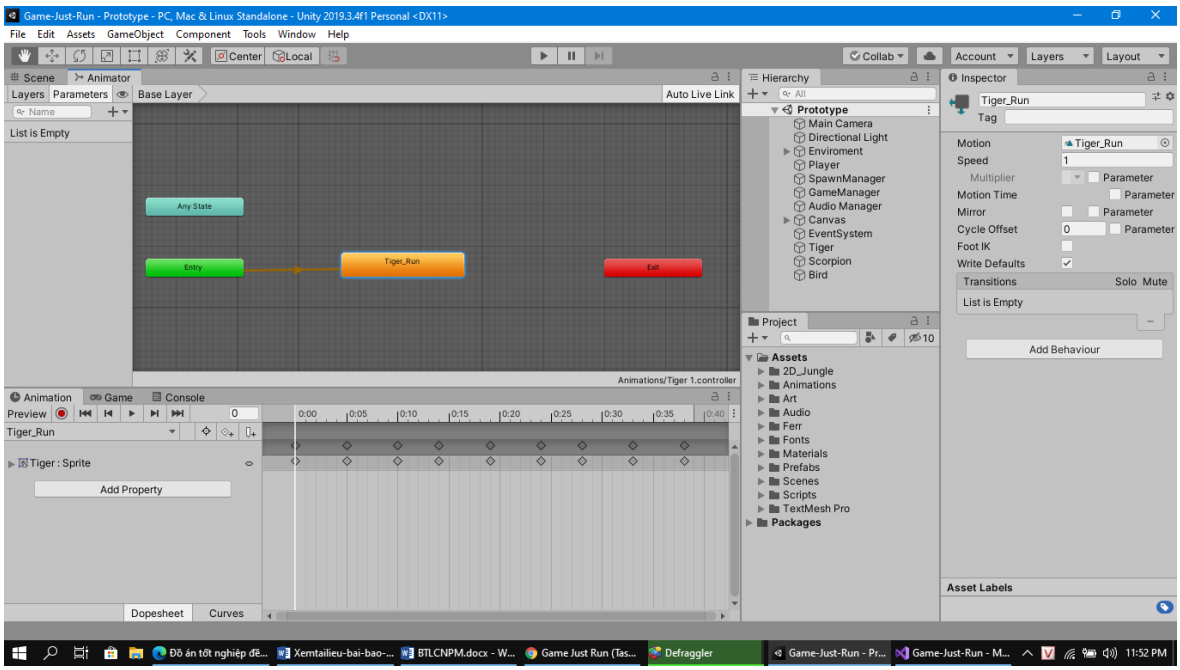


Hình 26: Hình ảnh Player Animation.

```
void Update()
    {
        if(player.CheckDeath())
        {
            playerAnimator.SetTrigger("Dead");
        }
        else
        {
            playerAnimator.speed = GameManager.Instance.Speed / 10f;
```

```
        }
    }
```

3.7.6. Animal Animation



Hình 27: Hình ảnh Animal Animation

```
void Update()
    {
        if (!GameManager.Instance.GameOver)
        {
            for (int i = 0; i < animators.Length; i++)
            {
                animators[i].speed = GameManager.Instance.Speed / 10f;
            }
        }
        else
        {
            for (int i = 0; i < animators.Length; i++)
            {
                animators[i].speed = 0;
            }
        }
    }
```

3.7.7. Enviroment Move

```
void Update()
    {
        if (transform.position.x >= leftRange)
        {
            transform.position = (Vector2)transform.position + Vector2.left * Time.deltaTime
* GameManager.Instance.Speed / SpeedInverseRatio;
        }
        else
        {
            transform.position = originalPosistion;
        }
```

```
    }
```

3.7.8. Game Manager

```
public void PlayGame(float speed)
    {
        string level = EventSystem.current.currentSelectedGameObject.name;
        PlayerPrefs.SetString("LEVEL", level);
        PlayerPrefs.SetFloat("Speed", speed);
        SceneManager.LoadScene(1);
    }

public void PauseGame()
    {
        paused = true;
        Time.timeScale = 0;
    }

public void ResumeGame()
    {
        paused = false;
        Time.timeScale = 1;
    }

public void StopGame()
    {
        Speed = 0;
        GameOver = true;
        gameOverScreen.SetActive(true);
        PlayerPrefs.SetInt(level + "HighScore", Mathf.Max(score, highScore));
    }

public void Restart()
    {
        Time.timeScale = 1;
        SceneManager.LoadScene(1);
    }

public void ReturnToMenuGame()
    {
        Time.timeScale = 1;
        SceneManager.LoadScene(0);
    }
```

CHƯƠNG 4: KẾT QUẢ VÀ ĐỊNH HƯỚNG PHÁT TRIỂN

4.1. Kết quả trò chơi

4.1.1. Gameplay

- Khi nhấn vào nút bắt đầu người chơi sẽ đứng ở vị trí xuất phát và bắt đầu chạy.

- Mục tiêu chính: Sống sót, né các vật cản và đạt điểm cao nhất

- Người chơi sử dụng phím ↑ và ↓ hoặc (W và S) để di chuyển vượt qua các vật cản để đến các mốc với lever cao hơn

- Khi chạy đến 1 vị trí nào đó khiên bảo vệ sẽ xuất hiện người chơi nhặt khiên lên để bảo vệ bản thân trước những vật cản.

- Nếu va chạm phải vật cản bạn sẽ chết và phải bắt đầu lại game.

4.1.2. Các chức năng cơ bản

Các tính năng chính được đưa ra và test trong demo bao gồm:

+ Chức năng cơ bản của player (chạy, chuyển làn).

+ Chức năng của animal (rượt theo người chơi).

+ Chức năng hiển thị của map (ánh sáng, vật dụng và quang cảnh).

+ Chức năng chơi nhạc.

+ Chức năng tính điểm.

+ Một số các cơ chế phụ trợ khác: khiên bảo vệ, tăng tốc độ di chuyển theo thời gian.

4.2. Định hướng phát triển

- Đa dạng nhân vật, động vật cùng với các âm thanh sống động.

- Thêm môi trường mới

- Thiết kế thêm thể loại, cách thức chơi mới.

- Các chức năng điều chỉnh options (âm thanh, ánh sáng).

4.3. Kết luận

Qua trò chơi demo chúng em đã từng bước tiếp cận được với công nghệ làm game, cách thức cũng như phương pháp, nền tảng tối ưu, biết cách kết hợp các phần mềm để tạo ra một

mô hình vật thể, sản phẩm. Trong quá trình xây dựng game chúng em nhận ra còn nhiều vấn đề phát sinh, nhiều khó khăn hơn mình suy nghĩ, rất nhiều bug trong code C# cần fix, đôi khi 1 lỗi nhỏ làm hỏng cả hệ thống, vì vậy khi thiết kế game chúng ta nên chia nhỏ từng công việc và module riêng ra để dễ xây dựng và gắn kết khi các thành phần đã chạy ổn định. Viết game rất tốn thời gian cho các công việc như design các nhân vật, bản đồ, đội ngũ xây dựng phải đông đảo hơn, cần sự kiên trì và sáng tạo hơn.