TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN MÔN**

**PHÁT TRIỂN TRÒ CHƠI**

**FOX 2D**

*Người hướng dẫn*: **GV Hà Lê Hoài Trung**

*Người thực hiện*: **LÊ CÔNG NGHĨA - 52000691**

**ĐẶNG TRUNG TÍN - 51800820**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2021**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÀI TẬP LỚN MÔN**

**PHÁT TRIỂN TRÒ CHƠI**

**FOX 2D**

*Người hướng dẫn*: **GV Hà Lê Hoài Trung**

*Người thực hiện*: **LÊ CÔNG NGHĨA - 52000691**

**ĐĂNG TRUNG TÍN - 51800820**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2021**

# LỜI CẢM ƠN

Nhân gian ta có câu tục ngữ sau “người không học như ngọc không mài”. Qua câu nói ấy cho chúng ta thấy được tầm quan trọng của việc học tập và liên tục trào dồi kiến thức có ý nghĩa lớn như thế nào trong việc hoàn thiện bản thân. Nhưng để có thể sản sinh ra những người học trò tài giỏi thì cũng cần phải có những người thầy, người cô vĩ đại. Và thực tế cho ta thấy rằng thầy cô luôn là những người hy sinh thầm lặng, cống hiến cả đời để các thế hệ học sinh, sinh viên chúng em vững bước trên con đường đời sau này. Biết được công lao to lớn ấy, em xin chân thành gửi lời cảm ơn đến thầy Vũ Đình Hồng và thầy Hà Lê Hoài Trung đã giúp chúng có thêm nhiều kiến thức về môn học “Phát triển trò chơi” bằng unity. Các thầy đã cugns cấp cho các em nhiều kiến thức bổ ích, giúp chugns em hiểu rõ hơn về cách thức hoạt động của Unity và cách tạo ra một game bằng Unity. Đây là bàn đạp, là cơ sở để chúng em hoàn thiện trò chơi này.

Tiếp theo, nhóm chúng em xin gửi lời cảm ơn đến khoa CNTT trường Đại học Tôn Đức Thắng vì đã tạo điều kiện cho chúng tôi được học tập và nghiên cứu môn học này. Khoa đã luôn sẵn sàng chia sẻ các kiến thức bổ ích cũng như chia sẻ các kinh nghiệm tham khảo tài liệu, giúp ích không chỉ cho việc thực hiện và hoàn thành đề tài báo cáo mà còn giúp ích cho việc học tập và rèn luyện trong quá trình thực hành tại trường Đại học Tôn Đức Thắng.

Mặc dù chúng em đã có nhiều cố gắng trong lúc làm báo báo, nhưng do kiến thức còn hạn chế nên bài báo cáo không tránh khỏi những thiếu sót. Chúng em mong rằng sẽ nhận được sự thông cảm và ý kiến đóng góp của các thầy cô để chúng em có thể hoàn thiện hơn trong những lần làm bài lần sau.

**ĐỒ ÁN ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là sản phẩm đồ án của riêng chúng tôi và được sự hướng dẫn của GV Hà Lê Hoài Trung;. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày tháng năm*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

*Lê Công Nghĩa*

*Đăng Trung Tín*

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

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

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

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

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

# TÓM TẮT

Bài báo cáo giữa kỳ của môn học Phát Triển Trò Chơi của trường đại học Tôn Đức Thắng. Bài báo cáo này trình bày về game FOX 2D mà nhóm em đã làm bằng phần mềm Unity. Chúng em đã trình bày một cách cụ thể các vấn đề khi xây dựng game nói chung và xây dựng game FOX 2D nói riêng. Bài báo cáo này giúp thầy cô có cái nhìn tổng quan nhất và hiểu được cách chơi cũng như cách xây dựng game FOX 2D của chúng em. Bài báo cáo được chia làm 6 chương như sau:

Chương 1 – MỞ ĐẦU: Trong chương này chúng em nói khái quát về cách chơi game FOX 2D. Mục tiêu và nhiệm vụ của nhóm em cho bài báo cáo giữa kỳ này.

Chương 2 – CƠ SỞ LÝ THUYẾT: Trong chương này chúng em trình bày về phầ mềm Unity. Đây là phần mềm làm game được nhà trường giảng dạy để chúng em tạ ra game của riêng mình. Nội dung trình bày về một số đặc điểm của Unity và giới thiệu về ngôn ngữ lập trình C# hỗ trợ cho việc phát triển game.

Chương 3 - MỘT SỐ VẤN ĐỀ VÀ GIẢI PHÁP KHI XÂY DỰNG GAME ENGINE TRÊN UNITY: Trong chương này chúng em trình bày về một số giải pháp khi xấy dựng game bằng Unity như: chuyển động nhân vật 2D, them sự kiện vào chuyển động của nhân vật, xây dựng giao diện game và tạo âm thanh cho game.

Chương 4 - ỨNG DỤNG GAME PHÁT TRIỂN TRÊN UNITY: Trong chương này nhóm chúng em trình bày về các dữ liệu trong game, các mô hình, âm thành, hiệu ứng và các class viết bằng C# trong game.

Chương 5 – KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN: Trong chương này chúng em trình bày về những kết quả đạt được sau khi thực hiện game FOX 2D. Đồng thời là những vấn đề chúng em mắc phải trong quá trình xây dựng game, hướng phát triển game cho những lần sau.

**MỤC LỤC**

[LỜI CẢM ƠN i](#_Toc88337221)

[TÓM TẮT iv](#_Toc88337222)

[CHƯƠNG 1 – MỞ ĐẦU 9](#_Toc88337223)

[1.1 Giới thiệu về game 9](#_Toc88337224)

[1.2 Mục tiêu và nhiệm vụ đề tài 11](#_Toc88337225)

[CHƯƠNG 2 – CƠ SỞ LÝ THUYẾT 12](#_Toc88337226)

[2.1 Tổng quan về Unity 12](#_Toc88337227)

[2.2 Các đặc điểm và tính năng của Unity 14](#_Toc88337228)

[2.3 Các thành phần cơ bản trong Unity 16](#_Toc88337229)

[2.4 Các thành phần trong Unity Editor 22](#_Toc88337230)

[2.5 Giới thiệu ngôn ngữ lập trình C# (Csharp) 26](#_Toc88337231)

[CHƯƠNG 3 – MỘT SỐ VẤN ĐỀ VÀ GIẢI PHÁP KHI XÂY DỰNG GAME ENGINE TRÊN UNITY 27](#_Toc88337232)

[3.1 Chuyển động mô hình nhân vật 2D 27](#_Toc88337233)

[3.2 Thêm sự kiện vào chuyển động của nhân vật 29](#_Toc88337234)

[3.3 Xây dựng giao diện game 30](#_Toc88337235)

[3.4 Tạo âm thanh trong game 1](#_Toc88337236)

[CHƯƠNG 4 - ỨNG DỤNG GAME PHÁT TRIỂN TRÊN UNITY 5](#_Toc88337237)

[4.1 Dữ liệu game FOX 2D 5](#_Toc88337238)

[4.2 Class trong game 17](#_Toc88337239)

[CHƯƠNG 5 – KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 22](#_Toc88337240)

[6.1 Những kết quả đạt được 22](#_Toc88337241)

[6.2 Những hạn chế 22](#_Toc88337242)

[6.3 Hướng phát triển 22](#_Toc88337243)

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

[Hình 1: Nơi bắt đầu 10](#_Toc88334831)

[Hình 2: Vị trí đích 10](#_Toc88334832)

[Hình 3: Trong quá trình di chuyển 11](file:///D:\BIG%20BOSS\Nghia\Phat_Trien_Tro_Choi\Thuc_Hanh\Bao_Cao_Giua_Ky\Bao_Cao.docx#_Toc88334833)

[Hình 4: Chú sóc rơi xuống vực sâu 11](file:///D:\BIG%20BOSS\Nghia\Phat_Trien_Tro_Choi\Thuc_Hanh\Bao_Cao_Giua_Ky\Bao_Cao.docx#_Toc88334834)

[Hình 5: Kết thúc trò chơi 12](file:///D:\BIG%20BOSS\Nghia\Phat_Trien_Tro_Choi\Thuc_Hanh\Bao_Cao_Giua_Ky\Bao_Cao.docx#_Toc88334835)

[Hình 6: Hỗ trợ đa nền tảng trong Unity 14](#_Toc88334836)

[Hình 7: Asset trong Unity 17](file:///D:\BIG%20BOSS\Nghia\Phat_Trien_Tro_Choi\Thuc_Hanh\Bao_Cao_Giua_Ky\Bao_Cao.docx#_Toc88334837)

[Hình 8: Các Scene của Unity 18](file:///D:\BIG%20BOSS\Nghia\Phat_Trien_Tro_Choi\Thuc_Hanh\Bao_Cao_Giua_Ky\Bao_Cao.docx#_Toc88334838)

[Hình 9: Kéo tài nguyên vào Scene để sử dụng 19](file:///D:\BIG%20BOSS\Nghia\Phat_Trien_Tro_Choi\Thuc_Hanh\Bao_Cao_Giua_Ky\Bao_Cao.docx#_Toc88334839)

[Hình 10: Các thành phần trong đối tượng Camera 20](file:///D:\BIG%20BOSS\Nghia\Phat_Trien_Tro_Choi\Thuc_Hanh\Bao_Cao_Giua_Ky\Bao_Cao.docx#_Toc88334840)

[Hình 11: Cách tạo file script mới 21](file:///D:\BIG%20BOSS\Nghia\Phat_Trien_Tro_Choi\Thuc_Hanh\Bao_Cao_Giua_Ky\Bao_Cao.docx#_Toc88334841)

[Hình 12: Một file Script đang gắn vào đối tượng 21](file:///D:\BIG%20BOSS\Nghia\Phat_Trien_Tro_Choi\Thuc_Hanh\Bao_Cao_Giua_Ky\Bao_Cao.docx#_Toc88334842)

[Hình 13: Một số đối tượng trong Prefabs 22](#_Toc88334843)

[Hình 14: Cửa sổ Hierarchy. 24](#_Toc88334844)

[Hình 15: Các loại hình ảnh trong cửa sổ game. 24](#_Toc88334845)

[Hình 16: Cửa sổ Project. 25](#_Toc88334846)

[Hình 17: Cửa sổ Inspector. 26](#_Toc88334847)

[Hình 18: Ngôn ngữ lập trình C# (Csharp) 27](#_Toc88334848)

[Hình 19: Mô hình 2D bên trong chứa nhiều animation 28](#_Toc88334849)

[Hình 20: Mô hình 2D chứa một animation. 29](file:///D:\BIG%20BOSS\Nghia\Phat_Trien_Tro_Choi\Thuc_Hanh\Bao_Cao_Giua_Ky\Bao_Cao.docx#_Toc88334850)

[Hình 21: Hình minh họa sau khi đổi tên và import vào project 29](file:///D:\BIG%20BOSS\Nghia\Phat_Trien_Tro_Choi\Thuc_Hanh\Bao_Cao_Giua_Ky\Bao_Cao.docx#_Toc88334851)

[Hình 22: GUI Button 32](#_Toc88334852)

[Hình 23: Áp dụng GUIStyle lên Label 33](#_Toc88334853)

[Hình 24: Thông tin file âm thanh. 2](#_Toc88334854)

[Hình 25: Thêm thành phần Audio Source 3](#_Toc88334855)

[Hình 26: Thêm file âm thanh cho thành phần AudioSource 3](#_Toc88334856)

[Hình 27: Assets 5](#_Toc88334857)

[Hình 28: Explosionsoundslite 5](#_Toc88334858)

[Hình 29: Sounds 6](#_Toc88334859)

[Hình 30: Footstep(Snow and Grass) 6](#_Toc88334860)

[Hình 31: StepsLoop 6](#_Toc88334861)

[Hình 32: StepsLoop > Grass 7](#_Toc88334862)

[Hình 33: Grass > Running 7](#_Toc88334863)

[Hình 34: Grass > Slowsteps 7](#_Toc88334864)

[Hình 35: Grass > Walking 7](#_Toc88334865)

[Hình 36: StepsLoop > Snow 8](#_Toc88334866)

[Hình 37: StepsLoop > Snow > Running 8](#_Toc88334867)

[Hình 38: StepsLoop > Snow > Slow steps 8](#_Toc88334868)

[Hình 39: StepsLoop > Snow > Walking 8](#_Toc88334869)

[Hình 40: Stepssingle sound 9](#_Toc88334870)

[Hình 41: Stepssingle sound > Grass 9](#_Toc88334871)

[Hình 42: Stepssingle sound > Snow 9](#_Toc88334872)

[Hình 43: Prefabs 10](#_Toc88334873)

[Hình 44: Scenes 10](#_Toc88334874)

[Hình 45: Script 10](#_Toc88334875)

[Hình 46: Sunnyland 11](#_Toc88334876)

[Hình 47: Artwork 11](#_Toc88334877)

[Hình 48: Environment 11](#_Toc88334878)

[Hình 49: Props 11](#_Toc88334879)

[Hình 50: Tiles 12](#_Toc88334880)

[Hình 51: Tiles > tiles 12](#_Toc88334881)

[Hình 52: Sprites 12](#_Toc88334882)

[Hình 53: Enemies 12](#_Toc88334883)

[Hình 54: Eagle 13](#_Toc88334884)

[Hình 55: frog 13](#_Toc88334885)

[Hình 56: Frog > Idle 13](#_Toc88334886)

[Hình 57: Frog > Jump 13](#_Toc88334887)

[Hình 58: Opossum 13](#_Toc88334888)

[Hình 59: Fx 14](#_Toc88334889)

[Hình 60: Enemy-death 14](#_Toc88334890)

[Hình 61: Item-feedback 14](#_Toc88334891)

[Hình 62: Items 14](#_Toc88334892)

[Hình 63: Cherry 14](#_Toc88334893)

[Hình 64: Gem 15](#_Toc88334894)

[Hình 65: Player 15](#_Toc88334895)

[Hình 66: Climb 15](#_Toc88334896)

[Hình 67: Crouch 15](#_Toc88334897)

[Hình 68: Hurt 16](#_Toc88334898)

[Hình 69: Player > Idle 16](#_Toc88334899)

[Hình 70: Player > Jump 16](#_Toc88334900)

[Hình 71: Player > Run 16](#_Toc88334901)

[Hình 72: Scenes 17](#_Toc88334902)

# CHƯƠNG 1 – MỞ ĐẦU

## Giới thiệu về game

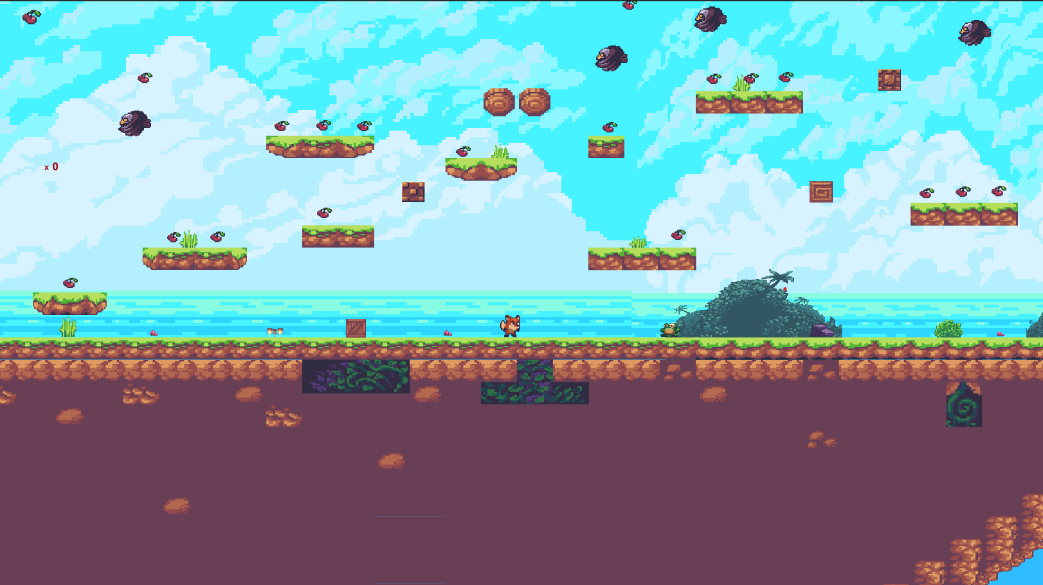
Kịch bản game của nhóm em thực hiện là game 2D có tên là FOX 2D. Game chỉ có một người chơi. Nhân vật chơi là một chú sóc. Khi vào game chú sóc được bắt đầu tại một ngôi nhà,

Hình 1: Nơi bắt đầu

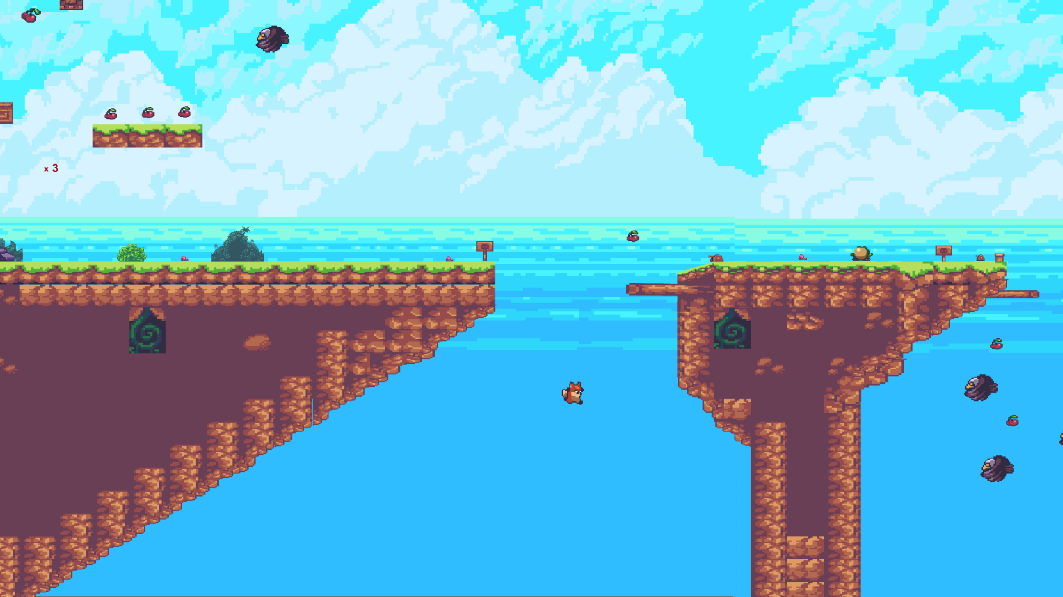
sau đó người chơi di chuyển chú sóc qua trái bằng nút A hoặc mũi tên sang trái, qua phải bằng nút D hoặc mũi tên sang phải và nhảy lên bằng nút Space để di chuyển về đích (đến cách cửa).



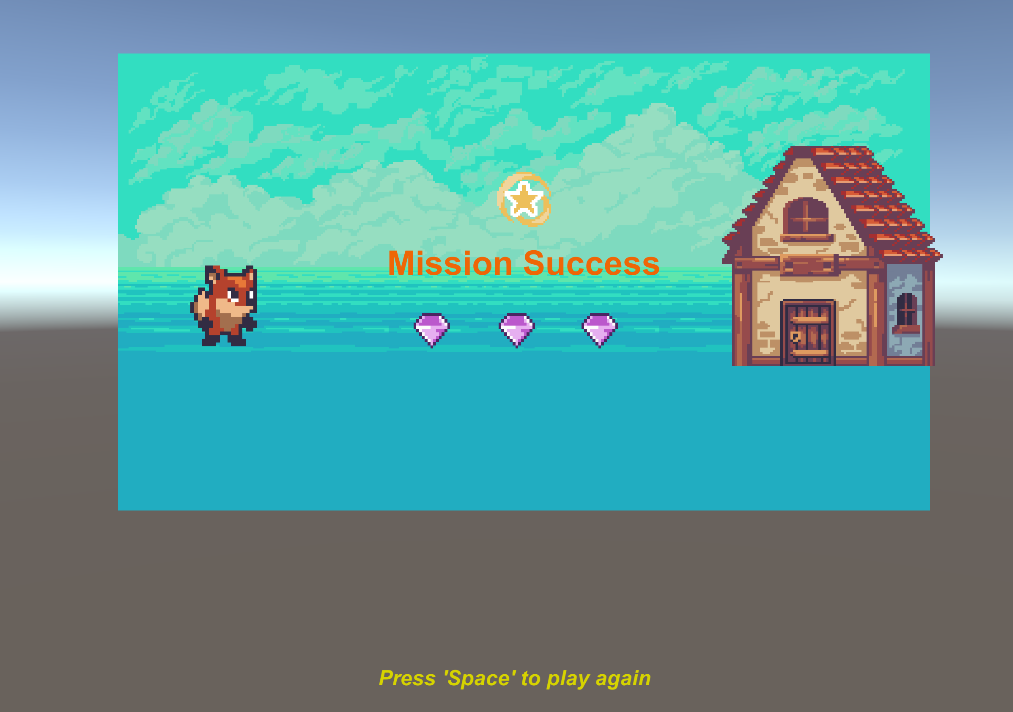
Hình 2: Vị trí đích

Trong quá trình di chuyển thì nhiêm vụ của người chơi là điều khiển chú sóc ăn những quả dâu tây để tăng thêm số điểm, mỗi khi chú sóc ăn được một quả dâu tây thì điểm số sẽ tăng lên một. Ngoài ra trong quá trình di chuyển sẽ có xuất hiện những con vật (cóc và chim). Những con vật này là chướng ngại vật mà người chơi cần tiêu diệt hoặc tránh né để nhanh chóng về đích.

Hình 3: Trong quá trình di chuyển

Để tiêu diệt được những con vật thì người chơi cần điều khiển chú sóc nhảy lên trên con vật, lúc này con vật sẽ bị tiêu diệt và biến mất. Chú ý rằng, những con vật này không làm cho chú sóc của người chơi bị chết, chúng chỉ cản trở đường đi của nhân vật chơi. Chú sóc chỉ chết khi bị rơi xuống vực sâu và lúc này trò chơi sẽ bắt đầu lại từ đầu (chú sóc sẽ đứng ở vị trí ngôi nhà).

Hình 4: Chú sóc rơi xuống vực sâu

Kết thúc trò chơi khi người chơi di chuyển chú sóc về vị trí đích (cánh cửa) và ăn hết những quả dâu tây trong suốt qua trình di chuyển. Khi về đến vị trí cuối cùng, người chơi có thể bắt đầu lại trò chơi bằng việc nhấn nút Space.

Hình 5: Kết thúc trò chơi

## Mục tiêu và nhiệm vụ đề tài

* Đề tài thuộc hướng tìm hiểu công nghệ từ đó xây dựng ứng dụng. Mục tiêu của đề tài là tìm hiểu Unity và sử dụng Unity xây dựng game 2D.
* Để thực hiện được điều này nội dung của bài báo cáo bao gồm:
* Tìm hiểu tổng quan về kiến thức của Unity và cách tạo lập các ứng dụng trong Unity.
* Tìm hiểu các vấn đề như load mô hình game, làm nhân vật chuyển động, cách tạo địa hình, giao diện, âm thanh và các hiệu ứng,… để từ đó đưa ra các giải pháp.
* Xây dựng và phát triển ứng dụng game bằng Unity.

# CHƯƠNG 2 – CƠ SỞ LÝ THUYẾT

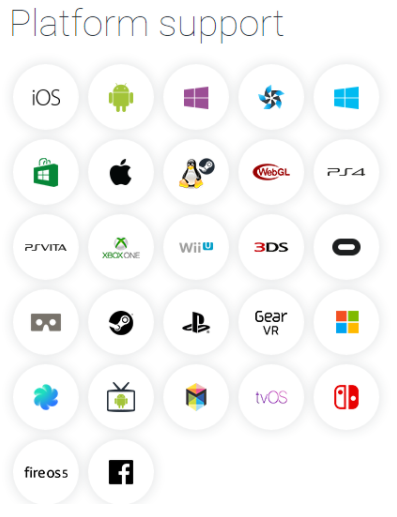
## 2.1 Tổng quan về Unity

1. **Khái niệm**

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 và trên websites.

**b. 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 6: Hỗ trợ đa nền tảng trong Unity

* Dễ sử dụng: Unity3D được built 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 đặc điểm và tính năng của Unity

1. **Rendering (kết xuất hình ảnh)**

* Giống như tất cả các Engine hoàn chỉnh khác, Unity hỗ trợ đầy đủ khả năng kết xuất hình ảnh (Redering) cùng nhiều hỗ trợ cho phép áp dụng các công nghệ phổ biến trong lĩnh vực đồ họa 3D nhằm cải thiện chất lượng hình ảnh. Các phiên bản gần đây nhất của Unity được xây dựng lại thuật toán nhằm cải thiện hiệu suất kết xuất hình ảnh đồng thời tăng cường chất lượng hình ảnh sau khi kết xuất.
* Một số hỗ trợ:
* Unity cung cấp sẵn 100 Shaders với đầy đủ các loại phổ biến nhất.
* Hỗ trợ Surface Shaders, Occlusion Culling, GLSL Optimizer.
* Hỗ trợ LOD.

1. **Lighting (ánh sáng)**

Ánh sáng là một điều thiết yếu giúp môi trường trở nên đẹp và thực tế hơn. Unity cũng cung cấp nhiều giải pháp đa dạng cho phép chúng ta áp dụng ánh sáng một cách tốt nhất vào môi trường trong trò chơi với nhiều loại nguồn sáng như ánh sáng có hướng (Directional Light), ánh sáng điểm (Point Light), ... Một số công nghệ và kỹ thuật về ánh sáng được Unity hỗ trợ: Lingtmapping, Realtime Shadows, hiệu ứng Sunshafts và Lens Flares.

1. **Terrains (địa hình)**

* Terrains còn gọi chung là địa hình bao gồm phần đất nền của môi trường trong trò chơi cùng các đối tượng gắn liền như cây, cỏ, …
* Unity cung cấp một công cụ hỗ trợ rất tốt khả năng này với tên gọi là Terrains Tools cho phép chúng ta thiết kế địa hình với các công cụ vẽ dưới dạng Brush có nhiều thông số tùy chỉnh để tạo hình và lát Texture cho địa hình. Cùng với Terrain Tools là Tree Creator, một công cụ mạnh mẽ cho phép chúng ta tạo ra cây cối với hình dạng, kích thước và kiểu cách đa dạng.

1. **Substances (Texture thông minh)**

Substances có thể hiểu đơn giản là một dạng tùy biến Textures nhằm làm đa dạng chúng trong nhiều điều kiện môi trường khác nhau. Unity cung cấp khả năng này thông qua các API dựng sẵn trong thư viện, hỗ trợ lập trình viên lập trình để tùy biến hình ảnh được kết xuất của Texture

1. **Physics (vật lí)**

PhysX là một Engine mô phỏng và xử lí vật lý cực kỳ mạnh mẽ được phát triển bởi nhà sản xuất card đồ họa hàng đầu thế giới NVIDIA. Unity đã tích hợp Engine này vào để đảm nhận mọi vấn đề vật lý. Một số vấn đề vật lý được hỗ trợ bởi Unity như: Soft Bodies, Rigitbodies, Ragdolls, Joints, Cars, …

1. **Pathfinding (tìm đường)**

Đây là một tính năng rất mới mẻ đến từ phiên bản Unity 3.5. Với các phiên bản trước, để phát triển khả năng tìm đường cho trí thông minh nhân tạo (AI), nhà phát triển phải hoàn toàn tự xây dựng cho mình một hệ thống tìm đường riêng biệt. Tuy nhiên ở phiên bản 3.5 đến nay, Unity hỗ trợ cho chúng ta tính năng Pathfinding cho phép tạo ra khả năng tìm đường cho AI nhờ vào khái niệm lưới định hướng (NavMesh).

1. **Audio (âm thanh)**

Về âm thanh, Unity tích hợp FMOD – công cụ âm thanh thuộc hàng mạnh nhất hiện nay. Qua đó Unity hỗ trợ chúng ta nhập và sử dụng nhiều định dạng tập tin âm thanh khác nhau.

**h. Programming (lập trình)**

* Lập trình là một trong những yếu tố quan trọng nhất trong phát triển Game. Lập trình cho phép nhà phát triển tạo nên khả năng tương tác, trí thông minh và yếu tố Gameplay cho trò chơi.
* Unity cho phép chúng ta lập trình bằng nhiều ngôn ngữ mạnh mẽ và phổ biến với các lập trình viên như: C#, Java Scrip và Boo.

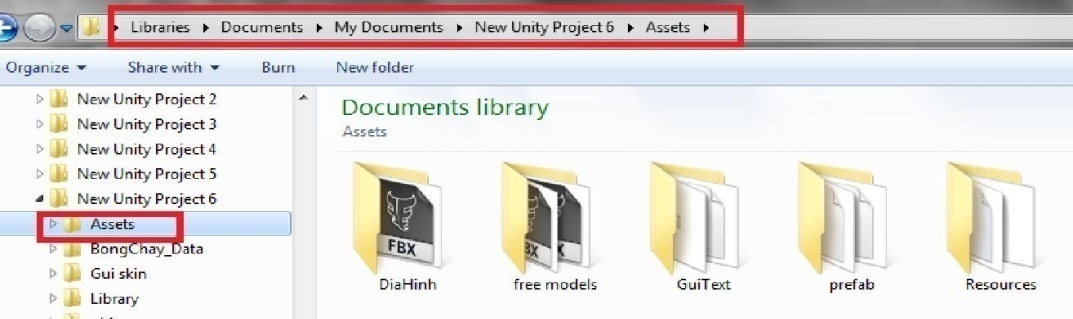
1. **Networking**

* Networking cho phép chúng ta tạo ra các trò chơi trực tuyến (online) – một trong những thể loại trò chơi thu hút được nhiều người chơi nhất. Tính năng này sẽ hỗ trợ đầy đủ để chúng ta tạo nên các khía cạnh phổ biến trong Game online như hệ thống điểm kinh nghiệm , chat và tương tác thời gian thực, …
* Một số tính tăng cung cấp bởi Networking như: State Synchronization, Realtime Networking, Remote Procedure Calls, Backend Connectivity, Web Browser Integration, Web Connectivity.

## 2.3 Các thành phần cơ bản trong Unity

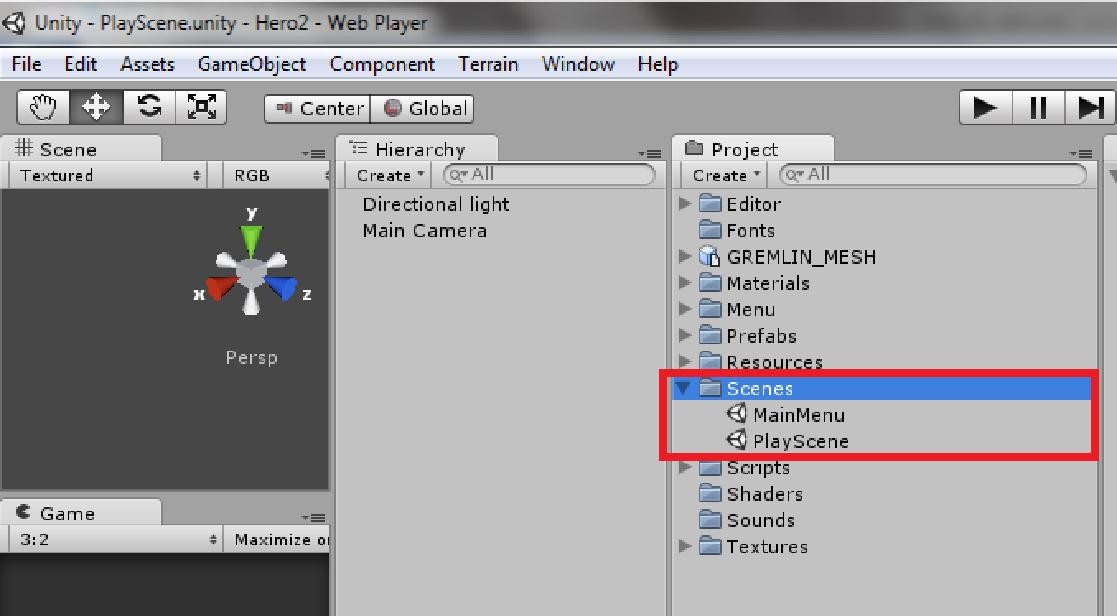
1. **Assets**

Assets là những tài nguyên xây dựng nên một dự án Unity. Từ những tập tin hình ảnh, mô hình 3D đến các tập tin âm thanh. Unity gọi các tập tin mà chúng ta dùng để tạo nên trò chơi là tài sản (Assets). Điều này lí giải tại sao tất cả các tập tin, thư mục của các dự án Unity đều được lưu trữ trong một thư mục có tên là “Assets”.



Hình 7: Asset trong Unity

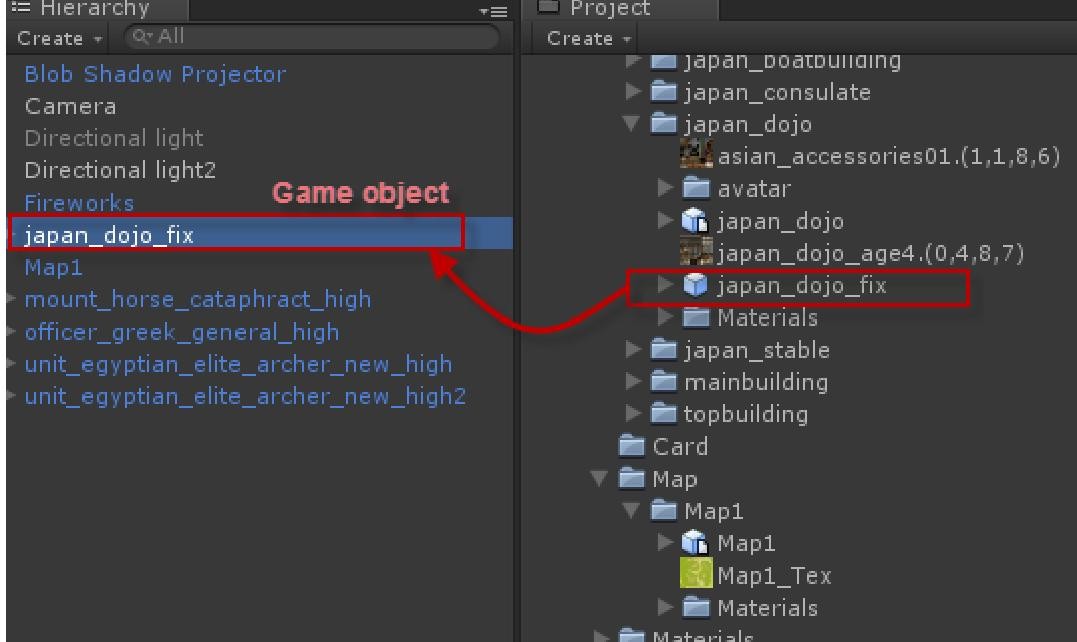
1. **Scenes**

Trong Unity, chúng ta cần hiểu một cảnh (hay một phân đoạn) nghĩa là một màn chơi riêng biệt hoặc một khu vực hay thành phần có trong nội dung của trò chơi (ví dụ như Game menu). Bằng cách tạo nên nhiều Scenes cho trò chơi, chúng ta có thể phân phối thời gian tải hoặc kiểm tra các phần khác nhau của trò chơi một cách riêng lẽ.

Hình 8: Các Scene của Unity

1. **GameObject**

* 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, người...
* Khi Assets được sử dụng trong Scene, chúng trở thành Game Object – một thuật ngữ được sử dụng trong Unity (đặc biệt là trong mảng lập trình). Tất cả các Game Object đều chứa ít nhất một thành phần là Transform. Transform là thông tin về vị trí, góc xoay và tỉ lệ của đối tượng, tất cả được mô tả bởi bộ 3 số X, Y, Z trong hệ trục tọa độ. Thành phần này có thể được tùy biến lại trong quá trình lập trình nhằm thay đổi vị trí, góc quay và tỉ lệ của đối tượng qua các đoạn mã. Từ các thành phần cơ bản này, chúng ta sẽ tạo ra Game Object với các thành phần khác, bổ sung chức năng cần thiết để xây dựng nên bất kỳ một thành phần nào trong kịch bản Game mà chúng ta đã tưởng tượng.

1. **Component**

Hình 9: Kéo tài nguyên vào Scene để sử dụng

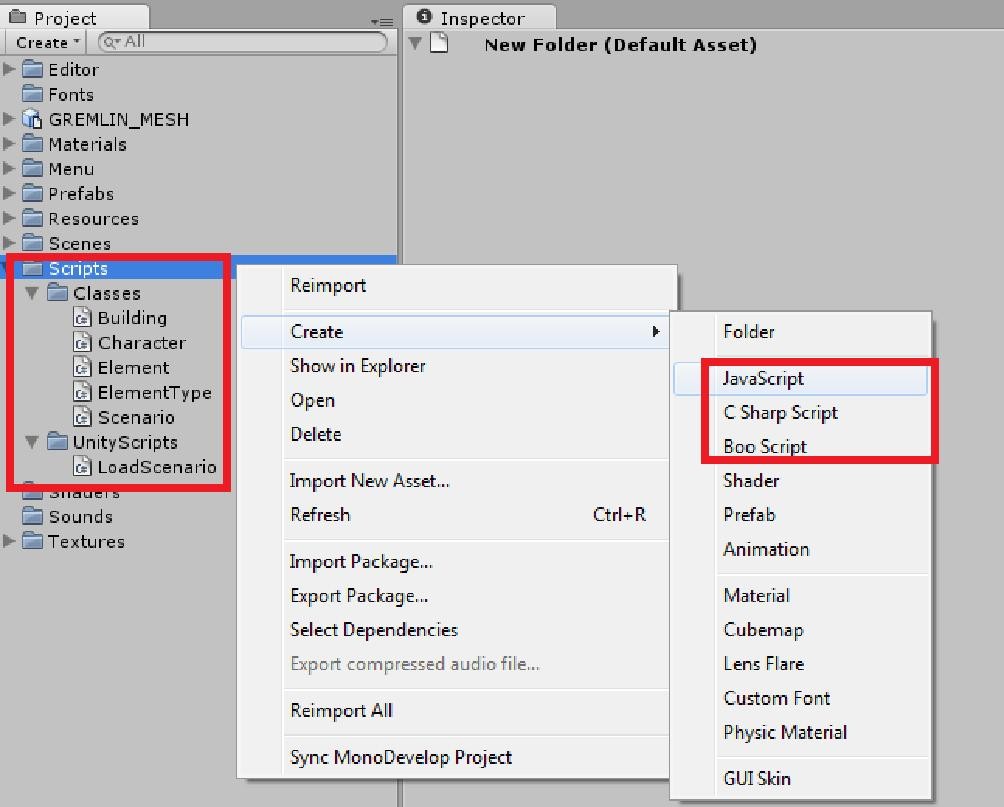
* Một GameObject sẽ có nhiều thành phần cấu tạo nên nó như là hình ảnh (sprite render), tập hợp các hành động (animator), thành phần xử lý va chạm (collision), tính toán vật lý (physical), mã điều khiển (script), các thành phần khác... mỗi thứ như vậy gọi là một component của GameObject.
* Components có nhiều hình thức khác nhau. Chúng có thể xác định hành vi, cách xuất hiện,… hay ảnh hưởng đến các khía cạnh khác trong chức năng của Game Object trong trò chơi. Bằng cách “gắn” chúng vào trong Game Object, chúng ta ngay lập tức có thể áp dụng tác động của chúng lên đối tượng. Những Components phổ biến trong quá trình phát triển trò chơi đều được Unity hỗ trợ sẵn. Ví dụ như thành phần Rigidbody đã được đề cập hay các yếu tố đơn giản khác như ánh sáng, Camera và nhiều thành phần khác. Để tạo nên các yếu tố tương tác trong trò chơi, chúng ta sẽ sử dụng Script (mã kịch bản), chúng cũng được xem như là một Components trong Unity.



1. **Sprite**

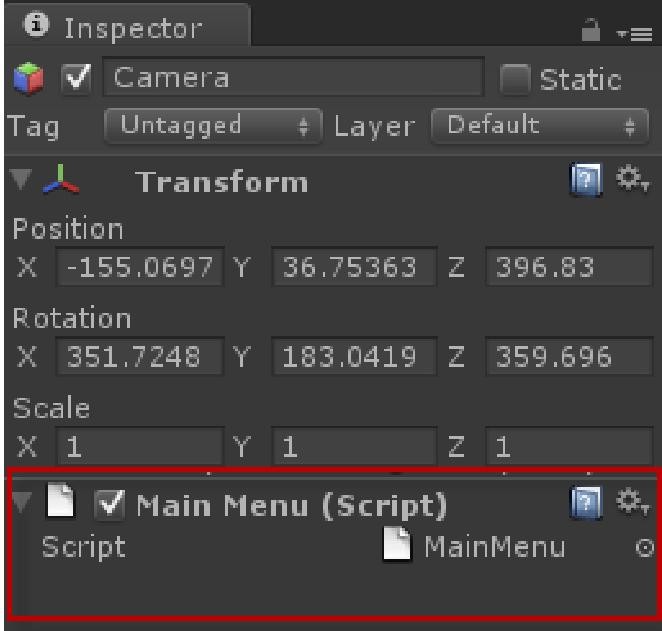
Hình 10: Các thành phần trong đối tượng Camera

* Là một hình ảnh 2D của một game object có thể là hình ảnh đầy đủ, hoặc có thể là một bộ phận nào đó.
* Được Unity xem như một Components, Script là một thành phần thiết yếu trong quá trình phát triển trò chơi và đáng được đề cập đến như một khái niệm “chìa khóa”. Unity cung cấp cho chúng ta khả năng viết Script bằng cả 3 loại ngôn ngữ là: JavaScript, C# và Boo (một dẫn xuất của ngôn ngữ Python).
* Unity không đòi hỏi chúng ta phải học làm thế nào để lập trình trong Unity, nhưng hầu như chúng ta phải sử dụng Script tại mỗi thành phần trong kịch bản mà chúng ta phát triển. Unity đã xây dựng sẵn một tập hợp đa dạng các lớp, hàm mà chúng ta hoàn toàn có thể ứng dụng trong quá trình lập trình cho trò chơi của mình.
* Để viết script, chúng ta sẽ làm việc với một trình biên tập Script độc lập của Unity, hoặc với chương trình Mono Developer được tích hợp và đồng bộ với Unity trong những phiên bản mới nhất hiện nay.
* Mono developer là một IDE khá tốt để lập trình khi cung cấp nhiều chức năng tương tự như Visual studio. Mã nguồn viết trên Mono Developer sẽ được cập nhật và lưu trữ trong dự án Unity.



Hình 11: Cách tạo file script mới

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



Hình 12: Một file Script đang gắn vào đối tượng

1. **Animation**

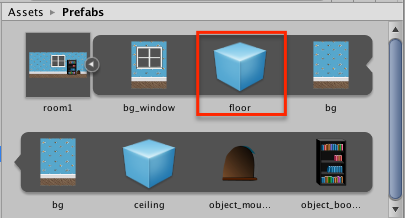
Là tập một hình ảnh động dựa trên sự thay đổi liên tục của nhiều sprite khác nhau.

1. **Key Frame**

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

1. **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 môt đối tượng ban đầu. Ví dụ: Các đối tượng là đồng tiên 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.
* Prefabs cho phép chúng ta lưu trữ các đối tượng với những Components và những thiết đặt hoàn chỉnh. Có thể so sánh với khái niệm cơ bản là MovieClip trong Adobe Flash, Prefabs chỉ đơn giản là một Container (một đối tượng chứa) rỗng mà chúng ta có thể đưa bất kì một đối tượng hay dữ liệu mẫu nào mà chúng ta muốn tái sử dụng về sau.



Hình 13: Một số đối tượng trong Prefabs

1. **Sounds**

Âm thanh trong game.

1. **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 C#, Java Script, BOO để lập trình Script.

1. **Camera**

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

1. **Transform**

Là 3 phép biến đổi tịnh tiến, quay theo các trục, và phóng to thu nhỏ một đối tượng

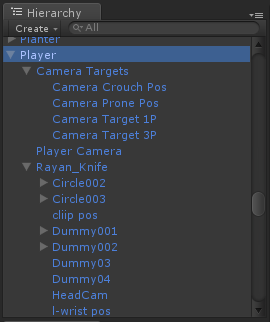
## 2.4 Các thành phần trong Unity Editor

1. **Cửa sổ Sences**

* Phần này phần 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 anh, cách nhìn 2D hay 3D ... -Khung nhìn Scene là nơi bố trí các Game Object 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.

1. **Cửa sổ Hierarchy**

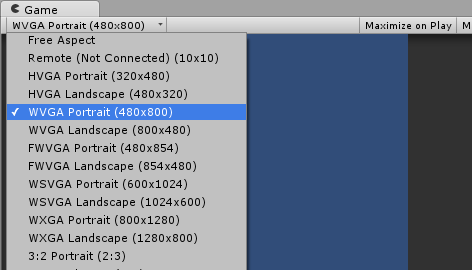
* Tab hierarchy là nơi hiển thị các Game Object trong Sences hiện hành. Khi các đối tượng được thêm hoặc xóa trong Sences, tương ứng với các đối tượng đó trong cửa sổ Hierarchy.Tương tự trong tab Project, Hierarchy cũng có một thanh tìm kiếm giúp quản lý và thao tác với các Game Object hiệu quả hơn đặc biệt là với các dự án lớn.



Hình 14: Cửa sổ Hierarchy.

1. **Cửa sổ Game**

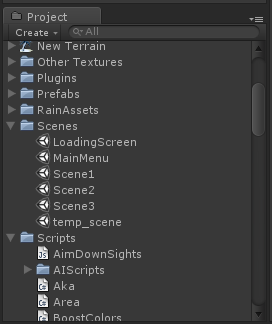
* Đây là mạn hình demo Game, là góc nhìn từ camera trong game.
* Thanh công cụ trong cửa sổ game cung cấp các tùy chỉnh về độ phân giải man hình, thông số (stats), gizmos, tùy chọn bật tắt các component…



Hình 15: Các loại hình ảnh trong cửa sổ game.

1. **Cửa sổ Project**

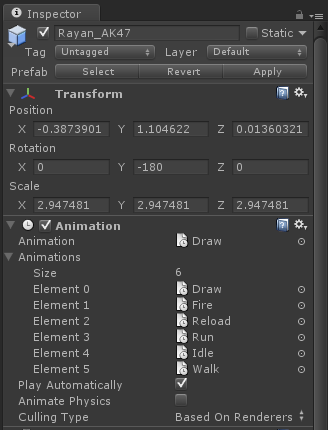
* Đây là cưa sổ explorer của Unity, hiển thị thông tin của tất cả các tài nguyên (Assets) trong game của bạn.
* Cột bên trái hiển thị assets và các mục yêu thích dưới dạng cây thư mục tương tự như Windows Explorer. Khi click vào một nhánh trên cây thư mục thì toàn bộ nội dung của nhánh đó sẽ được hiển thị ở khung bên phải. Ta có thể tạo ra các thư mục mới bằng cách Right click -> Create -> 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 là mục Favorites, giúp chúng ta truy cập nhanh vào những tài nguyên thường sử dụng. Chúng ta có thể đưa các tài nguyên vào Favorites bằng thao tác kéo thả.
* Đường dẫn của thư mục tài nguyên hiện tại. Chúng ta có thể dễ dàng tiếp cận các thư mục con hoặc thư mục gốc bằng cách click chuột vào mũi tên hoặc tên thư mục.



Hình 16: Cửa sổ Project.

1. **Cửa** **sổ Inspector**

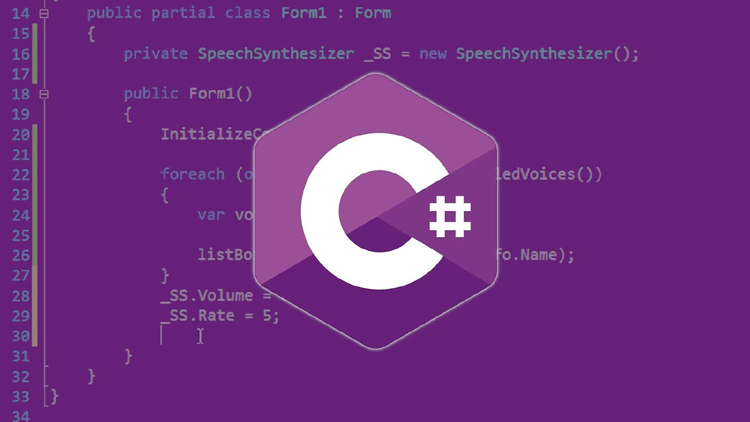
* Cửa sổ Inspector hiển thị chi tiết các thông tin về Game Object đang làm việc, kể cả những component được đính kèm và thuộc tính của nó. Bạn có thể điều chỉnh, thiết lập mọi thông số và chức năng của Game Object thông qua cửa sổ Inspector.
* Mọi thuộc tính thể hiện trong Inspector đều có thể dễ dàng tuỳ chỉnh trực tiếp mà không cần thông qua một kịch bản định trước. Tuy nhiên Scripting API cung cấp một số lượng nhiều và đầy đủ hơn do giao diện Inspector là có giới hạn.
* Các thiết lập của từng component được đặt trong menu. Các bạn có thể click chuột phải, hoặc chọn icon hình bánh răng nhỏ để xuất hiện menu.
* Ngoài ra Inspector cũng thể hiện mọi thông số Import Setting của asset đang làm việc như hiển thị mã nguồn của Script, các thông số animation, …



Hình 17: Cửa sổ Inspector.

## 

## 2.5 Giới thiệu ngôn ngữ lập trình C# (Csharp)



Hình 18: Ngôn ngữ lập trình C# (Csharp)

* C# là một ngôn ngữ lập trình hiện đại được phát triển bởi Microsoft và được phê duyệt bởi European Computer Manufacturers Association (ECMA) và International Standards Organization (ISO).
* C# được phát triển bởi Anders Hejlsberg và nhóm của ông trong việc phát triển .Net Framework.
* C# được thiết kế cho các ngôn ngữ chung cơ sở hạ tầng (Common Language Infrastructure – CLI), trong đó bao gồm các mã (Executable Code) và môi trường thực thi (Runtime Environment) cho phép sử dụng các ngôn ngữ cấp cao khác nhau trên đa nền tảng máy tính và kiến trúc khác nhau

# CHƯƠNG 3 – MỘT SỐ VẤN ĐỀ VÀ GIẢI PHÁP KHI XÂY DỰNG GAME ENGINE TRÊN UNITY

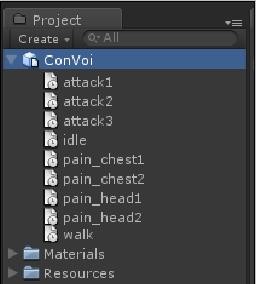
## 3.1 Chuyển động mô hình nhân vật 2D

**a. Vấn đề**

* Chúng ta đã load được mô hình 2D vào trong game, vậy làm sao để mô hình 2D này có thể chuyển động trong game.

**b. Giải pháp**

* Trước tiên mô hình 2D cần phải có sẵn animation bên trong. Khi import mô hình vào Unity, animation trong mô hình được tự động chuyển thành một **AnimationClip**. Điều này giúp animation này có thể dùng cho các mô hình khác trong project.
* Trước hết ta phải tạo AnimationClip từ animation có sẵn của mô hình. Có 2 loại mô hình 2D có sẵn animation:
* *Loại thứ nhất:* Mô hình 2D có chứa nhiều animation bên trong.

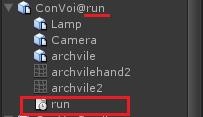


Hình 19: Mô hình 2D bên trong chứa nhiều animation

* Mô hình 2D trên sau khi import vào project game, bên trong đã có sẵn 9 animation, mỗi animation sẽ tự động được tạo thành một AnimationClip bên trong đối tượng game.
* *Loại thứ hai:* Mô hình 2D chỉ chứa một animation.

Hình 20: Mô hình 2D chứa một animation.

* Trong 3 mô hình trên, mỗi mô hình sau khi import có một AnimationClip duy nhất có tên “Default Take”. Về bản chất hình dạng mô hình là như nhau, chỉ khác nhau animation (Idle, Run, Walk). Vậy làm sao chúng ta kết hợp các AnimationClip này vào một đối tượng game duy nhất. Unity quy định như sau:
* Lấy một mô hình làm mô hình chính, có thể không cần animation kèm theo cũng được.
* Các mô hình còn lại, tên phải có 2 phần cách nhau bởi ‘@’, phần đầu phải trùng tên với mô hình chính đã chọn, phần thứ 2 sẽ là tên của animation.
* Với cách đặt tên như vậy, khi đưa các mô hình này vào project để sử dụng thì Unity sẽ tự động đổi tên animation mặc định trong mô hình thành tên trùng với phần tên mô hình nằm sau chữ ‘@’. Lưu ý là phải đổi tên cho mô hình từ bên ngoài project tức trên Windows vì nếu như chúng ta đổi tên trực tiếp trong project thì tên của animation của mô hình đó sẽ không bị thay đổi theo phần tên sau dấu ‘@’.
* Như hình trên, mô hình sẽ có 3 AnimationClip bên trong (Default Take, idle, run).



Hình 21: Hình minh họa sau khi đổi tên và import vào project

* Tiếp theo, để gọi thực hiện một AnimationClip trong một đối tượng ta dùng hàm CrossFade() của thuộc tính animation trong GameObject.
* Đoạn code sau đây đang được gắn vào đối tượng game cần chạy animation.

animation.CrossFade("Run"); // Run là tên của AnimationClip trong mô hình

* Đôi khi ta muốn chạy các chuyển động khác nhau của mô hình một cách tuần tự. Để làm điều này ta dùng hàm ***CrossFadeQueued()*** của thuộc tính **animation** trong GameObject. Để điều khiển cách chạy của AnimationClip, ta dùng thuộc tính **wrapMode**

// AnimationClip Idle sẽ chạy sau khi AnimationClip Run chạy xong animation.CrossFade("Run");

animation.CrossFadeQueued("Idle");

## 3.2 Thêm sự kiện vào chuyển động của nhân vật

**a. Vấn đề**

Trong lúc lập trình kịch bản game, chúng ta muốn biết khi nào một nhân vật chuyển động xong để có bước xử lý tiếp theo. Ví dụ như sau khi nhân vật thực thi chuyển động chết thì chúng ta phải hủy đối tượng đó khỏi bộ nhớ. Trong Unity, khi một chuyển động chạy xong không tự phát ra sự kiện.

**b. Giải pháp**

public class UnitDieCallback : MonoBehaviour

{

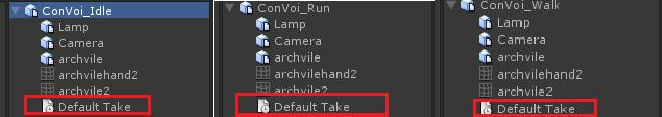
void Die()

{

// hủy đối tượng được attach file script này Destroy(gameObject);

}

}

Unity hỗ trợ lớp **AnimationEvent** giúp thêm sự kiện vảo frame bất kỳ trong một AnimationClip. Trước hết ta phải có một file script chứa hàm sẽ thực thi sau khi sự kiện xảy ra. Chẳng hạn, file script có nội dung như sau (chứa hàm hủy đối tượng):

Sau đó chúng ta tạo file script khác dùng để thêm event và gắn file script này vào đối tượng bất kỳ. Đoạn script mẫu bên dưới đang được gắn vào một mô hình 2D.

public class PlayAnimation : MonoBehaviour

{

void Start()

{

AnimationEvent animEvent = new AnimationEvent(); animEvent.functionName = "Die";

animEvent.time = obj.animation["die"].clip.length;

gameObject.animation["die"].clip.AddEvent(animEvent);

}

}

Với đoạn code trên, chúng ta đã thêm một sự kiện vào frame cuối cùng của AnimationClip “die”. Hàm được thực thi khi sự kiện xảy ra là hàm **Die()**.

Nếu chạy đoạn code trên sẽ xảy ra lỗi thực thi vì không tìm thấy hàm Die(). Chúng ta phải thêm file script chứa hàm Die() vào đối tượng với đoạn code sau:

gameObject.AddComponent<UnitDieCallback>();

Khi chạy game, lúc đối tượng chạy animation “die” xong thì sự kiện sẽ được xảy ra. Nếu chạy animation khác thì sự kiện không xảy ra.

## 3.3 Xây dựng giao diện game

**a. Vấn đề**

Giao diện đồ họa người dùng là một phần quan trọng không thể thiếu trong khi xây dựng một ứng dụng game hay bất cứ một ứng dụng nào để vẽ các đối tượng đồ họa như Button, Label, Checkbox, Slider, … lên màn hình.

**b. Giải pháp**

* Để làm được điều này chúng ta dùng lớp GUI, GUI là chữ viết tắt của “Graphical User Interface” – “Giao diện đồ họa người dùng”. Hệ thống GUI của Unity được gọi là GUIUnity. Để sử dụng được các phương thức trong GUI ta phải gọi thực hiện từ trong hàm **OnGUI()** giống như sự kiện Paint trong C#. Ví dụ sau đây sẽ tạo ra một button đơn giản:

**void OnGUI()**

**{**

**if (GUI.Button(new Rect(10, 10, 150, 100), "I am a button"))**

**{**

**print("You clicked the button!");**

**}**

**}**



Hình 22: GUI Button

* Để tạo một Button hay một thành phần trong GUI đẹp hơn, chúng ta sử dụng **GUIStyle** - style cài đặt cho các thành phần của GUI. GUIStyle chứa thông tin về font chữ, vị trí đặt biểu tượng, hình nền, khoảng cách, … Chỉ cần định nghĩa và áp dụng cho một bất kỳ thành phần nào như Button, Label, Checkbox,… Ta có thể xem GUIStyle như CSS khi thiết kế web site vậy.
* Ví dụ sau tạo một Label theo một định nghĩa GUIStyle, với kích thước font là 14, chữ in đậm, canh lề chữ ở giữa, màu chữ đỏ, font Beckasin (được lưu trước trong thư mục Resources), vị trí vẽ ra (Screen.width/2, Screen.height/2), kích thước hình chữ nhật bao quanh 150x50:

void OnGUI()

{

GUIStyle myStyle = new GUIStyle(); myStyle.fontSize = 14; myStyle.fontStyle = FontStyle.Bold;

myStyle.alignment = TextAnchor.MiddleCenter; myStyle.normal.textColor = Color. yellow;

myStyle.font = (Font)Resources.Load("Fonts/Beckasin");

GUI.Label(new Rect(Screen.width/2,Screen.height/2, 150,50),"This is my label",myStyle);

}



Hình 23: Áp dụng GUIStyle lên Label

* Ví dụ sau đây sẽ vẽ ra một Button với background là hình. Vì chỉ vẽ hình nên chúng ta không cần quan tâm đến định dạng text trong GUIStyle nữa, chúng ta chỉ quan tâm đến hình nền của button với các sự kiện chuột tương tác trên button này mà thôi.

void OnGUI()

{

GUIStyle myStyle = new GUIStyle();

myStyle.normal.background = (Texture2D)Resources.Load("Button/exit"); myStyle.hover.background = (Texture2D)Resources.Load("Button/exit\_hover");

if (GUI.Button(new Rect(Screen.width / 2, Screen.height / 2, 50, 50), "My button", myStyle)) {

* Để vẽ một Image ra màn hình, chúng ta sử dụng Texture2D, nó tương tự đối tượng Bimap hay Image bên ngôn ngữ C#. Để load hình từ Resources và vẽ ra màn hình chúng ta dùng phương thức **DrawTexture()** như sau:

Texture2D myImage = (Texture2D)Resources.Load("background\_image"); GUI.DrawTexture(new Rect(0,0,Screen.width,Screen.height), myImage);

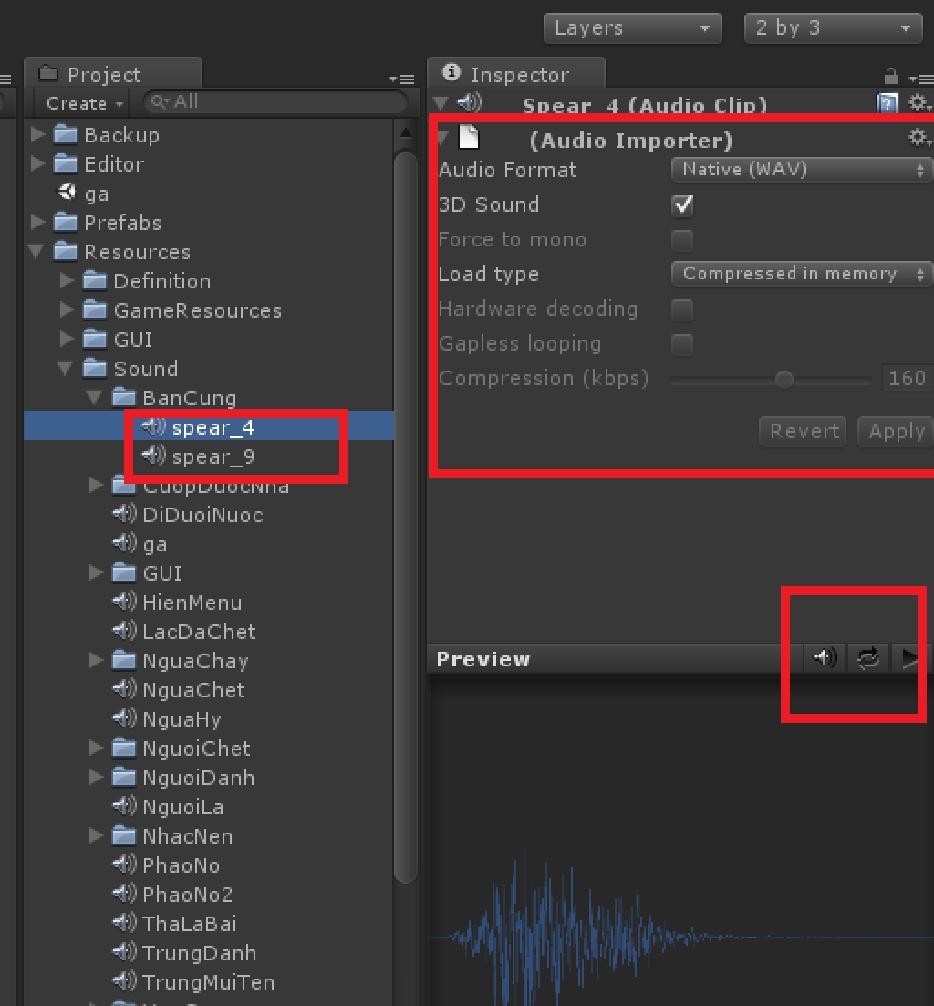
## 3.4 Tạo âm thanh trong game

**a. Vấn đề:**

Âm thanh là yếu tố không kém phần quan trọng trong ứng dụng game. Thật nhàm chán khi một cảnh đánh đánh nhau, bắn nhau hay các hiệu ứng đẹp mắt mà không có âm thanh. Âm thanh 3 chiều sẽ làm cho game thực hơn và sống động hơn.

**b. Giải pháp:**

Để chơi được một file âm thanh trong Unity có 2 cách: bằng code hoặc trên giao diện. Dù chọn cách nào thì trước hết chúng ta phải có sẵn các file âm thanh và import vào project. Sau khi import âm thanh vào project, nếu file hợp lệ chúng ta sẽ thấy như hình sau và có thể nhấn nút play để nghe thử.

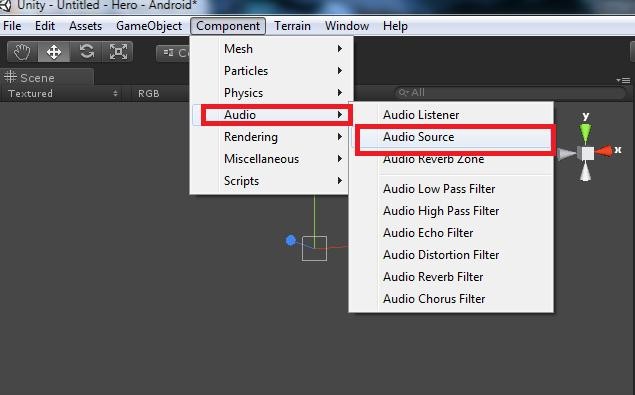


Hình 24: Thông tin file âm thanh.

* *Cách 1:* Tạo trên giao diện

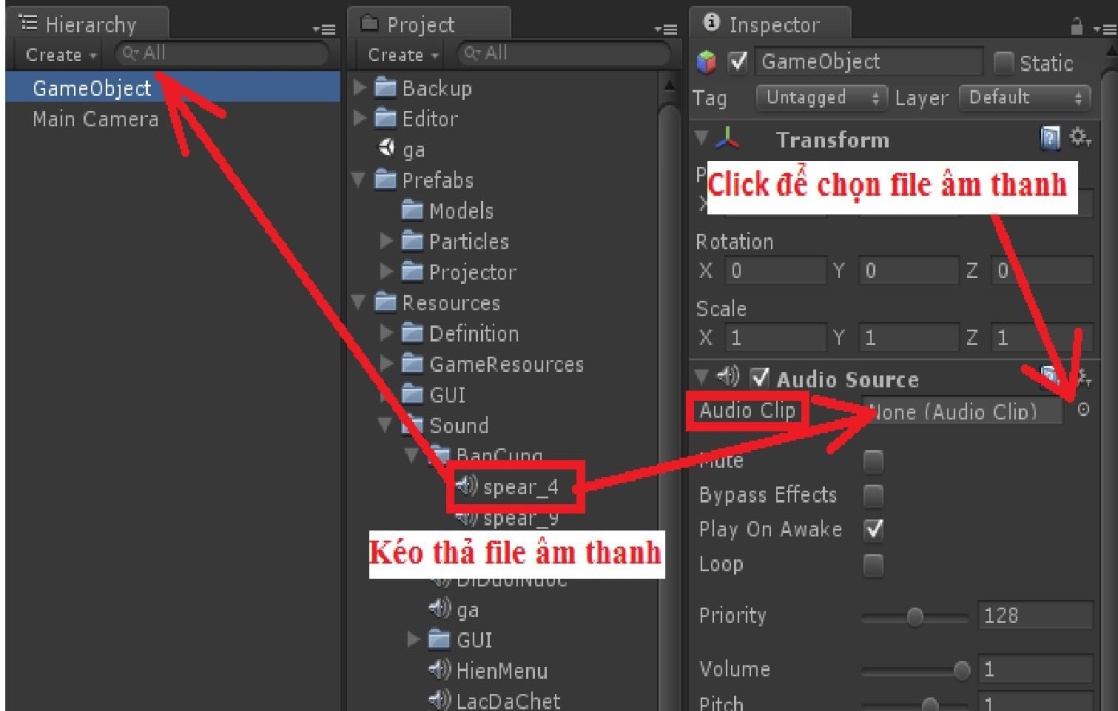
Trên menu của Unity, vào **GameObject Create Empty**.

Chọn đối tượng vừa tạo và gắn thành phần “AudioSource” cho đối tượng này. AudioSource là một đối tượng âm thanh. Muốn Play hay Stop, thay đổi cách lặp, tăng giảm volume nhạc thì phải thông qua đối tượng này.



Hình 25: Thêm thành phần Audio Source

Sau khi gắn thành phần âm thanh cho đối tượng vừa tạo, chúng ta dễ dàng chỉnh sửa các thông số và gán file âm thanh cho thành phần AudioSource này.



Hình 26: Thêm file âm thanh cho thành phần AudioSource

Nếu thực hiện xong các bước trên, chúng ta có thể chạy game để nghe thử. Việc tạo đối tượng âm thanh trên giao diện khá đơn giản, nhưng để áp dụng vào cho game thì không được linh hoạt bằng cách sử dụng script.

* *Cách 2:* Cách chơi nhạc bằng code

Trước tiên chúng ta cần import file âm thanh vào project trước.

Khởi tạo đối tượng game âm thanh **AudioSource** như sau:

AudioSource audioSource = (AudioSource)gameObject.AddComponent("AudioSource");

Gán đường dẫn file nhạc cho âm thanh:

audioSource.clip = (AudioClip)Resources.Load("TenFileAmThanh");

Sau đó chỉ cần gọi các phương thức Play() để chạy file âm thanh:

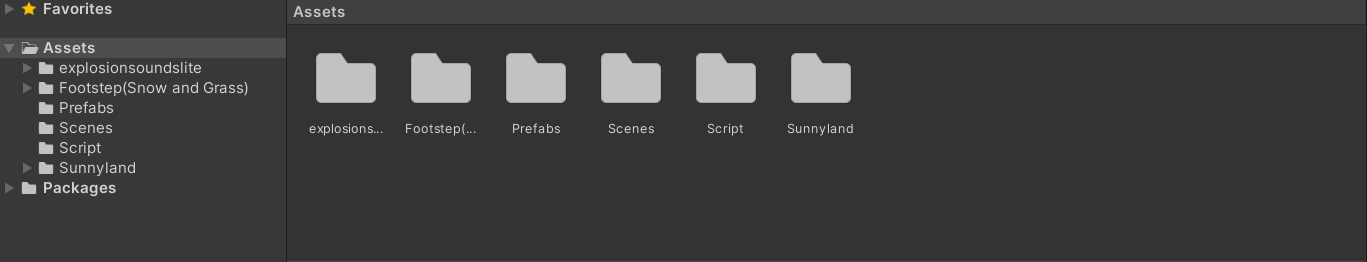
//play nhạc 1 lần audioSource.PlayOneShot();

//play nhạc lặp đi lặp lại audioSource.loop = true; audioSource.Play();

//stop chơi nhạc audioSource.Stop();

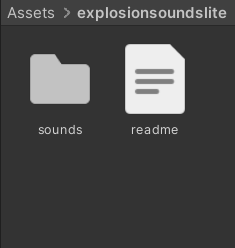
# CHƯƠNG 4 - ỨNG DỤNG GAME PHÁT TRIỂN TRÊN UNITY

## 4.1 Project game FOX 2D



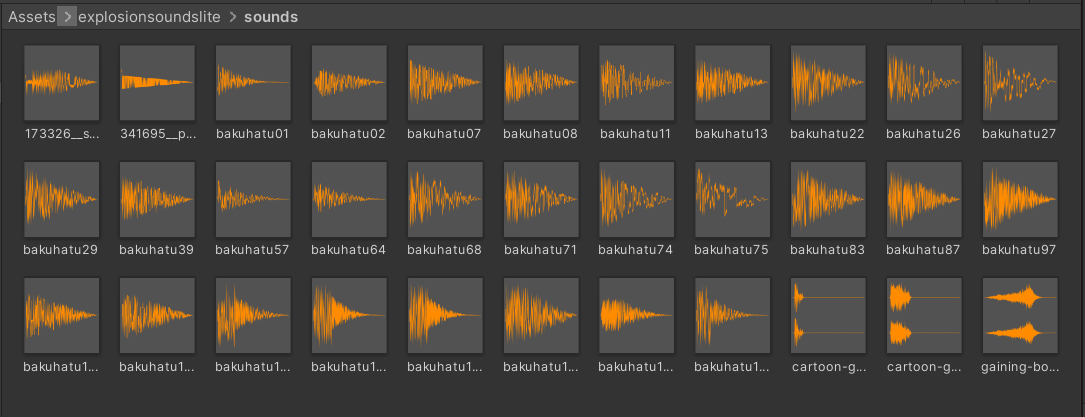
Hình 27: Assets

* Explosionsoundslite:



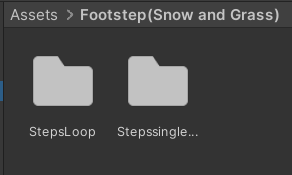
Hình 28: Explosionsoundslite

Sounds:



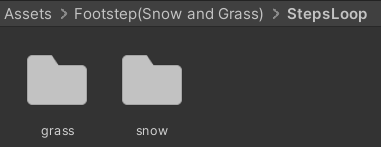
Hình 29: Sounds

* Footstep(Snow and Grass):



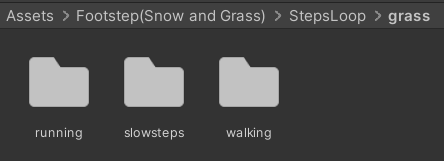
Hình 30: Footstep(Snow and Grass)

* StepsLoop:

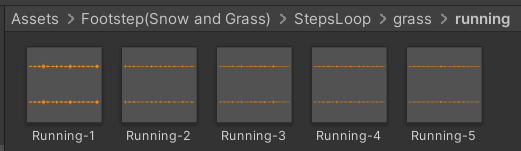


Hình 31: StepsLoop

* grass:



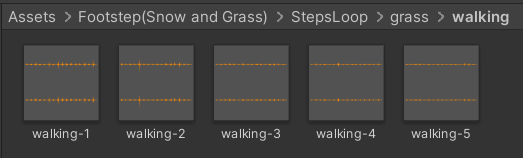
Hình 32: StepsLoop > Grass



Hình 33: Grass > Running

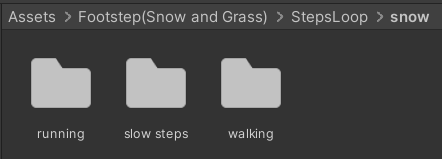


Hình 34: Grass > Slowsteps

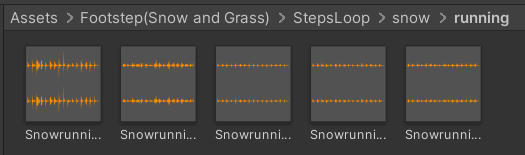


Hình 35: Grass > Walking

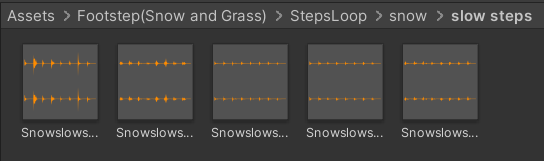
* snow:



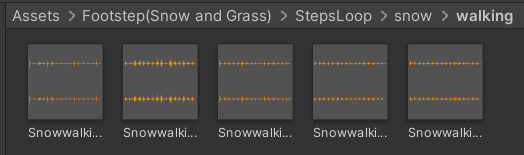
Hình 36: StepsLoop > Snow



Hình 37: StepsLoop > Snow > Running

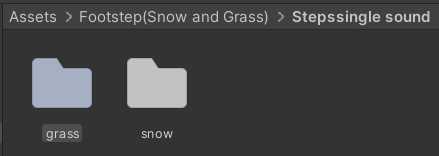


Hình 38: StepsLoop > Snow > Slow steps



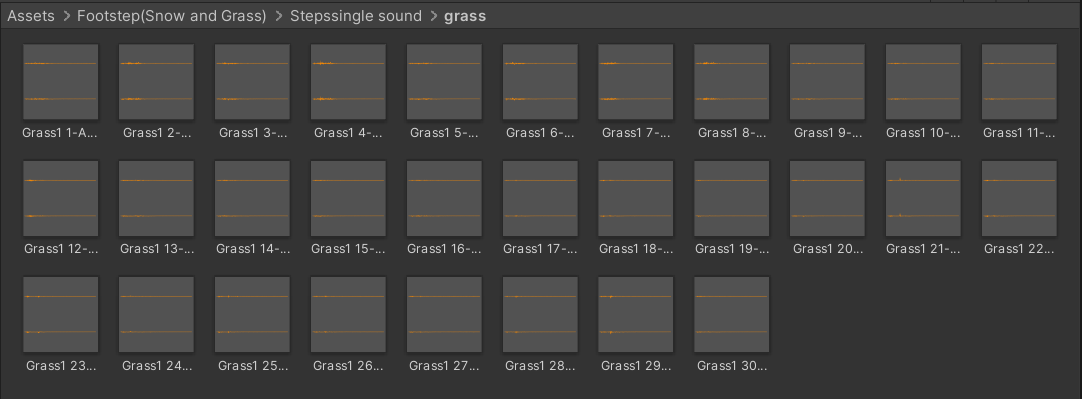
Hình 39: StepsLoop > Snow > Walking

* Stepssingle sound:



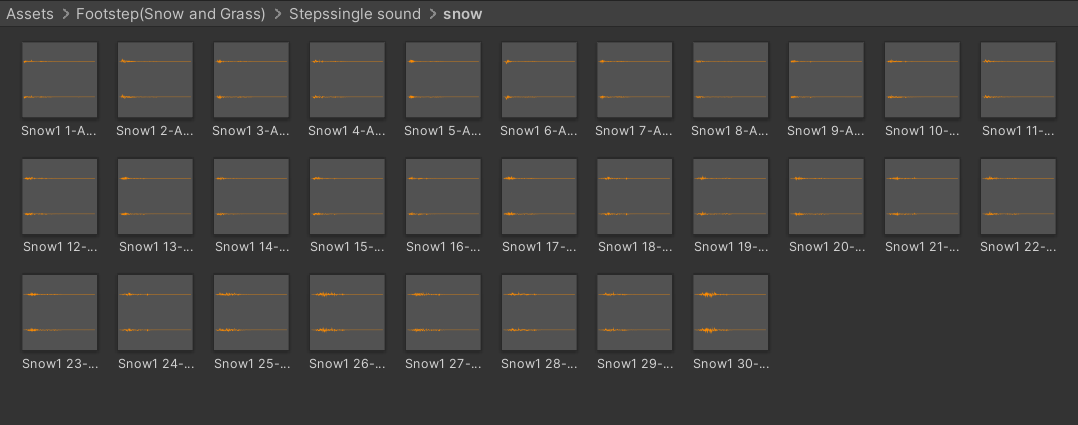
Hình 40: Stepssingle sound

* grass:



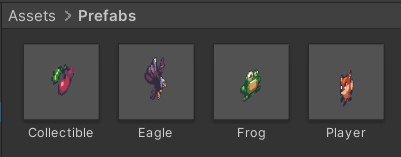
Hình 41: Stepssingle sound > Grass

* snow:



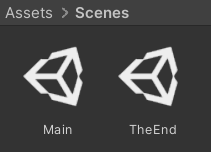
Hình 42: Stepssingle sound > Snow

* Prefabs:



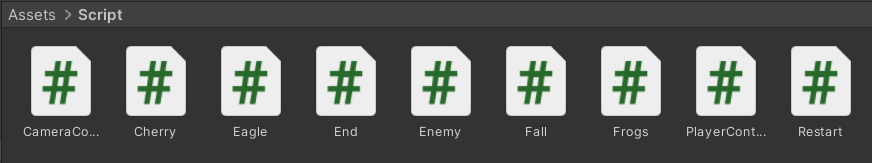
Hình 43: Prefabs

* Scenes:



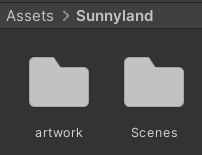
Hình 44: Scenes

* Script:



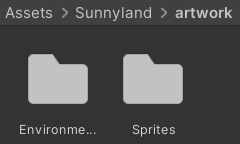
Hình 45: Script

* Sunnyland:



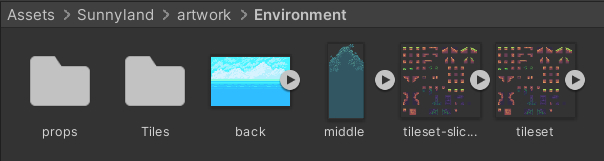
Hình 46: Sunnyland

* artwork:

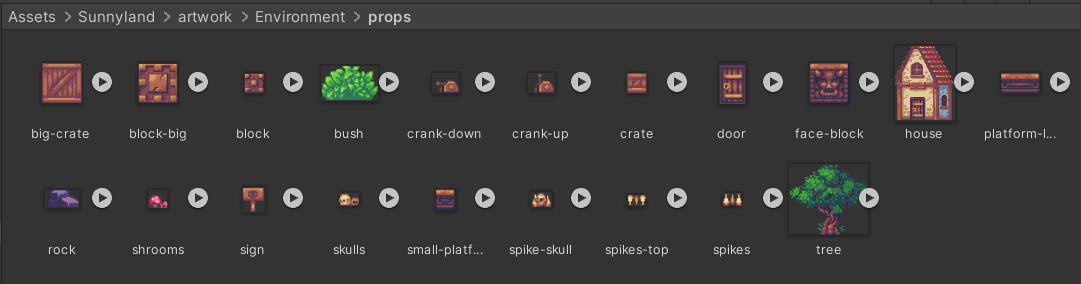


Hình 47: Artwork

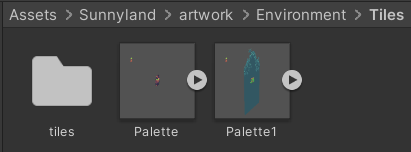
* Environment:



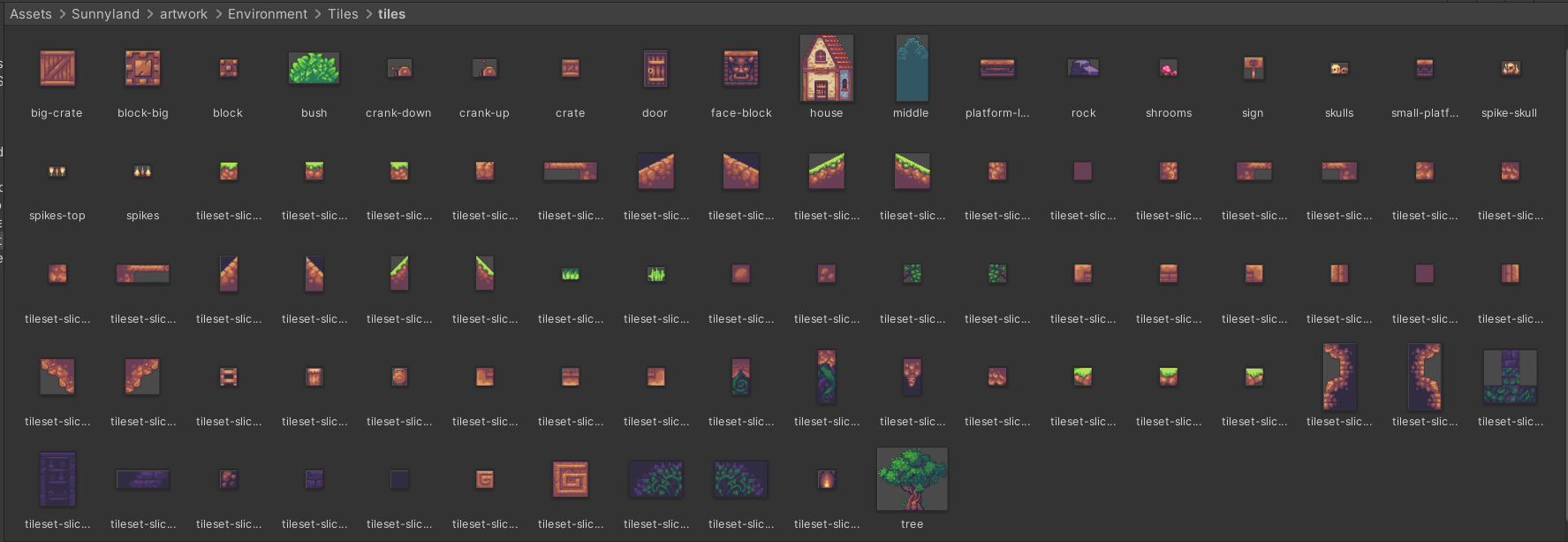
Hình 48: Environment



Hình 49: Props

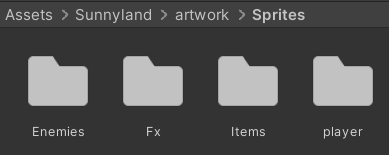


Hình 50: Tiles



Hình 51: Tiles > tiles

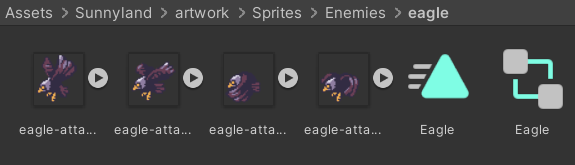
* Sprites:



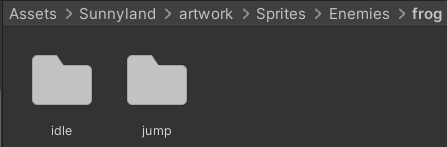
Hình 52: Sprites



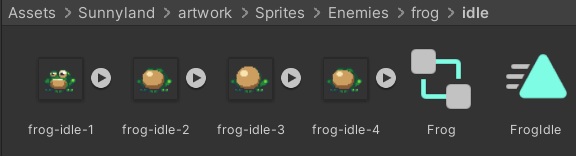
Hình 53: Enemies



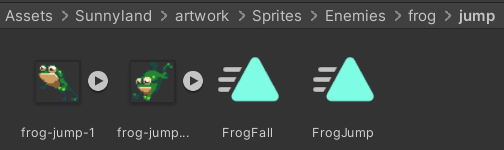
Hình 54: Eagle



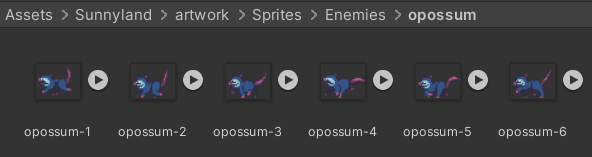
Hình 55: frog



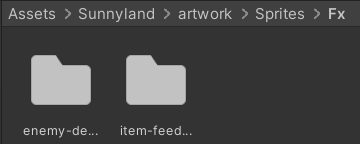
Hình 56: Frog > Idle



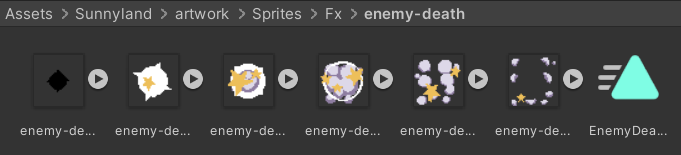
Hình 57: Frog > Jump



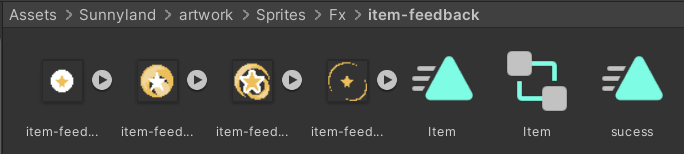
Hình 58: Opossum



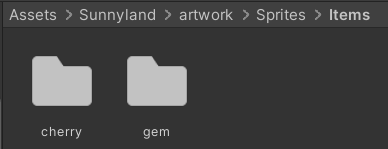
Hình 59: Fx



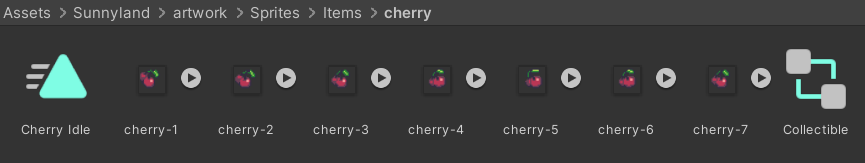
Hình 60: Enemy-death



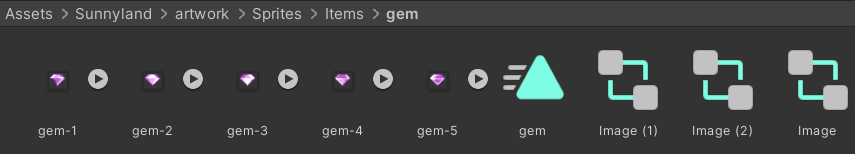
Hình 61: Item-feedback



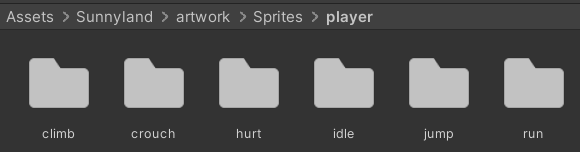
Hình 62: Items



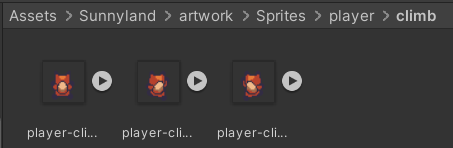
Hình 63: Cherry



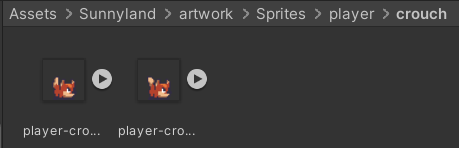
Hình 64: Gem



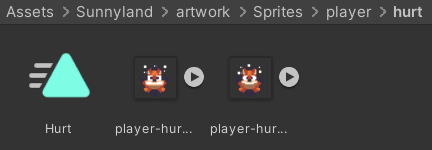
Hình 65: Player



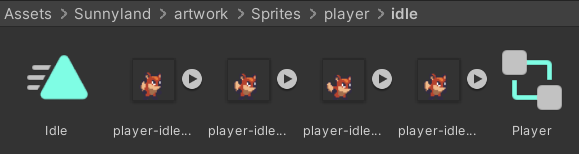
Hình 66: Climb



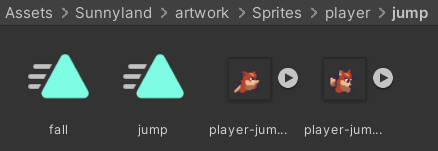
Hình 67: Crouch



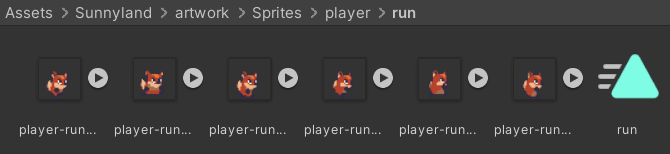
Hình 68: Hurt



Hình 69: Player > Idle

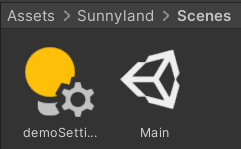


Hình 70: Player > Jump



Hình 71: Player > Run

* Scenes:



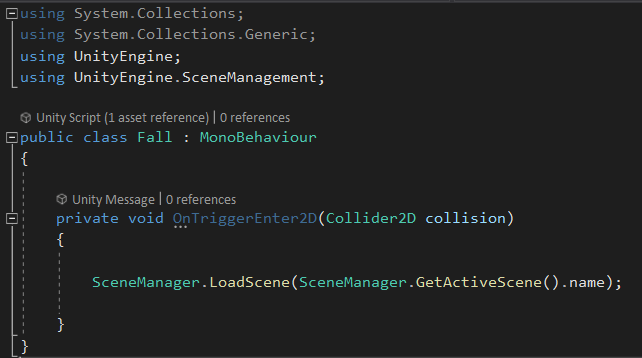
Hình 72: Scenes

## 4.2 Class trong game

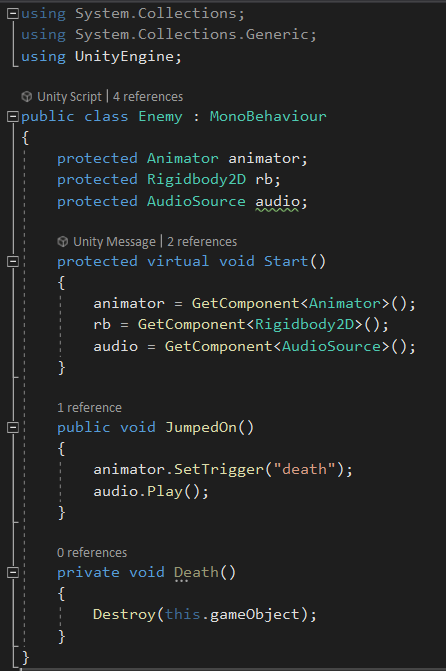
* PlayerController.cs
* CameraController.cs



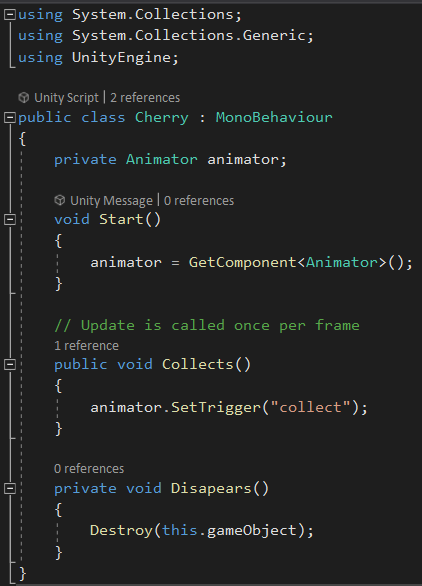
* Fall.cs



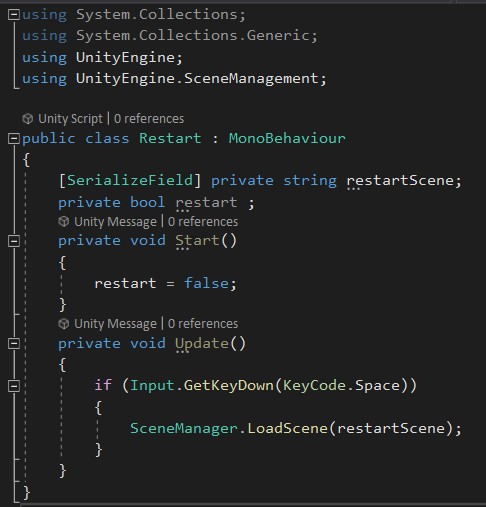
* Enemy.cs



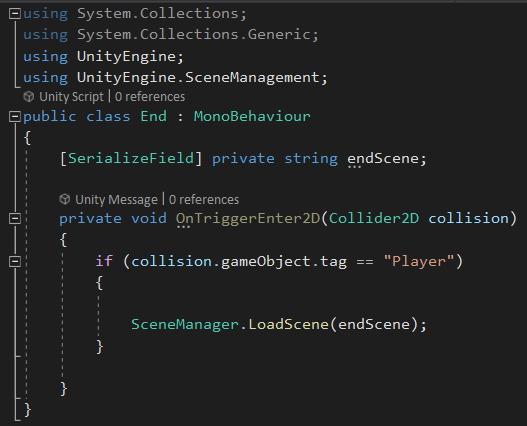
* Cherry.cs



* Restart.cs



* End.cs



## 4.3 Âm thanh trong game

Trong game gồm có âm thanh phayer khi chạy (Hình 33), khi chạy (Hình 29) khi player chạm vào enemy(Hình 29), khi tiêu diệt được enemy (Hình 29), khi ăn quả dâu (Hình 29), khi về đích (Hình 29).

## 4.4 Tạo AI

Player sẽ có animation khi chạy (Hình 71), các con cóc sẽ tự động (Hình 56), nhảy (Hình 57), và biến mất khi bị tiêu diệt.

## 4.5 UI

Trước khi vào game người chơi sẽ thấy một màn hình menu.

Sau khi về đích người chơi sẽ nhận được màn hình kết thúc game (Hình 5) và nếu người muốn chơi lại chỉ cần nhấn nút Space để quay lại màn chơi.

# CHƯƠNG 5 – KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 6.1 Những kết quả đạt được

* Sau khi làm xong đồ án làm một game trên nền tảng Unity nhóm em đã đạt được những kết quả sau:
* Nắm bắt được lý thuyết cơ bản về Unity.
* Học được những kiến thức cơ bản về làm game cũng như lập trình thực tế trên ngôn ngữ C#.
* Thực hành làm game 2D đơn giản (game FOX 2D).

## 6.2 Những hạn chế

* Bên cạnh những kết quả đạt được ở trên, nhóm em nhận thấy còn có những hạn chế cần khắc phục sau:
* Do kiến thức về Unity còn hạn hẹp nên làm game còn đơn giản.
* Hiệu ứng vẫn còn ít
* Chưa tạo được nhiều màn chơi

## 6.3 Hướng phát triển

Hướng phát triển sau này của nhóm em đó chính là hoàn thiện hơn những vấn đề chưa được giải quyết, nghiên cứu thêm các kỹ thuật làm game như làm game phức tạp hơn, tạo game 3D và tìm hiểu những công cụ mạnh hơn cho việc xây dựng đồ họa của game.

**TÀI LIỆU THAM KHẢO**

**Tiếng Anh**

1. Alvin Roe. Platformẻr Tutorial. Truy cập 10/11/2021, <https://www.youtube.com/watch?v=rMr1sHQ0_bc&list=PLpj8TZGNIBNy51EtRuyix-NYGmcfkNAuH>
2. Unity Documentation, Scripting API, Truy cập 10/11/2021, <https://docs.unity3d.com/ScriptReference/>
3. Sunny Land Preview. Truy cập 10/11/2021,

<https://www.youtube.com/watch?v=C6MSh-jENos&t=2s>