**ỦY BAN NHÂN DÂN TỈNH BÌNH DƯƠNG**

**TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT**

**KHOA KỸ THUẬT – CÔNG NGHỆ**

**BÁO CÁO TỐT NGHIỆP**

**XÂY DỰNG GAME JUMP UP SỬ DỤNG ADOBE PHOTOSHOP VÀ UNITY ENGINE TRÊN HỆ ĐIỀU HÀNH ANDROID**

**GVHD: TH.S VÕ QUỐC LƯƠNG**

**SVTH: LÊ DUY MINH**

**MSSV: 1524801030072**

**LỚP: D15PM02**

**BÌNH DƯƠNG – 5/2019**

**ỦY BAN NHÂN DÂN TỈNH BÌNH DƯƠNG**

**TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT**

**KHOA KỸ THUẬT – CÔNG NGHỆ**



**BÁO CÁO TỐT NGHIỆP**

**XÂY DỰNG GAME JUMP UP SỬ DỤNG ADOBE PHOTOSHOP VÀ UNITY ENGINE TRÊN HỆ ĐIỀU HÀNH ANDROID**

**GVHD: TH.S NGUYỄN VĂN B**

**SVTH: NGUYỄN VĂN A**

**MSSV: 03454566565**

**LỚP: DLB16HT201**

**BÌNH DƯƠNG – 5/2017**

LỜI CẢM ƠN

Người thực hiện đề tài xin chân thành cảm ơn Khoa Kỹ Thuật Công Nghệ , Trường Đại học Thủ Dầu Một đã tạo điều kiện tốt cho người thực hiện thực hiện đề tài này.

Người thực hiện đề tài xin chân thành cảm ơn Thầy Võ Quốc Lương, là người đã tận tình hướng dẫn người thực hiện đề tài, chỉ bảo người thực hiện đề tài trong suốt thời gian thực hiện đề tài. Cũng như đã có những trao đổi, những chỉ dẫn giúp người thực hiện đề tài giải quyết các vấn đề và hoàn thiện đề tài.

Người thực hiện đề tài xin gửi lòng biết ơn sâu sắc đến Ba, Mẹ, các anh chị và bạn bè đã ủng hộ, giúp đỡ và động viên trong những lúc khó khăn cũng như trong suốt thời gian học tập và nghiên cứu.

Mặc dù người thực hiện đề tài đã cố gắng hoàn thành luận văn trong phạm vi và khả năng cho phép, nhưng chắc chắn sẽ không tránh khỏi những thiếu sót, kính mong sự cảm thông và tận tình chỉ bảo của quý Thầy Cô và các bạn.

MỤC LỤC

[LỜI CẢM ƠN ii](#_Toc8488373)

[MỤC LỤC iii](#_Toc8488374)

[DANH MỤC HÌNH vi](#_Toc8488375)

[DANH MỤC BẢNG viii](#_Toc8488376)

[DANH SÁCH CÁC KÝ TỰ, CHỮ VIẾT TẮT ix](#_Toc8488377)

[MỞ ĐẦU x](#_Toc8488378)

[CHƯƠNG 1. TỔNG QUAN VỀ UNITY ENGINE VÀ KHÔNG GIAN 2D, 3D 1](#_Toc8488379)

[1.1. TỔNG QUAN VỀ UNITY ENGINE 1](#_Toc8488380)

[1.1.1. Unity là gì ? 1](#_Toc8488381)

[1.1.2. Sơ Lược Lịch Sử Hình Thành Và Phát Triển Của Unity 3](#_Toc8488382)

[1.1.3. Tính năng của Engine Unity 4](#_Toc8488383)

[1.2. TỔNG QUAN VỀ GAME ENGINE 5](#_Toc8488384)

[1.2.1. Khái niệm 5](#_Toc8488385)

[1.2.2. Lịch sử 5](#_Toc8488386)

[1.2.3. Mục đích 6](#_Toc8488387)

[1.2.4. Ảo hóa phần cứng 6](#_Toc8488388)

[1.2.5. Xu hướng hiện tại 7](#_Toc8488389)

[1.2.6. Phần mềm trung gian 7](#_Toc8488390)

[1.2.7. Phân loại Game Engine theo mức độ chuyên biệt 8](#_Toc8488391)

[1.3. TỔNG QUAN KIẾN TRÚC GAME ENGINE 9](#_Toc8488392)

[1.3.1. Kiến trúc tổng quan 9](#_Toc8488393)

[CHƯƠNG 2. UNITY ENGINE 12](#_Toc8488394)

[2.1. GIỚI THIỆU CHUNG 12](#_Toc8488395)

[2.2. CÁC ĐẶC ĐIỂM VÀ TÍNH NĂNG CỦA UNITY 13](#_Toc8488396)

[2.2.1. Rendering (Kết xuất hình ảnh) 13](#_Toc8488397)

[2.2.2. Lighting (Ánh sáng) 13](#_Toc8488398)

[2.2.3. Terrains (Địa hình) 14](#_Toc8488399)

[2.2.4. Substances (Texture thông minh) 14](#_Toc8488400)

[2.2.5. Physics (Vật lý) 14](#_Toc8488401)

[2.2.6. . Pathfinding (tìm đường) 14](#_Toc8488402)

[2.2.7. Audio 14](#_Toc8488403)

[2.2.8. Programming (lập trình) 14](#_Toc8488404)

[2.2.9. Networking 15](#_Toc8488405)

[2.3. CÁC THÀNH PHẦN TRONG UNITY 15](#_Toc8488406)

[2.3.1. Assets 15](#_Toc8488407)

[2.3.2. Scenes 16](#_Toc8488408)

[2.3.3. Game Object 16](#_Toc8488409)

[2.3.4. Components 17](#_Toc8488410)

[2.3.5. Scripts 17](#_Toc8488411)

[2.3.6. Prefabs 19](#_Toc8488412)

[2.4. GIAO DIỆN CỦA UNITY 19](#_Toc8488413)

[2.4.1. Giao diện 19](#_Toc8488414)

[2.4.2. Cửa sổ Scene và Hierarchy 20](#_Toc8488415)

[2.4.3. Cửa sổ Inspector 21](#_Toc8488416)

[2.4.4. Cửa sổ Project 22](#_Toc8488417)

[2.4.5. Cửa sổ Game 23](#_Toc8488418)

[CHƯƠNG 3. ADOBE PHOTOSHOP CHO UNITY ENGINE 25](#_Toc8488419)

[3.1. GIỚI THIỆU CHUNG 25](#_Toc8488420)

[3.1.1. Tổng quan 25](#_Toc8488421)

[3.1.2. Lịch sử 25](#_Toc8488422)

[3.2. PHOTOSHOP TRONG UNITY 26](#_Toc8488423)

[3.2.1. Sprite 27](#_Toc8488424)

[3.2.2. Animation 27](#_Toc8488425)

[3.2.3. Import file Photoshop vào Unity 28](#_Toc8488426)

[CHƯƠNG 4. ỨNG DỤNG GAME JUMP UP PHÁT TRIỂN TRÊN UNITY 31](#_Toc8488427)

[4.1. CÁC SƠ ĐỒ TRONG GAME 31](#_Toc8488428)

[4.1.1. Sơ đồ Use-case người chơi 31](#_Toc8488429)

[4.1.2. Sơ đồ tuần tự chơi game của người chơi 31](#_Toc8488430)

[4.1.3. Sơ đồ tuần tự mua nhân vật mới 32](#_Toc8488431)

[4.2. GIỚI THIỆU GAME JUMP UP 32](#_Toc8488432)

[4.2.1. Thể loại 32](#_Toc8488433)

[4.2.2. Nền tảng 33](#_Toc8488434)

[4.3. CÁC QUY LUẬT CHƠI CHÍNH 34](#_Toc8488435)

[4.3.1. Di chuyển 34](#_Toc8488436)

[4.3.2. Điểm số 34](#_Toc8488437)

[4.3.3. Item 34](#_Toc8488438)

[4.3.4. Game Over 35](#_Toc8488439)

[4.3.5. High Score 36](#_Toc8488440)

[4.3.6. Tạm dừng chơi 37](#_Toc8488441)

[4.4. GIAO DIỆN CHÍNH TRONG GAME 38](#_Toc8488442)

[4.4.1. Giao diện màn hình chính 38](#_Toc8488443)

[4.4.2. Giao diện volume 38](#_Toc8488444)

[4.4.3. Giao diện hướng dẫn chơi game 39](#_Toc8488445)

[4.4.4. Giao diện điểm cao 39](#_Toc8488446)

[4.4.5. Giao diện thoát game 40](#_Toc8488447)

[4.4.6. Giao diện cửa hàng 40](#_Toc8488448)

[4.4.7. Giao diện màn hình chơi chính 41](#_Toc8488449)

[4.4.8. Giao diện dừng chơi 41](#_Toc8488450)

[4.4.9. Giao diện gameover 42](#_Toc8488451)

[4.4.10. Giao diện đạt điểm cao mới 42](#_Toc8488452)

[4.5. MỘT SỐ CODE QUAN TRỌNG TRONG GAME 43](#_Toc8488453)

[4.5.1. Di chuyển camera. 43](#_Toc8488454)

[4.5.2. Khởi tạo một platform mới. 44](#_Toc8488455)

[CHƯƠNG 5. BUILD GAME RA HỆ ĐIỀU HÀNH ANDROID 48](#_Toc8488456)

[5.1. CÁC BƯỚC THỰC HIỆN 48](#_Toc8488457)

[KẾT LUẬN 53](#_Toc8488458)

[TÀI LIỆU THAM KHẢO 55](#_Toc8488459)

DANH MỤC HÌNH

[**Hình 1.1:** Sự đa nền tảng của unity 2](#_Toc8488322)

[**Hình 1.2:** Tổng quan kiến trúc Unity 10](#_Toc8488323)

[**Hình 2.1:** Logo của Unity Engine 12](#_Toc8488324)

[**Hình 2.2:** Asset trong Unity 15](#_Toc8488325)

[**Hình 2.3:** Các Scene của Unity 16](#_Toc8488326)

[**Hình 2.4:** Kéo tài nguyên vào Scene để sử dụng 16](#_Toc8488327)

[**Hình 2.5:** Các component trong đối tượng Camera 17](#_Toc8488328)

[**Hình 2.6:** Cách tạo file script mới 18](#_Toc8488329)

[**Hình 2.7:** Một file script đang gắn vào một đối tượng 19](#_Toc8488330)

[**Hình 2.8:** Một số đối tượng trong Prefabs 19](#_Toc8488331)

[**Hình 2.9:** Giao diện của Unity 20](#_Toc8488332)

[**Hình 2.10:** Các nút chức năng cho của sổ Scene 20](#_Toc8488333)

[**Hình 2.11:** Cửa sổ Hierarchy. 21](#_Toc8488334)

[**Hình 2.12:** Cửa sổ Inspector 22](#_Toc8488335)

[**Hình 2.13:** Cửa sổ Project 23](#_Toc8488336)

[**Hình 2.14:** Các kích thước màn hình của Game 24](#_Toc8488337)

[**Hình 3.1:** Logo của Adobe Photoshop 25](#_Toc8488338)

[**Hình 3.2:** Giao diện của Photoshop CC 2015 26](#_Toc8488339)

[**Hình 3.3:** Minh họa một sprite 27](#_Toc8488340)

[**Hình 3.4:** Ảnh minh họa một sprite sheet 28](#_Toc8488341)

[**Hình 3.5:** Skeletal 2D 28](#_Toc8488342)

[**Hình 3.6:** Import một file photoshop 29](#_Toc8488343)

[**Hình 3.7:** Import file photoshop 30](#_Toc8488344)

[**Hình 4.1:** Biểu đồ use-case người chơi 31](#_Toc8488345)

[**Hình 4.2:** Sơ đồ tuần tự chơi game của người chơi 32](#_Toc8488346)

[**Hình 4.3:** Sơ đồ tuần tự chơi mua nhân vật mới 32](#_Toc8488347)

[**Hình 4.4:** Di chuyển trong game 34](#_Toc8488348)

[**Hình 4.5:** Điểm số 34](#_Toc8488349)

[**Hình 4.6:** Item trong game 35](#_Toc8488350)

[**Hình 4.7:** Giao diện cửa hàng 35](#_Toc8488351)

[**Hình 4.8:** Giao diện game over 36](#_Toc8488352)

[**Hình 4.9:** Giao diện nhập tên khi được điểm cao 36](#_Toc8488353)

[**Hình 4.10:** Giao diện leaderboard 37](#_Toc8488354)

[**Hình 4.11:** Giao diện khi tạm dừng chơi 37](#_Toc8488355)

[**Hình 4.12:** Giao giạn mành hình chính 38](#_Toc8488356)

[**Hình 4.13:** Giao diện tăng giảm âm thanh 38](#_Toc8488357)

[**Hình 4.14:** Giao diện hướng dẫn chơi game 39](#_Toc8488358)

[**Hình 4.15:** Giao diện xem điểm cao 40](#_Toc8488359)

[**Hình 4.16:** Giao diện thoát game 40](#_Toc8488360)

[**Hình 4.17:** Giao diện cửa hàng 40](#_Toc8488361)

[**Hình 4.18:** Giao diện màn hình chơi chính 41](#_Toc8488362)

[**Hình 4.19:** Giao diện dừng chơi 41](#_Toc8488363)

[**Hình 4.20:** Giao diện gameover 42](#_Toc8488364)

[**Hình 4.21:** Giao diện đạt điểm cao mới 42](#_Toc8488365)

[**Hình 5.1:** Build Settings 48](#_Toc8488366)

[**Hình 5.2:** Lựa chọn hệ điều hành để build 49](#_Toc8488367)

[**Hình 5.3:** Player Settings 49](#_Toc8488368)

[**Hình 5.4:** Giao diện build ra file android 50](#_Toc8488369)

[**Hình 5.5:** Giao diện game đang được build 50](#_Toc8488370)

[**Hình 5.6:** Thông báo sau khi build game xong 51](#_Toc8488371)

[**Hình 5.7:** Game sao khi cài lên thiết bị 52](#_Toc8488372)

DANH MỤC BẢNG

[**Bảng 4.1:** Thị phần các thiết bị di động năm 2017 theo doanh số do Gartner công bố. 31](#_Toc8262718)

DANH SÁCH CÁC KÝ TỰ, CHỮ VIẾT TẮT

|  |  |
| --- | --- |
| **Từ viết tắt** | **Giải thích** |
| **Tiếng Việt** | |
| ĐVT | Đơn vị tính |
| NTH | Người thực hiện |
| NC | Người chơi |
| SP | Sản phẩm |
| **Tiếng Anh** | |
| UC | Use Case |

MỞ ĐẦU

Trong thời đại công nghệ thông tin như hiện nay, sản phẩm công nghệ ngày càng chịu sự đánh giá khắt khe hơn từ phía những người dùng, đặc biệt là về sản phẩm Game được nhận rất nhiều sự đánh giá từ phía các Game thủ, hay chỉ là những người chơi bình thường. Ngành công nghiệp Game hiện nay có thể nói là bùng nổ, với tốc độ phát triển đến chóng mặt, rất nhiều những Game hay và hấp dẫn đã được ra đời trong thời gian qua. Phía sau những Game phát triển và nổi tiếng như vậy đều có một Game Engine. Game Engine là một công cụ hỗ trợ, một Middleware giúp người phát triển viết Game một cách nhanh chóng và đơn giản, đồng thời cung cấp khả năng tái sử dụng các tài nguyên và mã nguồn cao do có thể phát triển nhiều Game từ một Game Engine.

Từ xu hướng phát triển và những bất cập trên, đồ án này sẽ khảo sát và nghiên cứu về Engine Unity – một Game Engine rất phổ biến và không kém mạnh mẽ hiện nay nhằm thực nghiệm việc phát triển một trò chơi (Demo) Jump up 2D. Chuẩn bị kiến thức và kỹ năng cho định hướng nghề nghiệp (phát triển Game) sau này của người thực hiện đề tài, góp phần vào sự phát triển của ngành công nghiệp Game nước nhà. Các chương đầu trong tài liệu sẽ trình bày lần lượt các khái niệm chung về Game Engine và thế giới 2D.

Các chương tiếp theo sẽ giới thiệu về Engine Unity bao gồm các đặc điểm, tính năng, công cụ và thành phần trong Engine này. Sau những nội dung về thiết kế, tài liệu sẽ trình bày về việc lập trình trên Unity khi giới thiệu về ngôn ngữ lập trình cùng các lớp, hàm trong thư viện dựng sẵn của Unity thông qua các ví dụ thực tế khi phát triển một game Jump up 2D. Cuối cùng là giới thiệu về bối cảnh, tài liệu hướng dẫn sử dụng của trò chơi Demo cùng chương tổng kết về các khó khăn và những thành quả trong suốt quá trình phát triển trò chơi Demo trên Unity.

1. TỔNG QUAN VỀ UNITY ENGINE VÀ KHÔNG GIAN 2D, 3D
   1. TỔNG QUAN VỀ UNITY ENGINE
      1. Unity là gì ?

Đã qua rồi thời kỳ làm game trên nền Flash căn bản và buồn chán với những chuyển động thật cứng nhắc. Unity mang lại sức mạnh kỳ diệu cho nhân vật mà chúng ta muốn thể hiện sống động hơn trong không gian 3 chiều đầy huyền ảo. Công nghệ cao này tạo ra một bước đột phá mới về sự khác biệt trong công nghệ làm game hiện nay, mang đến cho người chơi 1 cảm giác rất khác lạ và hào hứng trong từng chuyển động, tương lai công nghệ này được áp dụng vào game Việt Nam sẽ mở ra một trang mới trong thế giới game 2D, 3D huyền ảo.

Unity được dùng để làm video game, hoặc những nội dung có tính tương tác như thể hiện kiến trúc, hoạt hình 2D, 3D thời gian thực. Unity hao hao với Director, Blender game engine, Virtools hay Torque Game Builder trong khía cạnh dùng môi trường đồ họa tích hợp ở quá trình phát triển game là chính.

Unity là một trong những engine được giới làm game không chuyên cực kỳ ưa chuộng bởi khả năng tuyệt vời của nó là phát triển trò chơi đa nền. Trình biên tập có thể chạy trên Windows và Mac OS, và có thể xuất ra game cho Windows, Mac, Wii, iOS, Android. Game cũng có thể chơi trên trình duyệt web thông qua plugin Unity Web Player. Unity mới bổ sung khả năng xuất ra game trên widget cho Mac, và cả Xbox 360, PlayStation 3.



* + - 1. Sự đa nền tảng của unity

Chỉ với khoản tiền bỏ ra khá khiêm tốn (1.500 USD) là phiên bản pro đã nằm trong tay của chúng ta, dĩ nhiên tại Việt Nam số tiền này vẫn là quá lớn nhưng thật may là đã có phiên bản Unity Free. Tuy nhiên, nhiều tính năng quan trọng (Network) bị cắt giảm nhưng đó không phải là vấn đề quá lớn nếu muốn phát triển một tựa game tầm trung.

Vào năm 2009, Unity nằm trong top 5 game engine tốt nhất cho việc sản xuất game với chỉ sau 4 năm phát triển. Unity đứng thứ 4, xếp sau Unreal Engine 3, Gamebryo Engine (được VTC Studio mua về phát triển SQUAD) và Cry Engine 2. Lượng tài liệu hướng dẫn Unity rất phong phú. Hơn thế nữa nó còn có sẵn một cộng đồng cực lớn với diễn đàn riêng. Bất cứ điều gì không hiểu chúng ta đều có thể thoải mái hỏi và nhận được câu trả lời nhanh chóng, tận tâm.

Quá trình tạo địa hình cũng như truy xuất từ các phần mềm 3DSMax, Maya, Cinema4D... rất nhanh chóng. Sức mạnh và sự tiện lợi của Unity là vô cùng lớn.

Sức mạnh: Unity có thể tạo ra được nhiều loại game 2D, 3D đa dạng, dễ sử dụng với người làm game chưa chuyên nghiệp, chất lượng cao, chạy hầu hết trên các hệ điều hành.

Sự tiện lợi: nếu chúng ta là một người chuyên dùng 3Dmax, hay Maya hoặc phần mềm mã nguồn mở Blender thì quả là thật tuyệt, chúng ta sẽ có một lợi thế lớn khi viết game trên Unity này, bởi công việc tạo các mô hình 2D, 3D sẽ trở lên dễ dàng hơn rất nhiều, việc kết hợp giữa người lập trình và người thiết kế các mô hình sẽ nhanh và hiệu quả hơn. Trong Unity chúng ta có thể import trực tiếp các file mô hình đang thiết kế và sẽ thiết kế hoàn thiện tiếp nếu chưa xong trong khi đó công việc import chỉ diễn ra một lần. Không như việc phải dùng các công cụ khác để thực hiện viết game chúng ta sẽ phải xuất chúng ra một dạng nào đó và mỗi lần sửa lại phần mô hình chúng ta lại phải import lại, và như thế là quá mất thời gian trong việc tạo và chỉnh sửa các mô hình theo ý muốn. Ngoài ra Unity còn cho chúng ta trực tiếp tạo các mô hình nếu muốn. Việc đặt các thuộc tính vật lý trong Unity cũng cực kỳ dễ dàng và hỗ trợ sẵn nhiều chức năng

* + 1. Sơ Lược Lịch Sử Hình Thành Và Phát Triển Của Unity

**a.Hình thành**

Phần lõi của Unity ban đầu được viết bởi Joachim Ante vào năm 2001. Sau đó công ty được hình thành vào năm 2005 và bắt đầu với phiên bản 1.0. Đến năm 2007, Unity được nâng lên phiên bản 2.0. Unity bắt đầu hỗ trợ iPhone vào năm 2008. Vào tháng 6/2010, Unity chính thức hỗ trợ Android và cho ra đời phiên bản 3.0 có hỗ trợ Android vào tháng 9/2010 và bây giờ là phiên bản Unity 5. Có thể thấy tốc độ phát triển của Unity khá nhanh.

**b.Giải thưởng**

Unity đã đoạt được nhiều giải lớn với những giải chính sau:

Năm 2006, Unity đạt "Best Use of Mac OS X Graphics" tại Apple's WWDC. Đây là lần đầu tiên một công cụ phát triển game đạt được chất lượng do giải thưởng uy tín này đưa ra.

Năm 2009, Unity Technologies có tên trong "Top 5 công ty game của năm" do Gamasutra tổ chức.

Năm 2010, Unity đoạt giải Best Engine Finalist do Develop Magazine bình chọn, giải Technology Innovation Award của Wall Street Journal ở thể loại phần mềm.

**c.Khách hàng**

Unity được trên 250.000 người đăng ký sử dụng gồm Bigpoint, Cartoon Network, Coca-Cola, Disney, Electronic Arts, LEGO, Microsoft, NASA, Ubisoft, Warner Bros, các hãng phim lớn nhỏ, các chuyên gia độc lập, sinh viên và những người đam mê.

* + 1. Tính năng của Engine Unity

Môi trường phát triển được tích hợp với tính năng kế thừa, khả năng chỉnh sửa đồ họa, chức năng kiểm tra chi tiết, và đặc biệt tính năng xem trước game ngay trong lúc xây dựng (live game preview).

Triển khai được trên nhiều nền tảng:

Chương trình độc lập trên Windows và Mac OS.

Trên web, thông qua Unity Web Player plugin cho Internet Explorer, Firefox, Safari, Opera, Chrome, cho cả Windows và Mac OS.

Trên Mac OS Dashboard widget.

Cho Nintendo Wii (cần mua license thêm.)

Cho iPhone, iPad application (cần mua license thêm.)

Cho Google Android (cần mua license thêm.)

Cho Microsoft Xbox 360 (cần mua license thêm.)

Cho Sony PlayStation 3 (cần mua license thêm.)

Tài nguyên (model, âm thanh, hình ảnh, ...) được tải vào trong Unity và tự động cập nhật nếu tài nguyên có sự thay đổi. Unity hỗ trợ các kiểu định dạng từ 3DS Max, Maya, Blender, Cinema 4D, Photoshop và Cheetah3D.

Graphics engine sử dụng Direct3D (Windows), OpenGL (Mac, Windows), OpenGL ES (iPhone OS), và các API khác trên Wii.

Hỗ trợ bump mapping, reflection mapping, parallax mapping, Screen Space Ambient Occlusion v...v…

Unity Asset Server: Đây là một tính năng khá mới của Unity, theo đó Unity sẽ cung cấp một hệ thống quản lý theo dạng phiên bản cho tất cả asset và cả script. Đây là một kho chứa các tài nguyên cần thiết cho việc làm game. Khi import cũng như sửa chữa, trạng thái của asset ngay lập tức được cập nhật. Server chạy trên database opensource PostgreSQL và có thể truy cập trên cả Mac lẫn Windows, Linux. Asset Server đòi hỏi một khoản phí phụ trội là $499 cho mỗi bản copy Unity, và một license Unity Pro.

* 1. TỔNG QUAN VỀ GAME ENGINE
     1. Khái niệm

Một Game Engine (hay công cụ tạo Game / động cơ Game) là một phần mềm được viết cho mục đích thiết kế và phát triển video Game. Có rất nhiều loại Game Engine dùng để thiết kế Game cho các hệ máy như hệ Consoles hay máy tính cá nhân (PC). Chức năng cốt lõi của Game Engine phần lớn nằm trong công cụ dựng hình (kết xuất đồ họa) cho các hình ảnh 2 chiều (2D) hay 3 chiều (3D), công cụ vật lý (hay công cụ tính toán và phát hiện va chạm), âm thanh, mã nguồn, hình ảnh động (Animation), trí tuệ nhân tạo, phân luồng, tạo dòng dữ liệu xử lý, quản lý bộ nhớ, dựng ảnh đồ thị, và kết nối mạng. Quá trình phát triển Game tiết kiệm được rất nhiều thời gian và kinh phí vào việc tái sử dụng và tái thích ứng một Engine để tạo nhiều Game khác nhau.

* + 1. Lịch sử

Thuật ngữ "Game Engine" xuất hiện vào giữa những năm 90, đặc biệt là trong mối quan hệ giữa Game 3D. Như các thuơng hiệu nổi tiếng của id Sortware: Doom và Quake, thay vì phải làm việc từ đầu, các nhà phát triển khác (nếu được cấp phép) sẽ có quyền truy nhập vào phần lõi (mã nguồn) của Game và thiết kế các hình ảnh, nhân vật, vũ khí, và các màn chơi của riêng họ - gọi là Game Content (nội dung Game) hay "Game Assets" (tài sản Game). Các Game sau này, như Quake III Arena và sản phẩm năm 1998 của Epic Games: Unreal được thiết kế với cách tiếp cận mới này, Game Engine và nội dung Game được chia thành các phần riêng biệt để phát triển. Engine có thể tái sử dụng khiến việc phát triển Game tiếp theo nhanh hơn và dễ dàng hơn, một thuận lợi to lớn trong ngành công nghiệp cạnh tranh này. Game Engine hiện đại là một trong những ứng dụng được viết ra (bằng các ngôn ngữ lập trình) phức tạp nhất, thường xuyên phải có rất nhiều tinh chỉnh trong hệ thống để đảm bảo kiểm soát chính xác trải nghiệm người dùng. Sự phát triển liên tục của Game Engine đã tạo ra một sự phân chia mạnh mẽ giữa các công việc dựng hình, viết kịch bản, thiết kế hình ảnh, và thiết kế màn chơi. Hiện nay thông thường một đội ngũ phát triển Game điển hình phải có số lượng họa sĩ gấp vài lần số lượng lập trình viên.

* + 1. Mục đích

Game Engine cung cấp một bộ các công cụ phát triển trực quan và có thể tái sử dụng từng thành phần trong đó. Nói chung các bộ công cụ này cung cấp một môi trường phát triển tích hợp được đơn giản hóa. Phát triển ứng dụng nhanh (Rapid Application Development) cho Game theo cách lập trình hướng dữ liệu. Những Game Engine này đôi khi còn được gọi là các "phần mềm trung gian cho Game" (Game Middleware), như ý nghĩa của thuật ngữ, chúng cung cấp một nền tảng phần mềm linh hoạt và dễ dàng sử dụng lại với mọi chức năng cốt lõi cần thiết ngay trong nó để có thể phát triển một ứng dụng Game đồng thời giảm giá thành, độ phức tạp, và kịp thời hạn phát hành - tất cả các yếu tố quan trọng trong ngành công nghiệp Game đầy cạnh tranh. Giống như các phần mềm trung gian khác, Game Engine thường cung cấp một nền tảng trừu tượng hóa, cho phép một Game có thể chạy trên nhiều hệ máy bao gồm các hệ console hoặc máy tính cá nhân với một vài thay đổi (nếu cần) trong mã nguồn của Game đó. Thông thường, phần mềm trung gian cho Game được thiết kế với một nền tảng kiến trúc dựa trên các thành phần khác, cho phép các hệ thống khác nhau trong Engine có thể thay thế hoặc mở rộng với các phần mềm trung gian khác chuyên biệt hơn như là Havok cho hệ thống vật lý trong Game, Miles Sound System cho âm thanh, hay Bink cho các đoạn video. Một số Game Engine chỉ cung cấp khả năng dựng hình (kết xuất) 3D thời gian thực hay một khả năng riêng biệt nào khác thay vì rất nhiều chức năng trong phạm vi rộng mà Game yêu cầu. Loại Engine này thường được gọi là: "Graphics Engine", "Rendering Engine," hay "3D Engine" thay vì thuật ngữ bao quát hơn là "Game Engine”. Một vài ví dụ cho các Engine đồ họa là: RealmForge, Truevision3D, OGRE, Crystal Space, Genesis3D, Vision Engine, Irrlicht và JMonkey Engine.

* + 1. Ảo hóa phần cứng

Thông thường, Engine 3D hay hệ thống kết xuất đồ họa trong Game Engine thường được xây dựng trên một nền tảng giao diện lập trình ứng dụng đồ họa như Direct3D hay OpenGL cung cấp sẵn một hệ ảo hóa cho GPU hay card đồ họa. Thư viện mã nguồn cấp thấp (cấp cơ bản) như DirectX, SDL, và OpenAL (thư viện âm thanh mã nguồn mở) cũng thường được sử dụng trong các Game khi chúng cung cấp khả năng truy cập phần cứng độc lập.

* + 1. Xu hướng hiện tại

Với công nghệ tạo Game Engine càng phát triển và trở nên thân thiện hơn với người sử dụng, ứng dụng của nó càng được mở rộng, và giờ đây được sử dụng để tạo các Game mang mục đích khác với giải trí đơn thuần như: mô phỏng, huấn luyện ảo, y tế ảo, và mô phỏng các ứng dụng quân sự. Nhiều Game Engine đang được tạo ra bằng các ngôn ngữ lập trình cấp cao như Java và C# hay .NET (ví dụ: TorqueX, và Visual3D.NET) hay Python (Panda3D). Vì hầu hết các Game 3D hiện nay đều có giới hạn cho GPU (giới hạn bởi sức mạnh của card đồ họa), khả năng gây chậm máy của các ngôn ngữ lập trình cấp cao trở nên không đáng kể, trong khi việc tăng năng suất được cung cấp bởi các ngôn ngữ này lại có lợi cho các nhà phát triển Game Engine.

* + 1. Phần mềm trung gian

Một số hãng hiện nay chuyên tập trung vào phát triển các bộ phần mềm gọi là phần mềm trung gian (Middleware). Các nhà phát triển Middleware cố gắng phát triển một bộ phần mềm mạnh mẽ bao gồm hầu hết các yếu tố mà một nhà phát triển Game cần đến để xây dựng một Game. Phần lớn các Middleware cung cấp các cơ sở để phát triển game một cách dễ dàng, như hình ảnh, âm thanh, hệ thống vật lý và chức năng AI. Gamebryo và RenderWare là hai phần mềm trung gian được sử dụng khá rộng rãi. Một số Middleware chỉ làm một chức năng nhưng nó còn hiệu quả hơn cả toàn bộ Engine. Ví dụ, SpeedTree được sử dụng để dựng hình ảnh cây cối thêm thực tế trong Game nhập vai The Elder Scrolls IV: Oblivion. Bốn gói phần mềm trung gian được sử dụng rộng rãi nhất, thường được cung cấp dưới dạng một chức năng bổ trong một hệ thống Engine gồm có Bink của hang RAD Game Tools, hãng Firelight với FMOD, Havok, và GFx của Scaleform. RAD Game Tools phát triển Bink cho dựng hình video cơ bản, đi cùng với âm thanh Miles, và dựng hình 3D Granny . FMOD của Firelight là một thư viện và bộ công cụ phần mềm âm thanh mạnh mẽ giá rẻ. Havok cung cấp một hệ thống mô phỏng vật lý mạnh, cùng với các bộ phần mềm về hình ảnh động và mô phỏng hành vi. Scaleform đem lại GFx dùng cho các biểu diễn cao cấp trên nền giao diện Flash, cùng với các giải pháp trình chiếu video chất lượng cao, và một Input Method Editor (IME) phần bổ sung nhằm hỗ trợ trò chuyện trong Game bằng ngôn ngữ Châu Á.

* + 1. Phân loại Game Engine theo mức độ chuyên biệt

**a.****Roll-your-own Game Engine**

Roll-your-own Engine có thể được xem như những Engine ở mức thấp nhất. Ngày nay rất nhiều công ty tự tạo ra Engine cho riêng họ với những thứ có thể. Họ tận dụng những gói ứng dụng phổ biến, các API như XNA, DirectX, OpenGL, các API và SDL của Windows và Linux. Ngoài ra, họ có thể dùng những thư viện từ nhiều nguồn khác nhau, được mua hoặc Open-Source. Những thư viện đó có thể bao gồm cả những thư viện vật lý như Havok và ODE, những thư viện Scene-Graph như OpenSceneGraph và các thư viện GUI như AntTweakBar. Thực tế, những hệ thống “tự sản xuất” đó cho phép những lập trình viên dễ tương tác hơn rất nhiều vì họ biết cần gì và chọn những thành phần phù hợp từ đó tạo nên chính xác thứ cần thiết. Nhưng cũng vì vậy mà tốn rất nhiều thời gian. Thêm vào đó, những lập trình viên thường gặp rất nhiều lỗi khi các thư viện từ nhiều nguồn không tương thích với nhau. Điều này đã góp phần làm cho những Engine dạng này kém hiệu quả và không được ưa chuộng ngay cả với Engine được thiết kế một cách chuyên nghiệp.

**b. Mostly-ready Game Engine**

Mostly-ready Game Engine, những Engine nằm ở tầng trung. Các nhà thiết kế Game nghiệp dư rất thích những Engine dạng này. Chúng được thiết kế rất đầy đủ với mọi tính năng cần thiết, vẽ, tương tác, GUI, vật lý… ngoài ra chúng còn kèm theo rất nhiều tool mạnh. Những Engine tầng này khá phong phú từ ORGE và Genesis3D (open-source), rẻ tiền như Torge hoặc đắc tiền như Unreal, id Tech và Gamebryo. Về mức độ, những Engine kể trên đều cần lập trình thêm để gắn kết mọi thứ với nhau nhằm tạo nên một Game hoàn chỉnh. Hầu như mọi Mostly-ready Game Engine đều có phần hạn chế một ít so với roll-your-own Engine. Vì mặc dù nó là sản phầm của rất nhiều người tham gia phát triển trải qua thời gian dài nhằm mang đến cho khách hàng những “tiện nghi” tốt nhất nhưng không phải khách hàng nào cũng giống nhau và phát triển những Game tương tự nhau.

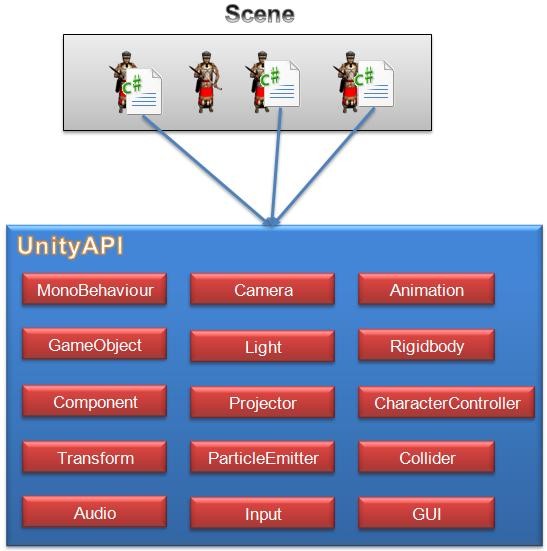
**c. Point-and-click Game Engine**

Point-and-click Engine được xếp ở tầng cao nhất. Những Engine này ngày càng trở nên phổ biến. Nó có tất cả mọi tool với đầy đủ chức năng cho phép chúng ta chỉ việc chọn, click và chọn để tạo ra Game. Những Engine như vậy, kể cả GameMaker, Torque Game Builder, và Unity3D được thiết kế để càng thân thiện với người dùng càng tốt. Nó cũng yêu cầu ít lập trình hơn. Điều này không có nghĩa việc lập trình không cần, nhưng nếu so với 2 loại Engine kể trên trên thì loại Engine này cần ít nhất. Nhược điểm rất lớn của những Engine point-and-click này đến từ việc những nhà phát triển bị hạn chế rất nhiều, vì với loại Engines này, chúng chỉ có thể cho ra một loại giao diện như tương tự nhau hoặc khác nhau một ít. Nhưng điều này không có nghĩa chúng vô dụng. Vì khi đối mặt với những hạn chế đó, những sáng tạo của con người sẽ càng được kích thích cao. Ngoài ra, Engine này cho phép chúng ta có thể làm việc nhanh hơn, làm ra Game nhanh hơn mà không phải làm quá nhiều.

* 1. TỔNG QUAN KIẾN TRÚC GAME ENGINE
     1. Kiến trúc tổng quan

Engine Unity hỗ trợ cho chúng ta UnityAPI để viết script game. UnityAPI là API lập trình game trong Unity rất mạnh. UnityAPI chứa các đối tượng và phương thức hỗ trợ hầu hết các đối tượng và các loại thành phần trong Unity.

Trong một scene thường có nhiều đối tượng game. Mỗi đối tượng này có thể có hoặc không có đoạn script nào gắn lên đó. Nếu muốn gắn script vào đối tượng, ta bắt buộc phải kế thừa class đó từ lớp MonoBehaviourcủa UnityAPI và tên class phải trùng với tên file script. Mỗi script khi gắn lên đối tượng game đều được đối tượng game xem như một thành phần bên trong và được cấp phát vùng nhớ khi chạy game.



* + - 1. Tổng quan kiến trúc Unity

Bên trong 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, ..

**CharacterController**: giúp điều khiển nhân vật di chuyển theo độ cao địa hình.

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

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

1. UNITY ENGINE
   1. GIỚI THIỆU CHUNG



* + - 1. Logo của Unity Engine

Nhà phát triển: Unity Technologies

Phiên bản mới nhất : Unity 2019.1.1f1

Được viết bởi ngôn ngữ : C++, C#

Phát triển Game cho các hệ điều hành : Windows, Mac OS X (tạo và đóng gói), Wii, iPhone/iPad, Xbox 360, Android, PS3 (chỉ đóng gói ; cần giấy phép bổ sung cho từng nền tảng)

Giấy phép: Độc quyền

Website: [www.unity3d.com](http://www.unity3d.com)

Unity là một 3D Game Engine, là một công cụ thiết kế Game dành cho PC, Mac và nhiều hệ máy di động khác.

Unity được sự hỗ trợ của Just-In-Time Compilation (JIT), sử dụng thư viện mã nguồn mở C++ Mono. Bằng việc sử dụng JIT, những Engine như Unity có thể tận dụng lợi thế của tốc độ biên dịch. Những đoạn code do chúng ta viết sẽ được Unity biên dịch ra Mono trước khi nó được thực thi. Điều này rất quan trọng cho Game để thực thi code vào những thời điểm cần thiết trong suốt thời gian chạy (Runtime).

Ngoài thư viện Mono, Unity cũng tận dụng chức năng của những thư viện phần mền khác vào chức năng của nó, như Engine mô phỏng vật lý PhysicX của Nvidia, OpenGL và DirectX cho kết xuất hình ảnh 3D, OpenAL cho âm thanh. Tất cả các thư viện này được xây dựng thành những tính năng tự động hoặc công cụ trực quan vào Unity, vì thế chúng ta không cần phải lo lắng về việc phải học thế nào để sử dụng chúng một cách riêng lẽ.

Unity có một cộng đồng người dùng rất mạnh (rất lớn) luôn chia sẻ những Plugins, công cụ của họ dưới hình thức gói phần mềm bổ sung.

Có thể sản xuất các trò chơi theo tiêu chuẩn chuyên nghiệp, xuất bản 3D cho cả Mac và PC cũng như sở hữu riêng một Web Player của riêng mình, Unity là một trong những Game Engine có tốc độ phát triển nhanh nhất. Unity cũng có phiên bản phát triển Game cho hệ máy Wii của Nintendo và Iphone của Apple, có nghĩa là một khi chúng ta làm chủ được những vấn đề cơ bản, không chỉ phát triển Game cho máy tính cá nhân mà chúng ta còn có thể phát triển Game cho các hệ máy console và thiết bị di động.

* 1. 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 lý)

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

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.

* + 1. 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#, UnityScrip và Boo.

Boo đã bị loại cùng với việc phát triển Unity 5.

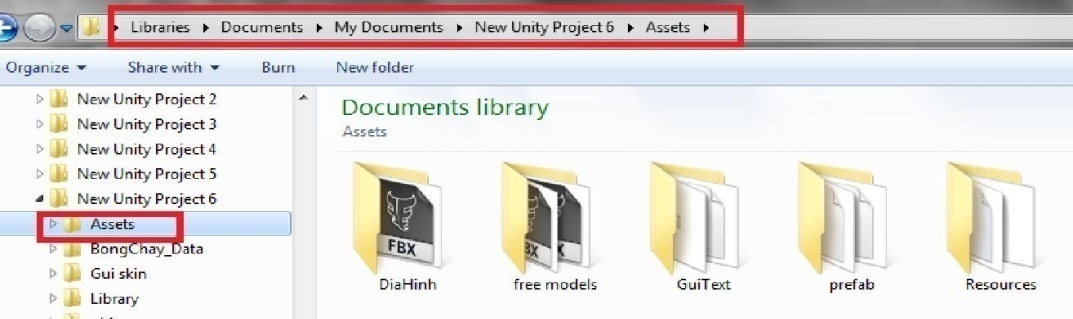
UnityScript bị loại vào tháng 8 năm 2017 sau khi phát hành Unity 2017.1 . UnityScript là một ngôn ngữ lập trình độc quyền có cú pháp tương tự JavaScript.

* + 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, …

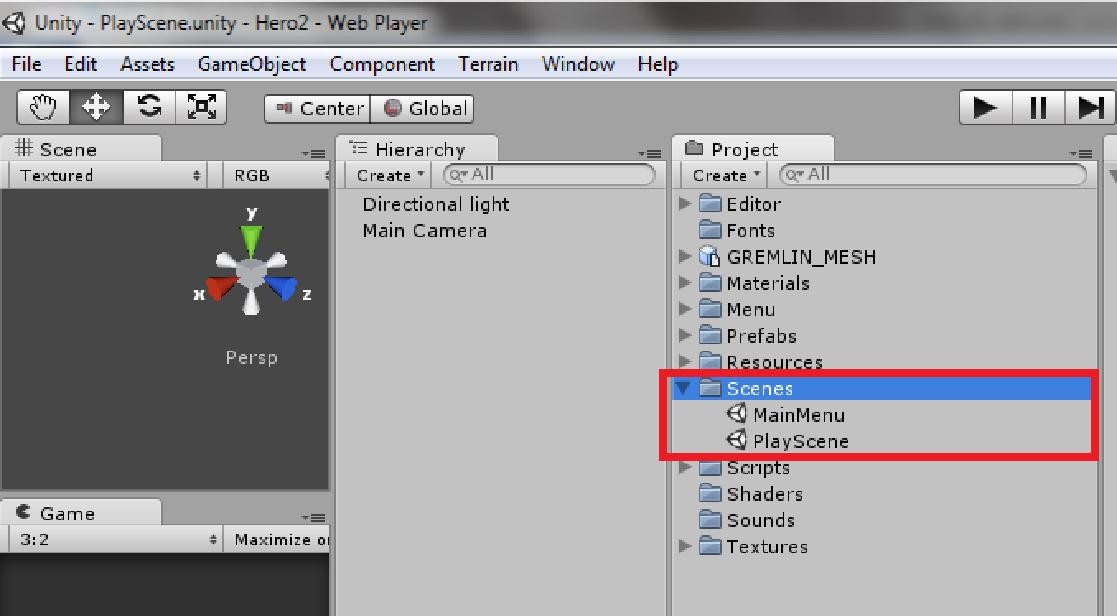
Một số tính tăng cung cấp bởi Networking như: State Synchronization, Realtime Networking, Remote Procedure Calls, Backend Connectivity, Web Browser Integration, Web Connectivity.

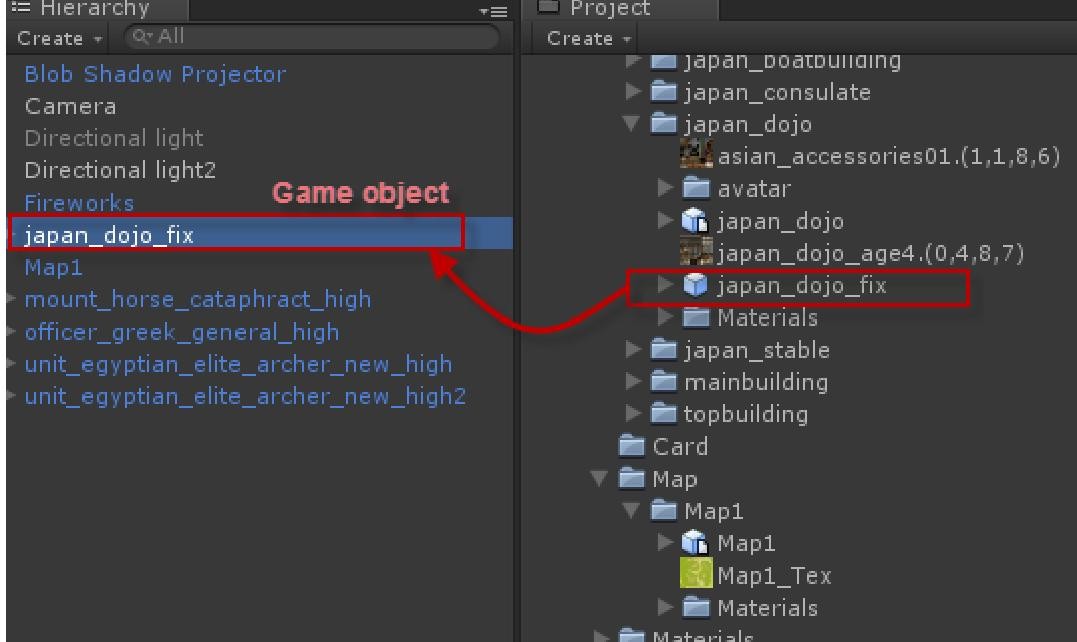
* 1. CÁC THÀNH PHẦN TRONG UNITY
     1. Assets

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”.

* + - 1. 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.

* + - 1. Các Scene của Unity
    1. Game Object

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. Kéo tài nguyên vào Scene để sử dụng
    1. Components

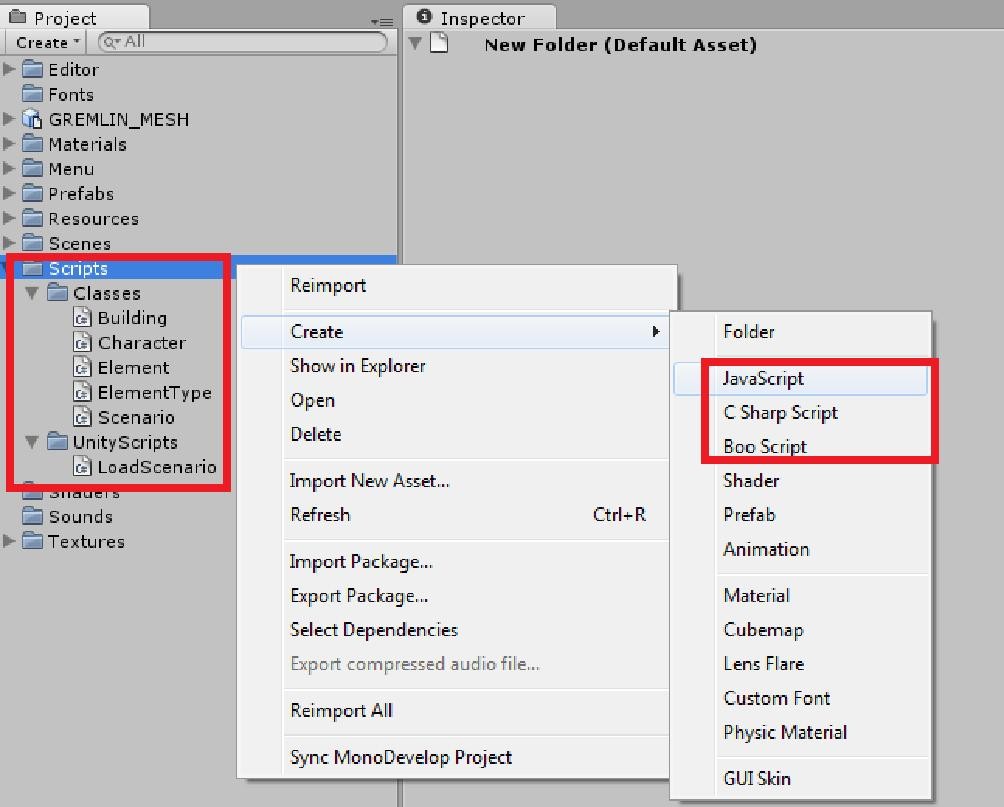
Components có nhiều hình thức khác nhau. Chúng có thể xác định hành vi, cách xuất hiện,… hay ảnh hưởng đến các khía cạnh khác trong chức năng của Game Object trong trò chơi. Bằng cách “gắn” chúng vào trong Game Object, chúng ta ngay lập tức có thể áp dụng tác động của chúng lên đối tượng. Những Components 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. Các component trong đối tượng Camera
    1. Scripts

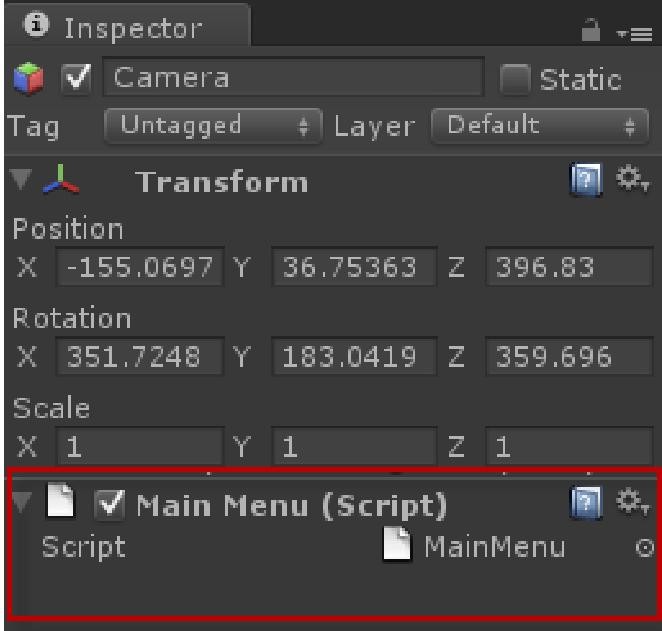
Được Unity xem như một Components, Script là một thành phần thiết yếu trong quá trình phát triển trò chơi và đáng được đề cập đến như một khái niệm “chìa khóa”. Unity cung cấp cho chúng ta khả năng viết Script bằng cả 3 loại ngôn ngữ là: JavaScript, C# và Boo (một dẫn xuất của ngôn ngữ Python).

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.

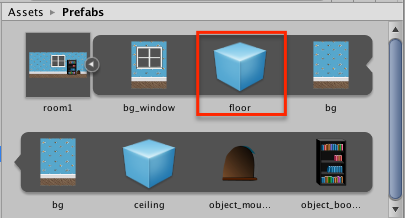
Mono developer là một IDE khá tốt để lập trình khi cung cấp nhiều chức năng tương tự như Visual studio. Mã nguồn viết trên Mono Developer sẽ được cập nhật và lưu trữ trong dự án Unity.

* + - 1. Cách tạo file script mới

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

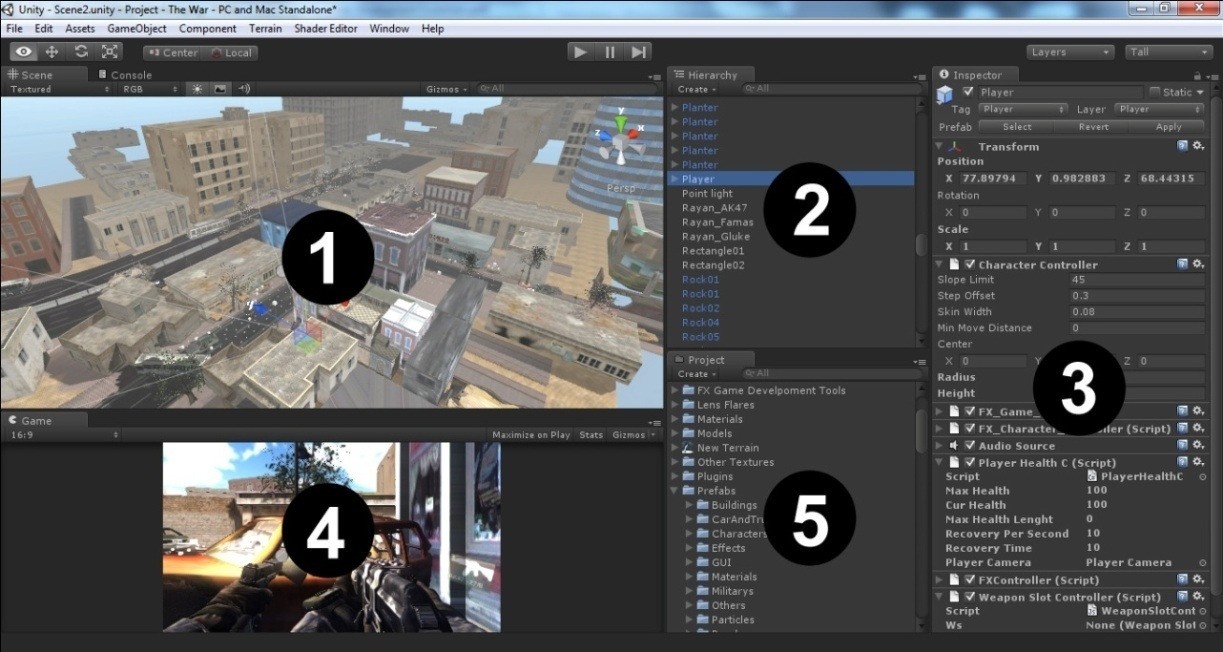
* + - 1. Một file script đang gắn vào một đối tượng
    1. Prefabs

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.



* + - 1. Một số đối tượng trong Prefabs
  1. GIAO DIỆN CỦA UNITY
     1. Giao diện

Giao diện của Unity có khả năng tùy chỉnh bố trí tương tự như nhiều môi trường làm việc khác. Dưới đây là một kiểu bố trí điển hình trong Unity:



* + - 1. Giao diện của Unity

Chú thích:

Scene (1): Nơi mà trò chơi sẽ được xây dựng.

Hierarchy (2): Danh sách các Game Object trong scene.

Inspector (3): Những thiết lập, thành phần, thuộc tính của đối tượng (hoặc Asset) đang được chọn.

Game (4): Cửa sổ xem trước, nó chỉ hoạt động trong chế độ “Play” (Preview – xem trước).

Project (5): Danh sách các Assets của dự án, được ví như thư viện của dự án.

* + 1. Cửa sổ Scene và Hierarchy

Cửa sổ scene là nơi mà chúng ta sẽ xây dựng các thực thể, đối tượng của dự án vào đó. Cửa sổ cung cấp góc nhìn phối cảnh (Perspective (góc nhìn 3D), chúng ta có thể chuyển qua các góc nhìn khác như từ trên xuống hoặc từ dưới lên (Top Down), từ trái sang phải hoặc phải sang trái (Side On), từ trước ra sau hoặc sau đến trước (Front On). Cửa sổ này sẽ kết hình xuất đầy đủ những hình ảnh trong thế giới của trò chơi mà chúng ta tạo ra dưới dạng một vùng biên tập mà chúng ta có thể biên tập, chỉnh sửa trực tiếp thế giới đó.

Khi kéo thả Assest vào cửa sổ Scene, Assets sẽ trở thành Game Object. Cửa sổ Scene được ràng buộc cùng với cửa sổ Hierarchy, cửa sổ Hierarchy liệt kệ danh sách các Game Object có trong Scene và được sắp xếp theo thứ tự chữ cái từ A-Z.



* + - 1. Các nút chức năng cho của sổ Scene

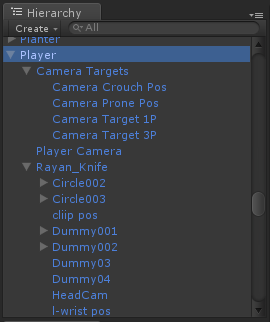
Cửa sổ Scene còn đi kèm với 4 bốn nút chức năng hữu ích được hiển thị dưới dạng hình ảnh như trên. Chúng có thể được lựa chọn thông qua các phím tắt Q, W, E và R. Những nút này có các chức năng như sau:

Công cụ bàn tay (Q): Công cụ này cho phép chúng ta di chuyển đến một khu vực nào đó trong Scene bằng thao tác kéo thả thuộc trái.

Công cụ di chuyển (W): Công cụ này cho phép chúng ta chọn một đối tượng trong cảnh và thực hiện thao tác di chuyển, thay đổi vị trí của đối tượng đó. Khi chọn, tại vị trí của đối tượng sẽ hiển thị các trục và mặt phẳng gắn liền với đối tượng cho phép chúng ta di chuyển đối tượng trượt theo các trục, mặt phẳng hoặc di chuyển một cách tùy ý.

Công cụ xoay (E): Công cụ này có đặc điểm và cách sử dụng giống với công cụ di chuyển, tuy nhiên thay vì để di chuyển vị trí của đối tượng thì công cụ này giúp chúng ta xoay đối tượng xoay quanh trục hay tâm của đối tượng.

Công cụ điều chỉnh tỉ lệ (R): Cũng tương tự như công cụ di chuyển và xoay, công cụ này cho phép chúng ta tùy chỉnh kích thước, tỉ lệ của đối tượng một cách tùy ý.

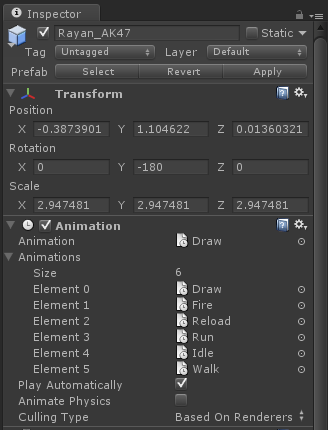


* + - 1. Cửa sổ Hierarchy.
    1. Cửa sổ Inspector

Cửa sổ Inspector có thể được xem như một công cụ cho phép chúng ta tùy chỉnh các thiết đặt, các thành phần của Game Object hoặc Assets đang được chọn.

Cửa sổ này sẽ hiển thị đầy đủ các Components của đối tượng mà chúng ta chọn. Nó cho phép chúng ta điều chỉnh các biến của Components dưới các hình thức như: Textbox, Slider, Button, Drop-dowm Menu…

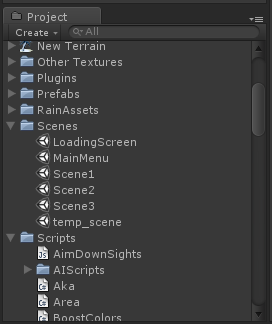
Ngoài việc hiển thị các Component của đối tượng được chọn, cửa sổ Inspector còn hiển thị các thiết đặt chung của hệ thống hay của trò chơi khi ta chọn chúng từ menu Edit.



* + - 1. Cửa sổ Inspector

Trong hình trên, chúng ta thấy cửa sổ Inspector đang hiển thị một vài thuộc tính, Components của một đối tượng đang được chọn. Trong đó, bao gồm 2 Components là Transform và Animation. Cửa sổ Inspector sẽ cho phép chúng ta thay đổi các thiết đặt trên. Các Components này còn có thể được tạm thời vô hiệu hóa vào bất kỳ lúc nào chúng ta muốn bằng cách bỏ chọn Checkbox ở góc trên bên trái của mỗi Component, việc này sẽ rất hữu ích cho chúng ta khi muốn kiểm tra hay thử nghiệm các Components này. Ngoài ra, cửa Inspector còn cho phép chúng ta vô hiệu hóa toàn bộ một đối tượng đang được chọn bằng cách bỏ chọn Checkbox ở trên cùng góc trái của cửa sổ Inspector.

* + 1. Cửa sổ Project



* + - 1. Cửa sổ Project

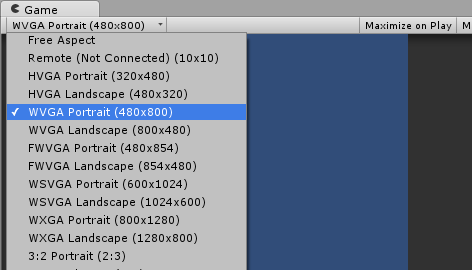
Cửa sổ Project là cửa sổ cho phép chúng ta nhìn thấy trực tiếp nội dung của thư mục Assets của dự án. Mỗi dự án Unity đều được chứa trong một thư mục cha. Trong đó có 3 thư mục con là Assets, Library và Temp (chỉ có khi Unity đang chạy). Đặt tất cả các Assets vào thư mục Assets có nghĩa là ngay lập tức chúng ta sẽ thấy chúng xuất hiện trong cửa sổ Project. Ngoài ra, khi thay đổi vị trí của Assets trong thư mục Assets hay lưu tập tin lại từ một chương trình ứng dụng thứ 3 nào khác (ví dụ như Photoshop), sẽ làm cho Unity nhập lại (Re-Import) Assets, phản ánh sự thay đổi này ngay lâp tức trong cửa sổ Project và Scene có sử dụng Assets vừa được thay đổi.

Cửa sổ Project được tích hợp nút Create, nút này cho phép chúng ta tạo mới bất kì một Assets mới nào, ví dụ như Script, Prefabs, Materials,…

* + 1. Cửa sổ Game

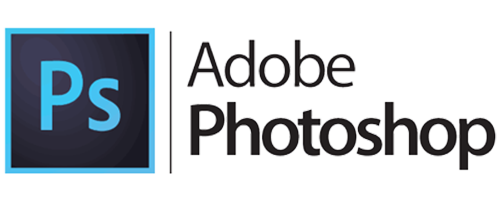
Cửa sổ Game sẽ được gọi khi chúng ta nhấn vào nút Play (là một hành động thực hiện test trò chơi). Cửa sổ này cho phép chúng ta tùy chọn về thiết đặt tỉ lệ màn hình, nó phản ánh phạm vi trong Scene mà người chơi có thể thấy được với mỗi tỉ lệ màn hình tương ứng, ví dụ như với mỗi tỉ lệ màn hình 4:3, 16:9 thì người chơi sẽ có một phạm vi nhìn thấy khác nhau.

Sau khi nhấn vào nút Play, chúng ta sẽ ở chế độ Testing, lúc này mọi thay đổi về các thuộc tính, Components, … của đối tượng sẽ chỉ là tạm thời. Tức là chúng sẽ trở về như ban đầu (trước khi nhấn nút Play) sau khi kết thúc chế độ Testing.



* + - 1. Các kích thước màn hình của Game

1. ADOBE PHOTOSHOP CHO UNITY ENGINE
   1. GIỚI THIỆU CHUNG
      1. Tổng quan



* + - 1. Logo của Adobe Photoshop

Adobe Photoshop (thường được gọi là Photoshop) là một [phần mềm chỉnh sửa đồ họa](https://vi.wikipedia.org/w/index.php?title=Ph%E1%BA%A7n_m%E1%BB%81m_%C4%91%E1%BB%93_h%E1%BB%8Da&action=edit&redlink=1) được phát triển và phát hành bởi hãng [Adobe Systems](https://vi.wikipedia.org/wiki/Adobe_Systems) ra đời vào năm 1988 trên hệ máy [Macintosh](https://vi.wikipedia.org/wiki/Macintosh). Photoshop được đánh giá là phần mềm dẫn đầu thị trường về sửa ảnh bitmap và được coi là chuẩn cho các ngành liên quan tới chỉnh sửa ảnh. Từ phiên bản Photoshop 7.0 ra đời năm 2002, Photoshop đã làm lên một cuộc cách mạng về ảnh [bitmap](https://vi.wikipedia.org/wiki/Bitmap). Phiên bản mới nhất hiện nay là Adobe Photoshop CC.

Ngoài khả năng chính là chỉnh sửa ảnh cho các ấn phẩm, Photoshop còn được sử dụng trong các hoạt động như thiết kế trang web, vẽ các loại tranh (matte painting và nhiều thể loại khác), vẽ texture cho các chương trình 3D... gần như là mọi hoạt động liên quan đến ảnh bitmap.(Nguồn wikipedia.org/wiki/Adobe\_Photoshop).

* + 1. Lịch sử

Ngày 19 tháng 2 năm 1990, phần mềm Adobe Photoshop 1.0 dùng cho máy Mac, có dung lượng 728 KB, được phát hành ở dạng đóng gói, gồm một đĩa mềm và tài liệu hướng dẫn.

Từ năm 1992, khi vai trò chuyên nghiệp của Photoshop đã được xác lập, các phần mềm khác có chức năng xử lý ảnh tương tự Photoshop (Photo-Paint, Paint Shop Pro tại Mỹ, Nuances tại Pháp,...) mới xuất hiện.

Đến năm 1995, [tập đoàn Adobe](https://vi.wikipedia.org/wiki/Adobe_Systems) mua bản quyền Photoshop từ anh em Knoll.

Kể từ năm 2003, khi [Adobe](https://vi.wikipedia.org/wiki/Adobe_Systems) bắt đầu gói tất cả công cụ Web và in ấn (bao gồm Photoshop) vào một gói ứng dụng có tên là [Creative Suite](https://vi.wikipedia.org/w/index.php?title=Creative_Suite&action=edit&redlink=1), hãng này thường xuyên nâng cấp bộ công cụ này dựa trên ý kiến đóng góp của cộng đồng, tập trung chủ yếu vào dịch vụ trực tuyến và phân tích [web](https://vi.wikipedia.org/wiki/World_Wide_Web).

Ngày 7 tháng 5 năm 2013 Adobe giới thiệu phiên bản Photoshop CC, chương trình của Creative Suite do đó đổi thành Adobe Creative Cloud. Bằng cách chuyển sang giải pháp [Cloud](https://vi.wikipedia.org/w/index.php?title=Cloud-Computing&action=edit&redlink=1) người dùng có thể sử dụng các thiết lập cá nhân của mình ở bất cứ nơi nào Photoshop CC được cài đặt. Các tính năng mới khác bao gồm bộ lọc để giảm hình bị rung, hình chữ nhật có thể chỉnh sửa tròn góc và xem trước trực tiếp. Kể từ phiên bản CC Photoshop chỉ có thể mướn, chứ không mua đứt nữa.

Trong tháng 6 năm 2014 Photoshop CC 2014 được xuất bản; một năm sau, vào ngày 15 tháng 6 năm 2015 Photoshop CC 2015 ra mắt người dùng.

Ngày 18 tháng 10 năm 2017, hãng ra mắt phần mềm CC 2018. .(Nguồn wikipedia.org/wiki/Adobe\_Photoshop).

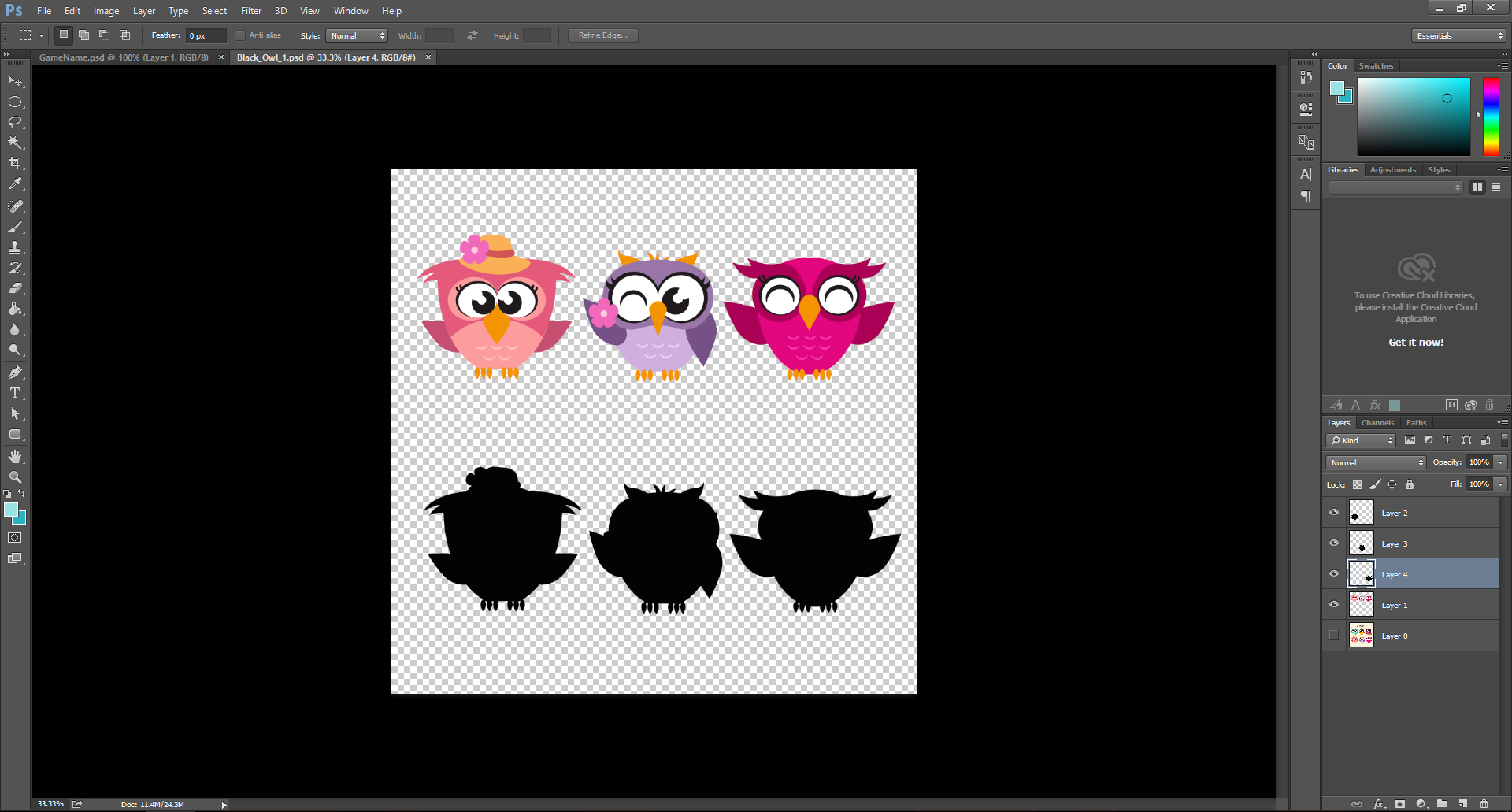


* + - 1. Giao diện của Photoshop CC 2015
  1. PHOTOSHOP TRONG UNITY

Là công cụ chính được sử dụng trong làm game 2D trên Unity, Photoshop có thể làm được mọi thứ ở game 2D về mặt hình ảnh như character, background, enemy, UI, …

* + 1. Sprite

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.



* + - 1. Minh họa một sprite
    1. Animation

Animation là một yếu tố không thể thiếu được trong bất kỳ game nào. Animation giúp lập trình viên diễn tả hành động của các nhân vật một cách chân thực và sống động hơn.

Có 2 cách tạo animation chính:

**a.Sprite sheet**: là một tập hợp nhiều Sprite vào một tập tin duy nhất.

Tác dụng của Sprite Sheet

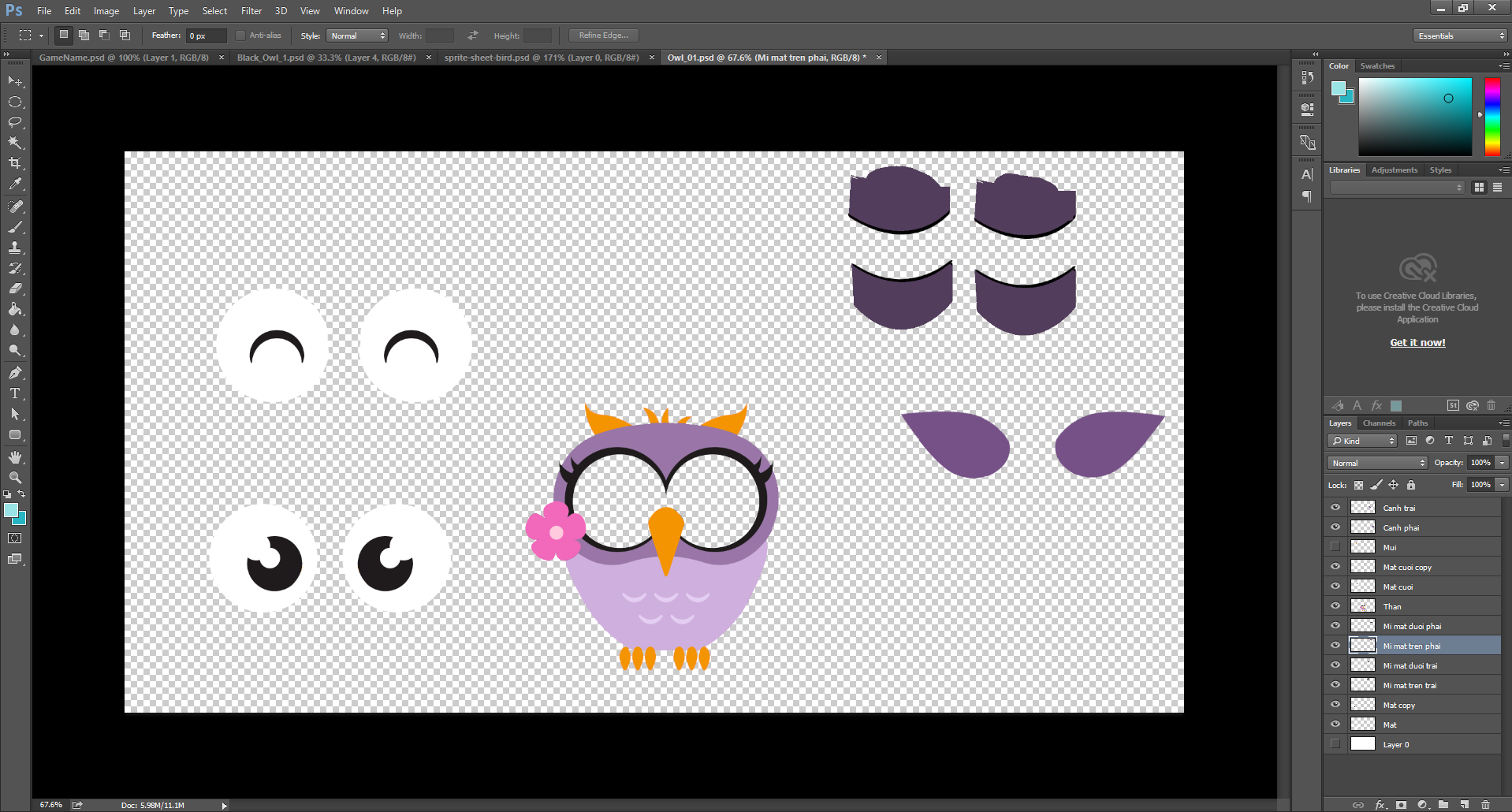
Giúp dễ dàng quản lý file hình ảnh của bạn hơn trong project.

Làm giảm kích thước của tập tin tổng thể so với việc có tập tin riêng cho mỗi Sprite, điều này giúp bạn làm giảm đáng kể việc sử dụng bộ nhớ, kích thước tập tin và thời gian tải, vậy nên hiệu suất của game cũng được tăng lên đáng kể.



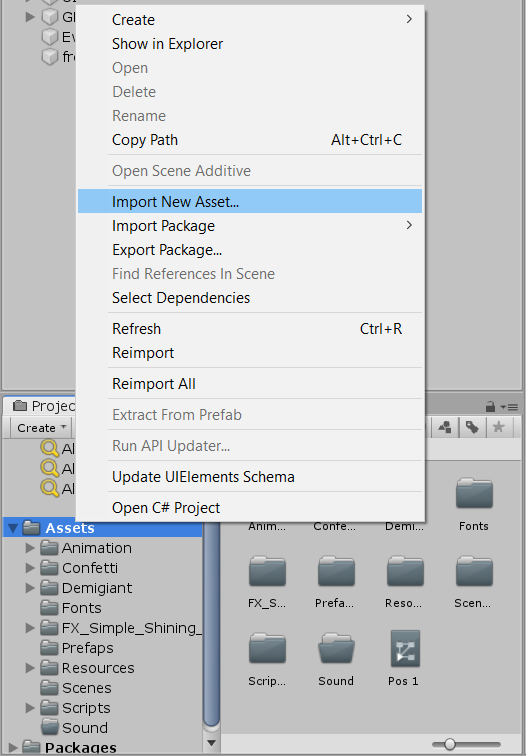
* + - 1. Ảnh minh họa một sprite sheet

**b.Skeletal 2D**: Ý tưởng là thay vì save mỗi frame hình cho animation, thay vào đó chỉ cần save từng bộ phận cho cơ thể một cách riêng lẻ như thế này:



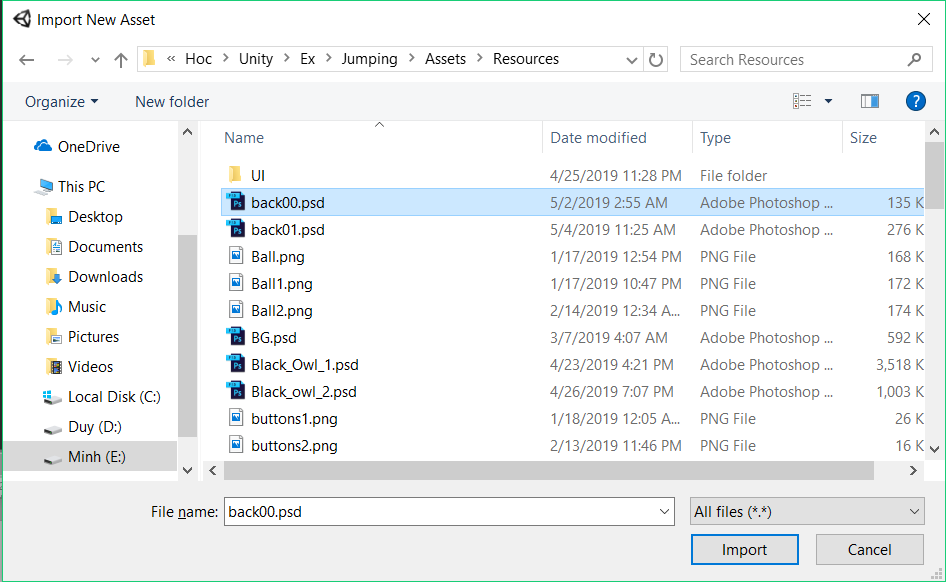
* + - 1. Skeletal 2D
    1. Import file Photoshop vào Unity

Từ phiên bản Unity 5 , Unity đã cho phép inport trực tiếp một file psd (photoshop file) vào Unity.



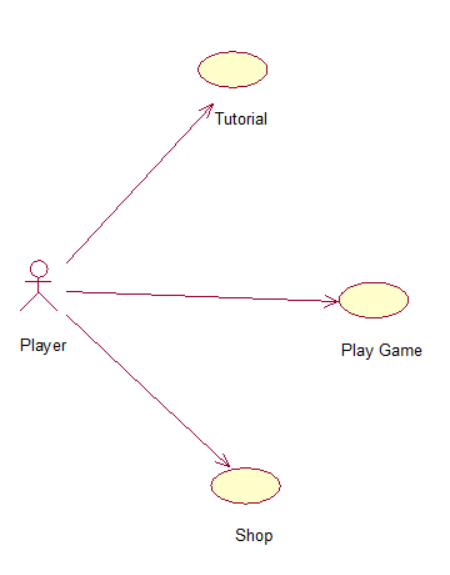
* + - 1. Import một file photoshop

Nhấn chuột phải vào một thư mục trong cửa sổ Project , Import New Asset sau đó có một cửa sổ hiện lên tìm đến thư mục chứa file psd, nhấn Import.



* + - 1. Import file photoshop

1. ỨNG DỤNG GAME JUMP UP PHÁT TRIỂN TRÊN UNITY
   1. CÁC SƠ ĐỒ TRONG GAME
      1. Sơ đồ Use-case người chơi



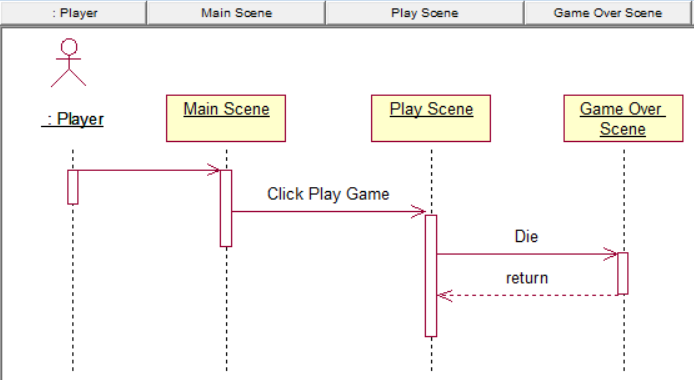
* + - 1. Biểu đồ use-case người chơi

Đặc tả use-case:

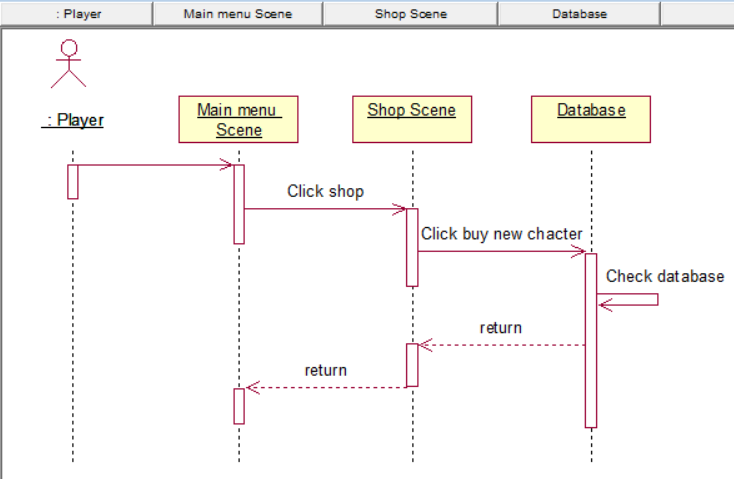
Tác nhân: người chơi.

Người chơi có 3 chức năng chính: xem hướng dẫn, chơi game và mua nhân vật mới ở shop.

* + 1. Sơ đồ tuần tự chơi game của người chơi



* + - 1. Sơ đồ tuần tự chơi game của người chơi
    1. Sơ đồ tuần tự mua nhân vật mới



* + - 1. Sơ đồ tuần tự chơi mua nhân vật mới
  1. GIỚI THIỆU GAME JUMP UP
     1. Thể loại

Thể loại: Casual, Endless

Game casual là thể loại game phổ thông, dễ chơi, game thường có độ dài ngắn. Game casual thường được chơi trực tuyến trên các [trình duyệt web](https://vi.wikipedia.org/wiki/Tr%C3%ACnh_duy%E1%BB%87t_web) bằng [máy tính](https://vi.wikipedia.org/wiki/M%C3%A1y_t%C3%ADnh) hay [điện thoại di động](https://vi.wikipedia.org/wiki/%C4%90i%E1%BB%87n_tho%E1%BA%A1i_di_%C4%91%E1%BB%99ng). Thời gian gần đây trò chơi loại này cũng được phổ biến trên các máy chơi game. Thể loại game này thường phù hợp với tất cả mọi người từ trẻ em đến người lớn.

Thể loại này đang phát triển với tốc độ khá nhanh và khá quan trọng đối với nhiều nhà sản xuất game.

Có thể nói Casual là thể loại game có số lượng áp đảo các thể loại kia trên thị trường, đặc biệt là thị trường game dành cho mobile. Dùng để chỉ những tựa game có các màn chơi ngắn, với cách chơi đơn giản, thiên về giải đố, sắp xếp, logic hay có khi chỉ là chạy, nhảy, né chướng ngại vật. Phù hợp cho thư giãn, luyện nhanh tay nhanh mắt nhanh phản xạ.

* + 1. Nền tảng

Nền tảng : Android

Hãng nghiên cứu Gartner công bố kết quả của một cuộc điều tra về số lượng người dùng các hệ điều hành di động. Theo đó, có khoảng 1.5 tỷ chiếc smartphone được bán ra vào năm 2017. Trong đó, **số lượng người dùng Android** gấp tới 6 lần iOS, chiếm 85% thị phần toàn ngành.

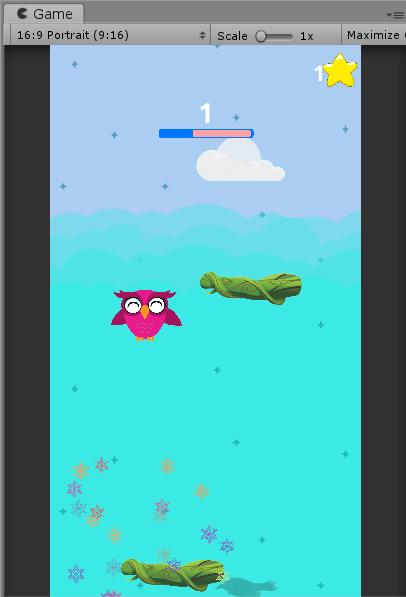
Sự phổ biến của Android bởi sức len lỏi của hệ điều hành này ở mọi thiết bị di động, từ điện thoại cao cấp đến những sản phẩm có giá phổ thông.

* + - * 1. Thị phần các thiết bị di động năm 2017 theo doanh số do Gartner công bố.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Operating System | 2017 Units | 2017 Marker Share (%) | 2016 Units | 2016 Marker Share (%) |
| Android | 1.320.118 | 85,9 | 1.268.562 | 84,8 |
| IOS | 214.924 | 14,0 | 216.064 | 14,4 |
| Other OS | 1.943 | 0,1 | 11.332 | 0,8 |
| Total | 1.536.535 | 100 | 1.495.959 | 100 |

* 1. CÁC QUY LUẬT CHƠI CHÍNH
     1. Di chuyển

Người chơi nhấn vào màn hình, nhân vật sẽ nhảy lên một mức nhất định, nếu rơi trúng vào các platform đang di chuyển thì được phép nhấn màn hình để nhảy lên tiếp.



* + - 1. Di chuyển trong game
    1. Điểm số

Khi nhân vật nhảy lên được các platform đang di chuyển sẽ có một thanh màu trắng giảm dần. Nếu người chơi nhảy lên được các platform tiếp theo thì thanh màu trắng sẽ được hồi lại và một số thể hiện số lần nhảy lên liên tiếp của người chơi.

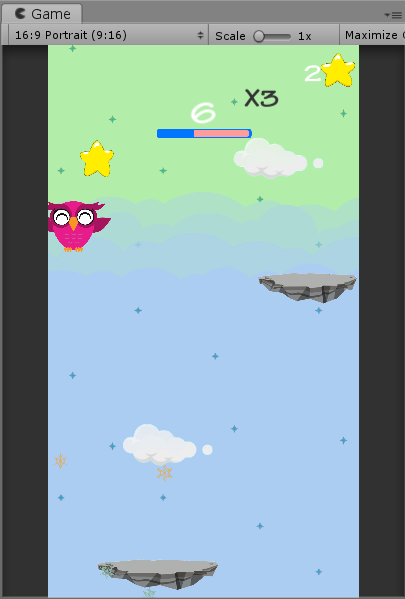


* + - 1. Điểm số

Điểm số được tính bằng số lần nhảy lên liên tiếp. Vd: x3 và điểm bằng 6 nếu nhảy lên được platform tiếp theo thì điểm là 9 và x4. Nếu thanh màu trắng chạy hết thì x sẽ trả về 0.

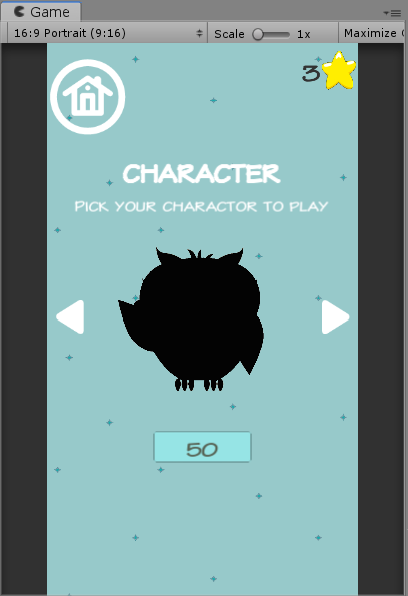
* + 1. Item

Khi chơi game sẽ ngẫu nhiên 20% xuất hiện ngôi sao.Ăn ngôi sao bằng cách nhảy lên chạm vào ngôi sao.



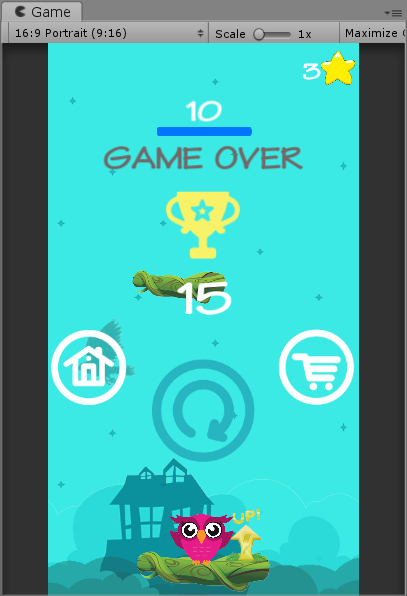
* + - 1. Item trong game

Ngôi sao được sử dụng để mua những nhân vật khác.



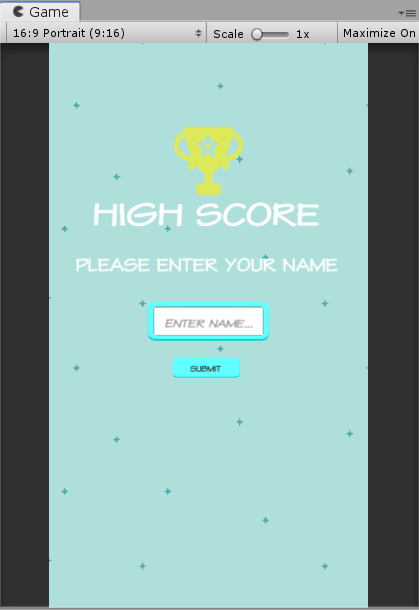
* + - 1. Giao diện cửa hàng
    1. Game Over

Người chơi thua khi nhân vật rơi xuống mà không chạm được vào platform.



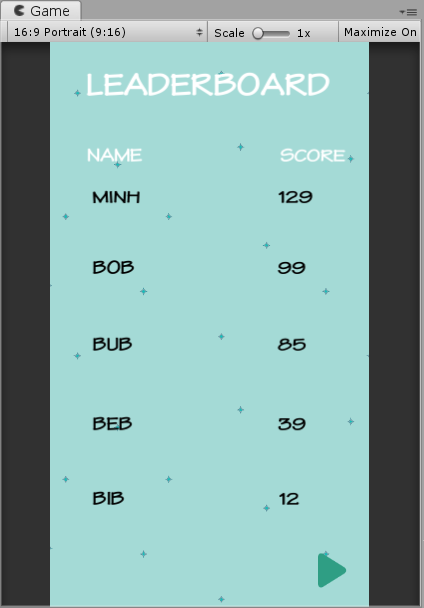
* + - 1. Giao diện game over
    1. High Score

Khi điểm số người chơi đạt được điểm cao hơn một trong 5 điểm cao nhất, khi lượt chơi kết thúc, một thông báo hiện lên để người chơi nhập tên vào.



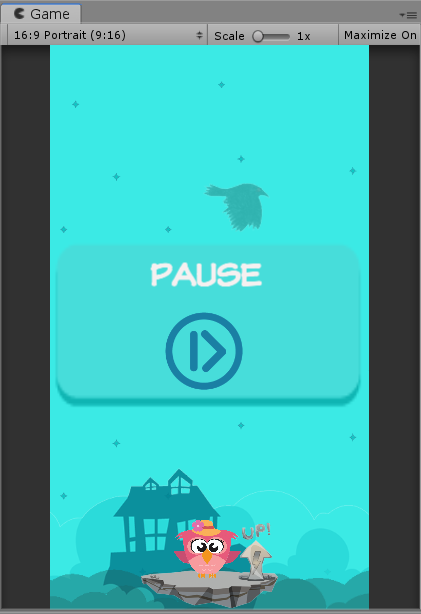
* + - 1. Giao diện nhập tên khi được điểm cao

Nhấn submit sẽ hiện lên Leaderboard để biết điểm số của top 5 điểm số tốt nhất bạn đạt được.

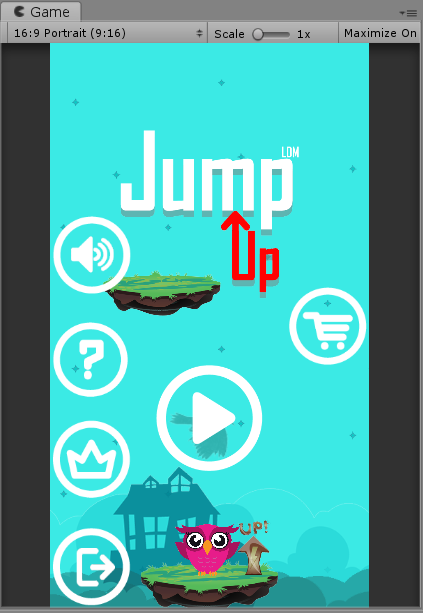


* + - 1. Giao diện leaderboard
    1. Tạm dừng chơi

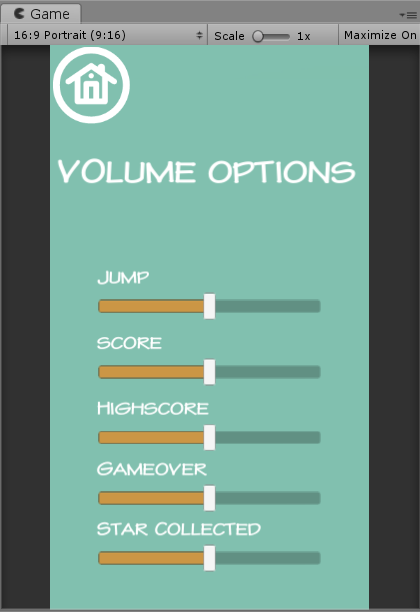
Khi đang chơi người chơi muốn tạm dừng nhấn biểu tượng pause , game sẽ được tạm dừng và hiện lên màn hình pause.



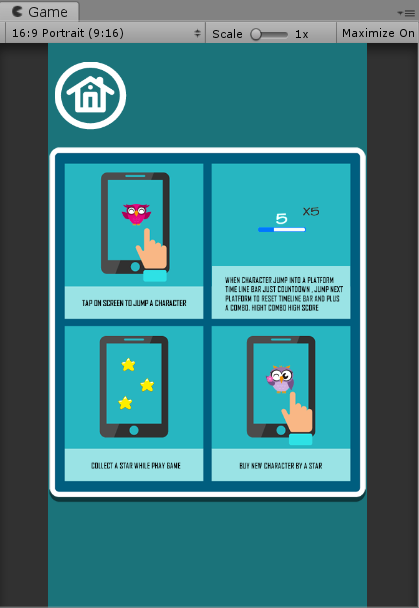
* + - 1. Giao diện khi tạm dừng chơi
  1. GIAO DIỆN CHÍNH TRONG GAME
     1. Giao diện màn hình chính



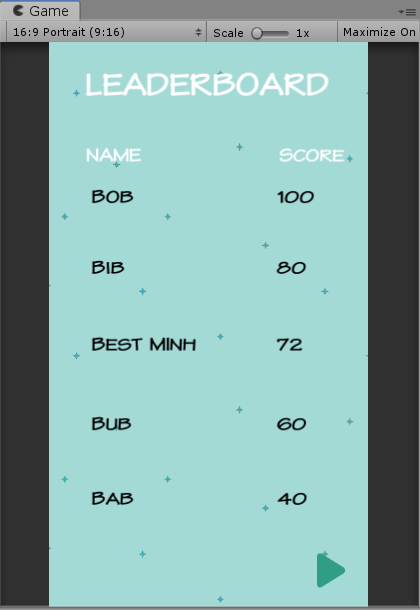
* + - 1. Giao giạn mành hình chính
    1. Giao diện volume



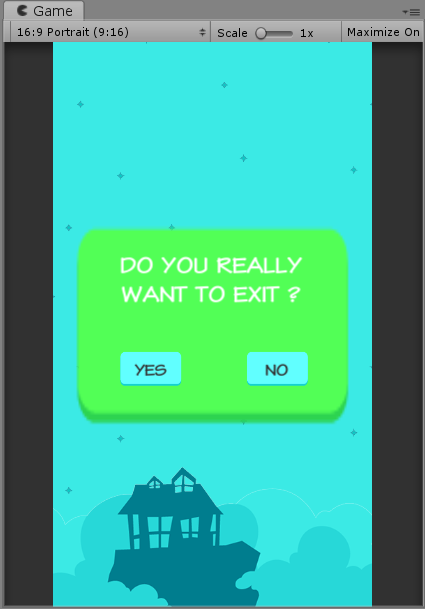
* + - 1. Giao diện tăng giảm âm thanh
    1. Giao diện hướng dẫn chơi game



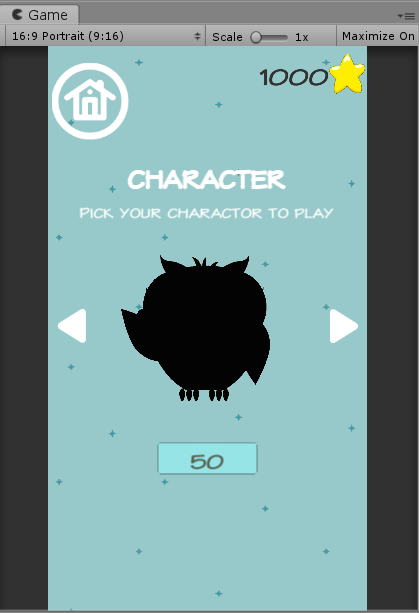
* + - 1. Giao diện hướng dẫn chơi game
    1. Giao diện điểm cao



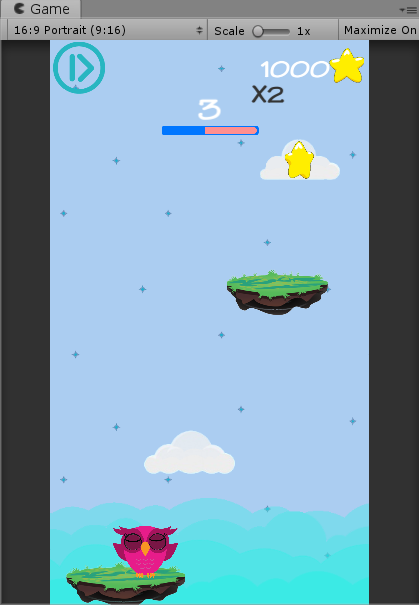
* + - 1. Giao diện xem điểm cao
    1. Giao diện thoát game



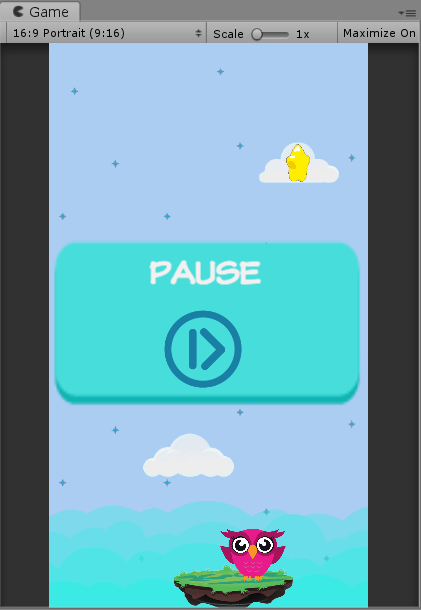
* + - 1. Giao diện thoát game
    1. Giao diện cửa hàng



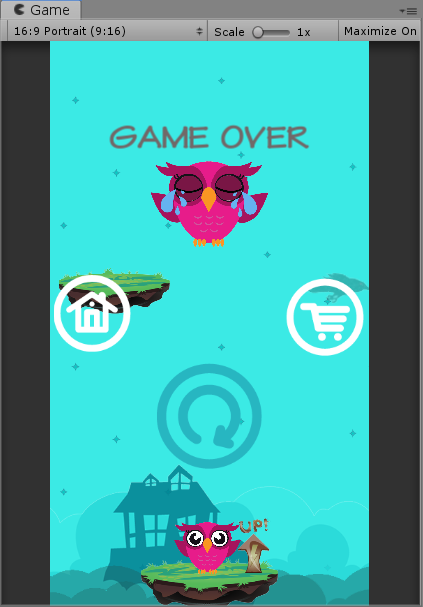
* + - 1. Giao diện cửa hàng
    1. Giao diện màn hình chơi chính



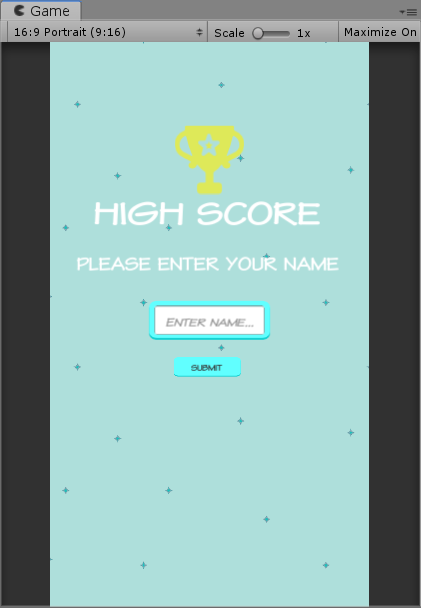
* + - 1. Giao diện màn hình chơi chính
    1. Giao diện dừng chơi



* + - 1. Giao diện dừng chơi
    1. Giao diện gameover



* + - 1. Giao diện gameover
    1. Giao diện đạt điểm cao mới



* + - 1. Giao diện đạt điểm cao mới
  1. MỘT SỐ CODE QUAN TRỌNG TRONG GAME
     1. Di chuyển camera.

Khi người chơi nhảy lên được 1 platform mới thì camera di chuyển lên.

Trong Player.cs khai báo.

public delegate void MoveCamera();

public static event MoveCamera move;

void Update()

{

if (hasJump && rb.velocity.y == 0)

{

if (move != null)

{

move();

}

}

}

Trong CameraScript.cs khai báo.

public class CameraScripts : MonoBehaviour {

private bool canMove;

public float distance = 4.1f;

public float newDestination;

public float timeMovingCamera = 3f;

// Update is called once per frame

void Update () {

MoveCamera();

}

private void OnEnable()

{

Player.move += Move;

}

private void OnDisable()

{

Player.move -= Move;

}

private void MoveCamera()

{

if (canMove)

{

Vector3 temp = transform.position;

temp.y += timeMovingCamera \* Time.deltaTime;

transform.position = temp;

if(transform.position.y >= newDestination)

{

canMove = false;

}

}

}

void Move()

{

newDestination = transform.position.y + distance;

canMove = true;

}

}

Giải thích:

Trong Player script

Khai báo 1 biến delegate. Biến delegate là biến có thể gán hàm vào trong đó và sử dụng biến đó như 1 biến bình thường.

Sau đó khai báo event của delegate . Event của delegate gần giống với multi-delegate. Event có thể đăng ký hàm gán vào delegate và cũng có thể gỡ hàm ra khỏi delegate.

Trong hàm Update , kiểm tra xem người chơi đã nhảy lên chưa, nếu nhảy lên rồi thì biến hasJump trả về true và vận tốc theo trục y bằng 0. Có nghĩa là người chơi đã nhảy lên và đứng lên một platform mới thì gọi hàm move của event.

Trong Camera script

Hàm Enable chạy khi GameObject active và OnDisable khi GameObject deactive, sử dụng điều này để khai báo hàm Move cho event ở script Player.

Khi event ở Player script được gọi thì hàm Move() trong Camera script cũng được gọi.

Nó sẽ set vị trị mới và giá trị canMove bằng true. Khi canMove bằng true thì điều kiện ở hàm MoveCamera đúng , nó sẽ thực hiện việc chạy camera lên.

* + 1. Khởi tạo một platform mới.

Trong Player script

void Update()

{

if (hasJump && rb.velocity.y == 0)

{

if (!platformBound)

{

GameController.instance.CreatePlatform();

}

}

}

Trong GameController script

public static GameController instance;

private void MakeSingleton()

{

if (instance == null)

{

instance = this;

}

}

void Awake()

{

MakeSingleton();

}

public void CreatePlatform()

{

lastPlatformPositionY += DistanceBetweenPlatform;

GameObject newPlat = ObjectPooler.Instance.GetFromPool();

newPlat.transform.position = new Vector3(0, lastPlatformPositionY, 0);

}

Trong ObjectPooler script

public class ObjectPooler : MonoBehaviour

{

public GameObject[] prefabs;

public int amountObject;

private Queue<GameObject> availableObjects = new Queue<GameObject>();

public static ObjectPooler Instance;

public GameController gameController;

private void Awake()

{

Instance = this;

}

private void Start()

{

GrowPool();

}

public GameObject GetFromPool()

{

if (availableObjects.Count == 0)

{

GrowPool();

}

GameObject instance = availableObjects.Dequeue();

instance.SetActive(true);

return instance;

}

void GrowPool()

{

for (int i = 0; i < amountObject; i++)

{

GameObject instanceToAdd = Instantiate(prefabs[gameController.ran]);

instanceToAdd.transform.SetParent(transform);

AddToPool(instanceToAdd);

}

}

public void AddToPool(GameObject instance)

{

instance.transform.position = Vector3.zero;

instance.SetActive(false);

availableObjects.Enqueue(instance);

}

}

Giải thích :

Trong Player script

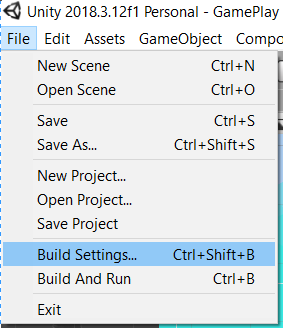
Khi người chơi nhảy lên 1 platform thì nó sẽ gọi tới hàm CreatePlatform ở script GameController thông qua biến static instance.Ở script GameController người lập trình sử dụng 1 kỹ thuật trong Design Pattern gọi là Singleton . Công dụng của Singleton pattern là gán giá trị ban đầu cho biến static , tránh để bị lỗi.

Trong GameController script

Ở hàm CreatePlatform người lập trình lại gọi đến một script khác là ObjectPooler .Ở đây người lập trình sử dụng một kỹ thuật Design Pattern khác gọi là Object Pool .Kỹ thuật này cho phép người lập trình sử dụng lại tài nguyên platform mà không sinh ra cái mới. Công dụng của Pattern này để giảm thiểu sử dụng tài nguyên phần cứng của thiết bị, làm giảm tình trạng giật , lag.

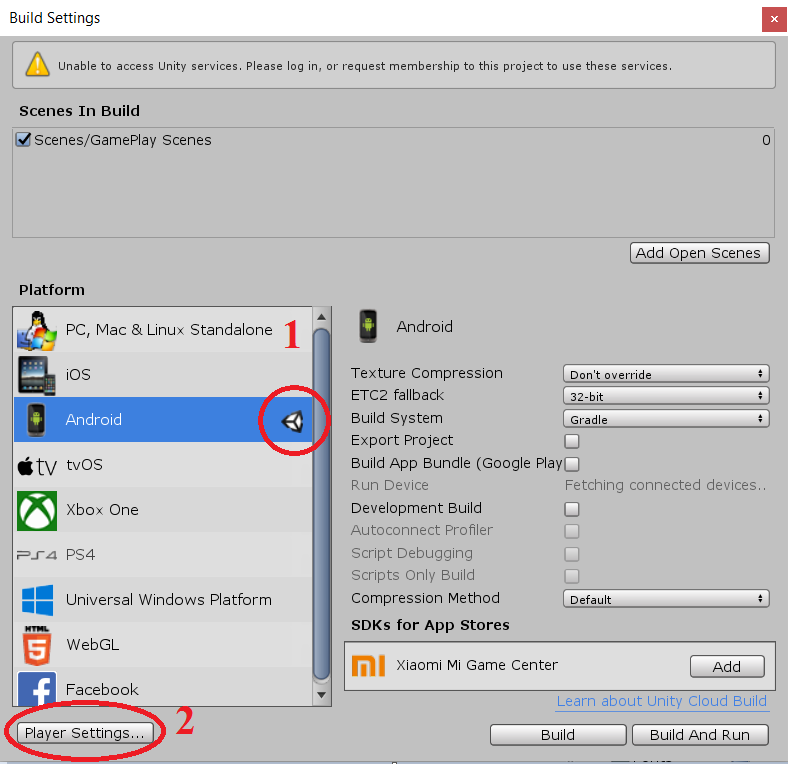
1. BUILD GAME RA HỆ ĐIỀU HÀNH ANDROID
   1. CÁC BƯỚC THỰC HIỆN

Chọn File -> Build Settings hay phím tắt Ctrl + Shift + B



* + - 1. Build Settings

Một bảng Build Setting hiện lên, nhấn tiếp vào Player Settings



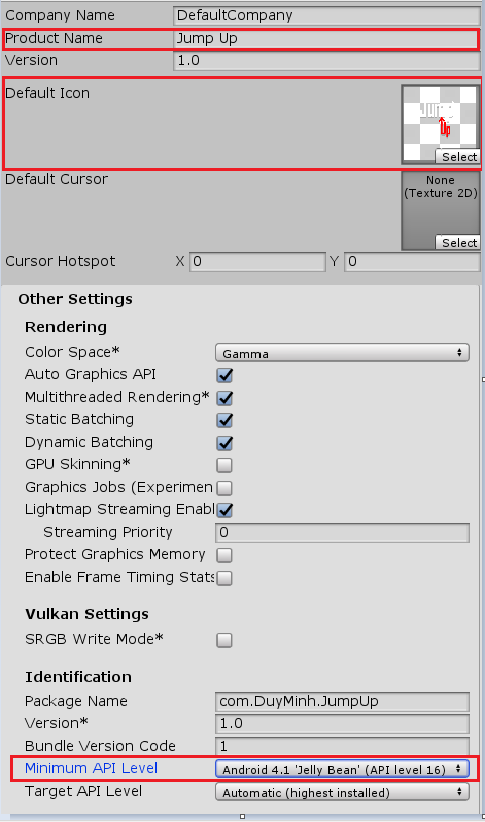
* + - 1. Lựa chọn hệ điều hành để build

Trong bảng Player Setting có 3 mục quan trọng:

Product Name: là tên game sau khi build ra file apk.

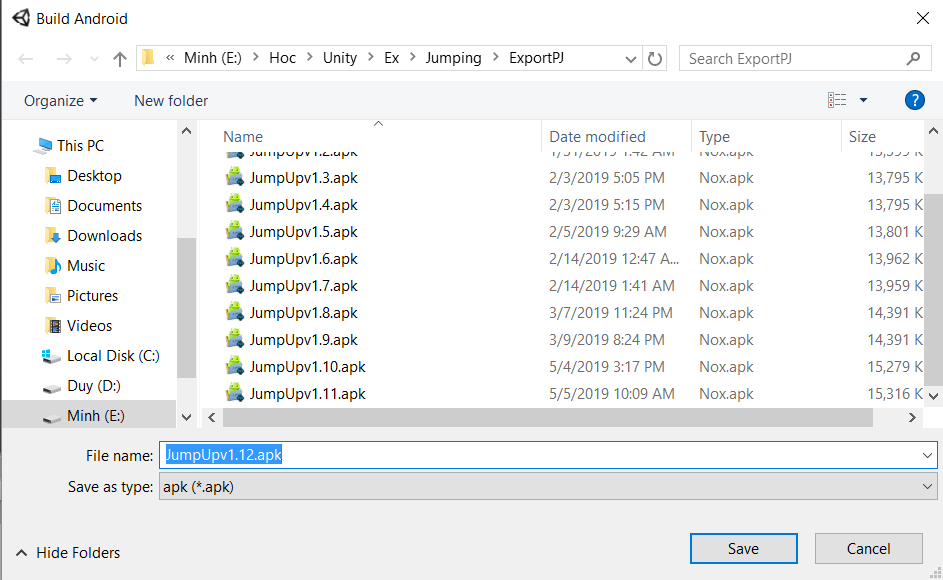
Default Icon là hình sẽ hiện lên trong thiết bị.

Minimum API Level: Phiên bản Android thấp nhất.



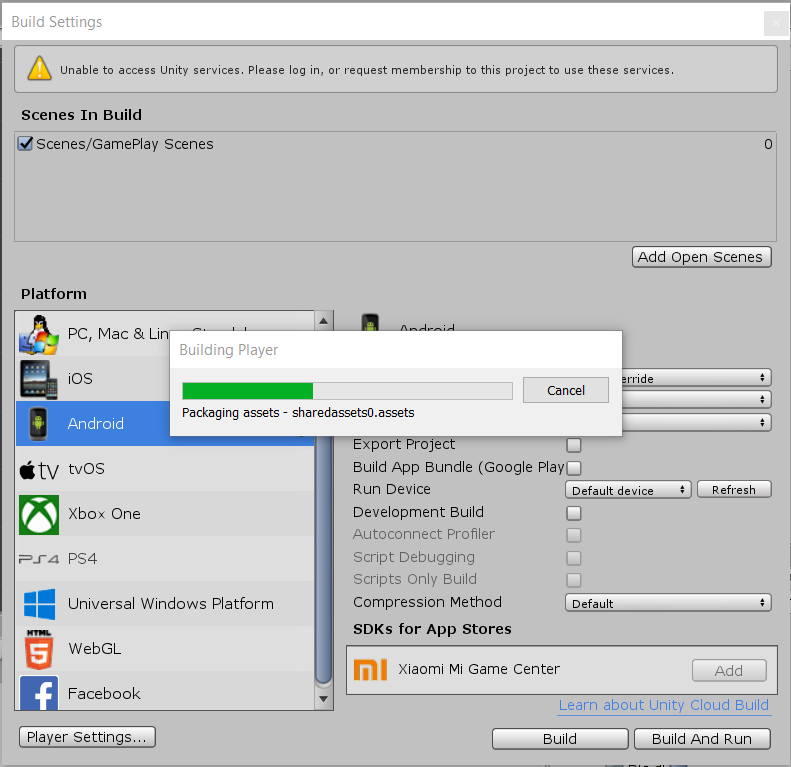
* + - 1. Player Settings

Sau khi đã setting xong trong Player Setting, quay lại Build Setting nhấn Build , một cửa sổ hiện ra file build đặt ở thư mục nào.



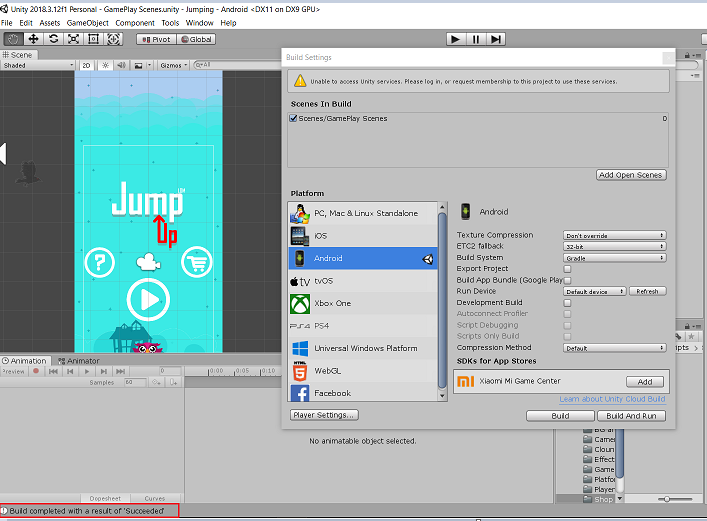
* + - 1. Giao diện build ra file android

Sao khi nhấn Save , Unity tiến hành build game .



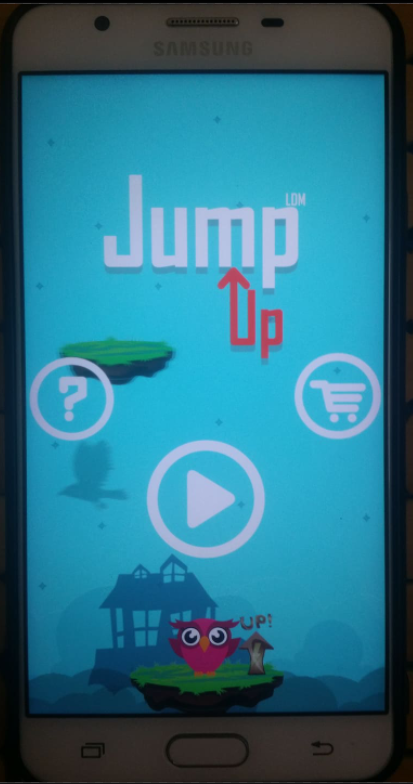
* + - 1. Giao diện game đang được build

Thông báo sau khi build xong.



* + - 1. Thông báo sau khi build game xong

Giao diện game sau khi cài lên thiết bị



* + - 1. Game sao khi cài lên thiết bị

KẾT LUẬN

Bước đầu tìm hiểu về Unity và Game Engine người thực hiện gặp rất nhiều khó khăn trong việc tìm tài liệu và xây dựng ý tưởng game vì tài liệu về unity đa phân bằng tiếng anh, nhưng sau khoảng 1 tháng cố gắng tìm tòi, học hỏi các bạn đi trước thì người thực hiện cũng đã làm cho mình được 1 game gần hoàn chỉnh để phục vụ cho đề tài tốt nghiệp. Trong thời gian làm đề tài người thực hiện cũng có một số những vấn đề đạt được và chưa đạt được.

Đạt Được

Hiểu được cấu trúc, cách thức hoạt động, cách thức phát triển ứng dụng trên Unity.

Nâng cao kiến thức lập trình và tiếng anh chuyên ngành thông qua những tài liệu về Unty bằng tiếng anh.

Xây dựng được các nhân vật trong game bằng phần mêm hỗ trợ photoshop

Tổ chức load các màn tương ứng thông qua các button.

Xây dựng các chức năng thông qua ngôn ngữ lập trình C# trên MonoDevelop.

Tạo âm thanh cho cho trò chơi và 1 số chức năng khác.

Ngoài những thứ đặt được thì người thực hiện còn rất nhiều những vấn đề chưa đạt được.

Chưa Đạt Được

Do kiến thức của người thực hiện đề tài về Unity còn hạn hẹp nên các chức năng trong game còn nghèo nàn.

Chưa xây dựng được hoàn chỉnh 1 game như mong muốn.

Các Script trong Unity tổ chức chưa thực sự rõ ràng.

Thuận Lợi

Vận dụng được các kiến thức mà người thực hiện đã học trong tường để làm đồ án này

Được sự chỉ dạy tận tình của thầy và các bạn

Tài liệu rất nhiều trên Internet

Khó Khăn

Unit là một môi trường lập trình mới đối với người thực hiện nên khi bắt tay vào làm thì người thực hiện không biết làm từ đâu và cảm thấy bỡ ngỡ.

Kinh Nghiệm Rút Ra

Phải biết sắp xếp thời gian 1 cách hợp lý nếu không sẽ gặp rất nhiều khó khăn

Làm việc phải tập trung mới có thể hoàn thành tốt được

Hướng Phát Triển

Tối ưu hóa để game có thể chạy nhanh hơn.

Tăng thêm nhân vật để người chơi có nhiều lựa chọn

Thêm các chướng ngại vật khác.

Cung cấp thêm công cụ để người chơi có thể chọn và chỉnh sửa tạo riêng cho mình những nhân vật, hiệu ứng âm thanh,…

TÀI LIỆU THAM KHẢO

[1]<https://perfectprogrammers.blog/2018/12/06/hieu-ve-delegate-trong-c/?fbclid=IwAR0eEKE70zADnJZ7zWq0qm3s8VR4u9MZatwMwM_IBXP2uKU9dXJ9aDla2Xs> (Truy cập 3-2019)

[2] <https://unity3d.com/learn/tutorials/s/2d-game-creation> (Truy cập 12-4-2019)

[3] <https://docs.unity3d.com/ScriptReference/Rigidbody.AddForce.html> (Truy cập 15-4-2019