Tại sao phải sử dụng Addressables?

Tóm tắt

Hệ thống `Addressables` của Unity là một hệ thống quản lý asset động (*dynamic*). Nó được xây dựng dựa trên nền tảng công nghệ `AssetBundles` của Unity đồng thời cung cấp các công cụ ngay bên trong Unity Editor để giúp bạn chuẩn bị asset để load theo yêu cầu. Bất kể nội dung nào dù trên thiết bị hay trên cloud.

Trong hướng dẫn này, bạn sẽ có một cái nhìn tổng quan về cách tiếp cận của Unity đối với việc quản lý asset động, các lợi ích mà `Addressables` mang lại.

Bài hướng dẫn này bạn sẽ:

- Định nghĩa thế nào là quản lý asset động `(dynamic asset management)`
- Giải thích khái niệm AssetBundle và các lợi ích của nó.
- Giải thích cách hệ thống Addressables đơn giản hóa việc quản lý AssetBundle.

Tổng quan

Bạn đã bao giờ muốn tạo ra một tựa game mobile có hiệu suất chạy mượt mà trên gần như mọi thiết bị chưa? Hay việc thêm các DLC hoặc cập nhật nội dung theo chủ đề ngày lễ cho game của bạn thì sao? Đây chính là lúc hệ thống `Addressables` phát huy tác dụng.

Hệ thống `Addressables` của Unity là một giải pháp quản lý asset động, giúp cung cấp cho người dùng chỉ những asset mà họ thực sự cần, vào đúng thời điểm họ cần chúng. Nó cũng giúp bạn quy hoạch các asset của mình để đóng gói thành các `AssetBundles` ngay từ bên trong Unity Editor trong quá trình bạn phát triển game. Đồng thời, Runtime API của nó cho phép bạn load và unload asset một cách linh hoạt khi người dùng đang chơi game.

Bạn có thể chọn đóng gói tất cả asset đi kèm với bản build hoặc lưu trữ chúng trên một mạng lưới phân phối từ xa (CDN) ví dụ như Unity Cloud Content Delivery của Unity. Vị trí lưu trữ của asset được trừu tượng hóa thông qua các địa chỉ là một `string` do bạn tự định nghĩa.

Trong bài hướng dẫn này, bạn sẽ tìm hiểu về cách tiếp cận của Unity đối với việc quản lý asset động và lý do tại sao hệ thống `Addressables` có thể là một sự bổ sung đáng giá cho dự án của bạn.

Bạn cũng sẽ nắm được thêm thông tin về khóa học này, để từ đó sẵn sàng bắt tay vào làm việc với hệ thống `Addressables`.

Trước khi bắt đầu

Get started with Addressables là một khóa trung cấp.

Để hoàn thành khóa học này bạn cần có kinh nghiệm lập trình với Unity. Bên cạnh các kiến thức nền tảng đã được đề cập trong lộ trình Junior Programmer, bạn cũng nên nắm vững kiến thức về `event handlers` và `asynchronous methods`.

Chúng tôi cũng khuyến nghị bạn hoàn thành lộ trình Creative Core, hoặc có các kinh nghiệm tương đương để làm quen với tất cả các asset trong Unity mà bạn có thể áp dụng hệ thống `Addressables` với chúng.

Quản lý asset động là gì?

Hệ thống quản lý asset động (`Dynamic asset management`) giúp bạn cải thiện hiệu suất game bằng cách cho phép bạn quyết định chính xác việc tải asset **khi nào** và **từ đâu,** quyết định khi nào cần nạp vào và xóa khỏi bộ nhớ trong runtime.

Trong suốt lịch sử phát triển của Unity, đã có một vài phương pháp khác nhau để quản lý asset trong bộ nhớ. Hệ thống `Addressables` được thiết kế để thay thế hoặc cái tiến các hệ thống cũ - những hệ thống vốn đã lỗi thời hoặc không còn đủ khả năng đáp ứng cho một dự án vượt xa khỏi giai đoạn prototype. Trong khóa học này, bạn sẽ tiến hành nâng cấp một tựa game đang sử dụng một số phương pháp cũ để chuyển hoàn toàn sang hệ thống `Addressables`.

Dưới đây là một vài công cụ quản lý asset khác mà Unity đang hoặc đã từng cung cấp.

Tham chi?u tr?c ti?p (Direct references)

Khi bạn kéo thả một asset vào một scene hoặc vào một component thông qua Inspector của Editor, bạn đang tạo ra một tham chiếu trực tiếp tới asset đó. Khi bạn build ứng dụng của mình các asset này được lưu lại trong một tệp dữ liệu riêng biệt liên kết chặt chẽ với scene của bạn. Khi ứng dụng chạy trên thiết bị và scene được load lên, Unity Player sẽ nạp toàn bộ asset này vào bộ nhớ RAM trước khi scene thực sự xuất hiện, nhằm đảm bảo rằng scene có sẵn quyền truy cập vào tất cả mọi thứ mà nó có tham chiếu.

Nếu bạn chỉ sử dụng tham chiếu trực tiếp trong game thì cách quản lý này không linh hoạt (*not dynamic*). Việc load và unload toàn bộ scene là quyền kiểm soát duy nhất mà bạn có. Bạn có thể sẽ phải đối mặt với rủi ro tạo ra một game có build size khổng lồ và hiệu suất siêu tệ đặc biệt là trên các thiết bị có dung lượng RAM hạn chế.

Th? m?c Resources

Thư mục Resources và Resource API đã từng cung cấp một cách đơn giản để quản lý asset trong bộ nhớ. Trong cấu trúc thư mục của các dự án Unity cũ, bạn có thể bắt gặp một hoặc nhiều thư mục Resources chứa asset. Trong quá trình build ứng dụng, Editor sẽ tìm kiếm asset ở tất cả các thư mục Resources và đóng gói chúng thành một tệp tuần tự (`serialized file`) đi kèm với nó là `metadata` và `indexing infomation`, tệp này sau đó được đóng gói thẳng vào ứng dụng của bạn. Resource API cho phép bạn viết script để load và unload các asset nằm trong thư mục Resources này.

Hệ thongse Resources tồn tại một số nhược điểm so với các hệ thống mới hơn:

- Nó không cho phép quản lý bộ nhớ một cách chi tiết.
- Nó làm chậm thời gian khởi động ứng dụng.
- Nó có thể làm phình to dung lượng bản build
- Và nó hoàn toàn không hỗ trợ phân phối nội dung từ xa (CDN)

H? th?ng AssetBundle

Hệ thống AssetBundle quy hoạch asset vào các thùng containers được gọi là AssetBundles. Giống như thư mục Resources, hệ thống AssetBundle gom asset thành các tệp riêng biệt. Tuy nhiên, khác với Resources, AssetBundles có thể được lưu trữ cục bộ đi kèm với bản build hoặc lưu trữ trên cloud.

Thông qua bộ API của mình, hệ thống AssetBundles giúp giảm thiểu tối đa sự tiêu tốn băng thông và asset hệ thống. Nó làm được điều đó bằng cách cho phép bạn tải xuống các bundle chỉ khi nào cần thiết, nhờ đó mà bạn có thể bổ sung DLC và các bản cập nhật nội dung sau khi game đã ra mắt . Ví dụ bạn có thể phân phối nội dung mới để người chơi nạp tiền mua mà không bắt buộc họ phải tải lại toàn bộ file cài đặt mới từ store. Một khi các bundle được tải về máy, AssetBundles API cũng cấp các phương thức để load và unload asset từ các bundle đó.

Mặc dù là một hệ thống mang tính tùy biến rất cao, AssetBundles vẫn có những hạn chế nhất định khiến các lập trình viên trước đây phải tự xây dựng các giải pháp của riêng họ.

- AssetBundles chỉ có thể sử dụng thông qua scripts, giao diện trong Unity Editor rất hạn chế và quá trình build AssetBundles cũng yêu cầu bạn phải viết code.
- Bản thân AssetBundles API không tự động theo dõi sự phụ thuộc của asset giữa các AssetBundles khác nhau. Ví dụ: nếu bạn muốn load một prefab từ AssetBundle A, bạn sẽ phải tự tìm mọi thành phần phụ thuộc của nó như mesh, material, texture... nằm rải rác ở các AssetBundle khác. Sau đó bạn phải đảm bảo rằng các AssetBundles phụ thuộc đó đã được load xong trong lúc game chạy và trước khi bạn load prefab đó.
- Việc cấp phát và giải phóng bộ nhớ là tự tiếp và hoàn toàn thủ công. Do đó, rất dễ xảy ra tình trạng bạn vô tình unload một asset khỏi AssetBundle trong khi các đoạn code khác vẫn cần sử dụng asset đó. Điều này có thể dẫn đến lỗi thiếu nội dung hoặc rò rỉ bộ nhớ. Vấn đề này trở nên cực kỳ rắc rối với race conditions đòi hỏi bạn phải gia cố code của mình thật chặt chẽ để phòng ngừa.
- AssetBundles runtime API không hề biết bạn đang lưu trữ các bundle đã build ở trên thiết bị hay trên cloud. Điều này bắt buộc bạn phải tự lưu và theo dõi đường dẫn của từng AssetBundle, xem nó nằm trên cloud hay trên thiết bị để gọi hàm tải về cho đúng.

Addressables

Hệ thống Addressables được xây dựng dựa trên nền tảng của AssetBundle API nhằm cung cấp cho bạn một giao diện người dùng trực quan ngay trong Unity Editor, đồng thời tự động hóa các quy trình mà trước đây vốn chỉ có thể quản lý thủ công thông qua việc viết code. Với hệ thống Addressables, bạn có thể quy hoạch các asset của mình ngay trong Unity Editor và để mặc cho hệ thống tự xử lý các vấn đề dependencies, vị trí lưu trữ cũng như việc cấp phát và giải phóng bộ nhớ.

Tại sao bạn nên sử dụng Addressables?

Khi một asset được đánh dấu là Addressable nó sẽ sở hữu một địa chỉ duy nhất mà bạn có thể dùng để tham chiếu tới trong scripts của mình thay vì phải gọi bằng tên hay vị trí trong bundle. Mặc dù nghe có vẻ đơn giản nhưng địa chỉ này lại là chìa khóa cho phép hệ thống Addressables tự động hóa rất nhiều chi tiết phức tạp ở tầng dưới.

Dưới đây là giải thích chi tiết tại sao bạn nên sử dụng Addressables

S? linh ho?t

Hệ thống Addressables mang lại cho bạn sự linh hoạt trong việc chỉ định nơi lưu trữ asset. Bạn có thể để asset đi kèm trong bản build hoặc tải xuống khi cần từ một máy chủ. Sau đó bạn có thể thay đổi vị trí lưu trữ của một asset cụ thể, chẳng hạn như từ local sang remote hoặc từ một bundle nguyên khối (*monolithic bundle*) sang các bundle phân mảnh (*granular bundles*) khác. Tát cả đều có thể thược hiện mà không cần phải viết lại source code.

Qu?n lý s? ph? thu?c

Hệ thống Addressables tự động nạp tát cả các thành phần phụ thuộc của bất kỳ asset nào mà bạn load sao cho tất cả các mesh, shader, animation và các asset khác được nạp đầy đủ trước khi asset mà bạn yêu cầu được load lên.

Ví dụ: nếu bạn load một nhân vật thì hệ thống sẽ tự động load đầy đủ mesh, material và cả texture, shader mà material cần.

Qu?n lý b? nh?

Hệ thống Addressables tự động hóa phần lớn công việc tẻ nhạt của quá trình quản lý bộ nhớ. Khi bạn load và unload asset khỏi game thông qua code, hệ thống sẽ tự động theo dõi việc cấp phát bộ nhớ.

Bạn cũng có thể sử dụng các công cụ profiler của hệ thống để cải thiện hiệu năng sử dụng bộ nhớ có game của mình.

?óng gói n?i dung hi?u qu?

Bởi vì hệ thống Addressables thiết lập bản đồ cho các chuỗi phụ thuộc phức tạp nên nó giúp bạn đóng gói asset một cách hiệu quả ngay cả khi bạn chuyển hoặc đổi tên asset. Bạn có thể lựa chọn mức độ phân mảnh mà bạn muốn đối với các asset tại một vị trí lưu trữ, ưu tiên đóng gói riêng biệt hoặc gộp chúng lại với nhau. Bạn cũng có thể dễ dàng chuẩn bị asset cho cả việc triển khai local lẫn remote để hỗ trợ các nội dung được load theo yêu cầu, DLC. Điều này có thể giúp giảm build size.

Cloud Build và Phân ph?i n?i dung

Hệ thống Addressables đã được tích hợp vào nền tảng Unity Gaming Services (UGS), cụ thể là các dịch vụ Cloud Content Delivery và Cloud Build, mang đến cho bạn các dịch vụ toàn diện từ cập nhật game trực tiếp cho đến build ứng dụng.

Scriptable Build Pipeline

Hệ thống Addressables sử dụng Scriptable Build Pipeline (SBP), một quy trình mạnh mẽ và ổn định hơn so với build AssetBundle cũ. Bạn có thể sử dụng các luồng build được thiết lập sẵn hoặc sử dụng phương pháp nâng cao hơn nếu muốn. Bạn có thể tự tạo luồng build của riêng mình thông qua việc sử dụng API đã được chia nhỏ.

Localization

Hệ thống này cũng được tích hợp gói Localization của Unity để bạn có thể sử dụng nhiều ngôn ngữ khác nhau trong dự án của mình.

Chúng ta sẽ làm gì trong khóa học này?

Trong khóa học này chúng ta sẽ được học:

- Các kiến thức cơ bản về giao diện của hệ thống Addressables
- Những cách để biến asset thành addressable và load chúng bằng code.
- Cách Unity build ứng dụng với asset là addressable và cách quy hoạch dự án sao cho build hiệu quả nhất.
- Tùy chọn lưu trữ asset phù hợp với mục đích của bạn.

Trong các bài hướng dẫn tiếp theo, bạn sẽ học các kiến thức cơ bản về hệ thống Addressables thông qua việc thực hành với dự án của chúng tôi, một tựa game có tên là Loady Dungeons. Loady Dungeons được phát triển bằng cách sử dụng các tham chiếu trực tiếp và hệ thống Resources cũ. Trong suốt quá trình học, bạn sẽ tiến hành nâng cấp dự án này để chuyển sang hệ thống Addressables và học hỏi mọi thứ về addressable.

Bước tiếp theo

Cho dù bạn là một indie dev muốn thu hút sự chú ý trên các nền tảng di động hay là thành viên của một đội ngũ sản xuất lớn với những nhu cầu phức tạp về phân phối nội dung, hệ thống Addressables sẽ giúp tựa game của bạn trở nên tinh gọn và hoạt động nhanh hơn, đồng thời tự động hóa nhiều tác vụ quản lý tài nguyên trong runtime.

Tiếp theo, đã đến lúc bắt tay vào thực hành trong Unity. Bạn sẽ khám phá dự án Loady Dungeons và bắt đầu tiến hành thiết lập các asset thành addressable.

Revision #2
Created 2026-04-13 15:53:47 UTC by ThanhDV
Updated 2026-04-14 16:38:31 UTC by ThanhDV