1. Script lifecycle flowchart

Awake()

OnEnable()

Start()

FixedUpdate()

Update()

LateUpdate()

OnGUI()

OnApplicationQuit()

OnDisable()

OnDestroy()

2. Phân biệt Awake và OnEnable

**Awake:**các hàm Awake đều được gọi trước **TẤT CẢ** các hàm Start của các MonoBehaviour có trong Scene hay gọi ngay sau khi gameObject đó được sinh ra bằng hàm Instantiate(). Chỉ được gọi một lần duy nhất trong vòng đời của một MonoBehaviour

**OnEnable*:*** hàm này được gọi khi một gameObject được đổi trạng thái từ *deactive -> active* hoặc khi enable Component và được gọi lần đầu tiên ngay sau hàm Awake của nó.

3. Một game object đc gán 1 script, trong script có 2 hàm là Awake và Start,

Trong trường hợp script không đc enable, thì hàm nào sẽ đc gọi?

Trong trường hợp gameobject deactive, thì hàm nào sẽ đc gọi?

Awake được gọi kể cả khi script KHÔNG được enable. Trong khi Start thì sẽ không được gọi.

Tuy nhiên trong trường hợp GameObject deactive thì không có hàm nào được gọi.

4. **Update** là hàm được gọi mỗi lần vào mỗi frame và được xem như hàm xử lý chính của vòng lặp game. Vậy thì tại sao lại còn có 2 hàm LateUpdate và FixedUpdate

* LateUpdate() được gọi như Update, mỗi frame một lần, điểm khác biệt duy nhất là nó được gọi sau khi các hàm Update của tất cả các MonoBehaviour đã được thực thi xong.

Một chức năng kinh điển của hàm này là khi sử dụng camera follow theo player, sau khi vị trí cũng như rotation của player đã được tính toán hoàn tất trong Update, thì ta sẽ LateUpdate vị trí và góc xoay của camera theo player, đảm bảo được độ chính xác cao.

* Các xử lý liên quan tới vật lý thì chúng ta nên đặt ở trong **FixedUpdate** thay vì Update để Physics System của Unity có thể cập nhật vị trí và xét va chạm kịp thời.

**FixedUpdate**: Hàm này được gọi không phụ thuộc vào vòng lặp chính của game mà gọi theo vòng lặp của vật lý trong game, thường nó được gọi cố định vào mỗi 0.02s theo mặc định của project setting, con số này gọi là Fixed Timestep hay fixedDeltaTime. Vì vậy FixedUpdate có thể được gọi nhiều hơn một lần hoặc không gọi trong 1 frame tùy theo FPS của game.

5. Giả sử có 1 game chúng ta chơi trên PC có thể đạt 60 FPS, trong khi máy console cũng cùng game đó bị giới hạn ở 30 FPS. Có cách nào để giải quyết việc phụ thuộc vào frame rate không ?

Delta time is the solution to solving our problem, freeing our code from the frame rate, making it **frame rate independent.**

5. Giả sử có 2 gameObjects đều có component Colilders mà chúng không phát hiện va chạm?

2 gameObjects chỉ có Collider mà không có Rigidbody ở một trong 2 sẽ không có va chạm, bởi chúng không tính là xử lý vật lý nếu không có Rigidbody.

6. Trong môi trường game 2D, khi sử dụng rigibody2D, có gặp trường hợp 1 game object di chuyển xuyên qua hoặc chồng lên 1 game object khác? Nếu có thì cách khắc phục là gì?

Thay thế Discrete = Continuous.

Continuous xử lý được trường hợp object di chuyển nhanh hoặc fixedDeltaTime lớn. Nhưng Performance thấp hơn Discrete

7. Loại thư mục đặc biệt nào trong Assets mà các files trong thư mục này có thể load lên lúc game chạy? Assets/Resources và StreamingAssets

Đây là nơi có thể đặt các assets như prefab, scriptable object, texture, sprite… sau đó dùng script để tìm và load lúc game chạy (runtime).

Tất cả các assets nằm trong folder Resources luôn được build kể cả khi các asset này không được “trỏ” hoặc link tới.

8. Khác biệt giữa 2 folder StreamingAssets và Resources?

Các files ở trong folder này cũng có thể tìm và load lúc runtime bằng script đồng thời luôn đi “kèm” với build.

các files trong StreamingAssets không được “encoded” tức là nó sẽ không được build mà là “đi kèm” với build.

Ví dụ: file .png đặt trong Resources sẽ được “encoded” hay compress thành ETC compressed texture. Còn file .png trong StreamingAssets thì vẫn giữ là .png

9. Dùng gì để Wait for frame, wait for real time

10. Addressable

11. Asset bundle

12. Tối ưu hóa game

13. <https://phuongne.com/graphic-for-game-developer/>

14. Tại sao Singleton lại là anti-pattern

<https://phuongne.com/singleton-pattern-trong-unity3d/>

15. Lighting

16. 9-slicing Sprites

17. Tilemap

17. AR, VR

2. Shader là gì, có tác dụng như thế nào, đc viết dưới dạng ngôn ngữ nào?

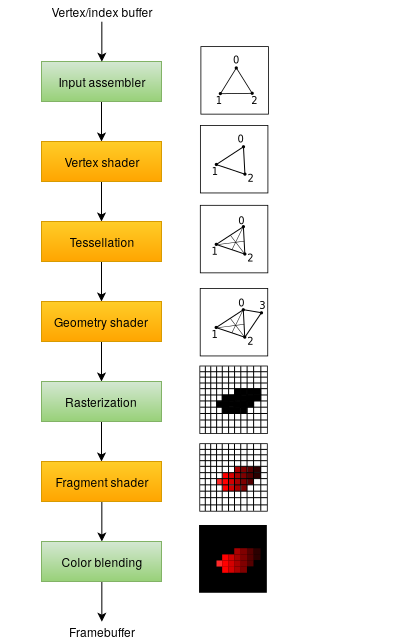
Shader có thể coi là tên gọi chung của loại script được viết ra để chạy trên GPU hay can thiệp vào quá trình vẽ để tạo các vật thể, hiệu ứng đặc biệt.

Shader thì được viết bằng ngôn ngữ bậc cao Cg/HLSL. High Level Shading Language

Phong shading and the [Phong reflection model](https://en.wikipedia.org/wiki/Phong_reflection_model) were developed at the [University of Utah](https://en.wikipedia.org/wiki/University_of_Utah) by [Bui Tuong Phong](https://en.wikipedia.org/wiki/Bui_Tuong_Phong), who published them in his 1973 Ph.D dissertation[[3]](https://en.wikipedia.org/wiki/Phong_shading#cite_note-3)[[4]](https://en.wikipedia.org/wiki/Phong_shading#cite_note-4) and a 1975 paper

Unity hỗ trợ 3 loại shader bao gồm: Surface shader, Vertex & Fragment shader và Fixed Function

* Nếu cần một Material đơn giản có khả năng mô phỏng hay tương tác với ánh sáng, dùng Surface shader
* Vertex & Fragment shader: Loại shader này làm việc như gần như là cách mà GPU vẽ một vật thể vậy, không có hỗ trợ thêm lighting như surface shader, tuy nhiên điều này cũng cho ta khả năng custom cao hơn, khả năng kiểm soát cũng tốt hơn.
* Fixed Function shader đã trở nên lỗi thời khi chỉ có thể tạo các hiệu ứng đơn giản, không còn phù hợp trong thời đại game hiện nay.

Quá trình xử lý với vertex & fragment shader

3. Làm thế nào GPU có thể vẽ các dữ liệu đã nhận được từ CPU lên màn hình?