Bài 3: Tạo và cập nhật Heads-Up Display

Giảng viên:

Mục tiêu

- Vai trò điển hình của một Heads-Up Display (HUD)
- Các bước cần thực hiện một HUD
- Tạo các thành phần HUD
- Định vị các thành phần của HUD sử dụng Hệ tọa độ GUI
- Khai báo và khởi tạo các thành phần thông qua Scripts
- Cập nhật HUD dựa trên World và Events

Vai trò của Head-Up Display

- Thông báo cho người sử dụng về trạng thái của nhân vật, ví dụ như sức khỏe (health), năng lượng (power), số lượng mạng sống của character
- Thông tin cập nhật về khả năng theo thời gian thực mà character tập hợp được
- Timers – thời gian giảm đến 0 hoặc tổng thời gian
- Camera hỗ trợ điều hướng trong một thế giới
- Các mục menu để truy cập, tùy chọn, thay đổi và cài đặt.

Các bước cần thiết để thiết lập một HUD

- Yêu cầu hoặc tạo sản phẩm theo định dạng thích hợp
- Import HUD thông qua **Asset- Import New Asset**
- Tạo một **GUI Texture Game Object** và liên kết câc sprite với texture object
- Vị trí các thành phần của HUD trong giao diện
- Tạo các script để khởi tạo và cập nhật HUD theo trạng thái và sự kiện
- Attach scripts vào objects để thay đổi trạng thái world và sự kiện trigger khi cập nhật giá trị thích hợp trong HUD script

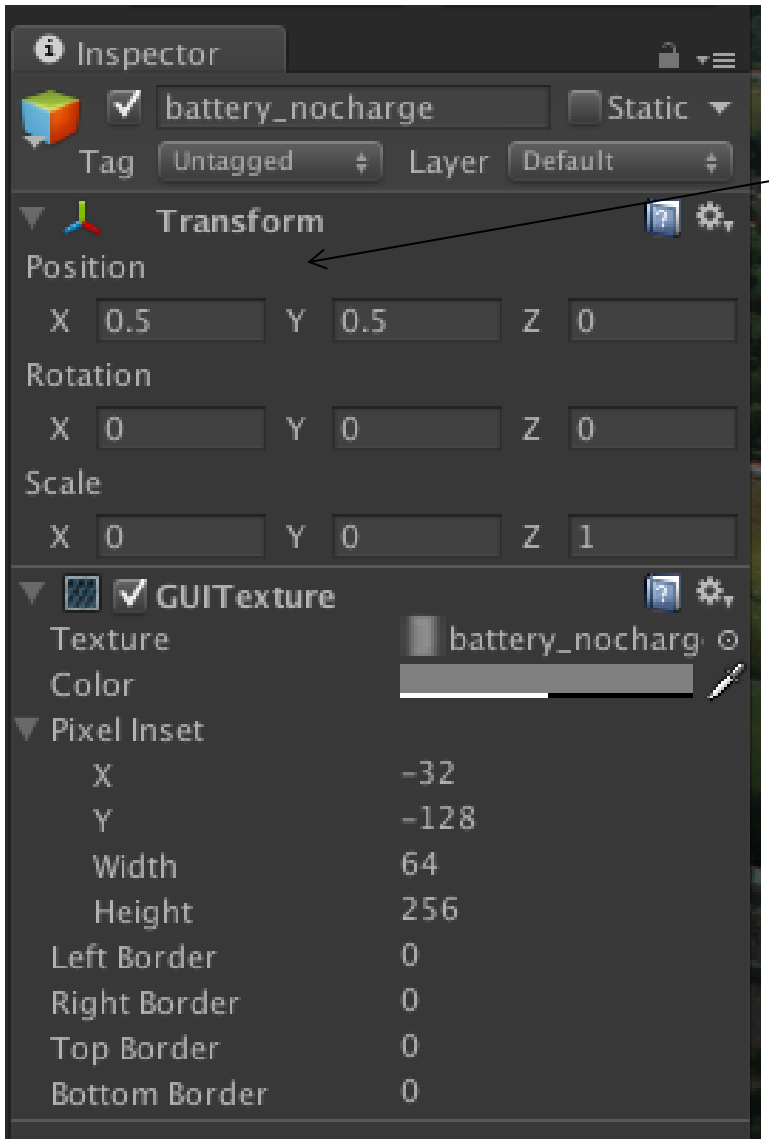
Tạo các thành phần của HUD

- Tạo và custom artwork cho phù hợp với kịch bản đưa ra (images text và kiểu số nếu cần...)
- Sử dụng file.psd để tạo nội dung và giữ lại bản gốc chất lượng cao
- Xuất định dạng đồ họa 32 bit để giữ kênh alpha, trong suốt(.png 32)

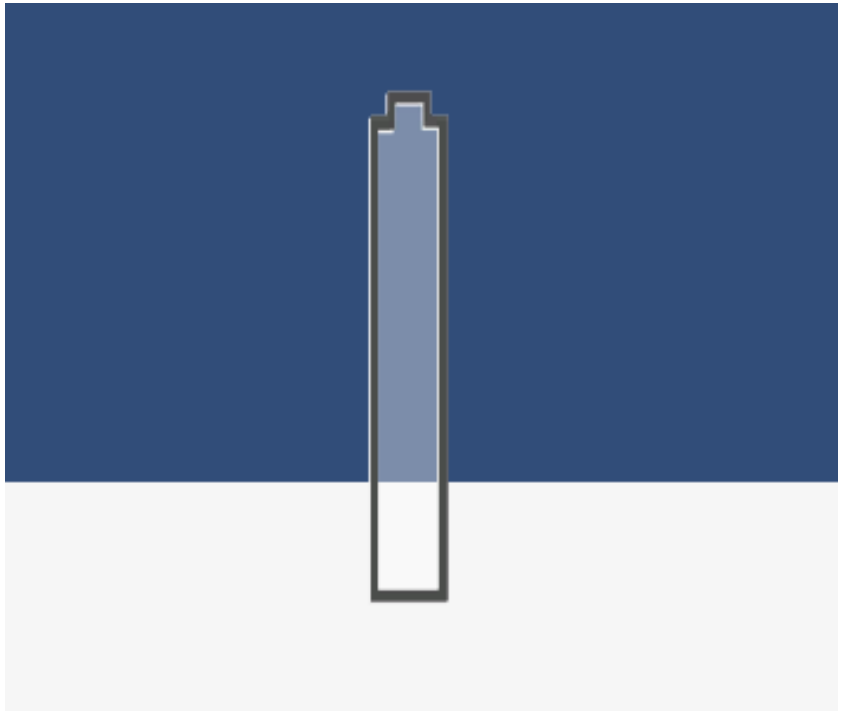
Vị trí các thành phần của HUD

- Thành phần HUD sử dụng có vị trí giống như Camera Overlays. Toạ độ X nằm giữa 0..1 và và toạ độ y nằm giữa 0..1

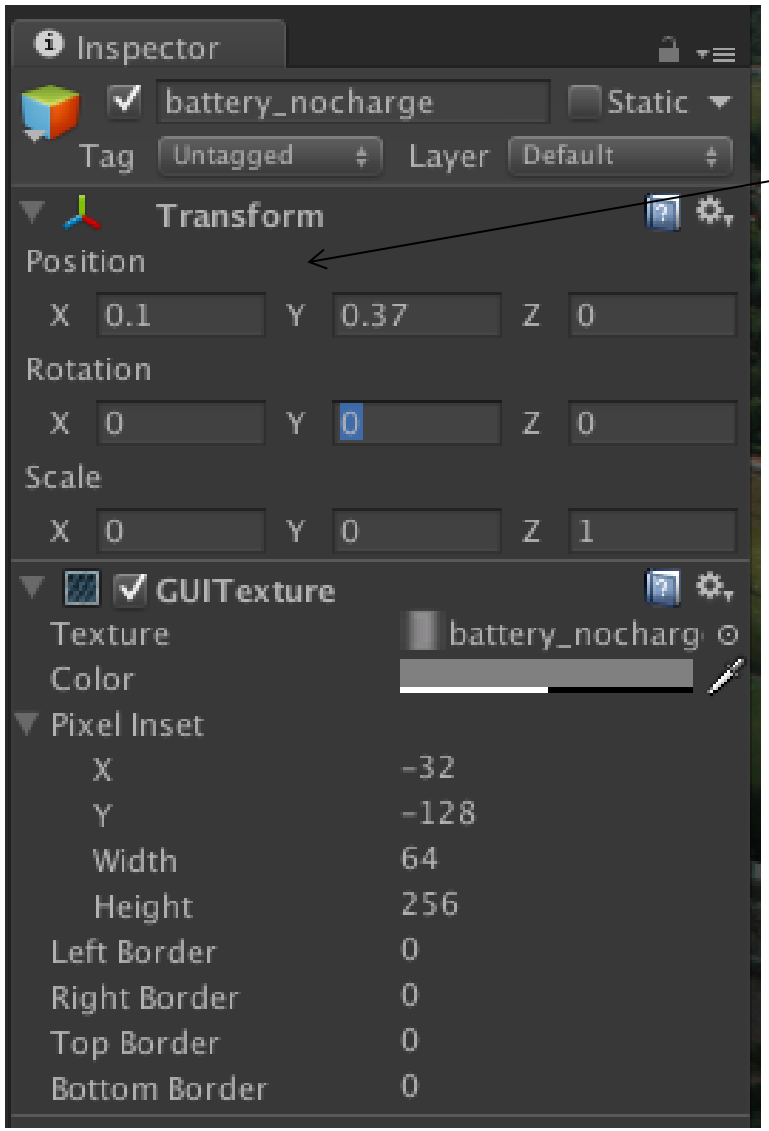
GUI Texture trong Default Position



Texture tâm tại x = 0.5, y = 0.5

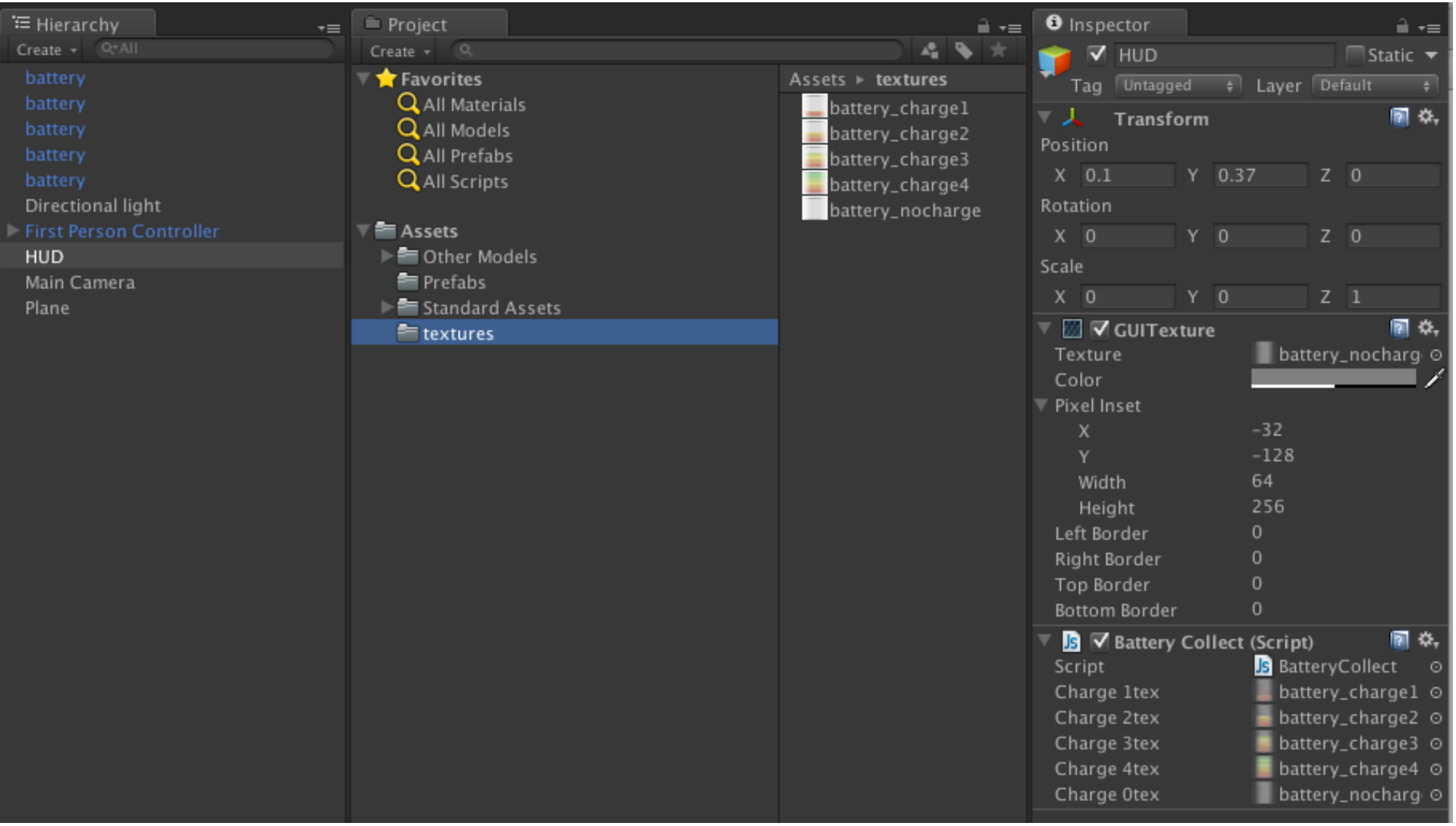


GUI Texture trong Default Position



Texture vị trí trái tại x = 0.1, y = 0.37

Khai báo và khởi tạo HUD



public variables associated with textures

Cập nhật HUD qua Scripts (BatteryCollect.js)

Script này được attached vào GUI Texture GameObject

```
// static variable is accessible by other scripts i.e. its
// variable scope is global
static var charge:int = 0;

var charge1tex:Texture2D;
var charge2tex:Texture2D;
var charge3tex:Texture2D;
var charge4tex:Texture2D;
var charge0tex:Texture2D;

// initialise GUI texture to false (don't display it)
function Start(){

        guiTexture.enabled = false;
          charge = 0;

}
```

Cập nhật HUD qua Scripts

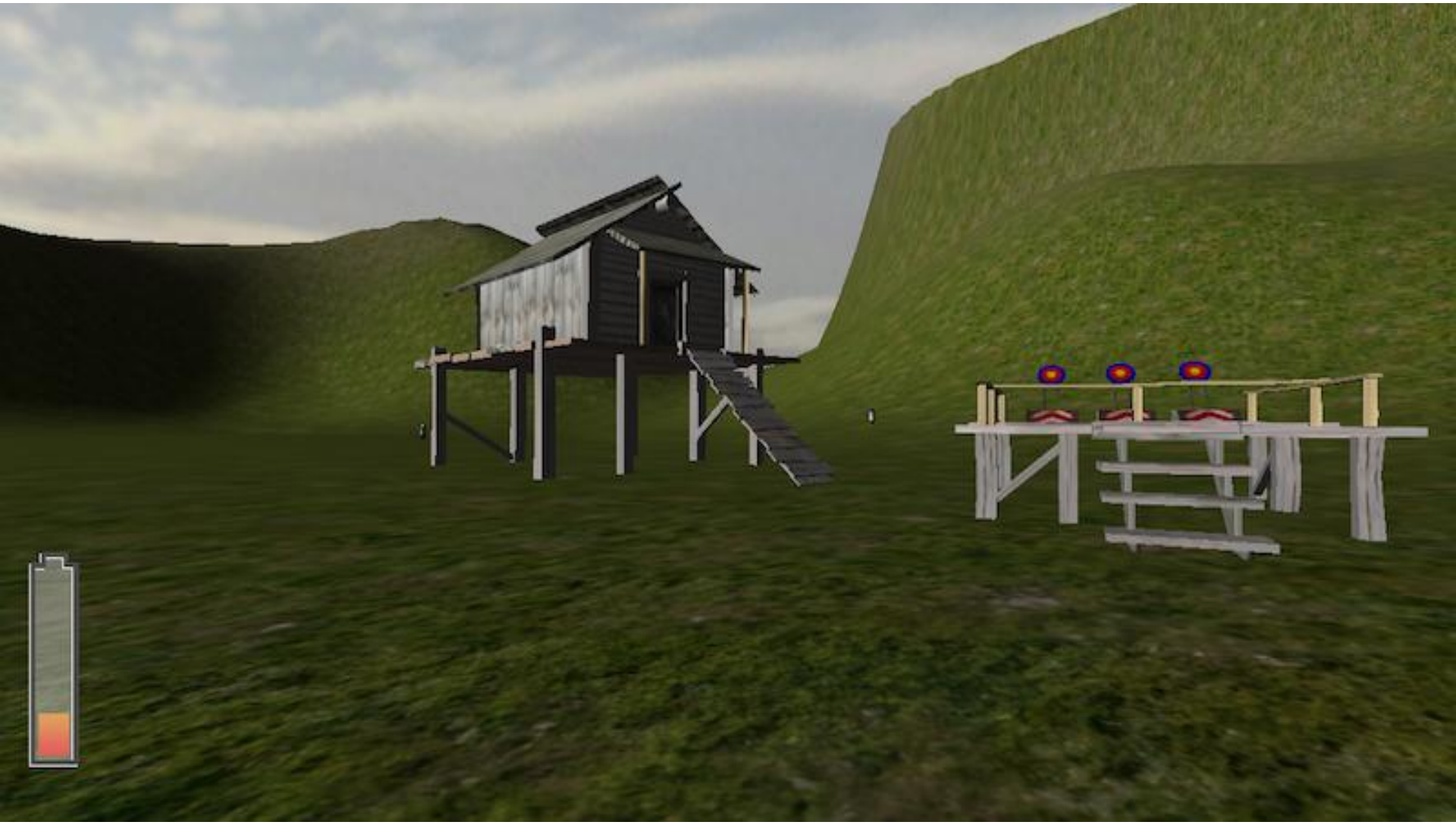
```
/* the Update function checks status of charge variable
which is increased via an external script
each time the player collides (collects) with a battery
*/
function Update () {
        /*
          first battery collected
          assign the first texture image to guiTexture
          and enable (display) texture
        */

        if(charge == 1){

                guiTexture.texture = charge1tex;
                guiTexture.enabled = true;
                }
        // display subsequent textures to indicate power
collected
else if(charge == 2){
        guiTexture.texture = charge2tex;
} // etc.
```

Cập nhật HUD qua Scripts (PlayerCollisions.js)

script này được attach vào First Person Controller

```
function Start () {

}

function Update () {

}

function OnTriggerEnter(collisionInfo : Collider){

if(collisionInfo.gameObject.tag == "battery"){

BatteryCollect.charge++;
Destroy(collisionInfo.gameObject);
}

}
```

Ví dụ về Island HUD



Sound

- **Audio Listener:**
  - Giống như một thiết bị Microphone. Nó nhận đầu vào từ bất cứ một nguồn âm thanh nào trong Scenes và các âm thanh thông qua máy tính.
  - Đối với hầu hết các ứng dụng, nó là ý nghĩa nhất để gắn tai nghe lên Main Camera

Sound

- **Audio Listener:**
  - Ranh giới/độ ảnh hưởng âm thanh nghe của **Reverb Zone** được áp dụng cho toàn Scene thì có thể nghe thấy âm thanh trên bất cứ địa điểm nào của Scene.

Sound

- **Properties:** Phải thêm âm thanh vào Scene và chỉnh sửa thuộc tính của âm thanh bên Inspector View.
  - **Audio Clip:** Chọn file âm thanh cho Scene.
  - **Mute:** Tắt âm thanh cho Scene.
  - **Bypass Effects:** Lọc nhanh hiệu ứng “by-pass” to audio source. Một cách dễ dàng nhất để bật/tắt hiệu ứng(effect)
  - **Play on Wake:** Nếu enable, âm thanh sẽ được chạy ngay khi ra mắt Scene. Nếu để Disable, khi cần chạy âm thanh chúng ta phải gọi phương thức/chức năng **Play()** từ Script

Sound

- **Priority:** Xác định độ ưu tiên của Audio Source trong số tất cả các Audio Source có trong Scene.
  - Priority = 0: Rất quan trọng. Sử dụng ở mức 0 cho bài nhạc để tránh bị thường xuyên trao đổi.
  - Priority =256: Độ quan trọng thấp nhất(Độ ưu tiên ở mức thấp nhất).
  - Mặc định(default) = 128.

Sound

- **Volumn:**
  - Làm thế nào để âm thanh lớn trên một bộ phận của thế giới Scene từ Audio Source.
- **Pitch:**
  - Số dùng thay đổi tốc độ của âm thanh. Tốc độ bằng 1 là ở mức chạy bình thường.
- **3D Source Setting:**
  - Cài đặt file được áp dụng cho Audio source nếu Audio Clip là một file âm thanh 3D.

Sound

- **Pan Lever:** Cài đặt để máy 3D có được hiệu ứng từ Audio source.
- **Spread:** Cài đặt góc ảnh hưởng tới âm thanh 3D Stereo nếu Audio Clip là một âm thanh 3D.
- **Doppler Lever:** Xác định bao nhiêu hiệu ứng âm thanh Dopper sẽ được áp dụng cho Audio Source(Nếu cài đặt là 0 thì sẽ không có hiệu ứng nào được áp dụng).

Sound

- **Min Distance:** Với Min Distance, âm thanh sẽ đạt mức to nhất có thể. Ngoài Min Distance, nó sẽ bắt đầu suy yếu đi. Tăng MinDistance của âm thanh để được âm thanh lớn hơn trong thế giới 3D và giảm MinDistance để cho âm thanh nhỏ hơn.
- **Max Distance:** Khoảng cách mà âm thanh dừng suy giảm. Sau khoảng cách này nó sẽ tạm ngừng không cho MinDistance tăng nữa. Tức là đây là giá trị Max mà MinDistance sẽ đạt tới.

Sound

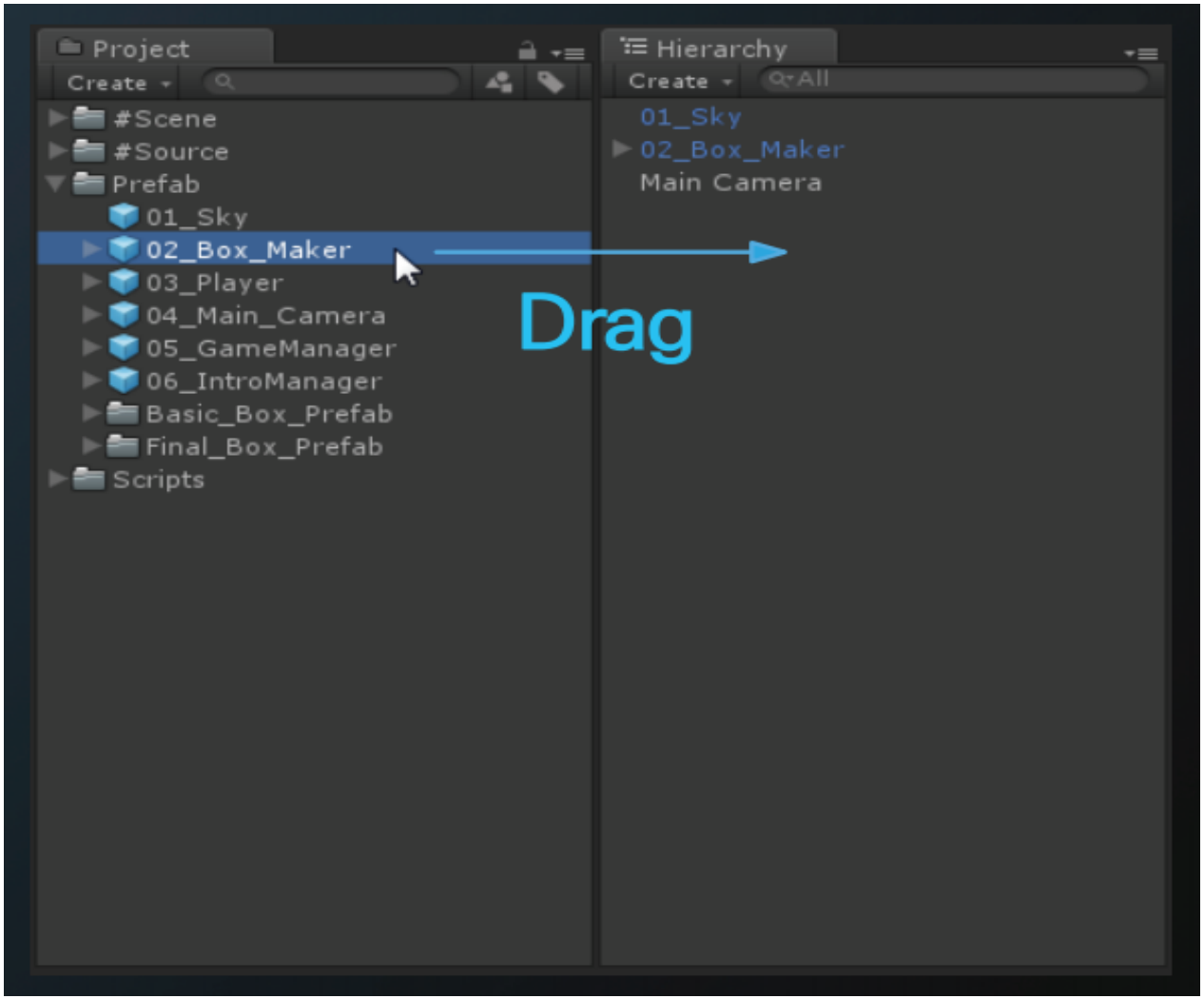
- **Rolloff Mode:** Làm thế nào nhanh chóng mất dần âm thanh. Giá trị cao hơn, gần gũi hơn Listener có được trước khi nghe âm thanh.(Điều này được xác định bởi một Graph).
  - **Logarithmic Rolloff:** Âm thanh là lớn khi bạn ở gần nguồn âm thanh. Nhưng khi chúng ta nhận được từ các đối tượng nó sẽ giảm đi nhanh chóng một cách đáng kể.
  - **Linear Rolloff:** Càng xa nguồn âm thanh, bạn nghe thấy càng ít.

DEMO

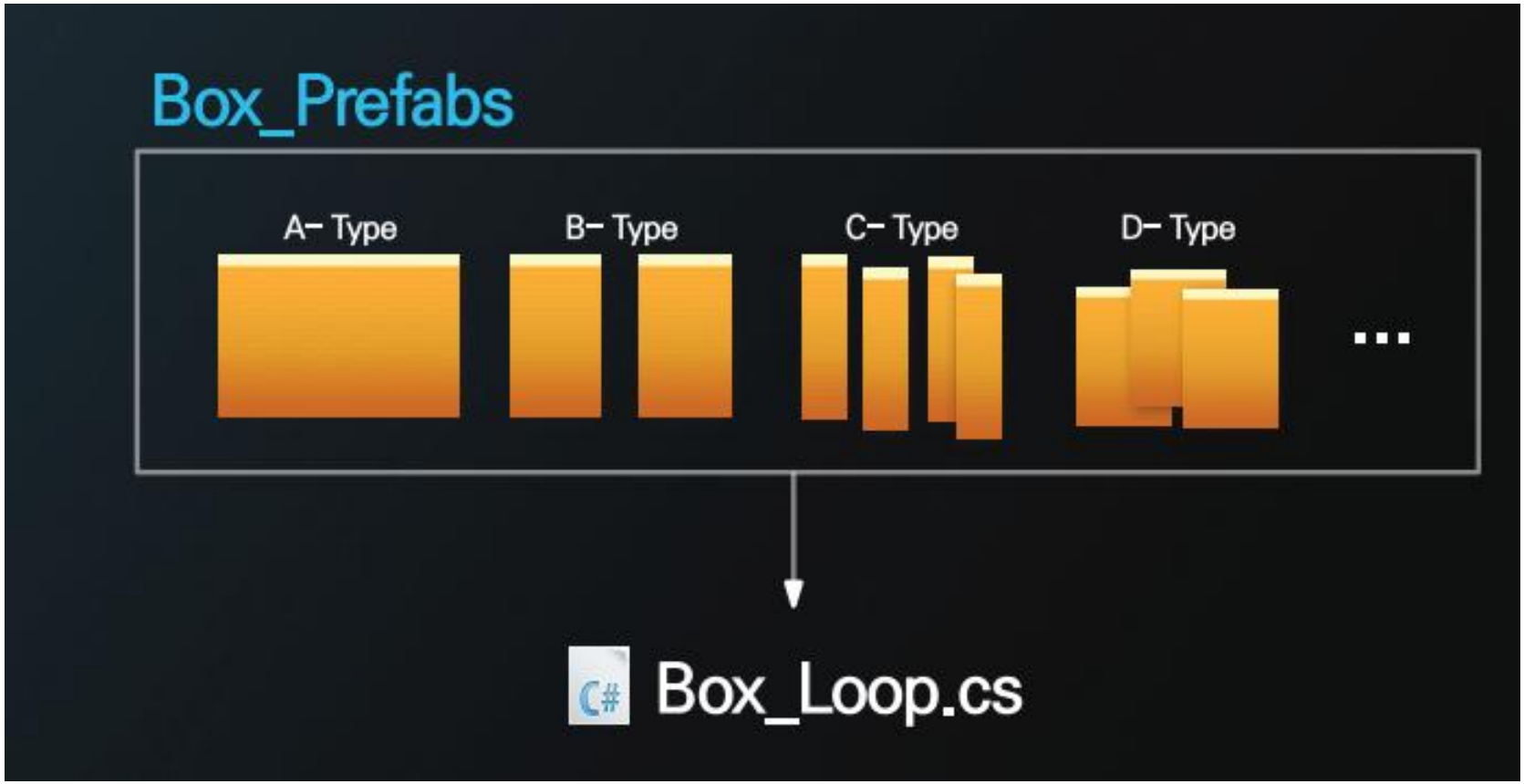
Xây dựng scene Box_Maker:

Background và chướng ngại vật chuyển động không cùng tốc độ

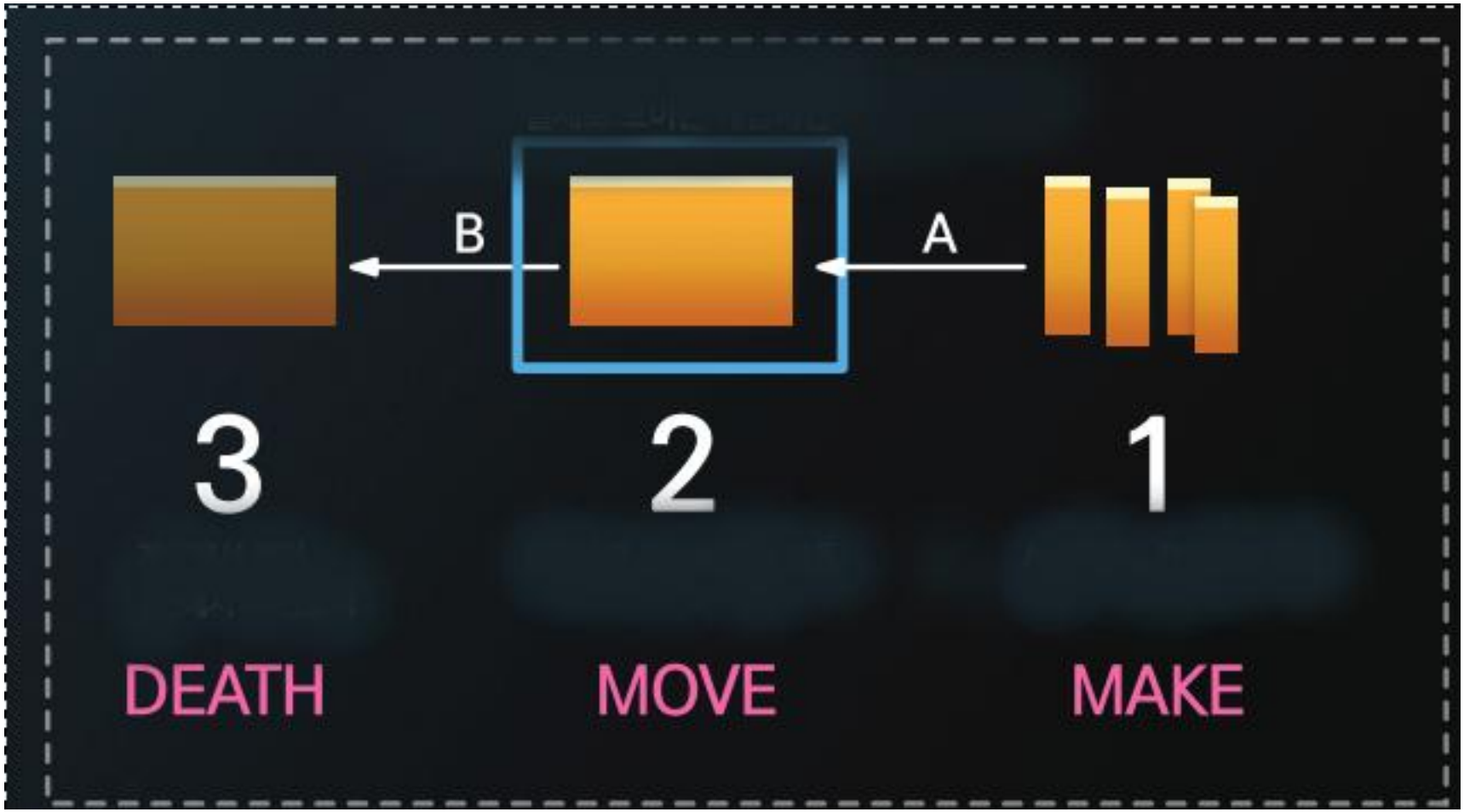
Kéo Box_Maker vào Hierachy



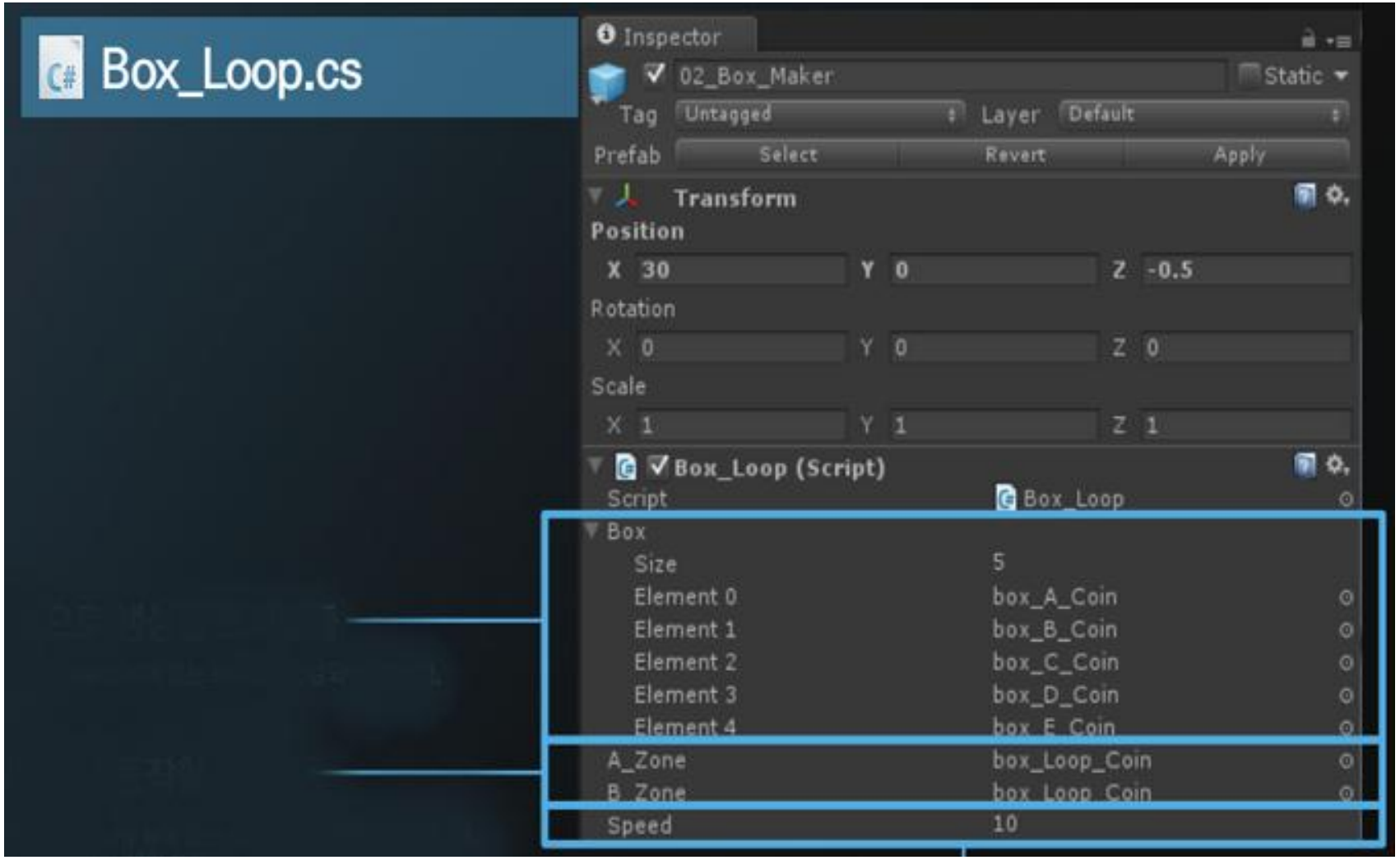
Sử dụng các kiểu chướng ngại vật A, B, C, D



Các chướng ngại vật xen kẽ nhau



Bố trí các Box Coin



Cài đặt Box_Loop.cs

- Khai báo mảng Box
  - public GameObject[] Box;
- Khai báo game object kiểu A
  - public GameObject A_Zone;
- Khai báo game object kiểu B
  - public GameObject B_Zone;
- Khai báo tốc độ di chuyển của chướng ngại vật
  - public float Speed = 5f;

Cài đặt Box_Loop.cs

- Hàm tạo chướng ngại vật
  - public void MAKE()
- Gán vị trí chướng ngại vật B sang A
  - B_Zone=A_Zone;
- Lấy ngẫu nhiên một vị trí
  - int a = Random.Range(0,5);
- Khởi tạo chướng ngại vật A
  - A_Zone = Instantiate(Box[a], new Vector3(30,0,0), transform.rotation) as GameObject;

Cài đặt Box_Loop.cs

- Hàm di chuyển chướng ngại vật
  - public void MOVE(){
- Di chuyển vị trí chướng ngại vật A đến một vị trí xác định theo một tốc độ nhất định
  - A_Zone.transform.Translate(Vector3.left * Speed *Time.deltaTime, Space.World);
- Di chuyển vị trí chướng ngại vật B đến một vị trí xác định theo một tốc độ nhất định
  - B_Zone.transform.Translate(Vector3.left * Speed *Time.deltaTime, Space.World);

Cài đặt Box_Loop.cs

- Hàm di chuyển chướng ngại vật
  - public void MOVE(){
- Nếu trùng vị trí với chướng ngại vật trước thì huỷ chướng ngại vật trước đó
  - if(A_Zone.transform.position.x<=0){
    DEATH();}

Cài đặt Box_Loop.cs

- Hàm huỷ chướng ngại vật
  - public void DEATH()
- Gọi hàm huỷ chướng ngại vật B
  - Destroy(B_Zone);
- Gọi hàm tạo chướng ngại vật
  - MAKE();

Kết luận

- Heads Up Displays là một khía cạnh quan trọng của kinh nghiệm người dùng và nâng cao khả năng sử dụng.
- HUDs trong Unity 3D dễ cài đặt nhưng thường yêu cầu nội dung artwork HUD phải được tạo rất tốn thời gian.
- artwork được import như một new asset và kết hợp với một **GUI Texture GameObject** thông qua biến toàn cục.
- Các kịch bản trực tiếp thay đổi các yếu tố HUDs sẽ được gắn vào GUI Texture GameObject.
- Các kịch bản mà nên sự kiện giao tiếp với các kịch bản của GUI Texture thông qua một biến tĩnh (**static variable**)
- Kịch bản này được attach vào First Person Controller

Fpt University

FPT POLYTECHNIC

THANK YOU!

www.poly.edu.vn