BÁO CÁO THỰC HÀNH UNITY

Coordinate System – Local/World/Screen Space – WorldToScreen (Graphics Pipeline)

Họ và tên: Phạm Như Thắng
MSSV: BCS230080
Lớp: 23SODE
Ngày thực hiện: 14/01/2026
Unity Version: 3.15.4
Nền tảng: Unity Hub

1. Mục tiêu bài thực hành

Sau khi hoàn thành bài tập, sinh viên đạt được các mục tiêu sau:

- Hiểu cách Unity chuyển đổi từ không gian 3D sang hiển thị 2D thông qua các bước chính của Graphics Pipeline.
- Sử dụng hệ tọa độ Cartesian (x, y, z) trong không gian ba chiều.
- Phân biệt đặc trưng của Left-Handed Coordinate System trong Unity.
- Phân biệt Local Space, World Space và Screen Space.
- Sử dụng Camera để chuyển đổi từ World Space sang Screen Space.

2. Kiến thức nền tảng

2.1. Hệ tọa độ Cartesian trong Unity

Unity sử dụng hệ tọa độ không gian ba chiều với ba trục vuông góc:

- Trục X biểu diễn hướng trái – phải.
- Trục Y biểu diễn hướng dưới – trên.
- Trục Z biểu diễn hướng sau – trước (forward).

2.2. Tổng quan Graphics Pipeline

Trong Unity, một điểm 3D được hiển thị lên màn hình phải đi qua các bước chuyển đổi chính:

1. Local Space (tọa độ gắn với đối tượng).
2. World Space (tọa độ trong không gian Scene).
3. View/Camera Space (tọa độ tương đối so với Camera).

4. Clip Space và Normalized Device Coordinates (sau phép chiếu).
5. Screen Space (tọa độ pixel trên màn hình).

Việc thay đổi Camera hoặc các thông số chiếu sẽ ảnh hưởng trực tiếp đến kết quả hiển thị 2D.

PHẦN A – COORDINATE SYSTEM & WORLD SPACE (20%)

A1. Tạo Cube tại vị trí (2, 1, 5)

Một đối tượng Cube được tạo trong Scene và thiết lập Transform Position:

- X = 2
- Y = 1
- Z = 5

Vị trí này được xác định trong World Space, tức là dựa trên hệ trục chung của toàn Scene.

A2. Quan sát Gizmos trong Scene View

Gizmos được bật trong Scene View để quan sát các trục tọa độ:

- Trục X (màu đỏ)
- Trục Y (màu xanh lá)
- Trục Z (màu xanh dương)

(Chèn hình minh họa Scene View thể hiện đầy đủ ba trục tọa độ)

A3. Trả lời câu hỏi

Trục nào hướng lên trên trong Unity?

Trục Y là trục hướng lên trên trong Unity.

Trục nào hướng về phía Camera?

Trong thiết lập mặc định, Camera thường đặt ở phía Z âm và nhìn về hướng Z dương. Do đó, hướng về phía Camera là chiều âm của trục Z.

PHẦN B – LEFT-HANDED COORDINATE SYSTEM (15%)

B1. Xoay Cube với Rotation Y = 90

Đối tượng Cube được xoay quanh trục Y một góc 90 độ.

B2. Quan sát và nhận xét

Cube quay theo chiều nào?

Khi tăng Rotation Y lên giá trị dương (90 độ), Cube quay theo chiều kim đồng hồ nếu quan sát từ phía trên (theo hướng trục Y dương nhìn xuống).

Điều này thể hiện Left-Handed Coordinate System như thế nào?

Unity sử dụng quy ước trong đó trục Z hướng về phía trước. Khi áp dụng phép quay quanh trục Y dương, hướng forward của đối tượng thay đổi từ trục Z dương sang trục X dương, phù hợp với cách xác định hướng trong hệ tọa độ Left-Handed.

PHẦN C – LOCAL SPACE VÀ WORLD SPACE (25%)

C1. Tạo Empty GameObject “Parent”

Một Empty GameObject tên “Parent” được tạo tại vị trí (5, 0, 0) trong World Space.

C2. Thiết lập quan hệ cha – con

Cube được đặt làm con của Parent và thiết lập Local Position của Cube là (0, 2, 0).

C3. Ghi nhận vị trí của Cube

- Local Position của Cube: (0, 2, 0)
- World Position của Cube: (5, 2, 0)

World Position của Cube bằng tổng World Position của Parent và Local Position của Cube (trong trường hợp Parent không xoay và có Scale mặc định).

C4. Di chuyển Parent sang (8, 0, 0)

Local Position của Cube có thay đổi không?

Local Position của Cube không thay đổi vì nó được xác định trong hệ tọa độ của Parent.

World Position của Cube thay đổi như thế nào?

World Position của Cube thay đổi theo Parent và trở thành (8, 2, 0).

PHẦN D – GRAPHICS PIPELINE (20%)

D1. Di chuyển Camera

Camera được di chuyển dọc trục Z từ -10 đến -3, tiến gần hơn về phía đối tượng trong Scene.

D2. Thay đổi thông số Camera và phân tích

Vì sao object trông to hoặc nhỏ hơn dù không đổi vị trí?

Khi Camera tiến gần hoặc lùi xa đối tượng, hoặc khi thay đổi Field of View, ma trận chiếu của Camera thay đổi. Điều này làm thay đổi kích thước biểu kiến của đối tượng trên màn hình dù vị trí trong World Space không đổi.

Vì sao object có thể biến mất khỏi màn hình?

Object có thể biến mất khi nằm ngoài vùng nhìn của Camera hoặc khi giá trị Near Clip Plane được đặt lớn hơn khoảng cách từ Camera đến object, khiến object bị cắt khỏi vùng hiển thị.

PHẦN E – SCREEN SPACE (20%)

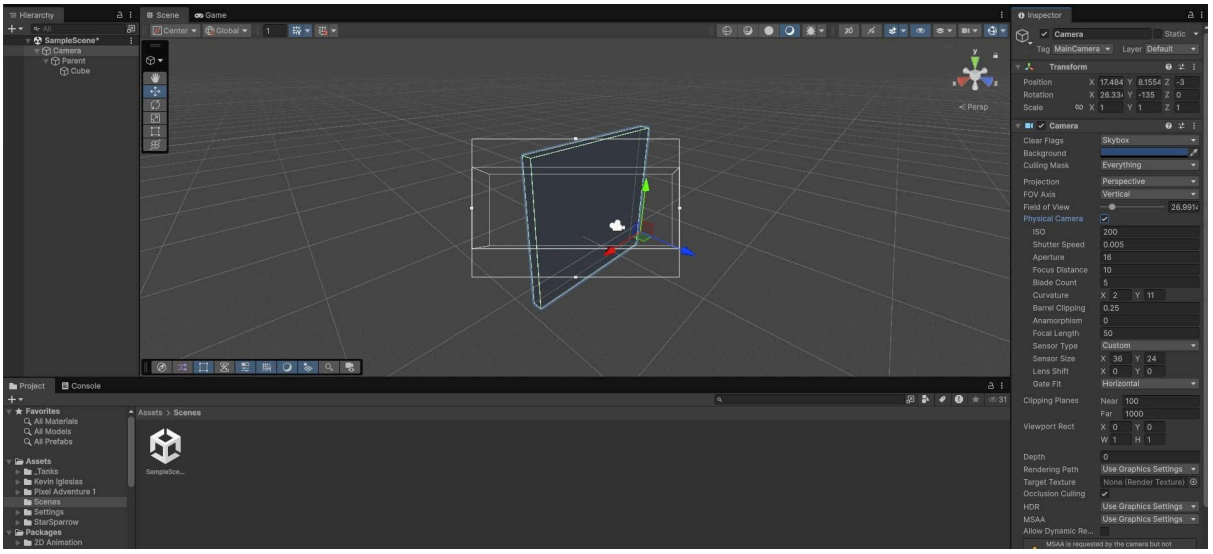
E1–E2. Chuyển đổi World Space sang Screen Space

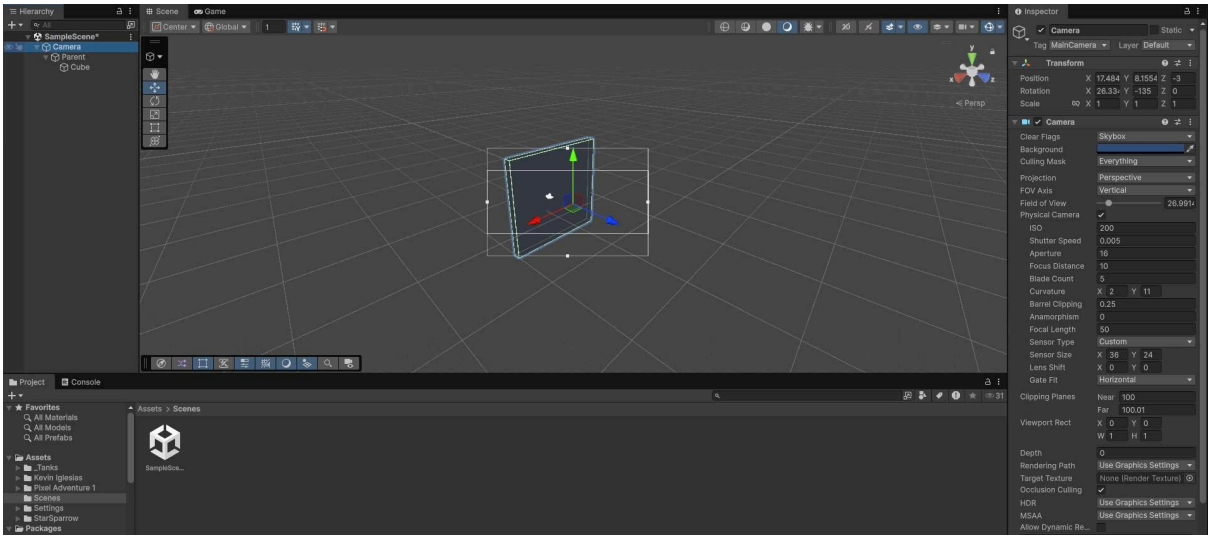
Script WorldToScreen.cs được gắn vào Cube để in ra tọa độ Screen Space của đối tượng mỗi frame bằng hàm `WorldToScreenPoint()`.

E3. Ghi nhận Screen Position

- Screen Position khi Cube ở giữa màn hình: (……, ……, ……)
- Screen Position khi Cube ở góc dưới bên trái màn hình: (……, ……, ……)

Các giá trị này phụ thuộc vào độ phân giải Game View và vị trí Camera.





E4. Trả lời câu hỏi

Gốc tọa độ của Screen Space nằm ở đâu?

Gốc tọa độ Screen Space nằm ở góc dưới bên trái của màn hình Game.

Screen Space khác World Space như thế nào?

World Space là hệ tọa độ ba chiều dùng để mô tả vị trí đối tượng trong Scene, không phụ thuộc độ phân giải màn hình. Screen Space là hệ tọa độ hai chiều theo pixel dùng để hiển thị đối tượng lên màn hình và phụ thuộc vào độ phân giải hiển thị.

3. Kết luận

Qua bài thực hành, sinh viên đã nắm được cách Unity sử dụng hệ tọa độ ba chiều, cách phân biệt Local Space, World Space và Screen Space, cũng như vai trò của Camera trong Graphics Pipeline khi chuyển đổi từ không gian 3D sang hiển thị 2D.

4. Nhận xét cá nhân

Thông qua bài thực hành này, sinh viên hiểu rõ hơn cách Unity xử lý tọa độ và Camera. Việc trực tiếp quan sát sự thay đổi của đối tượng khi thay đổi Camera và sử dụng WorldToScreenPoint giúp liên hệ rõ ràng giữa lý thuyết Graphics Pipeline và kết quả hiển thị thực tế. Kiến thức này là nền tảng quan trọng cho việc phát triển game và các ứng dụng đồ họa 3D trong Unity.