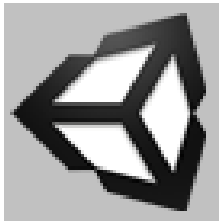
LẬP TRÌNH GAME VỚI UNITY

Bài 4: Player Characters

ThS. Thái Duy Quý

Đà Lạt, Tháng 03 năm 2016

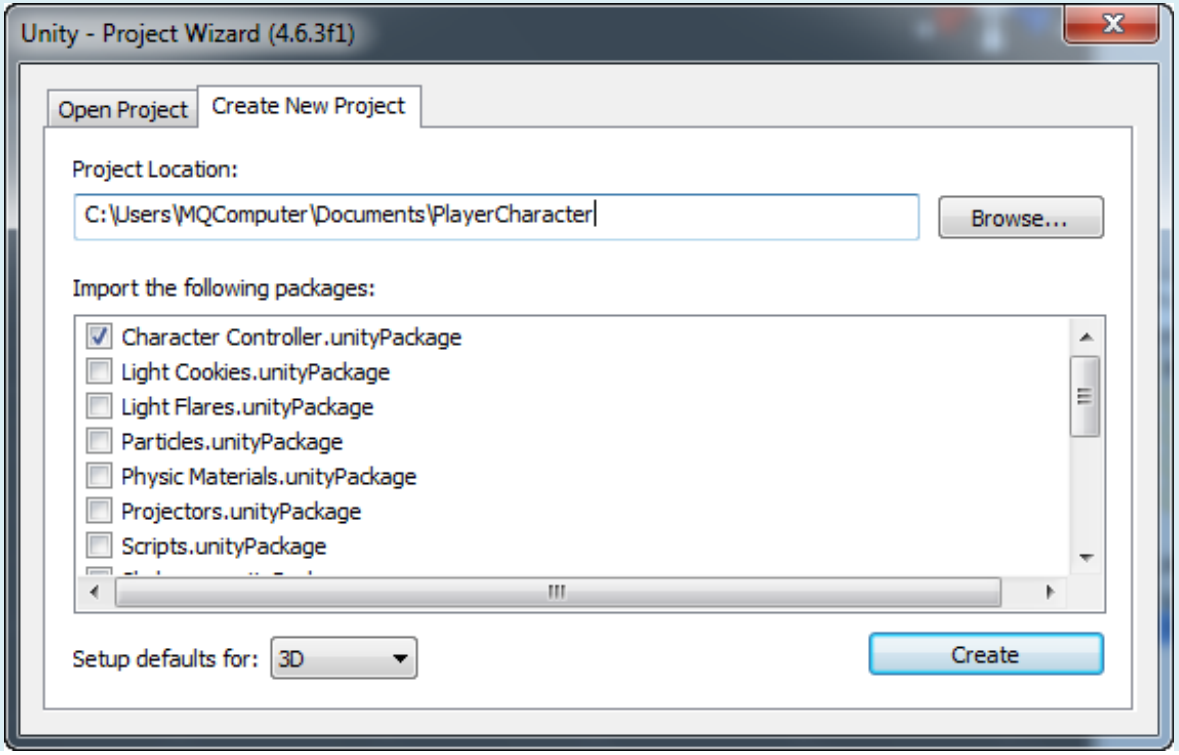


Nội dung

- Tạo dự án với Character Controller
- Làm việc với Inspector
- Prefab
- Đối tượng First person
- Graphics
- Main Camera
- Scripts

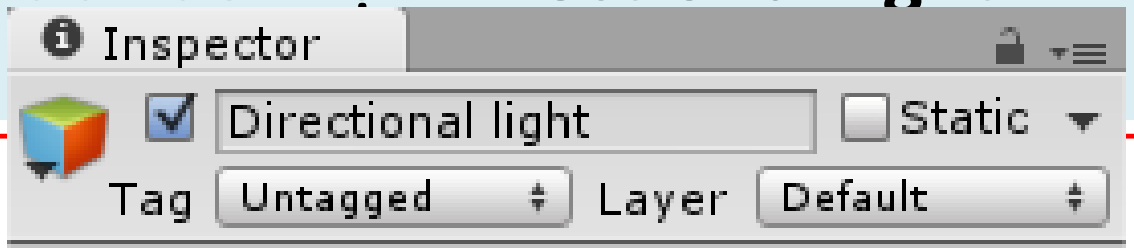
Tạo dự án

- Để thực hiện được minh họa, sinh viên tạo dự án với gói Character Controller (hình)



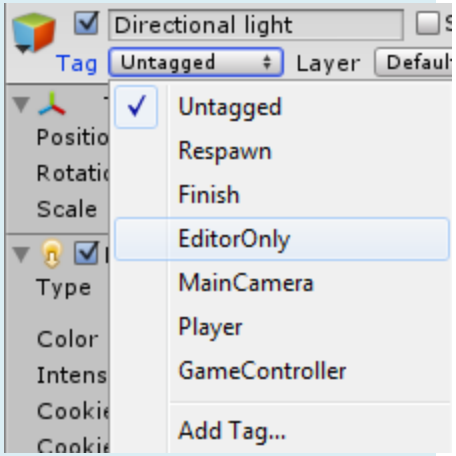
Làm việc với Inspector

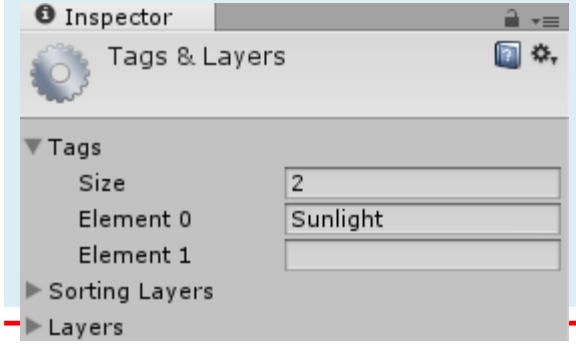
- Inspector biểu diễn trạng thái của Game.
- Trên đầu Inspector có 3 chức năng:
  - Nút 3 màu: dùng để đánh dấu đối tượng trong Game.
  - Tên và trạng thái đối tượng Game
  - Tag: quản lý thẻ
  - Layer: Quản lý lớp của Game.
- Ví dụ: đưa vào một **Directional light:**

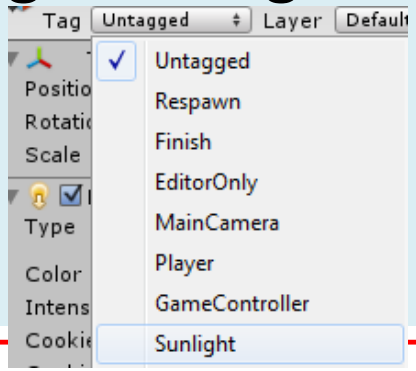


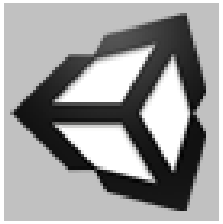
Tags

- Tags là một từ khóa đơn giản, dùng để gán cho Game Object, để quản lý đối tượng.
- Có thể thêm Tag bằng 2 cách:
  - Dùng Tag sẵn có trong Tag Manager.
  - Thêm vào một Tag mới bằng cách sử dụng chức năng Add Tag







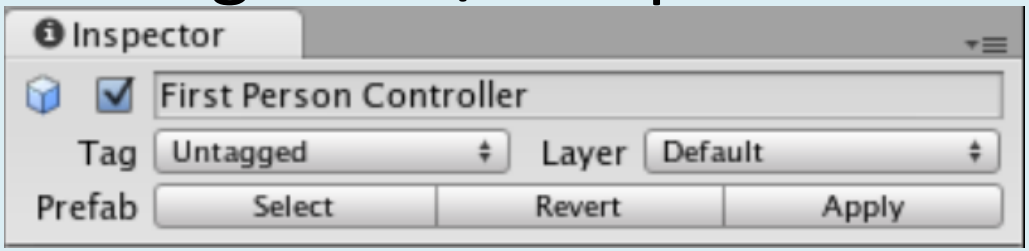


Layer

- Layer dùng để gom nhóm các đối tượng để dễ quản lý.
- Các layer thường được áp dụng khi ta muốn quản lý một tập hợp các đối tượng hoặc quản lý khi dùng mặt nạ.

Prefabs và Inspector

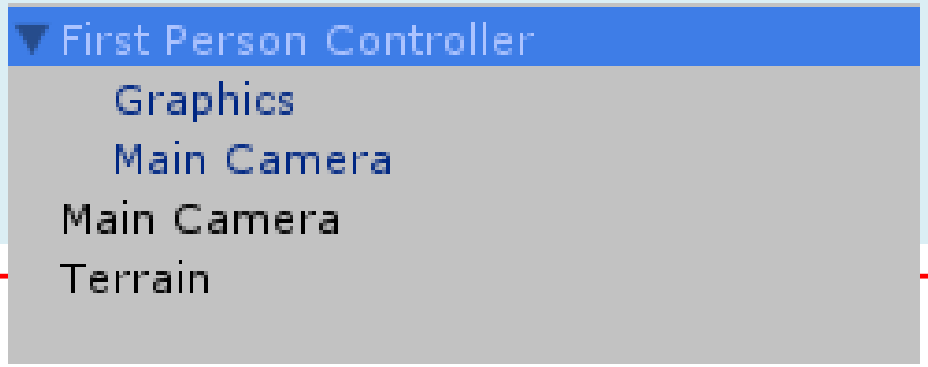
- Nếu một Game Object được lựa chọn là Prefab, khi đó giao diện Inspector sẽ như sau:



- Bên cạnh Tag và Layer, còn có thêm các nút:
  - **Select**: Chọn lựa prefab này để áp dụng.
  - **Revert**: Đảo lại các chức năng đã sửa.
  - **Apply**: Thay đổi các thiết lập lên toàn bộ các Game Object của Prefab.

Đối tượng First Person Controller

- Đối tượng **First Person Controller (FPC)** khi đưa vào màn hình, trên Hirachy sẽ nhìn thấy hai thành phần:
  - Thành phần gốc: FPC
  - Thành phần con:
    - ✓**Graphic:** Là Capsule
    - ✓**Camera**: Là thành phần đi kèm

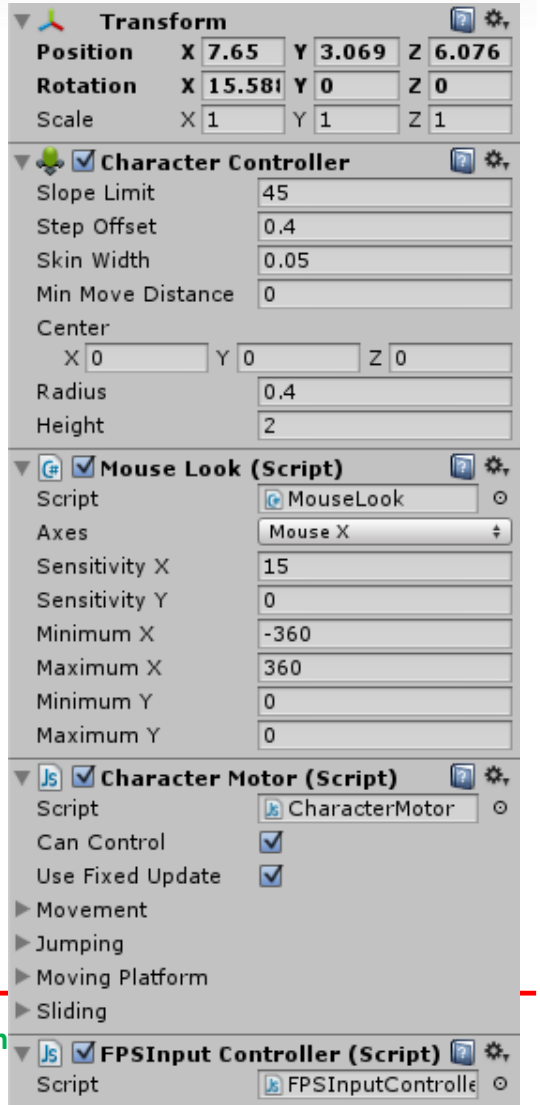


Mối quan hệ Cha – con

- Khi một đối tượng nằm trong một đối tượng khác, ta có mối quan hệ Parent-Child:
  - Giá trị của vị trí và phép quay của đối tượng con phụ thuộc đối tượng cha. Ví dụ: Vị trí đối tượng cha là (500,500,10) nhưng đối tượng con là (0,0,0)
  - Khi quay, hoặc dịch đối tượng con thì đối tượng cha không ảnh hưởng.
  - Nhưng ngược lại thì đối tượng con phải theo.
  - Muốn tìm kiếm đối tượng con, dùng phím F

Object 1: First Person Controller

- Khi đối tượng cha FPC được chọn, ta có các thành phần:
  - Transform
  - Character Controller
  - Mouse Look (Script).
  - Character Moto (Script)
  - FPSInput (Script)

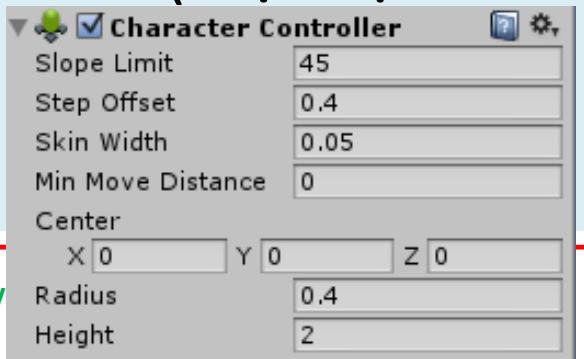


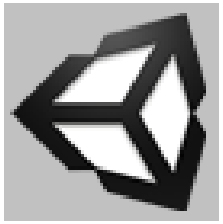
Transform

- Là một thuộc tính mặc định, đối tượng Transform cho phép thiết lập vị trí, quay và co dãn của đối tượng.

Character Controller

- Đối tượng này hoạt động như một Collider (là một thành phần giúp xác định va chạm) được thiết kế giúp nhân vật di chuyển trong môi trường của game.
- Bao gồm các tham số chính sau:
  - **Height**: Chiều cao của character.
  - **Radius**: Bán kính của character (mặc định bằng đối tượng Graphics).

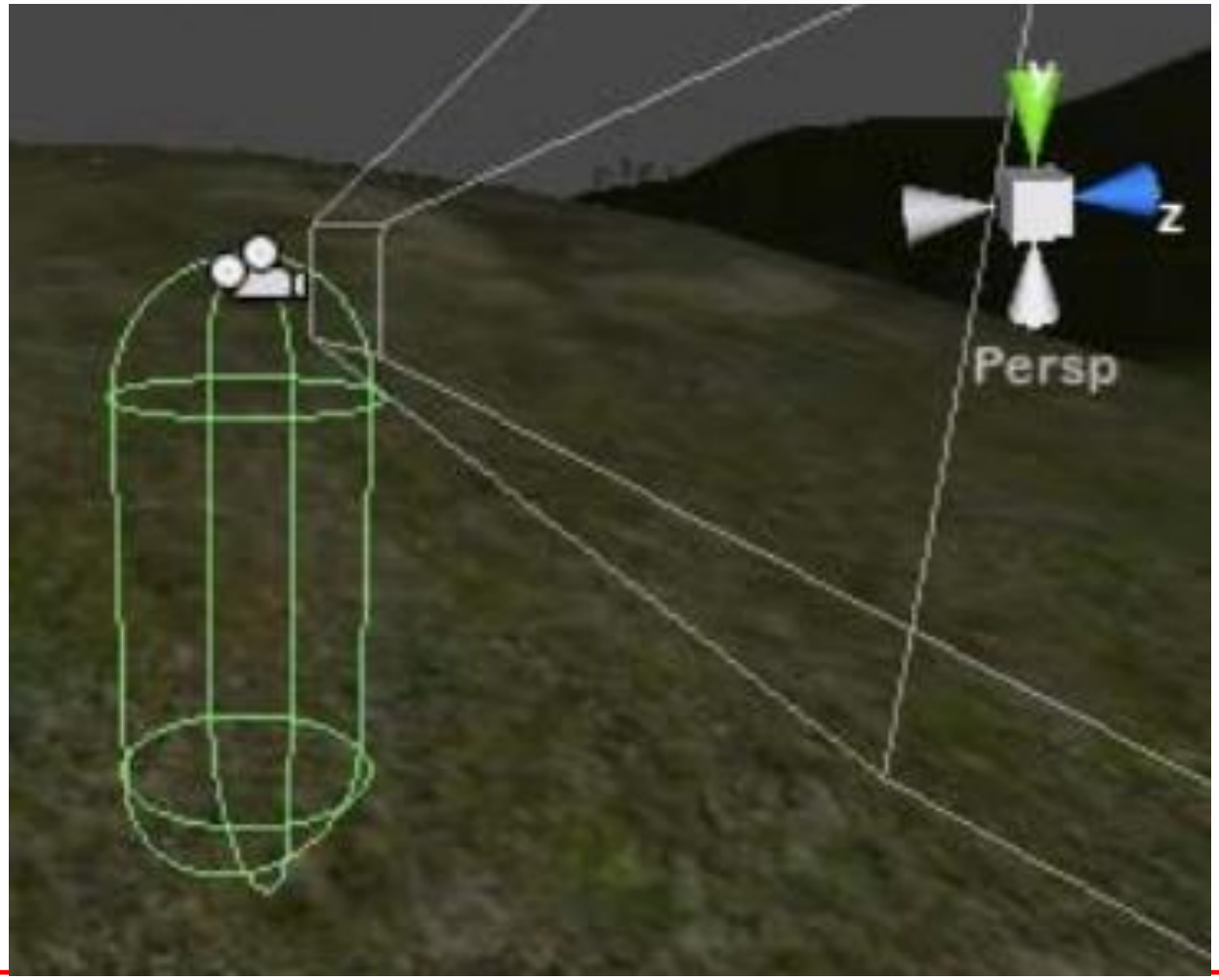




Character Controller

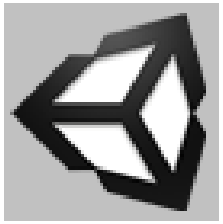
- **Slope Limit**: Thiết lập điểm dừng khi leo lên dốc.
- **Step Offset**: Độ dài bước đi của nhân vật
- **Skin Width**: Độ rộng của lớp bảo vệ nhân vật. Giúp xác định mức độ va chạm.
- **Min Move Distance**: Giúp thiết lập bước đi tối thiểu của nhân vật.
- **Center**: Vị trí trung tâm của nhân vật.

Character Controller



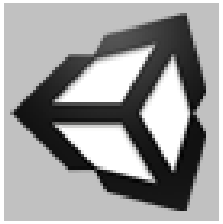
Mouse Look (Script)

- Được viết bằng C#, giúp điều khiển khung nhìn của Camera.
- Bao gồm các biến như sau:
  - **Axes**: Cho phép chọn lựa chiều quay của camera gắn trong nhân vật.
  - **Sensitivity X/Y**: Độ nhạy khi rê chuột.
  - **Minimum X/Maximum X**: Độ quay theo chiều X.
  - **Minimum Y/Maximum Y**: Độ quay theo chiều Y



CharacterMoto (Script)

- Giúp điểu khiển bước đi của nhân vật.
- Bao gồm các thuộc tính như sau:
  - **Movement**: Điều khiển di chuyển nhân vật.
  - **Jumping**: Điều khiển bước nhảy của nhân vật.
  - **Moving Platform**: Thiết lập thông số khi di chuyển lên bục.
  - **Sliding**: Thiết lập chế độ trượt của nhân vật.

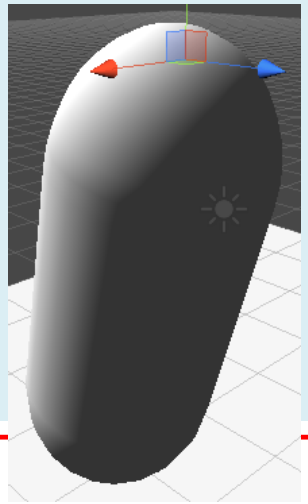


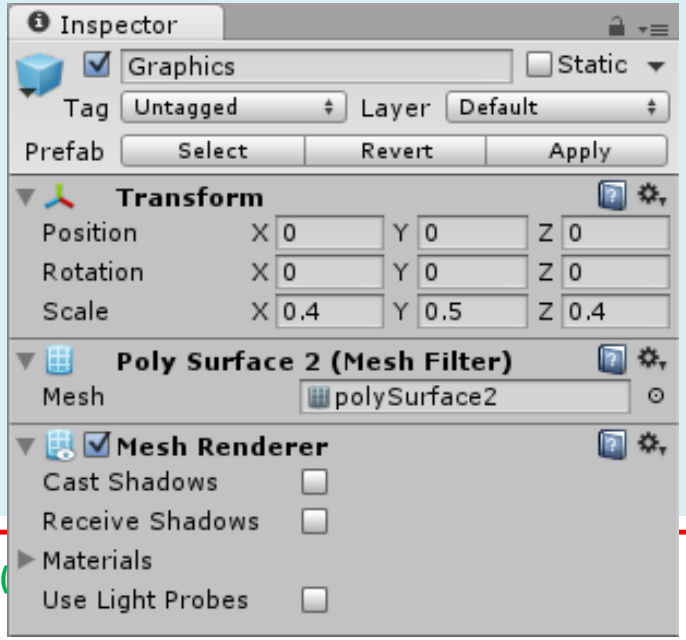
FPSInputController (Script)

- Điều khiển một số thiết bị đầu vào cho nhân vật.

Object 2: Graphics

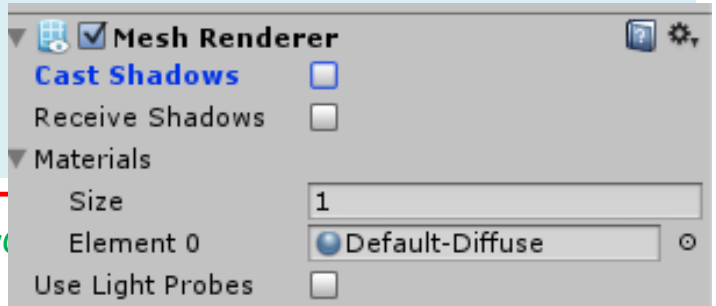
- Thành phần Graphics được tạo nên từ một game object có tên Capsule, dùng để giả lập người chơi.
- Chọn Graphics, và xem trong Inspector, sẽ có các thành phần sau:

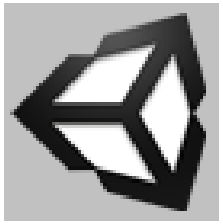




Object 2: Graphics

- **Transform**: thành phần quy định vị trí, chiều quay và độ co dãn của đối tượng. Các giá trị này sẽ lấy đối tượng cha làm chuẩn.
- **Mesh Filter**: là thành phần chứa lưới và các thành phần của lưới, dùng để cấu tạo nên vật thể.
- **Mesh Renderer**: biểu diễn vật thể. Bao gồm:
  - ✓**Cast Shadows**: tạo bóng cho bề mặt.
  - ✓**Receive Shadows**: nhận bóng đổ bề mặt.
  - ✓**Materials**: chất liệu của vật thể.



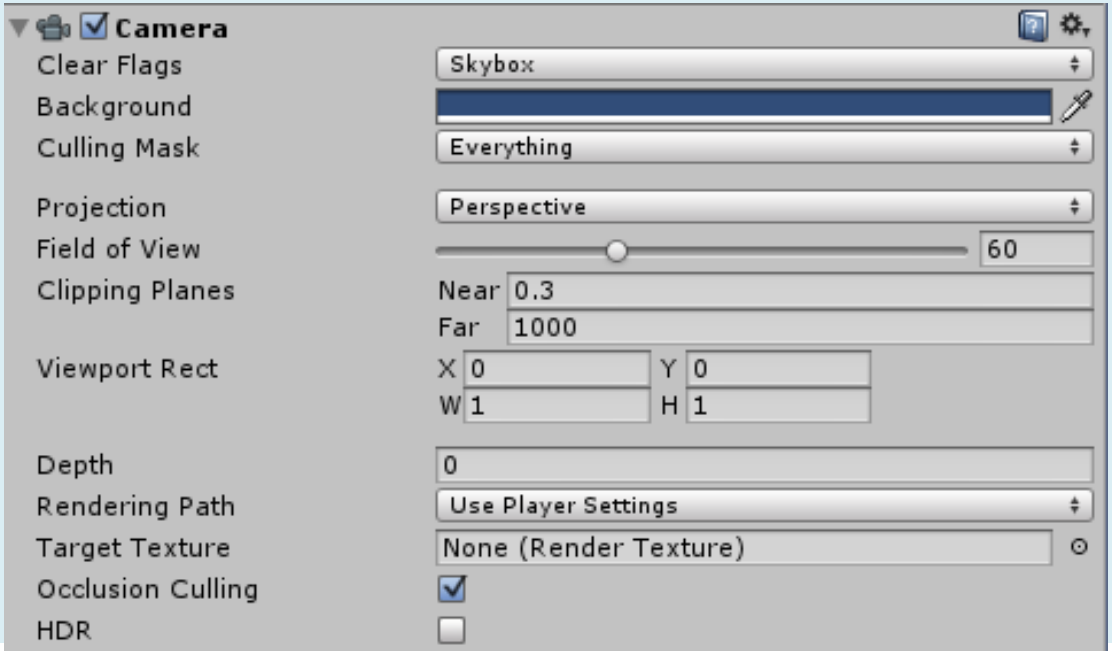


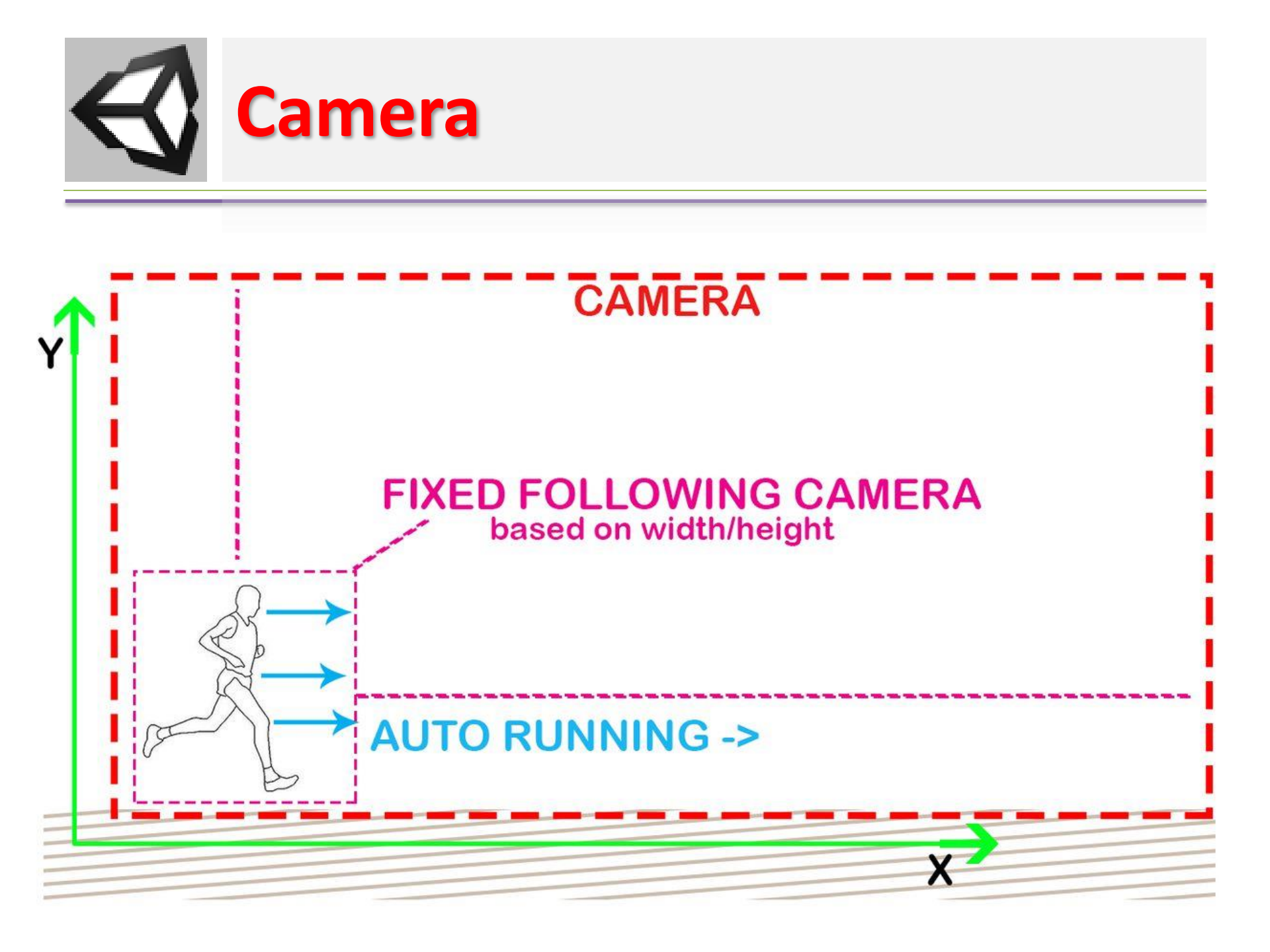
Object 3: Main Camera

- Là Camera gắn trực tiếp vào vật thể, dùng để quan sát môi trường khi người chơi di chuyển.
- Camera được gắn ở phía trên và được điều khiển bởi Script.
- Quan sát trong Inspector sẽ thấy các thành phần như: Transform, Camera, GUILayer, FlareLayer, Mouse Look (script).
- Thành phần Transform quy định vị trí, độ quay và độ co dãn của đối tượng

Camera

- Là thành phần thiết yếu của Camera, dùng để thiết lập khung nhìn cho vật thể.
- Bao gồm các tham số được mô tả ở slides sau

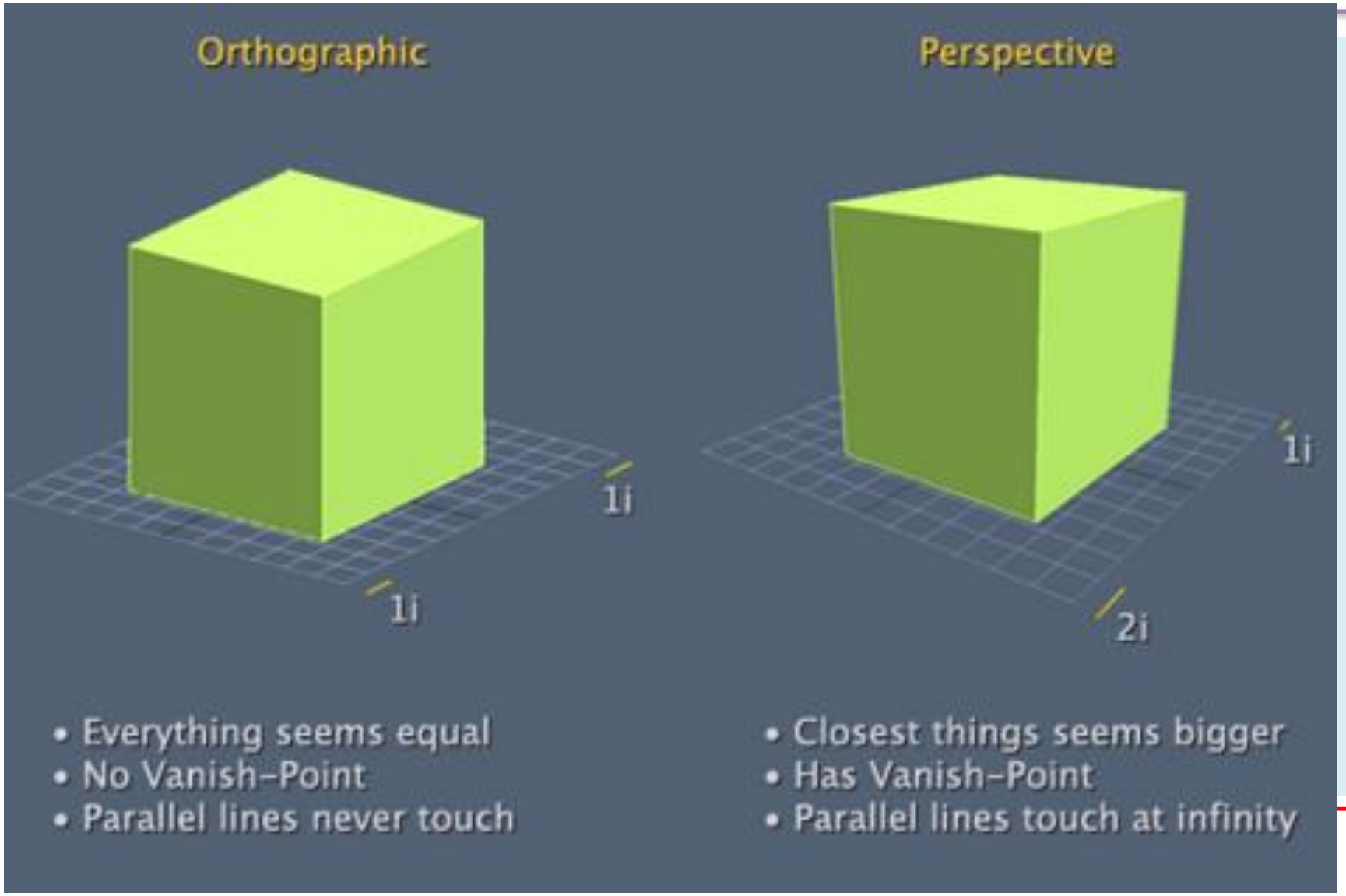



Camera
CAMERA
Y
FIXED FOLLOWING CAMERA
based on width/height
AUTO RUNNING ->
X

Camera (tt)

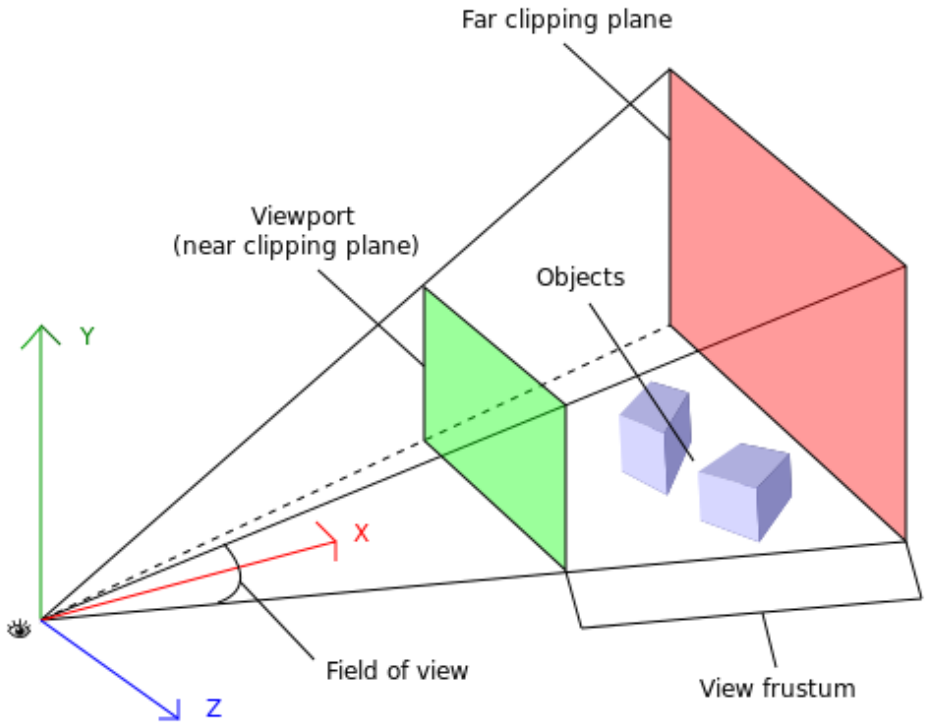
- Tham số:
  - **Clear Flags**: Cờ xác định thành phần nào bị xóa.
  - **Back Ground Color**: Màu của bầu trời xa phía sau vật thể.
  - **Culling Mask**: bao gồm hoặc bỏ qua lớp của các đối tượng đưa ra bởi Camera (áp dụng cho Layer).
  - **Projection**: phép chiếu camera có hai loại là chiếu phối cảnh (Perspective) hoặc chiếu song song (Orthographic).

Perspective & Orthographic



Camera (tt)

- Tham số (tt):
  - **Field of View**: Phạm vi chiếu của Camera
  - **Cliping Planes**: mức độ xa gần của Camera.



Camera (tt)

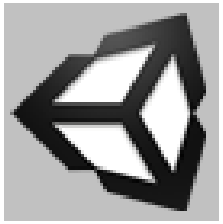
- Tham số (tt):
  - **Field of View**: Phạm vi chiếu của Camera
  - **Cliping Planes**: mức độ xa gần của Camera.
  - **Field of view**: Kích thước của Camera và vị trí trên màn hình. Được quy định bởi (X,Y,W,H).
  - **Dept**: Mức độ phía sau của Camera.
  - **Rendering Path**: sử dụng trong việc render ra màn hình của người dùng

GUILayer and Flare Layer

- **GUILayer** cho phép hiện ra các thành phần dạng 2D như Menu, text.
- **Flare Layer** cho phép camera hiện các thành phần ánh sáng.

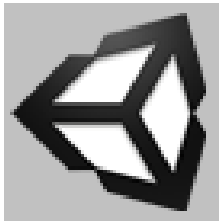
Mouse Look (Script)

- Cũng là script được gán cho First Person Character nhưng thay đổi giá trị Aexes thành Mouse Y.
- Kết quả thay đổi này sẽ ảnh hưởng đến khung nhìn Camera quanh trục X.



Audio listener

- Là một thành phần, giúp lưu giữ Audio trong GameObject.
- Thành phần này ở đây mặc định, sẽ được mô tả chi tiết trong các phần sau.



Scripting basics

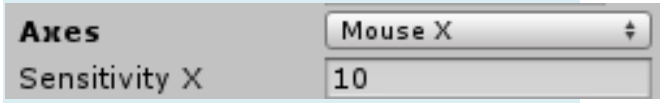
- Script đóng vai trò quan trọng trong việc điều khiển các nhân vật, các thành phần hoạt động khi Game đã được chạy.
- Script có thể được viết bằng C# hoặc bằng JavaScript.
- Có sẵn một bộ Editor hoặc tích hợp vào Visual.
- Các Script thông dụng có thể tìm thêm tại:

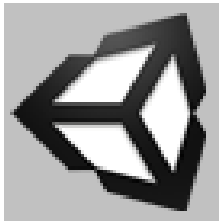
http://unity3d.com/support/documentation/ScriptReference/

Scripting basics

- **Commands**:
  - Một lệnh được kết thúc bằng dấu chấm phẩy.
- **Variables**:
  - Biến được khai báo có tên và gán cho kiểu dữ liệu.
  - Kiểu biến Vector3 là kiểu biến 1 tập 3 giá trị (X,Y,Z).
  - Các kiểu biến dạng public được Unity tự hiểu, tự tách ra và hiển thị lên Inspector. Ví dụ:

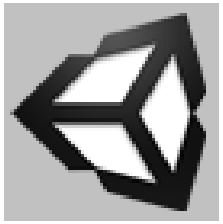
```
public enum RotationAxes { MouseXAndY = 0, MouseX = 1, MouseY = 2 }
public RotationAxes axes = RotationAxes.MouseXAndY;
public float sensitivityX = 15F;
public float sensitivityY = 15F;

public float minimumX = -360F;
```





Scripting basics (tt)

- **Functions**: Hàm được viết ra hoặc có sẵn.
  - Hàm **Update**(): dùng để cập nhật liên tục các tham số.
  - Hàm **OnMouseDown**(): điều khiển khi nhấn chuột
- If else statements: lệnh điều khiển.
- Một số dạng khác.
- Giải thích một số Script thông dụng và ví dụ minh họa.



End