



UNREAL  
ENGINE

## LECTURE 4

UE4 Gameplay Framework Classes

## LECTURE GOALS AND OUTCOMES

### Goals

---

The goals of this lecture are to

- Identify the Gameplay Framework classes
- Present the Game Mode class and its default classes
- Explain the Player Controller
- Present the Pawn class and some subclasses

### Outcomes

---

By the end of this lecture you will be able to

- Identify the common classes of the Gameplay Framework
- Know how to possess a Pawn using a Controller
- Draw simple information using the HUD class

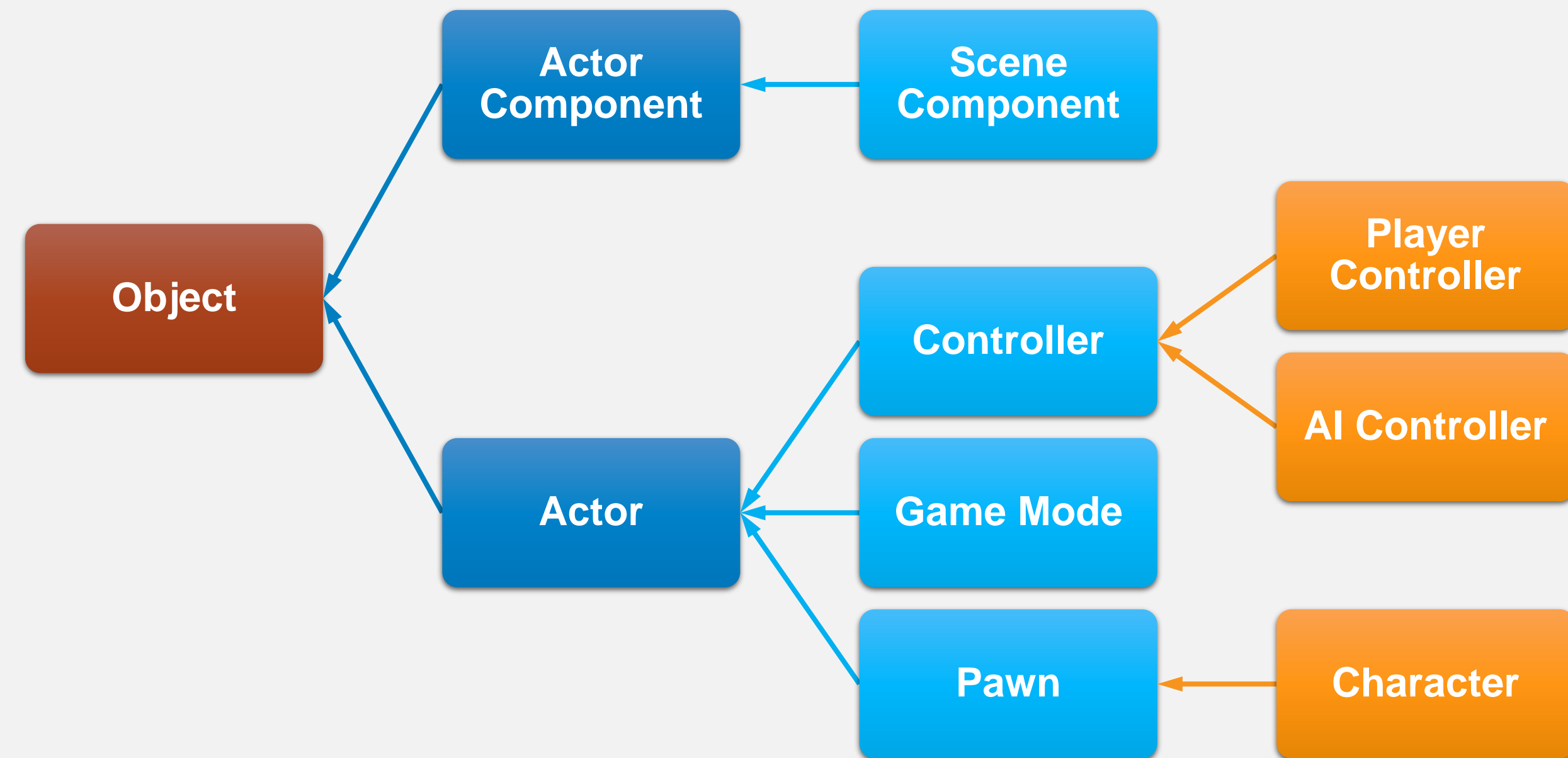




## COMMON CLASSES

When creating a Blueprint, you will be presented with a list of classes commonly used as the Blueprint parent class. These **common classes** are part of the **Gameplay Framework** and are used to represent players, characters, controllers, and game rules.

The image on the right shows the hierarchy of the common classes. The arrow indicates that the class on the right is inheriting from the class on the left. There is a base class in Unreal called the **Object** class.



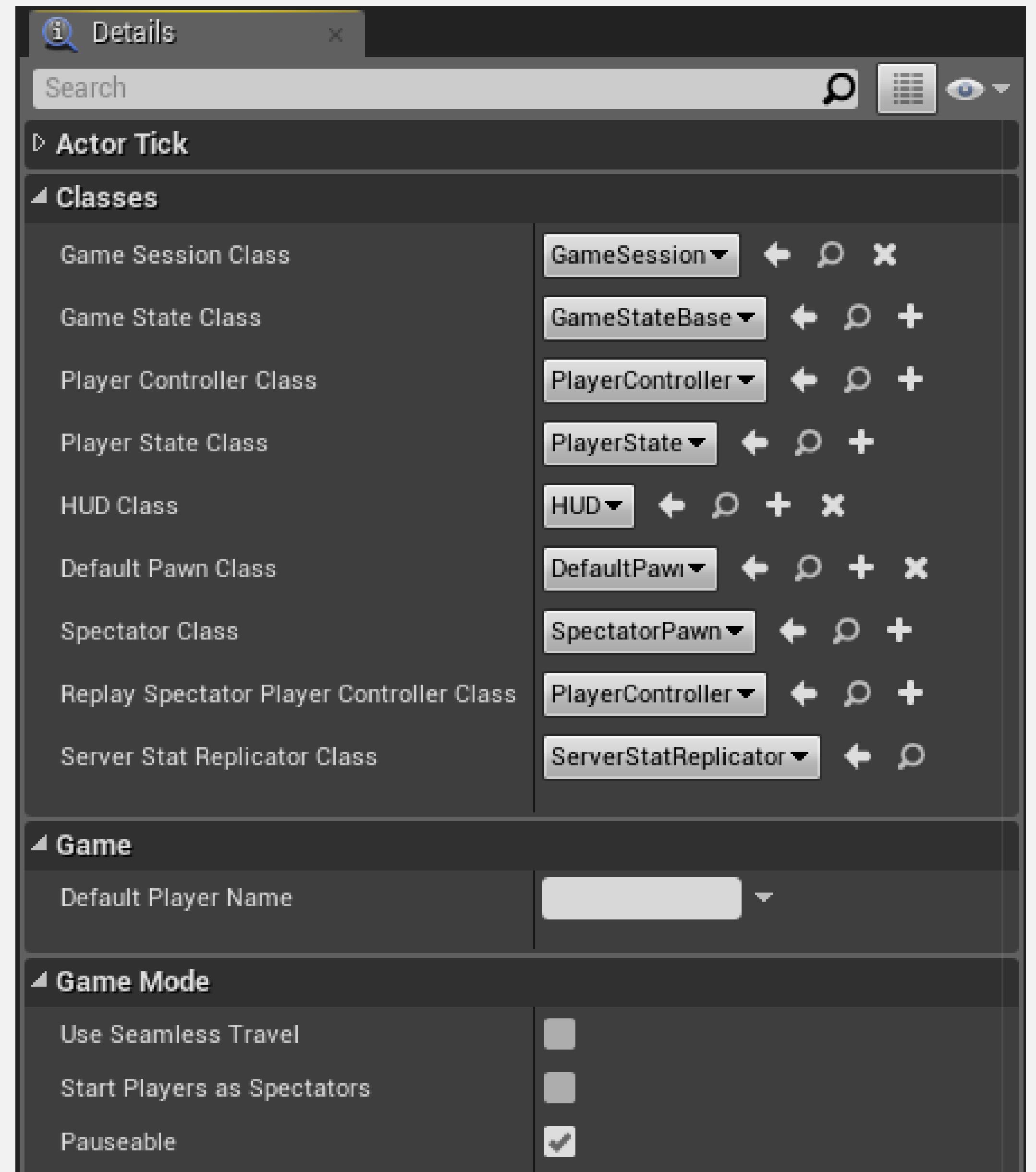


## GAME MODE

The **Game Mode** class is used to define the rules of the game. The Game Mode also specifies the default classes that will be used for the creation of **Pawn**, **Player Controller**, **Game State**, **HUD**, and other classes, as seen in the image on the right.

Each Level can have a different Game Mode. If a Game Mode is not specified for the Level, it will use the Game Mode that has been set for the project.

In a multiplayer game, the Game Mode exists only on the server and is not replicated to the clients.



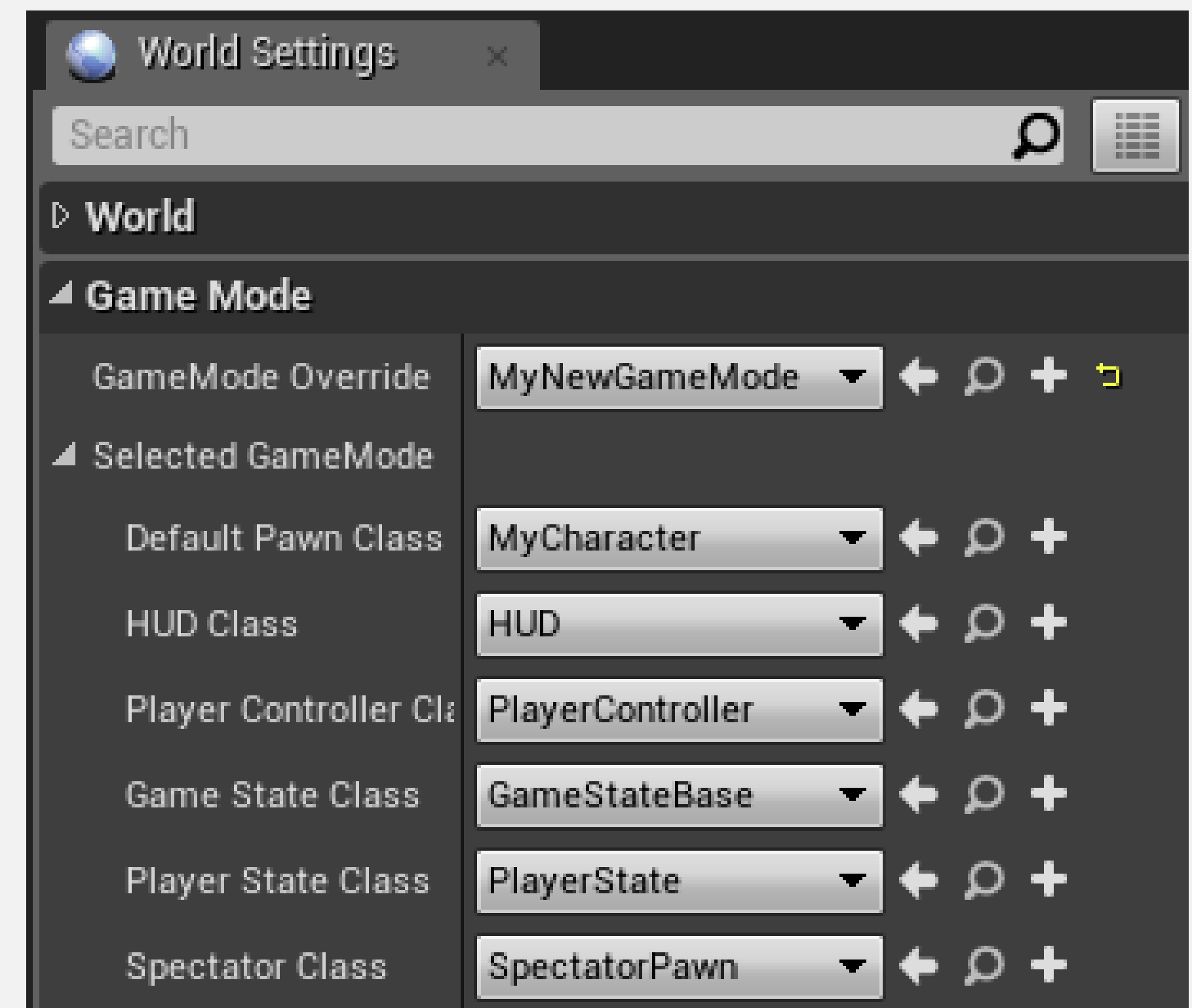
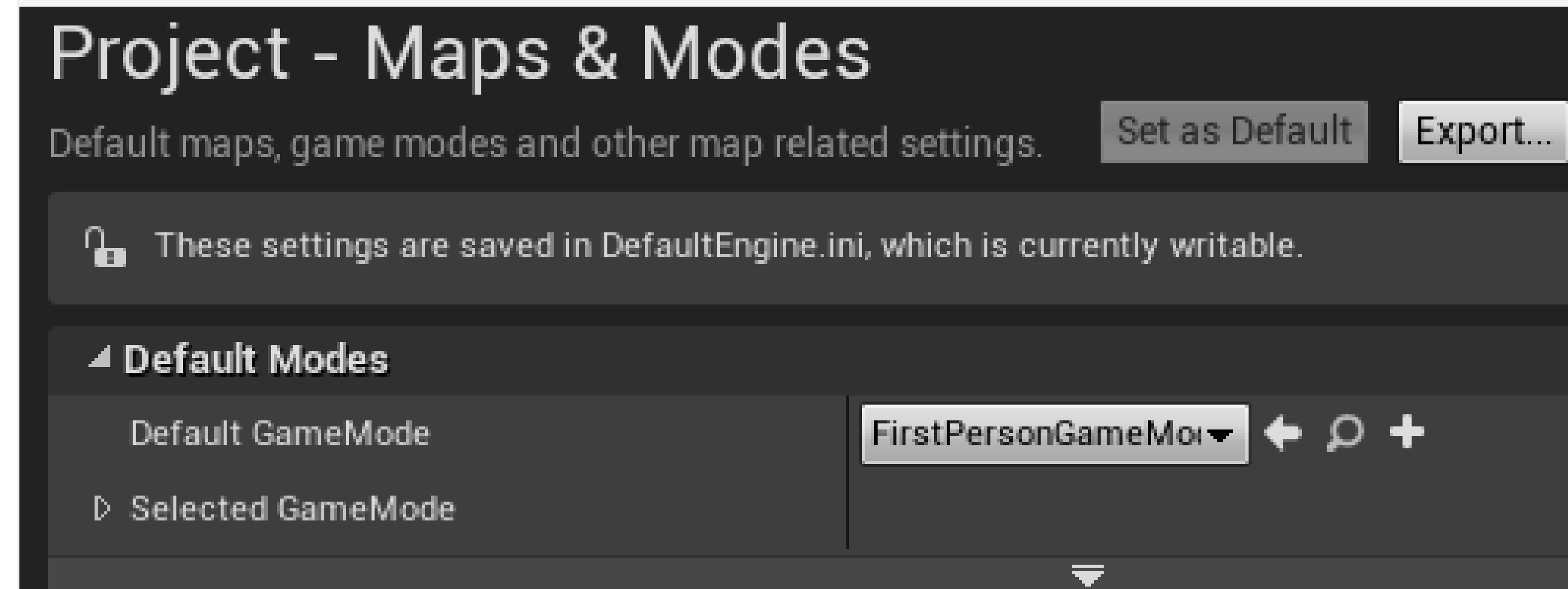


## ASSIGNING A GAME MODE

To specify the default Game Mode of a project, go to **Edit > Project Settings...** in the **Level Editor** and in the **Project** category select the **Maps & Modes** option. Choose the Game Mode in the **Default GameMode** property's drop-down, as seen in the top image on the right.

To specify the Game Mode of a Level, click the **Settings** button in the **Level Editor** and choose the **World Settings** option. Choose the Game Mode in the **GameMode Override** property's drop-down, as seen in the bottom image.

The Game Mode of a Level overrides the default Game Mode of the project.



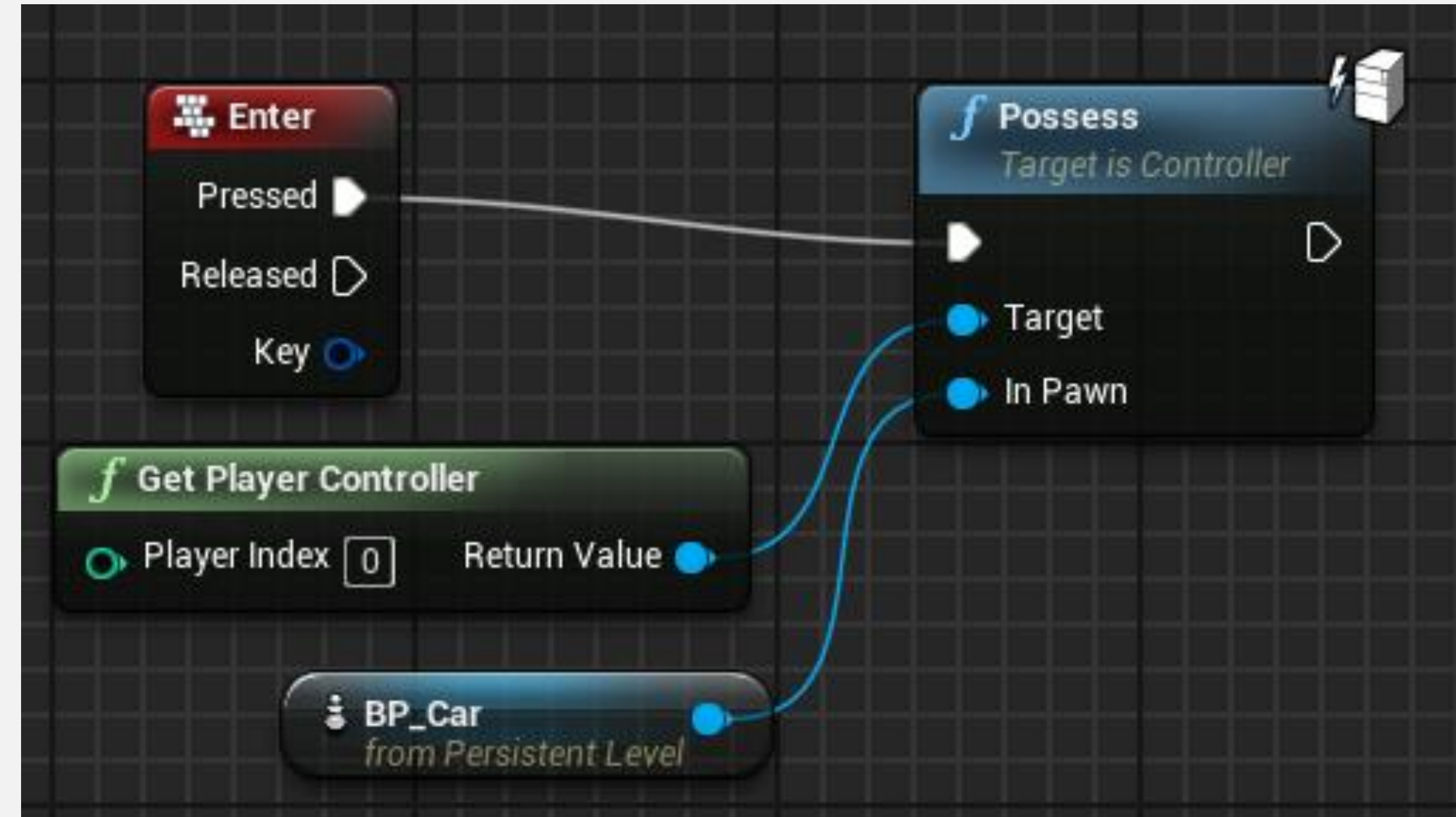


## PLAYER CONTROLLER

The **Controller** class has two main subclasses. The **Player Controller** class is used by human players, and the **AI Controller** class uses artificial intelligence to control the **Pawn**.

**Pawn** and **Character** classes only receive input events if they are being possessed by a Player Controller.

The Pawn class being possessed by a Player Controller can be changed in-game. The image on the right is from a Level Blueprint and shows the use of the **Possess** function. In this example, a **Player Controller** will possess the **BP\_Car** Pawn Actor in the Level when the **Enter** key is pressed.





## PAWN

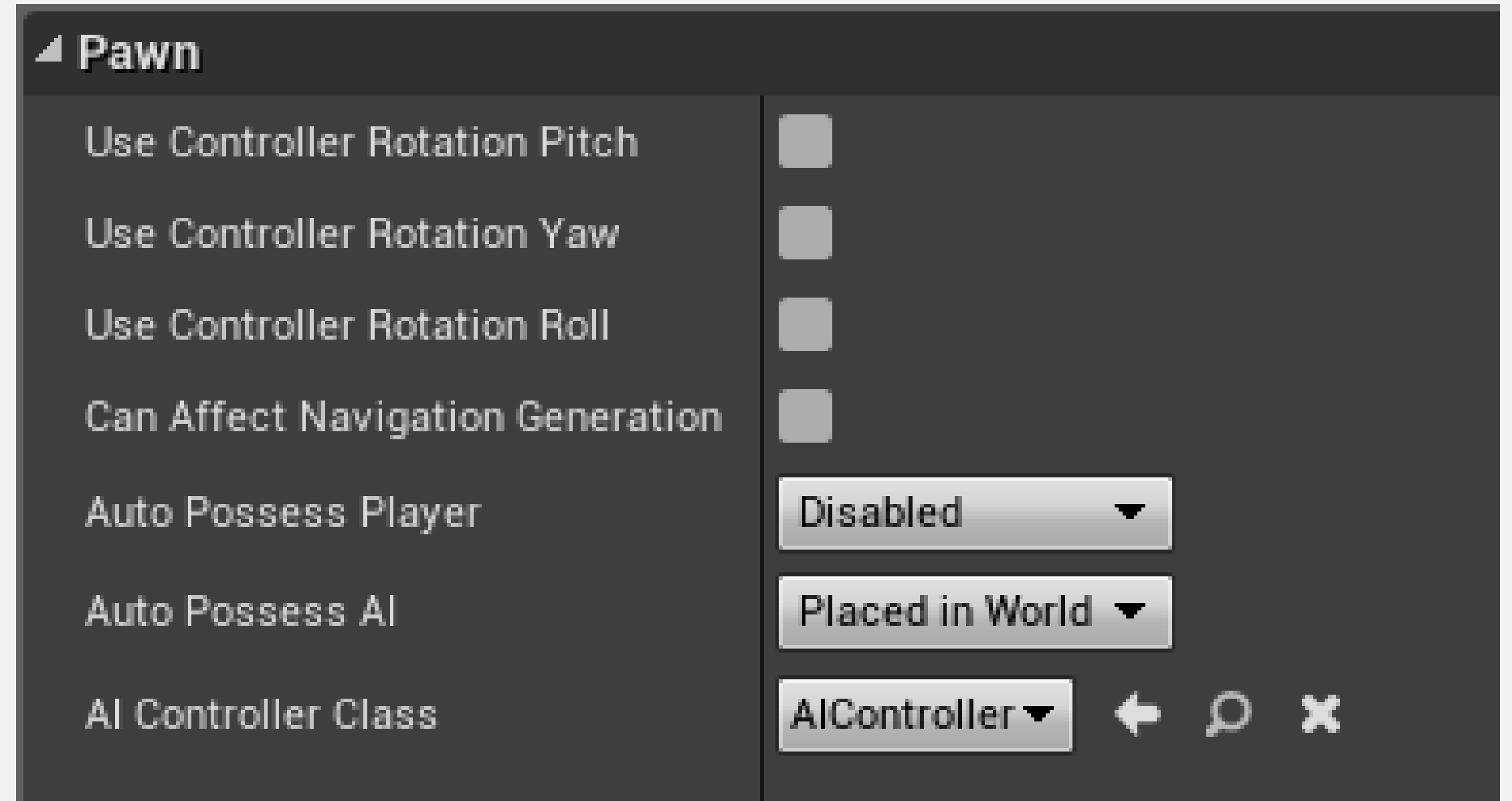
---

**Pawns** are **Actors** that can be controlled (possessed) by **Controllers** (Player or AI).

The Pawn class represents the physical body, and the Controller class represents the brain.

The image on the right shows some of the parameters that are inherited from the Pawn class. The Pawn class can use the rotation values of the Controller that is possessing it.

Other attributes indicate how the Pawn is possessed by a Controller.



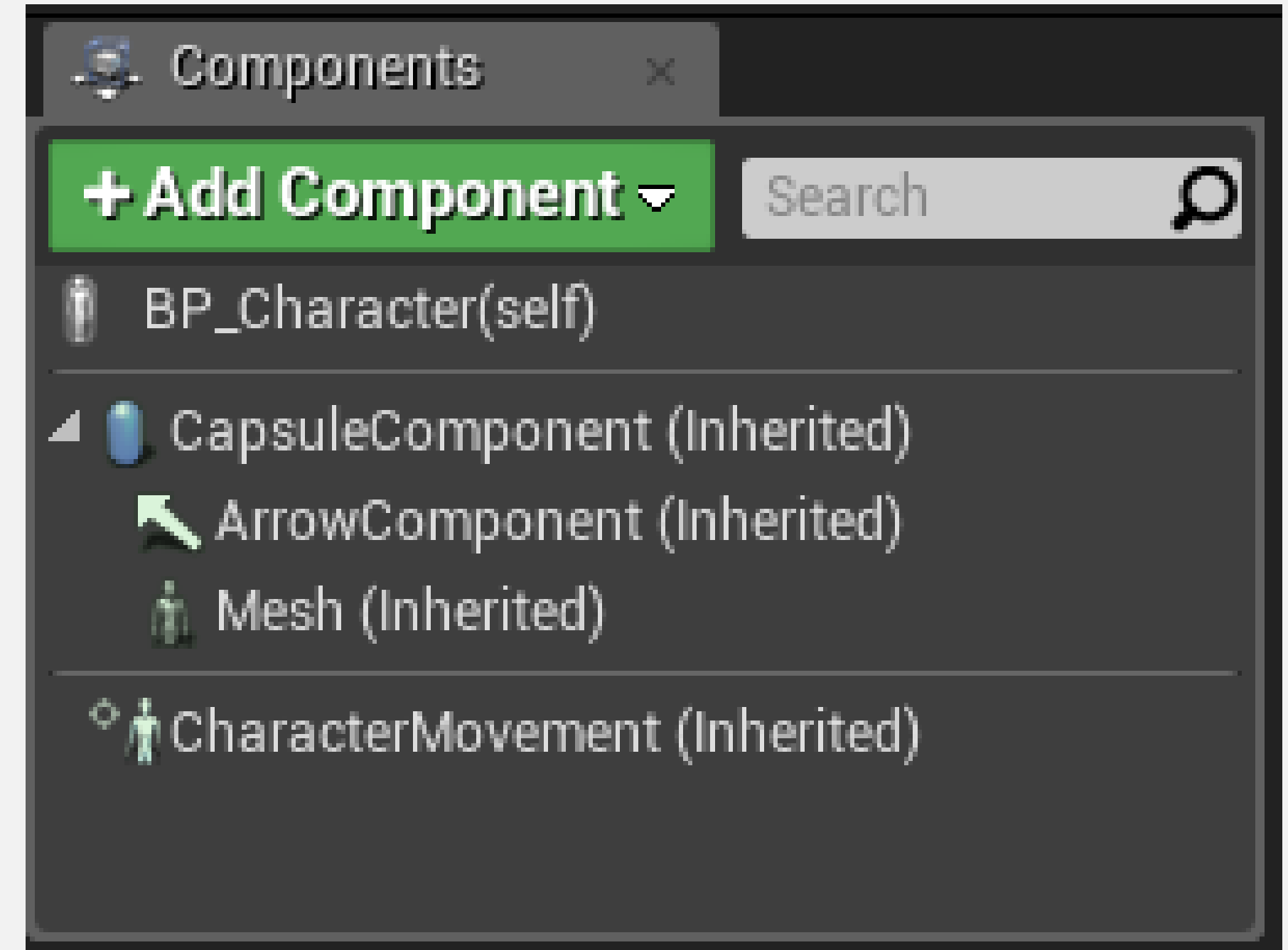


## PAWN: CHARACTER

The **Character** class is a subclass of the **Pawn** class that is made to represent bipedal characters that can walk, run, jump, swim, and fly. This class already has a set of components to assist in this goal.

The image on the right shows the components present in the Character class.

- The **CapsuleComponent** is used for collision testing.
- The **ArrowComponent** indicates the current direction of the character.
- The **Mesh** component is the Skeletal Mesh that visually represents the character. The animation of the Mesh component is controlled by an **animation Blueprint**.
- The **CharacterMovement** component is used to define various types of character movement such as walking, running, jumping, swimming, and flying.



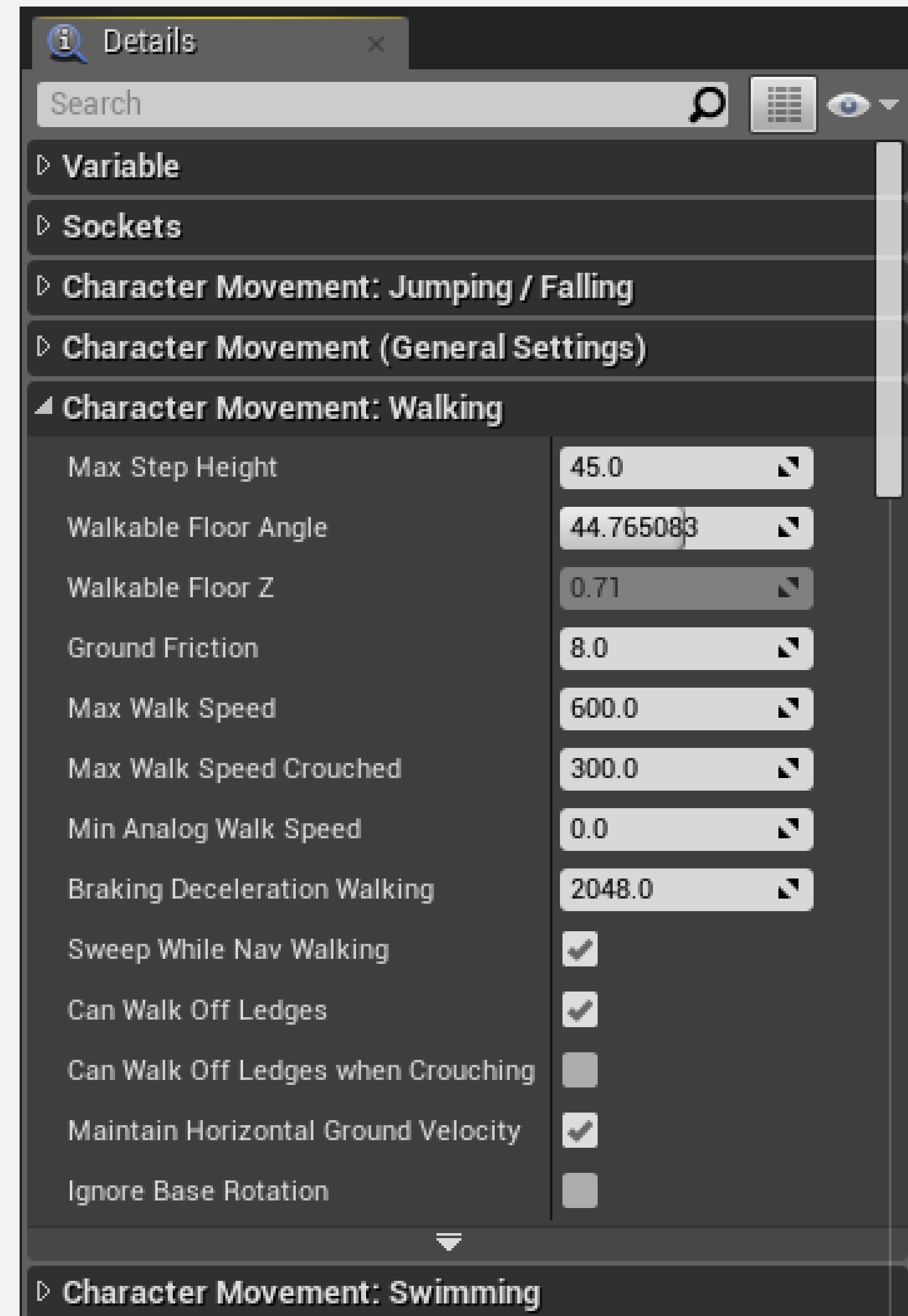




# CHARACTER MOVEMENT

The **CharacterMovement** component is written in C++ and handles movement as well as replication and prediction in multiplayer games.

There are a lot of properties for various types of movements that can be adjusted on the component.





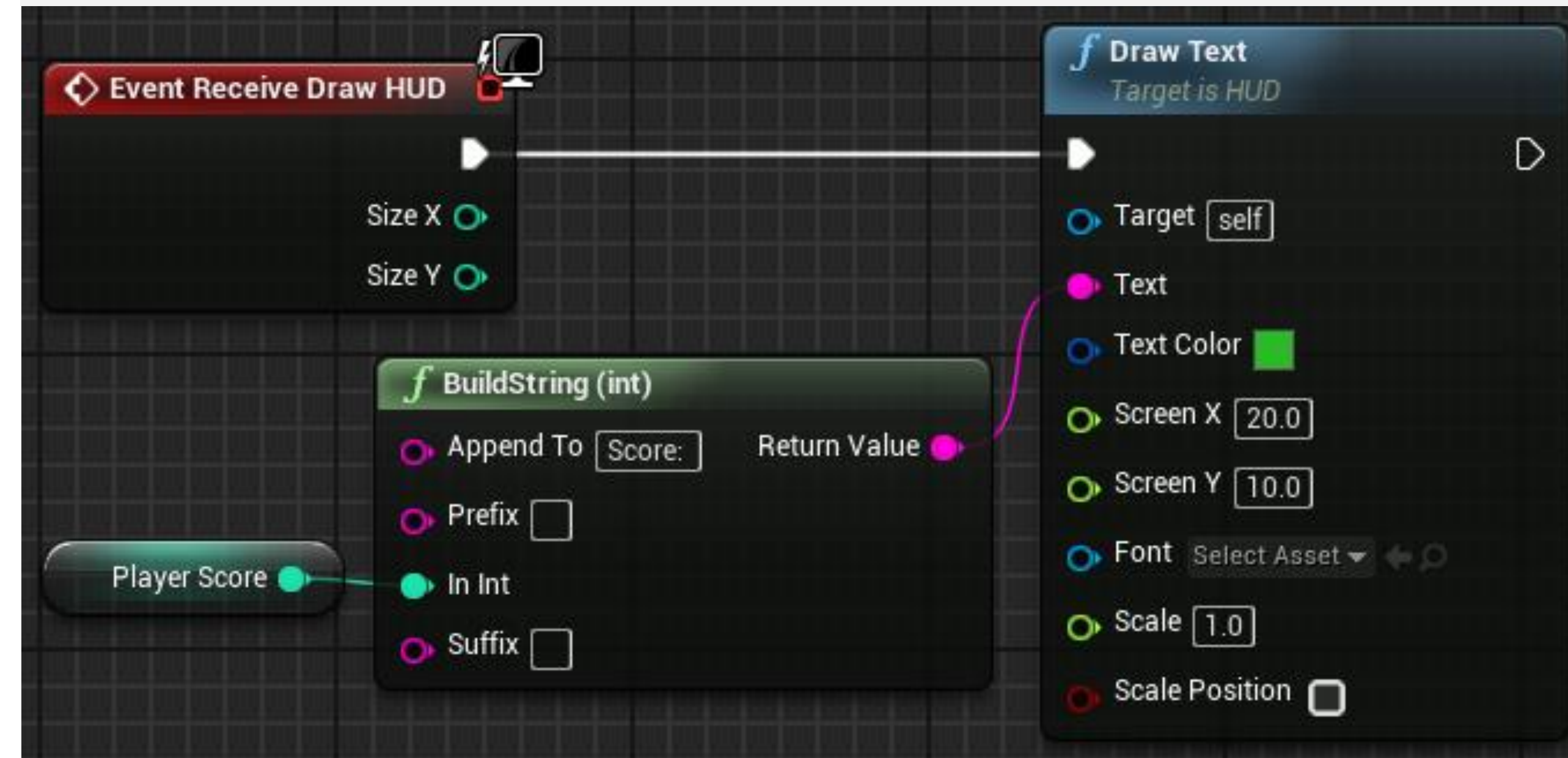
# HUD

The **HUD** class is used to create a heads-up display (HUD), which is a kind of on-screen display that provides quick access to important information. The HUD is used in games to display various information to the players, such as their score, time, energy, and so on.

In Unreal Engine, the HUD class is a base class that contains a **Canvas**, which is an object on which primitives, such as text and textures, can be drawn.

The HUD class contains an event named “**Receive Draw HUD**”, which is used to draw the primitives each frame.

The HUD class exists only on each client and does not replicate.



# SUMMARY

---

This lecture presented the Gameplay Framework classes and the relationship between some common classes.

It showed how to possess a Pawn using a Controller and how to draw simple information using the HUD class.

