Active Ragdoll Character Controller:

Table of Contents:

1. Quick Start Guide

This asset requires Cinemachine, so if you don't already have it installed simply navigate to Window > Package Manager > Unity Registry and install it. For the quickest start, simply navigate to ***Active Ragdoll Controller / Prefabs***, drag out either the SimplePlayerPrefab or FullRigPrefab into your scene, and hit play! The character controller prefabs contain all the necessary components such as cameras, scripts, and other components ready to go without the need for any other initial setup. For full physics interaction, make sure all your rigidbodies are tagged as 'grabbable' in order to be able to be picked up by the player. Since the active ragdoll system automatically uses the given joint strengths to apply forces using physics, the weight, drag, etc. of the grabbed object will automatically be taken into account. For jump detection, make sure the ground is tagged as 'Ground'.

If the default values of the weights and physics strengths feel okay for your game, then this is all the setup that is required to get the active ragdoll going, and you can now focus on the level and game design! If you would like to tweak some of these values, Section 3 covers the parameters for the player related physics controller scripts, and what each of them does. Most of the other scripts in the Scripts folder are much simpler one-offs like placing an item inside of a box or pushing a button to trigger an event, or activating a gas canister / explosive barrel after a significant collision, and are much easier to understand. However, if you run into any difficulties, don't hesitate to reach out at jneumeister24@gmail.com.

2. Active Ragdoll System Overview

In both the simple and fullbody player prefabs you will find two objects: the ragdoll itself, and a helper object to smooth the input from the player. In the fullbody prefab, these are labeled as Ragdoll and Helper, and in the simple prefab: PlayerCapsule and Player respectively.

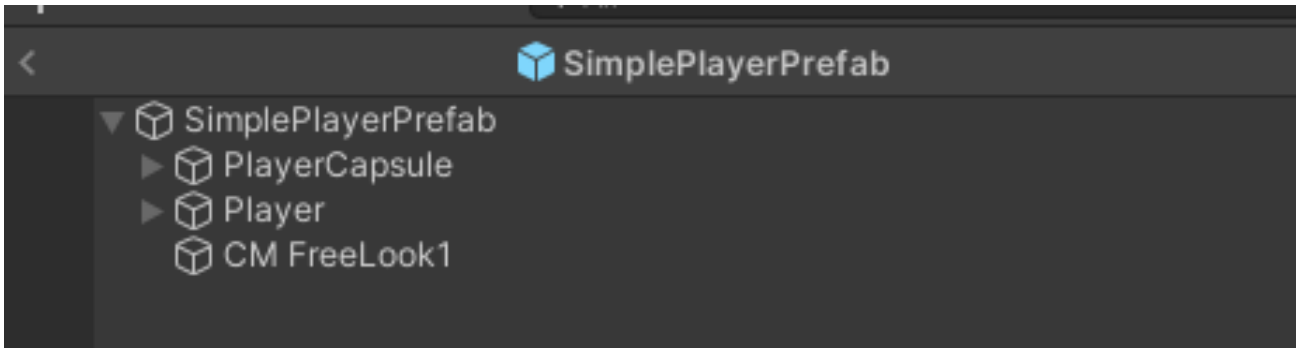

Fig 2.1: (Simple Prefab) **PlayerCapsule**: *Ragdoll*, **Player**: *Input / Helper*

Fig 2.2: (Fullbody Prefab) **Ragdoll**: *Ragdoll*, **Helper**: *Input / Helper*

The ragdoll will contain all of the physics limbs as separate objects held together using ConfigurableJoints, as well as a PhysicsJointController script that allows for easier modifying of the joint at runtime: limb strength, toggle ragdoll / limp, etc.

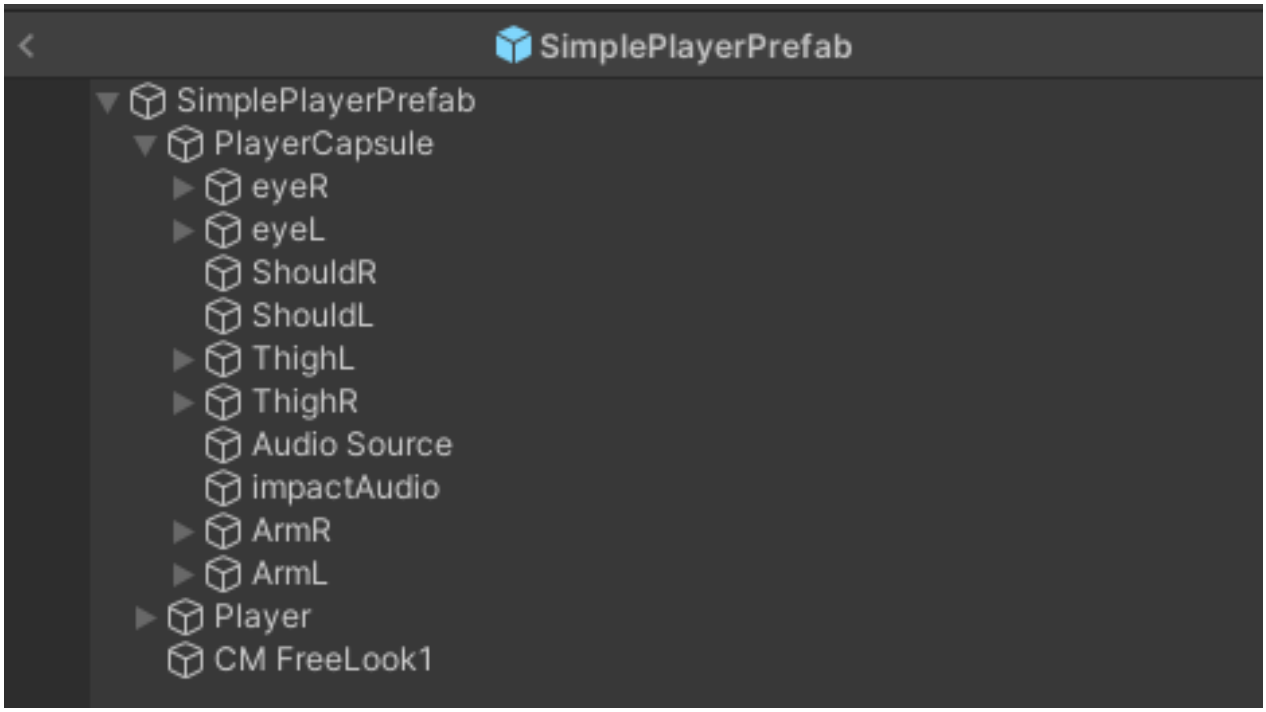

Fig 2.3: Hierarchy of physics limbs in Simple Prefab

The helper / player object is used to receive input from the keyboard / mouse and provide that as direction for the ragdoll using the PlayerController script (see Section 3a). This object will also contain an ActiveRagdollController script (Section 3b), which is used to easily trigger events such as going limp as well as manage limb strength on the fly. In both of these prefabs will also be an animator object, referenced by the PlayerController, which is used to manage the animation states as well as set the target rotation of the ragdoll's physics joints. In order to add a new animation / pose to the active ragdoll, simply record a new animation for this object using Unity's built-in animation system, and update the state manager located in ***Active Ragdoll Character Controller / Prefabs / Animation***.

3. Components / Scripts

a. PlayerController:

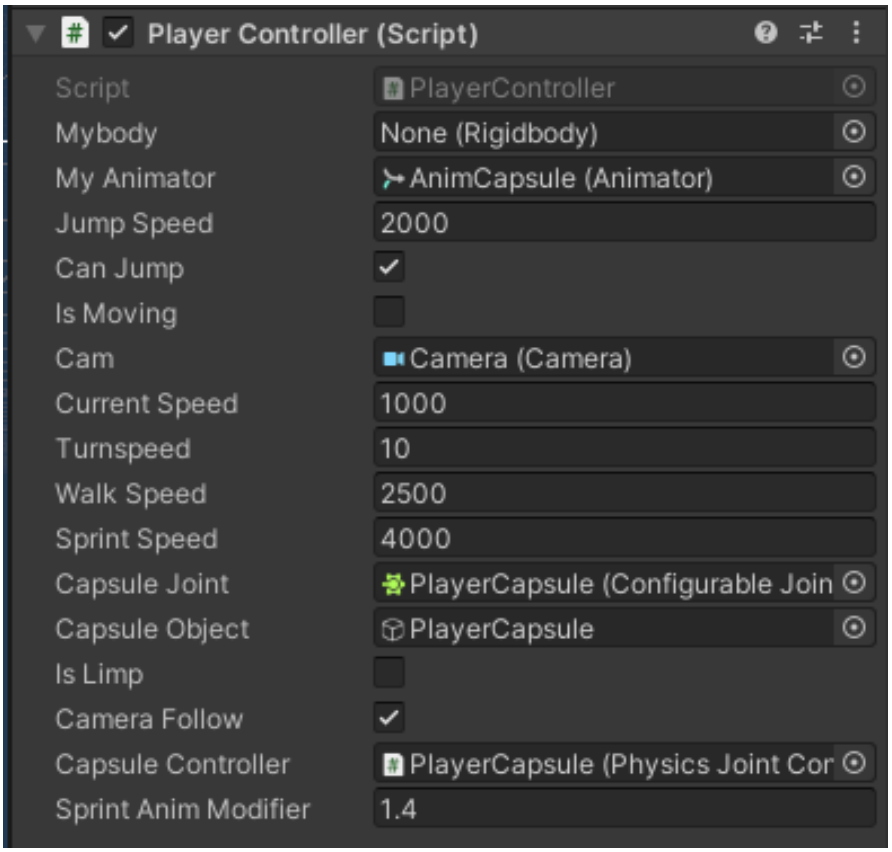

Fig 3.1: PlayerController Script - Located on Helper / Input Receiver

- **Mybody**: Rigidbody of the player helper / input receiver object, assigned to the Rigidbody of the object it is attached to on Start()
- **MyAnimator**: Animator used to set the physics limbs' target rotation
- **JumpSpeed**: Force used to jump
- **IsMoving**: Bool managed by script to determine whether to play running animations, etc. based on current Rigidbody velocity
- **CurrentSpeed**: Switches between WalkSpeed and SprintSpeed depending on whether the player holds shift, used to apply movement force.
- **CapsuleJoint**: ConfigurableJoint that attaches the ragdoll to the helper / input receiver
- **CapsuleObject**: Reference to Ragdoll Object
- **IsLimp**: Bool managed by script to determine whether the PlayerController script should receive input or not (If the player is limp / ragdolled, do not receive input)

- **CameraFollow**: Bool toggled by pressing ‘Z’ to enable / disable the camera following the player’s rotation
- **CapsuleController**: Reference to the PhysicsJointController script that controls the joint attaching the ragdoll to the helper / input receiver, used to allow free movement when enabling limp / ragdoll mode
- **SprintAnimModifier**: Rate at which to play the walking animation when the player is sprinting (holding shift)

b. ActiveRagdollController:

Fig 3.2: ActiveRagdollController Script - Located on Helper / Input Receiver

- **Limbs**: List of PhysicsJointController scripts attached to each of the player’s limbs
- **IsLimp**: Current state of ragdoll
- **PlayerBaseBody**: Reference to the helper / input receiver object
- **Player**: Reference to PlayerController script on helper / input receiver object
- **CapsuleBody**: Reference to Ragdoll GameObject
- **MaxDistance**: Maximum distance between the helper / input receiver and the ragdoll before teleporting the helper to the ragdoll’s position

c. GrabScript:

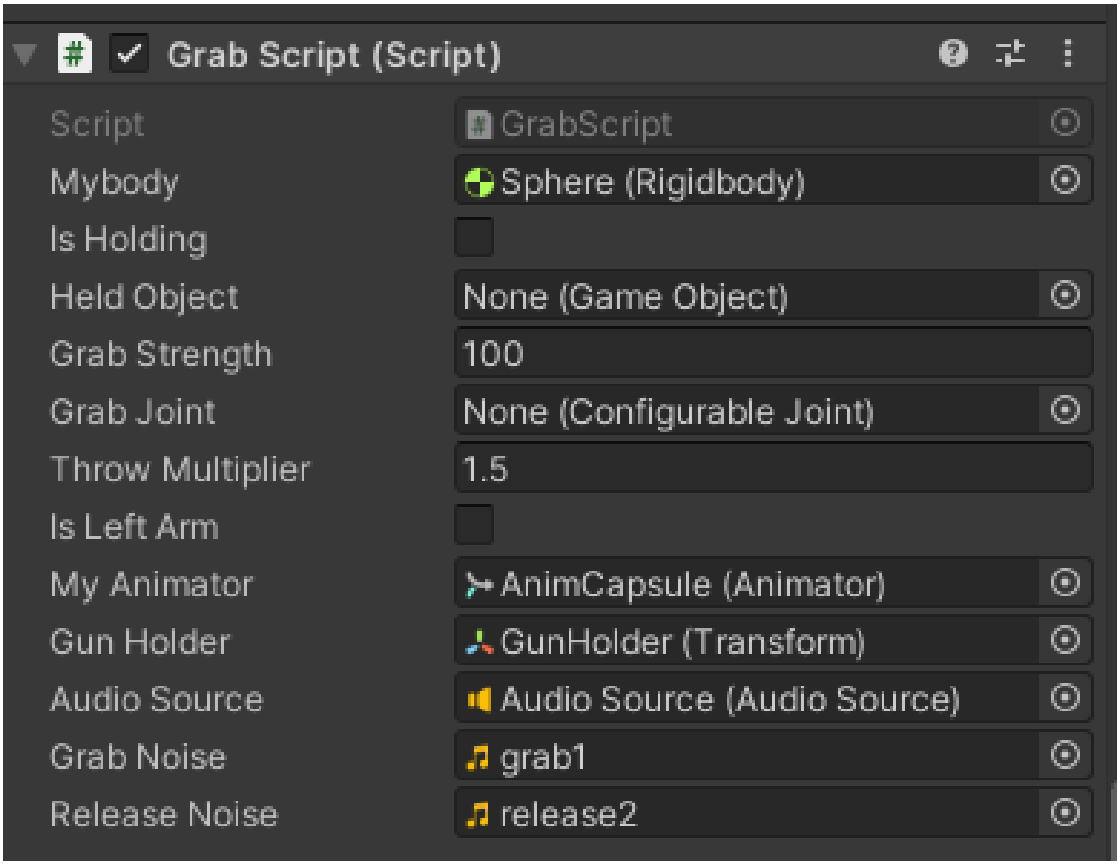

Fig 3.3: GrabScript - Located on hand objects

- **Mybody**: Rigidbody of the hand this script is attached to
- **HeldObject**: Reference to the object currently being grabbed, if any
- **GrabStrength**: Angular force used to keep the held object from rotating in the hand
- **GrabJoint**: ConfigurableJoint being used to grab, if any (Instantiated at runtime)
- **ThrowMultiplier**: Velocity multiplier applied to objects as they are released
- **MyAnimator**: Reference to the player's animator
- **GunHolder**: If picking up an object labeled 'gun', move the object to this position / rotation before attaching the grab joint
- **AudioSource** / Noises: Audio Source and noises played when the player picks up / releases an object

d. PhysicsJointController:

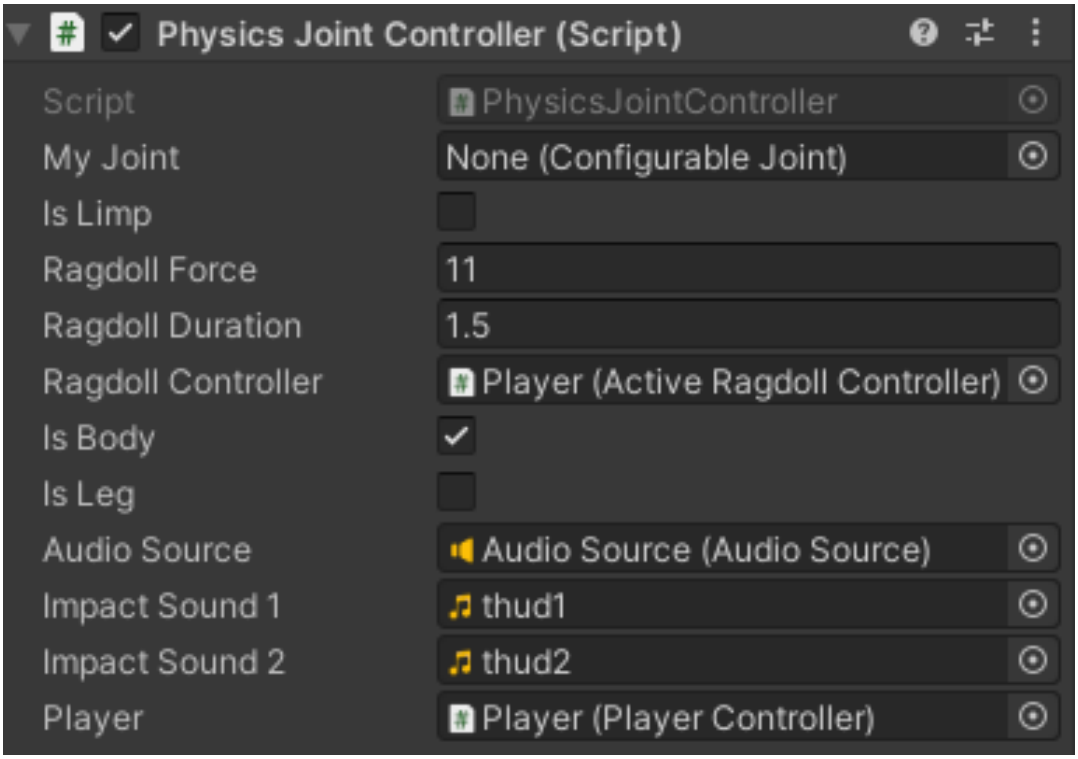

Fig 3.4: PhysicsJointController Script - Located on each physics limb

- **MyJoint**: ConfigurableJoint of the GameObject this script is located on, assigned at Start()
- **RagdollForce**: Magnitude of force required to trigger ragdoll / limp mode on the player
- **RagdollDuration**: How long the player should stay ragdolled for after hit
- **RagdollController**: Reference to the ActiveRagdollController script on the helper / input receiver object used to trigger changes to the ragdoll
- **IsBody** / IsLeg: Used to determine locational damage
- **AudioSource / Sounds**: Played at random when colliding with enough force
- **Player**: Reference to the helper / input receiver PlayerController script