# Celestial Rewind

Generated by Doxygen 1.8.17

1	Hierarchical Index	1
	1.1 Class Hierarchy	1
2	Class Index	3
	2.1 Class List	3
3	Class Documentation	5
	3.1 AudioController Class Reference	5
	3.1.1 Detailed Description	5
	3.1.2 Member Function Documentation	6
	3.1.2.1 Update()	6
	3.1.2.2 WaitForNextStep()	6
	3.2 DebugPanel Class Reference	6
	3.2.1 Detailed Description	7
	3.2.2 Member Function Documentation	7
	3.2.2.1 Start()	7
	3.2.2.2 Update()	7
	3.2.3 Member Data Documentation	7
	3.2.3.1 Drag	8
	3.2.3.2 Gravity	8
	3.2.3.3 Leniency	8
	3.2.3.4 Move	8
	3.2.3.5 panel	8
	3.2.3.6 ppc	8
	3.2.3.7 RCooldown	9
	3.2.3.8 RewindSpeed	9
	3.2.3.9 RLimit	9
	3.2.3.10 Speed	9
	3.2.3.11 zc	9
	3.3 FallingRocksObstacle Class Reference	9
	3.3.1 Member Function Documentation	10
	3.3.1.1 Timeout()	10
	3.4 FirstPersonLook Class Reference	10
	3.5 IRewinder Class Reference	11
	3.5.1 Detailed Description	12
	3.5.2 Member Function Documentation	12
	3.5.2.1 HasStates()	12
	3.5.2.2 NeedUpdate()	12
	3.5.2.3 Pause()	12
	3.5.2.4 Play()	13
	3.5.2.5 Reset()	13
	3.5.2.6 RewindState()	13
	3.5.2.7 Start()	13
	· · · · · · · · · · · · · · · · · · ·	

3.5.2.8 Store()	13
3.5.3 Member Data Documentation	14
3.5.3.1 printDebug	14
3.5.3.2 states	14
3.5.3.3 zController	14
3.5.4 Property Documentation	14
3.5.4.1 startPos	14
3.5.4.2 startRot	14
3.6 LevelLoader Class Reference	15
3.7 LevelTransitionTrigger Class Reference	15
3.8 MenuScript Class Reference	15
3.9 MovementScript Class Reference	16
3.10 PhysicsPlayerController Class Reference	17
3.10.1 Detailed Description	18
3.10.2 Member Function Documentation	18
3.10.2.1 DoJump()	18
3.10.2.2 DoMovement()	18
3.10.2.3 OnCollisionEnter()	19
3.10.2.4 Start()	19
3.10.2.5 Update()	19
3.10.2.6 UpdateCamera()	19
3.10.3 Member Data Documentation	20
3.10.3.1 cam	20
3.10.3.2 grounded	20
3.10.3.3 maxSpeed	20
3.10.3.4 mouseXSensitiviy	20
3.10.3.5 mouseYSensitiviy	20
3.10.3.6 playerAcceleration	21
3.10.3.7 rigidBody	21
3.10.3.8 turn	21
3.11 PhysicsRewinder Class Reference	21
3.11.1 Detailed Description	22
3.11.2 Member Function Documentation	22
3.11.2.1 OnCollisionStay()	22
3.11.2.2 Pause()	22
3.11.2.3 Play()	23
3.11.2.4 RewindState()	23
3.11.2.5 Start()	23
3.11.2.6 Store()	24
3.11.2.7 Update()	24
3.11.3 Member Data Documentation	24
3.11.3.1 rb	24

3.11.3.2 rewindVelocity	24
3.11.3.3 startForce	24
3.12 PlayerController Class Reference	25
3.12.1 Detailed Description	25
3.13 RoomController Class Reference	26
3.14 RoomSettingsScript Class Reference	26
3.15 SnapState Class Reference	26
3.15.1 Detailed Description	27
3.16 SpawningRocks Class Reference	27
3.17 TextureResizeScript Class Reference	27
3.18 ThirdPersonMovement Class Reference	28
3.19 TriggerWalls Class Reference	28
3.20 TriigerWinLoss Class Reference	28
3.21 VignetteRewind Class Reference	29
3.22 ZController Class Reference	29
3.22.1 Detailed Description	30
3.22.2 Member Function Documentation	30
3.22.2.1 Approximate()	31
3.22.2.2 OnTriggerEnter()	31
3.22.2.3 Pause()	31
3.22.2.4 Start()	31
3.22.2.5 Update()	32
3.22.3 Member Data Documentation	32
3.22.3.1 active	32
3.22.3.2 approximateLeniency	32
3.22.3.3 cooldown	32
3.22.3.4 deltaTime	32
3.22.3.5 recordInterval	33
3.22.3.6 rewindCooldown	33
3.22.3.7 rewindLimit	33
3.22.3.8 rewinds	33
3.22.3.9 rewindScale	33
Index	35

# **Chapter 1**

# **Hierarchical Index**

# 1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

MonoBehaviour	
AudioController	Ę
DebugPanel	6
FallingRocksObstacle	ç
FirstPersonLook	10
IRewinder	11
PhysicsRewinder	21
LevelLoader	15
LevelTransitionTrigger	15
MenuScript	15
MovementScript	16
PhysicsPlayerController	17
PlayerController	25
RoomController	26
RoomSettingsScript	26
SpawningRocks	27
TextureResizeScript	27
ThirdPersonMovement	28
TriggerWalls	28
TriigerWinLoss	
VignetteRewind	29
ZController	29
SnanState	26

2 Hierarchical Index

# Chapter 2

# **Class Index**

# 2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

AudioController	
A controller for player audio	5
DebugPanel	
A controller for the debug settings. Currently contains a manually maintained map of UI elements	
to the corresponding object value	6
FallingRocksObstacle	ç
FirstPersonLook	10
IRewinder	
An abstraction of the rewinder functionality for use with a ZController to rewind an object	11
LevelLoader	15
LevelTransitionTrigger	15
MenuScript	15
MovementScript	16
PhysicsPlayerController	
A Rigidbody based player controller	17
PhysicsRewinder	
An implementation of IRewinder made for operating with Rigidbody physics	21
PlayerController	
Modified from unity docs: https://docs.unity3d.com/ScriptReference/←	
CharacterController.Move.html To be removed in later versions	25
RoomController	26
RoomSettingsScript	26
SnapState	
A representation of a gameObject's current state	26
SpawningRocks	27
TextureResizeScript	27
ThirdPersonMovement	28
TriggerWalls	28
TriigerWinLoss	28
VignetteRewind	29
ZController	
A script to control all IRewinders in the current GameObject's children and process their state	
capture motion and rewind	20

4 Class Index

# **Chapter 3**

# **Class Documentation**

## 3.1 AudioController Class Reference

A controller for player audio.

## **Public Member Functions**

IEnumerator WaitForNextStep ()

A coroutine method that waits until the next step audio should be played

## **Public Attributes**

- AudioSource audioSource
- AudioClip[] walkSounds
- AudioSource timeAudio
- AudioClip timeRewindSound
- float waitStep = .5f
- PhysicsPlayerController ppc

## **Private Member Functions**

• void Update ()

Controls player walking and time rewind sounds.

## **Private Attributes**

- int walkSoundIndex
- bool playSound = true

## 3.1.1 Detailed Description

A controller for player audio.

## 3.1.2 Member Function Documentation

## 3.1.2.1 Update()

```
void AudioController.Update ( ) [inline], [private]
```

Controls player walking and time rewind sounds.

## 3.1.2.2 WaitForNextStep()

```
IEnumerator AudioController.WaitForNextStep ( ) [inline]
```

A coroutine method that waits until the next step audio should be played

Returns

A WaitForSeconds object.

The documentation for this class was generated from the following file:

· Assets/Scripts/AudioController.cs

## 3.2 DebugPanel Class Reference

A controller for the debug settings. Currently contains a manually maintained map of UI elements to the corresponding object value.

#### **Public Attributes**

· TMP InputField Gravity

A text field representing gravity's value

• TMP\_InputField RewindSpeed

A text field representing how often per second a rewind capture is performed

TMP\_InputField Drag

A text field representing drag

• List< TMP\_InputField > Move

A list of input fields corresponding to the three values of a player's movement acceleration

List< TMP\_InputField > Speed

A list of input fields corresponding to the three values of a player's max velocity

TMP\_InputField RLimit

A text field representing how long rewind can be held before time resumes.

TMP\_InputField RCooldown

A text field representing how long the player must wait after burning out the rewind before they can use it again.

• TMP InputField Leniency

The leniency for the ZController's check for no motion.

PhysicsPlayerController ppc

The PhysicsPlayerController the fields are referencing.

· ZController zc

The ZController the fields are referencing

· GameObject panel

The parent container for all the text fields.

#### **Private Member Functions**

• void Start ()

Set up all of the elements. If they have not been set in the editor, they will be extracted from the children of the panel GameObject. For each element, the corresponding value is pulled from game data and populated into the text fields.

· void Update ()

Every frame, so long as the panel itself is active, all values in the gui will be copied into their corresponding fields in the game. This is done through TryParse calls, where a failure will result in the original value being maintained. If the key (f1) for the debug is called, the panel will be toggled.

## 3.2.1 Detailed Description

A controller for the debug settings. Currently contains a manually maintained map of UI elements to the corresponding object value.

Not as automated as I'd like but that would be difficult to do without reflection, which is already used slightly in instantiation.

#### 3.2.2 Member Function Documentation

#### 3.2.2.1 Start()

```
void DebugPanel.Start ( ) [inline], [private]
```

Set up all of the elements. If they have not been set in the editor, they will be extracted from the children of the panel GameObject. For each element, the corresponding value is pulled from game data and populated into the text fields.

#### 3.2.2.2 Update()

```
void DebugPanel.Update ( ) [inline], [private]
```

Every frame, so long as the panel itself is active, all values in the gui will be copied into their corresponding fields in the game. This is done through TryParse calls, where a failure will result in the original value being maintained. If the key (f1) for the debug is called, the panel will be toggled.

## 3.2.3 Member Data Documentation

## 3.2.3.1 Drag

TMP\_InputField DebugPanel.Drag

A text field representing drag

Possibly could be removed now

## 3.2.3.2 Gravity

TMP\_InputField DebugPanel.Gravity

A text field representing gravity's value

## 3.2.3.3 Leniency

TMP\_InputField DebugPanel.Leniency

The leniency for the **ZController**'s check for no motion.

## 3.2.3.4 Move

List<TMP\_InputField> DebugPanel.Move

A list of input fields corresponding to the three values of a player's movement acceleration

## 3.2.3.5 panel

GameObject DebugPanel.panel

The parent container for all the text fields.

## 3.2.3.6 ppc

PhysicsPlayerController DebugPanel.ppc

The PhysicsPlayerController the fields are referencing.

#### 3.2.3.7 RCooldown

TMP\_InputField DebugPanel.RCooldown

A text field representing how long the player must wait after burning out the rewind before they can use it again.

#### 3.2.3.8 RewindSpeed

TMP\_InputField DebugPanel.RewindSpeed

A text field representing how often per second a rewind capture is performed

#### 3.2.3.9 RLimit

TMP\_InputField DebugPanel.RLimit

A text field representing how long rewind can be held before time resumes.

## 3.2.3.10 Speed

List<TMP\_InputField> DebugPanel.Speed

A list of input fields corresponding to the three values of a player's max velocity

## 3.2.3.11 zc

ZController DebugPanel.zc

The **ZController** the fields are referencing

The documentation for this class was generated from the following file:

• Assets/Scripts/UI/DebugPanel.cs

# 3.3 FallingRocksObstacle Class Reference

## **Public Attributes**

- float moveSpeed = 0.05f
- float timeout = 2f

## **Private Member Functions**

- void Start ()
   place script on the prefab that SpawningRocks script will spawn.
- void FixedUpdate ()
- IEnumerator Timeout ()

## 3.3.1 Member Function Documentation

#### 3.3.1.1 Timeout()

```
IEnumerator FallingRocksObstacle.Timeout ( ) [inline], [private]
```

destroys obstacles after they have already fallen, after they already

The documentation for this class was generated from the following file:

• Assets/Scripts/FallingRocksObstacle.cs

## 3.4 FirstPersonLook Class Reference

## **Public Attributes**

- float turnSpeed = 1
- float mouseXSensitiviy = 1
- float mouseYSensitiviy = 1
- Transform cam
- Transform orientation
- · Transform player
- Transform playerObj

## **Private Member Functions**

- void Start ()
- void Update ()
- void MyInput ()
- · void rotateCamera ()
- void TurnPlayer2 ()
- void TurnPlayer1 ()
- void resetTurn ()

#### **Private Attributes**

- Vector2 turn
- float horizontalInput
- · float veritcalInput

The documentation for this class was generated from the following file:

Assets/Scripts/TestMovement/FirstPersonLook.cs

## 3.5 IRewinder Class Reference

An abstraction of the rewinder functionality for use with a ZController to rewind an object.

#### **Public Member Functions**

· virtual void Start ()

Initializes SnapState list, sets initial position and rotation, and extracts the ZController from parent if it was not explicitly set. Finally, pauses the object, awaiting the ZController's signal.

virtual bool NeedUpdate ()

Checks to see if the object has moved since the last state capture. Uses ZController.Approximate(Vector3, Vector3) to compare the previous state's position with the current's. If they are not equivalent within the range of leniency, then a signal is sent stating that an update is required.

virtual void Store ()

Stores the current position and rotation of the object on the rewind stack.

virtual SnapState RewindState ()

Rewinds the states from the stack frame one state at a time, allowing for easier speed-up of rewind. If there are no states left, the function returns null

virtual void Reset ()

Resets the object to its initial position and rotation and clears the SnapState stack.

abstract void Play ()

Continues time wherever it was left off, however the concrete implementation needs to accomplish this.

• abstract void Pause ()

Pauses the movement of the object however the concrete implementation needs to accomplish this.

• bool HasStates ()

Checks whether the SnapState stack contains elements.

## **Public Attributes**

ZController zController

Parent ZController of the current GameObject; Primarily for use of ZController. Approximate (Vector3, Vector3) method.

#### **Protected Attributes**

Stack < SnapState > states

Stack of SnapState objects; used to stash and then unwind different points within the IRewinder's lifecycle over the course of the ZController's rewind capture.

bool printDebug

Debug variable to toggle output in the debug console; in child methods, whenever debug is required, wrap it with an if statment checking this field.

## **Properties**

```
    Vector3 startPos [get, protected set]
        Initial position state of the attachee GameObject

    Quaternion startRot [get, protected set]
        Initial rotation state of the attachee GameObject
```

## 3.5.1 Detailed Description

An abstraction of the rewinder functionality for use with a ZController to rewind an object.

#### 3.5.2 Member Function Documentation

## 3.5.2.1 HasStates()

```
bool IRewinder.HasStates ( ) [inline]
```

Checks whether the SnapState stack contains elements.

Returns

## 3.5.2.2 NeedUpdate()

```
virtual bool IRewinder.NeedUpdate ( ) [inline], [virtual]
```

Checks to see if the object has moved since the last state capture. Uses ZController.Approximate(Vector3, Vector3) to compare the previous state's position with the current's. If they are not equivalent within the range of leniency, then a signal is sent stating that an update is required.

Returns

True if the object has moved

## 3.5.2.3 Pause()

```
abstract void IRewinder.Pause ( ) [pure virtual]
```

Pauses the movement of the object however the concrete implementation needs to accomplish this.

Implemented in PhysicsRewinder.

#### 3.5.2.4 Play()

```
abstract void IRewinder.Play ( ) [pure virtual]
```

Continues time wherever it was left off, however the concrete implementation needs to accomplish this.

Implemented in PhysicsRewinder.

#### 3.5.2.5 Reset()

```
virtual void IRewinder.Reset ( ) [inline], [virtual]
```

Resets the object to its initial position and rotation and clears the SnapState stack.

## 3.5.2.6 RewindState()

```
virtual SnapState IRewinder.RewindState ( ) [inline], [virtual]
```

Rewinds the states from the stack frame one state at a time, allowing for easier speed-up of rewind. If there are no states left, the function returns null

Returns

The popped SnapState or null if there are no states remaining.

Reimplemented in PhysicsRewinder.

#### 3.5.2.7 Start()

```
virtual void IRewinder.Start ( ) [inline], [virtual]
```

Initializes SnapState list, sets initial position and rotation, and extracts the ZController from parent if it was not explicitly set. Finally, pauses the object, awaiting the ZController's signal.

Reimplemented in PhysicsRewinder.

#### 3.5.2.8 Store()

```
virtual void IRewinder.Store ( ) [inline], [virtual]
```

Stores the current position and rotation of the object on the rewind stack.

Reimplemented in PhysicsRewinder.

## 3.5.3 Member Data Documentation

## 3.5.3.1 printDebug

```
bool IRewinder.printDebug [protected]
```

Debug variable to toggle output in the debug console; in child methods, whenever debug is required, wrap it with an if statment checking this field.

## 3.5.3.2 states

```
Stack<SnapState> IRewinder.states [protected]
```

Stack of SnapState objects; used to stash and then unwind different points within the IRewinder's lifecycle over the course of the ZController's rewind capture.

#### 3.5.3.3 zController

```
ZController IRewinder.zController
```

Parent ZController of the current GameObject; Primarily for use of ZController.Approximate(Vector3, Vector3) method.

## 3.5.4 Property Documentation

#### 3.5.4.1 startPos

```
Vector3 IRewinder.startPos [get], [protected set]
```

Initial position state of the attachee GameObject

#### 3.5.4.2 startRot

```
Quaternion IRewinder.startRot [get], [protected set]
```

Initial rotation state of the attachee GameObject

The documentation for this class was generated from the following file:

Assets/Scripts/Rewinders/IRewinder.cs

## 3.6 LevelLoader Class Reference

## **Public Member Functions**

- IEnumerator LoadLevel (int sceneIndex)
- · void ChangeScene (int sceneIndex)
- void NextScene ()
- void ReloadScene ()

## **Public Attributes**

- List< Scene > SceneList
- Animator Transition
- float TransitionTime = 1

#### **Private Member Functions**

• void Start ()

The documentation for this class was generated from the following file:

· Assets/Scripts/Transitions/LevelLoader.cs

# 3.7 LevelTransitionTrigger Class Reference

#### **Public Member Functions**

void OnTriggerEnter (Collider other)

## **Public Attributes**

- LevelLoader ILoad
- bool TargetScene
- int **SelectedScene** = 0

The documentation for this class was generated from the following file:

• Assets/Scripts/Transitions/LevelTransitionTrigger.cs

# 3.8 MenuScript Class Reference

## **Public Attributes**

GameObject panel

## **Private Member Functions**

• void Update ()

The documentation for this class was generated from the following file:

· Assets/Scripts/UI/MenuScript.cs

# 3.9 MovementScript Class Reference

## **Public Member Functions**

• IEnumerator WaitForNextStep ()

## **Public Attributes**

- Transform orientation
- · Rigidbody rb
- AudioSource audioSource
- AudioSource timeAudio
- float moveSpeed = 1
- float jumpForce
- · float jumpCooldown
- · float airMultiplier
- bool readyToJump
- AudioClip[] walkSounds
- AudioClip timeRewindSound
- float waitStep = .5f
- KeyCode jumpKey = KeyCode.Space
- float playerHeight
- · float groundDrag
- · LayerMask whatIsGround
- bool grounded

#### **Private Member Functions**

- void Start ()
- void Update ()
- void FixedUpdate ()
- · void MyInput ()
- · void MovePlayer ()
- void SpeedControl ()
- void Jump ()
- void ResetJump ()

#### **Private Attributes**

- · int walkSoundIndex
- bool playSound = true
- · float horizontalInput
- · float verticalInput
- · Vector3 moveDir

The documentation for this class was generated from the following file:

· Assets/Scripts/TestMovement/MovementScript.cs

# 3.10 PhysicsPlayerController Class Reference

A Rigidbody based player controller.

#### **Public Member Functions**

• void UpdateCamera ()

Updates camera rotations using player mouse input. Player GameObject is also transformed for y rotations, allowing forward to always be where the player is facing.

void DoMovement (ref float xVel, ref float zVel)

Calculates the player's x and z direction movement. Movement is converted into a Rigidbody force using the mass and the playerAcceleration for that direction. If the current velocity is higher than the max, however, it will be reduced to the max and stored in xVel and zVel.

void DoJump (ref float yVel)

Calculates the player's vertical velocity and processes jump input. Checks for ground using a raycast, then if the player is on the ground and jump is pressed enacts an impulse force upwards using the value specified in playerAcceleration. If not on the ground and the velocity is greater than the max speed, yVel will be set to the value in maxSpeed.

• void OnCollisionEnter (Collision collision)

Detects a collision with the player and calculates the force of the impact. If the impact is too high, the player will be killed.

## **Public Attributes**

Vector3 playerAcceleration = new Vector3(10, 10, 20)

A Vector3 containing the accelleration value for the player in each direction.

• Vector3 maxSpeed = new Vector3(3, 15, 5)

A Vector3 containing the maximum absolute velocity that the player can go in any direction.

· bool grounded

A bool representing whether or not the player is currently touching a ground.

Transform cam

The Transform for the player's main Camera.

• float mouseXSensitiviy = 1

The mouse sensitivity for the camera going up and down.

• float mouseYSensitiviy = 1

The mouse sensitivity for the camera going side to side.

## **Private Member Functions**

• void Start ()

Attempts to set the Rigidbody and camera Transform if not set within the editor, and locks the CursorState.

• void Update ()

Updates the player's movement based on input; calls UpdateCamera, DoMovement(ref float, ref float), and DoJump(ref float).

Also applies velocity cap.

## **Private Attributes**

· Rigidbody rigidBody

The the attachee GameObject's Rigidbody.

- · LayerMask ground
- · Vector2 turn

A Vector2 keeping track of the current x and y rotations of the player camera.

## 3.10.1 Detailed Description

A Rigidbody based player controller.

## 3.10.2 Member Function Documentation

#### 3.10.2.1 DoJump()

```
void PhysicsPlayerController.DoJump (
ref float yVel ) [inline]
```

Calculates the player's vertical velocity and processes jump input. Checks for ground using a raycast, then if the player is on the ground and jump is pressed enacts an impulse force upwards using the value specified in playerAcceleration. If not on the ground and the velocity is greater than the max speed, *yVel* will be set to the value in maxSpeed.

## **Parameters**

```
yVel A reference to the z velocity of the player; will be capped to the maxSpeed value if too high
```

## 3.10.2.2 DoMovement()

Calculates the player's x and z direction movement. Movement is converted into a Rigidbody force using the mass and the playerAcceleration for that direction. If the current velocity is higher than the max, however, it will be reduced to the max and stored in xVel and zVel.

#### **Parameters**

xVel	A reference to the x velocity of the player; will be capped to the maxSpeed value if too high
zVel	A reference to the z velocity of the player; will be capped to the maxSpeed value if too high

## 3.10.2.3 OnCollisionEnter()

Detects a collision with the player and calculates the force of the impact. If the impact is too high, the player will be killed.

#### 3.10.2.4 Start()

```
void PhysicsPlayerController.Start ( ) [inline], [private]
```

Attempts to set the Rigidbody and camera Transform if not set within the editor, and locks the CursorState.

## 3.10.2.5 Update()

```
void PhysicsPlayerController.Update ( ) [inline], [private]
```

Updates the player's movement based on input; calls UpdateCamera, DoMovement(ref float, ref float), and DoJump(ref float).

Also applies velocity cap.

## 3.10.2.6 UpdateCamera()

```
void PhysicsPlayerController.UpdateCamera ( ) [inline]
```

Updates camera rotations using player mouse input. Player GameObject is also transformed for y rotations, allowing forward to always be where the player is facing.

## 3.10.3 Member Data Documentation

#### 3.10.3.1 cam

Transform PhysicsPlayerController.cam

The Transform for the player's main Camera.

## 3.10.3.2 grounded

 $\verb|bool PhysicsPlayerController.grounded|\\$ 

A bool representing whether or not the player is currently touching a ground.

#### 3.10.3.3 maxSpeed

Vector3 PhysicsPlayerController.maxSpeed = new Vector3(3, 15, 5)

A Vector3 containing the maximum absolute velocity that the player can go in any direction.

## 3.10.3.4 mouseXSensitiviy

float PhysicsPlayerController.mouseXSensitiviy = 1

The mouse sensitivity for the camera going up and down.

## 3.10.3.5 mouseYSensitiviy

float PhysicsPlayerController.mouseYSensitiviy = 1

The mouse sensitivity for the camera going side to side.

#### 3.10.3.6 playerAcceleration

Vector3 PhysicsPlayerController.playerAcceleration = new Vector3(10, 10, 20)

A Vector3 containing the accelleration value for the player in each direction.

#### 3.10.3.7 rigidBody

Rigidbody PhysicsPlayerController.rigidBody [private]

The the attachee GameObject's Rigidbody.

#### 3.10.3.8 turn

Vector2 PhysicsPlayerController.turn [private]

A Vector2 keeping track of the current x and y rotations of the player camera.

The documentation for this class was generated from the following file:

· Assets/Scripts/PhysicsPlayerController.cs

## 3.11 PhysicsRewinder Class Reference

An implementation of IRewinder made for operating with Rigidbody physics.

#### **Public Member Functions**

override void Start ()

Initializes the Rigidbody if not set in the editor.

Initializes SnapState list, sets initial position and rotation, and extracts the ZController from parent if it was not explicitly set. Finally, pauses the object, awaiting the ZController's signal.

• override void Store ()

Stores the current position and rotation of the object on the rewind stack.

Also stores the Rigidbody's current velocity and current angular velocity.

override SnapState RewindState ()

Rewinds the states from the stack frame one state at a time, allowing for easier speed-up of rewind. If there are no states left, the function returns null

Sets the rewind velocity to the popped SnapState's velocity.

· void Update ()

Updates the rewindVelocity every frame, so long as the the Rigidbody is not in kinematic mode.

• override void Play ()

Continues physics for the object, and begins by applying the force if there is no history within the SnapState stack. If there is a state within the stack, attempts to resume movement by applying the previous state's velocity

• override void Pause ()

Halts physics for the object and sets the Rigidbody to kinematic mode.

• void OnCollisionStay (Collision collision)

In the case of collision while the GameObject is rewinding, a force needs to be applied to the colliding object since the Rigidbody cannot do it for itself in kinematic mode.

## **Public Attributes**

• Vector3 startForce = Vector3.zero

The initial force that should be applied to the Rigidbody when time starts playing.

· Rigidbody rb

The attachee GameObject's attached Rigidbody.

#### **Private Attributes**

· Vector3 rewindVelocity

A cached state of the previously popped SnapState's velocity; used for calculating the force imparted onto any non-rewound object while rewinding time (primarily the plyaer)

## **Additional Inherited Members**

## 3.11.1 Detailed Description

An implementation of IRewinder made for operating with Rigidbody physics.

#### 3.11.2 Member Function Documentation

#### 3.11.2.1 OnCollisionStay()

In the case of collision while the GameObject is rewinding, a force needs to be applied to the colliding object since the Rigidbody cannot do it for itself in kinematic mode.

Using the difference between the next state's velocity and the cached rewind velocity, the delta is multiplied by the Rigidbody's mass and applied to the colliding Rigidbody as a force. In this implementation, the change in time is ignored because it produces better results.

#### **Parameters**

```
collision A Collision containing data about the colliding body.
```

#### 3.11.2.2 Pause()

```
override void PhysicsRewinder.Pause ( ) [inline], [virtual]
```

Halts physics for the object and sets the Rigidbody to kinematic mode.

Implements IRewinder.

## 3.11.2.3 Play()

```
override void PhysicsRewinder.Play ( ) [inline], [virtual]
```

Continues physics for the object, and begins by applying the force if there is no history within the SnapState stack. If there is a state within the stack, attempts to resume movement by applying the previous state's velocity

Implements IRewinder.

#### 3.11.2.4 RewindState()

```
override SnapState PhysicsRewinder.RewindState ( ) [inline], [virtual]
```

Rewinds the states from the stack frame one state at a time, allowing for easier speed-up of rewind. If there are no states left, the function returns null

Sets the rewind velocity to the popped SnapState's velocity.

## Returns

Rewinds the states from the stack frame one state at a time, allowing for easier speed-up of rewind. If there are no states left, the function returns null

Reimplemented from IRewinder.

#### 3.11.2.5 Start()

```
override void PhysicsRewinder.Start ( ) [inline], [virtual]
```

Initializes the Rigidbody if not set in the editor.

Initializes SnapState list, sets initial position and rotation, and extracts the ZController from parent if it was not explicitly set. Finally, pauses the object, awaiting the ZController's signal.

Reimplemented from IRewinder.

## 3.11.2.6 Store()

```
override void PhysicsRewinder.Store ( ) [inline], [virtual]
```

Stores the current position and rotation of the object on the rewind stack.

Also stores the Rigidbody's current velocity and current angular velocity.

Reimplemented from IRewinder.

#### 3.11.2.7 Update()

```
void PhysicsRewinder.Update ( ) [inline]
```

Updates the rewindVelocity every frame, so long as the the Rigidbody is not in kinematic mode.

#### 3.11.3 Member Data Documentation

#### 3.11.3.1 rb

```
Rigidbody PhysicsRewinder.rb
```

The attachee GameObject's attached Rigidbody.

## 3.11.3.2 rewindVelocity

```
Vector3 PhysicsRewinder.rewindVelocity [private]
```

A cached state of the previously popped SnapState's velocity; used for calculating the force imparted onto any non-rewound object while rewinding time (primarily the plyaer)

## 3.11.3.3 startForce

```
Vector3 PhysicsRewinder.startForce = Vector3.zero
```

The initial force that should be applied to the Rigidbody when time starts playing.

The documentation for this class was generated from the following file:

• Assets/Scripts/Rewinders/PhysicsRewinder.cs

## 3.12 PlayerController Class Reference

Modified from unity docs: https://docs.unity3d.com/ScriptReference/Character← Controller.Move.html To be removed in later versions.

#### **Public Member Functions**

IEnumerator WaitForNextStep ()

## **Public Attributes**

- CharacterController controller
- float playerSpeed = 5.0f
- float jumpHeight = 2.0f
- float MAX\_VELOCITY\_TOLERANCE = 100
- Transform cam
- float turnSmoothnessTime = 0.1f
- float mouseXSensitiviy = 1
- float mouseYSensitiviy = 1
- AudioSource audioSource
- AudioClip[] walkSounds
- AudioSource timeAudio
- AudioClip timeRewindSound
- float waitStep = .5f

## **Private Member Functions**

- · void Start ()
- · void Update ()
- void **OnTriggerEnter** (Collider other)

## **Private Attributes**

- Vector3 playerVelocity
- bool groundedPlayer
- float gravityValue = -9.81f
- Vector2 turn
- float turnSmoothSpeed
- int walkSoundIndex
- bool playSound = true

## 3.12.1 Detailed Description

Modified from unity docs: https://docs.unity3d.com/ScriptReference/Character← Controller.Move.html To be removed in later versions.

The documentation for this class was generated from the following file:

Assets/Scripts/PlayerController.cs

# 3.13 RoomController Class Reference

## **Private Member Functions**

- · void Start ()
- · void Update ()

## **Private Attributes**

- IEnumerable < Animator > animators
- float animTime
- float maxTime

The documentation for this class was generated from the following file:

• Assets/Scripts/RoomController.cs

## 3.14 RoomSettingsScript Class Reference

#### **Private Member Functions**

· void Start ()

## **Private Attributes**

Color fogColor

The documentation for this class was generated from the following file:

• Assets/Scripts/RoomSettingsScript.cs

# 3.15 SnapState Class Reference

A representation of a gameObject's current state.

## **Public Attributes**

- · Vector3 position
- Quaternion rotation
- · Vector3 velocity
- Vector3 angularVelocity

## 3.15.1 Detailed Description

A representation of a gameObject's current state.

Currently contains

- · Position
- Rotation
- · Velocity
- · Angular Velocity

The documentation for this class was generated from the following file:

Assets/Scripts/Rewinders/IRewinder.cs

## 3.16 SpawningRocks Class Reference

## **Public Attributes**

· GameObject rockPrefab

#### **Private Member Functions**

- void Start ()
- IEnumerator SpawnFalling ()

The documentation for this class was generated from the following file:

· Assets/Scripts/SpawningRocks.cs

## 3.17 TextureResizeScript Class Reference

## **Private Member Functions**

· void Start ()

## **Private Attributes**

- float resize = 4
- float scale = 1
- Transform t
- Renderer r
- · Material m

The documentation for this class was generated from the following file:

Assets/Scripts/TextureResizeScript.cs

## 3.18 ThirdPersonMovement Class Reference

## **Public Attributes**

- CharacterController \_controller
- Transform cam
- float walkSpeed = 5f
- float turnSmoothnessTime = 0.1f

## **Private Member Functions**

• void Update ()

## **Private Attributes**

· float turnSmoothSpeed

The documentation for this class was generated from the following file:

• Assets/Scripts/ThirdPersonMovement.cs

## 3.19 TriggerWalls Class Reference

## **Public Attributes**

- GameObject[] targets
- bool Disable = false

## **Private Member Functions**

· void OnTriggerEnter (Collider other)

The documentation for this class was generated from the following file:

· Assets/Scripts/TriggerWalls.cs

# 3.20 TriigerWinLoss Class Reference

## **Public Attributes**

- GameObject WinLossScreen
- bool WinTrigger = false
- bool LoseTrigger = false
- string **TriggerMessage** = "Unspecified Trigger"

#### **Private Member Functions**

- void Start ()
- · void OnTriggerEnter (Collider other)

#### **Private Attributes**

TMPro.TextMeshProUGUI txt

The documentation for this class was generated from the following file:

• Assets/Scripts/TriigerWinLoss.cs

# 3.21 VignetteRewind Class Reference

## **Private Member Functions**

- · void Update ()
- · void VignetteOn ()
- void VignetteOff ()

## **Private Attributes**

- · Volume volume
- float \_intensity

The documentation for this class was generated from the following file:

• Assets/Scripts/VignetteRewind.cs

## 3.22 ZController Class Reference

A script to control all IRewinders in the current GameObject's children and process their state capture, motion, and rewind.

## **Public Member Functions**

· void Pause ()

Pauses all IRewinders under this controller's control.

• bool Approximate (Vector3 a, Vector3 b)

Checks if two vectors are approximately equivalent with approximateLeniency leniency.

#### **Public Attributes**

• float recordInterval = .2f

The interval at which a record state should be taken in seconds.

• float rewindScale = 1f

A measure of how quickly the rewinds should occur as a multiplier

float rewindLimit = -1f

How long in seconds a rewind can be held at a fully rewound state before "burning out" and needing to cooldown. -1 is disabled

float rewindCooldown = 2f

How long the player should be made to wait until they can use the rewind again after burning it out.

float approximateLeniency = .01f

A measure of how close to zero a vector needs to be for a child IRewinder to consider it to be zero; use primarily for IRewinder.NeedUpdate.

· bool active

A boolean to determine whether or not this controller is active.

#### **Private Member Functions**

• void Start ()

Initialize the time and extract the children IRewinders. If the rewind scale is below zero, set it to 1 to prevent crashes.

· void Update ()

Handle all rewind code.

- If the player is doing a rewind, then iterate through the rewind states for each IRewinder
- If all states have been rewound and the rewind limit is reached, set the cooldown and play motion.
- If there is an IRewinder that is still moving, all IRewinders capture their current states.
- If the player releases the rewind, then play all IRewinders, allowing them to continue motion.
- void OnTriggerEnter (Collider other)

Allows for multiple controllers to exist within the same scene, setting activation to a bound set of triggers. Whenever the trigger attached to this GameObject is entered, all other controllers will be paused.

## **Private Attributes**

• IEnumerable < IRewinder > rewinds

A collection of all IRewinders within this controller's influence.

float deltaTime

The amount of time that has passed since an update; used to calculate the record interval and the rewind limit

• float cooldown = 0f

Keeps track of how long the cooldown has left before players can use rewind again.

## 3.22.1 Detailed Description

A script to control all IRewinders in the current GameObject's children and process their state capture, motion, and rewind.

Because like, control + z, get it? Because it undoes?

#### 3.22.2 Member Function Documentation

## 3.22.2.1 Approximate()

```
bool ZController.Approximate ( \label{eq:controller} \mbox{Vector3 a,} \mbox{ Vector3 b ) [inline]}
```

Checks if two vectors are approximately equivalent with approximateLeniency leniency.

#### **Parameters**

а	First vector
b	Second vector

#### Returns

True if the difference between the vectors is less than approximateLeniency

## 3.22.2.2 OnTriggerEnter()

Allows for multiple controllers to exist within the same scene, setting activation to a bound set of triggers. Whenever the trigger attached to this GameObject is entered, all other controllers will be paused.

## 3.22.2.3 Pause()

```
void ZController.Pause ( ) [inline]
```

Pauses all IRewinders under this controller's control.

## 3.22.2.4 Start()

```
void ZController.Start ( ) [inline], [private]
```

Initialize the time and extract the children IRewinders. If the rewind scale is below zero, set it to 1 to prevent crashes.

## 3.22.2.5 Update()

```
void ZController.Update ( ) [inline], [private]
```

Handle all rewind code.

- · If the player is doing a rewind, then iterate through the rewind states for each IRewinder
- If all states have been rewound and the rewind limit is reached, set the cooldown and play motion.
- If there is an IRewinder that is still moving, all IRewinders capture their current states.
- If the player releases the rewind, then play all IRewinders, allowing them to continue motion.

#### 3.22.3 Member Data Documentation

#### 3.22.3.1 active

```
bool ZController.active
```

A boolean to determine whether or not this controller is active.

The Unity version won't work because it disables the update methods when the object is inactive.

## 3.22.3.2 approximateLeniency

```
float ZController.approximateLeniency = .01f
```

A measure of how close to zero a vector needs to be for a child IRewinder to consider it to be zero; use primarily for IRewinder.NeedUpdate.

#### 3.22.3.3 cooldown

```
float ZController.cooldown = Of [private]
```

Keeps track of how long the cooldown has left before players can use rewind again.

#### 3.22.3.4 deltaTime

```
float ZController.deltaTime [private]
```

The amount of time that has passed since an update; used to calculate the record interval and the rewind limit

## 3.22.3.5 recordInterval

```
float ZController.recordInterval = .2f
```

The interval at which a record state should be taken in seconds.

#### 3.22.3.6 rewindCooldown

```
float ZController.rewindCooldown = 2f
```

How long the player should be made to wait until they can use the rewind again after burning it out.

#### 3.22.3.7 rewindLimit

```
float ZController.rewindLimit = -1f
```

How long in seconds a rewind can be held at a fully rewound state before "burning out" and needing to cooldown.
-1 is disabled

## 3.22.3.8 rewinds

```
IEnumerable<IRewinder> ZController.rewinds [private]
```

A collection of all IRewinders within this controller's influence.

#### 3.22.3.9 rewindScale

```
float ZController.rewindScale = 1f
```

A measure of how quickly the rewinds should occur as a multiplier

The documentation for this class was generated from the following file:

· Assets/Scripts/ZController.cs

# Index

active	HasStates, 12
ZController, 32	NeedUpdate, 12
Approximate	Pause, 12
ZController, 30	Play, 12
approximateLeniency	printDebug, 14
ZController, 32	Reset, 13
AudioController, 5	RewindState, 13
Update, 6	Start, 13
WaitForNextStep, 6	startPos, 14
	startRot, 14
cam	states, 14
PhysicsPlayerController, 20	Store, 13
cooldown	zController, 14
ZController, 32	,
	Leniency
DebugPanel, 6	DebugPanel, 8
Drag, 7	LevelLoader, 15
Gravity, 8	LevelTransitionTrigger, 15
Leniency, 8	55 7
Move, 8	maxSpeed
panel, 8	PhysicsPlayerController, 20
ppc, 8	MenuScript, 15
RCooldown, 8	mouseXSensitiviy
RewindSpeed, 9	PhysicsPlayerController, 20
RLimit, 9	mouseYSensitiviy
Speed, 9	PhysicsPlayerController, 20
Start, 7	Move
Update, 7	DebugPanel, 8
zc, 9	MovementScript, 16
deltaTime	
ZController, 32	NeedUpdate
DoJump	IRewinder, 12
PhysicsPlayerController, 18	
DoMovement	OnCollisionEnter
PhysicsPlayerController, 18	PhysicsPlayerController, 19
Drag	OnCollisionStay
DebugPanel, 7	PhysicsRewinder, 22
Debugranei, 7	OnTriggerEnter
FallingRocksObstacle, 9	ZController, 31
Timeout, 10	
FirstPersonLook, 10	panel
That ersoneoux, To	DebugPanel, 8
Gravity	Pause
DebugPanel, 8	IRewinder, 12
grounded	PhysicsRewinder, 22
PhysicsPlayerController, 20	ZController, 31
. Hydiodi layor dontrollor, 20	PhysicsPlayerController, 17
HasStates	cam, <mark>20</mark>
IRewinder, 12	DoJump, 18
	DoMovement, 18
IRewinder, 11	grounded, 20

36 INDEX

maxSpeed, 20	RoomController, 26
mouseXSensitiviy, 20	RoomSettingsScript, 26
	ricomocilingscorpt, 20
mouseYSensitiviy, 20	ChanCtata OC
OnCollisionEnter, 19	SnapState, 26
playerAcceleration, 20	SpawningRocks, 27
rigidBody, 21	Speed
Start, 19	DebugPanel, 9
	Start
turn, 21	
Update, 19	DebugPanel, 7
UpdateCamera, 19	IRewinder, 13
PhysicsRewinder, 21	PhysicsPlayerController, 19
OnCollisionStay, 22	PhysicsRewinder, 23
	ZController, 31
Pause, 22	startForce
Play, 23	
rb, 24	PhysicsRewinder, 24
RewindState, 23	startPos
rewindVelocity, 24	IRewinder, 14
Start, 23	startRot
•	IRewinder, 14
startForce, 24	•
Store, 23	states
Update, 24	IRewinder, 14
Play	Store
IRewinder, 12	IRewinder, 13
PhysicsRewinder, 23	PhysicsRewinder, 23
	ye.ee. termineer, 20
playerAcceleration	TextureResizeScript, 27
PhysicsPlayerController, 20	•
PlayerController, 25	ThirdPersonMovement, 28
ppc	Timeout
DebugPanel, 8	FallingRocksObstacle, 10
	TriggerWalls, 28
printDebug	TriigerWinLoss, 28
IRewinder, 14	1111g01 111112000, 20
ii townidor, TT	turn
intownidal, Ti	turn
rb	turn PhysicsPlayerController, 21
	PhysicsPlayerController, 21
rb PhysicsRewinder, 24	**
rb PhysicsRewinder, 24 RCooldown	PhysicsPlayerController, 21
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8	PhysicsPlayerController, 21  Update AudioController, 6
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval	PhysicsPlayerController, 21  Update AudioController, 6 DebugPanel, 7
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval	PhysicsPlayerController, 21  Update AudioController, 6 DebugPanel, 7 PhysicsPlayerController, 19 PhysicsRewinder, 24
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32	PhysicsPlayerController, 21  Update AudioController, 6 DebugPanel, 7 PhysicsPlayerController, 19 PhysicsRewinder, 24 ZController, 31
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown	PhysicsPlayerController, 21  Update AudioController, 6 DebugPanel, 7 PhysicsPlayerController, 19 PhysicsRewinder, 24 ZController, 31
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33	PhysicsPlayerController, 21  Update AudioController, 6 DebugPanel, 7 PhysicsPlayerController, 19 PhysicsRewinder, 24 ZController, 31  UpdateCamera PhysicsPlayerController, 19
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit	PhysicsPlayerController, 21  Update AudioController, 6 DebugPanel, 7 PhysicsPlayerController, 19 PhysicsRewinder, 24 ZController, 31  UpdateCamera PhysicsPlayerController, 19  VignetteRewind, 29
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33	PhysicsPlayerController, 21  Update AudioController, 6 DebugPanel, 7 PhysicsPlayerController, 19 PhysicsRewinder, 24 ZController, 31  UpdateCamera PhysicsPlayerController, 19  VignetteRewind, 29
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewinds ZController, 33 rewinds	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewinds ZController, 33 rewindScale ZController, 33	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewindScale ZController, 33 RewindSpeed	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6  zc     DebugPanel, 9
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewinds ZController, 33 rewindScale ZController, 33	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewindScale ZController, 33 RewindSpeed	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6  zc     DebugPanel, 9
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewinds ZController, 33 rewindScale ZController, 33 RewindSpeed DebugPanel, 9 RewindState	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6  zc     DebugPanel, 9  ZController, 29     active, 32
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewinds ZController, 33 rewindScale ZController, 33 RewindSpeed DebugPanel, 9 RewindState IRewinder, 13	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6  zc     DebugPanel, 9  ZController, 29     active, 32     Approximate, 30
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewinds ZController, 33 rewindScale ZController, 33 RewindSpeed DebugPanel, 9 RewindState IRewinder, 13 PhysicsRewinder, 23	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6  zc     DebugPanel, 9  ZController, 29     active, 32     Approximate, 30     approximateLeniency, 32
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewindScale ZController, 33 RewindSpeed DebugPanel, 9 RewindState IRewinder, 13 PhysicsRewinder, 23 rewindVelocity	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6  zc     DebugPanel, 9  ZController, 29     active, 32     Approximate, 30     approximateLeniency, 32     cooldown, 32
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewindScale ZController, 33 RewindSpeed DebugPanel, 9 RewindState IRewinder, 13 PhysicsRewinder, 23 rewindVelocity PhysicsRewinder, 24	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6  zc     DebugPanel, 9  ZController, 29     active, 32     Approximate, 30     approximateLeniency, 32     cooldown, 32     deltaTime, 32
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewindScale ZController, 33 RewindSpeed DebugPanel, 9 RewindState IRewinder, 13 PhysicsRewinder, 23 rewindVelocity PhysicsRewinder, 24 rigidBody	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6  zc     DebugPanel, 9  ZController, 29     active, 32     Approximate, 30     approximateLeniency, 32     cooldown, 32
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewindScale ZController, 33 RewindSpeed DebugPanel, 9 RewindState IRewinder, 13 PhysicsRewinder, 23 rewindVelocity PhysicsRewinder, 24	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6  zc     DebugPanel, 9  ZController, 29     active, 32     Approximate, 30     approximateLeniency, 32     cooldown, 32     deltaTime, 32
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewindScale ZController, 33 RewindSpeed DebugPanel, 9 RewindState IRewinder, 13 PhysicsRewinder, 23 rewindVelocity PhysicsPlayerController, 21	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6  zc     DebugPanel, 9  ZController, 29     active, 32     Approximate, 30     approximateLeniency, 32     cooldown, 32     deltaTime, 32     OnTriggerEnter, 31     Pause, 31
rb PhysicsRewinder, 24 RCooldown DebugPanel, 8 recordInterval ZController, 32 Reset IRewinder, 13 rewindCooldown ZController, 33 rewindLimit ZController, 33 rewinds ZController, 33 rewindScale ZController, 33 RewindSpeed DebugPanel, 9 RewindState IRewinder, 13 PhysicsRewinder, 23 rewindVelocity PhysicsRewinder, 24 rigidBody	PhysicsPlayerController, 21  Update     AudioController, 6     DebugPanel, 7     PhysicsPlayerController, 19     PhysicsRewinder, 24     ZController, 31  UpdateCamera     PhysicsPlayerController, 19  VignetteRewind, 29  WaitForNextStep     AudioController, 6  zc     DebugPanel, 9  ZController, 29     active, 32     Approximate, 30     approximateLeniency, 32     cooldown, 32     deltaTime, 32     OnTriggerEnter, 31

INDEX 37

```
rewindLimit, 33
rewinds, 33
rewindScale, 33
Start, 31
Update, 31
zController
IRewinder, 14
```