Three.js

Mesh
visible:Boolean

Geometry

geometry
1

BoxGeometry

Object3D

1 model

Material

material
1

MeshBasicMaterial

MeshPhongMaterial

1 redMaterial

1 yellowMaterial

RayCaster

1

Skeleton

1 skeleton

Camera
1 camera

Scene

WebGLRenderer
renderer
1

AnimationMixer

mixer 1

SpotLight
1
spotLight

AnimationAction

actions 0..*

AmbientLight

AnimationClip

cloudMesh
1

MyScene
cameraDistance:Float =3.0
camMinDistance:Float =3.0
cameraHeight:Float =1.6
cameraLookHeight:Float =0.8
cameraAspectRatio:Float =45
maxDistance:Float =6
healthBar1:THREE.Object3D
healthBar2:THREE.Object3D
musicaFondo:DOMElementObject
pauseScreen:DOMElementObject
camAux:THREE.Vector3
camNecesitaRotación:Boolean =false
constructor(canvas:DOMElementObject)
createCamera()
createLights(ambientLightIntensity:Float, ambientLightColor:Hexadecimal, spotLightIntensity:Float)
createGround(groundColor:Hexadecimal)
createSky(skyColor:Hexadecimal)
createClouds()
createGUI():GUI
createRenderer(myCanvas:DOMElementObject)
getCamera():THREE.Camera
setCameraAspect(ratio:Float)
onWindowResize()
checkAttackCollision(att:MyModel,obj:MyModel):{ Boolean, { MyBoxCollider, MyBoxCollider } }
update()
updateCamera()

Threex.keyboardstate.js

KeyboardState

keyboard
1

Tween.js

Tween

dat.gui.js

GUI

gui
1

1 robot

1
robot2

MyBoxCollider
setVisibility (visible:Boolean)
changeToRed()
changeToYellow()
checkCollision(collidableMeshList:Mesh[1..*])
constructor(sizeX:Float, sizeY:Float, sizeZ:Float)

activeHurtBoxes
0..*

0..* hitBox

0..* hurtBox

MyModel
direction:THREE.Vector3
paused:Boolean =false
cargado:Boolean =false
moveSpeed:Float =1.2
maxStepDelay:Float =0.3
color:THREE.Color
health:Float =1
necesitaRotacion:Boolean =false
animationClock:THREE.Clock
stateClock:THREE.Clock
moveClock:THREE.Clock
state.signalReceived:Boolean =false
constructor()
initModel()
initStates()
initColliders()
setBoxesVisibility(visibility:Boolean)
setColor(color:Hexadecimal)
setColor(color:THREE.Color)
deactivateHurtBoxes()
getHealth():Float
update()
getHitBoxes():MyBoxCollider[0..*]
getActiveHurtBoxes():MyBoxCollider[0..*]
receiveDamage(direction:THREE.Vector3, collision.hitBox:MyBoxCollider, collision.hurtBox:MyBoxCollider)
pegarPatada()
pegarPunietazo()
getMoveSpeed():Float
actualizarMoveClock()
actualizarStateClock()
andarAdelante()
andarAtras()
saltar()
agachar()
levantarse()
detener()
pausar()
reanudar()

state.combatState 1

<<enumeration>>
STATES
idle
right
left
up
down
golpeando
saltando
bloqueando
agachado
rotando
corriendoDcha
corriendoIzda
recuperando

1
state.lastMove

1
state.bodyState