﻿using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

public class Player : MonoBehaviour

{

[SerializeField] // allows speed to be changed within unity whilst keeping variable private

private float speed;

[SerializeField]

private float speedMultiplier;

private float rspeed = 2.0F;

private float rspeedvert = 0f;

private Vector3 direction; // vector 3 = x y z axis

public GameObject playercam;

public GameObject aqcuireChan;

private float m\_RunningStart = 0.5f;

private bool crouch = false;

private bool run = false;

private float baseHeight ;

private float baseFOV;

public float crouchFOV;

public float runFOV;

public float stamina;

public float cooldown;

public float detectability;

// Start is called before the first frame update

void Start()

{

stamina = 100;

baseFOV = playercam.GetComponent<Camera>().fieldOfView;

baseHeight = playercam.transform.position.y;

cooldown = 0;

}

// Update is called once per frame

void FixedUpdate()

{

//GetComponentInChildren<Slider>().value = stamina / 100;

playercam.transform.position = new Vector3(playercam.transform.position.x, baseHeight, playercam.transform.position.z);

speedMultiplier = 1;

GetInput();

Move();

float r = rspeed \* Input.GetAxis("Mouse X");

float rvert = rspeedvert \* Input.GetAxis("Mouse Y");

transform.Rotate(-rvert\*0, r, 0);

float ang = playercam.GetComponent<Camera>().fieldOfView;

if (crouch)

{

playercam.transform.position = new Vector3(playercam.transform.position.x, playercam.transform.position.y-0.25f, playercam.transform.position.z);

ang = Mathf.Lerp(ang, crouchFOV, 3 \* Time.deltaTime);

playercam.GetComponent<Camera>().fieldOfView = ang;

cooldown = cooldown + Time.deltaTime;

if(stamina < 100 && cooldown >=1)

{

stamina = stamina + Time.deltaTime\*25; // Crouch Stamina Recovery

}

}

if (run)

{

ang = Mathf.Lerp(ang, runFOV, 3 \* Time.deltaTime);

playercam.GetComponent<Camera>().fieldOfView = ang;

if (stamina > 0)

{

cooldown = 0;

stamina = stamina - Time.deltaTime \* 25; // Run Stamina Cost

}

}

if (!crouch && !run)

{

ang = Mathf.Lerp(ang, baseFOV, 3 \* Time.deltaTime);

playercam.GetComponent<Camera>().fieldOfView = ang;

cooldown = cooldown + Time.deltaTime;

if (stamina < 100 && cooldown >= 1)

{

stamina = stamina + Time.deltaTime\*12.5f; // Walk Stamina Recovery

}

}

//playercam.transform.Rotate(-rvert \* 0, r, 0);

}

public void Move()

{

//Rigidbody rigidbody = GetComponent<Rigidbody>();

//rigidbody.transform.Translate(direction \* speed \* Time.deltaTime \* speedMultiplier);

//MovePosition(direction \* speed \* Time.deltaTime \* speedMultiplier);

transform.Translate(direction \* speed \* Time.deltaTime \* speedMultiplier);

}

private void GetInput() // sets variable direction based on the player input

{

direction = Vector3.zero; // resets direction every loop

/\*

if (Input.GetKey(KeyCode.W))

{

direction += Vector3.forward;

}

if (Input.GetKey(KeyCode.A))

{

direction += Vector3.left;

}

if (Input.GetKey(KeyCode.S))

{

direction += Vector3.back;

}

if (Input.GetKey(KeyCode.D))

{

direction += Vector3.right;

}

\*/

float vert = Input.GetAxis("Vertical");

float hor = Input.GetAxis("Horizontal");

direction = new Vector3(hor, 0f, vert);

if (Input.GetKey(KeyCode.LeftShift) && !Input.GetKey(KeyCode.LeftControl) && !Input.GetKey(KeyCode.S) && stamina > 0)

{

speedMultiplier = 4;

run = true;

}

if (!Input.GetKey(KeyCode.LeftShift) || Input.GetKey(KeyCode.S) || stamina <= 0)

{

run = false;

}

if (Input.GetKey(KeyCode.LeftControl))

{

crouch = true;

speedMultiplier = 0.5f;

}

if (!Input.GetKey(KeyCode.LeftControl))

{

crouch = false;

}

}

//TESTING AQCUIRE ANIMATION ON CAPSULE

private Animator m\_Animator = null;

private float m\_MoveTime = 0;

private float m\_MoveSpeed = 0.0f;

private bool m\_IsGround = true;

private void Awake()

{

m\_Animator = this.GetComponentInChildren<Animator>();

m\_MoveSpeed = speed;

}

private void Update()

{

if (null == m\_Animator) return;

// check ground

float rayDistance = 0.3f;

Vector3 rayOrigin = (this.transform.position + (Vector3.up \* rayDistance \* 0.5f));

bool ground = Physics.Raycast(rayOrigin, Vector3.down, rayDistance, LayerMask.GetMask("Default"));

if (ground != m\_IsGround)

{

m\_IsGround = ground;

}

// input

float h = Input.GetAxis("Horizontal");

float v = Input.GetAxis("Vertical");

bool isMove = ((0 != h) || (0 != v));

m\_MoveTime = isMove ? (m\_MoveTime + Time.deltaTime) : 0;

bool isRun = ((m\_RunningStart <= m\_MoveTime) && Input.GetKey(KeyCode.LeftShift));

// move speed (walk / run)

float moveSpeed = isRun ? speedMultiplier : speed;

m\_MoveSpeed = isMove ? Mathf.Lerp(m\_MoveSpeed, moveSpeed, (8.0f \* Time.deltaTime)) : speed;

// m\_MoveSpeed = moveSpeed;

if (isMove)

{

}

m\_Animator.SetBool("isMove", isMove);

if (Input.GetKey(KeyCode.LeftShift) && !Input.GetKey(KeyCode.LeftControl) && !Input.GetKey(KeyCode.S) && stamina>0)

{

m\_Animator.SetBool("isRun", isRun);

}

if (!Input.GetKey(KeyCode.LeftShift) || Input.GetKey(KeyCode.S) || Input.GetKey(KeyCode.LeftControl) || stamina<=0)

{

m\_Animator.SetBool("isRun", false);

}

//Turn Aqcuire in direction of movement

/\*Vector3 rotdif = transform.rotation.eulerAngles.normalized - direction.normalized;

if (rotdif != Vector3.zero)

{

aqcuireChan.transform.rotation = Quaternion.LookRotation(rotdif, new Vector3(0, 1, 0));

}

\*/

}

}