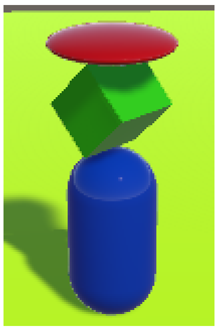
Lab URL: https://hub.labs.coursera.org:443/connect/sharedvwdrazml?forceRefresh=false

Task 1: Create a hand animation for an object



Initially for this task I wanted to create a keyframe animation for mixamo character however, unity does not support humanoid for keyframe animation and before that I tried it doesn't allow me to record keyframe changes for the humanoid. So I have created a simple object for this task which is a player object that consist of a sphere(hat), cube(head) and capsule(body) and the colour I used for this object is red, green, and blue(rgb which is the 3 colours used for monitor display).

I have created 3 key animation which is Idle, SwingAround and Walk animation. For idle animation, it changes the body, hat, head rotation and hat and head position for every half a second. The idle animation last for 3.5 seconds. For swingAround animation, it changes the body and head rotation for every half a second. The swingAround animation last for 2.5 seconds. For walk animation, it changes the body, head, hat rotation and hat, head position for every 0.1 second. The walk animation last for 0.5 second.

Task 2: Create an animation state machine for an object that is controlled by keyboard (or other input) or an animation timeline for a simple animated movie

For this task, I have created a state machine which have a default animation of idle. From the idle animation, it can go to the swing around animation or the walk animation. From the walk animation, it can go to the idle animation or swing around animation. From the swing around animation it can only go back to the idle animation. To trigger the walk animation, simply press "w" to move forward, "a" to move left, "s" to move backward, "d" to move right and spacebar to jump. To trigger the swing around animation, just press left click when in idle or walk state. The screenshot of the code can be found here. All the animation and code written are knowledge learnt from coursera.

Extended Task: Animate a human like character

For this task, I used youtube tutorial with coursera provided code. I chose the youtube video code than the coursera code because is a bit complex for my understanding so I sticked with the youtube tutorial. The screenshot of the code can be found here. For the character to look at the player object, we must attach the player object to objTarget and enable IK pass in the animator controller. From the start function, we get the animator component and create a game object named DummyPivot on the character object. This DummyPivot will only create when the scene is first run and will be removed after stopping the scene. The DummyPivot have to shift it's position to align with the character head so it will at eye level of the character. Next OnAnimatorIK function, we lookat our player object and get it's y value. So if the y value is above 0.7f, the lookWeight value will be linearly interpolate to 1 else it will linearly interpolate to 0. The reason for this if statement is to check if the targeted object is in sight, if it is behind it's visible sight, the character will not look at the player object. As for why the pivotRotY must be bigger than 0.7f, this value is found by adding a Debug.Log and drag around the player object in the scene to find where is the max value of it's left and right. After checking the pivotRotY and having the lookWeight value, we can check if we have a valid animator component and if we have, we check if the objTarget is not null and if is not null we will set the lookWeight and objTarget position to the animator. If we do not have a valid animator, we will set the lookWeight to 0 which is facing front. Up to this point, the character will always look at our player which is not very human like.

So with the coursera code, I have implemented the a timer to make the character look away and look at the player object. So in order to let the character look at the player and look away, we have to set a default time for look at and look away which is 5 and 3. Before that, the character I used is doozy with breathing idle animation. I have added LookAtTime and LookAwayTime into the curves of the animation property. I have attached the idle animation as the base animation in the state machine and added float LookAtTime and LookAwayTime as the parameters. So in the start function of the code, I set the timer to have the value of lookAwayTime and get the id of the 2 parameters and store it into lookAtTimeParamId and lookAwayTimeParamId. In the update function is just deducting the timer. In the OnAnimatorIK function, if the timer is bigger than 0, it will set the lookWeight else, I will set the lookAwayTime and lookAtTime value. If the lookAwayTimeParamId and lookAtTimeParamId is not 0(meaning there is this parameter in the state machine), it will get the float value from them else it will set the default value which is 5 for lookAtTime and 3 for lookAwayTime. So if isLooking(a Boolean) is true, I will set isLooking to false and set the timer to lookAwayTime value else it will set the isLooking to true and timer to lookAtTime value.

References

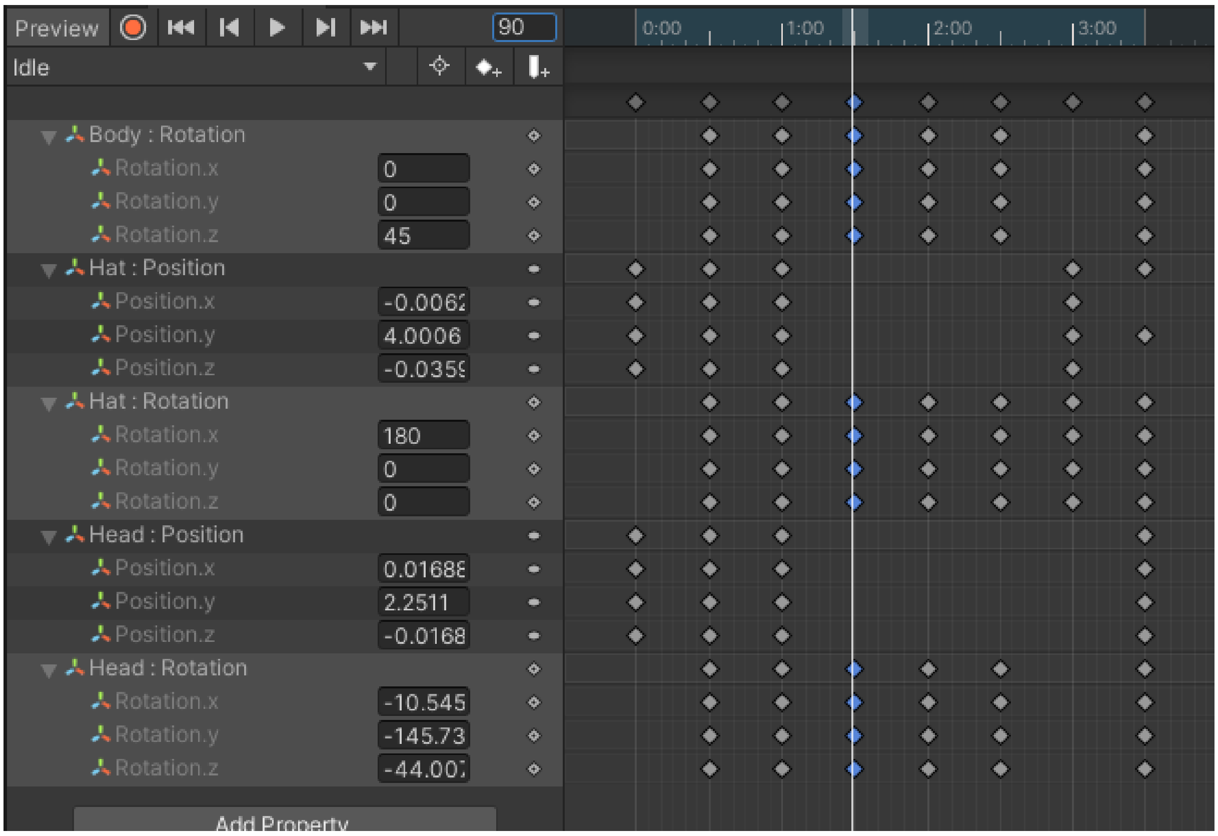
Character look at using inverse kinetic: https://www.youtube.com/watch?v=SLAYPZ7lukY

Character used: https://www.mixamo.com/#/?page=1&query=doozy&type=Character

Breathing Idle:
https://www.mixamo.com/#/?page=1&query=breathing+idle&type=Motion%2CMotionPack

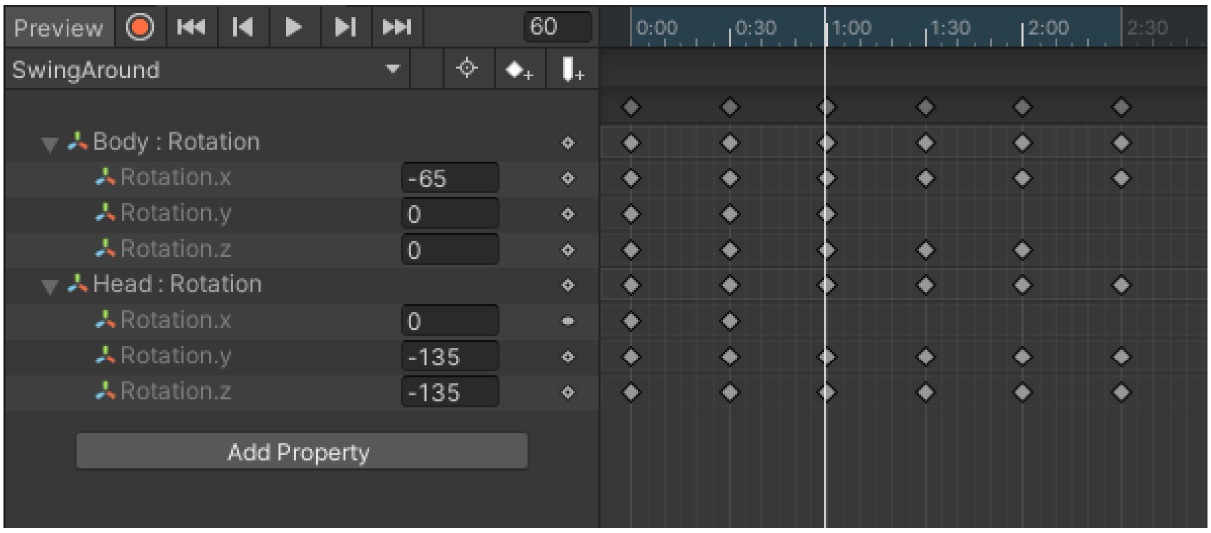
Not able to make keyframe for humanoid: https://forum.unity.com/threads/is-it-possible-to-create-humanoid-animations-in-editor.380049/

Idle Keyframe



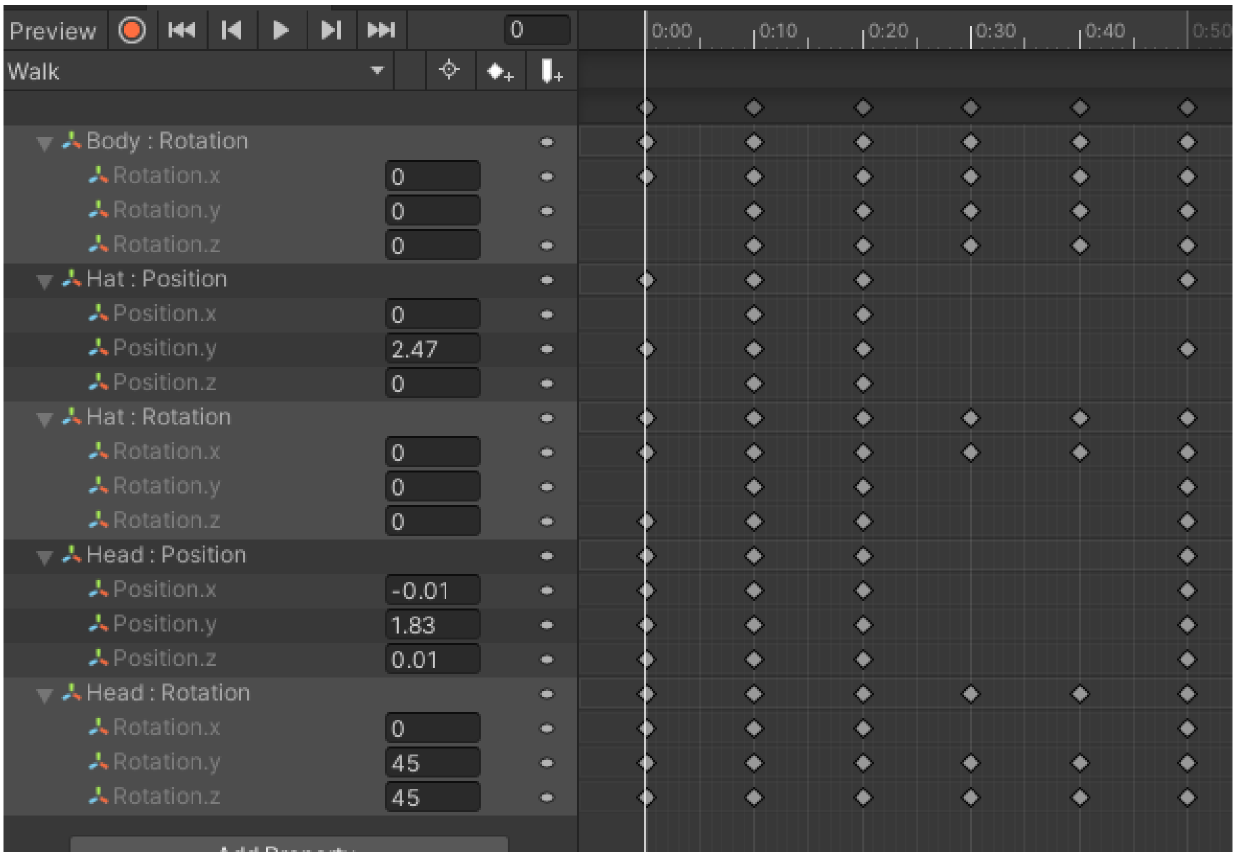
Body : Rotation
Rotation.x 0
Rotation.y 0
Rotation.z 45
Hat : Position
Position.x -0.0062
Position.y 4.0006
Position.z -0.0359
Hat : Rotation
Rotation.x 180
Rotation.y 0
Rotation.z 0
Head : Position
Position.x 0.01688
Position.y 2.2511
Position.z -0.0168
Head : Rotation
Rotation.x -10.545
Rotation.y -145.73
Rotation.z -44.007

SwingAround Keyframe



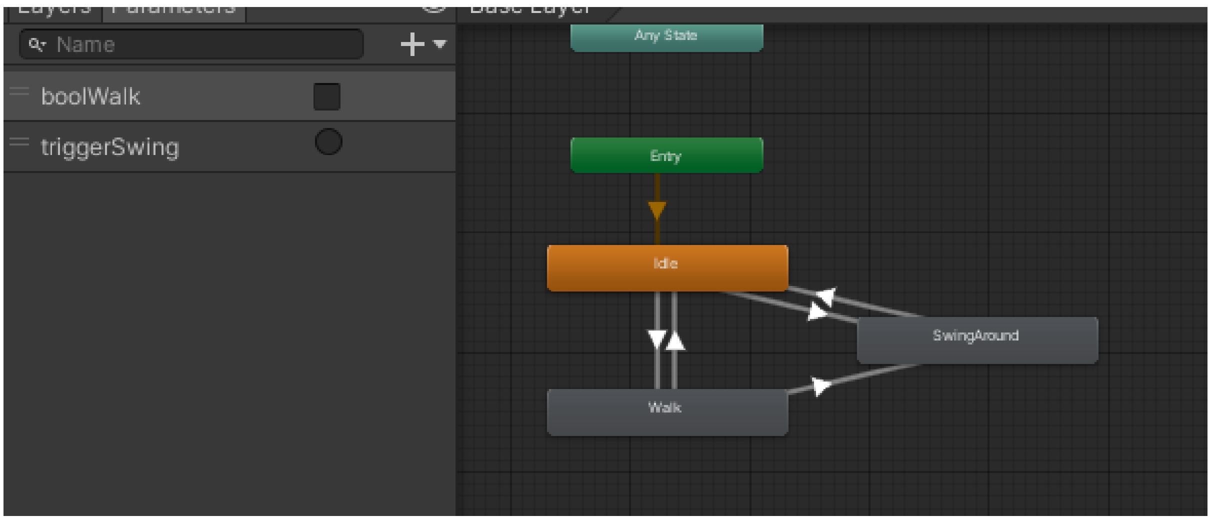
Body : Rotation
Rotation.x -65
Rotation.y 0
Rotation.z 0
Head : Rotation
Rotation.x 0
Rotation.y -135
Rotation.z -135

Walk Keyframe



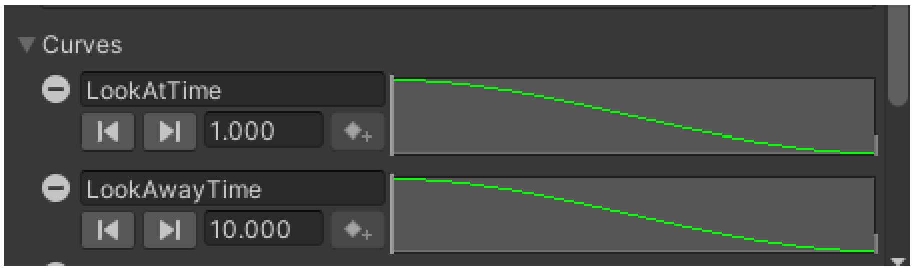
Body : Rotation
Rotation.x 0
Rotation.y 0
Rotation.z 0
Hat : Position
Position.x 0
Position.y 2.47
Position.z 0
Hat : Rotation
Rotation.x 0
Rotation.y 0
Rotation.z 0
Head : Position
Position.x -0.01
Position.y 1.83
Position.z 0.01
Head : Rotation
Rotation.x 0
Rotation.y 45
Rotation.z 45

Player State Machine

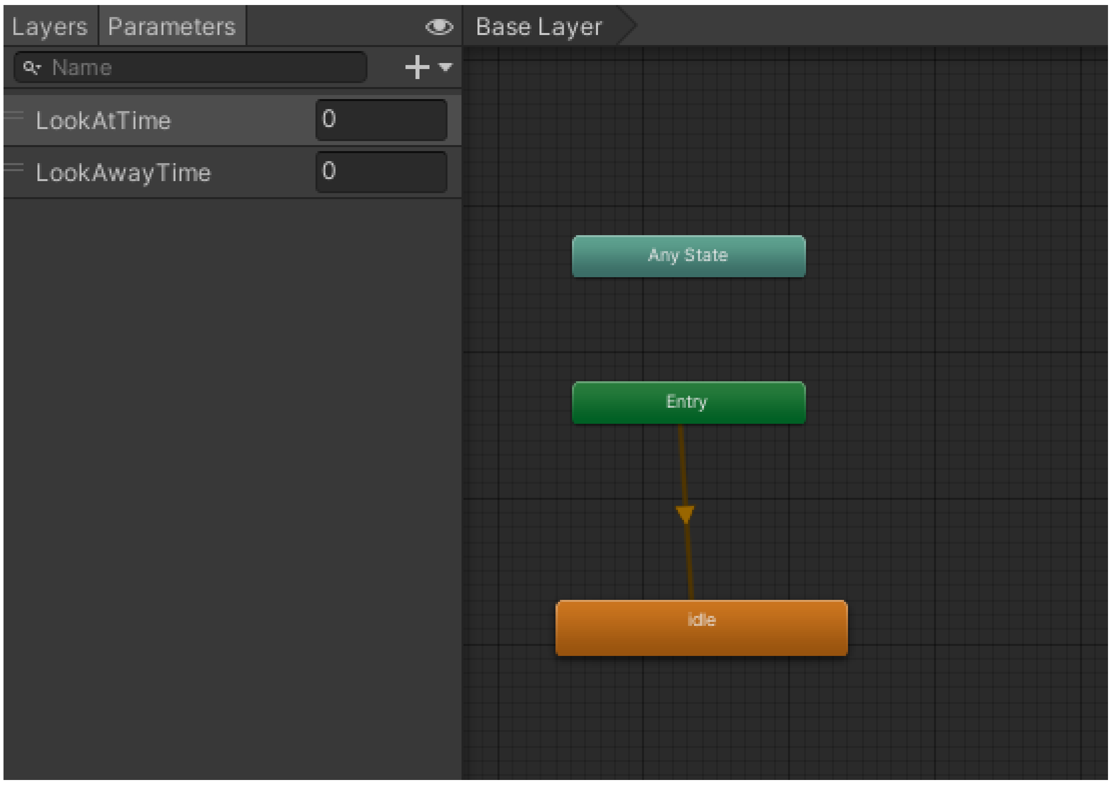


boolWalk
triggerSwing

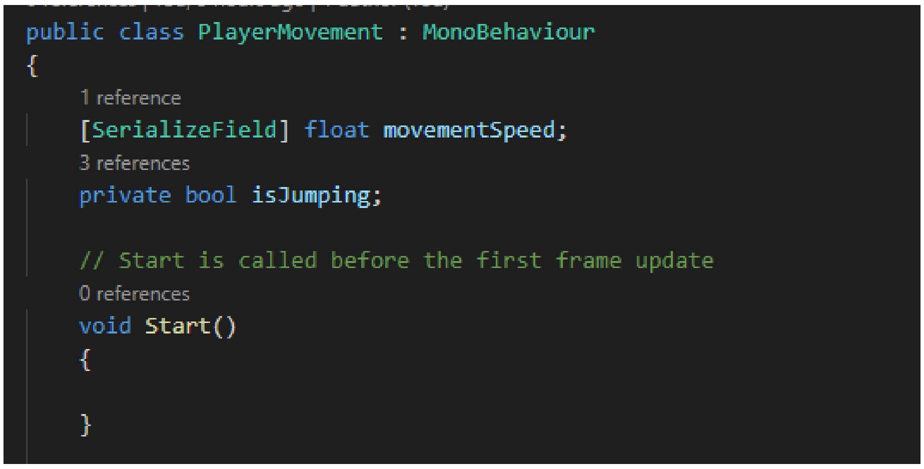
Character Idle Curves



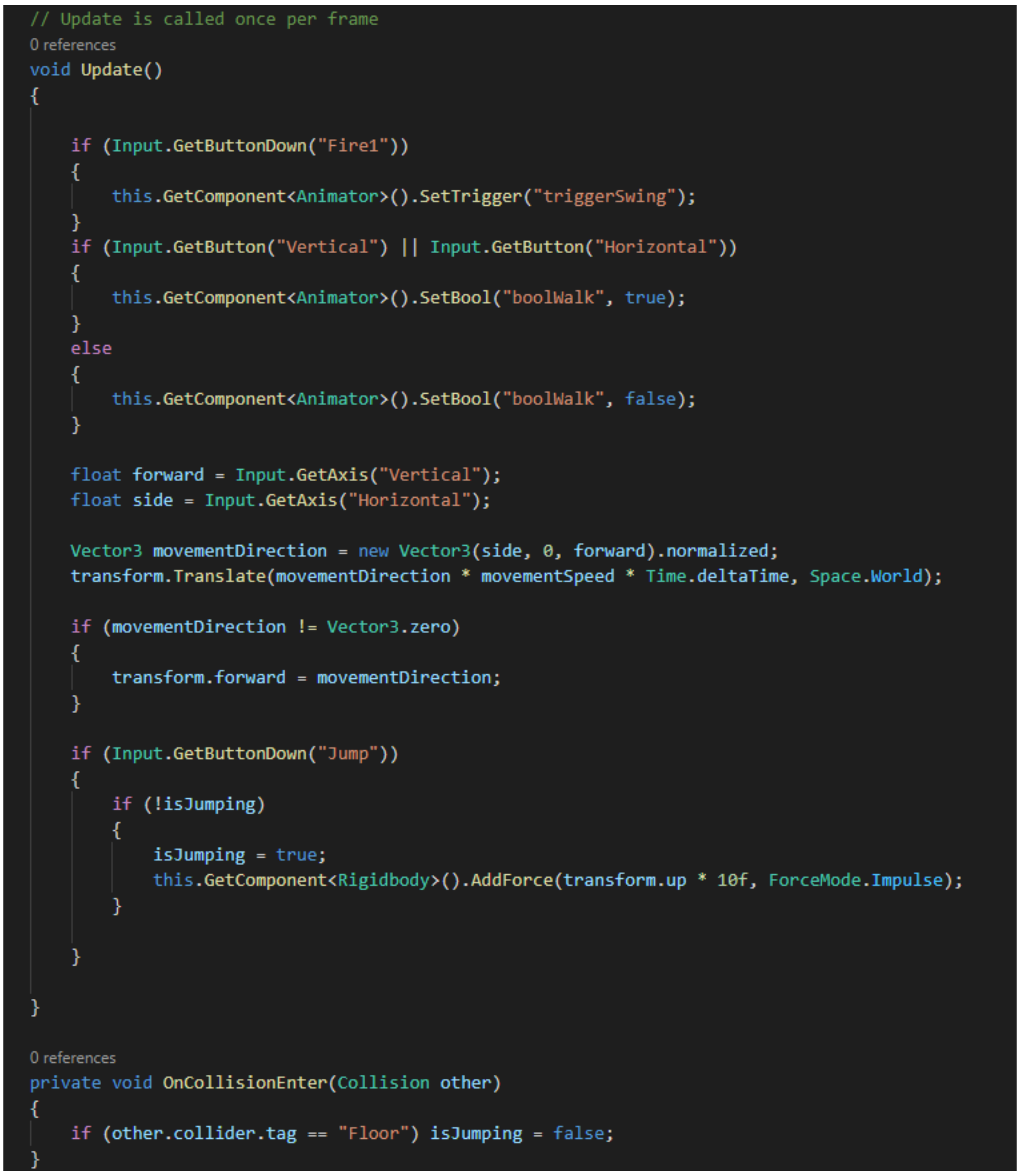
Character State Machine



PlayerMovement

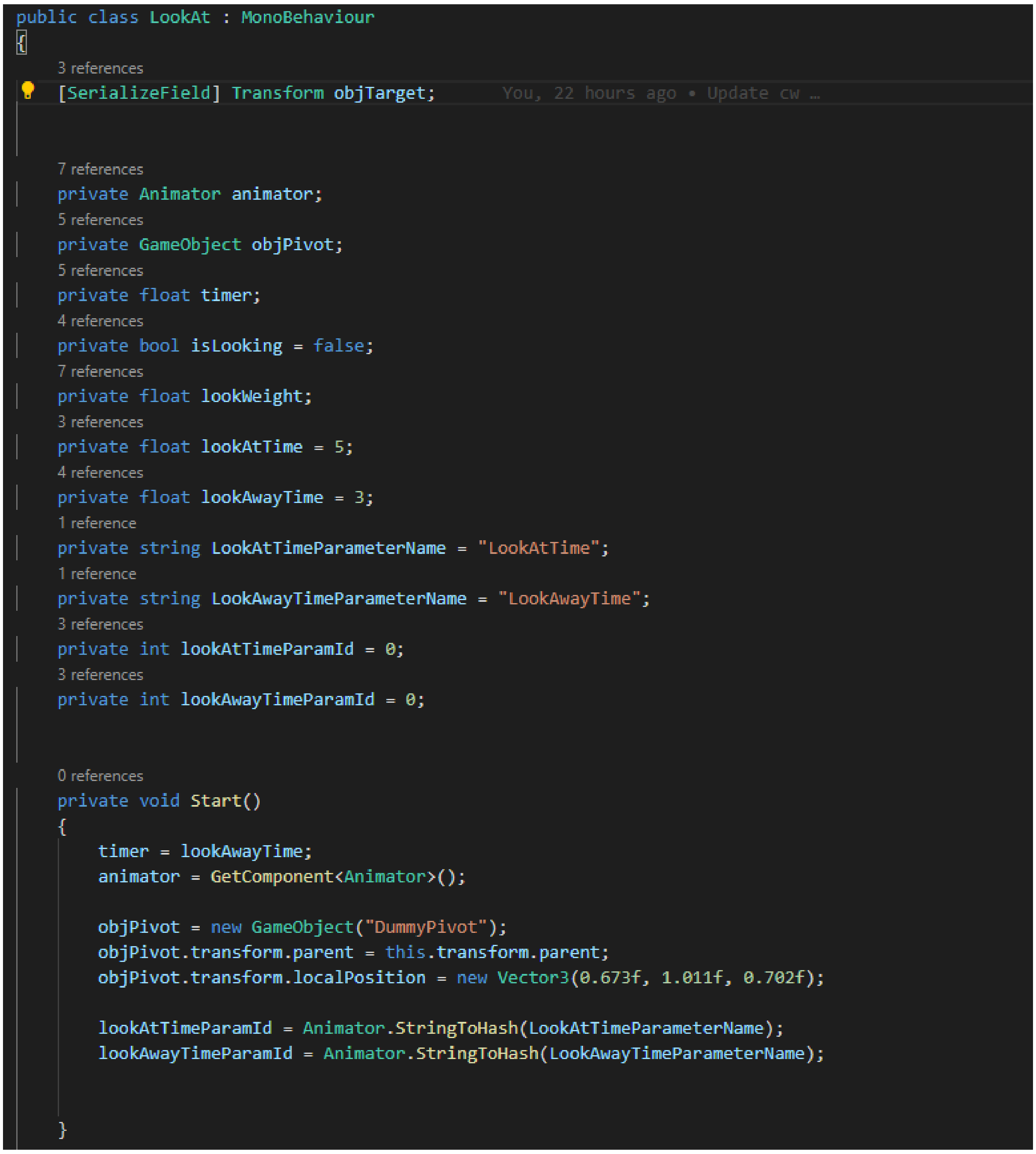


```
public class PlayerMovement : MonoBehaviour
{
    1 reference
    [SerializeField] float movementSpeed;
    3 references
    private bool isJumping;

    // Start is called before the first frame update
    0 references
    void Start()
    {

    }
```

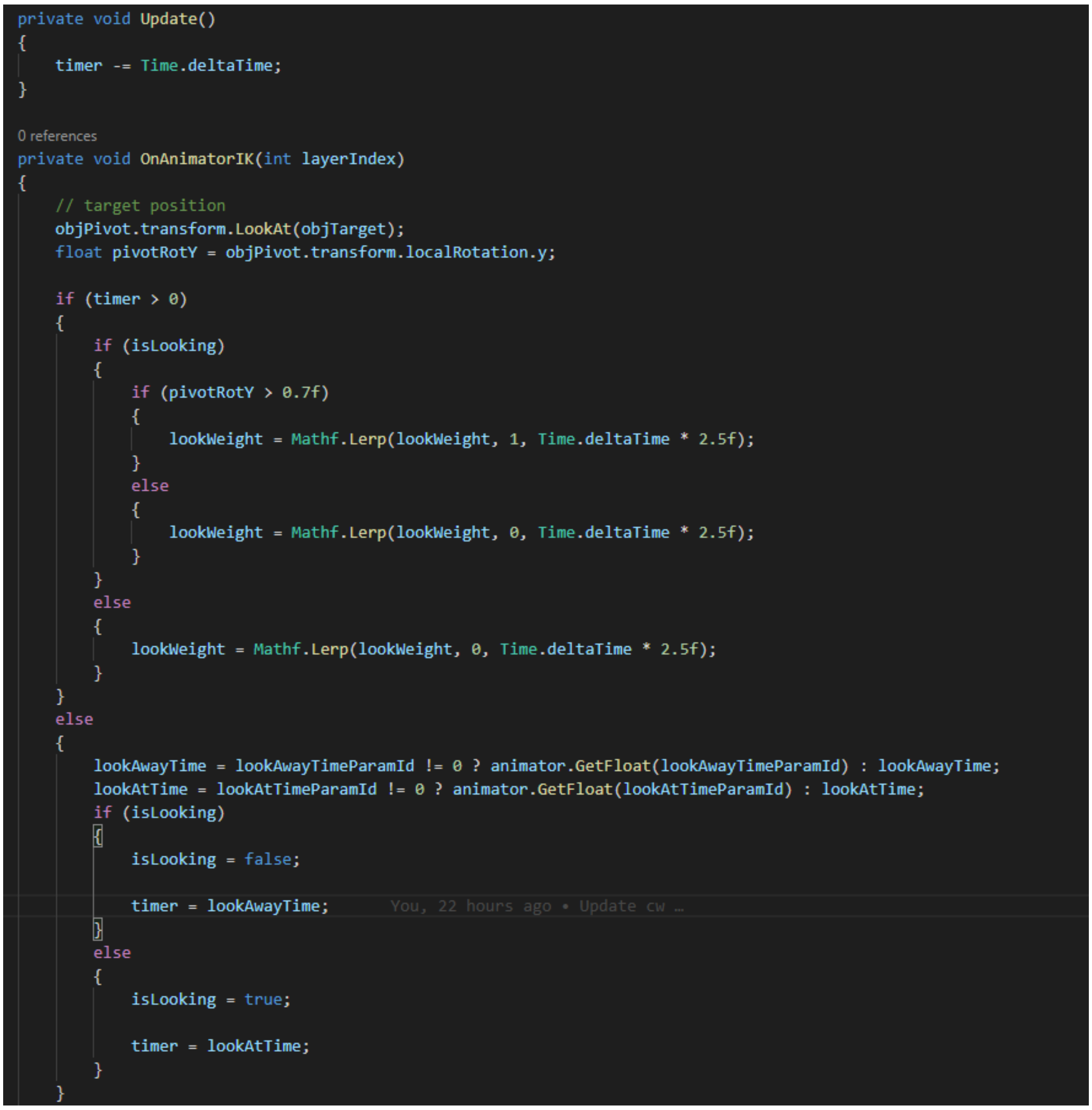


```
// Update is called once per frame
0 references
void Update()
{

    if (Input.GetButtonDown("Fire1"))
    {
        this.GetComponent<Animator>().SetTrigger("triggerSwing");
    }
    if (Input.GetButton("Vertical") || Input.GetButton("Horizontal"))
    {
        this.GetComponent<Animator>().SetBool("boolWalk", true);
    }
    else
    {
        this.GetComponent<Animator>().SetBool("boolWalk", false);
    }

    float forward = Input.GetAxis("Vertical");
    float side = Input.GetAxis("Horizontal");

    Vector3 movementDirection = new Vector3(side, 0, forward).normalized;
    transform.Translate(movementDirection * movementSpeed * Time.deltaTime, Space.World);

    if (movementDirection != Vector3.zero)
    {
        transform.forward = movementDirection;
    }

    if (Input.GetButtonDown("Jump"))
    {
        if (!isJumping)
        {
            isJumping = true;
            this.GetComponent<Rigidbody>().AddForce(transform.up * 10f, ForceMode.Impulse);
        }

    }

}

0 references
private void OnCollisionEnter(Collision other)
{
    if (other.collider.tag == "Floor") isJumping = false;
}
```

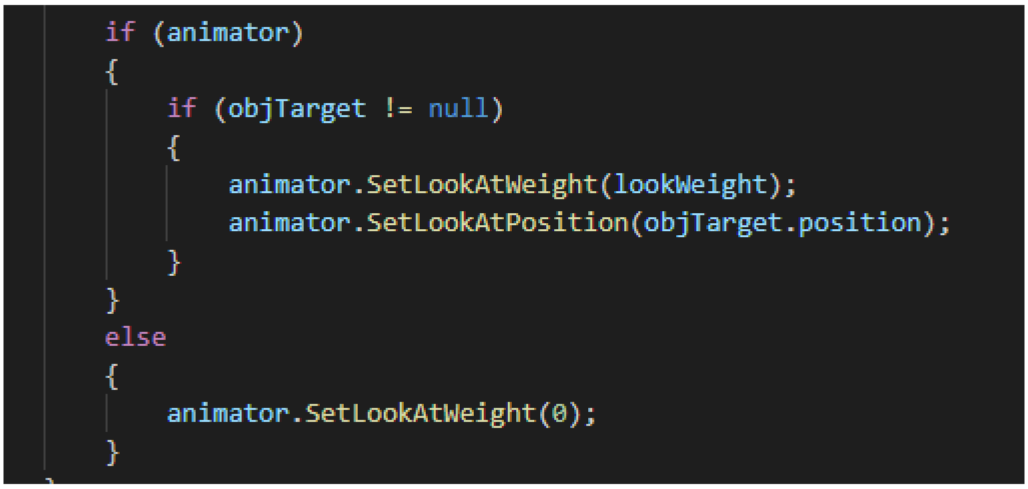
LookAt



```
public class LookAt : MonoBehaviour
{
    3 references
    [SerializeField] Transform objTarget;        You, 22 hours ago • Update cw …


    7 references
    private Animator animator;
    5 references
    private GameObject objPivot;
    5 references
    private float timer;
    4 references
    private bool isLooking = false;
    7 references
    private float lookWeight;
    3 references
    private float lookAtTime = 5;
    4 references
    private float lookAwayTime = 3;
    1 reference
    private string LookAtTimeParameterName = "LookAtTime";
    1 reference
    private string LookAwayTimeParameterName = "LookAwayTime";
    3 references
    private int lookAtTimeParamId = 0;
    3 references
    private int lookAwayTimeParamId = 0;


    0 references
    private void Start()
    {
        timer = lookAwayTime;
        animator = GetComponent<Animator>();

        objPivot = new GameObject("DummyPivot");
        objPivot.transform.parent = this.transform.parent;
        objPivot.transform.localPosition = new Vector3(0.673f, 1.011f, 0.702f);

        lookAtTimeParamId = Animator.StringToHash(LookAtTimeParameterName);
        lookAwayTimeParamId = Animator.StringToHash(LookAwayTimeParameterName);


    }
```



```
private void Update()
{
    timer -= Time.deltaTime;
}

0 references
private void OnAnimatorIK(int layerIndex)
{
    // target position
    objPivot.transform.LookAt(objTarget);
    float pivotRotY = objPivot.transform.localRotation.y;

    if (timer > 0)
    {
        if (isLooking)
        {
            if (pivotRotY > 0.7f)
            {
                lookWeight = Mathf.Lerp(lookWeight, 1, Time.deltaTime * 2.5f);
            }
            else
            {
                lookWeight = Mathf.Lerp(lookWeight, 0, Time.deltaTime * 2.5f);
            }
        }
        else
        {
            lookWeight = Mathf.Lerp(lookWeight, 0, Time.deltaTime * 2.5f);
        }
    }
    else
    {
        lookAwayTime = lookAwayTimeParamId != 0 ? animator.GetFloat(lookAwayTimeParamId) : lookAwayTime;
        lookAtTime = lookAtTimeParamId != 0 ? animator.GetFloat(lookAtTimeParamId) : lookAtTime;
        if (isLooking)
        {
            isLooking = false;

            timer = lookAwayTime;        You, 22 hours ago • Update cw …
        }
        else
        {
            isLooking = true;

            timer = lookAtTime;
        }
    }
```



```
    if (animator)
    {
        if (objTarget != null)
        {
            animator.SetLookAtWeight(lookWeight);
            animator.SetLookAtPosition(objTarget.position);
        }
    }
    else
    {
        animator.SetLookAtWeight(0);
    }
```