

## Projeto: GTA - Parte 2

### Apocalipse Zumbi



Conteúdos de Computação Gráfica abordados aqui:

- Colisão de objetos 3D
- Animação de Avatar
- Máquina de estados

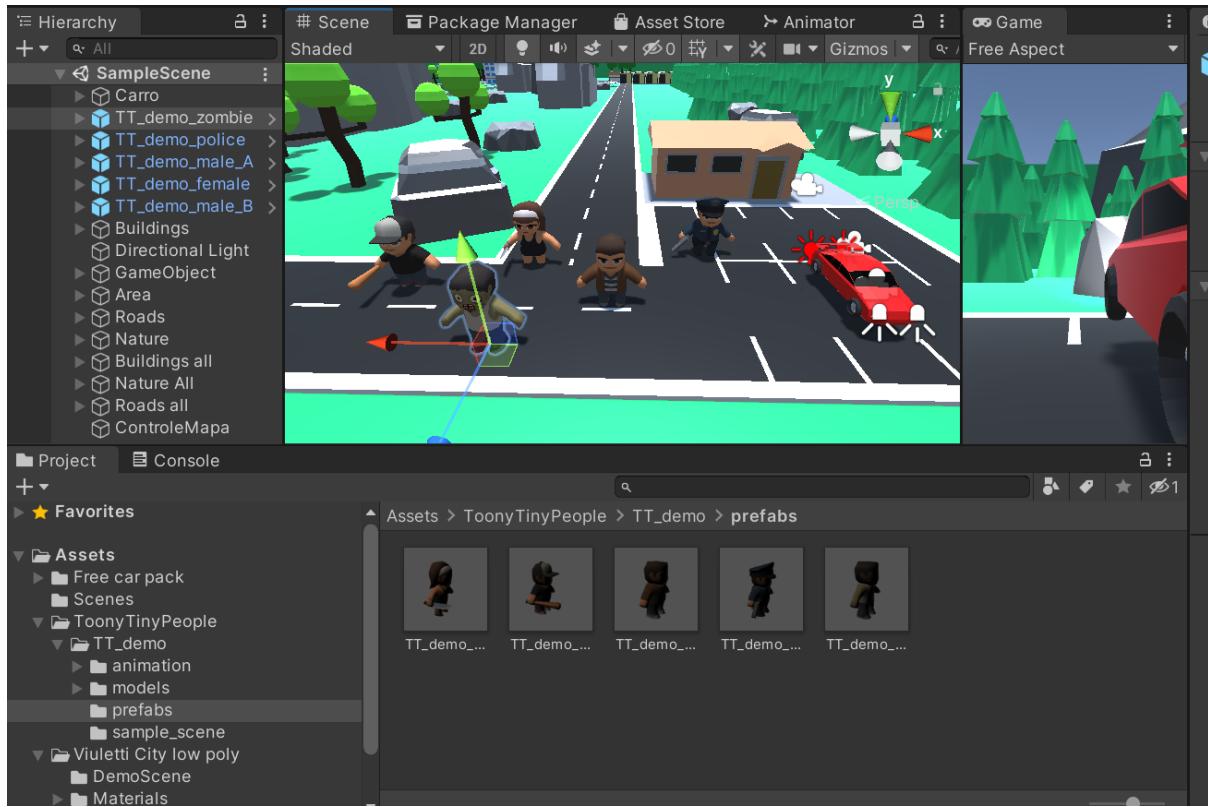
Agora que temos uma cidade e um carro para dirigir vamos dar emoção ao nosso jogo, vamos iniciar um Apocalipse Zumbi!!

1 - Faça o download de pessoas e zumbis:

<https://assetstore.unity.com/packages/3d/characters/toony-tiny-people-demo-113188>

The screenshot shows the Unity Asset Store interface. At the top, there's a navigation bar with categories like 3D, 2D, Add-Ons, Audio, Decentralization, Essentials, Templates, Tools, VFX, and Sale. Below the navigation is a search bar and various moderation status indicators. The main content area displays the 'Toony Tiny People Demo' asset, which is categorized under Characters. It shows a preview image of several stylized 3D models, including a police officer and zombies. The asset is labeled as 'FREE'. Below the preview, there's a review from a user named 'reopucino' with a 5-star rating and the comment 'Nice for demo'. At the bottom of the asset page, there are links for 'License agreement' and 'Standard Unity Asset Store EULA', and 'License type'.

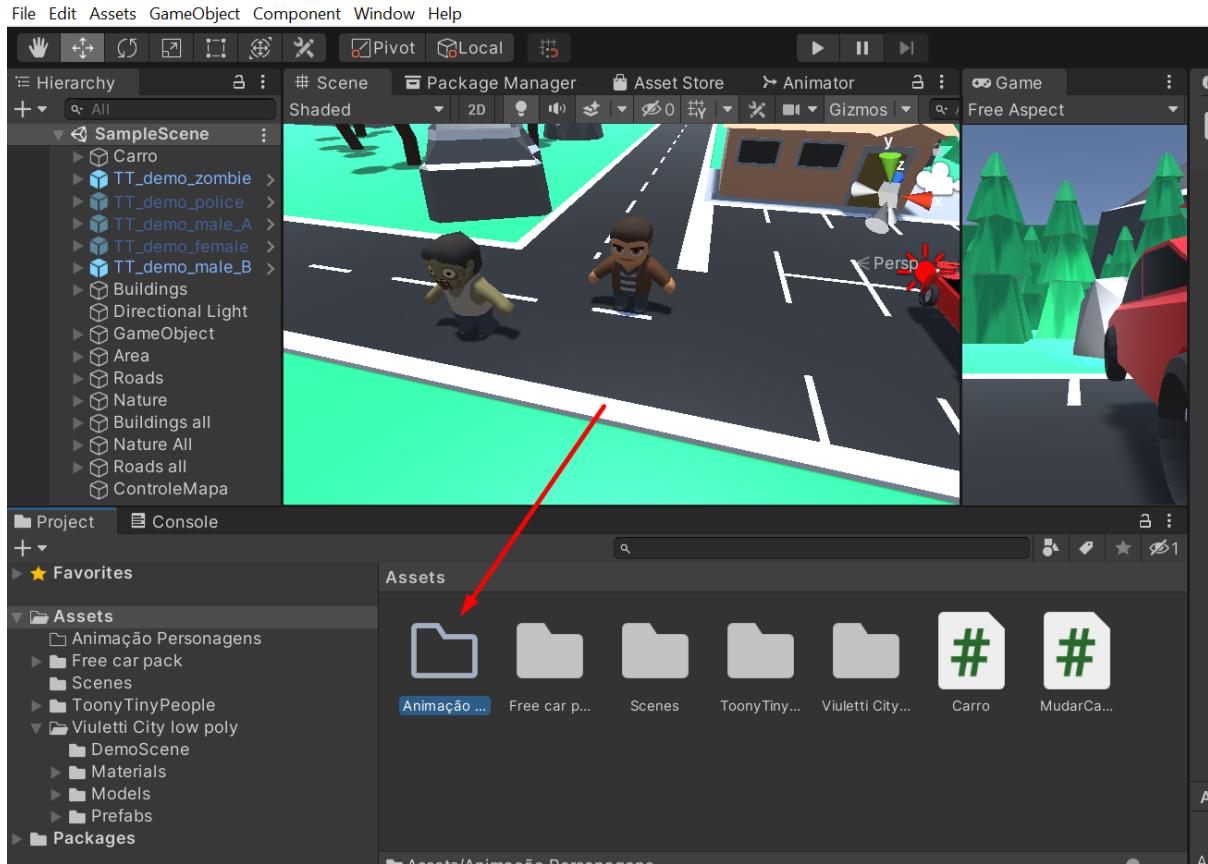
2 - Após o download encontre os Prefabs dos personagens na pasta “Toony Tiny People” e adiciona um exemplo de cada personagem a nossa cena:



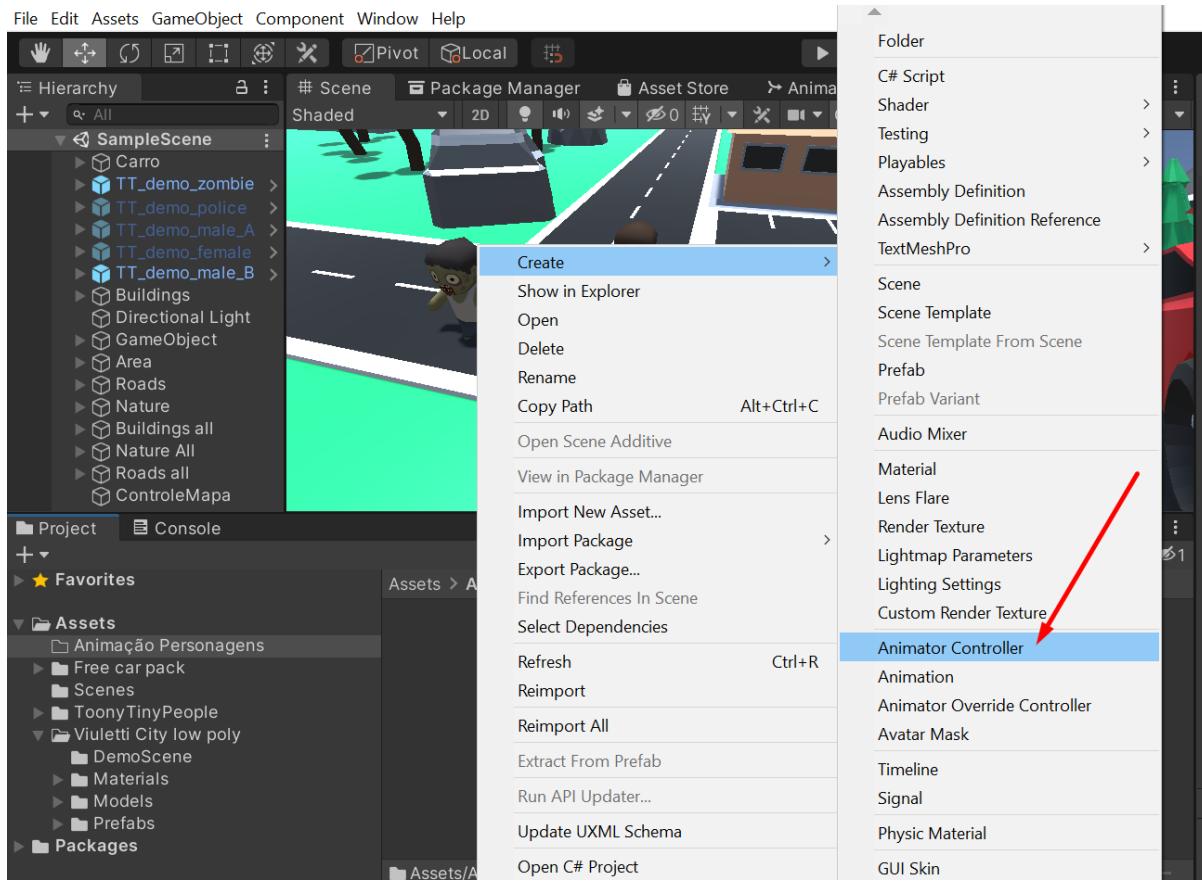
3 - Desative alguns personagens, para começar vamos precisar do personagem desarmado e do zombi:



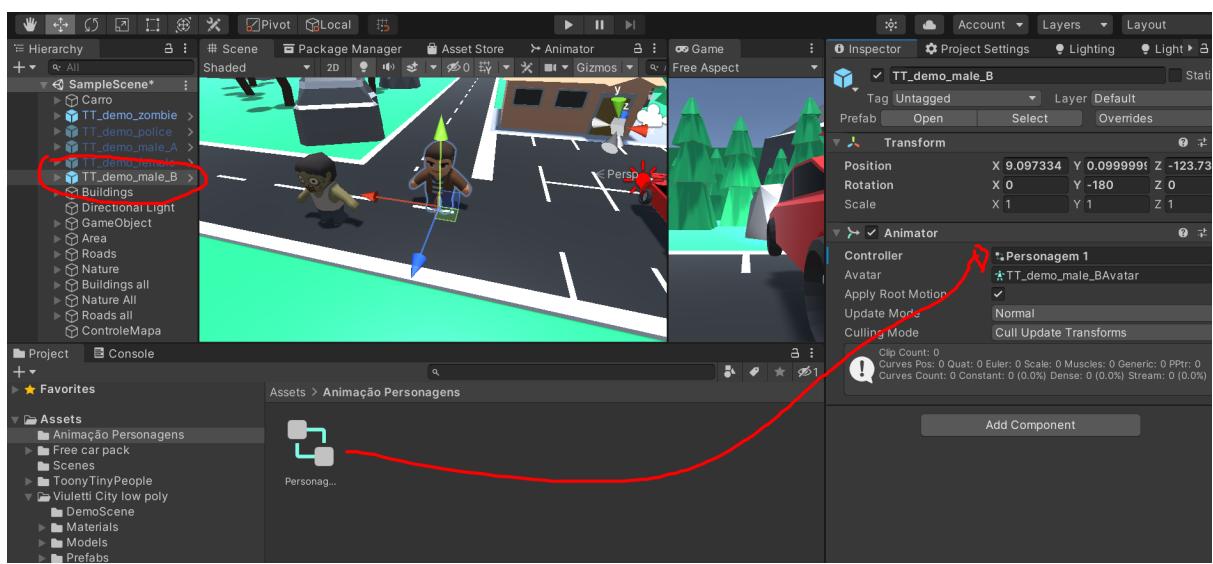
4 - Crie uma nova pasta chamada Animação Personagens, aqui iremos colocar o Controller de animação



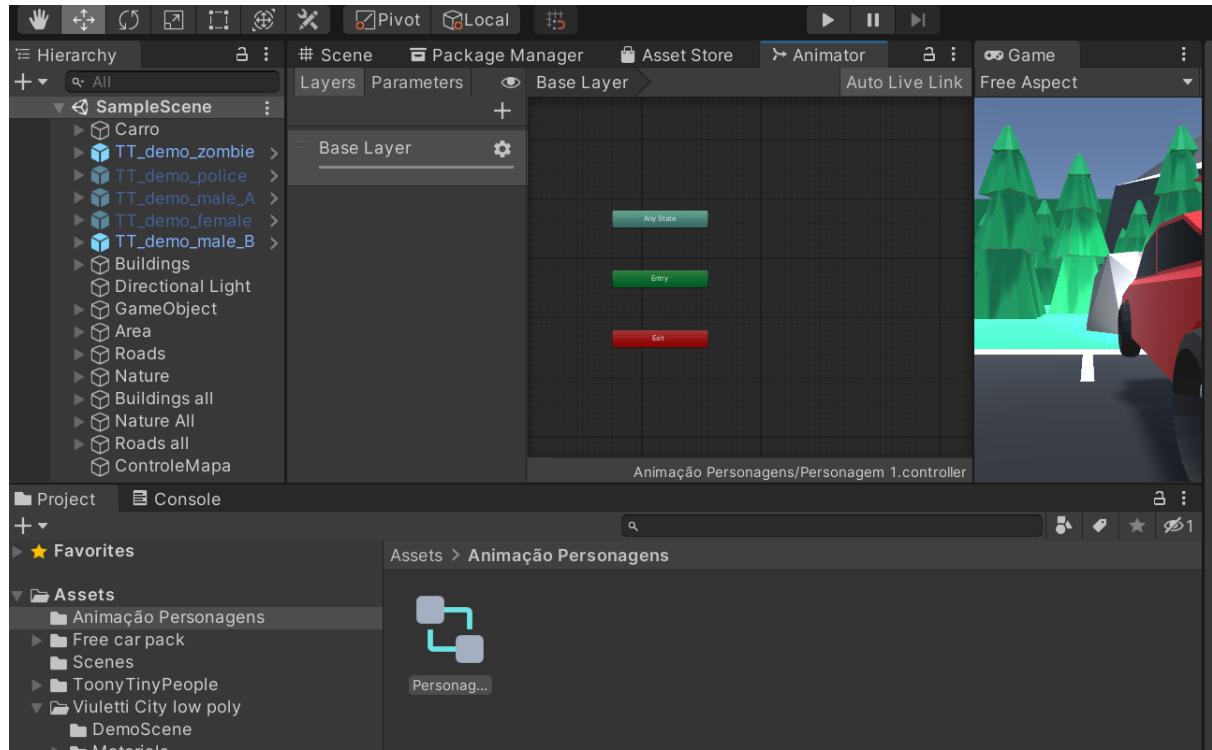
5 - Dentro desta pasta, com o botão direito, crie um Animator Controller chamado "Personagem 1".



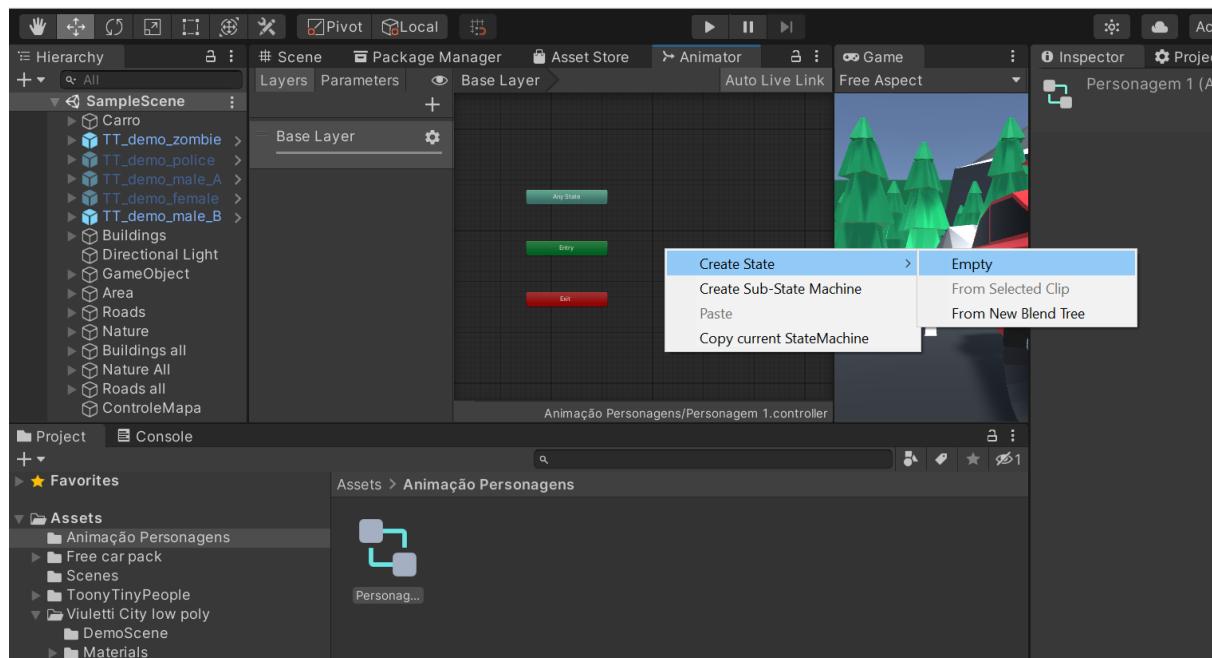
6 - Selecione o nosso personagem desarmado e adicione o Animator Controller criado na propriedade Animator. Agora podemos controlar as animações.



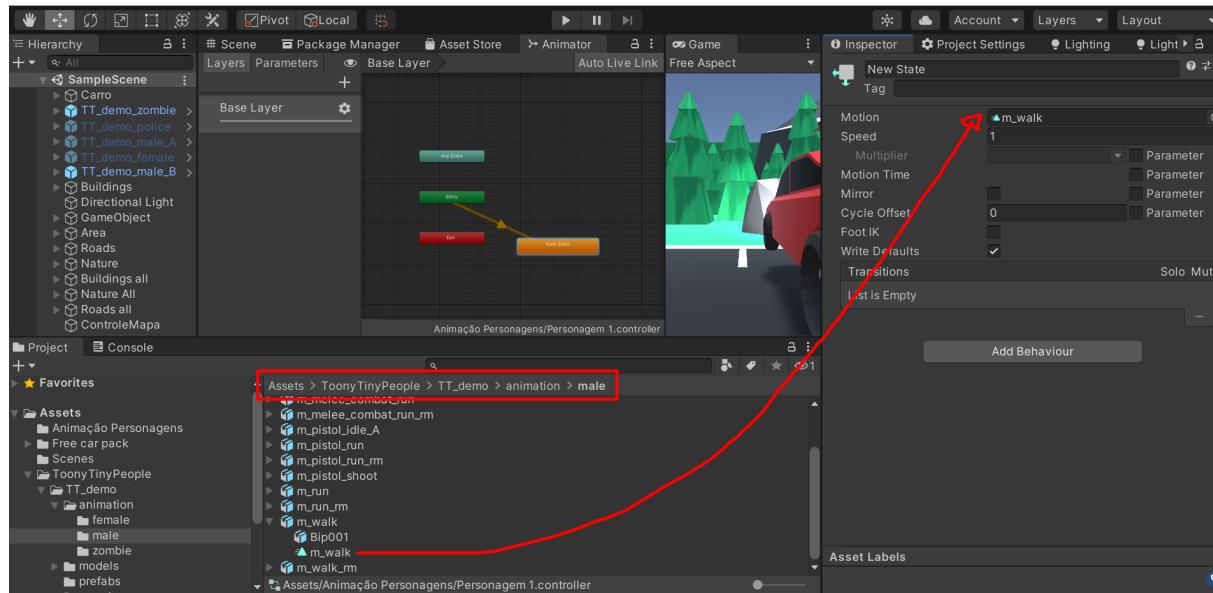
7 - Dê duplo clique no Animator Controller “Personagem 1”, uma aba chamada Animator irá surgir.



8 - Dentro da aba Animator dê um botão direito e crie um novo estado vazio.

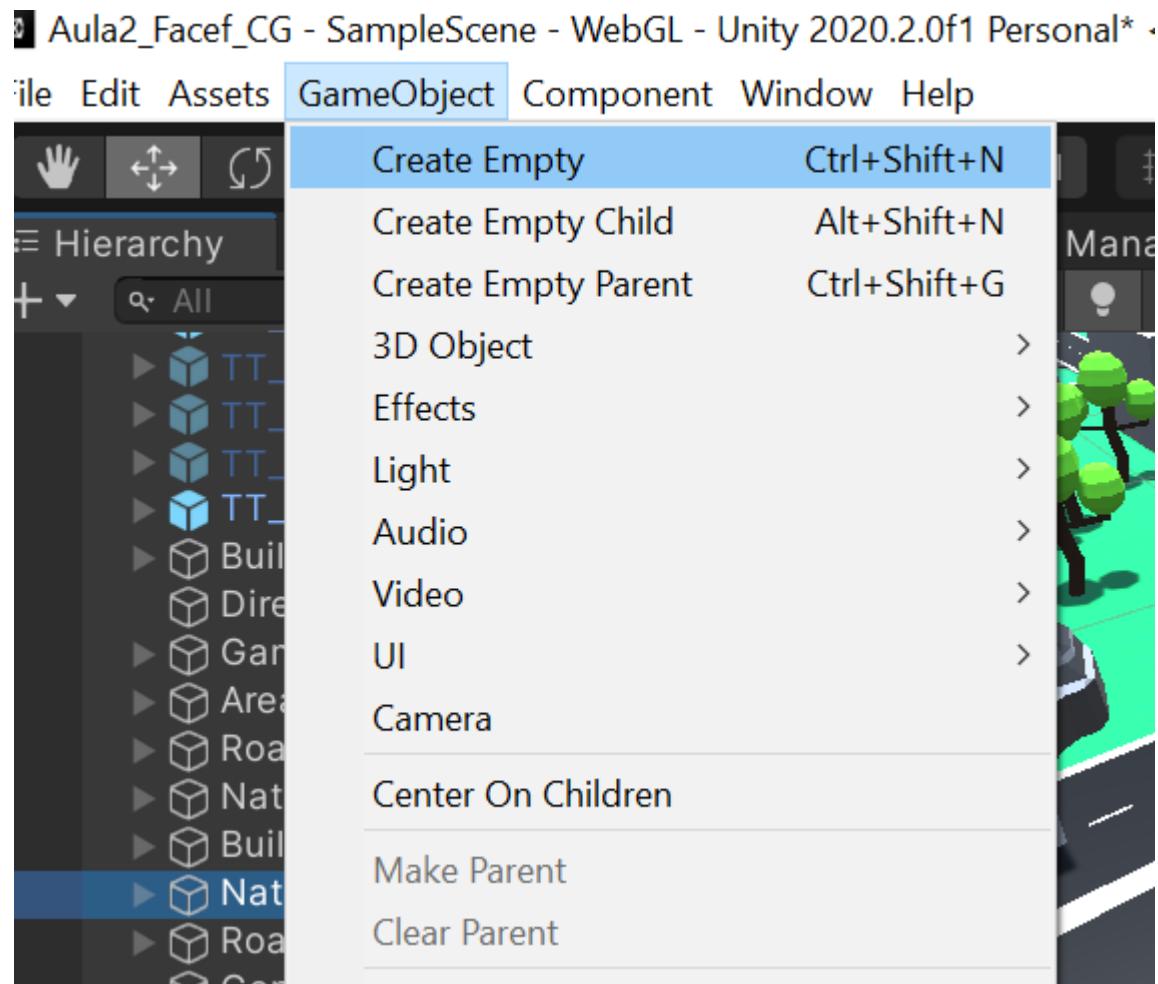


9 - Clique nesse novo estado (quadrado laranja) para que na barra lateral (Inspector) apareça a opção Motion. Abra a pasta animation dentro da pasta Toony Tiny People, abra o prefab m\_walk e arrast a animação “m\_walk” para motion como mostrado abaixo:

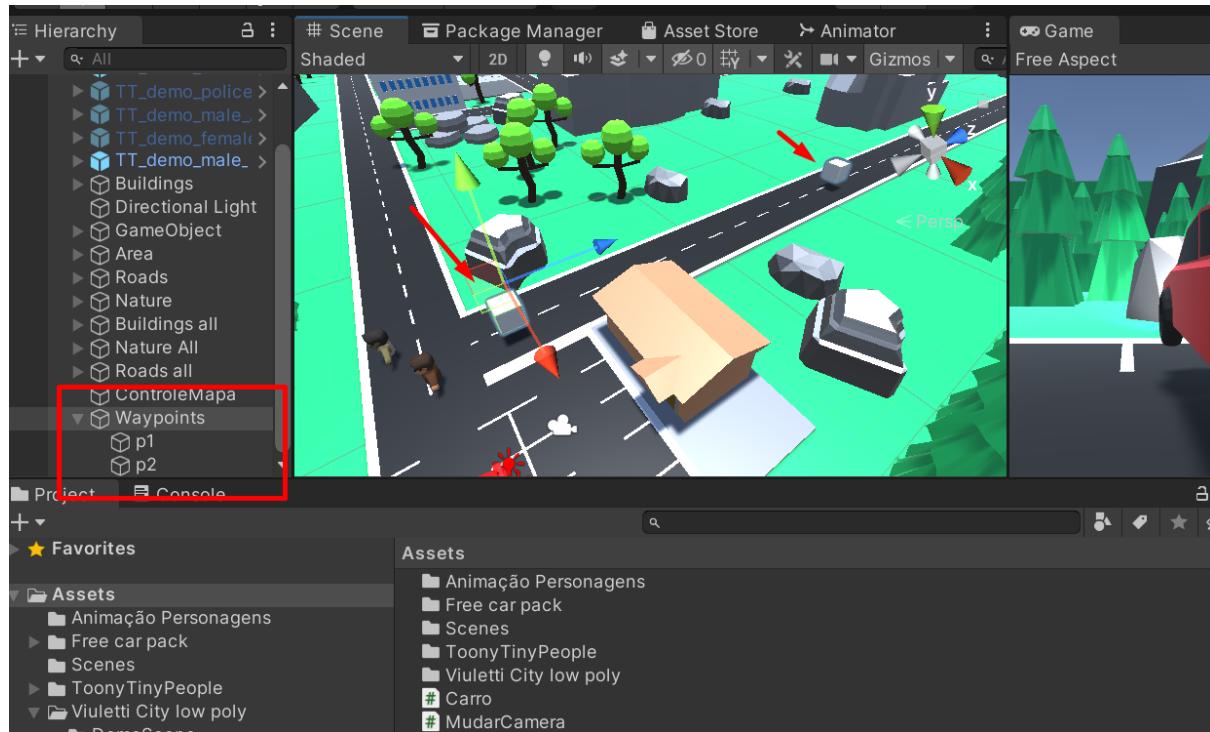


10 - Dê play na cena e veja se o personagem está sendo com a animação de andar. Veja que isso não o faz andar realmente, será apenas animação.

11 - Agora vamos fazer o personagem ficar andando dentro de um caminho que iremos determinar (como se fosse a rua ou calçada). Crie um um objeto vazio chamado Waypoints:



12 - Dentro desse objeto vazio crie dois cubos, um chamado p1 e outro p2. Eles serão as referências do caminho (path) que o personagem irá andar.



13 - Crie um novo script chamado “followPath” e coloque o código abaixo:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class followPath : MonoBehaviour
{
    [SerializeField] public Transform[] allwayPoints;
    [SerializeField] public float rotationSpeed = 0.5f;
    [SerializeField] public float movementSpeed = 0.5f;
    [SerializeField] public int currentTarget;

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

    }

    // Update is called once per frame
    void Update()
    {
        Movement();
        Rotate();
    }

    void Movement()
    {
        Vector3 targetPosition = allwayPoints[currentTarget].position;
        transform.position = Vector3.MoveTowards(transform.position, targetPosition, movementSpeed);
    }

    void Rotate()
    {
        if (currentTarget < allwayPoints.Length - 1)
        {
            Vector3 targetPosition = allwayPoints[currentTarget + 1].position;
            Vector3 targetRotation = targetPosition - transform.position;
            transform.rotation = Quaternion.LookRotation(targetRotation);
        }
    }
}
```

```

    ChangeTarget();
}

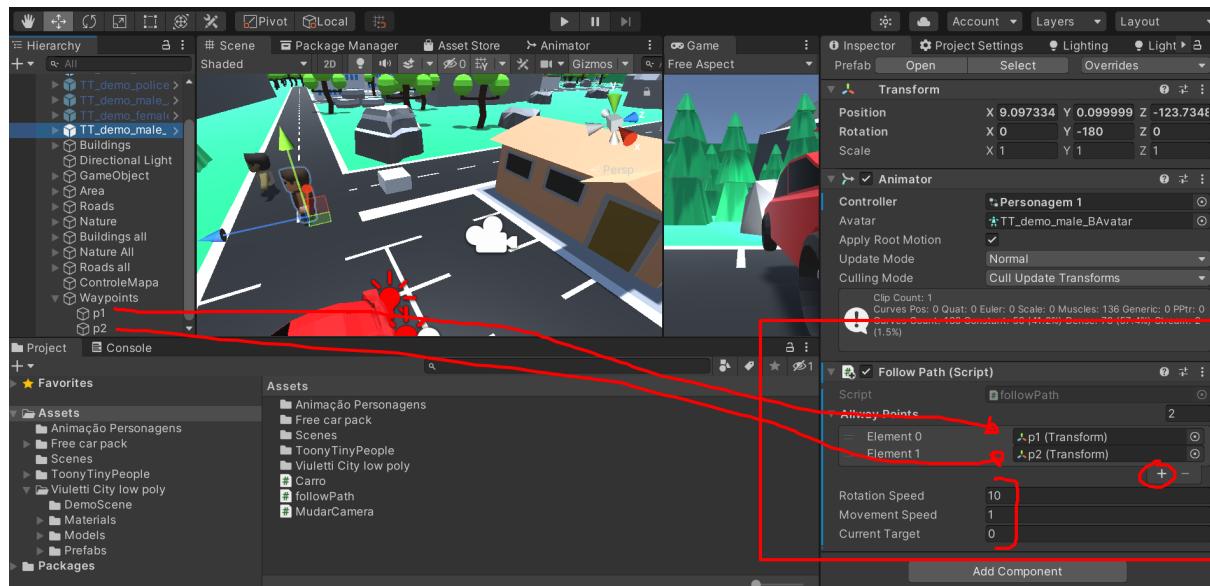
void Movement()
{
    transform.position = Vector3.MoveTowards(
        transform.position,
        allwayPoints[currentTarget].position,
        movementSpeed * Time.deltaTime);
}

void Rotate()
{
    transform.rotation = Quaternion.Slerp(transform.rotation,
    Quaternion.LookRotation(
    allwayPoints[currentTarget].position-transform.position),
    rotationSpeed*Time.deltaTime);
}

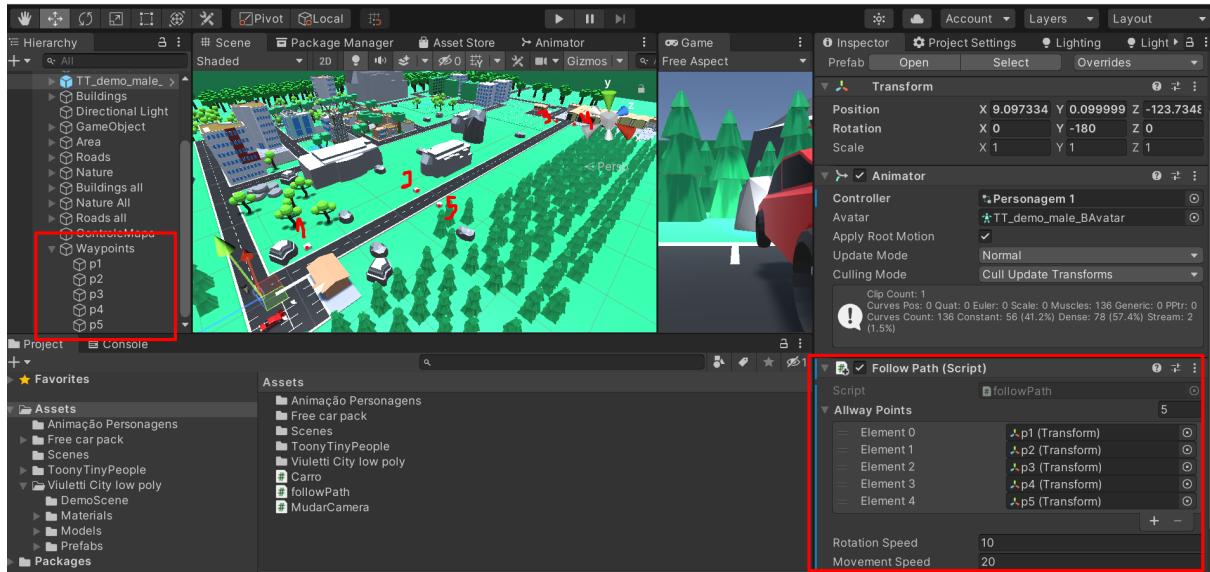
void ChangeTarget()
{
    if (transform.position == allwayPoints[currentTarget].position)
    {
        currentTarget++;
        currentTarget = currentTarget % allwayPoints.Length;
    }
}
}

```

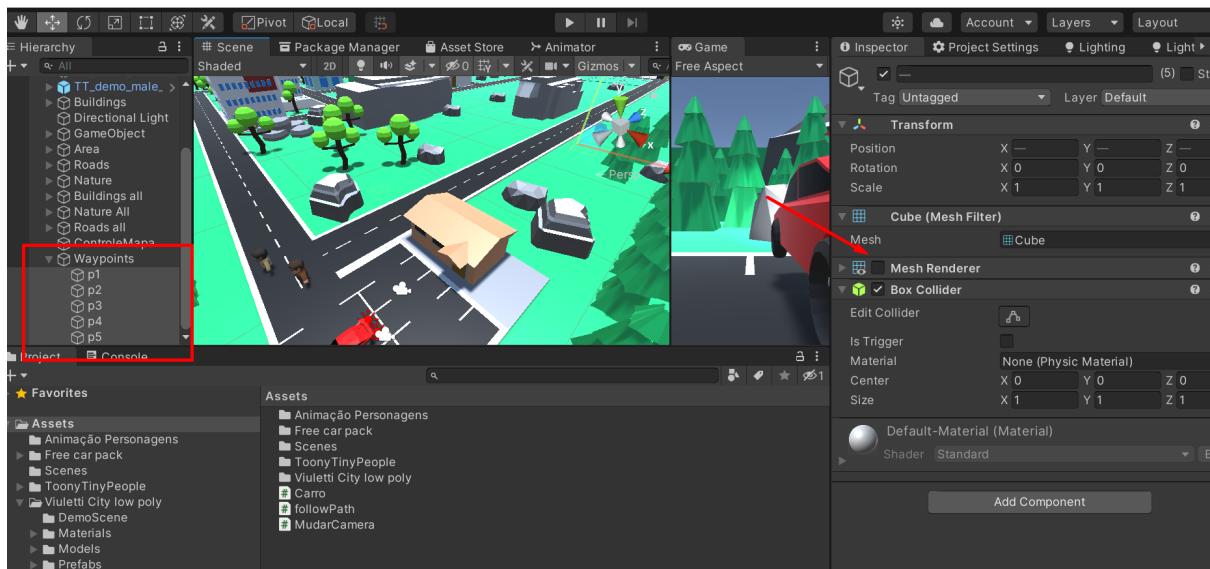
14 - Associe o script criado em nosso personagem e coloque cada um dos cubos associados ao script como indicado abaixo. Abaixe um pouco os cubos para que o personagem não fique voando sobre o mapa.



15 - Agora vá copiando os cubo, renomeando e vá posicionando onde você quer que ele fique em loop andando pelo mapa. Sempre que ele chegar no último cubo ele irá voltar para o primeiro.



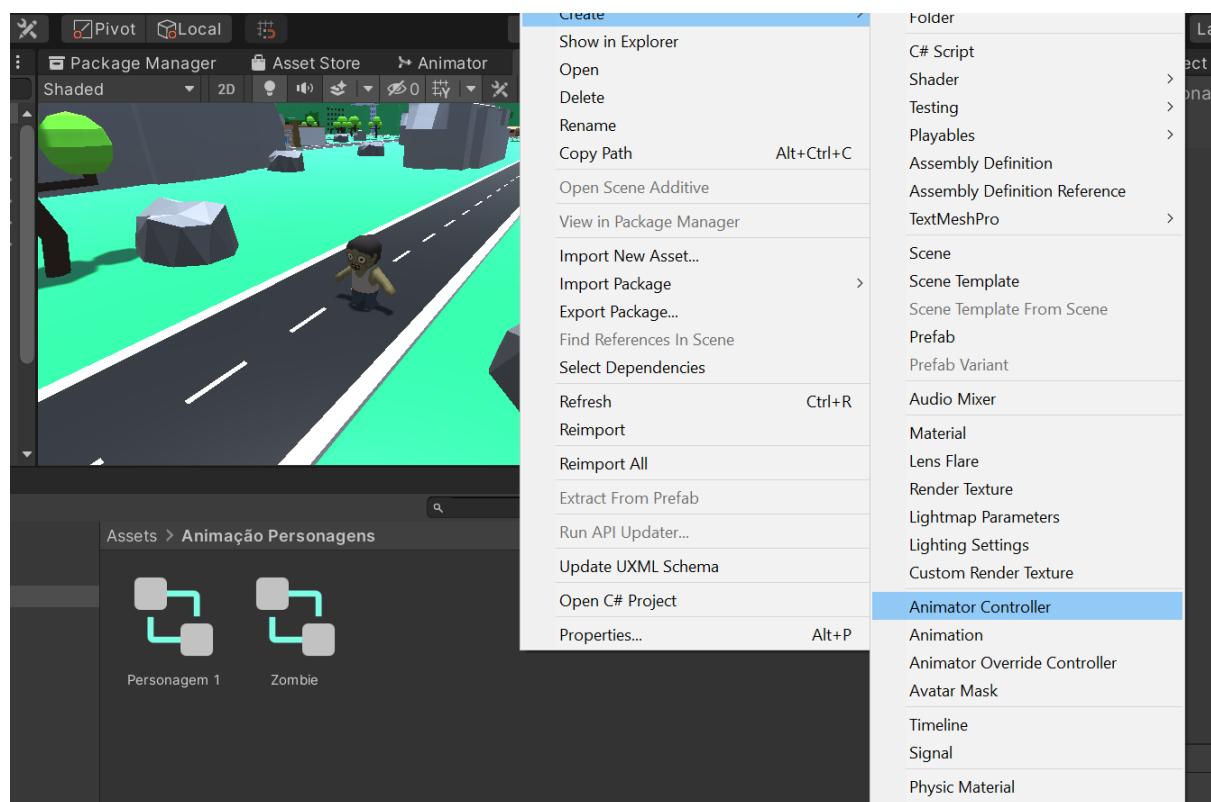
16 - Para que os cubos não fiquem aparecendo para o jogador, selecione todos os cubos (apenas os cubos) e desmarque a opção chamada Mesh Render (isso fará com que sua textura não seja renderizada durante a execução do jogo)



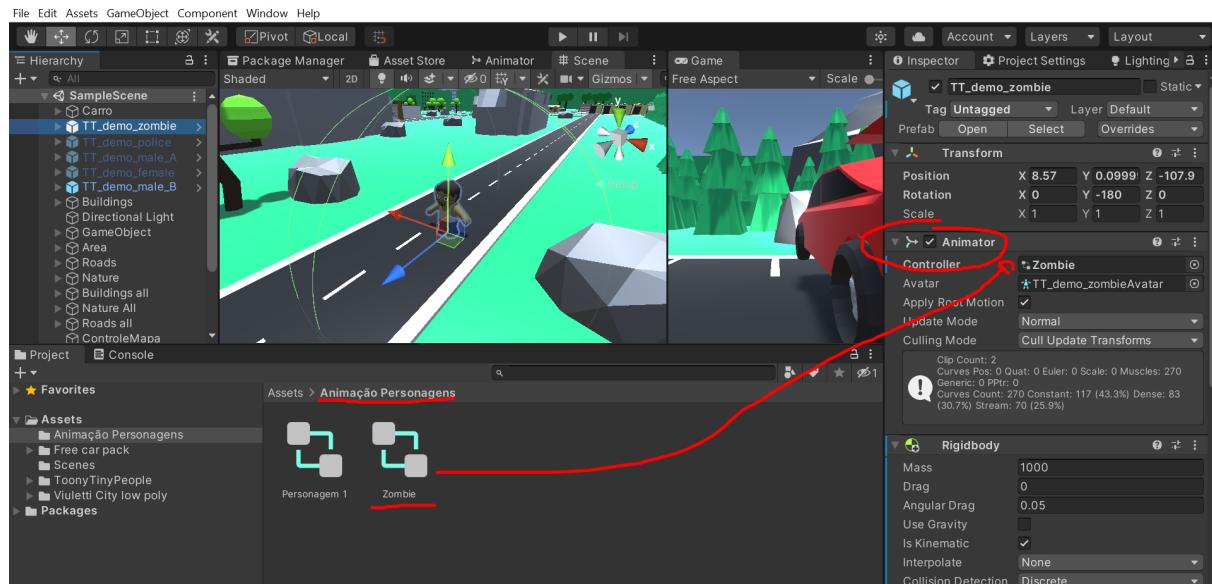
17 - Agora que o personagem está andando por aí, vamos trabalhar com o Zumbi. Primeiro vamos cuidar da animação. Adicione ao Zumbi um Rigidbody e Sphere Collider. Clicando em Edit Collider você pode escolher qual o tamanho da Sphera, imagine que esse tamanho será o que o Zumbi conseguirá ouvir ou enxergar para atacar alguém.



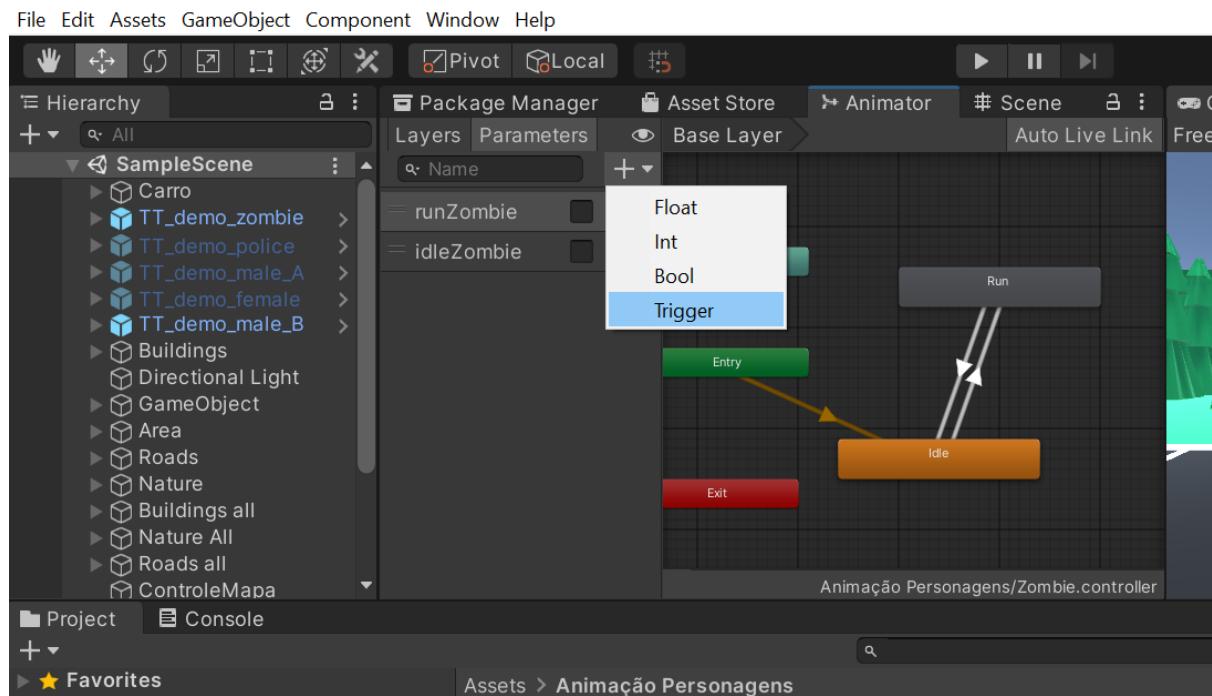
18 - Vá até a pasta Animação Personagem, crie um novo Animator Controller chamado Zombie



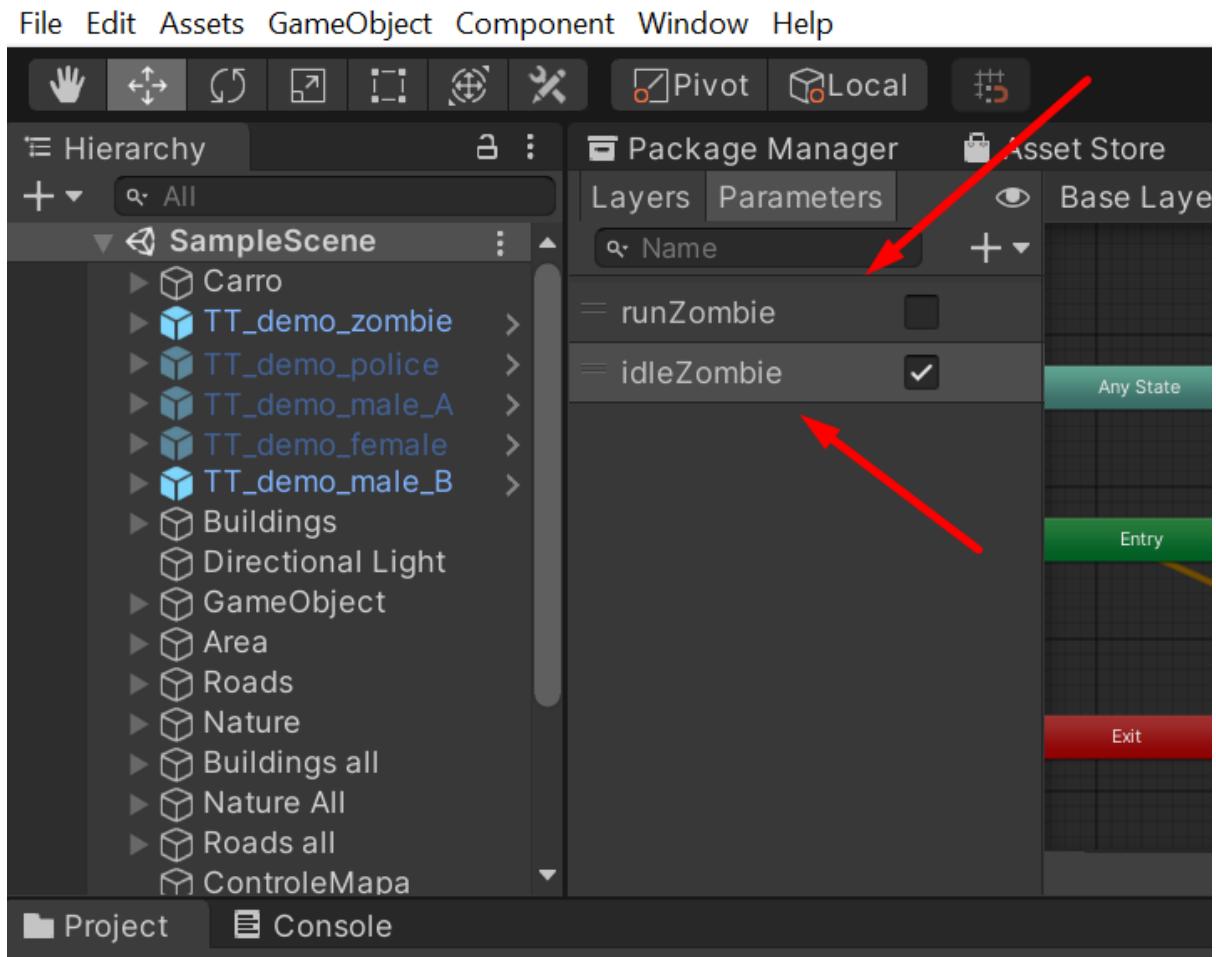
19 - No nosso zumbi, adicione um Animator, ele irá receber o Controle Animator que você acabou de criar



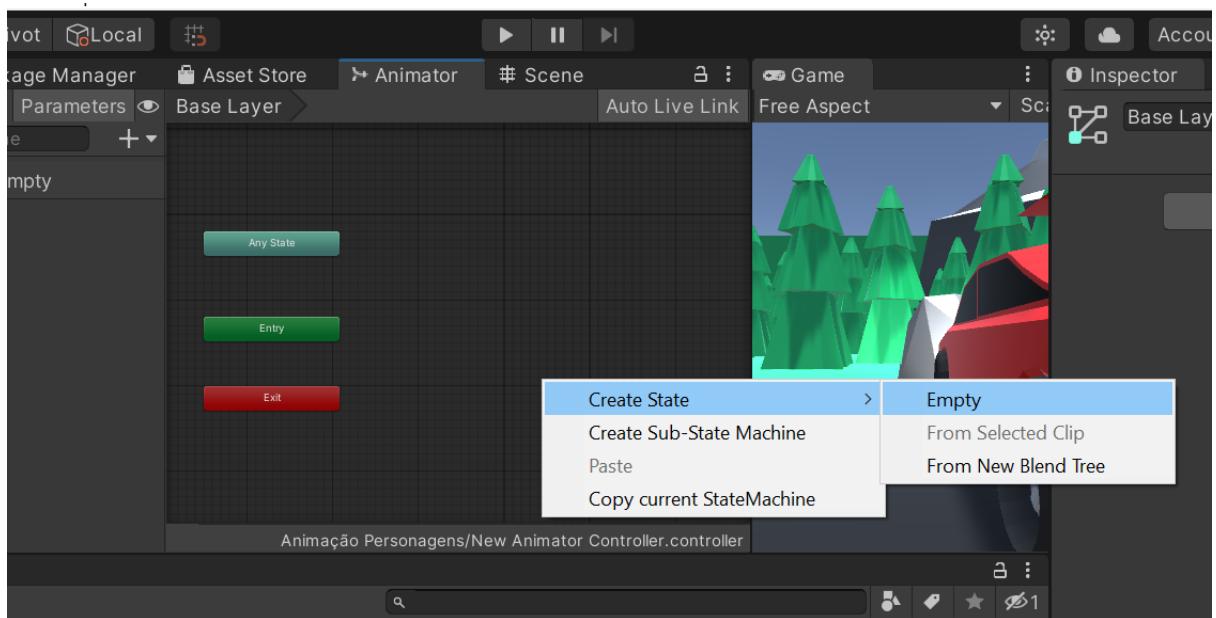
20 - Agora vamos programar o controle de estados da animação do Zumbi. Clique duas vezes no seu Animator Controller chamado Zombie, dentro da pasta Animação Personagens. Uma nova aba chamada Animator irá abrir. Dentro dessa aba clique na opção Parameters, clique no sinal de + e escolha a opção Bool:



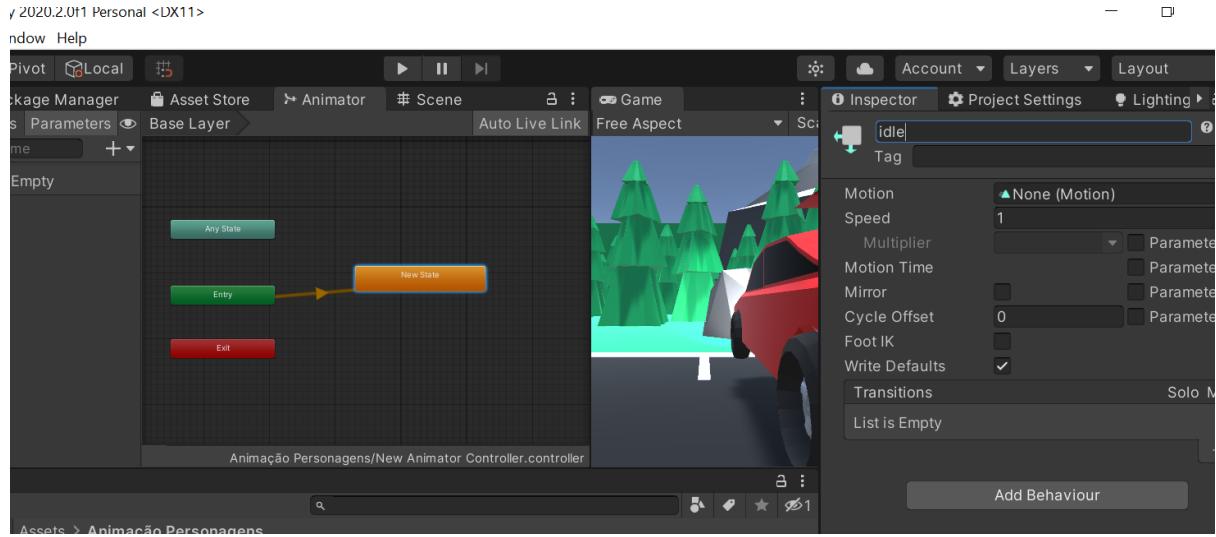
21 - Crie 2 Bool um chamado runZombie e outro idleZombie



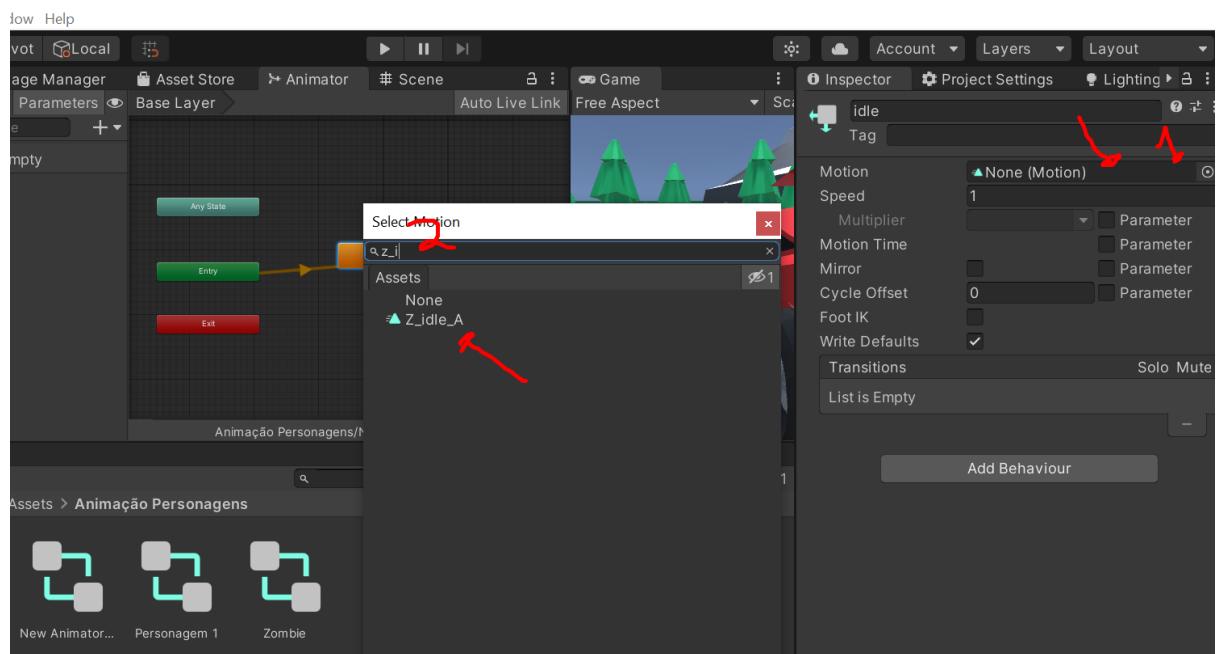
22 - Agora vamos criar os estados da animação. Dê um botão direito, Create State e Empty, irá aparecer um estado na cor laranja.



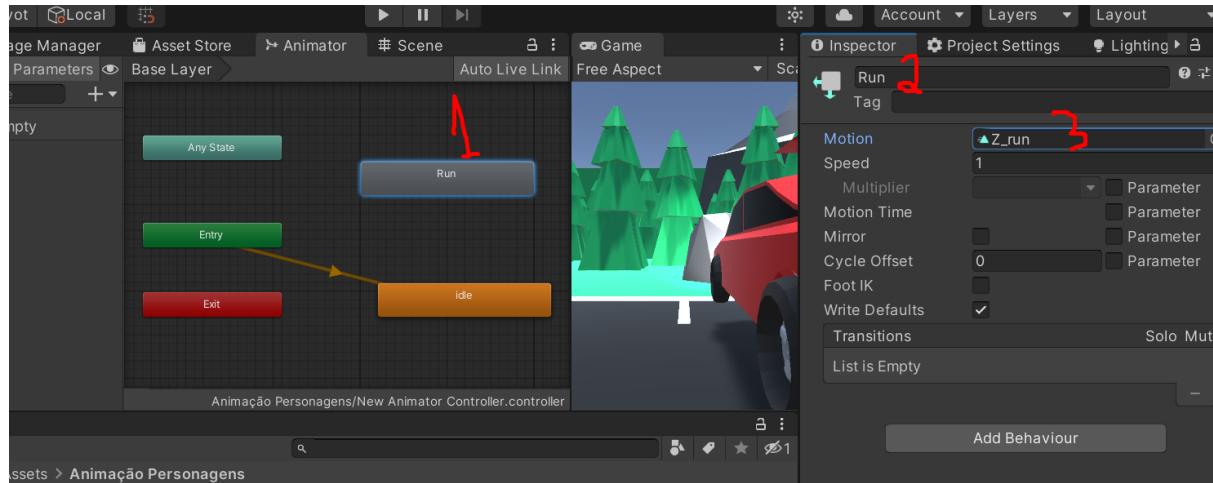
23 - Renomeie esse estado laranja para idle (parado). Sempre que quisermos que o Zumbi fique parado devemos chamar esse estado



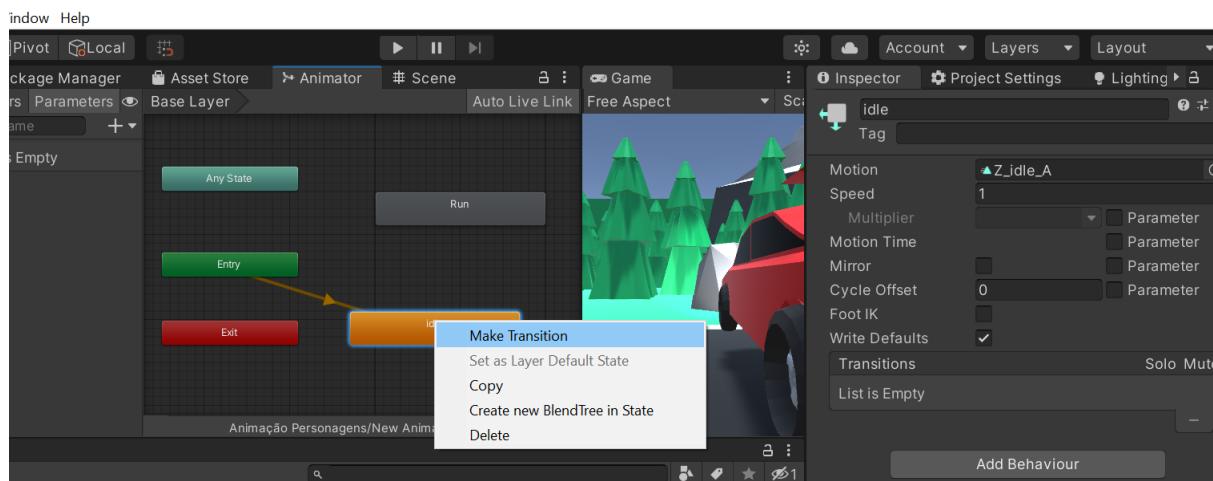
24 - Clique na opção Motion e selecione a animação Z\_idle\_A:



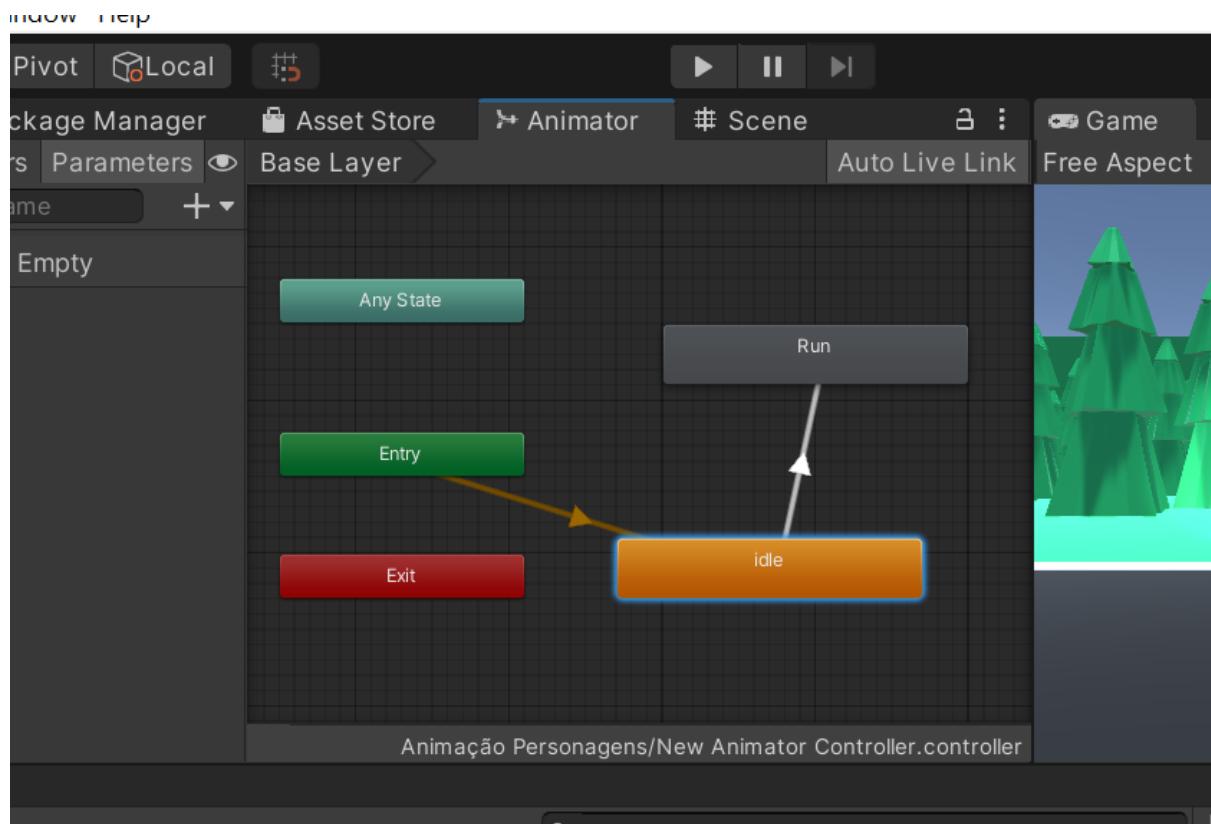
25 - Crie um novo estado, renomeie ele como Run e na opção Motion escolha a animação Z\_run



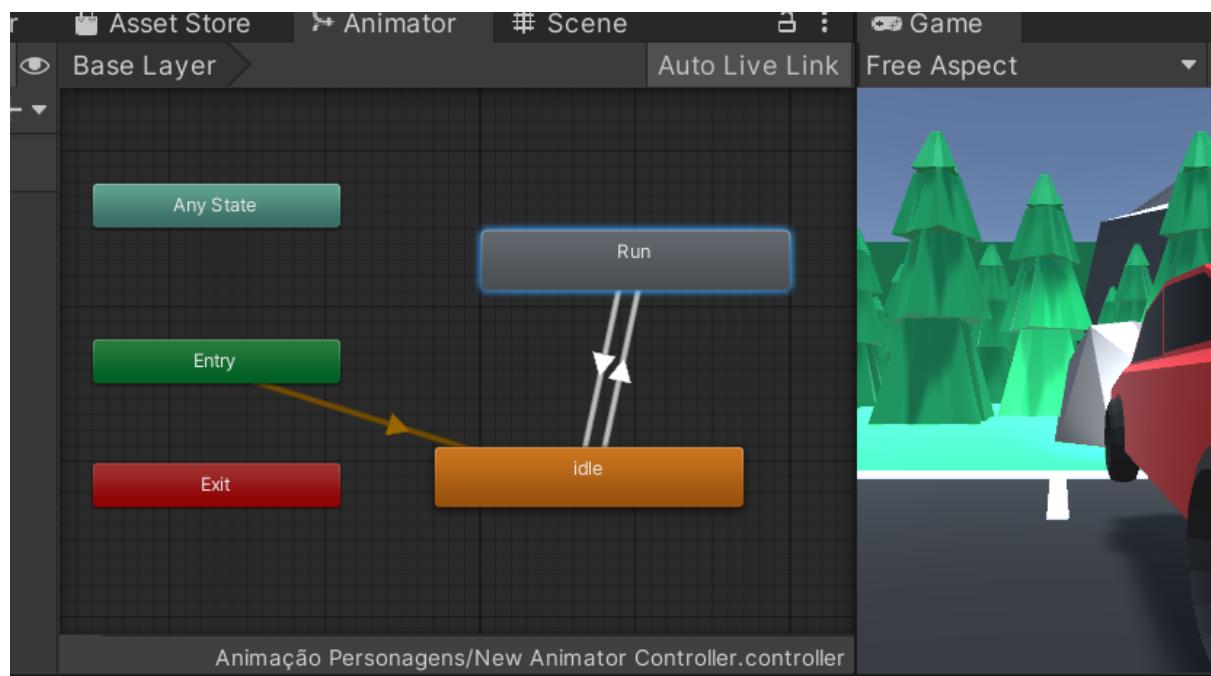
26 - Agora dê um botão direito no estado idle (laranja) e selecione Make Transition:



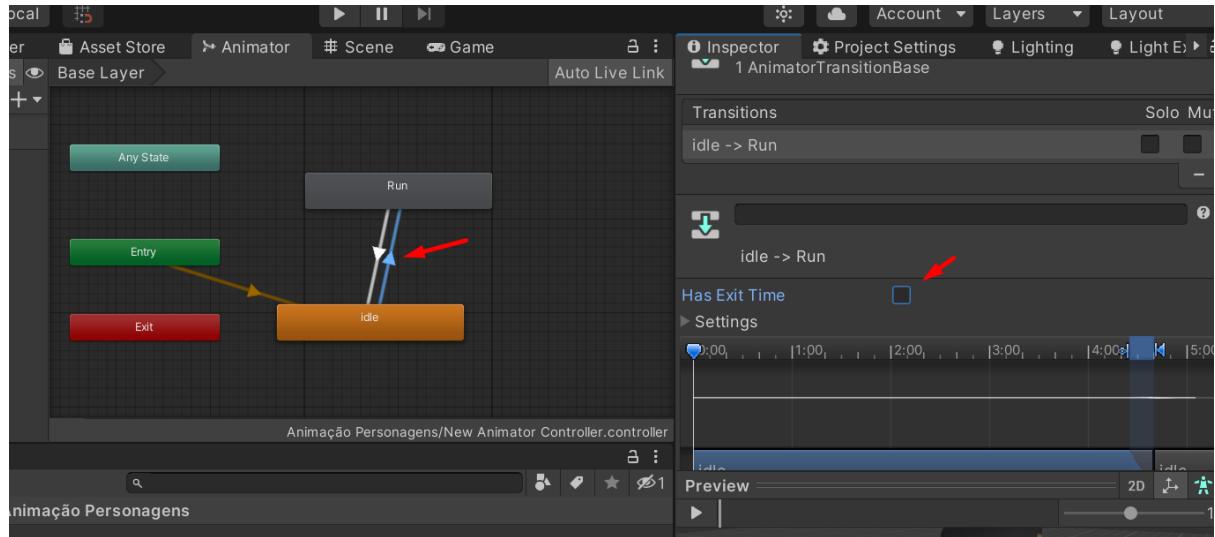
27 - Uma seta irá surgir, clique no estado Run:



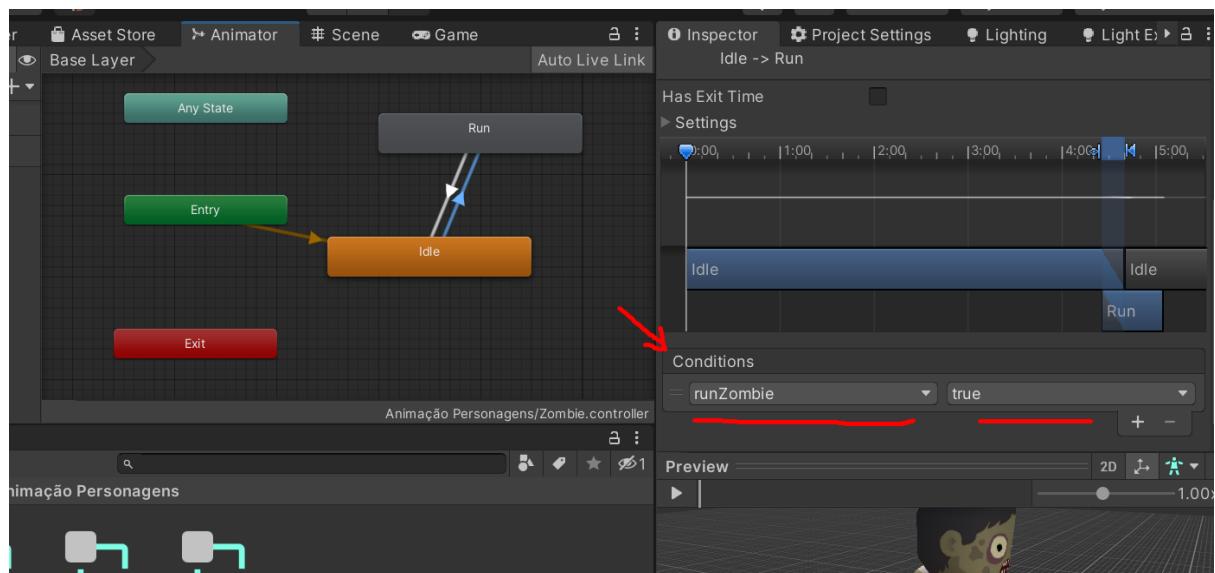
28 - Faça o inverso agora, crie uma transição de Run para Idle



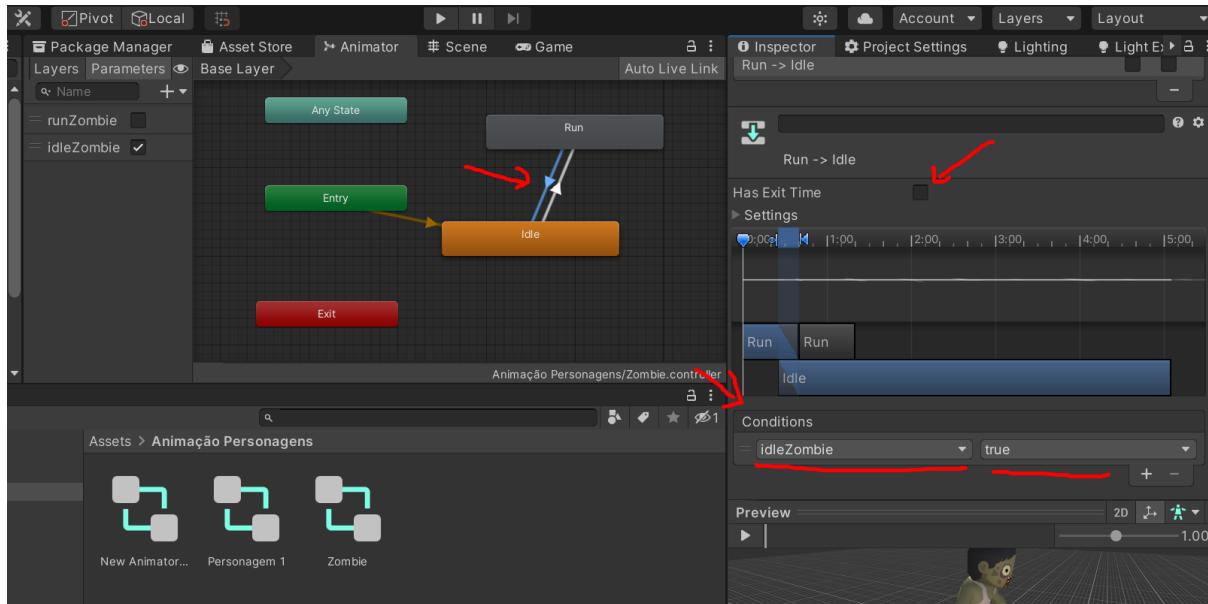
29 - Vamos agora configurar o gatilho, clique na seta que vai de Idle para Run, desmarque a opção Has Exit Time:



30 - Na opção, Conditions criei uma nova condição onde a variável runZombie seja true. Ou seja, essa transição vai ocorrer quando essa variável (runZombie) estiver true, iremos controlar a animação via script depois.



31 - Agora clique na seta que vai de Run para Idle e faça as configurações abaixo indicadas:



32 - Bora programar, crie o script abaixo chamado zombieRadar

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class zombieRadar : MonoBehaviour
{
    private Transform target; //drag and stop player object in the inspector
    public float _range;
    public float speed;
    public float rotationSpeed;
    private Animator zombieAnim;

    void Start()
    {
        zombieAnim = GetComponent<Animator>();
    }

    // Update is called once per frame
    void Update()
    {

    }

    private void OnCollisionEnter(Collision collision)
    {

```

```
if(collision.gameObject.tag == "Personagem")
{
    Debug.Log("Enter");
}

private void OnCollisionStay(Collision collision)
{
    if (collision.gameObject.tag == "Personagem")
    {
        zombieAnim.SetBool("runZombie", true);

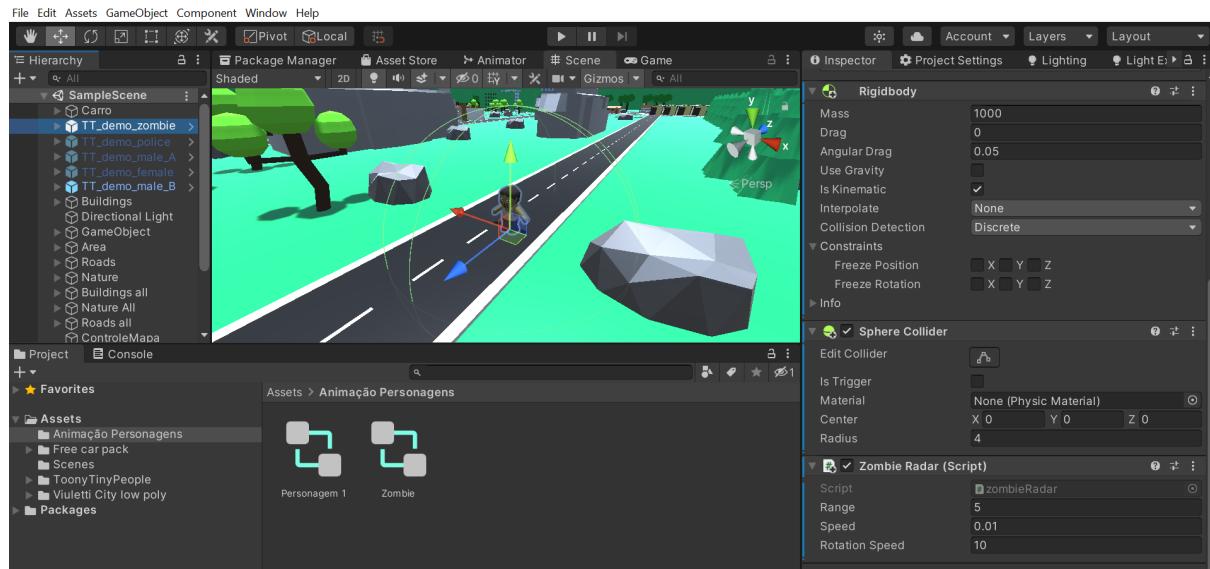
        target = collision.gameObject.transform;
        transform.rotation = Quaternion.Slerp(transform.rotation,
        Quaternion.LookRotation(
        target.position - transform.position),
        rotationSpeed * Time.deltaTime);

        float dist = Vector3.Distance(target.position, transform.position);

        if (dist <= _range)
        {
            transform.position = Vector3.MoveTowards(transform.position,
            target.transform.position, speed);
        }
        Debug.Log("Stay");
    }
}

private void OnCollisionExit(Collision collision)
{
    if (collision.gameObject.tag == "Personagem")
    {
        zombieAnim.SetBool("runZombie", false);
        zombieAnim.SetBool("idleZombie", true);
        transform.position = Vector3.MoveTowards(transform.position,
        target.transform.position, 0);
        Debug.Log("Exit");
    }
}
```

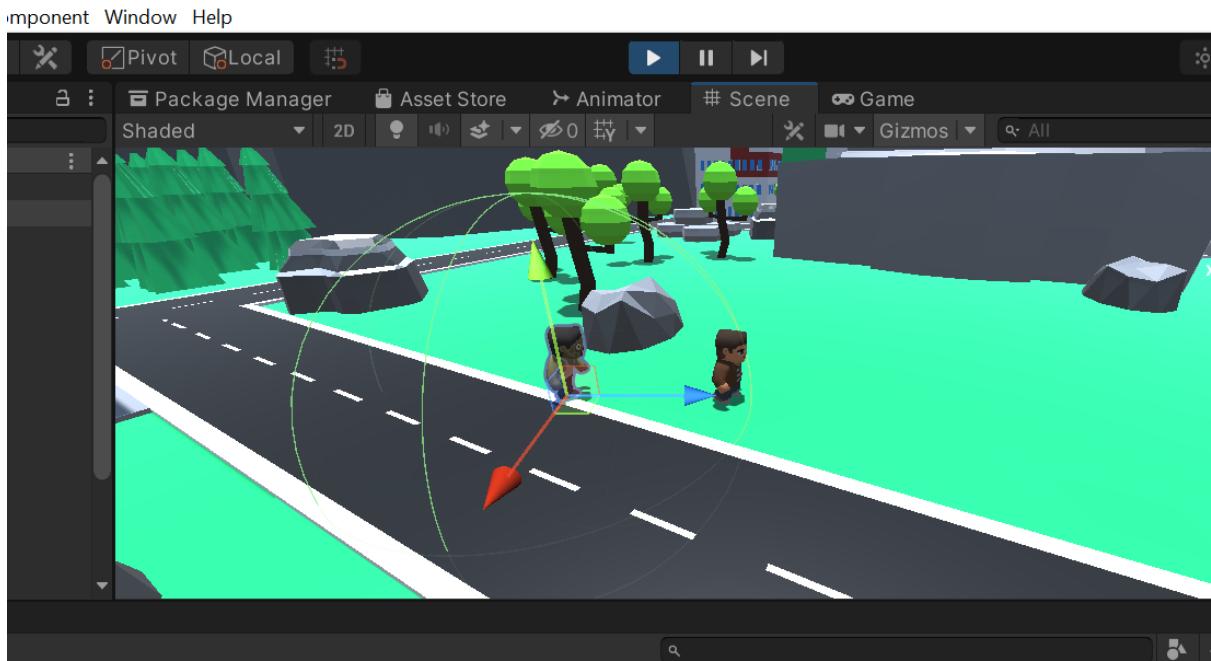
33 - Associe o script zombieRadar ao nosso Zumbi, configure qual será o Range, Speed e Rotation Speed:



34 - Agora clique no personagem, vá na opção Tag e crie uma nova Tag na opção Add Tag, essa tag irá chamar Personagem. Deixe a opção Personagem selecionada.



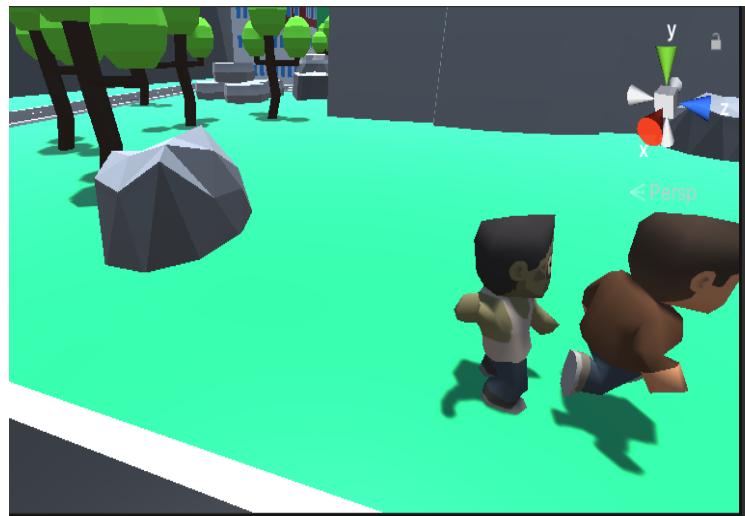
35 - Pronto! Agora o Zumbi correrá atrás do personagem 😊



## Exercícios

01 - Quando o Zumbi chegar bem perto do Personagem ele irá começar uma animação de atacar o personagem. (Dica: precisar adicionar mais transições de animação, e a adicionar algumas linhas de código no script zombieRadar).

02 - Quando o Zumbi chegar bem perto do Personagem o personagem deve mudar a animação de andando (walk) para correndo (run). (Dica: precisar adicionar mais transições de animação, e a adicionar algumas linhas de código no script zombieRadar).



03 - Monte seu apocalipse zumbi, adicione mais pessoas e zumbis no centro da cidade, posicione os waypoints para fazer caminhos.



04 - Quando o Zumbi chegar bem perto do Personagem o personagem deve aumentar sua velocidade atual para poder escapar dos zombies. No script o novo valor da velocidade do personagem será variável dentro de um range, ou seja, alguns vão correr mais rápido e escapar e outros não. (Dica: no script zombieRadar você precisa usar o GetComponent para poder ter acesso a variável “movementSpeed” que está no script followPath).



