

# Tópicos Especiais em Engenharia de Software (Jogos II)

## Aula 06 – Animação

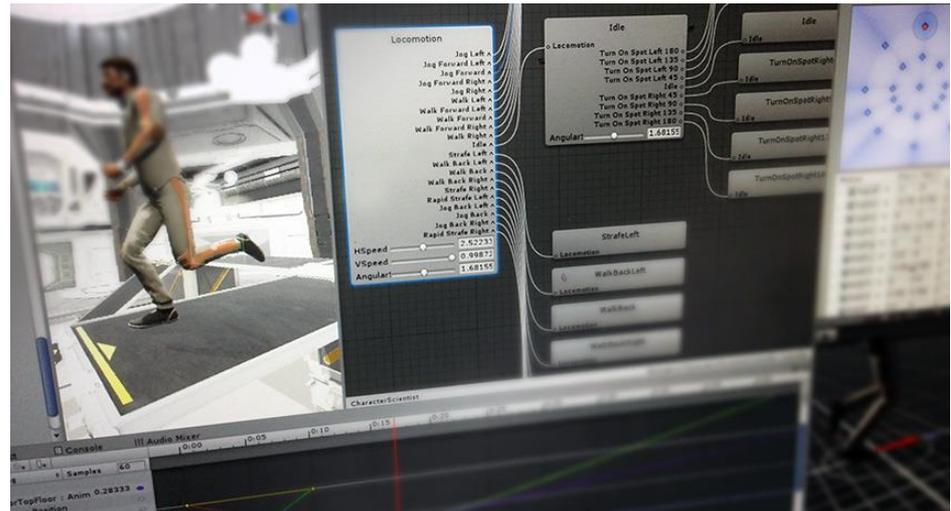
Edirlei Soares de Lima  
<edirlei@iprj.uerj.br>



# Unity 3D: Animação

- A Unity suporta animações criadas nas principais ferramentas de modelagem e animação 3D.

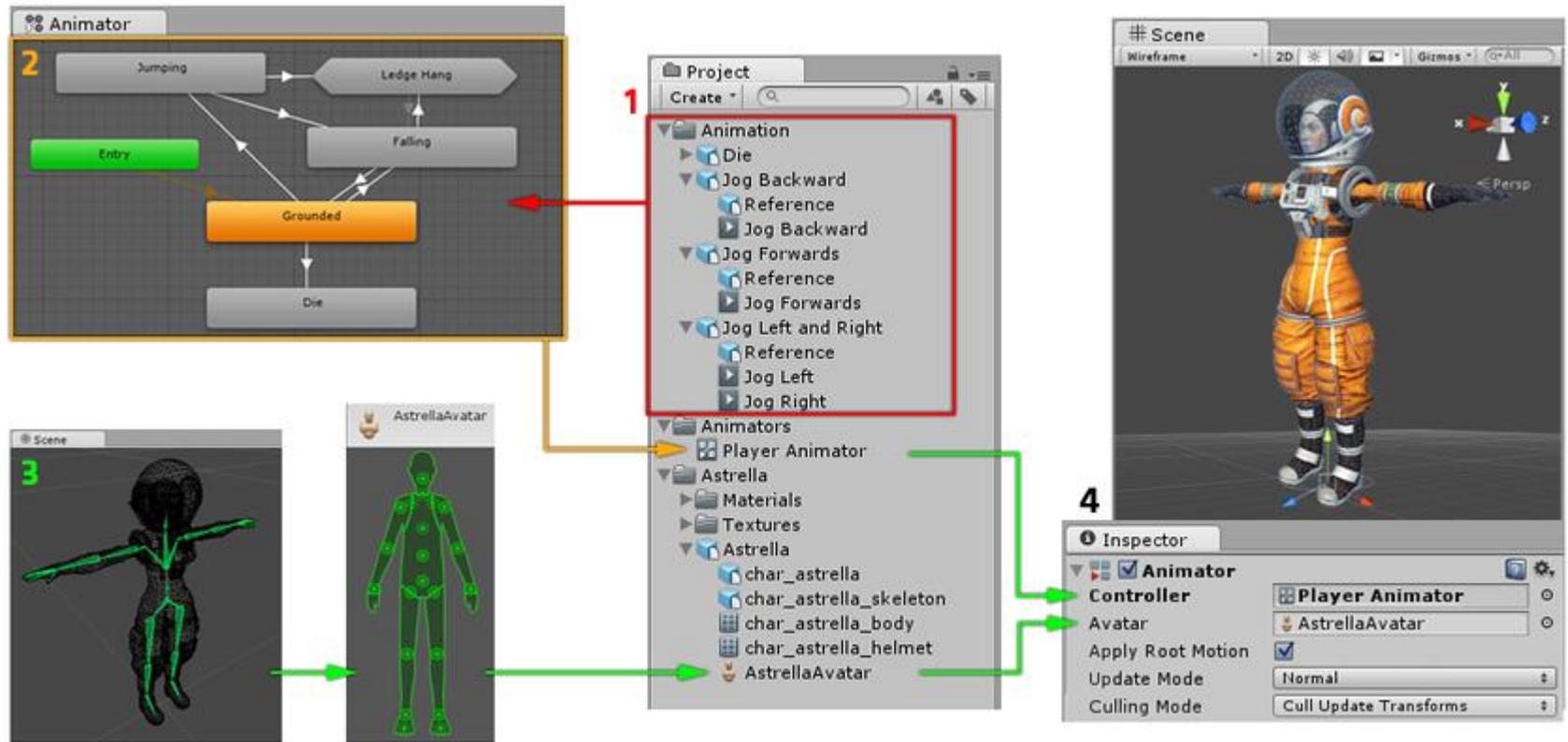
- Animation Clips
- Importação de Animações
- Animator Component
- Animator Controller
- Humanoid Avatars



# Animation System

- O sistema de animação da Unity baseia-se em clipes de animação (Animation Clips), os quais contêm informações sobre como os objetos deve mudar de posição e rotação ao longo do tempo.
  - Clipes de animação de fontes externas são criados por artistas ou animadores com ferramentas de modelagem 3D (3DS Max, Maya, etc.), ou são criados em estúdios de captura de movimento.
  - Também é possível criar e editar clipes de animação pelo sistema de animação da Unity.

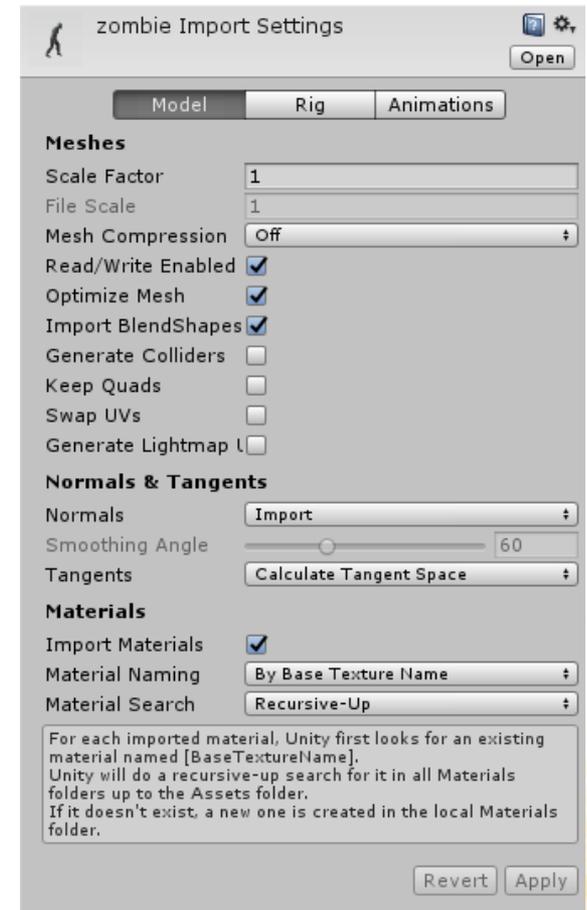
# Animation System



# Animações de Fontes Externas

- Animações de **fontes externas** são importadas para a Unity da mesma forma que os modelos 3D.
  - Em alguns casos as informações de animação estão presentes no mesmo arquivo do modelo 3D, em outros casos podem estar em arquivos separados.
  - Animações podem ser reusadas em outros personagens que compartilhem a mesma estrutura de esqueleto.
- Guia para exportar animações nas principais ferramentas de modelagem 3D: <https://docs.unity3d.com/Manual/HOWTO-importObject.html>
- Exemplos de modelos e animações:  
[http://www.inf.puc-rio.br/~elima/jogos3d/modelos\\_personagens.html](http://www.inf.puc-rio.br/~elima/jogos3d/modelos_personagens.html)

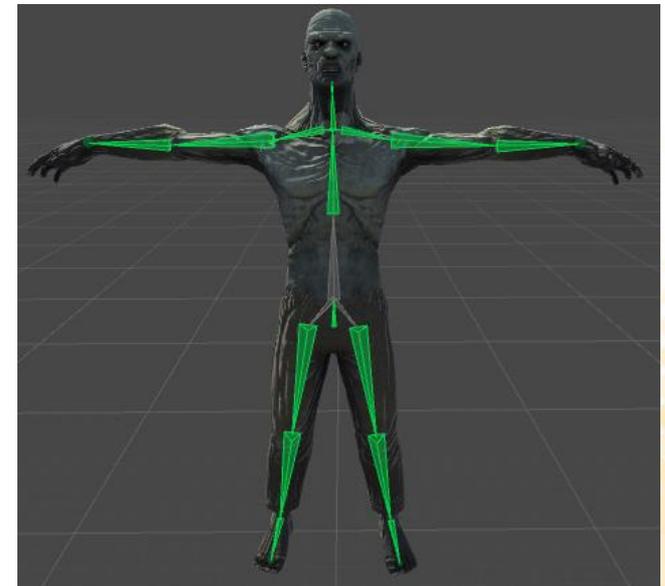
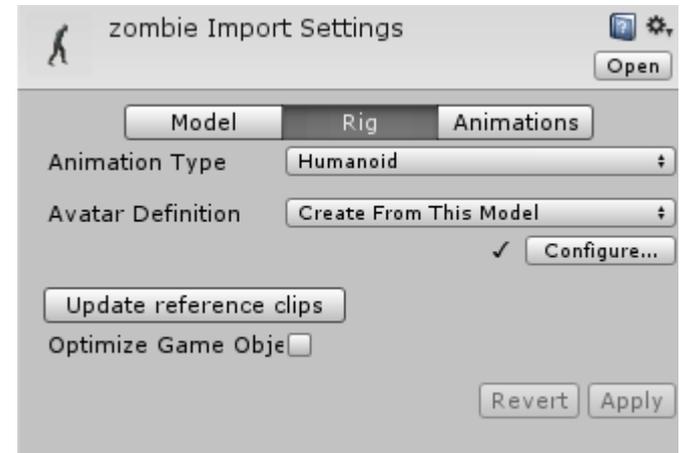
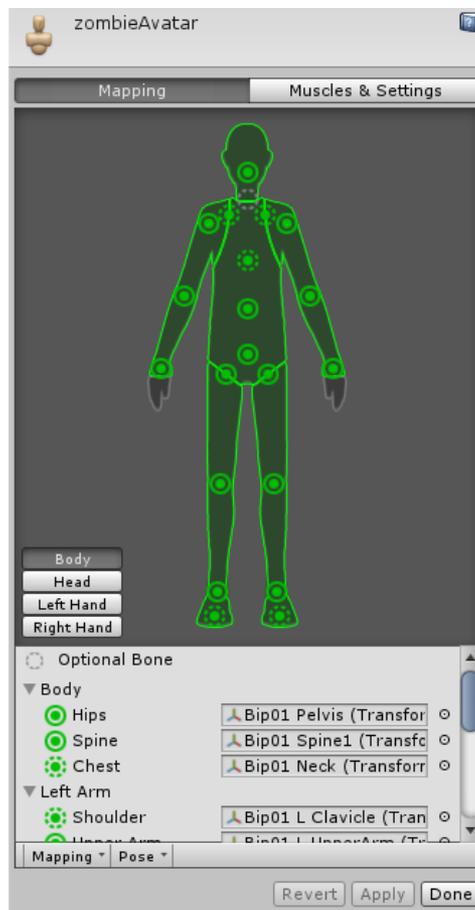
# Importando Animações



- Modelo: [http://www.inf.puc-rio.br/~elima/jogos3d/zombie\\_model.html](http://www.inf.puc-rio.br/~elima/jogos3d/zombie_model.html)

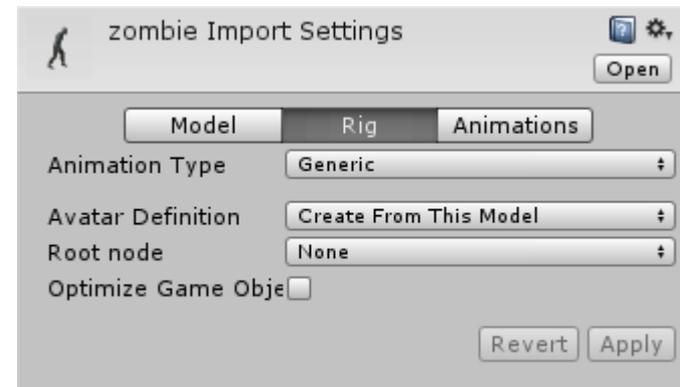
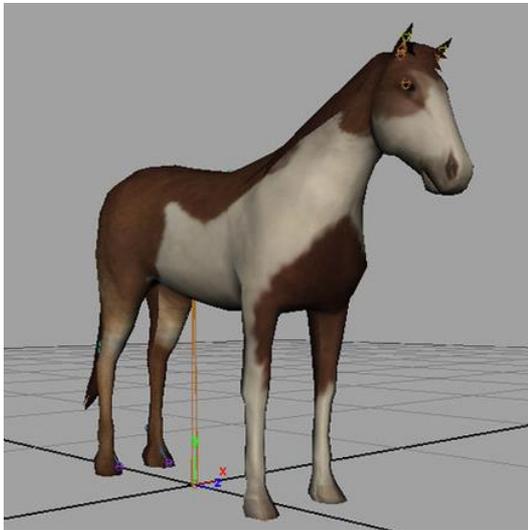
# Humanoid Animations

- Rig -> Animation Type -> Humanoid



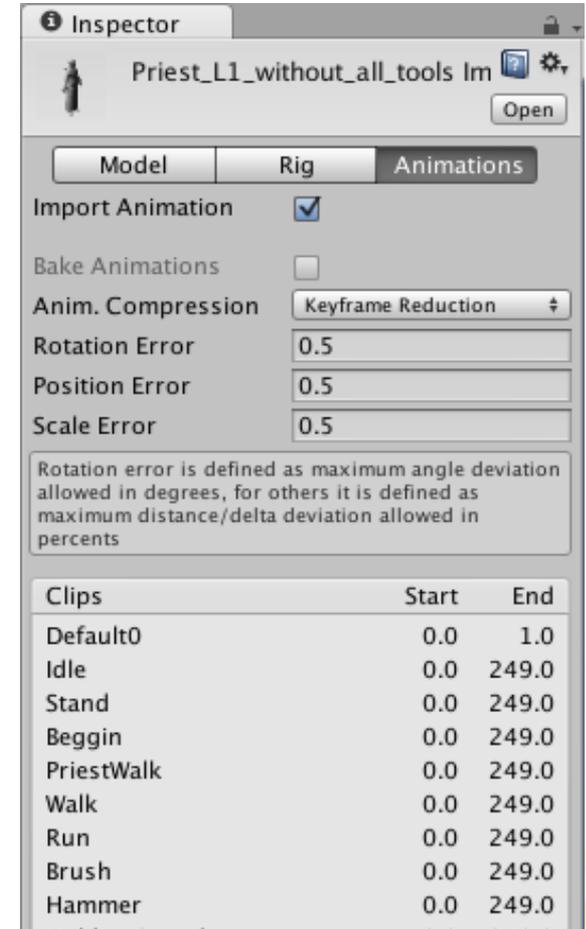
# Generic Animations

- Animações de objetos e personagens que não possuem um esqueleto humano podem ser representadas usando a opção “Generic”:
  - Rig -> Animation Type -> Generic



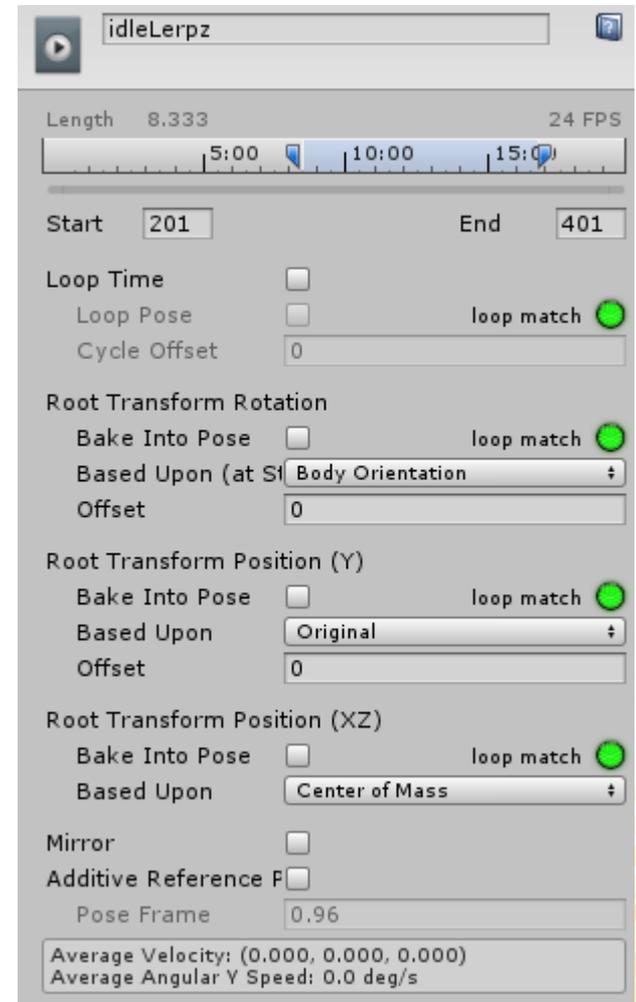
# Animation Clips

- Os cliques de animações importados podem estar definidos de três formas:
  - 1) Um único modelo contém todas as animações já separadas em cliques e nomeadas;
    - Algumas vezes é necessário corrigir os intervalos dos frames;
  - 2) Um único modelo contém todas as animações em um único clique;
    - É necessário incluir, nomear e definir o intervalo de frames de cada animação manualmente.
  - 3) Um arquivo contém a geometria do modelo e outros arquivos contêm os cliques das animações;
    - Os arquivos devem ser nomeados seguindo a nomenclatura: “modelName@animationName.fbx”.



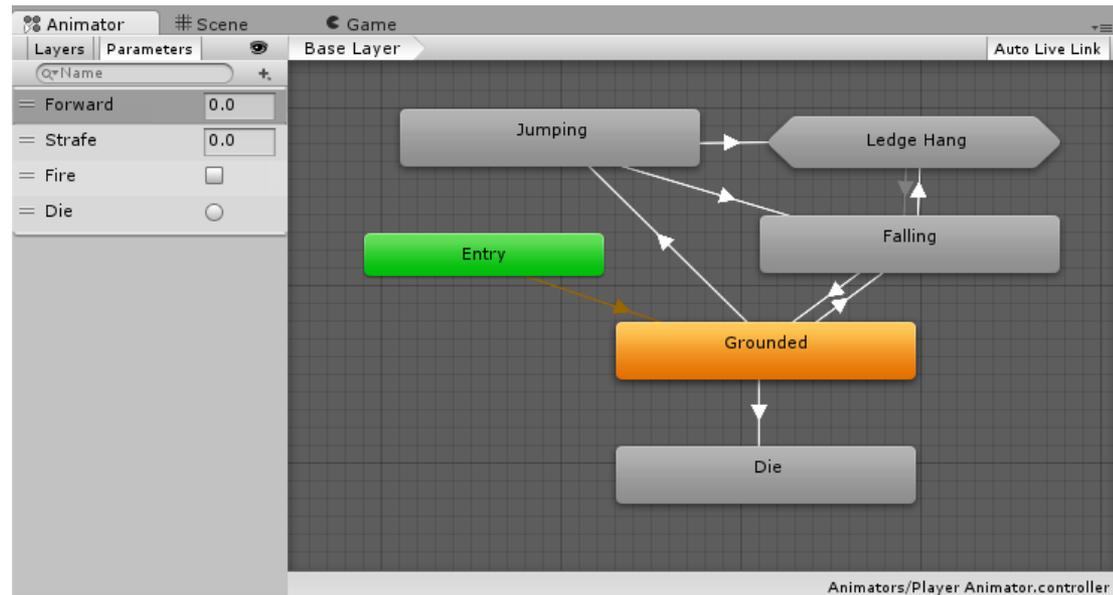
# Looping Animations

- Para a maioria das animações, é extremamente importante que elas crie sequencias em loop que possam ser executadas sem saltos.
  - Os indicadores verdes indicam que os frames da animação estão em loop.
  - Os indicadores laranja indicam que os frames da animação não estão em loop e podem gerar saltos.
  - Os indicadores vermelhos indicam que os frames da animação não estão em loop e vão gerar saltos.



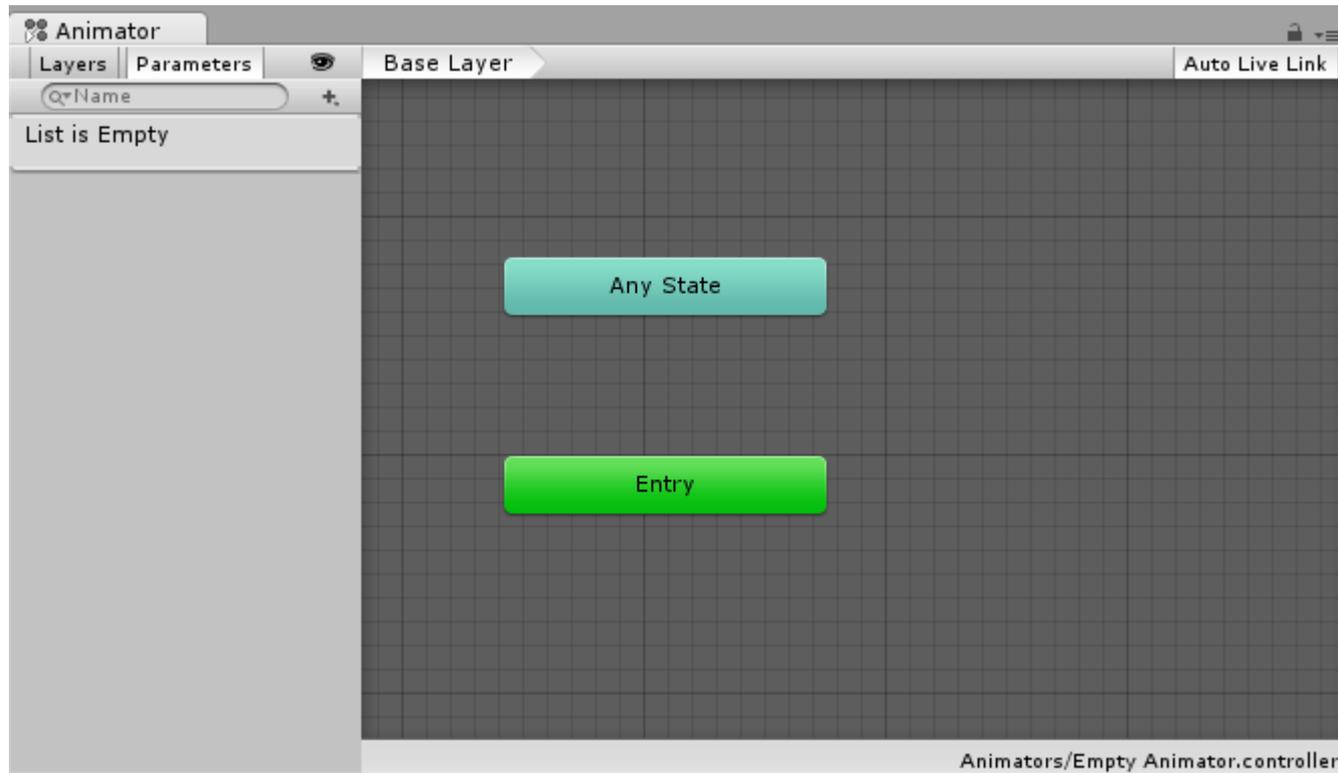
# Animator Controllers

- Um **Animator Controller** permite organizar e manter o conjunto de animações de um personagem ou outro objeto animado.
  - O controlador tem as referências aos clipes de animações e gerencia todos os estados e transições entre eles usando uma **máquina de estados**.
  - Criar um novo Animator Controller: Create -> Animator Controller



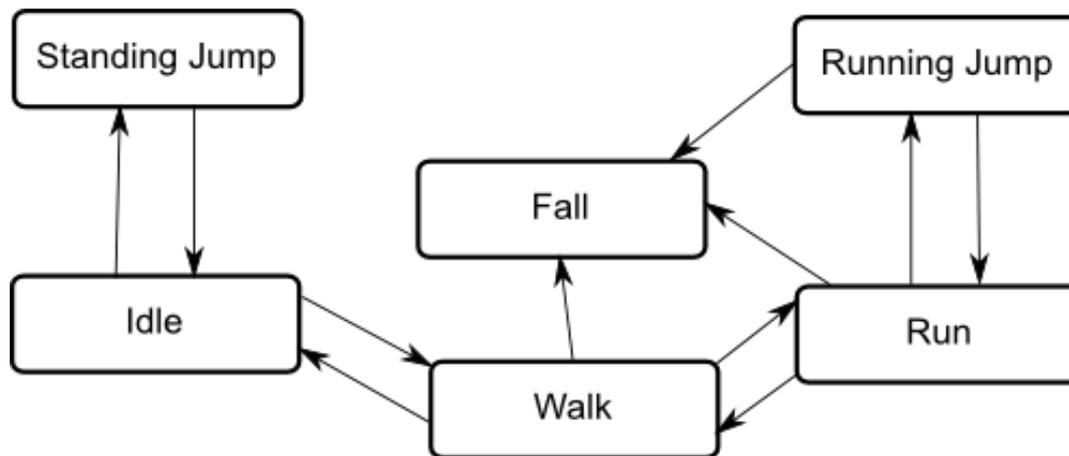
# Animator Controllers

- A janela Animator permite ver e modificar os Animator Controllers:



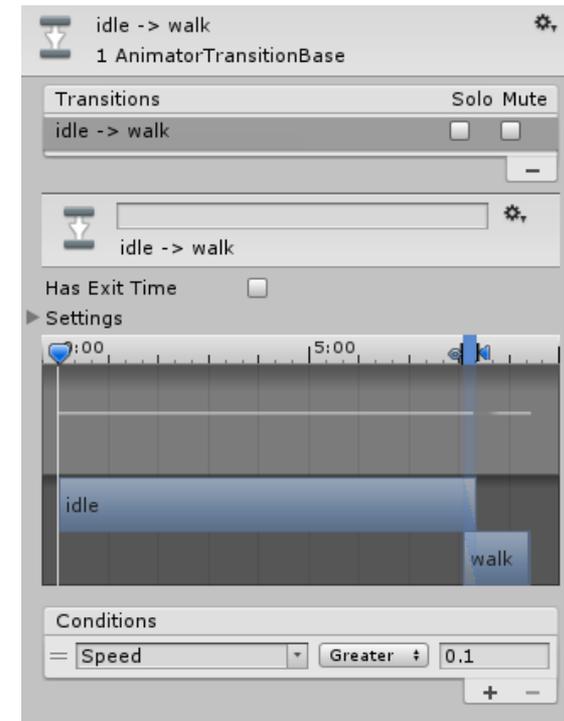
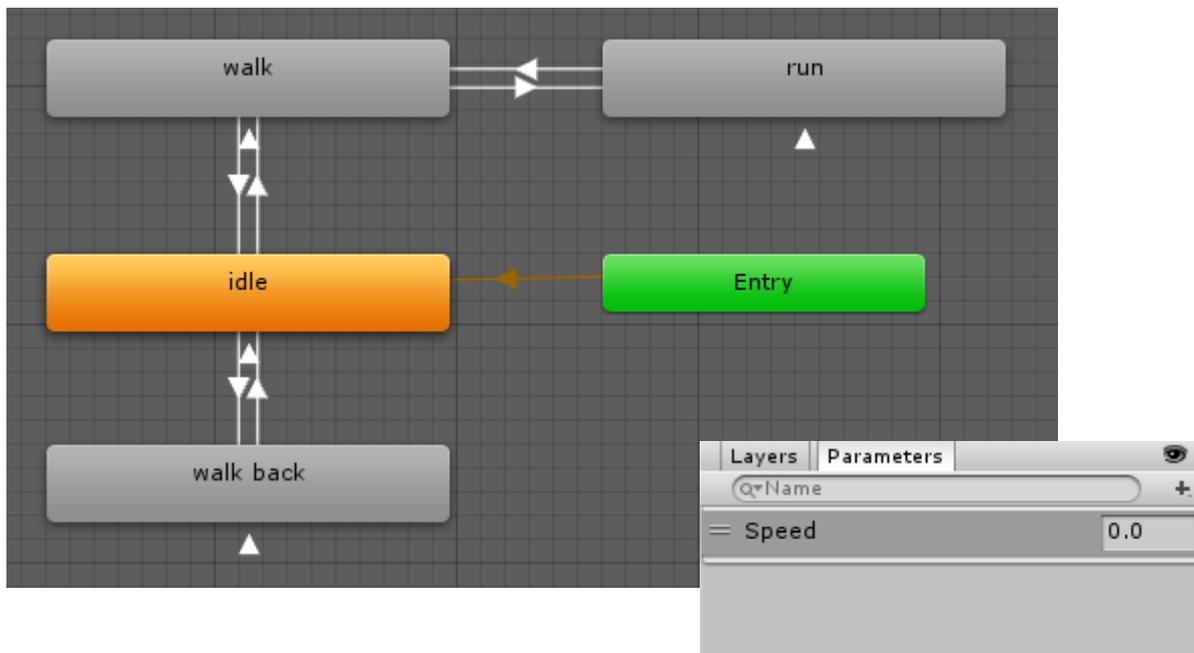
# Máquina de Estados

- As máquinas de estados de animação baseiam-se na ideia de que os personagens de um jogo estão sempre realizando alguma ação (parado, andando, pulando, atacando, etc.).
  - As ações são estados;
  - As transições entre ações são definidas por transições entre estados;
  - Cada transição pode ter um conjunto de pré-condições (parâmetros);



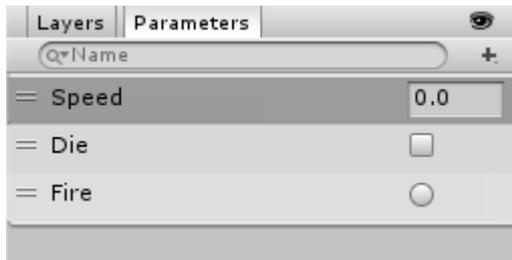
# Máquina de Estados

- É possível criar as máquina de estados visualmente:
  - Animações são associadas aos estados;
  - Transições são estabelecidas entre os estados;
  - Os valores dos parâmetros das transições podem ser controlados por scripts.

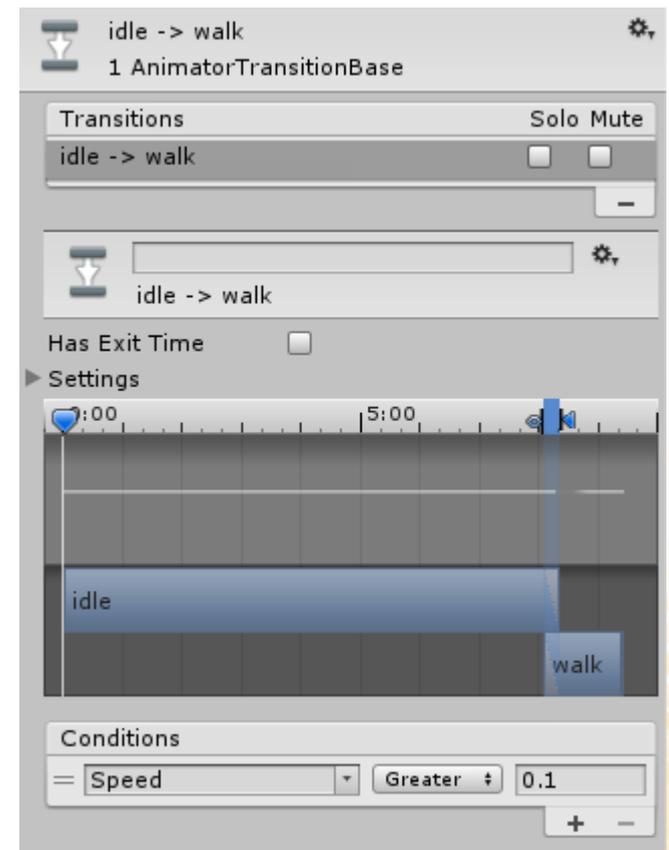


# Parâmetros das Animações

- Os parâmetros das animação são variáveis que são definidas dentro do Animator Controller e podem ser usadas como condições para as transições.



- As variáveis podem ser acessadas por scripts:
  - SetFloat, SetInt, SetBool, SetTrigger



# Controlando Animações

```
public class Player : MonoBehaviour {
    public float WalkSpeed = 70;
    public float RotationSpeed = 50;
    private Animator animator;
    private CharacterController ch;

    void Start(){
        animator = GetComponent<Animator>();
        ch = GetComponent<CharacterController>();
    }

    void Update(){
        float forwardMove = Input.GetAxis("Vertical");
        float rotation = Input.GetAxis("Horizontal");
        animator.SetFloat("Speed", forwardMove);
        ch.SimpleMove(transform.forward * forwardMove *
                       WalkSpeed * Time.deltaTime);
        transform.Rotate(Vector3.up * rotation *
                        RotationSpeed * Time.deltaTime);
    }
}
```

# Exercício 16

- Utilize o Animator Controller para criar uma maquina de estados para controlar os comportamentos e animações do Lerpz.
  - a) A maquina de estados ser composta pelos seguintes estados/animações: parado, andando, correndo, pulando, chutando e socando.
  - b) Em seguida, implemente um script para utilizar a maquina de estados para controlar e animar o Lerpz.

<http://www.inf.puc-rio.br/~elima/jogos3d/lerpz.html>

