Uma imagem com Tipo de letra, logótipo, Gráficos, texto

Descrição gerada automaticamente

JavaLife

Programação Avançada

2023/2024

João Almas | 2021138417

João Pinto | 2022143707

Paulo Sá | 2021142819

Índice

[Introdução 2](#_Toc168233204)

[Decisões tomadas na Implementação 2](#_Toc168233205)

[*Finite State Machine* 2](#_Toc168233206)

[Outros *Softwate Design Patterns* aplicados 2](#_Toc168233207)

[Classes utilizadas 2](#_Toc168233208)

[Relacionamento entre as Classes 2](#_Toc168233209)

[Funcionalidades 2](#_Toc168233210)

[Conclusão 2](#_Toc168233211)

# Introdução

No âmbito da unidade curricular de Programação Avançada foi proposto o desenvolvimento da aplicação para simulação de evolução de um ecossistema ao longo do tempo: **JavaLife**.

# Decisões tomadas na Implementação

O único ponto no enunciado deste trabalho prático que não é claro acerca do seu modo de implementação é a representação do elemento Fauna para o resto do sistema, sendo que utiliza uma Finite State Machine.

Esta dúvida surge porque a classe "Ecossistema" agrega os elementos num conjunto (Set) de objetos que implementam a interface "IElemento". Esta interface só pode ser implementada por objetos derivados da classe "ElementoBase". A classe "ElementoBase" só pode derivar três classes: "Inanimado", "Flora" e "Fauna".

Na Finite State Machine do elemento Fauna, a classe utilizada para a representar no resto do sistema é a classe "FaunaContext", o que cria um conflito de interesses, uma vez que a classe "Ecossistema" não poderia agregar a classe "FaunaContext".

A solução encontrada foi renomear a classe "FaunaContext" para "Fauna", que consequentemente passará a derivar da classe "ElementoBase" e a implementar as interfaces "IElemento" e "IElementoComForca". A classe anteriormente chamada "Fauna" passará a chamar-se "FaunaData" e ficará responsável apenas por guardar os dados do elemento, à exceção da área, que é herdada da classe "ElementoBase"

# *Finite State Machine* - Fauna

|  |  |
| --- | --- |
| **Ações (Interface)** | **Estados (*Enum*)** |
| * ***Act*** | * ***Moving*** * ***Eating*** * ***LookingForFood*** * ***Hunting*** * ***ChasingPartner*** * ***Dead*** |

Transições de Estados

|  |  |  |
| --- | --- | --- |
| **Estado Atual** | **Possíveis Transições** | **Descrição** |
| *Moving* | T1 → *ChasingPartner* | **Ação**: Act  **Condições**:   * Força > 50 |
| T2 → LookingForFood | **Ação**: Act  **Condições**:   * Força < 35 * Existência de árvores no Ecossistema |
| T3 → *Hunting* | **Ação**: Act  **Condições**:   * Força < 35 * Existem árvores no Ecossistema |
| T4 → *Dead* | **Ação**: Act  **Condições**:   * PosiçãoAdjacente == Animal * Estado atual do Animal == *“Hunting”.* * Forca < Força do Animal **OU** Força < 10 |

|  |  |  |
| --- | --- | --- |
| **Estado Atual** | **Possíveis Transições** | **Descrição** |
| *Eating* | T5 → *Moving* | **Ação**: Act  **Condições**:   * PosiçãoAdjacente != Arvore * PosiçãoAdjacente != Animal * Força >= 80 |
| T6 → *LookingForFood* | **Ação**: Act  **Condições**:   * PosiçãoAdjacente != Arvore * PosiçãoAdjacente != Animal * Força < 80 * Existem árvores no Ecossistema |
| T7 → *Hunting* | **Ação**: Act  **Condições**:   * PosiçãoAdjacente != Arvore * PosiçãoAdjacente != Animal * Força < 80 * Não existem árvores no Ecossistema |
| T8 → *Dead* | **Ação**: Act  **Condições**:   * PosiçãoAdjacente == Animal * Estado atual do Animal == *“Hunting”* * Forca < Força do Animal **OU** Força < 10 |

|  |  |  |
| --- | --- | --- |
| **Estado Atual** | **Possíveis Transições** | **Descrição** |
| *LookingForFood* | T9 → *Eating* | **Ação**: Act  **Condições**:   * PosiçãoAdjacente == Arvore |
| T10 → Dead | **Cenário 1**  **Ação**: Act  **Condições**:   * PosiçãoAdjacente != Arvore * Força <= 0 |
| **Cenário 2**  **Ação**: Act  **Condições**:   * PosiçãoAdjacente == Animal * Estado atual do Animal == “*Hunting*” * Força < Força do Animal **OU** Força < 10 |

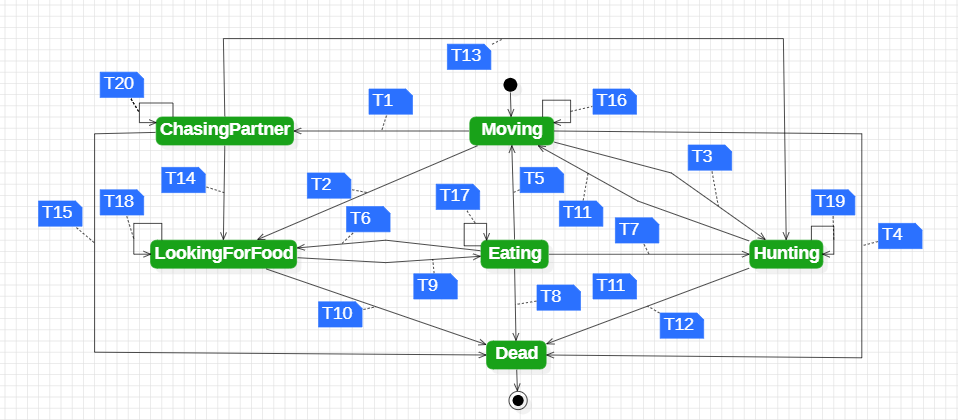
|  |  |  |
| --- | --- | --- |
| **Estado Atual** | **Possíveis Transições** | **Descrição** |
| *Hunting* | T11 → *Moving* | **Ação**: Act  **Condições**:   * PosiçãoAdjacente == Animal * Estado atual do Animal == **“***Dead***”** * Força > 0 |
| T12 → *Dead* | **Cenário 1**  **Ação**: Act  **Condições**:   * PosiçãoAdjacente == Animal * Estado atual do Animal == “*Hunting*” * Força < Força do Animal **OU** Força < 10 |
| **Cenário 2**  **Ação**: Act  **Condições**:   * PosiçãoAdjacente == Animal * Estado atual do Animal != “*Hunting*” * Força < 10 |
| **Cenário 3**  **Ação**: Act  **Condições**:   * PosiçãoAdjacente != Animal * Força <= 0 |

|  |  |  |
| --- | --- | --- |
| **Estado Atual** | **Possíveis Transições** | **Descrição** |
| *ChasingPartner* | T13 → *Hunting* | **Cenário 1**  **Ação**: Act  **Condições**:   * Força < 35 * Não existem árvores no Ecossistema |
| **Cenário 2**  **Ação**: Act  **Condições**:   * Força < 35 (após a multiplicação) * Não existem árvores no Ecossistema |
| T14 → *LookingForFood* | **Cenário 1**  **Ação**: Act  **Condições**:   * Força < 35 * Existem árvores no Ecossistema |
| **Cenário 2**  **Ação**: Act  **Condições**:   * Força < 35 (após a multiplicação) * Existem árvores no Ecossistema |
| T15 →  *Dead* | **Ação**: Act  **Condições**:   * PosiçãoAdjacente == Animal * Estado atual do Animal == “*Hunting*” * Forca < Força do Animal **OU** Força < 10 |

Auto Transições

|  |  |  |  |
| --- | --- | --- | --- |
| **Estado Atual** | **N.º** | | **Descrição** |
| ***Moving*** | T16 | | **Ação**: Act  **Condições**:   * Força >= 35 * Força <= 50 |
| ***Eating*** | T17 | | **Ação**: Act  **Condições**:   * PosiçãoAdjacente == Arvore   **OU** PosiçãoAdjacente == Animal   * Estado atual do Animal == Dead * Força < 100 |
| ***LookingForFood*** | T18 | | **Ação**: Act  **Condições**:   * PosiçãoAdjacente != Arvore * Força > 0 |
| ***Hunting*** | T19 | | **Ação**: Act  **Condições**:   * PosiçãoAdjacente != Animal * Força > 0 |
| ***ChasingPartner*** | T20 | **Cenário 1**  **Ação**: Act  **Condições**:   * Counter < 10 * Força > 0 | |
| **Cenário 2**  **Ação**: Act  **Condições**:   * Força > 50 | |

Figura 1 – Diagrama de Estados



# Outros *Softwate Design Patterns* aplicados

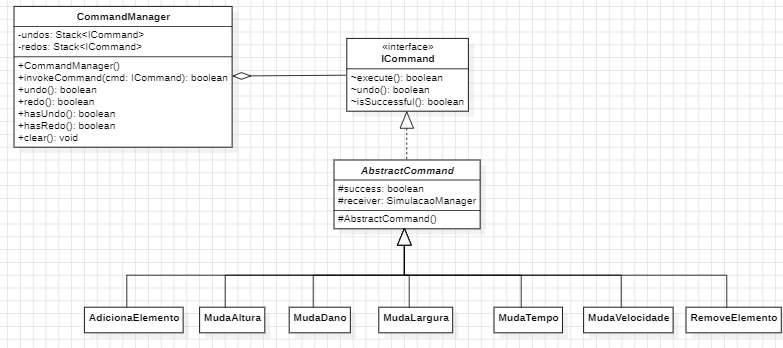
* Command

Figura 2 - Command

* Uma imagem com texto, captura de ecrã, file, Tipo de letra

  Descrição gerada automaticamenteMemento

Figura 3 - Multiton

* Uma imagem com texto, diagrama, file, captura de ecrã

  Descrição gerada automaticamenteFacade

Figura 4 - Facade

* Uma imagem com texto, captura de ecrã, Tipo de letra, file

  Descrição gerada automaticamenteMultiton

Figura 5- Multiton

# Classes utilizadas

Comandos e Gestão de Comandos

* **AbstractCommand**: Classe abstrata para comandos, definindo a interface para execução e desfazer operações.
* **AdicionaElemento**: Comando para adicionar um novo elemento ao ecossistema.
* **MudaAltura**: Comando para alterar a altura de um elemento.
* **MudaDano**: Comando para alterar o dano associado a um elemento.
* **MudaLargura**: Comando para alterar a largura de um elemento.
* **MudaTempo**: Comando para alterar o tempo associado a uma operação ou estado.
* **MudaVelocidade**: Comando para alterar a velocidade de um elemento.
* **RemoveElemento**: Comando para remover um elemento do ecossistema.
* **CommandManager**: Classe responsável por gerenciar e executar comandos, incluindo a capacidade de desfazer e refazer comandos.

Ecossistema e Elementos

* **Ecossistema**: Classe principal que contém e gere todos os elementos do ecossistema.
* **Elemento**: Interface ou classe base representando um elemento genérico no ecossistema.
* **ElementoBase**: Classe base de onde todos os elementos do ecossistema (como fauna e flora) derivam.
* **Fauna**: Representa os elementos de fauna no ecossistema.
* **FaunaData**: Classe responsável por armazenar os dados dos elementos de fauna.
* **Flora**: Representa os elementos de flora no ecossistema.

*Finite State Machine* e Simulação

* **Simulacao**: Classe que encapsula a lógica de simulação do ecossistema.
* **SimulacaoManager**: Classe responsável por gerir a execução da simulação.
* **SimulacaoState**: Classe ou interface para representar estados diferentes na simulação.
* **ChasingPartnerState**: Estado representando a busca por um parceiro.
* **DeadState**: Estado representando um elemento morto.
* **EatingState**: Estado representando a alimentação de um elemento.
* **HuntingState**: Estado representando a caça de um elemento.
* **LookingForFoodState**: Estado representando a procura de alimento.
* **MovingState**: Estado representando o movimento de um elemento.
* **FaunaState**: Estado específico para elementos de fauna.
* **FaunaStateAdapter**: Adaptador para transições entre diferentes estados de fauna.

Snapshot

* **Snapshot**: Representa um snapshot do estado atual do ecossistema.

Gestão de Recursos e Imagens

* **CSSManager**: Classe responsável pela gestão de folhas de estilo CSS.
* **ImageManager**: Classe responsável pela gestão de imagens utilizadas na simulação.
* MultitonImage: Implementação do padrão Multiton para gerir instâncias únicas de imagens

Interface Gráfica do Utilizador

* **MainMenuUI**: Classe para o menu principal da interface gráfica.
* **MenuAddElemento**: Classe para o menu de adição de elementos.
* **MenuDefinicao**: Classe para o menu de definições.
* **MenuEditeElem**: Classe para o menu de edição de elementos.
* **MenuRemoveElem**: Classe para o menu de remoção de elementos.
* **MenuSimulacao**: Classe para o menu de simulação.
* **RootPane**: Classe principal do painel de interface gráfica.
* **ToolbarSimulacao**: Barra de ferramentas para controlar a simulação.
* **ToolbarToggles**: Barra de ferramentas para alternar opções na interface gráfica.

Motor de Jogo e Interfaces

* **GameEngine**: Classe principal do motor de jogo que executa a lógica da simulação.

# Relacionamento entre as Classes

Uma imagem com diagrama, Desenho técnico, Esquema, esboço

Descrição gerada automaticamente

Figura 6 - Relações entre Classes

# Funcionalidades

# Conclusão