

Simulador de Ambiente de Execução (SAE)

Classe Simulador

Simulador de ambiente com visualização gráfica

Construtor

__init__(num_amb: int, controlo: Controlo, largura: int=LARGURA, reiniciar: bool=False)

num_amb: número do ambiente (1 – 5)

controlo: controlo do agente

largura: largura do ambiente em pixeis

reiniciar: reiniciar simulação True/False

LARGURA = 600 (valor por omissão da largura do ambiente)

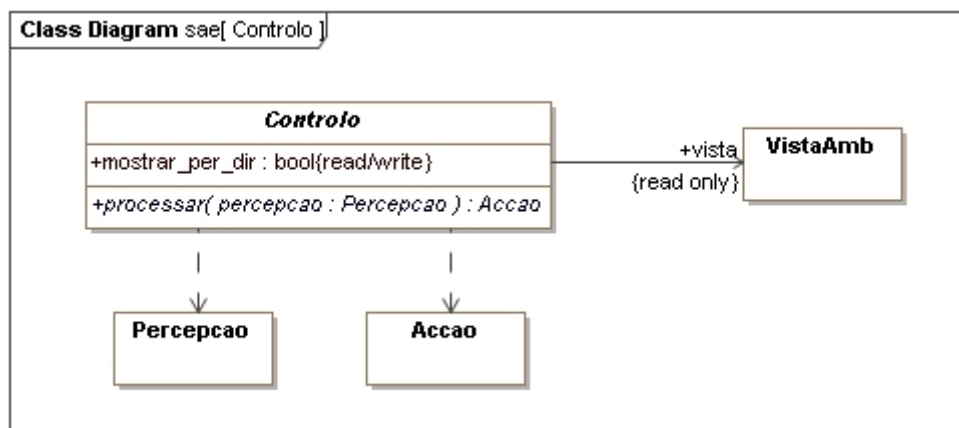
Métodos

executar()

Executar simulação

Classe Controlo {abstract}

Classe base para implementação do controlo de um agente



Propriedades

vista {read only}: vista para visualização de informação do agente

mostrar_per_dir {read/write}: mostrar percepção direccional True/False

Métodos

processar(percepcao: Percepcao): Accao {*abstract*}

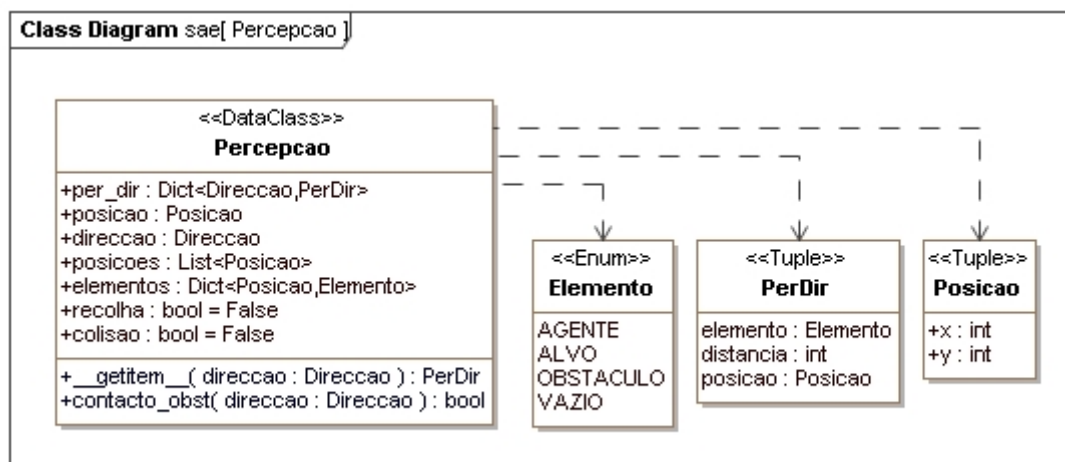
Processar percepção

percepcao: percepção a processar

return: acção a realizar

Classe Percepcao

Registo de informação sensorial



Propriedades

per_dir: percepção direccional nas várias direcções

posicao: posição do agente

direccao: direcção do agente

posicoes: lista de posições do ambiente

elementos: elementos do ambiente

recolha: ocorreu uma recolha de alvo True/False

colisao: ocorreu colisão com obstáculo True/False

Métodos

__getitem__(direccao: Direccao): PerDir

Acesso indexado à percepção direccional

direccao: direcção de percepção

return: percepção direccional (elemento, distância, posição)

contacto_obst(direccao: Direccao): bool

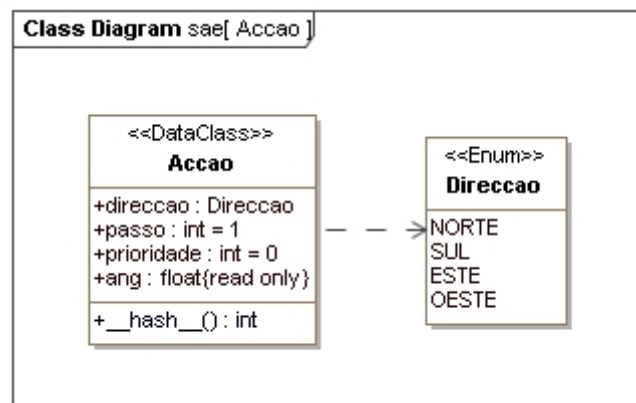
Obter informação de contacto com obstáculo

direccao: direcção de percepção

return: contacto com obstáculo True/False

Classe Accao

Representação de acção do agente



Propriedades

direccao: direcção de movimento

passo = 1: distância de movimento

prioridade = 0: prioridade da acção

ang: ângulo da direcção da acção

Métodos

__hash__()

Identificação por valor

return: identificação da acção

Enum Elemento

Elemento do ambiente

AGENTE, ALVO, OBSTACULO, VAZIO

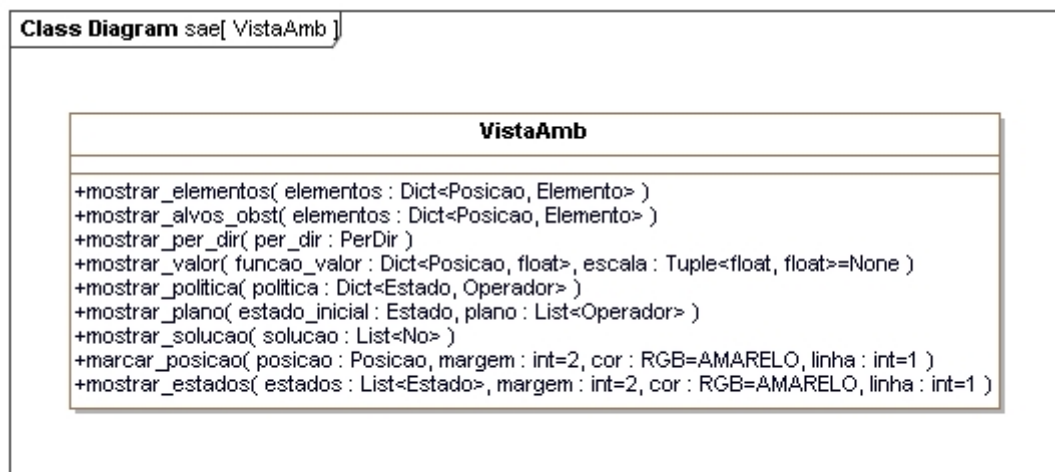
Enum Direccao

Direcção de movimento

NORTE, SUL, ESTE, OESTE

Classe VistaAmb

Vista de apresentação de informação de ambiente e de agente



Métodos

mostrar_elementos(elementos: Dict<Posicao, Elemento>)

Visualizar elementos do ambiente

elementos: dicionário com elementos a mostrar

mostrar_alvos_obst(elementos: Dict<Posicao, Elemento>)

Visualizar alvos e obstáculos de um conjunto de elementos

elementos: dicionário com elementos, dos quais serão mostrados os alvos e os obstáculos

mostrar_per_dir(per_dir: PerDir)

Mostrar percepção direccional

per_dir: percepção direccional

mostrar_valor(funcao_valor: Dict<Posicao, float>, escala: Tuple<float, float>=None)

Visualizar função valor

funcao_valor: dicionário {posição: valor}

escala: (valor mínimo, valor máximo)

mostrar_politica(politica: Dict<Estado, Operador>)

Visualizar política

politica: dicionário {posição: operador}, operador deve ter propriedade *ang*

mostrar_plano(estado_inicial: Estado, plano: List<Operador>)

Visualizar plano

estado_inicial: estado inicial

plano: sequência de operadores com propriedade *ang*

mostrar_solucacao(solucacao: List<No>)

Visualizar solução de PEE

solucacao: sequência de nós com operadores com propriedade *ang*

marcar_posicao(posicao: Posicao, margem:int=2, cor:RGB=AMARELO, linha:int=1)

Marcar posição

posicao: posição a marcar

margem: margem em pixels

cor: RGB

linha: espessura de linha (0 - preencher)

mostrar_estados(estados: List<Estado>, margem:int=2, cor:RGB=AMARELO, linha:int=1)

Mostrar conjunto de estados

estados: conjunto de estados

margem: margem em pixels

cor: RGB

linha: espessura de linha (0 - preencher)

Nota:

Os tipos *Estado*, *Operador* e *No*, são especificados nos respectivos documentos de apoio ao projecto.