Conception détaillée

Projet L3D2 - Systèmes proies-prédateurs

Année 2022-2023

***Les informations d’identification du document***

| Référence du document: | CD\_L3D2\_v1 |
| --- | --- |
| Version du document: | 1 |
| Date du document: | 12/02/2023 |
| Auteurs: | Fu Suntanqing  Rossignol Paul  Xu Remy  Ye Frédéric |

***Les éléments de vérification du document***

| En attente de validation: | LATTAUD Claude |
| --- | --- |
| Soumis le: | 15/02/23 |
| Validé le: | \_\_/\_\_/23 |
| Type de diffusion: | Document électronique (.pdf) |
| Confidentialité: | Réservé au jury, professeurs encadrants et aux étudiants de l’Université Paris Cité. |

**SOMMAIRE**

[**I. Introduction**](#_1s4ci3dztqq) **4**

**II.Liste des classes implémentées ou modifiées 4**

1-Agent 4

2-Prise de contrôle 7

3-Mimétisme 7

4-Interaction avec l’eau 7

5-Monde 8

6-Météo possible 9

7-Météo 9

## **I.Introduction :**

La conception détaillée présente les classes, attributs et méthodes qui sont nécessaires au bon développement de l’application. En effet, elle permet de faciliter et garantir la bonne implémentation de toutes les composantes de l’application aux développeurs.

## **II.Liste des classes implémentées ou modifiées**

1- Agent

class Agent

Permet de gérer l’ensemble des agents et leurs comportements.

Attribut :

regimea : String[]

Liste de l’ensemble des proies possible de l’agent.

masse : Double

ter : Terrain

Désigne sur quelle type de terrain se trouve l’agent.

agent : NavMeshAgent

Désigne le type de l’agent (loup,tigre,...).

vitessechasser : Float

portee : Integer

besoinenerg : Double

apportenerg : Double

age : Integer

coordx: Double

coordy : Double

vitesse : Double

vitessemax : Double

vitessemin : Double

frequencereproductionmin : Double

frequencereproductionmax : Double

sexe : Boolean

veutsereproduire : Boolean

estenceinte : Boolean

tempsgrossesse : Double

agematuration : Integer

estadulte : Boolean

agematurationmin : Double

agematurationmax : Double

tempsdisgestion : Integer

tempsconsommationproie : Integer

envie : Boolean

enfuite : Boolean

tempsrestantdigestion : Double

endurance : Integer

endurancemax : Integer

causedeces : String

nomespece : String

peur : Integer

soif : Integer

faim : Integer

modele : 3DModel

La représentation de l’agent dans la simulation.

id : Integer

agentencontrole : Boolean

Si agentEnControle est true l’agent est sous le contrôle de l’utilisateur, sinon il ne l’est pas.

champsdevision : Integer

canmove : Boolean

actionencours : String

Fonction :

miseAJour () : Void

Mise à jour des coordonnées de l’agent.

affectation () : Void

deplacement () : Void

Permet à l’agent de se déplacer d’un endroit à un autre.

chercher() : Void

Permet à l’agent de se déplacer afin d’avoir une interaction avec le monde.

chasser() : AbstractVoid

Permet à l’agent de chasser une proie.

eviterObstacle() : AbstractVoid

Permet à l’agent d’éviter un obstacle.

naissanceReproduction() : Void

Permet à l’agent de créer un agent.

reproduction() : Void

Permet à l’agent d’avoir de se reproduire.

manger(int Masse) : Void

Permet à l’agent de manger.

getNomEspece() : String

Permet de récupérer le nom de l’espèce.

getRegime() : String[]

Permet de récupérer la liste du régime alimentaire de l’agent.

setRegime(String[]) : Void

Permet de modifier la liste du régime alimentaire de l’agent.

accelerer() : Void

Permet à l’agent d'accélérer.

tetaniser() : Void

Empêche l’agent de se déplacer à cause de la peur.

fuite() : Void

Permet à l’agent d’effectuer l’action de fuite.

setModele(3DModel modele) : Void

Permet de modifier le modèle 3D d’un agent, exemple si un loup meurt de faim le modèle 3D pourra être modifier.

getBesoin() : Integer[]

retourne la liste des besoins de l’agent(faim,,soif,peur,..) .

setBesoin(Integer [] tabBesoin) : Void

permet de modifier avec une liste les besoins de l’agent(faim,,soif,peur,..) .

switchControle() : Void

Changer le boolean agentencontrole.

switchCanMove() : Void

Change le boolean canmove.

setChampdsDeVision(Integer )

Permet de définir le champ de vision d’un agent afin de détecter les obstacles et les autres agents.

setAction(String s) : void

Permet de modifier l’action en cours de l’agent.

getAction() : String

Permet de récupérer l’action en cours de l’agent.

2- Prise de contrôle

class Controle extends Agent

Permet la prise de contrôle d’un agent

Fonction :

* public void deplacement() :

permettra à l’utilisateur de se déplacer en fonction de ses actions sur le clavier s’il ne rentre pas en conflit avec un objet de l’environnement.

* public void acceleration() :

permettra à l’utilisateur lorsque la touche “MAJ” sera prise entrée de passer en mode accélération.

* public void interact () :

permettra à l’utilisateur lorsque la touche “E” sera prise en entrée d’effectuer une action en fonction de l’objet devant l’agent.

3- Mimétisme

class Mimic

Permet de gérer le mimétisme des agents.

Fonction :

detectIfAgentIsTakeOver() : Void

Permet aux agents contrôlés par une intelligence artificielle de repérer un agent contrôlé par l’utilisateur.

detectActionOfAgent() : Void

Permet aux agents contrôlés par une intelligence artificielle d’observer les actions d’un agent contrôlé par l’utilisateur.

mimicAction(): Void

Permet aux agents contrôlés par une intelligence artificielle de reproduire les actions effectuées par un agent contrôlé par l’utilisateur si elles permettent de satisfaire leurs besoins.

saveAction() : Void

Sauvegarde l’action dans la base de données dédiée.

4- Interaction avec l’eau

class WaterInteraction extends Agent

Attribut :

canswim : boolean

Fonction :

swim():

permet à l’utilisateur de nager, s’il le peut.

boire() : Void

permet à l’utilisateur de boire.

5 - Monde

class World

Permet la gestion de la météo et de son impact sur le monde

Attribut :

ter : Terrain

Indique le terrain utilisé pour le monde.

temperature : Integer

La température du monde.

tauxhumidite : Integer

Le taux d’humidité du monde.

vitessemoyvent : Integer

La vitesse moyenne du vent.

saison : String

La saison actuelle du monde.

meteo : Meteo

La météo actuelle du monde.

Fonction :

getTemperature() : Integer

Permet de récupérer la température.

getTauxHumidite() : Integer

Permet de récupérer le taux d’humidité.

getVitesseMoyVent() : Integer

Permet de récupérer la vitesse moyenne du vent.

getSaison() : String

Permet de récupérer la saison en cours.

getMeteo() : Meteo meteo

Permet de récupérer la météo en cours

setTemperature(Integer temprature) : Void

Permet de modifier la température.

setTauxHumidite(Integer tauxHumidite) : Void

Permet de modifier le taux d’humidité.

setSaison(String saison) : Void

Permet de modifier la saison.

setMeteo (String saison,Meteo meteo) : Void

Permet de modifier la météo.

6 - Météo possible

Énumère l’ensemble des météos possibles.

enemueration PossibiliteMeteo

ORAGE

SECHERESSE

PLUIE

TEMPS\_CLAIR

BROUILLARD

VENT

7 - Météo

class Meteo extend PossibiliteMeteo

Permet de modifier les différents attributs en fonction de la météo actuelle.

Attribut :

meteo : PossibiliteMeteo

variationtemperature double[]

tauxhumidite : Double

puissanceffet : Double

influenceSoif : Double

influenceReproductionPlante : Double

influenceChampsVision : Double

Fonction :

variationTemperature(Double temperature) : Void

Permet de définir les températures possibles avec avec borne dans un tableau.

drawVisualEffects (Météo m) : Void

Permet d’afficher dans le monde un effet visuel pour chaque meteo possible.

modifInfluenceSoif(Double d) : Void

Permet de modifier l’influence de la météo sur le besoin de soif des agents.

modifInfluenceReproductionPlante(Double d):Void

Permet de modifier l'influence de la météo sur la reproduction des agents autotrophes.

modifInlfuenceChampsDeVision(Double): Void

Permet de modifier le champ de vision des agents en fonction de la météo.

III - Packages :

L’ensemble du programme est contenue dans un dossier asset contenant 12 dossiers. Les dossiers, “CodeMonkey”, “TerrainData”, “TextMeshPro”, “Shaders”, “HDRPDefaultRessources” et “NavMeshComponents” sont des packages créaient automatiquement par unity afin divers outils de programmation.

Le dossier “Animals” contient l’ensemble des modèles 3D, animations et l'habillage de l’ensemble des agents de la base de données.

Le dossier