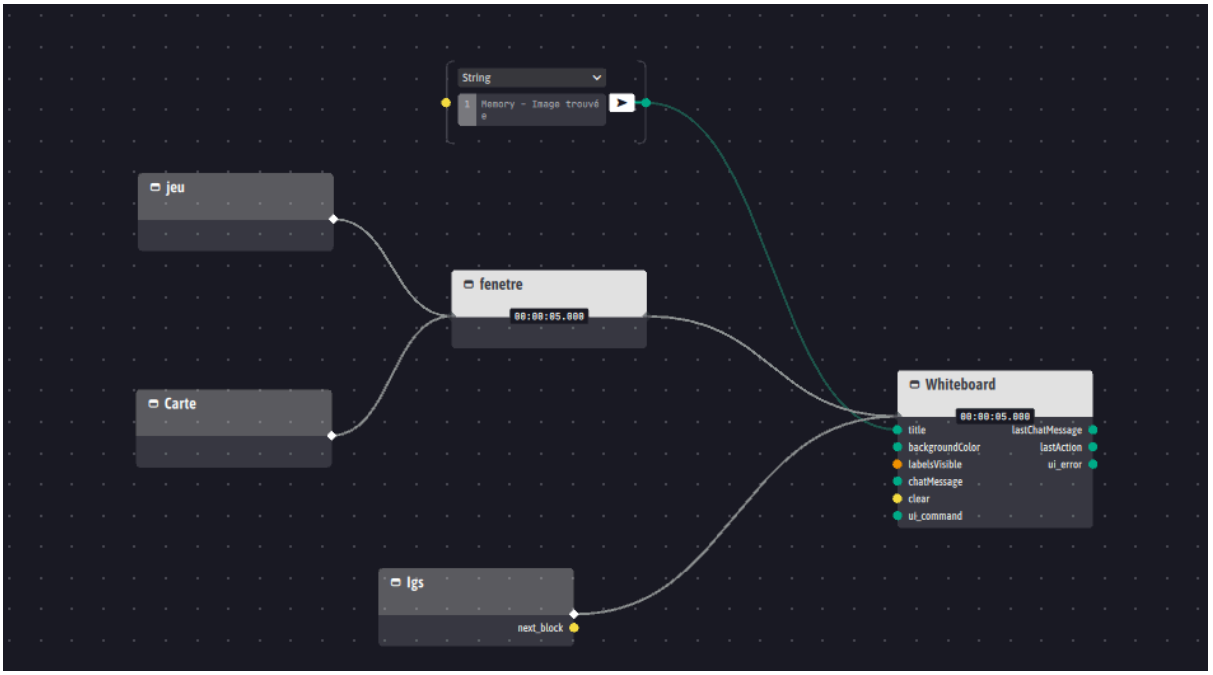
Rapport du Projet de Jeu Memory sur IngenuityScape

Introduction

Ce rapport présente le développement d'un jeu Memory réalisé à l'aide du logiciel IngenuityScape, en collaboration avec un partenaire. Le jeu a été conçu pour fonctionner sur un tableau blanc interactif fourni par l'outil, et utilise une interface utilisateur graphique (GUI) basée sur **Tkinter**. Le projet implique quatre agents principaux : `Whiteboard`, `Jeu`, `Fenetre`, et `Carte`. Chacun de ces agents remplit un rôle spécifique dans le fonctionnement global du jeu, depuis l'affichage des cartes jusqu'à la gestion de la logique de jeu.

Structure des Agents

Voici un aperçu des quatre agents impliqués et de leurs fonctions :



1. Agent `Fenetre`

Cet agent est responsable de la création et de la gestion de la fenêtre principale de l'application. Il assure également le nettoyage de l'interface graphique entre chaque partie.

- **Fichiers principaux** :
    - `fenetre.py`
    - `main.py`

- **Fonctionnalités clés** :
    - **Création de la fenêtre** : L'agent `Fenetre` initialise la fenêtre de jeu avec les dimensions et couleurs spécifiées. Cela inclut la configuration des widgets tels que les boutons et les labels pour interagir avec l'utilisateur.
    - **Nettoyage de la fenêtre** : Avant chaque nouvelle partie, l'agent nettoie la fenêtre en supprimant les éléments de la partie précédente.
- **Code important** :
    - `creer_fenetre(fenetre, taille, titre, couleur)` : Configure la fenêtre principale.
    - `nettoie_fenetre(fenetre, can, liste_widgets)` : Supprime les éléments graphiques avant de commencer une nouvelle partie.

2. Agent `Carte`

Cet agent gère les images des cartes du jeu Memory. Il permet la création et la gestion des images, qui sont utilisées pour représenter les cartes à l'écran.

- **Fichier principal** :
    - `carte.py`
- **Fonctionnalités clés** :
    - **Création d'images** : Les images sont chargées depuis des fichiers, et un objet `PhotoImage` est créé pour chaque image.
    - **Gestion de la taille des images** : Méthode pour déterminer les dimensions des images.
- **Code important** :
    - `creer_item(tim)` : Crée un objet image basé sur le nom de fichier fourni.
    - `liste_image(nombre)` : Génère une liste d'images utilisées dans le jeu.

3. Agent `Jeu`

Cet agent est le cœur de la logique du jeu. Il gère la mécanique du jeu Memory, y compris la vérification des paires de cartes et le calcul des scores.

- **Fichier principal** :
    - `jeu.py`

- **Fonctionnalités clés** :
    - **Mélange des cartes** : Mélange la liste des cartes avant chaque partie pour garantir l'unicité du jeu.
    - **Gestion des clics** : Détecte les clics sur les cartes, vérifie les paires et met à jour l'interface graphique.
    - **Calcul des scores** : Suivi du nombre de paires trouvées et du score du joueur.
- **Code important** :
    - `melange(l_nombre)` : Mélange aléatoirement la liste des cartes.
    - `verifie_memoire(...)` : Vérifie si une paire correspond et met à jour l'affichage en conséquence.

4. Agent `Whiteboard`

Dans le jeu Memory, on a choisi de créer une interface spéciale car il n'y avait pas de moyen d'interagir avec le tableau blanc via la souris. Ainsi, lorsque des paires d'images sont trouvées, elles sont affichées sur le tableau blanc. Ce choix a été fait car il était plus simple d'avoir une interface de jeu dédiée pour gérer les interactions, tout en affichant les résultats sur le tableau blanc.

Fonctionnement Général

Le jeu commence par le lancement de l'agent `Fenetre`, qui crée l'interface utilisateur avec un tableau de cartes cachées. Chaque carte est associée à une image spécifique, et les cartes sont mélangées pour chaque partie. L'agent `Jeu` gère les interactions de l'utilisateur, détecte les clics sur les cartes, et vérifie si une paire a été trouvée. Lorsque l'utilisateur découvre une paire correcte, l'image est révélée sur le tableau. L'agent `Carte` fournit les images nécessaires pour les cartes, et `Whiteboard` affiche le tout sur l'interface d'IngenuityScape.

Déroulement d'une Partie

1. **Initialisation** : La fenêtre est configurée, les images sont chargées, et les cartes sont mélangées.
2. **Début du Jeu** : Le joueur clique sur une carte pour révéler son image. Si deux cartes identiques sont cliquées consécutivement, elles restent visibles.
3. **Fin de Partie** : Le jeu se termine lorsque toutes les paires sont trouvées. Le score est affiché, et une option pour recommencer une nouvelle partie est proposée.

Technologies Utilisées

- **Python** : Langage principal pour la logique du jeu et l'interface graphique.
- **Tkinter** : Librairie utilisée pour l'interface graphique, permettant de créer des boutons, labels, et la gestion du canvas.
- **Ingescape** : Utilisé pour l'intégration avec le tableau blanc interactif et pour la communication entre agents.

Lancement du Jeu

Pour démarrer le jeu, assurez-vous que **Ingescape Circle v4** et le **whiteboard** sont bien lancés et connectés au réseau Wi-Fi sur le port 2007. Une fois cette étape terminée, il suffit d'exécuter le fichier `Jeu_Memory_start.bat` pour lancer le jeu de memory.

Ce fichier `.bat` contient la ligne de commande nécessaire pour initialiser le jeu de manière automatique, simplifiant ainsi le processus de démarrage en un seul clic.

Conclusion

Le projet est une implémentation réussie d'un jeu Memory utilisant un environnement distribué et interactif. Les agents ont été conçus pour fonctionner de manière modulaire, ce qui permet une gestion claire et efficace de chaque composant du jeu. Le code est fonctionnel.

Le système pourrait être amélioré en ajoutant des fonctionnalités supplémentaires, comme un mode multijoueur ou des niveaux de difficulté.

V&V Script pour le Jeu Memory

1. Vérification de l'Interface Utilisateur

- L'interface de jeu s'affiche correctement au lancement.
- Chaque carte est visible et cliquable.
- Les interactions sont enregistrées (clic sur une carte retourne l'image).

2. Vérification des Fonctions

- Lorsque deux cartes identiques sont sélectionnées, elles sont marquées comme trouvées.
- Si une paire est trouvée, les images sont affichées sur le tableau blanc.
- L'interface de jeu se met à jour après chaque interaction.

3. Validation du Comportement du Jeu

- L'utilisateur peut jouer sans erreurs et suivre le jeu via le tableau blanc.
- Les paires d'images trouvées restent visibles sur le tableau blanc.
- Le jeu est jouable jusqu'à ce que toutes les paires soient trouvées.