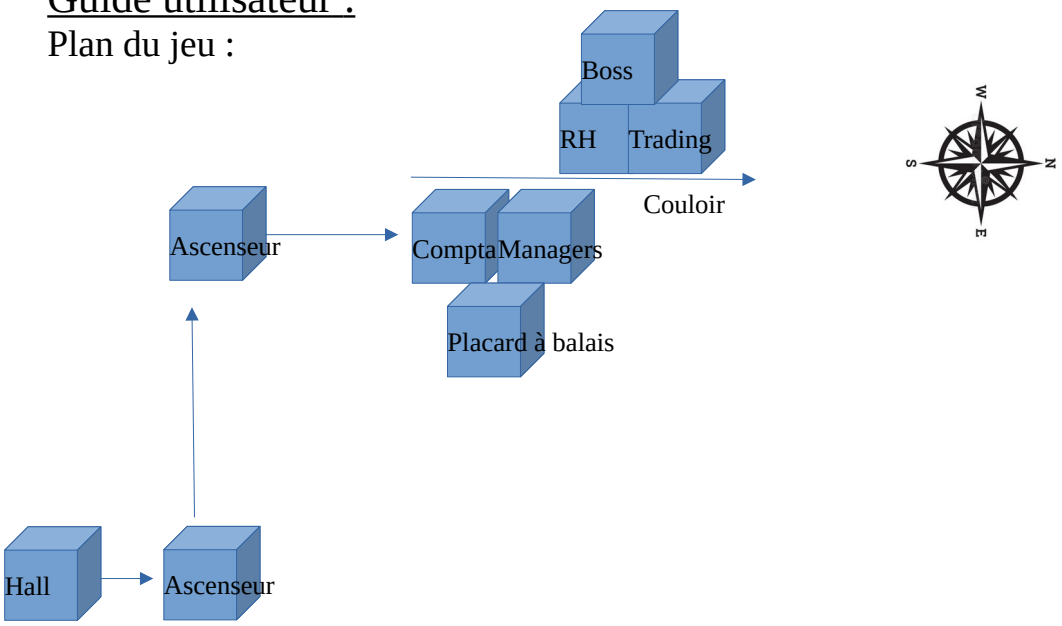
Rapport Projet Text Based Aventure en Python

SAAD-EDDINE Amine & DELEHELLE Maxence

Guide utilisateur :

Plan du jeu :



Attention : Nous avons volontairement placé le Nord dans le sens de direction du personnage. Cela peut paraître contre-intuitif au début, mais c'est un choix de notre part pour une meilleure visualisation des mouvements.
De plus , vous remarquerez que certaines portes sont verrouillées. Une fois deverouillées, il faut retaper la commande de direction pour rentrer dans la salle.
Exemple :
>>>go O
>>>La porte est verrouillée. Entrez le code :
>>>Code : 1234
>>>Code correct. La porte est déverrouillée (Vous n'êtes pas dans la pièce)
>>>go O (Vous êtes dans la pièce que vous venez de déverrouiller)

Contexte : Vous êtes un reporter infiltré dans une multinationale qui doit dérober des documents confidentiels prouvant l'assassinat d'un journaliste qui s'est intéressé d'un peu trop près à une histoire de gisement de pétrole. Vous débutez le jeu dans le hall d'accueil et n'avez d'autre direction possible que l'ascenseur. L'ascenseur vous emmène au 52ème étage. Une fois arrivez vous débouchez dans un long couloir qui se divise en deux parties. Dans la première partie, à votre gauche le bureau des RH, à votre droite le bureau de comptabilité. Dans la deuxième partie, à votre gauche, le bureau de trading, à votre droite le bureau des managers. Au 53ème étage, le bureau du patron qui n'est accessible que via le bureau de trading ou le bureau des RH. Au 51ème étage, le placard à balais qui n'est accessible que via le bureau des managers ou le bureau de comptabilité.

Condition de victoire : Vous arrivez à dérober les fameux documents et vous échapper.

Condition de défaite : Vous dérobez les documents mais repassez par le bureau de comptabilité. La salle est bourrée de caméras de surveillance et vous vous faites repérer.

Solution pour gagner : Le bureau de trading est verrouillé par un code. Ce code est le chiffre d'affaire de l'année 2024 de la société. On trouve ce nombre en allant sur l'ordinateur dans le bureau de comptabilité et en effectuant la commande « enter Ordinateur ». Une page s'affiche et le chiffre d'affaire de la dernière année est 2549 millions d'euros. Ainsi, le code est 2549. Une fois déverrouillé, rentrer dans le bureau de trading et prendre la clé qui déverrouillera la porte du boss. Rentrer dans le bureau du boss et prendre les Documents. S 'échapper sans repasser par le bureau de comptabilité.

Guide développeur

Classe Actions : Elle regroupe les fonctions permettant au joueur d'interagir avec le jeu.,On dispose des fonctions go(), qui(), help(), back(), look(), check(), take(), drop(), enter().
Enter() utilise l'interface graphique Tkinter.

Paramètres (ils sont les mêmes pour toutes les fonctions) :
game : Game
lis_of_words : list[str]
number_of_parameters : int

Classe Character : Elle définit les caractéristiques des personnages non joueurs.

Atrributs : name : str
description : str
current_room : Room
msg : list[str]

Méthodes : init(self,name, description, current_room, msg) → None
str(self) → str
move(self) → None. Fais bouger les pnj demanière aléatoire
get_msg(self) → str

Classe Command : Associe une fonction de la classe Actions à une chaine de caractère

Attributs : command_str : str
help_string : str

Méthodes : __init__(self, command_word, help_string)
action : Action
number_of_parameters : int

Classe Item : Elle définit les caractéristiques des objets présents dans les pièces qui peuvent être pris par le joueur.

Attributs : name : str
description : str
weight : int

Méthodes : __init__(self, ,name, description, weight) → None
__str__(self) → str

Classe Player : Elle décrit le joueur

Attributs : name : str
inventory : dict[str, Items]
history : list[Rooms]
current_room : Room
hall : Room
last_room : Room
win : bool
max_weight : int

Méthodes : __init(self, name, hall) → None
move(self, direction) → bool
get_history(self) → str

get_inventory(self) → str

Classe Room : Elle définit et décrit les lieux du jeu.

Attributs : name : str
description : str
inventory : set(Items)
exits : dict[str, Room]
character : dict[str, Character]
locked : bool
code : str

Méthodes : __init(self, name, description, locked) → None
get_exit(self, direction) → Room | None
get_exit_string(self) → str
get_long_description(self) → str
get_inventory(self) → str

Classe Game : Elle définit la classe principale du jeu. Les items, personnages, salles sont définies dans la méthode setup(). La méthode play() implémente le mécanisme itératif du jeu tour par tour. La méthode process_command() convertie les chaînes de caractère entrée au clavier en objet Python qu'il peut exploiter.

Attributs : finished : bool
rooms : list[Rooms]
commands : dict[str, Command]
player : Player

Méthodes : __init__(self) → None
setup(self) → None
play(self) → None
process_command(self, command_string) → None
print_welcome(self) → None
win(self) → bool
loose(self) → bool

Perspectives de développement

Notre jeu ne comporte que très peu d'interface graphique (seulement l'écran de l'ordinateur avec le chiffre d'affaire affiché). On aurait pu penser à une map interactive pour aider le joueur à se repérer. Nous aurions aussi pu faire une petite interface graphique simple pour que le joueur rentre le code de la salle de trading plutôt que sur le clavier.

La complexité du jeu pourrait être augmenté avec des codes et des clés supplémentaires à trouver.

Il n'y a qu'un seul scénario perdant. Pour augmenter la difficulté du jeu, une multitude de scénarios perdants pourrait être envisageable.