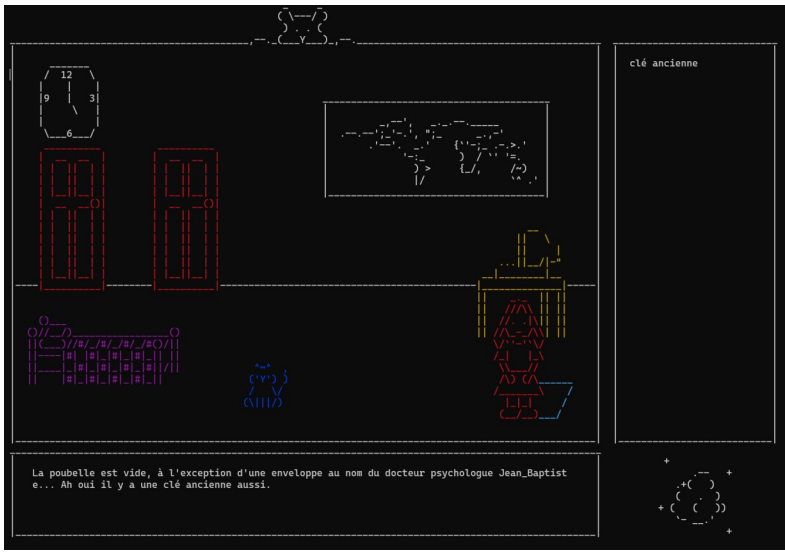
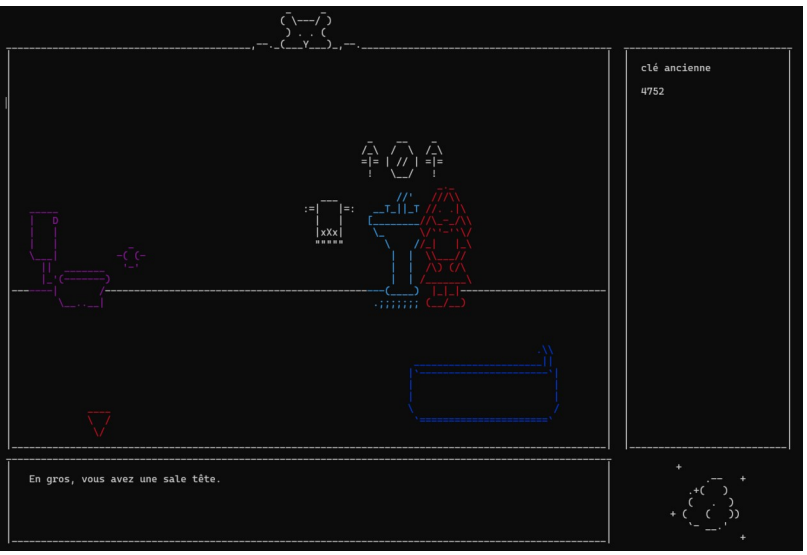
CONCEPTION


Intéragissez avec le lit pour commencer...
z : haut
q : gauche
s : bas
d : droite
e : intéragir
echap : quitter


clé ancienne
La poubelle est vide, à l'exception d'une enveloppe au nom du docteur psychologue Jean_Baptist
e... Ah oui il y a une clé ancienne aussi.


clé ancienne
4752
En gros, vous avez une sale tête.


clé ancienne
4752
clé rouge
clé bleue
Il y a une pinte vide et une pleine de bière sur la table. Vous la buvez. Espérons que les eff
ets n'est aucun impact sur vous. Au fond se trouve une clé bleue.


clé ancienne
4752
clé rouge
clé bleue
bouteilles vides
Vous ouvrez tous les tirroirs, il y a des dizaines de boites d'antidépresseurs et des bouteill
es de bières.


clé ancienne
4752
clé rouge
clé bleue
bouteilles vides
Merci d'avoir joué à mon jeu! J'espère que cette expérience vous aura plus!
Designé et codé par Katell Toullec.

STARLESS NIGHT

Type	**Conception**
Nom du projet	**StarlessNight**
Commentaire	**Exemple illustratif cours IPI, S2A, ENIB**
Auteur	**Katell TOULLEC**
Version	**1.0**
Date	**10/11/2023**

Table des matières

1) Rappel du cahier des charges

1.1) Contraintes techniques

- Le logiciel est associé à un cours, il doit donc fonctionner sur les machines de TP de l'ENIB pour que les élèves puissent le tester.
- Le langage utilisé en cours est Python. Le développement devra donc se faire en python(et plus précisément en python3).
- Les notions de programmation orientée objet n'ayant pas encore été abordées, le programme devra essentiellement s'appuyer sur le paradigme de la programmation procédurale.
- Le logiciel devra être réalisé en conformité avec les pratiques préconisées en cours de IPI : barrière d'abstraction, modularité, unicode, etc...
- L'interface sera réalisée en mode texte dans un terminal

1.2) Fonctionnalités

- F1 : Commencer le jeu
  - F1.1 Afficher l'écran de début
  - F1.2 : interagir avec l'item de départ et appuyer sur e pour commencer
- F2 : jouer au jeu
  - F2.1 Afficher le jeu:
    - cadre de l'inventaire
    - rectangle de dialogue
    - pièce
  - F2.2 déplacer le personnage dans un cadre et dans l'espace, vers la gauche, droite, vers le haut ou le bas
  - F2.3 Interagir avec les objets à proximité
  - F2.4 Changer de pièce en passant par les portes
  - F2.5 changer de pièce en allant sur les flèches de retour en arrière
  - F2.6 afficher les dialogues dans le rectangle prévu à cet effet
  - F2.7 effacer les dialogues en bougeant l'avatar
  - F2.8 ramasser des objets et les ajouter dans l'inventaire
  - F2.9 utiliser les objets de notre inventaire sur d'autres objets
  - F2.10 saisir un code dans la boite de dialogue
  - F2.11 indiquer que l'utilisateur peut interagir avec l'objet

- F3 : finir le jeu
  - F3.1 Afficher l'écran de fin de jeu
  - F3.2 Quitter

2) Principes des solutions techniques

2.1) Langage
Conformément aux contraintes énoncées dans le cahier des charges, le codage est réalisé avec langage python. Nous choisissons la version 3.

2.2) Architecture du logiciel
Nous mettons en œuvre le principe de la barrière d'abstraction. Chaque module correspond à un type de donnée et fournit toutes les opérations permettant de le manipuler de manière abstraite.

2.3) Interface utilisateur

L'interface utilisateur se fera via un terminal de type linux.
Nous reprenons la solution donnée en cours de MDD en utilisant les modules : `termios, sys, select`.

2.4) Boucle de simulation
Le programme mettra en œuvre une boucle de simulation qui gérera l'affichage et les événements clavier.

2.5) Affichage
L'affichage se fait en communicant directement avec le terminal en envoyant des chaînes de caractères sur la sortie standard de l'application.

2.6) Gestion du clavier
L'entrée standard est utilisé pour détecter les actions de l'utilisateur.
Le module `tty` permet de rediriger les événements clavier sur l'entrée standard.
Pour connaître les actions de l'utilisateur il suffit de lire l'entrée standard.

2.7)Image ascii-art
Pour dessiner certaines parties de l'interface nous utilisons des « images ascii ».
Dans l'idée de séparer le code et les données, les différentes images ASCII seront stockées dans des fichiers textes : `arrow.txt, background.txt, bath.txt, bathroom.txt, bed.txt, bedroom.txt, bench.txt, buffet.txt,cat.txt, desk.txt, door.txt, end.txt, fin.txt, girl.txt, lamp.txt, library.txt, livingroom.txt, nightstand.txt, secretroom.txt, sink.txt, sofa.txt, start.txt, table.txt, television.txt, text.txt toilet.txt, trash.txt`

2.8) Pièces, objets…

Pour modéliser les différentes pièces et objets du jeu, on utilisera les fichiers suivants :

- éléments d'affichage du jeu : `background.txt`
- pièces : start.txt, `bathroom.txt, bedroom.txt, livingroom.txt, secretroom.txt, end.txt`
- meubles et objets : `arrow.txt, bath.txt, bed.txt, bench.txt, buffet.txt, cat.txt, desk.txt, door.txt, fin.txt, lamp.txt, library.txt, mom.txt nightstand.txt, sink.txt, sofa.txt, table.txt, television.txt, text.txt, text2.txt, text3.txt toilet.txt, trash.txt`
- personnage : `girl.txt`

3) Analyse

3.1) Analyse noms/verbes :

- Verbes :

  Afficher, interagir, appuyer, commencer, jouer, afficher, déplacer, changer, effacer, ramasser, utiliser, saisir, indiquer, finir, quitter

- Nom :

  écran du début, item, jeu, inventaire, dialogue, pièce, personnage, espace, objets, portes, flèches, code, écran de fin

3.2) Types de donnée

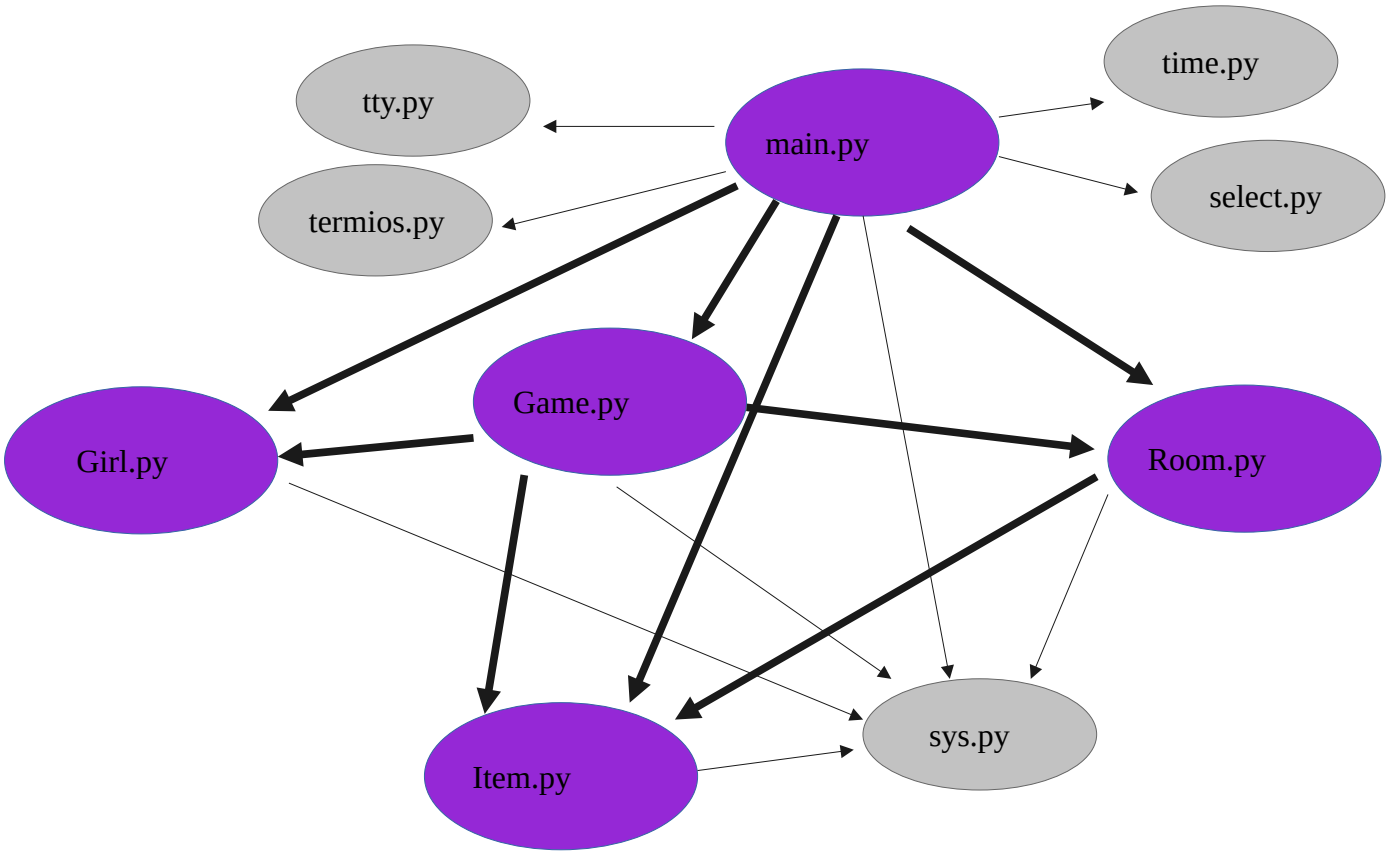
```
Type : Game = struct
          room_index : int
          inventary  : liste de chaîne de caractères
          background : chaîne de caractères
          rooms      : liste de Room
          room_position : (int,int)
          inventary_position : (int,int)
          dialogue_position : (int,int)
          deplacement_zone : (int,int)
     fstruct

type : Room = struct
          filename       : chaîne de caractères
       fstruct

type : Item = struct
          filename       : chaîne de caractères
          position   : (int, int)
          dialogue       : chaîne de caractères
          access     : int
          activated  : bool
          key        : chaîne de caractères
          activated_dialogue : chaîne de caractères
          loot       : chaîne de caractères
       fstruct

type: Girl = struct
          position     : (int, int)
          color        : chaîne de caractères
          image        : chaîne de caractères
     fstruct
```

3.3) Dépendance entre modules



3.4) Analyse descendante :

3.4.1)Arbre principal :

```
Main.main()
 +-- Main.init()
 |    +-- Game.create()
 |
 +-- Main.show()
 +-- Main.interact()
```

3.4.2)Arbre affichage

```
Main.show()
     +-- Game.show()
```

3.4.3)Arbre interaction

```
Main.interact()
     +-- Game.move()
     +-- Game.start()
     +-- Game.set_items_dialogue()
     +-- Game.set_items_activated()
     +-- Game.change_room()
     +-- Game.get_interactions()
     +-- Game.end()
```

4) Description des fonctions

4.1) Programme Principal : main.py

- Main.**main()**
- Main.**init()**
- Main.**run()**
- Main.**show()**
- Main.**interact()**
- Main.**isData()**

Main.**main()**->rien
Description : fonction principale du jeu
Paramètres : data
Valeur de retour :aucune

Main.**init()**->rien
Description : initialisation du jeu
Paramètres : data
Valeur de retour :aucune

Main.**run()**->rien
Description : boucle de simulation
Paramètres : data
Valeur de retour :aucune

Main.**show()**->rien
Description : affichage du jeu
Paramètres : data
Valeur de retour :aucune

Main.**interact()**->rien
Description : gère les événements clavier
Paramètres : data
Valeur de retour :aucune

Main.**isData()**→rien
Description : récupère les évènements clavier
Paramètres : aucun
Valeur de retour : aucune

4.2) Game.py

- Game.**create()**
- Game.**set_room_index()**
- Game.**get_room_index()**
- Game.**set_inventary()**
- Game.**get_inventary()**
- Game.**set_background()**
- Game.**get_background()**
- Game.**set_rooms()**
- Game.**get_rooms()**
- Game.**set_inventary_position()**
- Game.**get_inventary_position()**
- Game.**set_rooms_position()**
- Game.**get_rooms_position()**
- Game.**set_dialogue_position()**
- Game.**get_dialogue_position()**
- Game.**set_deplacement_zone_x()**
- Game.**get_deplacement_zone_x()**
- Game.**set_deplacement_zone_y()**
- Game.**get_deplacement_zone_y()**
- Game.**add_to_inventary()**
- Game.**move()**
- Game.**set_items_dialogue()**
- Game.**start()**
- Game.**get_interactions()**
- Game.**change_room()**
- Game.**reset_items_dialogue()**
- Game.**set_items_activated()**
- Game.**end()**
- Game.**show()**

Game.**create()**
Description: créé le type game
Paramètres: room_index, inventary, background, rooms,
room_position, inventary_position,
dialogue_position, deplacement_zone
Valeur de retour: game

Game.**set_room_index()**
description : definit l'index de la piece
paramètres : game, room_index
valeur de retour : rien

Game.**get_room_index()**
description : recuperer la valeur de l'index de la room
paramètres : game
valeur de retour : room_index

Game.**set_inventary()**

description : definir l’inventaire
paramètres : game, inventary
valeur de retour : rien

Game.**get_room_index()**
description : permets de recuperer la liste de l’inventaire
paramètres : game
valeur de retour : inventary

Game.**set_background()**
description : definit le background
paramètres : game, background
valeur de retour : rien

Game.**get_background()**
description : permet de recuperer le background
paramètres : game
valeur de retour : background

Game.**set_rooms()**
description : definit la liste des rooms
paramètres : game, rooms
valeur de retour : rien

Game.**get_rooms()**
description : permet de recuperer la liste des rooms
paramètres : game
valeur de retour : rooms

Game.**set_inventary_position()**
description : definit la position de l’affichage de
l’inventaire dans le terminal
paramètres : game, inventary_position
valeur de retour : rien

Game.**get_inventary_position()**
description : permet de recuperer la position de l’affichage
de l’inventaire dans le terminal
paramètres : game
valeur de retour : inventary_position

Game.**set_rooms_position()**
description : definit la position de l’affichage de
la pièce dans le terminal
paramètres : game, rooms_position
valeur de retour : rien

Game.**get_rooms_position()**
description : permet de recuperer la position de l’affichage
de la pièce dans le terminal
paramètres : game

valeur de retour : rooms_position

Game.**set_dialogue_position()**
description : definit la position de l'affichage du dialogue dans le terminal
paramètres : game, dialogue_position
valeur de retour : rien

Game.**get_dialogue_position()**
description : permet de recuperer la position de l'affichage du dialogue dans le terminal
paramètres : game
valeur de retour : dialogue_position

Game.**set_deplacement_zone_x()**
description : definit la position des pixels sur l'axe x où nous pouvons deplacer l'avatar
paramètres : game, deplacement_zone_x
valeur de retour : rien

Game.**get_deplacement_zone_x()**
description : permet de recuperer la zone de deplacement possible en x
paramètres : game
valeur de retour : deplacement_zone_x

Game.**set_deplacement_zone_y()**
description : definit la position des pixels sur l'axe y où nous pouvons deplacer l'avatar
paramètres : game, deplacement_zone_y
valeur de retour : rien

Game.**get_deplacement_zone_y()**
description : permet de recuperer la zone de deplacement possible en y
paramètres : game
valeur de retour : deplacement_zone_y

Game.**add_to_inventary()**
description : permet d'ajouter un item à la liste d'item de l'inventaire
paramètres : game, item
valeur de retour : rien

Game.**move()**
description : permet de déplacer l'avatar de la girl dans le terminal et en respectant les collisions
paramètres : game, direction
valeur de retour : rien

Game.**start()**

description : permet de gerer l’affichage de l’écran de départ
paramètres : game
valeur de retour : rien

Game.**get_interactions()**
description : permet de gérer les interactions sur des objets spécifiques
paramètres : game
valeur de retour : rien

Game.**change_room()**
description : permet de changer de pièce en passant par des portes ou des flèches
paramètres : game
valeur de retour : rien

Game.**reset_items_dialogue()**
description : met le dialogue à la valeur d’une chaîne de caractère vide
paramètres : game
valeur de retour : rien

Game.**set_items_activated()**
description : permet de gérer l’activation des items
paramètres : game
valeur de retour : rien

Game.**end()**
description : permet de gérer l’affichage de l’écran de fin
paramètres : game
valeur de retour : rien

Game.**show()**-> rien
Description: affiche l’écran de jeu, les cadres des dialogues et de l’inventaire, les différents dialogues et les objets dans l’inventaire, il affiche aussi l’avatar
Paramètres: game
Valeur de retour : rien

4.3) Girl.py

- Girl.**create()**
- Girl.**set_position()**
- Girl.**get_position()**
- Girl.**show()**

Girl.**create()**->girl
Description: créé la fonction de l'avatar de la girl
Paramètres: color, position, image
Valeur de retour: girl

Girl.**set_position(p)**→girl
Description: permet de changer la position de l'avatar
Paramètres: girl, position
Valeur de retour : rien

Girl.**get_position(p)**→girl
Description: permet de récupérer la position de l'avatar
Paramètres: girl
Valeur de retour : position

Girl.**show(p)**-> girl
Description: affiche l'avatar
Paramètres: girl
Valeur de retour : rien

4.4) Room.py

- Room.**create()**
- Room.**get_room_items()**
- Room.**add_item()**
- Room.**show()**

Room.**create()**
Description: créé le type Room
Paramètres: filename
Valeur de retour: room

Room.**get_room_items()**
Description: renvoie la liste des items d'une pièce donnée
Paramètres: room
Valeur de retour: items

Room.**add_items()**
Description: ajoute un item à la liste des items d'une pièce
Paramètres: room, item
Valeur de retour: rien

Room.**show()**
Description: affiche la pièce demandée dans le terminal avec ses items à leurs positions
Paramètres: room
Valeur de retour : rien

4.5) Item.py

- Item.**create()**
- Item.**set_name()**
- Item.**get_name()**
- Item.**set_position()**
- Item.**get_position()**
- Item.**set_dialogue()**
- Item.g**et_dialogue()**
- Item.**set_access()**
- Item.g**et_access()**
- Item.**set_activated()**
- Item.g**et_activated()**
- Item.**set_key()**
- Item.**get_key()**
- Item.**set_activated_dialogue()**
- Item.**get_activated_dialogue()**
- Item.**set_loot()**
- Item.**get_loot**
- Item.**set_zone()**
- Item.**get_zone()**
- Item.**get_current_dialogue()**
- Item.**test_collision()**
- Item.**test_shift()**
- Item.**show()**

Item.**create()**->item
Description: créé la fonction de l'item
Paramètres: filename, position, dialogue, access, activated,
key, activated_dialogue, loot
Valeur de retour: item

Item.**set_name()**→item
Description: permet de changer le nom de l'item
Paramètres: item, name
Valeur de retour : rien

Item.**get_name()**→item
Description: permet de récupérer le nom de l'item
Paramètres: item
Valeur de retour : name

Item.**set_position()**→item
Description: permet de changer la position de l'item
Paramètres: item, position
Valeur de retour : rien

Item.**get_position()**→item
Description: permet de récupérer la position de l'item
Paramètres: item
Valeur de retour : position

Item.**set_dialogue()**→item
Description: permet de changer le dialogue d'un item
Paramètres: item, dialogue, activated_dialogue
Valeur de retour : rien

Item.**get_dialogue()**→item
Description: permet de récupérer le dialogue d'un item
Paramètres: item
Valeur de retour : dialogue

Item.**set_access()**→item
Description: permet de changer dans quelle pièce et où est l'objet dans la pièce
Paramètres: item, access
Valeur de retour : rien

Item.**get_access()**→item
Description: permet de récupérer la position de l'item est dans quelle pièce il est
Paramètres: item
Valeur de retour : access

Item.**set_activated()**→item
Description: permet de changer l'état de l'objet, si il est activé ou non
Paramètres: item, activated
Valeur de retour : rien

Item.**get_activated()**→item
Description: permet de récupérer l'état actuel de l'objet, si il est activé ou non
Paramètres: item
Valeur de retour : activated

Item.**set_key()**→item
Description: permet de changer la clé
Paramètres: item, key
Valeur de retour : rien

Item.**get_key()**→item
Description: permet de récupérer la clé
Paramètres: item
Valeur de retour : key

Item.**set_activated_dialogue()**→item
Description: permet de changer le dialogue pour l'etat activé
Paramètres: item, activated_dialogue
Valeur de retour : rien

Item.**get_activated_dialogue()**→item
Description: permet de le dialogue pour l'etat activé
Paramètres: item
Valeur de retour : activated_dialogue

Item.**set_loot()**→item
Description: permet de definir le loot qu'on recupere quand l'item est activé
Paramètres: item, loot
Valeur de retour : rien

Item.**get_loot()**→item
Description: permet de récupérer le loot
Paramètres: item
Valeur de retour : loot

Item.**set_zone()**→item
Description: permet de créer une zone de collision
Paramètres: item, zone
Valeur de retour : rien

Item.**get_zone()**→item
Description: permet de récupérer la zone de collision créée
Paramètres: item
Valeur de retour : zone

Item.**show()**-> item
Description: affiche l'item à sa position dans le terminal
Paramètres: item
Valeur de retour : rien

5) Calendrier et suivi de développement

5.1) fonctions à développer

fonctions	codées	testées	commentaires
Main.**main()**	**13/12**	**13/12**	
Main.**init()**	**13/12**	**13/12**	
Main.**run()**	**13/12**	**13/12**	
Main.**show()**	**13/12**	**13/12**	
Main.**interact()**	**13/12**	**13/12**	
Main.**isData()**	**13/12**	**13/12**	
Game.**create()**	**06/12**	**06/12**	
Game.**set_room_index()**	**13/12**	**13/12**	
Game.**get_room_index()**	**13/12**	**13/12**	
Game.**set_inventary()**	**06/12**	**06/12**	
Game.**get_inventary()**	**06/12**	**06/12**	
Game.**set_background()**	**06/12**	**06/12**	
Game.**get_background()**	**06/12**	**06/12**	
Game.**set_rooms()**	**06/12**	**06/12**	
Game.**get_rooms()**	**06/12**	**06/12**	
Game.**set_inventary_position()**	**06/12**	**06/12**	
Game.**get_inventary_position()**	**06/12**	**06/12**	
Game.**set_rooms_position()**	**06/12**	**06/12**	
Game.**get_rooms_position()**	**06/12**	**06/12**	
Game.**set_dialogue_position()**	**06/12**	**06/12**	
Game.**get_dialogue_position()**	**06/12**	**06/12**	
Game.**set_deplacement_zone_x()**	**13/12**	**13/12**	
Game.**get_deplacement_zone_x()**	**13/12**	**13/12**	
Game.**set_deplacement_zone_y()**	**04/01/24**	**04/01/24**	
Game.**get_deplacement_zone_y()**	**04/01/24**	**04/01/24**	
Game.**add_to_inventary()**	**06/12**	**06/12**	
Game.**move()**	**13/12**	**13/12**	Serait parfait si j'avais réussi à faire une collision entre la fille et les items pour qu'ils soient « physiques » et qu'elles ne puissent pas passer au travers. Cela ne marche que sur

			certains items.
Game.**set_items_dialogue()**	**12/12**	**12/12**	
Game.**start()**	**20/12**	**20/12**	
Game.**get_interactions()**	**12/12**	**07/01/24**	Depuis la dernière mise à jour, j'ai réussi à rendre accessible la porte sous les bonnes conditions.
Game.**change_rooms()**	**12/12**	**12/12**	
Game.**reset_items_dialogues()**	**13/12**	**13/12**	
Game.**set_items_activated()**	**12/12**	**12/12**	Depuis la dernière mise à jour, j'ai réglé le problème concernant un affichage de dialogue à l'état désactivé s'affichant avant celui à l'état activée alors que l'item est déjà activé.
Game.**end()**	**20/12**	**20/12**	
Game.**show()**	**06/12**	**06/12**	
Girl.**create()**	**06/12**	**06/12**	
Girl.**set_position()**	**06/12**	**06/12**	
Girl.**get_position()**	**06/12**	**06/12**	
Girl.**show()**	**06/12**	**06/12**	
Room.**create()**	**06/12**	**06/12**	
Room.**get_room_items()**	**13/12**	**13/12**	
Room.**add_item()**	**20/12**	**20/12**	
Room.**show()**	**06/12**	**06/12**	
Item.**create()**	**29/11**	**29/11**	
Item.**set_name()**	**29/11**	**29/11**	
Item.**get_name()**	**29/11**	**29/11**	
Item.**set_position()**	**29/11**	**29/11**	
Item.**get_position()**	**29/11**	**29/11**	
Item.**set_dialogue()**	**29/11**	**29/11**	
Item.**get_dialogue()**	**29/11**	**29/11**	
Item.**set_access()**	**29/11**	**29/11**	
Item.**get_access()**	**29/11**	**29/11**	
Item.**set_activated()**	**29/11**	**29/11**	
Item.**get_activated()**	**29/11**	**29/11**	
Item.**set_key()**	**29/11**	**29/11**	

Item.**get_key()**	**29/11**	**29/11**	
Item.**set_activated_dialogue()**	**29/11**	**29/11**	
Item.**get_activated_dialogue()**	**29/11**	**29/11**	
Item.**set_loot()**	**06/12**	**06/12**	
Item.**get_loot()**	**06/12**	**06/12**	
Item.**set_zone()**	**13/12**	**13/12**	
Item.**get_zone()**	**13/12**	**13/12**	
Item.**get_current_dialogue()**	**20/12**	**20/12**	
Item.**test_collision()**	**13/12**	**13/12**	
Item.**test_shift()**	**20/12**	**20/12**	Ce bout de code ne fonctionne que sur certains objets, je ne comprends pas pourquoi. Il sert à ce que l'avatar ne traverse pas les objets mais je n'arrive pour l'instant pas à utiliser la fonction dans le module game.
Item.**show()**	**06/12**	**06/12**	

5.2) autres

arrow.txt
background.txt
bath.txt
bathroom.txt
bed.txt
bedroom.txt
bench.txt
buffet.txt
cat.txt
desk.txt
door.txt
end.txt
fin.txt
girl.txt
lamp.txt
library.txt
livingroom.txt
nightstand.txt
mom.txt
secretroom.txt
sink.txt
sofa.txt
start.txt

```
table.txt
television.txt
text.txt
text2.txt
text3.txt
toilet.txt
trash.txt
```