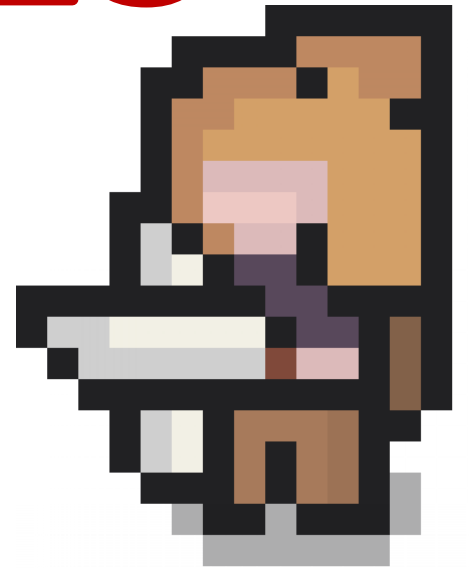
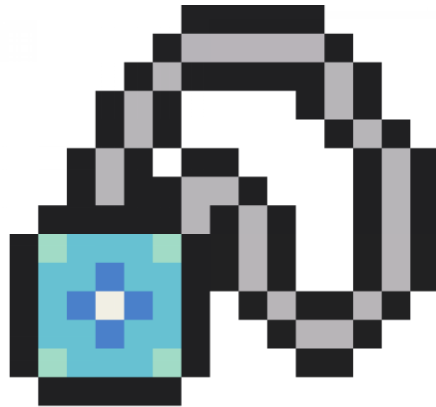
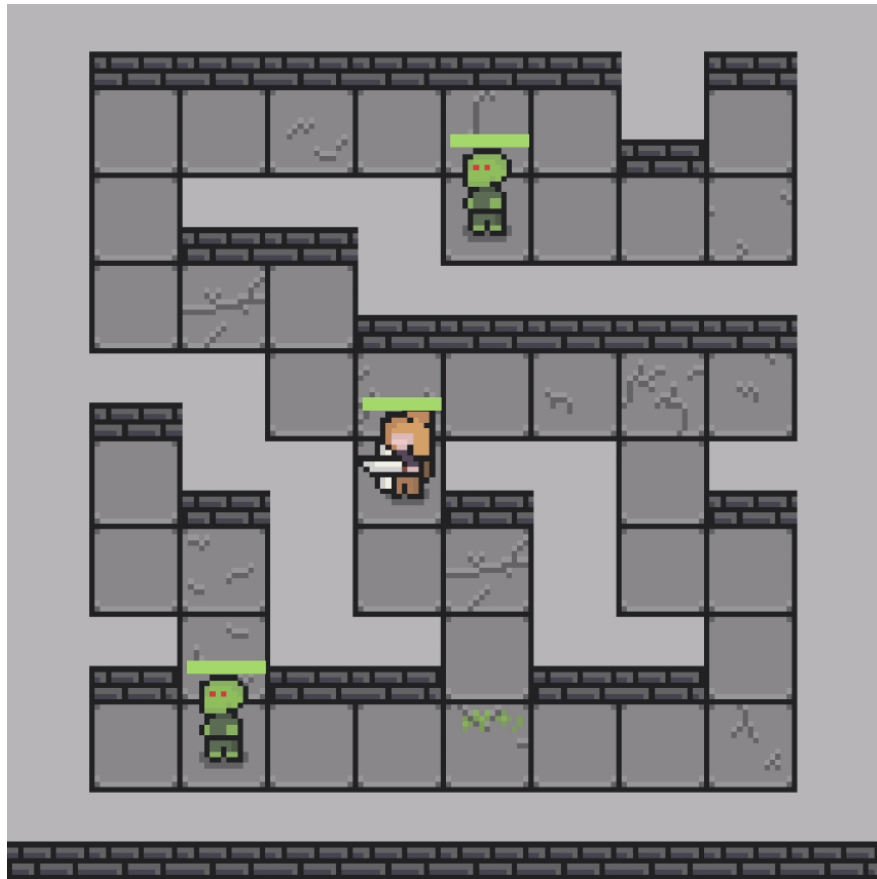


ZELDIABLO **TEAM DIABOLO**



@julienleows @florian-math @bilal-scouri @thibaut1308

PRESENTATION ET FONCTIONNALITES DEMO



VIDEO DEMONSTRATION :

Disponible en annexe sur le dépôt Github :



 [demo_zeldiablo_team_diabolo.mp4](#)

VERSIONS

V1

- Création d'un personnage,
- Création de case,
- Création labyrinthe,
- Collision du personnage avec les murs.

V2

- Gestion de niveaux,
- Génération des monstres immobiles,
- Intégration Graphique.

V3

- Déplacement aléatoire monstres,
- Attaque du joueur.

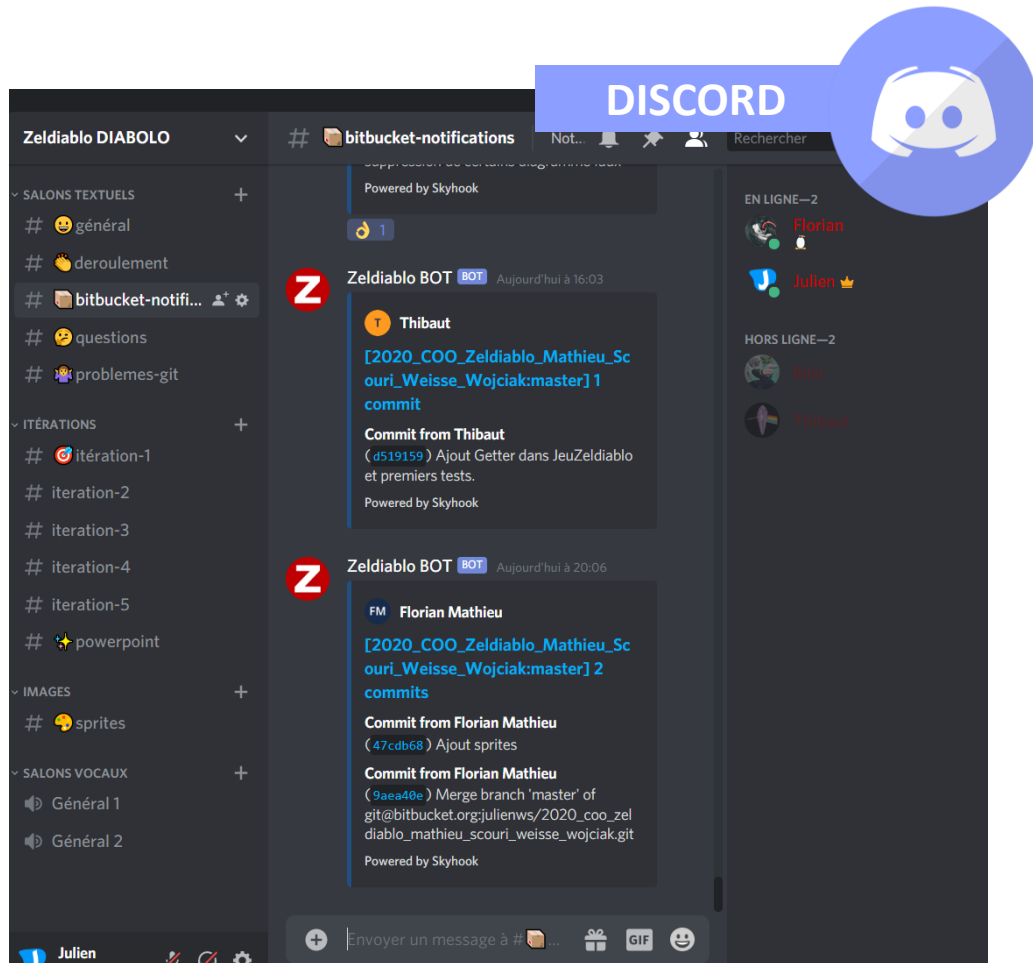
V4

- Morts des monstres,
- Attaque des Monstres,
- Collision avec un monstre,
- Fin du jeu : mort du héros,

V5

- Affichage avec des sprites,
- Mise en place Amulette,
- Acquisition Amulette,
- Gestion d'un labyrinthe multi-étages,
- Fin jeu : victoire du héros,
- Windows / linux,

REPARTITIONS DES TÂCHES



GOOGLE-DRIVE

```

* Fonctionnalités version 1 :

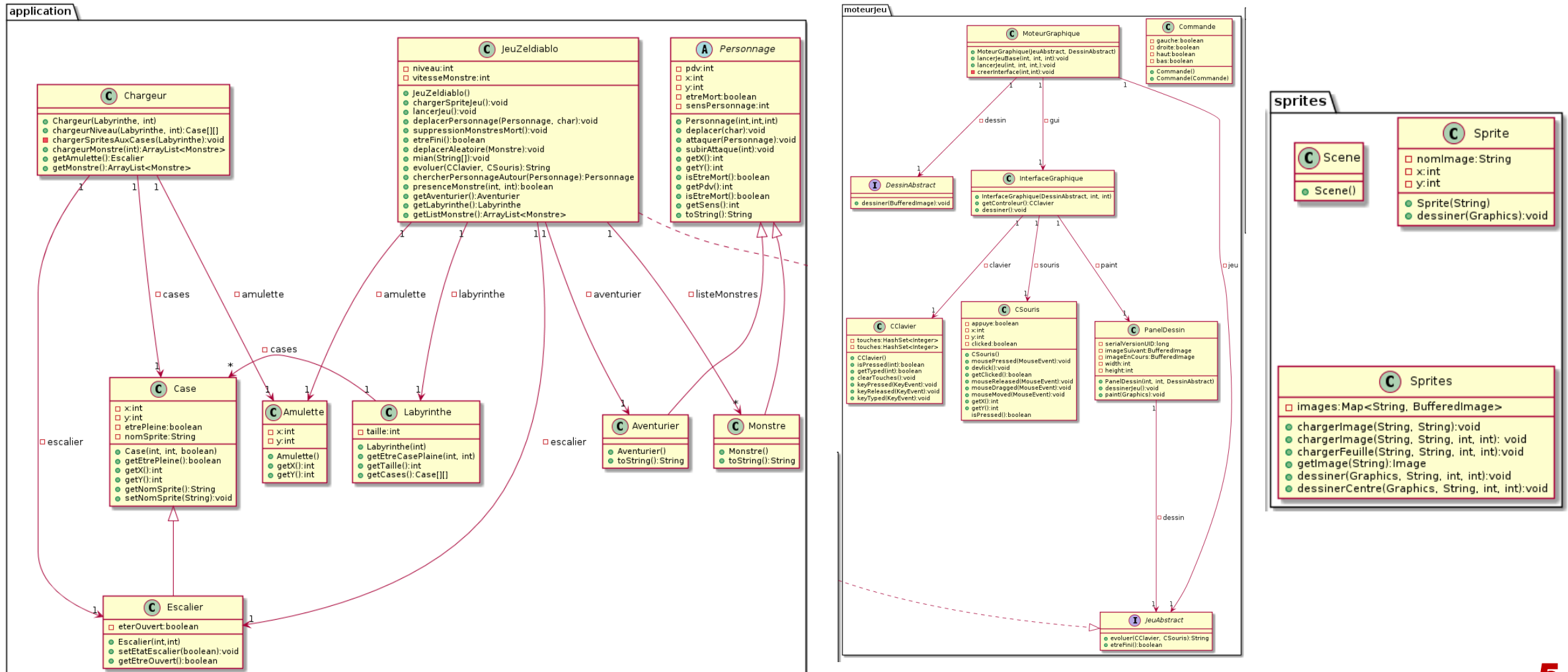
- fonctionnalité 1 : création d'un personnage / Aventurier
- descriptif texte : définir un personnage avec tous ses attributs
- critère de validation :
  - critère 1 : doit avoir une position (x,y),
  - critère 2 : doit pouvoir être mort ou vivant(booleen),
  - critère 3 : ne doit pas dépasser les coordonnées du labyrinthe lors de
    création et de ses déplacements,
  - critère 4 : doit avoir des points de vie.
- responsable : julienws, bilal, florian

- fonctionnalité perso 2 : création de case
- descriptif texte : Au lancement du jeu, un labyrinthe par défaut est créé avec les
  cases
- critère de validation :
  - critère 1 : les cases ont des coordonnées,
  - critère 2 : les cases peuvent être des cases pleines ou des case vide,
  - critère 3 : methode pour connaitre si la cases est un mur ou non.
- responsable : Florian Bilal

- fonctionnalité 3 : création d'un labyrinthe par défaut
- descriptif texte : Au lancement du jeu, un labyrinthe par défaut est créé et le
  personnage est placé au centre de celui-ci.
- critère de validation :
  - critère 1 : le labyrinthe par défaut est constitué d'une arène de taille 10*10,
  - critère 2 : création d'une arène de de taille 10*10,
  - critère 3 : doit pouvoir transmettre sa taille max,
  - critère 4 : le personnage placé au centre de l'arène.
- responsable : Thibaut

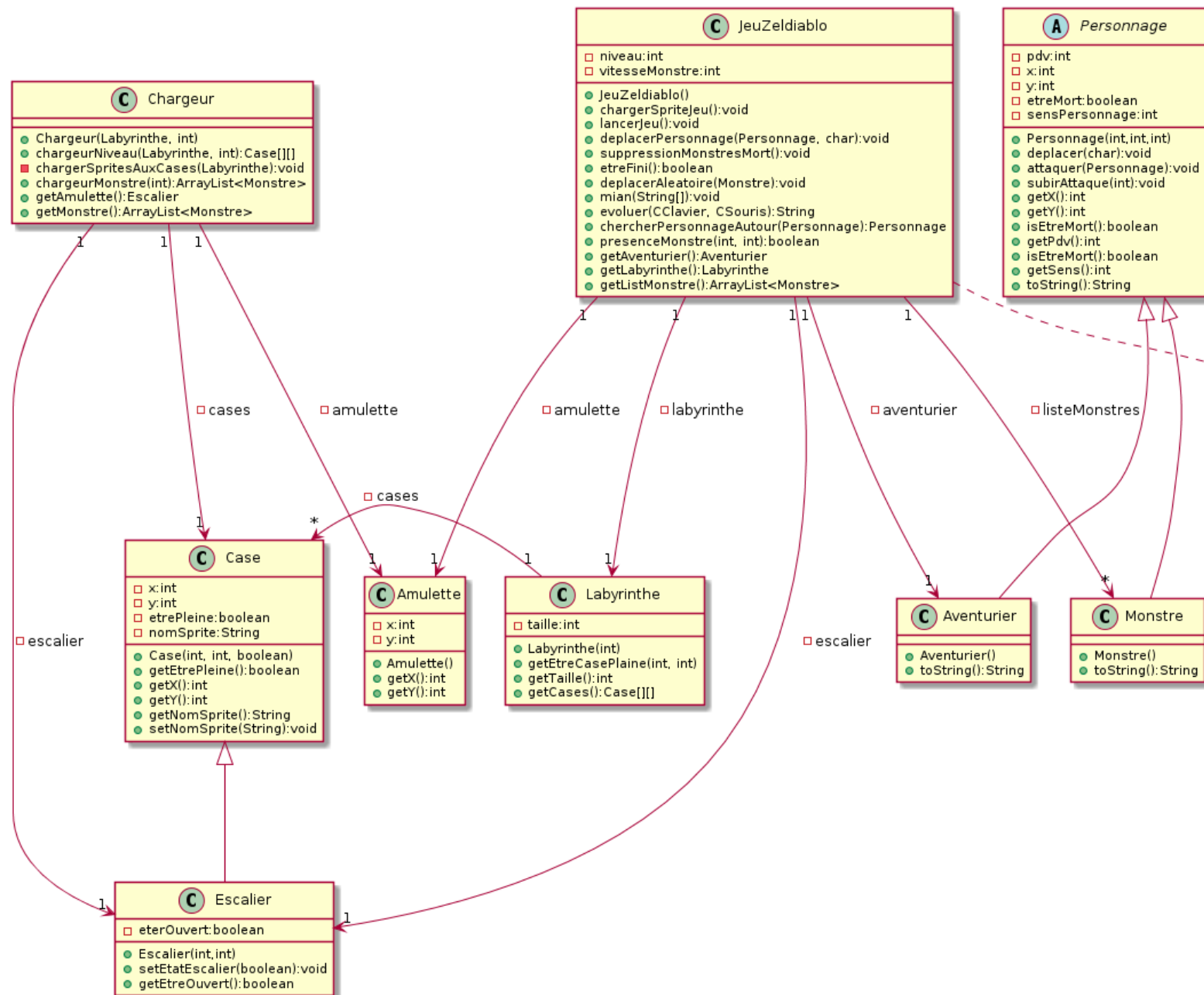
-fonctionnalité 4 : Collision avec les Murs
- descriptif texte : Lorsque le joueur demande à déplacer le personnage en direction
  d'un mur, le personnage reste bloqué et ne se déplace pas.
- critère de validation :
  - critère 1 : La collision fonctionne dans les 4 directions
  - critère 2 : Si la case est vide, le personnage avance.
- responsable : Florian
  
```

DIAGRAMMES DE CLASSE

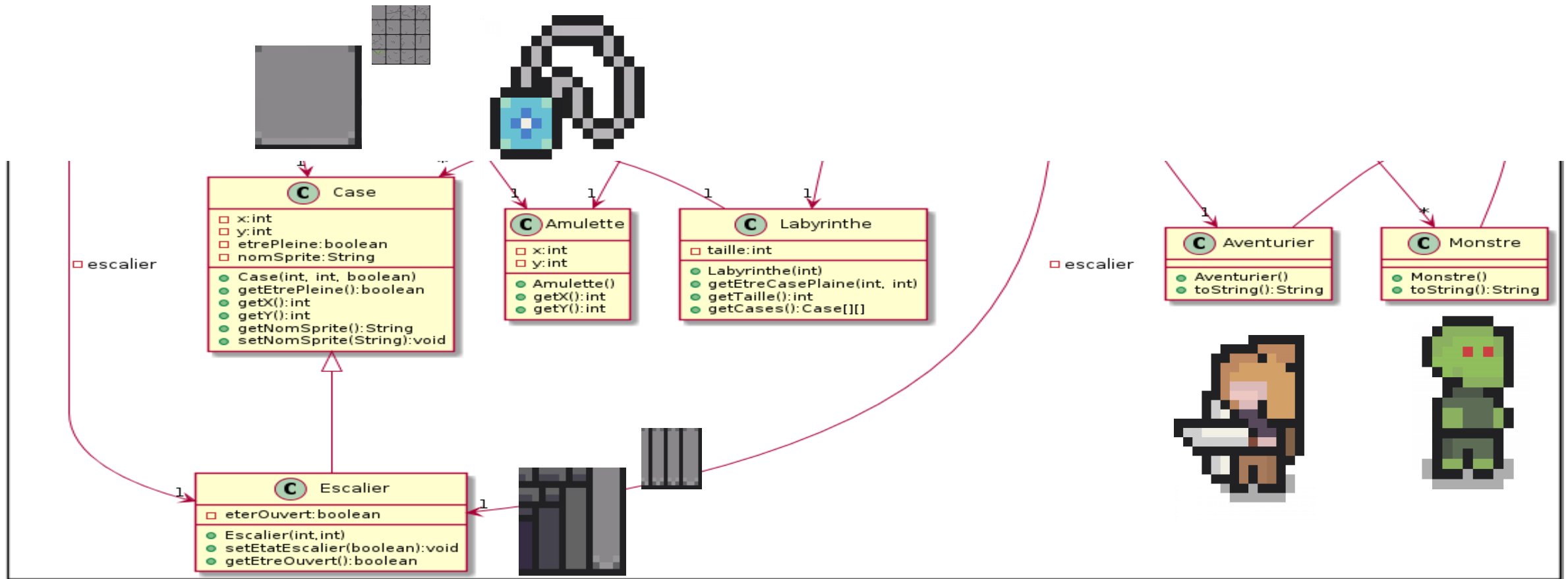




application



DIAGRAMMES DE CLASSE SPRITES (16*16)



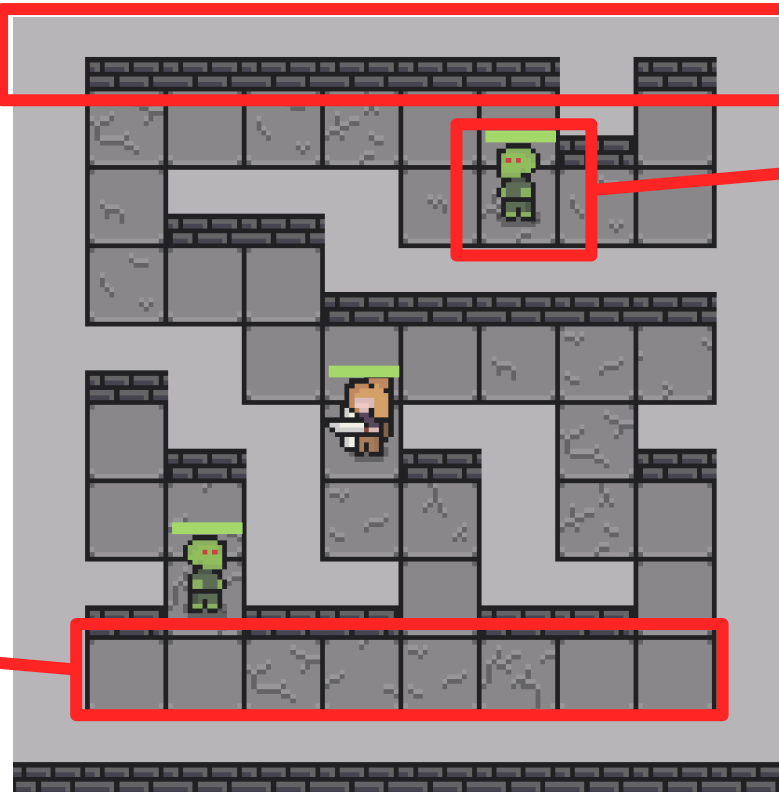
CHARGEMENT DU NIVEAU

niveau1.lab

```
1 MMMMMMMMMM
2 MVVVVVVVM
3 MVMMVVVVV
4 MVVVM MMMM
5 MMVVVVVVV
6 MVMMVMVM
7 MVVMVVMVM
8 MMVMMVMVM
9 M VVVVVVVV
10 MMMMMMMMMM
```

C Chargeur

- Chargeur()
- chargeurMonstre(int): ArrayList<Monstre>
- chargeurNiveau(int): Case[][]

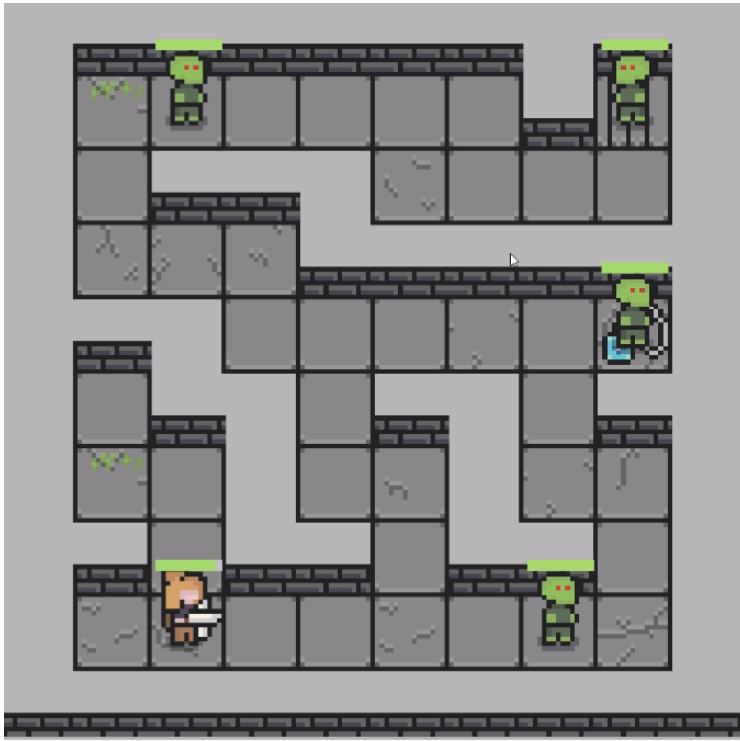


niveau1monstre.lab

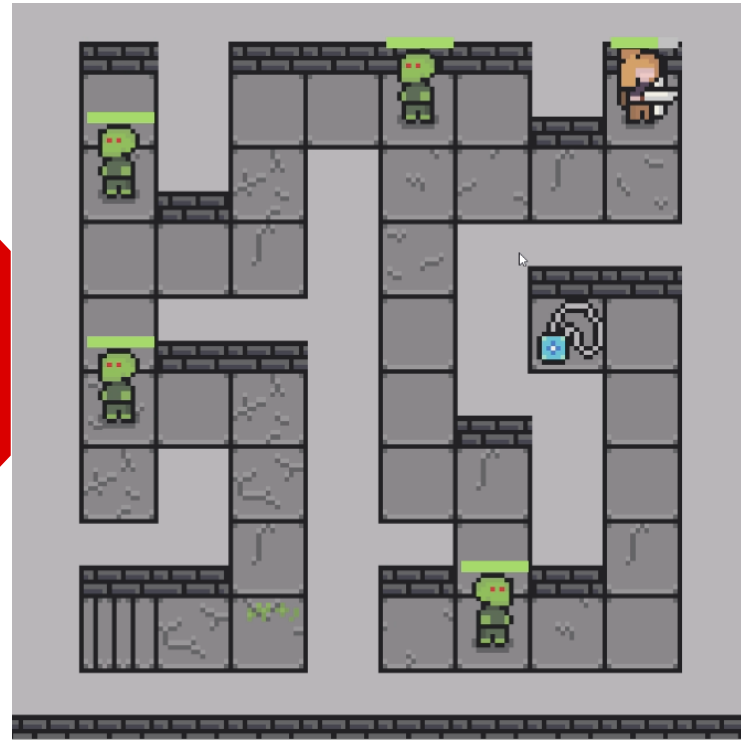
```
1 MMMMMMMMMM
2 MVMVVVVVM
3 MVMV O VVM
4 MVMVVVVVM
5 MVMVVVMVM
6 MVMVMVMVM
7 MVVMVVMVM
8 MVOVMVMVM
9 MVVVVVVVV
10 MMMMMMMMMM
```


LES NIVEAUX

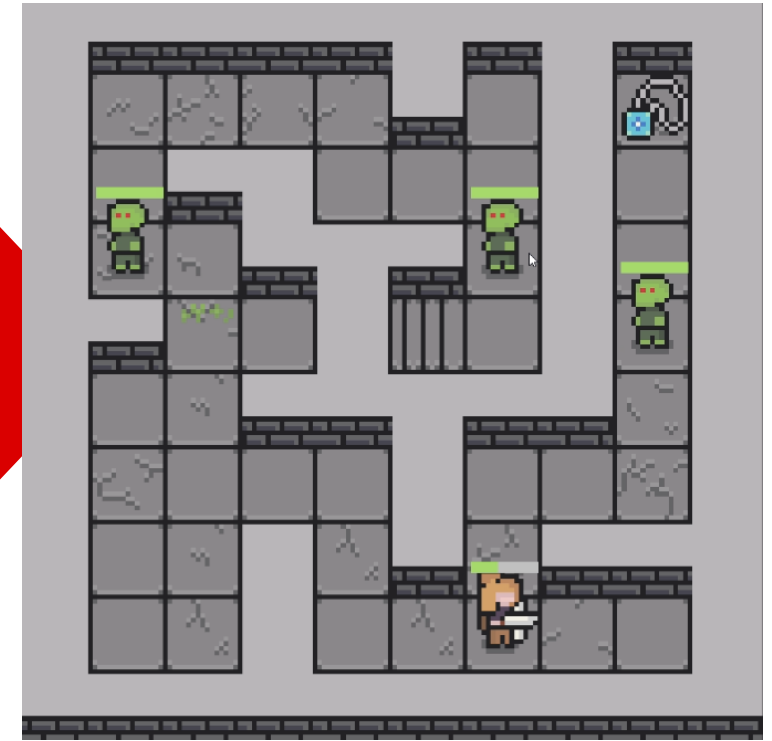
Niveau 1



Niveau 2

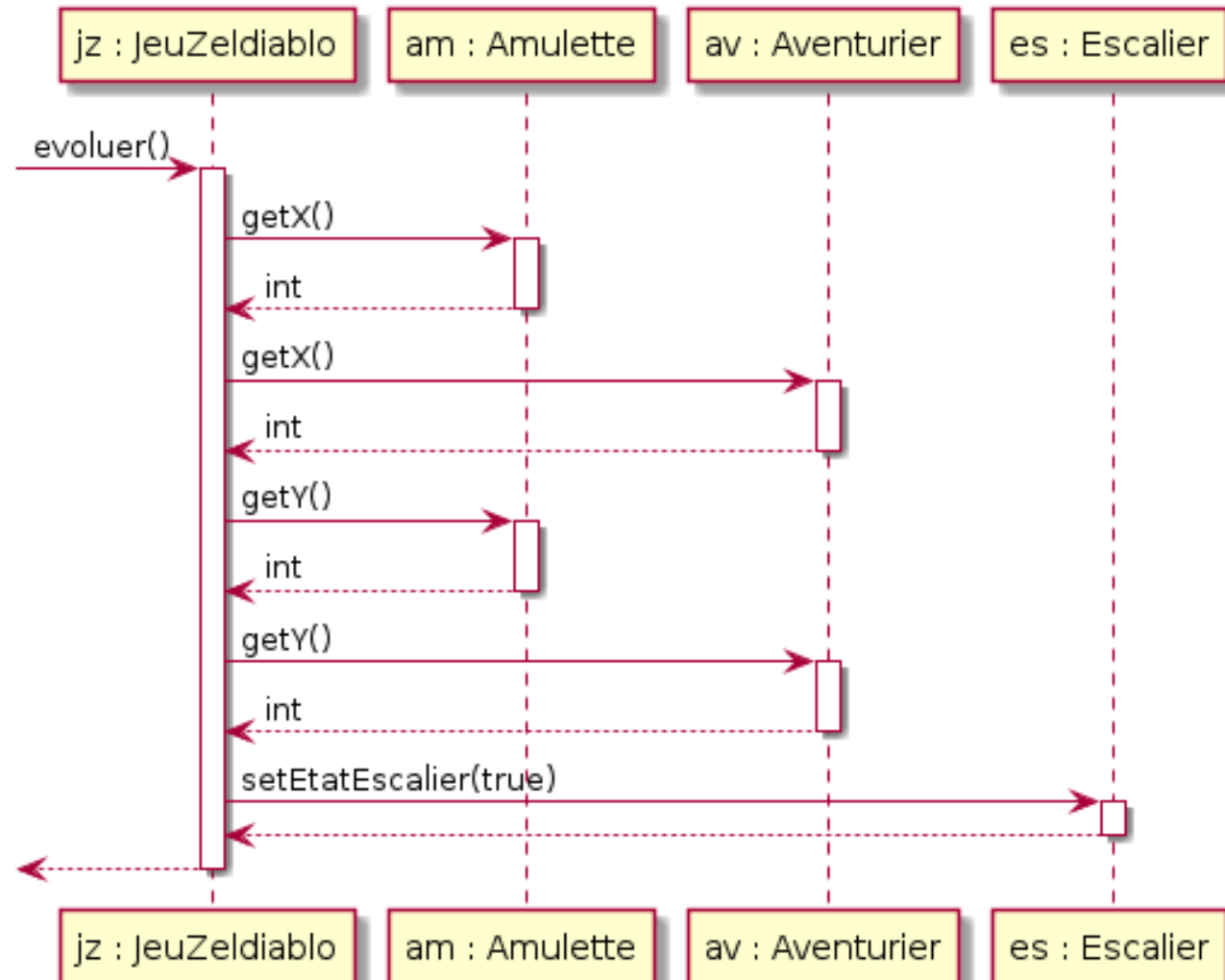


Niveau 3



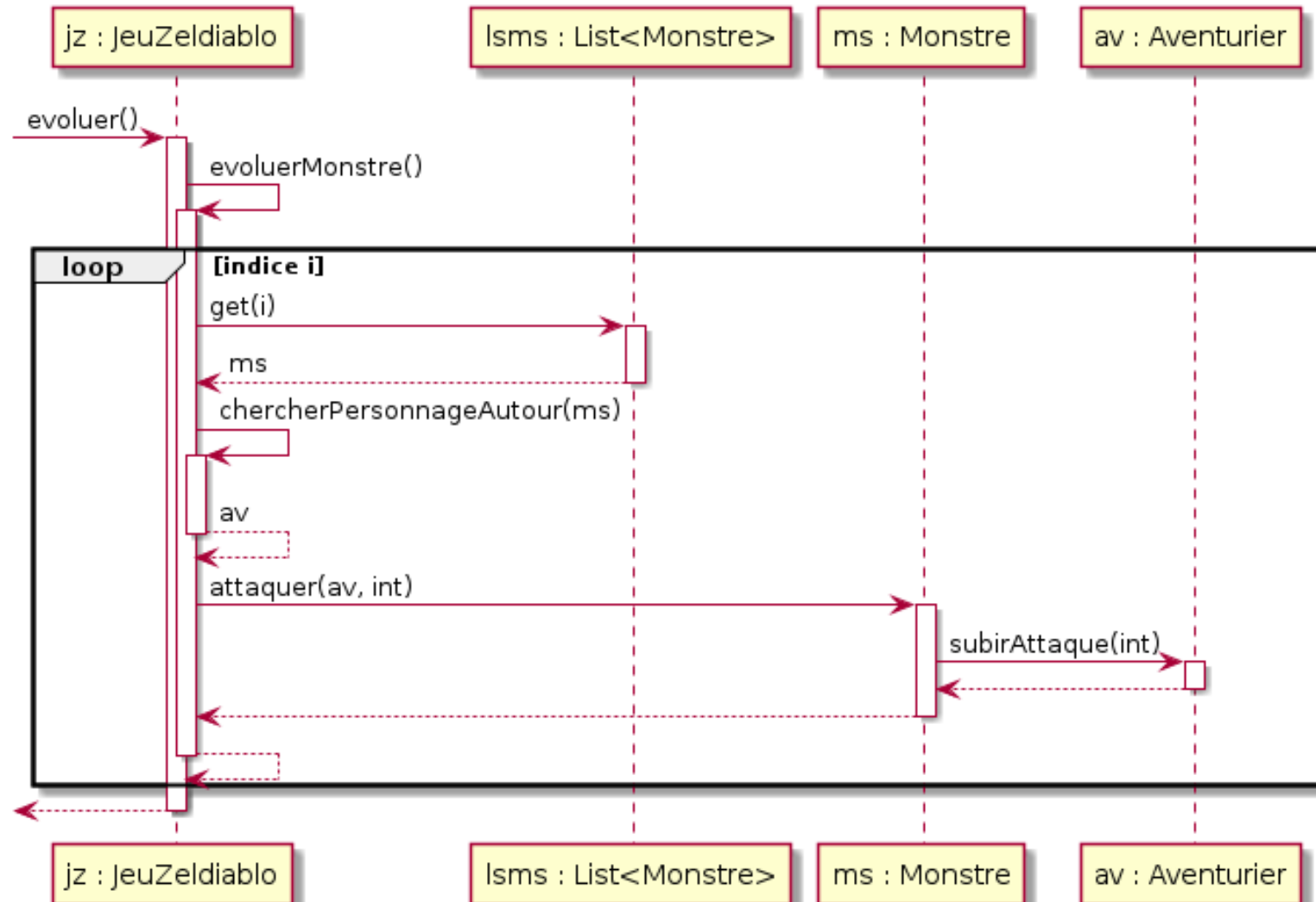
ACQUISITION AMULETTE

Diagramme de séquence de l'aquisition de l'amulette

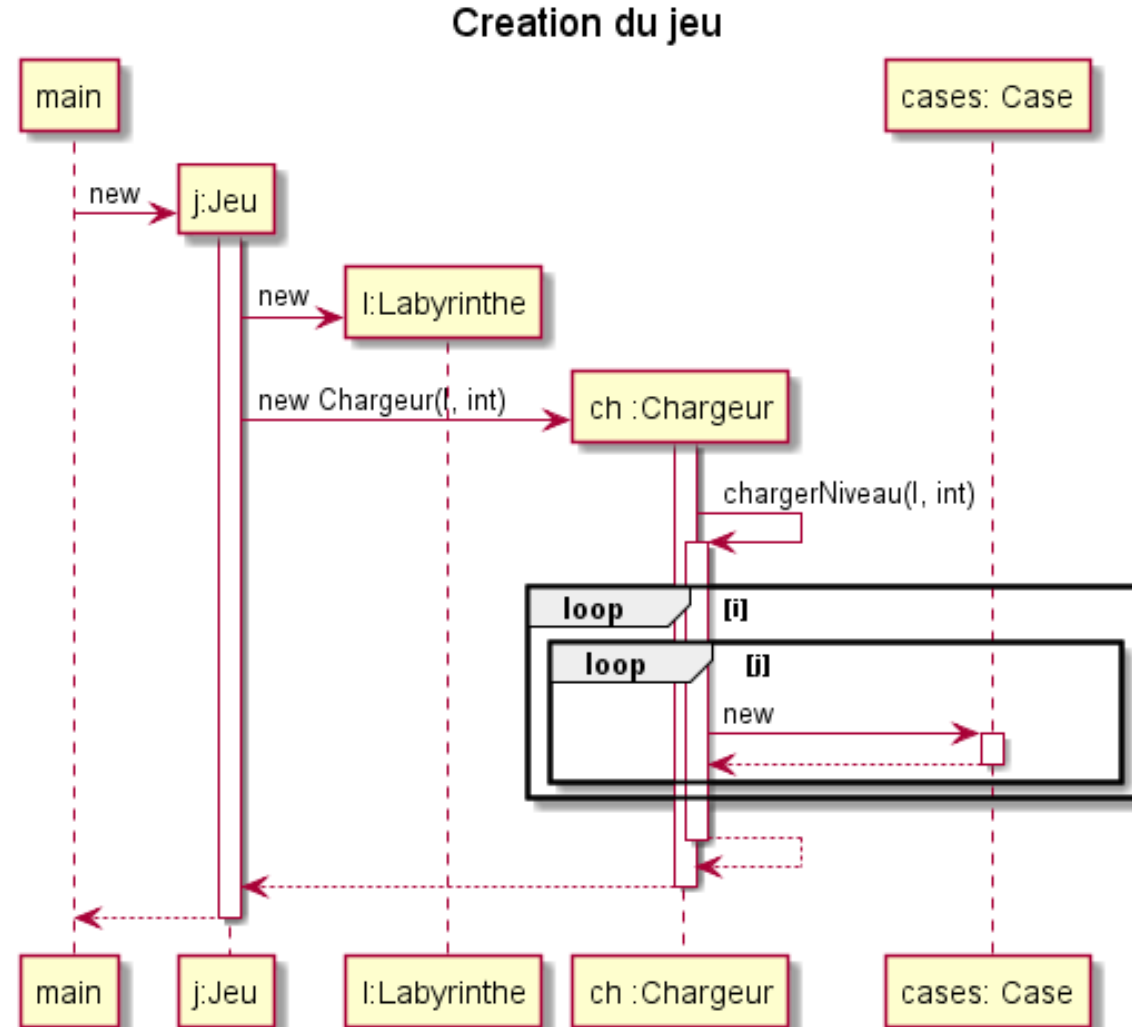


ATTAQUE

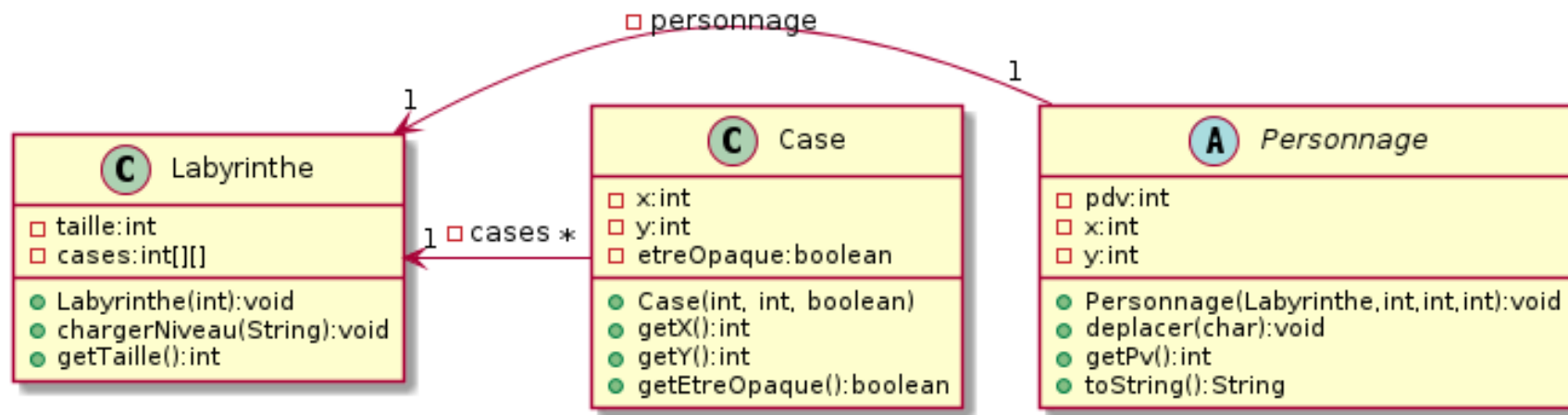
Diagramme de séquence de la méthode evoluerMonstre



CREATION DU JEU



EVOLUTION DIAGRAMMES DE CLASSE



CONCLUSION

- Un départ avec plein d'idées : trop de fonctionnalités pour la 1^{er} itération
- Il faut penser au plus simple => un seul type de monstre
- Plusieurs conceptions (à 3 fois)
- Gestion du groupe avec git et discord
- Gestion des conflits git
- Gestion du temps
- 1er projet en de conception en groupe
- Visualisation de notre projet à la fin => satisfaction

CONCLUSION

