Sprint review — Sprint 2

Escape

Le backlog en détail : https://github.com/victord54/escape/tree/main/docs/sprints/2

Rappel backlog sprint 2 :

- ✅ Sprites : Victor
- ✅ Sauvegarde du jeu : Antoine
  - Menu pause
  - Sauvegarde automatique des paramètres
  - Sauvegarder et charger/supprimer une partie
  - Prévention basique de la triche (sauvegardes chiffrées)
- ✅ Attaque (monstres) : Théo
- ✅ Attaque (Héros) : Théo
- ✅ Point de vie : Claire
- ✅ Ramasser cœurs : Claire
- ✅ Cases piégées : Claire
- Niveaux : Antoine
  - Retiré du backlog, pas assez de temps
- ✅ [bugfix] Impossible de se placer entre deux ElementMonde : Victor
- [bugfix] GUI lignes entre cases : Antoine
  - état actuel : en attente de la fonctionnalité “Sprites” (s’est terminée tard)
- ✅ [bugfix] Déplacement en diagonale : Théo

Modifications du backlog :

- ✅ [bugfix] Amélioration pathfinding : Claire
  - FPS très bas
  - Monstres qui se bloquaient légèrement
- ✅ Utilisation du JSON pour les cartes : Antoine

Problèmes rencontrés :

→ **Sauvegarde :** Le premier blocage rencontré était “Où mettre les fichiers du jeu ?”. La solution de facilité aurait été de les placer au niveau du JAR, mais ce n’est pas agréable pour l’utilisateur. On voulait donc les enregistrer ailleurs, mais alors, comment garantir que ça fonctionne sur tous les OS ? La solution a finalement été trouvée avec l’utilisation d’une library Maven qui permet de déterminer le chemin vers les données utilisateurs des applications pour chaque système (AppData/Local/… sur Windows, .local/share/… sur Linux, …).

→ **Sauvegarde :** Cette fonctionnalité était très dense, il y a beaucoup de choses qui se cachent derrière ce simple mot. De tout ce qui a été fait sur cette fonctionnalité, le plus difficile a été de réaliser la vue des parties enregistrées. C’est une vue avec une liste dans laquelle chaque élément correspond à une sauvegarde que l’on peut supprimer ou charger. Réaliser cette vue est donc complexe, en effet, il faut créer un composant de cellule personnalisé, c’est peu pratique à faire, et il ne se charge pas exactement de la même

manière. Les composants sont complexes à intégrer, surtout quand on veut gérer les événements des boutons qu'ils contiennent.

→ **Sprites :** Lors de l'implémentation des sprites, tout semblait fonctionner, mais de par la nature de l'implémentation des images dans Java, le code n'était pas compatible quand on lançait le programme sans interface. Lors des tests, une erreur au niveau du chargement des sprites apparaissait et faisait donc échouer le workflow. La solution trouvée a été de tester une valeur en fonction du démarrage de l'app (CLI ou GUI) pour charger ou non les sprites et ainsi éviter les erreurs de chargements de ressources.

Diagramme de classes général :

Consulter avec une meilleure qualité (permalink), Consulter la version du sprint précédent

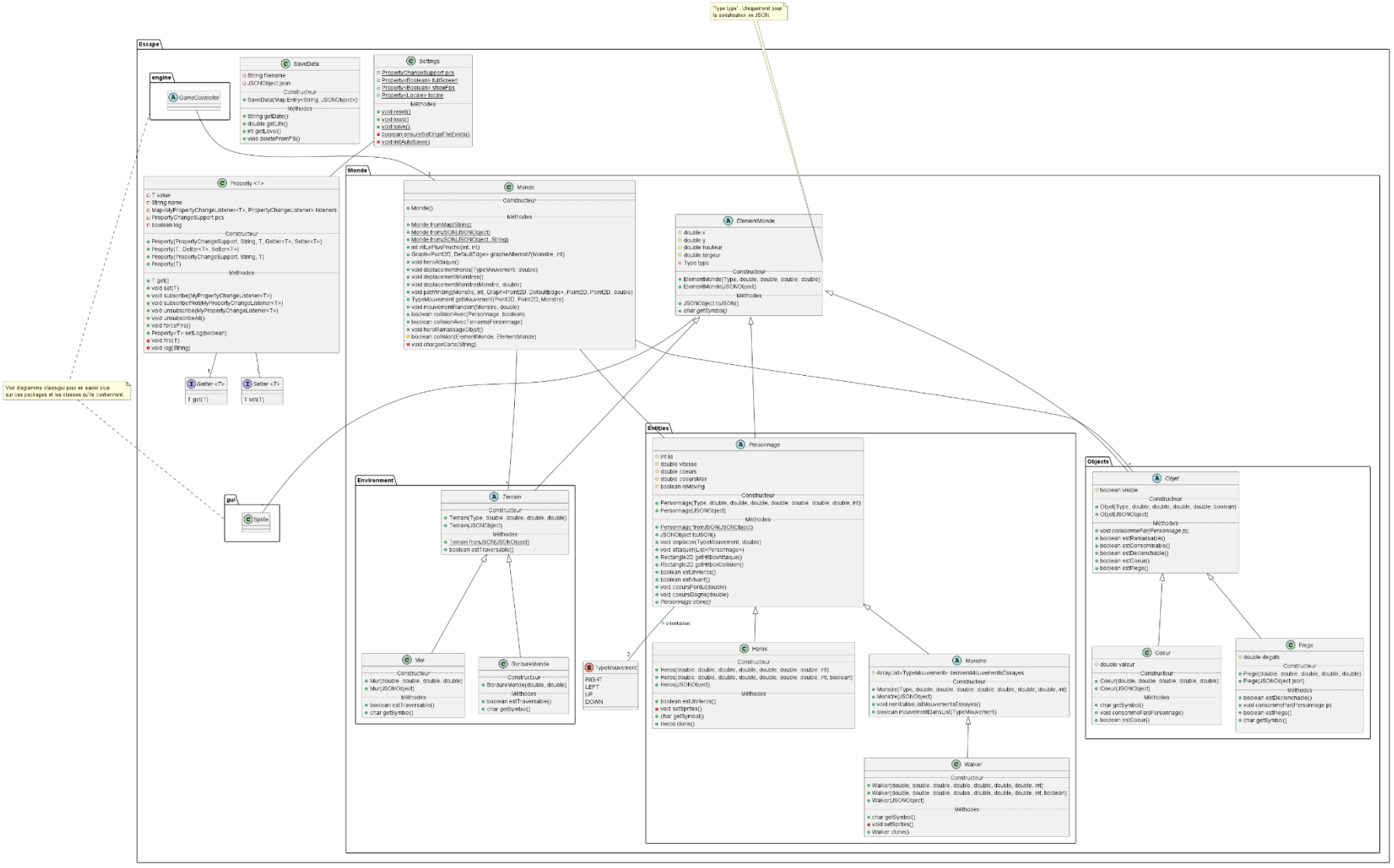

Diagramme de classes de l'interface graphique :

Consulter avec une meilleure qualité (permalink), Consulter la version du sprint précédent

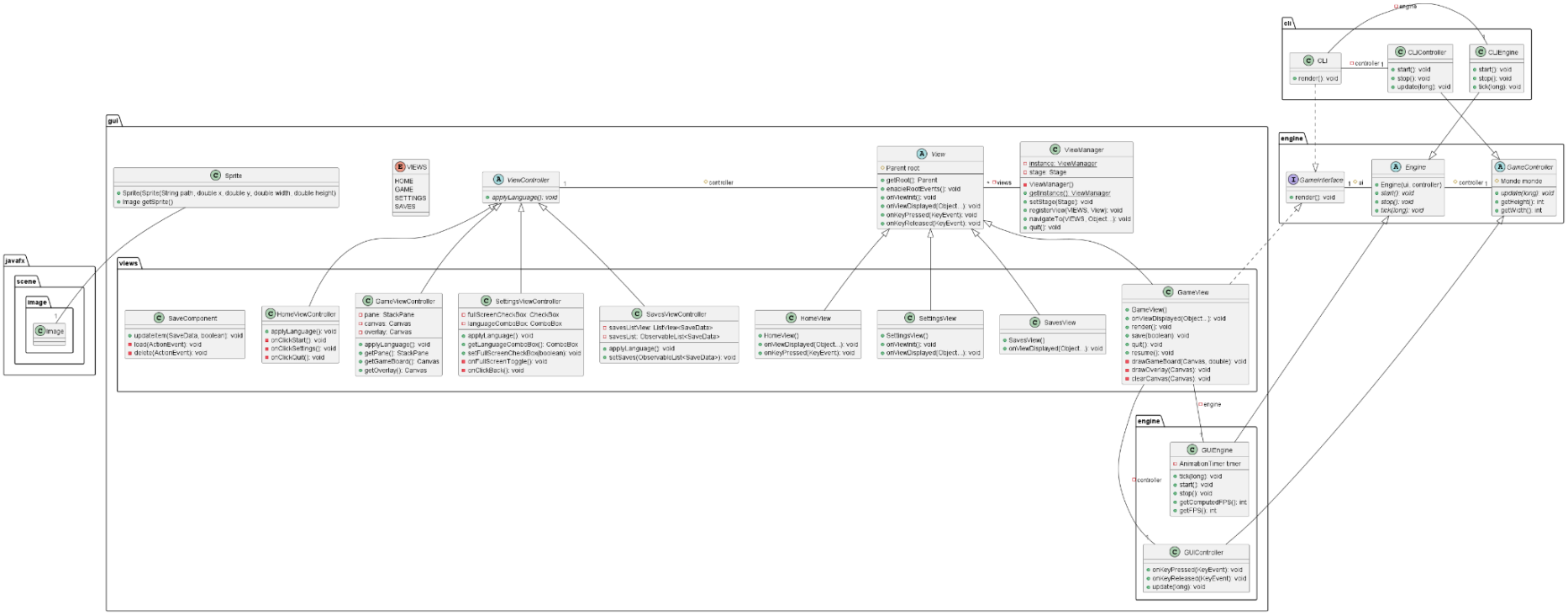

Diagramme de classes des outils :

Consulter avec une meilleure qualité (permalink)