Conception et Programmation Objet
Projet long : Bomber7

À l'attention de : M. Xavier Crégut, Mme. Aurélie Hurault, Étudiants de FISA L3 SN

Team 4 : Thomas SILVESTRE - Nikita ZIUZIN - Stephane LOPPINET - Yoann FRANCOIS - Corentin PRADIER - Pierre CHAVEROUX

Sommaire

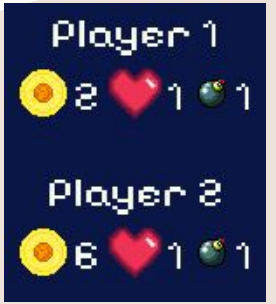
Partie 1 : Démonstration

Partie 2 : Présentation technique

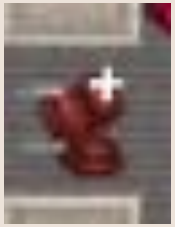
Partie 3 : Gestion de projet

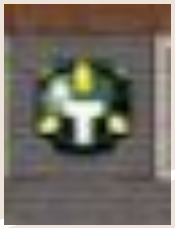
Partie 4 : Les perles

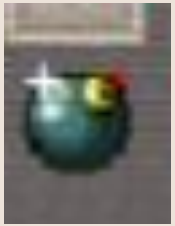
Partie 1 : Démonstration

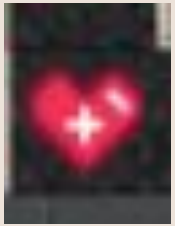



Eleve 1
Prof
Demo 2









Partie 2 : Présentation Technique

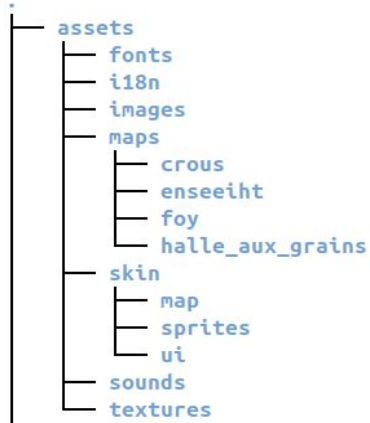
A. Packages

```
bomber7
├── core
│   ├── components
│   ├── controller
│   ├── model
│   │   ├── entities
│   │   ├── exceptions
│   │   ├── map
│   │   ├── square
│   │   └── texture
│   ├── screens
│   └── views
└── utils
```

Les différents paquetages de notre application :

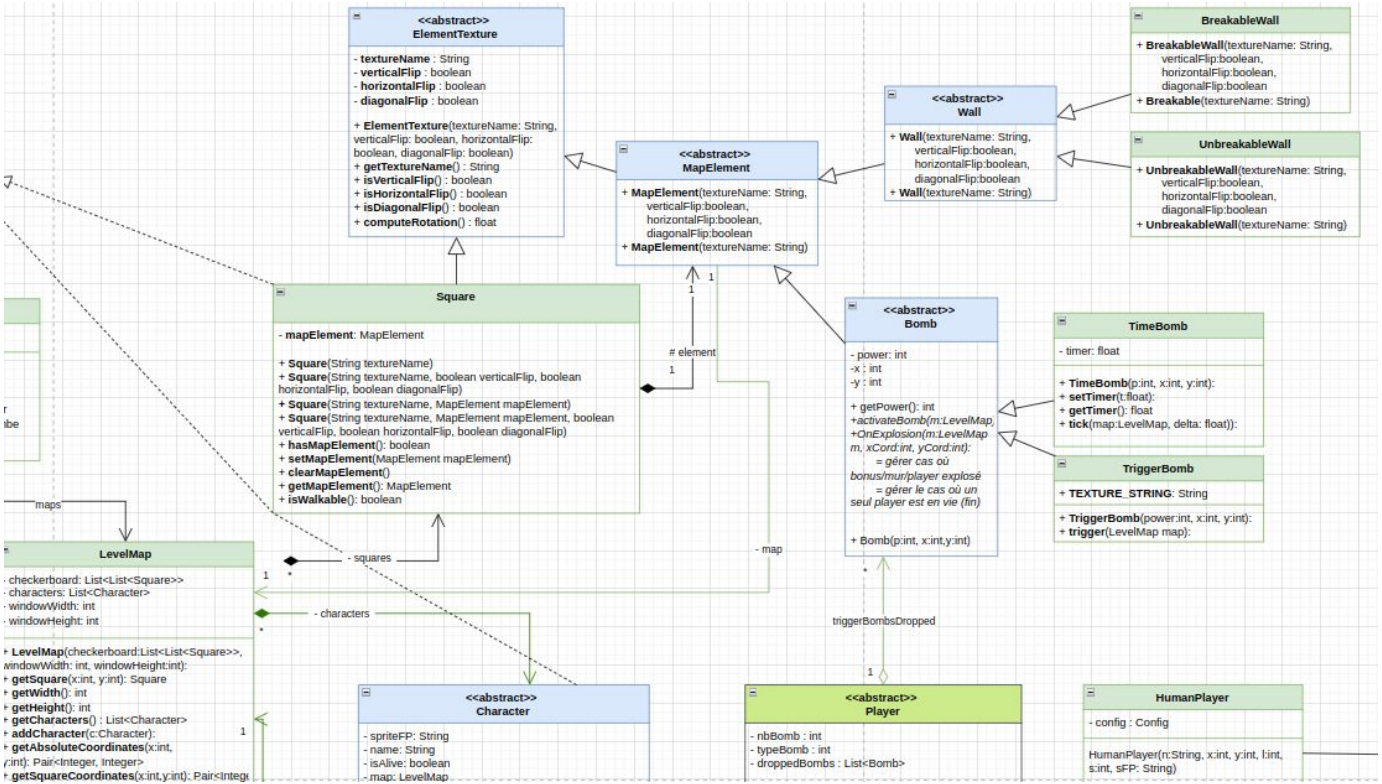
- components : éléments graphiques personnalisés
- controller : configuration des touches du joueur
- model : la logique de notre application
  - entities : les entités du jeu → Joueurs, IA, monstres, etc.
  - exceptions : règles sur les setters (vie, vitesse, …)
  - map : création des cartes
  - square : éléments relatifs aux cases de la map
  - texture : manipulation des textures
- screens : écrans du jeu, visibles à l'utilisateur
- views : sous-mécanismes d'affichages

Les ressources statiques du jeu :



Partie 2 : Présentation Technique

B. Architecture UML : Modèle

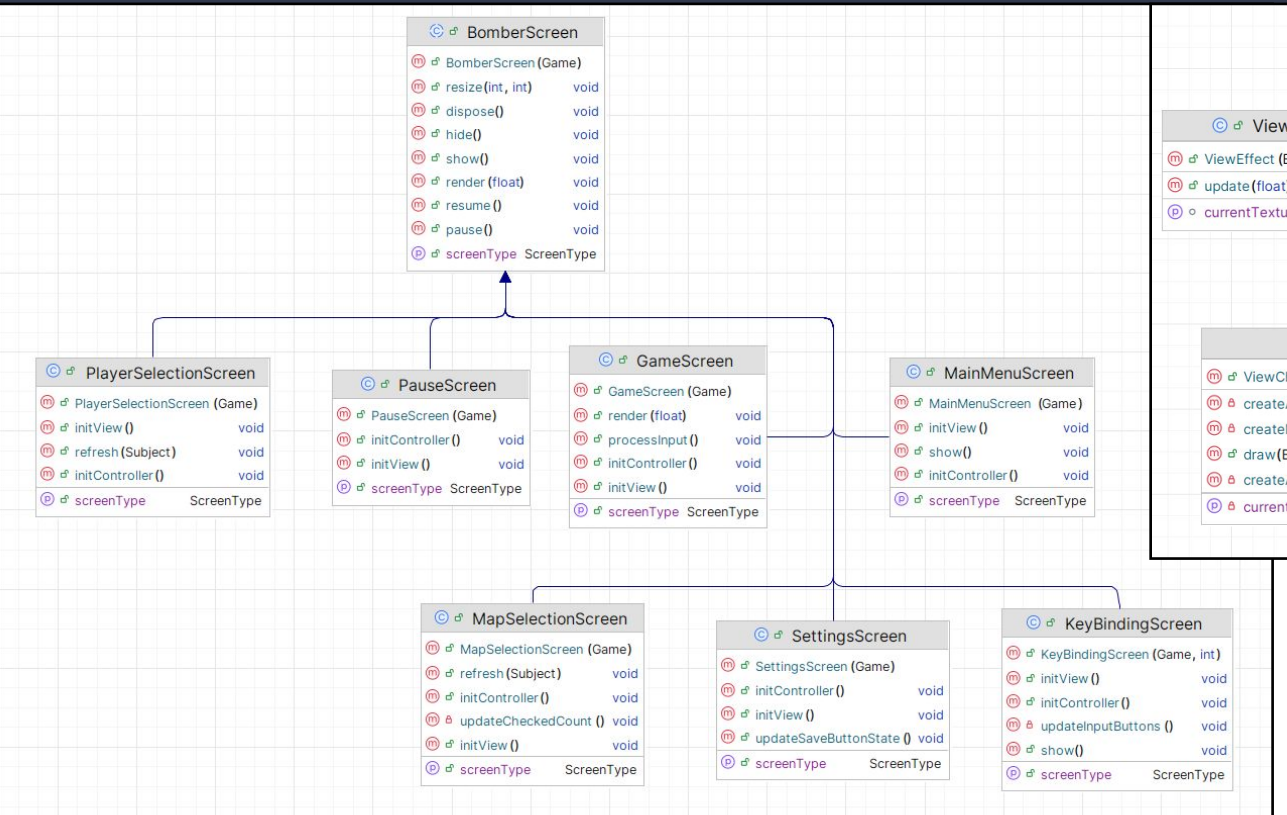


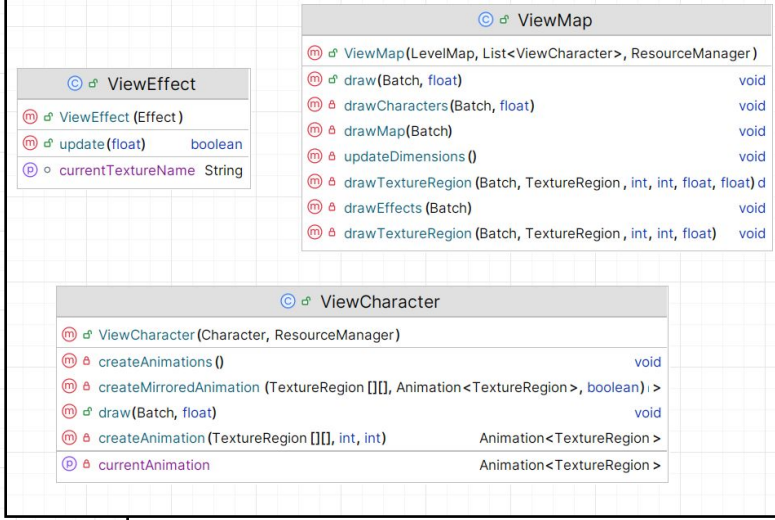
Les classes importantes :

- Character : Player, …
- Wall : (un)breakable
- Bomb : Time, Trigger
- Bonus : Life, Speed, ..

Partie 2 : Présentation Technique

B. Architecture UML : Vue



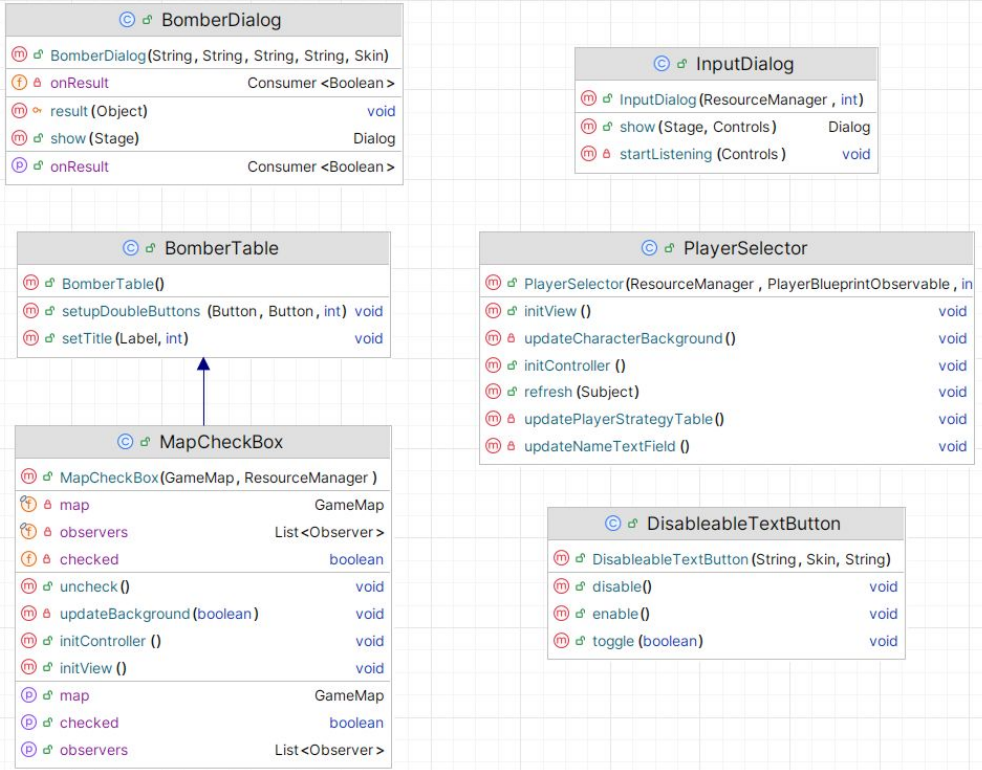


Les classes importantes :

- BomberScreen, Screen
- ViewCharacter, ViewMap, ViewEffect,

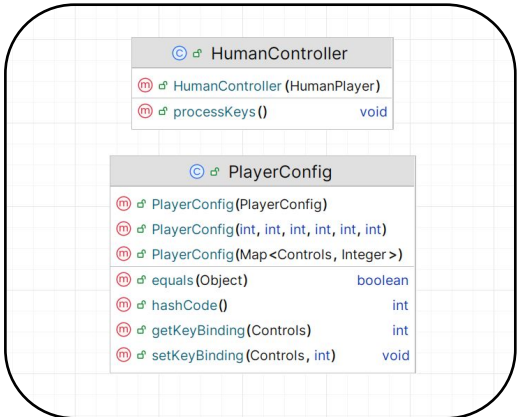
Partie 2 : Présentation Technique

B. Architecture UML : Contrôleurs



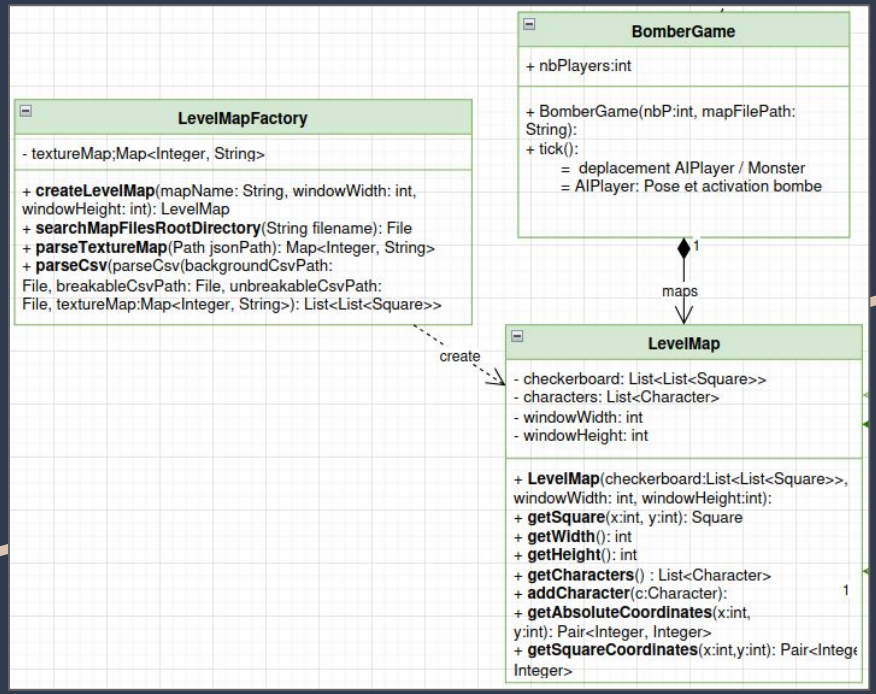
Deux types de contrôleurs

- Contrôleurs pour les “bindings” de touches
- Contrôleurs pour les éléments graphiques



Partie 2 :
Présentation Technique

B. Choix de conception



Les patrons de conception :

- **MVC :** HumanPlayer, ViewCharacter, HumanContr.

→ Modèle-Vue-Contrôleur

- **Observateur :** Components uniques (PlayerSelector - GC)

→ Vues interrogent le modèle sur quel élément afficher

- **Singleton :** Ressources, SoundManager, …

→ Une seule instance et fournit un point d'accès global

- **Fabrique :** LevelMapFactory

→ Interface pour créer des objets : ici les levelMap

- **Stratégie :** AIDropBomb → AIDumb, AISmart

→ Méthode commune, comportements différents

Partie 2 :
Présentation Technique

C. Choix techniques

Les outils utilisés :

- **LibGDX :** Framework Open Source Java

→ Facilitation du rendu 2D, de la gestion de l'horloge, de la musique et des effets graphiques (sprites)

- **Gradle :** Gestion des dépendances

→ Téléchargement automatique des bibliothèques Java depuis des dépôts reconnus : Maven Central.

→ Permets d'exécuter les tests et vérifier les erreurs de checkstyle.

- **GitHub Pipeline CI/CD :**

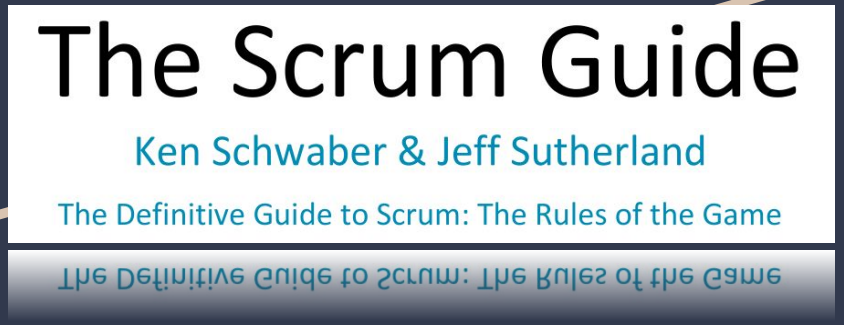
→ WorkFlows avec vérification Checkstyle

→ Run des tests pour le maintien de la qualité du code

→ Protection de la main (Merge Request Reviews, …)

Partie 3 :
Gestion de projet

A. Méthodes agiles



Outils :

- GitHub Projects :

→ Gestion des sprints, des ressources, des missions

→ Graphiques utiles pour le post-mortem du projet

- 4 sprints aux missions dédiées :

→ Sprint 1 : Architecture applicative - UML - Pipeline

→ Sprint 2 : Classes principales du modèle + tests

→ Sprint 3 : Vues (map, joueur, bombe) et contrôleurs

→ Sprint 4 : Merge en main et interfaçage final

- Palliatif aux cours de GDP :

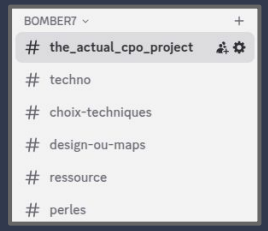
"The Scrum Guide", Ken Schwaber & Jeff Sutherland

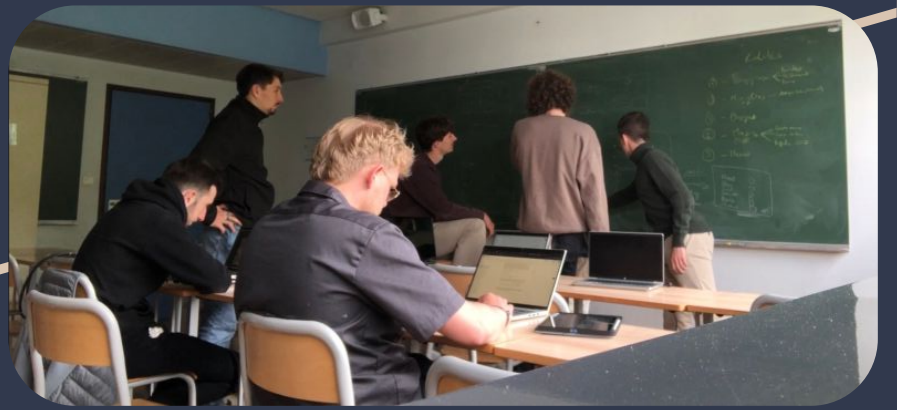
→ Edition 2020,

→ https://www.scrum.org/resources/scrum-guide

Partie 3 :
Gestion de projet

B. Outils GDP





Médias de communication :

- Présentiel :

→ Travail en classe sur les créneaux en autonomie

→ Mise en commun des connaissances sur tableau

- Distanciel :

→ Discord : partages des ressources, idées …

Outils de gestion de projet :

- GitHub Projects :

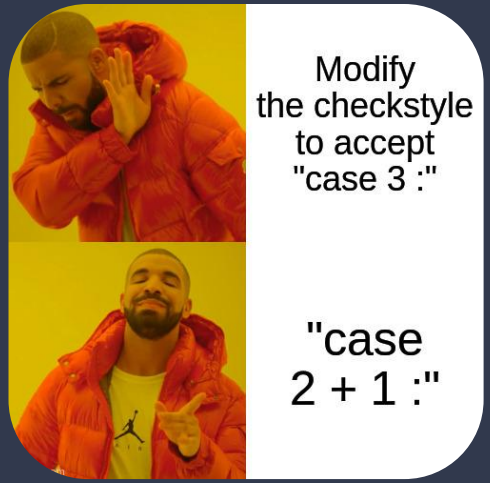
→ Visualisation des sprints, répartition des tâches

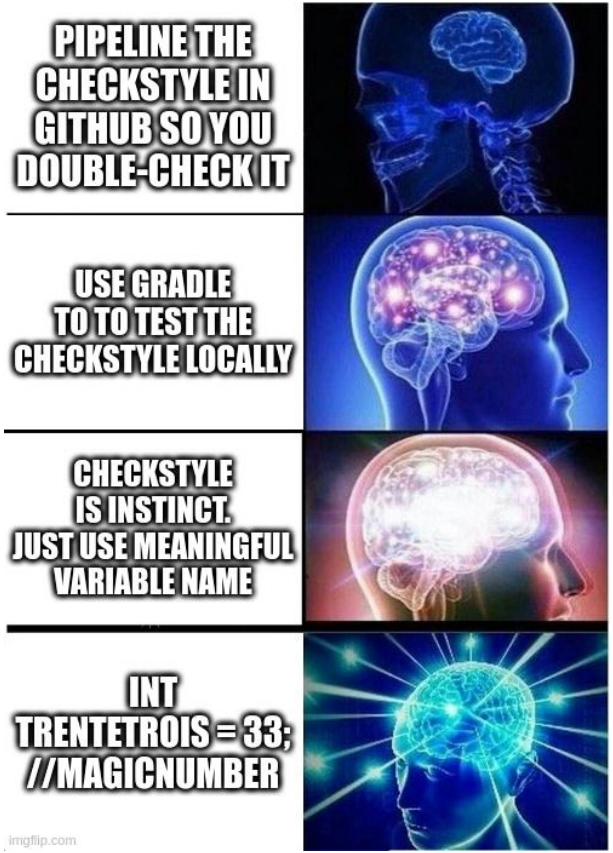
→ Suivi du chemin critique du projet

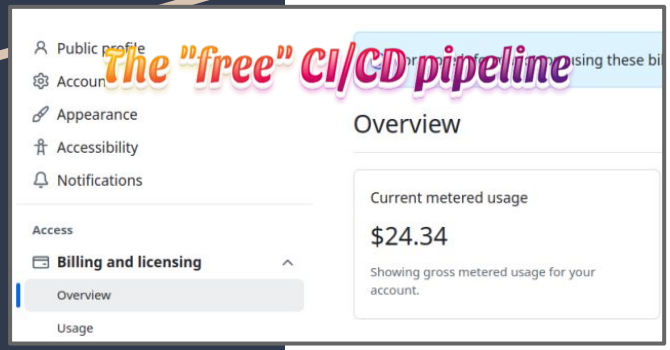
→ Mise en commun des avancés sur le code : PR reviews validées après minimum trois validations

→ https://github.com/users/Stephane-Lpt/projects/5

Partie 4 : Les perles







Merci pour votre écoute !
Place aux questions

Sources :

Logo Signal : https://en.wikipedia.org/wiki/Signal_%28software%29
Logo Discord : https://www.logiciels.pro/logiciel-saas/discord/
The Scrum Guide : https://www.scrum.org/resources/scrum-guide
Mooc GDP : https://gestiondeprojet.pm/
MemeGenerator : https://imgflip.com/
LibGDX : https://seeklogo.com/vector-logo/402393/libgdx
GitHub Pipeline Logo : https://medium.com/@blogs4devs/
Gradle Logo : https://cdnlogo.com/logo/gradle_43471.html

BACKUP

Présentation Technique

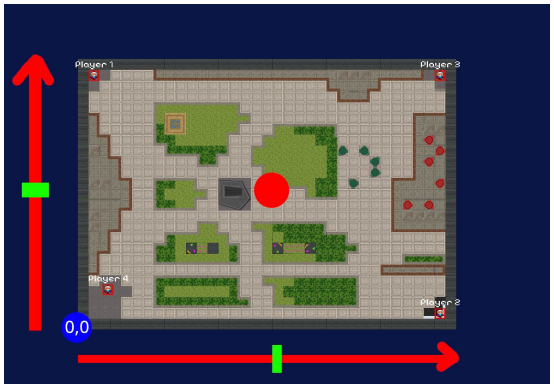
Map

Dessiner les éléments :

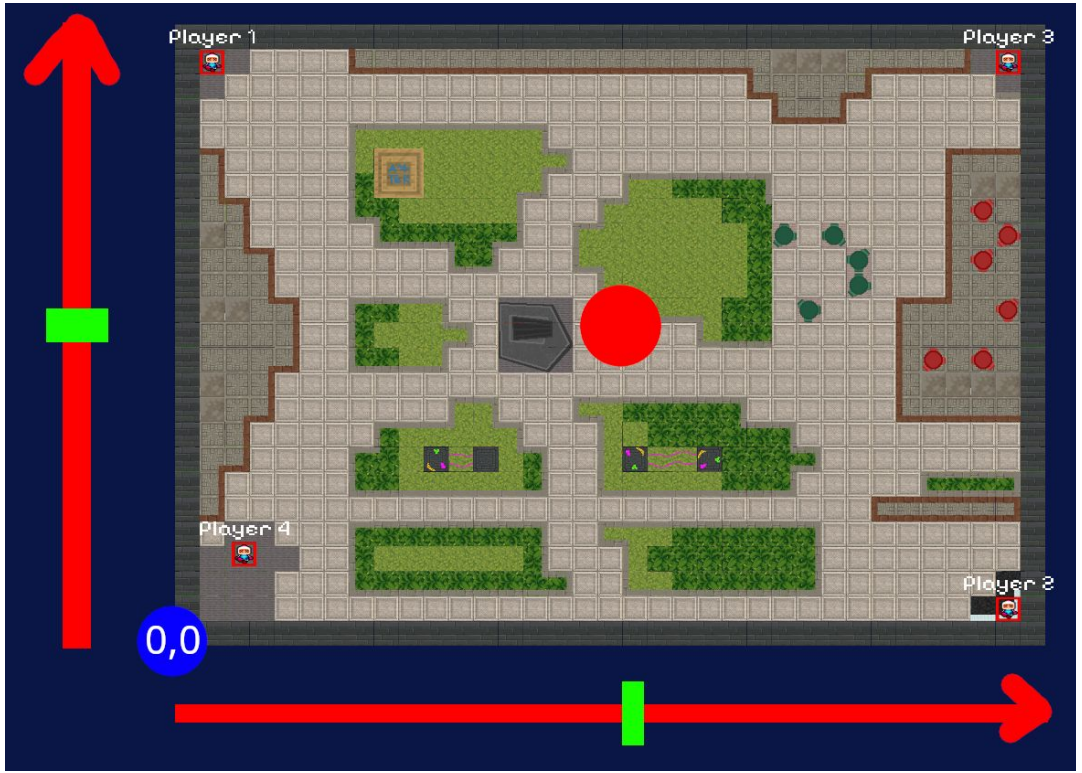
- Gérer logique affichage
- Problématique : Vitesse en pixels
  - Sur quelle case suis-je ?

Collision :

- `+ getAbsoluteCoordinates(int x, int y): Pair<Integer, Integer>`
- `+ getSquareCoordinates(int x, int y): Pair<Integer, Integer>`




Player 4
y
my
mx,
my
x
mx


Player 1
Player 3
Player 4
Player 2
0,0

Partie 2 :
Présentation Technique

D. UMLv1 VS UMLv2

V1:

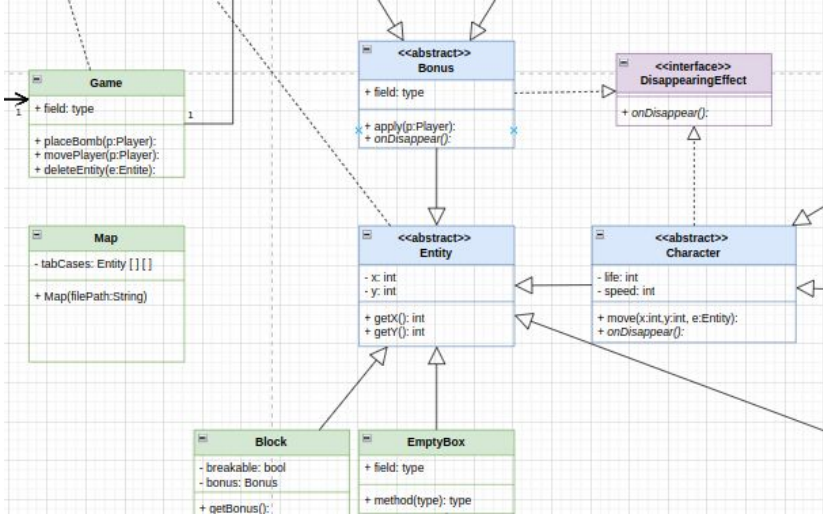
1. Tout hérite de Entity.
2. Map contient une liste d'entités

Problème :

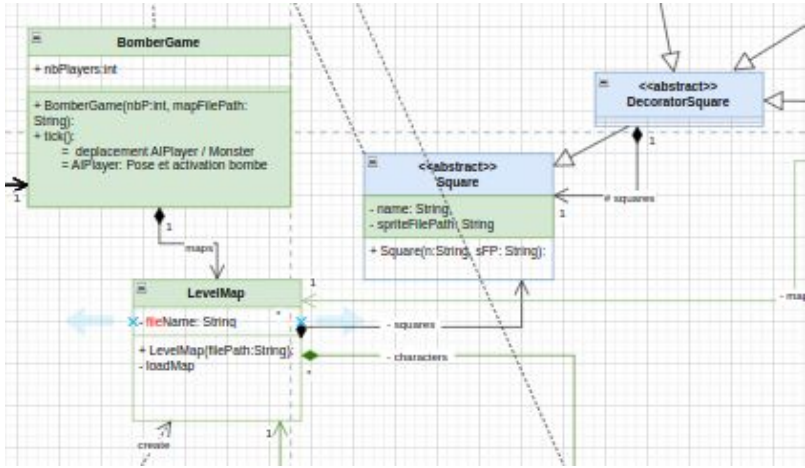
1. Manque de hiérarchie.
2. Déplacement du personnage = Le rechercher dans tout le tableau

Solution : Distinguer les acteurs principaux du reste

V1



V2



Partie 2 :
Présentation Technique

D. UMLv2 VS UMLv3

V2:

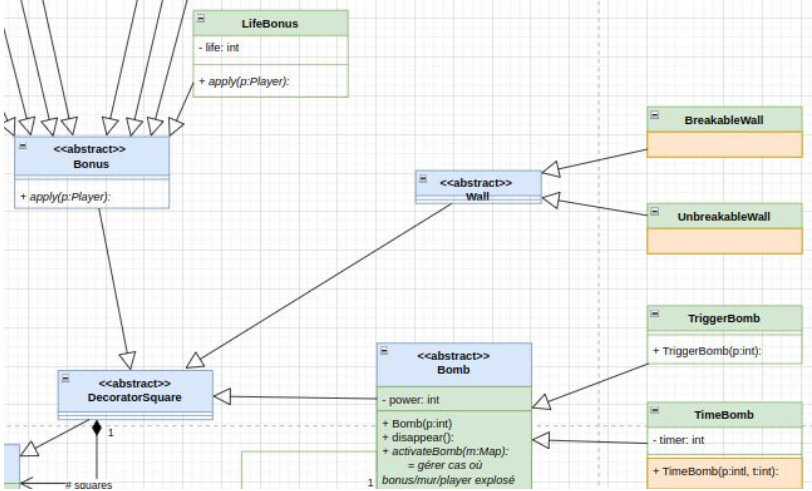
1. Décorateur = plusieurs éléments qui peuvent s'accumuler sur une case

Problème :

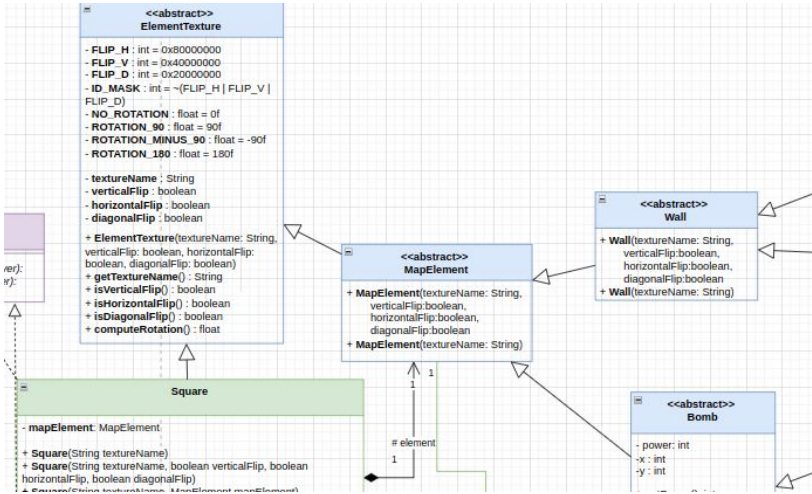
1. Il n'y aura jamais deux éléments au même endroit (soit un mur, soit un bonus, …).

Solution : Classe abstraite MapElement

V2

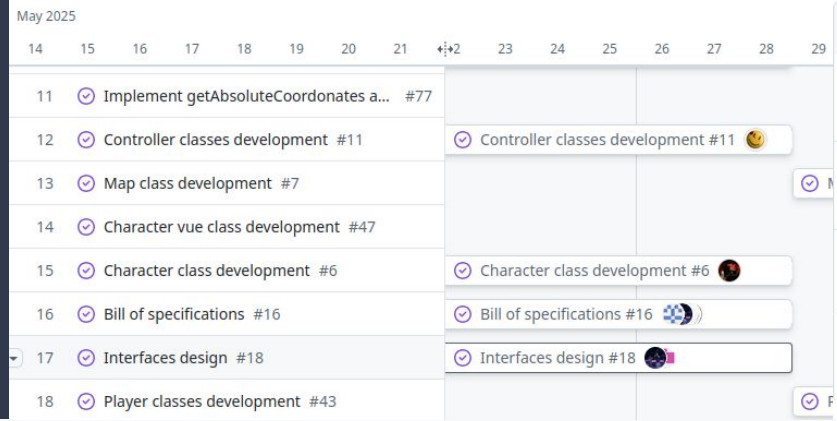


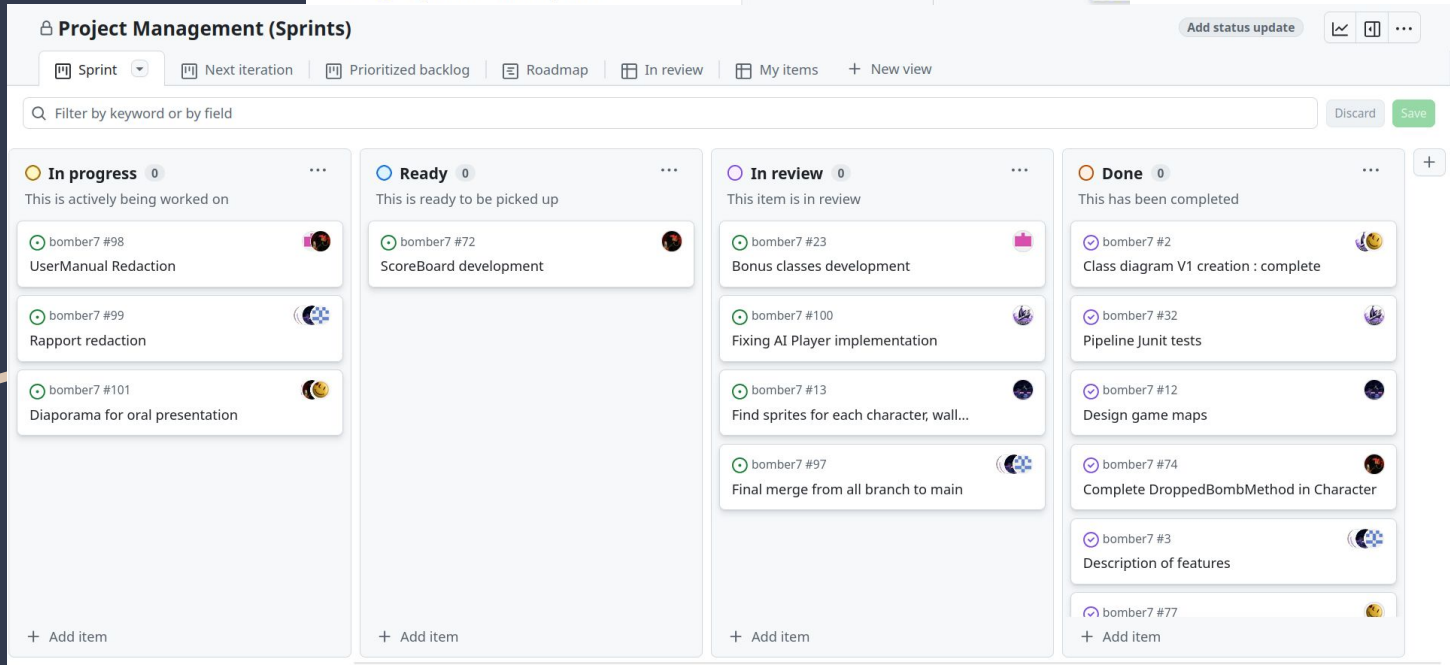
V3



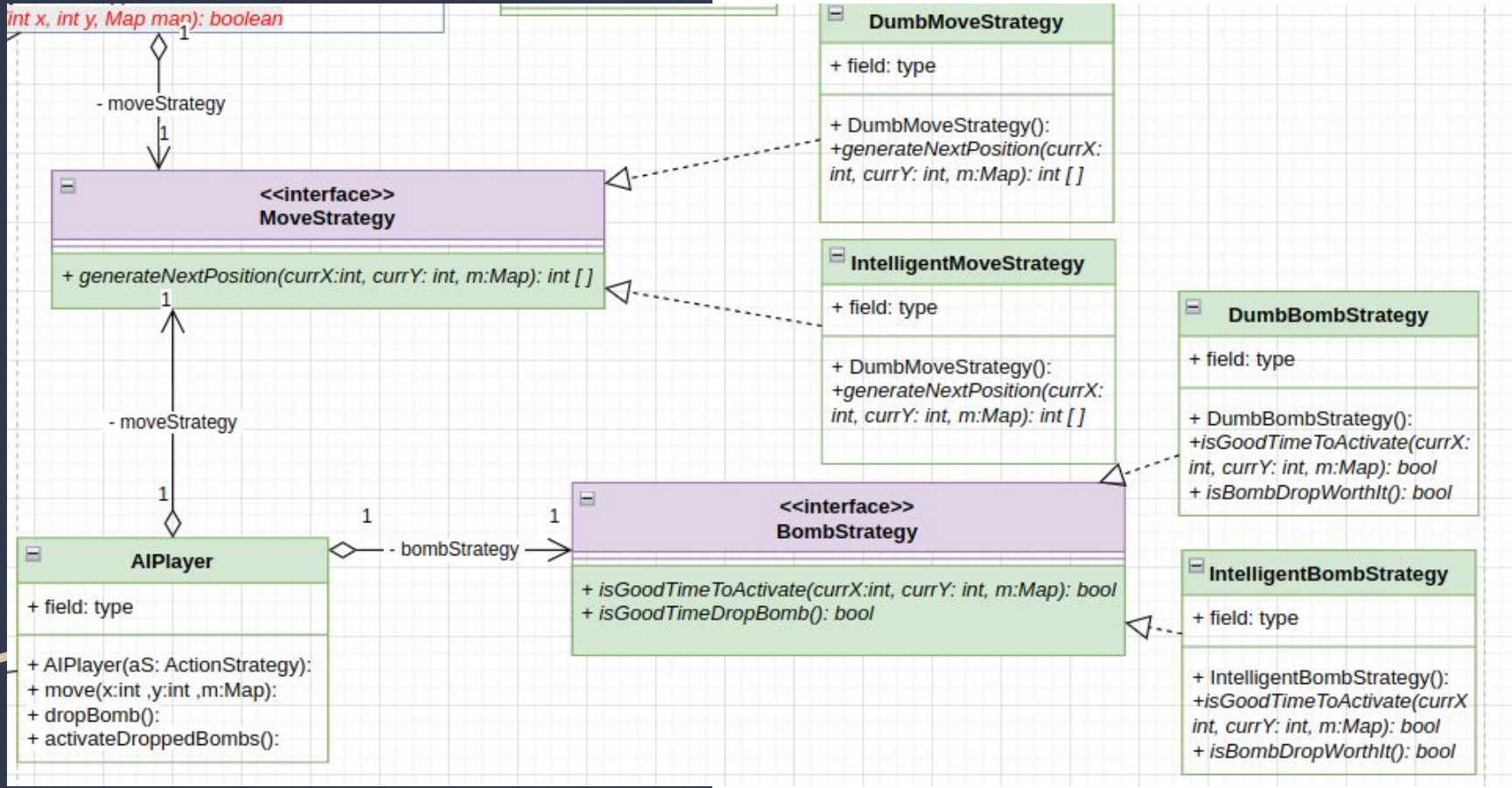
Partie 3 :
Gestion de projet

A. Méthodes agiles



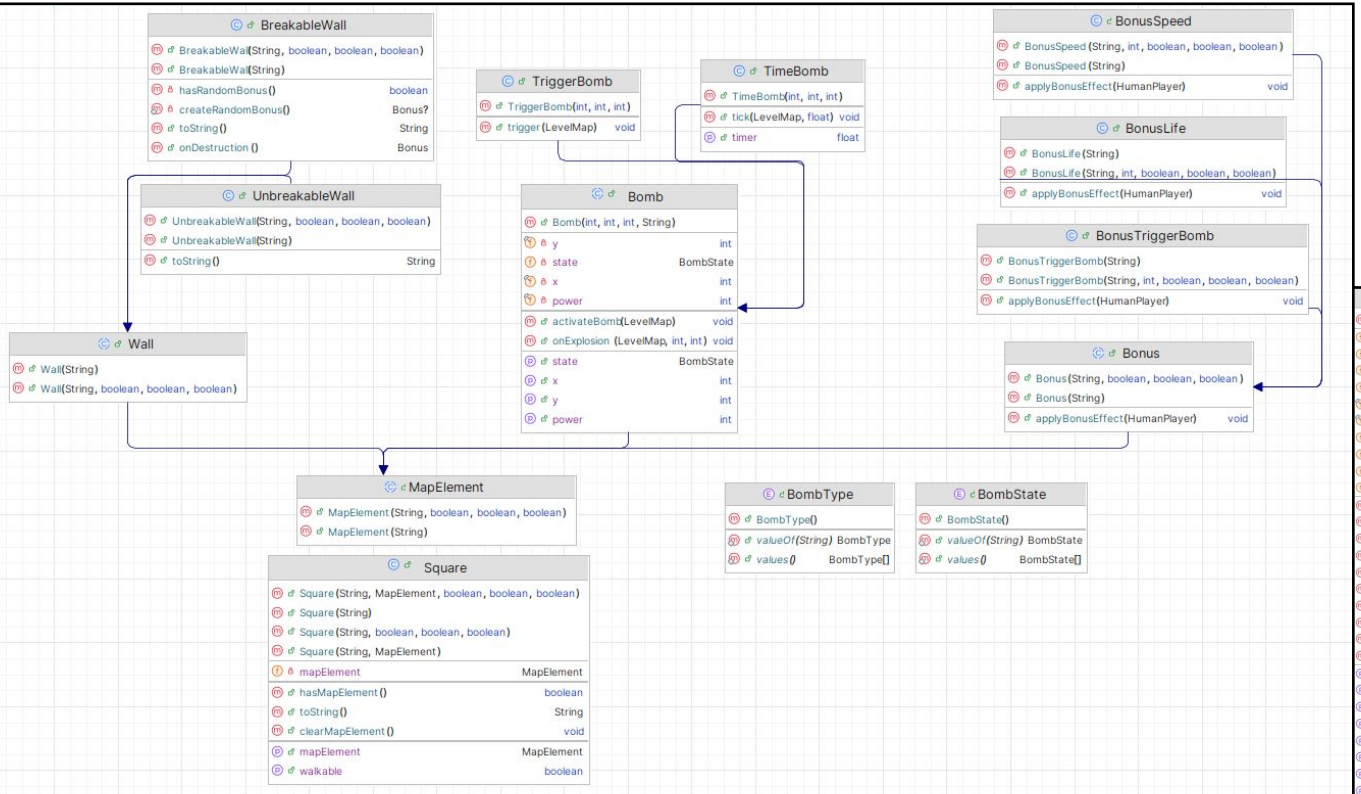


Patron Stratégie :



Partie 2 : Présentation Technique

B. Architecture UML : Modèle



Les classes importantes :

- Character : Player, …
- Wall : (un)breakable
- Bomb : Time, Trigger
- Bonus : Life, Speed, ..