Représentation de l'air de jeu:

Tableau 2 dimensions , utilisation de numpy, chaque case contenant:

- *: mur
- H: tool
- X: gardien
- S: Mac Gyver

Exemple:

Génération du jeu:

Le labyrinthe du jeu est généré de manière aléatoire a chaque lancement. L'algorithme DFS permet de générer un labyrinthe de manière aléatoire
Cette algorithme utilise la récursivité.

https://en.wikipedia.org/wiki/Depth-first_search

Pseudo code (gamecodeur.fr)

```
Cellule de départ
Tant que l'on a pas visité ttes  les cellules du labyrinthe
  Explorer
     Si il y au moins  une cellule adjacente non visité
        Choisir une direction au hasard vers la cellule non visitée
        Casser le mur (digwall)
        Explorer depuis le nouvelle cellule
   Sinon reculer
```

Nombre impair de case, condition'd'arret de l'algo nblig * nbcol / 2

L'algorithme génère un arbre, les différentes branches de cette arbre sont sauvegardées pour le placement de Mac Gyver et des tools.

Placement des éléments:

Le gardien est placé est placé à la racine de l'arbre. Sur la première branche issue de la racine on place Mac Gaver, non le place sur la dernière
Cellule.

Les tools sont positionnés sur les branches issues de cette première branche.

Mac Gyver est placé sur la dernière case de la première branche

Organisation des classes:

- Personnage
- Board (récursivité)
- Manager_base/Manager (Héritage)

Difficultés rencontrés:

Utilisation de numpy mélange entre les lignes et les colonnes
Placement des tools, pour éviter de croiser le Gardien pour la récupération d'un outil
Pygame sur Mac avec version python 2.7 KO.