DungeonMaster

Réalisation d'un jeu-vidéo en utilisant la POO

Daniel GIRARD

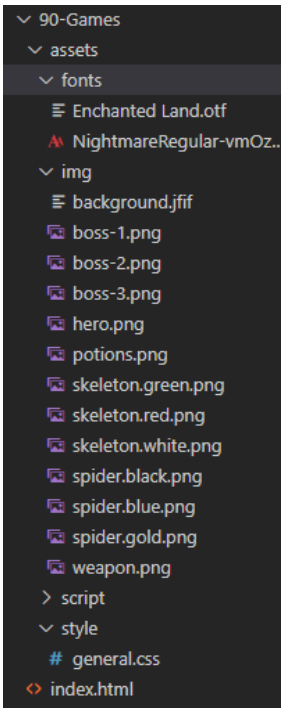
DungeonMaster

Afin de mettre en pratique l'ensemble des éléments vus cette année, je vous propose de réaliser un jeu-vidéo de type RPG. Dans ce jeu, vous incarnez une héroïne devant affronter 3 types de monstres différents afin d'aller le plus loin possible dans le donjon.

Ce projet sera réalisé en groupe de 3 personnes maximum afin de permettre un rendu le plus finalisé possible. Le but est de mettre en pratique les connaissances sur la POO vu en cours. Vous pouvez vous aider des cours, des corrections mais évitez d'utiliser des technologies trouvables sur le net que nous n'avons pas vu ensemble.

Le template du projet est fourni avec l'ensemble des éléments dont vous pourriez avoir besoin qui sont identifiés grâce à des **Id**. Vous devez utiliser uniquement le javascript natif avec vos connaissances en POO et en écriture modulaire. Aucune librairie ou Framework n'est tolérée.

Le travail demandé est suffisamment conséquent pour que chaque membre du groupe soit en mesure d'expérimenter chacun des problématiques dont nous avons discuté en cours. N'oubliez pas de décomposer au maximum les difficultés que vous rencontrez et n'essayez pas de tout faire en même temps !



L'arborescence du projet est fournie et vous propose déjà un fichier de template (**index.html**), une modification du style (**general.css**) et des Sprite (images) permettant de figurer les différents ennemis et objets dont vous pourriez avoir besoin.

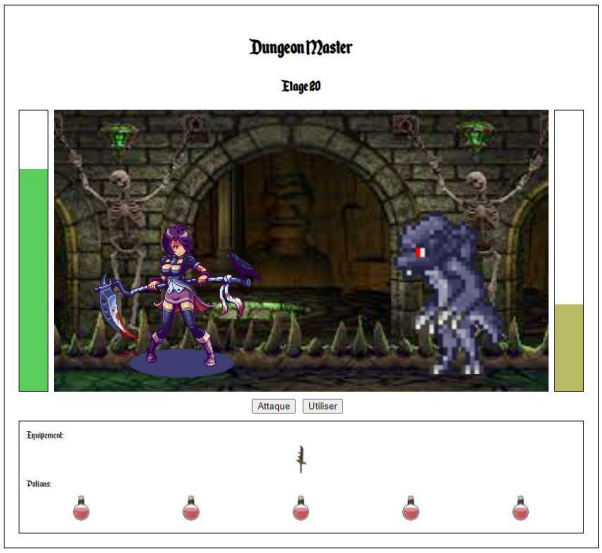
Vous êtes totalement libre de modifier le squelette mis à votre disposition et il en sera tenu compte dans la correction.

Pour ce projet, vous devrez prévoir l'affichage de 3 types d'ennemis différents :

- Une premier créature courante disponible en 3 couleurs (ici les araignées)

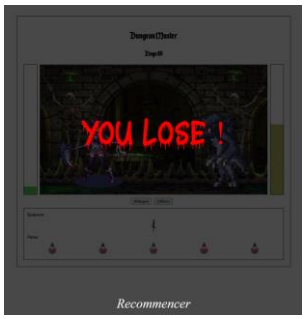
- Une deuxième qui n'arrivera qu'une fois tous les 5 étages disponible aussi en 3 couleurs (ici les squelettes)

- Une créature de type « boss » qui interviendra tous les 10 étages et qui sera aussi en 3 formes différentes



Au niveau de l'interface :

- L'étage est mis à jour à chaque passage au niveau suivant
- L'écran principal contiendra votre héros à gauche et l'ennemi qui sera mis à jour à chaque niveau à droite
- De chaque côté de l'écran principal, figure la barre des points de vie
    - A gauche de notre héroïne
    - A droite de la créature à abattre
- En dessous, figurera des boutons vous permettant d'effectuer les différentes actions de votre personnage :
    - Attaquer : pour taper sur l'ennemi – visible uniquement quand il y a un ennemi
    - Utiliser : pour utiliser une potions rendant des points de vie – visible uniquement quand vous possédez des potions
    - Suivant : qui permet de passer à l'étage d'après – visible uniquement quand l'ennemi est mort
- Un emplacement pour votre inventaire
    - La première partie contiendra les armes que vous obtiendrez sur les monstres (maximum 4 armes)
    - La seconde contiendra les potions que vous obtiendrez aussi sur les créatures (maximum 10 potions)
- Enfin, si vous perdez un écran apparaît (#lose dans le CSS) afin de rechargez la page et de recommencer une partie



Le principe de fonctionnement :

- Lors du démarrage d'une partie :
    - Je pense à indiquer l'étage
    - J'initialise mon personnage
    - Selon l'étage j'affiche dans une couleur aléatoire le monstre correspondant
    - J'active les événements sur les boutons qui doivent être visible

- A chaque clique sur le bouton attaquer :
    - Je récupère l'attaque du personnage et je la retranche au point de vie du monstre – sa défense
    - J'enchaine directement sur la riposte du monstre en réalisant la même action à l'envers
    - Je permets l'utilisation de potions afin de récupérer des points de vie lors du clique sur le bouton « utiliser »

- Quand le monstre meurt :
    - De manière aléatoire je génère un taux de récupération d'une potion ou d'une arme. Les taux étant différents selon le type de monstre
    - Je rends visible le bouton permettant de passer à l'étage suivant

- Au 10ème étage (5ème étage boss intermédiaire) :
    - J'affiche un boss
    - Ce boss possède des chances de récupération beaucoup plus élevé que les monstres de bases

- Après chaque boss de palier (étage multiple de 10) :
    - J'applique un coefficient afin de rendre l'attaque des prochains monstres et leur nombre de point de vie plus importants

Consignes :

1- Création d'une classe gérant le héro

Cette classe contiendra l'ensemble des caractéristiques de votre héros avec les propriétés et les méthodes suivantes (Accesseurs et Mutateurs non comptés) :

- Nom
- Prénom
- Force
- Dégâts
- Défense
- Point de vie
- Sac de potions
- Sac d'armes
- Méthode d'attaque prenant en paramètre la cible et qui renvoie **true** si la cible est vaincue sinon qui mets à jour les points de vie de la cible
- Méthode qui permet de récupérer automatiquement de l'équipement. Elle prend en paramètre le nom de l'équipement et doit vérifier si le nombre de potions (10) et d'équipement (4) maximum n'est pas atteint. Elle doit aussi automatiquement ajouter dans le sac correspondant du personnage l'objet et dans le cas d'une arme ajouter 4 aux dégâts de base du personnage
- Méthode d'utilisation de la potion : elle doit permettre de retrancher une potion et d'augmenter la vie du personnage de 10 sans dépasser le nombre de point de vie maximum

Afin de garantir une part d'aléatoire, lors de l'initialisation de mon personnage la force (3 – 10) et la défense (1 – 5) sont générées de manière aléatoire. Les dégâts provoqués sont égaux au produit de la force par les dégâts de base (3).

Pour simplifier la gestion, je vous encourage à utiliser des variables statiques vous permettant de sauvegarder au sein de votre classe les éléments qui serviront vos calculs lors de l'initialisation. On peut ainsi imaginer les propriétés static suivantes (pour les utiliser nomDeLaClasse.VariableStatic):

- Dégâts = 3
- MinForce = 3
- MaxForce = 10
- MinDef = 1
- MaxDef = 5

2- Création des monstres

En premier, je vous invite à créer une classe mère **Monster** qui regroupera l'ensemble des méthodes et propriétés communes à toutes les créatures. Elle sera composée des propriétés et des méthodes suivantes (sans compter Accesseur et Mutateur) :

- Nom
- Attaque
- Défense
- Point de vie
- Images
- Target
- Méthode d'attaque qui prend en paramètre la cible, qui mets à jour les points de vie de la cible en diminuant l'attaque du monstre par la défense de la cible. Elle renvoie **true** si sa cible est morte
- Méthode d'ajout dans le dom : qui est vide et sera construire dans les classes filles
- Méthode de suppression du dom : pareil qu'au-dessus
- Méthode permettant de lâcher une potion / arme : pareil qu'au-dessus
- Aucune démarche particulière sur l'initialisation (constructor)

Vous devrez ensuite créer les classes filles, une pour chaque type de créature qui hériteront à chaque fois de la classe mère. Dans chacune des classes filles vous devrez définir de manière spécifique les méthodes. Je vous invite une fois de plus, afin de faciliter l'initialisation de vos créatures à ajouter pour chacune d'entre elles des variables static permettant la gestion :

- linkToImg : indiquant le dossier où se trouve les images
- variousColor : un tableau regroupant les différents noms de créature dans les différents coloris
- La valeur maximum pour générer un nombre aléatoire permettant de savoir si la créature posséder une potion ou une arme

En effet, si vous mettez le chiffre 5 dans cette dernière variable statique, il vous suffira de générer un nombre aléatoire entre 1 et 5. Puis, si le résultat est 1 de dire que le monstre possédait cet équipement. On aura ici 1/5 d'obtenir un équipement.

3- Création des fonctions du jeux ou d'une classe les regroupant

Ce troisième point vous permettra de développer les fonctions vous permettant de réaliser une partie complète jusqu'au décès de votre personnage.

Si vous manquez de temps ou de pratique, l'ensemble de ces informations pourra donner lieu à un affichage console qui vous donnera moins de points qu'un affichage dans le template fourni.

L'objectif de cet examen est la réalisation de classes cohérente et utilisable dans cette troisième partie. Cette partie sera donc moins noté que les précédentes.

Si vous ne créez pas de classe, je vous invite à exporter chacune des fonctions que vous allez créez ici afin de bien séparer les déclarations de leurs utilisations.

Dans la correction que j'ai préparée, j'ai construit 4 fonctions :

- La première prend en paramètre le container et le titre contenant l'étage et me permet d'initialiser une partie et de retourner dans un objet tous les éléments dont j'ai besoin dans mes classes à savoir :
    - L'étage
    - Le héro
    - Le monstre
    - Les 3 boutons (attaque, utiliser, suivant)
    - Les sacs d'équipements (potions et arme)
    - Le container (où j'ajoute mes monstres)
    - Les points de vie de base de chaque créature et de mon héros (afin de calculer le pourcentage de vie perdue)

- La seconde prend en paramètre l'objet Game que j'ai initialiser au-dessus et permet la gestion de l'affichage et de l'obtention des récompenses lorsqu'une monstre meurt

- La troisième prend en paramètre l'objet Game et le type (monstre ou héros) afin de mettre à jour l'affichage de la barre de vie

- La dernière prend en paramètre l'objet Game et me permet de programmer ce qu'il se passe quand je change d'étage :

    - Augmentation du numéro d'étage
    - L'affichage du bouton attaquer (disparu lorsque le monstre meurt)
    - L'augmentation des points de vie grâce à un coefficient d'augmentation défini par calcul (Etage / 10)
    - Quel monstre je dois afficher et où en fonction de l'étage
    - Remise à 0 des points de vie du monstre

4- Le fichier de jeu

Dans ce fichier, je vais utiliser l'ensemble des éléments que j'ai défini avant en réaction avec les actions que l'utilisateur va effectuer.

Je vais aussi initialiser mon super objet Game qui regroupera l'ensemble des informations nécessaires au bon fonctionnement de mon jeu.

Dans ce script, j'ai géré ce qu'il se passe quand :

- Je charge la page : initialisation de la partie

- Je clique sur le bouton attaquer : gestion des dégâts, de la mort du monstre ou de la mort du héros

- Je clique sur le bouton utilisation : gestion des points de vie de l'utilisateur (maximum 100 et barre de vie à modifier) et de son sac (suppression d'une potion de mon héro affichage compris)

- Je clique sur le bouton suivant : j'appelle ma fonction calculant et préparant le nouvel étage

Barème

Consignes	Notes
Création d'une classe gérant le héros	/10
Création d'une classe mère gérant les monstres	/10
Création d'une classe gérant le monstre simple	/5
Création d'une classe gérant le boss intermédiaire (tous les 5 niveaux)	/5
Création d'une classe gérant le boss (tous les 10 niveaux)	/7
Création des fonctions nécessaires à la réalisation d'une partie	/20
Partie fonctionnelle	/5
Lisibilité / Structure du code / Arborescence	/5
Apparence globale	/5

Les groupes constituées seront indiqués avec le lien vers le dépôt du projet dans le tableau mis à votre disposition.

Ce barème est donné à titre indicatif, il peut évoluer selon le niveau moyen de la classe.

Le détail que je vous ai fourni pour la construction des classes ou des fonctions peut varier selon les solutions que vous allez trouver. Certaines propriétés peuvent apparaitre pour faciliter la gestion du DOM dans vos classes par exemple. Si c'est le cas, et qu'une telle propriété n'est utilisé qu'au sein de la classe, vous n'êtes pas obligé de créer les accesseurs et mutateurs correspondant. Attention les variables statiques n'ont pas de getters / setters.