Projet - BackPack Hero

Le but de ce projet est de réaliser une version simplifiée du jeu Backpack Hero. Il s'agit d'un jeu pour un seul joueur : le héros doit explorer un donjon peuplé de monstres et de trésors, et ne pourra s'aider que du matériel que l'on transporte dans son sac à dos. C'est un jeu de type *rogue-like* dont une des composantes principales consiste en du placement de tuiles qui interagiront ; la nature de ces interactions sera décrite dans les sections ci-dessous.

Sommaire

Attention :

Le calendrier détaillé du projet (dates de rendus, soutenances, etc.) n'est pas encore finalisé.

Les dates précises seront communiquées ultérieurement, ainsi que d'éventuelles consignes supplémentaires.

Conseils :

— Lisez l'ensemble de l'énoncé avant même de commencer quoi que ce soit !

— Ce jeu existe sur de multiples plates-formes ; avoir joué au jeu permet sans doute de mieux comprendre ses mécanismes, mais ce n'est pas indispensable. Si vous n'en n'avez jamais entendu parler, vous pouvez facilement trouver des vidéos de parties sur YouTube pour vous faire une idée du mécanisme de jeu.

— Le but de ce projet est de nous montrer que vous savez programmer "objet". Vous serez donc pénalisés si vous n'exploitez pas suffisamment les notions vues en cours.

— Lisez bien les règles du jeu et prenez bien le temps de réfléchir à la meilleure façon de mettre en place tel ou tel mécanisme du jeu avant de commencer à coder. Si vous commencez à coder avant d'avoir suffisamment réfléchi, vous serez, en pratique, obligés de revenir en arrière, ce qui vous demandera beaucoup plus de temps et d'efforts.

— Le sujet est évolutif : respectez bien les phases de réalisation, mais gardez à l'esprit ce que vous devrez faire dans les phases suivantes lorsque vous faites des choix d'implantation !

Le jeu :

Ce sujet fournit une présentation succincte du jeu. Vous trouverez des informations beaucoup plus complètes sur le Wiki du jeu : https://backpack-hero.fandom.com/ . Attention : si les informations données dans le Wiki sont en contradiction avec le sujet, c'est le sujet qui prévaut.

Le jeu commence dans un donjon dont le héros souhaite s'échapper. Pour l'aider dans sa quête de liberté, le héros peut compter sur son fidèle sac à dos, dans lequel il pourra entreposer des armes, boucliers, baguettes magiques, et toute sorte de matériel. Mais le sac a une capacité limitée, et il faudra donc choisir quels éléments on souhaite transporter, et où les placer dans le sac.

Lorsque le héros rencontre des ennemis, un combat au tour par tour commence. Au début de chaque tour, le héros dispose de trois points d'énergie, qu'il peut utiliser comme bon lui semble pour se protéger ou attaquer ses ennemis. De leur côté, les ennemis ne sont pas de très bons comédiens, et laissent donc apercevoir quelle action (se protéger, attaquer le héros, lui lancer une malédiction, ...) ils ont l'intention d'effectuer quand leur tour viendra. Une fois tous les ennemis vaincus, le héros gagnera de l'expérience, des trésors que ses ennemis gardaient jalousement et, si son expérience lui permet de gagner un niveau, il obtiendra même un peu plus de place dans son sac à dos. Le héros pourra alors de nouveau consulter la carte du donjon et choisir quelle est la prochaine pièce qu'il souhaite explorer ; il peut même décider de passer une porte de sortie, auquel cas il découvrira un nouvel étage du donjon.

Carte du niveau

Le joueur peut se déplacer au sein de l'étage du donjon sur lequel il se trouve. Chaque étage se compose d'un ensemble de salles, et chaque salle peut contenir :

- rien : c'est juste un couloir ;
- un trésor ;
- une ou plusieurs portes de sortie de l'étage ;
- un ou plusieurs ennemis ;
- un marchand, qui pourra acheter et vendre du matériel ;
- un guérisseur, qui nous apportera vie et santé moyennant quelques pièces d'or ;
- une rencontre surprenante, dont on ne connaîtra la teneur qu'arrivé sur place ;
- une grille, que l'on ne pourra ouvrir que si on y sacrifie une clé ; débloquer une grille nous permet de traverser la salle concernée, et derrière celle-ci, on trouvera usuellement une sortie, un marchand, un trésor, ...

En outre, chaque étage du donjon a les caractéristiques suivantes :

— les salles de l'étage occupent les cases d'un rectangle de taille 5×11 ;
— l'étage est connexe : deux salles de l'étage sont toujours reliées par au moins un chemin ;
— quand on clique sur la carte, on voit quelles salles contiennent quels éléments ;
— quand on clique sur une salle de la carte, et s'il existe un chemin permettant d'aller à cette salle en ne passant que par des couloirs, le héros se rend à sa destination ; sinon, il ne bouge pas ;
— quand on clique sur une porte de sortie, on se retrouve dans un nouvel étage, et on ne pourra plus jamais revenir à l'étage précédent.

Combats

Lorsque le héros rencontre un ou plusieurs ennemis, il les affronte, selon un mécanisme de combat au tour par tour. C'est d'abord au tour du héros, puis au tour des ennemis, puis au tour du héros, et ainsi de suite.

Au début du combat, les ennemis sont en pleine santé. Le héros, quant à lui, démarre avec autant de points de vie qu'il en avait à l'issue de son dernier combat (sauf, par exemple, s'il est allé voir un guérisseur entre-temps). Les pricipaux moyens dont dispose le héros pour gagner un combat sont de :

— diminuer le nombre de points de vie d'un ou plusieurs ennemis ;
— augmenter son niveau de protection contre les attaques qu'il subira au prochain tour ;
— s'appliquer des *effets de combat* positifs, ou appliquer à un ou plusieurs ennemis des effets de combat négatifs, ou encore se purger des effets de combat négatifs qu'il a reçus ;
— rendre ses autres équipements plus efficaces.

Pour ce faire, le héros doit utiliser ses équipements, qui pourront être des épées, des boucliers, des baguettes magiques, des boîtes de conserve, … Cependant, nombre de ces équipements ont un *coût d'activation*, qui nécessite d'être payé pour utiliser l'équipement. Ce coût peut consister en

— des points d'énergie : au début de chaque tour, le héros dispose de 3 points d'énergie ;
— des points de mana : certains équipements confèrent au héros un ou plusieurs points de mana, et se rechargent au début de chaque combat ;

— de l'or.

Les ennemis du héros fonctionnent de manière légèrement différente : au lieu d'utiliser des équipements, ils annoncent simplement, avant leur tour (c'est-à-dire durant le tour du héros) quelle action ils vont accomplir. Cette action peut consister à infliger des points de dégât au héros, à augmenter leur niveau de protection, à appliquer ou annuler des effets de combat, ... Un même ennemi peut effectuer plusieurs actions lors d'un même tour, tant qu'il les annonce toutes ; il peut aussi infliger au héros des *malédictions*, qui seront décrites dans deux sections.

Enfin, après chaque combat remporté, le héros reçoit des trésors ainsi que de l'expérience. Plus précisément, chaque ennemi tombé pendant la bataille lui rapporte un certain nombre de points d'expérience, tandis que les trésors reçus sont tirés au hasard. Accumuler de l'expérience permet au héros de monter en niveau, et avec chaque nouveau niveau, de gagner de la place dans son sac à dos.

Exemple de combat

Fatigué par son dernier combat, le héros dispose de 30 points de vie sur ses 40 points de vie initiaux. Il affronte un rat-loup doté de 20 points de vie. Le héros dispose dans son sac des équipements suivants :

— une épée en bois, dont chaque utilisation coûte 1 point d'énergie et enlève 6 points de vie à l'ennemi ciblé ;
— un bouclier en bois, dont chaque utilisation coûte 1 point d'énergie et augmente le niveau de protection du héros de 7 points *au prochain tour uniquement* ;
— une pierre de santé, dont l'utilisation est gratuite et rapporte 1 point de vie au héros (sans dépasser le nombre de points de vie maximal du héros) ; cette pierre de santé peut être utilisée au plus une fois par tour ;
— du thon en boîte, dont l'utilisation est gratuite et rapporte 1 point d'énergie au héros, mais qui sera détruit dès sa première utilisation (puisque le thon aura été mangé).

En face, le rat-loup indique qu'il s'apprête à attaquer le héros et à lui infliger 12 points de dégâts quand viendra son tour.

Le héros décide alors d'utiliser sa pierre de santé, sa boîte de thon, puis son bouclier en bois (2 fois) et son épée en bois (2 fois). À l'issue de ce premier tour de combat, le héros dispose de 31 points de vie, d'un niveau de protection est de 14 points (qui sera valable uniquement pendant le prochain tour du rat-loup), et le rat-loup n'a plus que 14 points de vie. Comme annoncé, le rat-loup alors tente d'infliger 12 points de dégât au

héros ; comme le héros avait 14 points de protection, il ne perd aucun point de vie, mais sa protection est désormais de 2 points seulement.

C'est maintenant le second tour du héros, dont le niveau de protection est retombé à zéro à la fin du tour du rat-loup. Cette fois-ci, le rat-loup indique qu'il s'apprête à infliger 16 points de dégâts au héros. Mais le héros décide d'utiliser 1 fois sa pierre de santé puis 3 fois son épée ; il inflige donc 18 points de dégât au rat-loup, qui meurt dans d'atroces souffrances. Le héros a gagné, et il dispose même de 32 points de vie, soit davantage qu'au début du combat ! Sa victoire et la mort du rat-loup lui valent de recevoir 6 points d'expérience, et plusieurs équipements, qu'il pourra décider d'insérer dans son sac s'il le souhaite.

Sac à dos et équipements

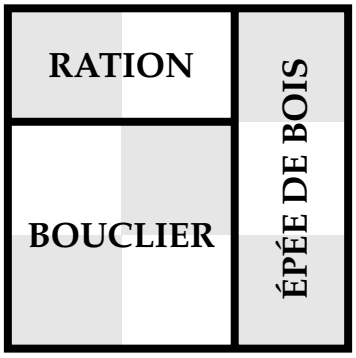
Le sac à dos consiste en un certain nombre de cases contiguës, toutes comprises dans un rectangle 5×7, c'est-à-dire formé de 5 lignes et 7 colonnes. Initialement, le sac à dos consiste en 9 cases, qui forment un grand carré de 3 lignes et 3 colonnes. Puis, à chaque nouveau niveau du héros, le sac gagne 3 ou 4 cases, dont le héros choisit l'emplacement ; on pourra ainsi privilégier des sacs plutôt allongés, très hauts, ou encore à peu près carrés.

Dans ce sac à dos, on pourra ranger divers types d'équipements. Chaque pièce d'équipement a une forme spécifique, et sera donc plus ou moins facile à ranger dans le sac à dos :

— des armes de mêlée : épées, dagues, gourdins, … ;
— des armes à distance : fléchettes, arcs et flèches, … ;
— des objets magiques : baguettes, anneaux, … ;
— des armures, boucliers, casques ou bottes ;
— de l'or : il s'agit de la monnaie d'échange pour acquérir les services d'un marchand ou d'un guérisseur ; stocker de l'or requiert d'y consacrer une case du sac à dos, dans laquelle on pourra mettre autant d'or que désiré ;
— des pierres de mana ; chaque pierre de mana occupe une case, mais contrairement à l'or, deux pierres de mana différentes doivent occuper deux cases différentes ;
— de tout plein d'objets divers ;
— de malédictions : il s'agit d'équipements nocifs que les ennemis nous ont forcés à mettre dans notre sac à dos, et dont on se débarrassera dès que possible.

Afin d'insérer plus facilement un équipement donné, on peut également le translater, et on peut également le tourner de 90° dans le sens des aiguilles d'une montre (sauf s'il s'agit d'une malédiction : on ne peut pas faire tourner les malédictions). C'est ainsi

que l'on peut faire tenir dans notre sac à dos initial une épée de bois de taille 1×3, un bouclier de taille 2×2 et une ration de survie de taille 1×2, en procédant comme ceci :



Par contre, si on n'a pas la place de mettre tous les équipements souhaités dans le sac à dos, il faudra se résoudre à abandonner certains d'entre eux. Ainsi, un équipement n'est pas autorisé à dépasser du sac à dos ou à utiliser une case en commun avec un autre équipement.

En outre, certains équipements peuvent interagir entre eux. Par exemple, toute utilisation d'une arme occupant une case adjacente à la *gemme de cœur* rapporte 1 point de vie au héros ; au contraire, la hachette est une arme dont l'utilisation enlève 4 points de vie à l'ennemi ciblé, sauf si le héros porte une armure, auquel cas la hachette enlève simplement 1 point de vie à l'ennemi ciblé.

Enfin, chaque objet a un niveau de rareté : plus un objet est rare, moins on a de chances de le recevoir à la fin d'un combat (incroyable !) et plus il vaudra cher. En particulier, vendre un objet rare dont on n'a pas besoin est parfois un moyen efficace d'obtenir de l'argent, et donc d'acquérir un objet cher dont on avait besoin.

Malédictions

Une malédiction est un équipement spécial que nous inflige un ennemi. Quand on la reçoit, on a le choix entre prendre des points de dégâts ou insérer la malédiction dans notre sac à dos ; mais attention : le $k^{\text{ème}}$ refus nous vaudra de recevoir k points de dégâts !

Quand le héros insère une malédiction dans son sac à dos, il peut la translater mais pas la tourner. En outre, il devra jeter tous les équipements qui utilisaient une case sur laquelle il pose la malédiction, et il n'a pas le droit de réorganiser son sac. Par exemple, si on insère dans notre sac à dos (obtenu à la section précédente) une malédiction de la forme

on a deux manières de le placer ; la première nous oblige à jeter tous nos équipements ; la seconde nous autorise à garder notre ration ; comme il n'est pas possible de tourner la malédiction de $90°$, la troisième manière, qui nous aurait permis de garder notre épée, est en fait invalide !

Méthode n°1

Méthode n°2

Méthode n°3
(invalide)

Enfin, il est possible de se débarrasser d'une malédiction après l'avoir insérée dans son sac à dos ; cela permet de libérer de la place dans le sac, même si on ne pourra pas récupérer les objets que l'on a laissés tomber pour insérer la malédiction. Cependant, en général, quand on se débarrasse d'une malédiction, on subit une pénalité jusqu'à la fin du prochain combat.

Mais en détail, ça marche comment ?

Comme indiqué en préambule, tous les détails sont disponibles sur le Wiki du jeu. Ces détails incluent notamment :

— une liste des divers équipements, incluant notamment leur forme, leur rareté, leur coût d'utilisation et leurs effets mais également les prix de vente et d'achat de ceux-ci, ainsi que les reliques que l'on peut recevoir après avoir battu un *boss* : https://backpack-hero.fandom.com/wiki/Atlas ;

— une liste des divers ennemis, incluant notamment leur nombre de points de vie initial, la puissance de leurs attaques (quand ils décident d'attaquer) et de leur niveau de protection (quand ils décident de se défendre), le nombre de points d'expérience que leur mort rapporte au héros, ainsi que les effets qu'ils pourront appliquer au héros (si l'effet est négatif) ou à eux-mêmes (si l'effet est positif) : https://backpack-hero.fandom.com/wiki/Bestiary ;
— une liste des diverses rencontres surprenantes que l'on pourrait faire : https://backpack-hero.fandom.com/wiki/Events ;
— une liste des divers effets, positifs ou négatifs : https://backpack-hero.fandom.com/wiki/Status_Effects .

Toutefois, le Wiki du jeu est parfois très incomplet ; par exemple, il ne précise pas quelle est la probabilité d'occurrence d'un objet *légendaire* après un combat, ni l'influence que peut avoir sur cette probabilité le fait de porter un anneau de chance. Dans ce cas, c'est à vous de décider des choix d'implémentation que vous devrez effectuer.

Interface graphique

Vous aurez besoin de réaliser une interface graphique simple. Attention ! Ce n'est pas la qualité de l'interface graphique qui est évaluée dans ce projet, mais vos compétences en conception et programmation objet.

— Vous **devez** utiliser la bibliothèque d'interface graphique `zen` fournie avec ce sujet (fichier `zen-6.0.jar`).

  Pour ajouter un jar à un projet sous Eclipse, il faut :

  — Rajouter un dossier `lib` dans le répertoire du projet et y placer le fichier `.jar`.
  — Dans Eclipse, faire un clic droit sur le fichier `.jar` et choisir `Build Path > Add to Build Path`.

— On vous fournit également un mini-exemple (très incomplet) de code utilisant cette bibliothèque et suivant un modèle de développement classique appelé MVC (Modèle-Vue-Contrôleur) dans votre code :

  — Une classe `SimpleGameData` est utilisée pour gérer les données du jeu (le modèle) ainsi que toutes les actions possibles, suivant les règles du jeu.
  — Une interface `GameView` implémentée par le record `SimpleGameView` permet de gérer l'affichage graphique (la vue).

— Le contrôleur implémente la boucle de jeu et la gestion des événements utilisateur : clics, touches, ...

Le programme à réaliser, en trois étapes

Phase 1 : La base

À la fin de la phase 1, on doit disposer des bases du jeu, avec les fonctionnalités suivantes correctement implémentées :

- Le héros débute la partie avec 40 points de vie ;
- Les équipements n'interagissent pas ;
- Les seuls ennemis sont des petits rats-loups et des rats-loups : leurs seules actions possibles sont d'attaquer le héros ou d'augmenter leur propre niveau de protection ; ces actions sont choisies aléatoirement à chaque tour ;
- On doit proposer au moins :
  - une arme de mêlée ;
  - une arme à distance (fléchette, ou bien un arc et des flèches) ;
  - une armure ;
  - un bouclier ;
  - un objet magique requérant de la mana pour fonctionner ;
  - des pierres de mana ;
  - de l'or, bien sûr !
- Le sac à dos est constitué de 15 cases, réparties en 3 lignes de 5 colonnes chacune, et il ne changera pas de forme ;
- Il n'y a pas encore d'*effets de combat* : les seules choses que l'on peut faire sont d'attaquer l'ennemi ou d'augmenter son propre niveau de protection ;
- Le donjon est constitué de 3 étages, et on vous autorise (pour cette phase uniquement) à coder en dur le plan de chaque étage plutôt qu'à le générer aléatoirement ; le but du jeu est alors de sortir du 3$^{\text{ème}}$ étage du donjon ;
- Chaque étage contient
  - autant de couloirs que vous le souhaitez
  - trois salles peuplées d'ennemis,
  - une salle avec un marchand,
  - une salle avec un guérisseur,

  - deux salles contenant un trésor,
  - une porte de sortie vers l'étage supérieur.
- Puisque l'on vous a autorisé à coder en dur le plan de chaque étage, on vous autorise aussi (pour cette phase uniquement) à coder en dur la liste des équipements disponibles à l'issue de chaque combat et dans chaque trésor.

Phase 2 : Le jeu en version β

À la fin de la phase 2, on doit disposer d'une version β du jeu, que l'on pourrait proposer aux utilisateurs pour les intéresser et les pousser à acheter le jeu complet :

- Chaque ennemi nous fait gagner de l'expérience quand il meurt, et accumuler suffisamment d'expérience nous fait gagner un niveau ainsi que 3 ou 4 cases dans notre sac à dos ;
- Les équipements peuvent désormais interagir ;
- Il y a maintenant des effets de combat ainsi que des malédictions ;
- Les équipements disponibles après chaque combat gagné ou dans chaque trésor sont choisis au hasard parmi les objets que votre donjon est susceptible de contenir.

Phase 3 : Le jeu complet

Lorsque la phase 2 est complètement finie, vous devez ajouter les fonctionnalités suivantes, par ordre de priorité décroissante (la première fonctionnalité est la plus importante, car elle nous assure que vous avez vérifié la jouabilité de votre jeu) :

- On doit pouvoir gérer un score (selon une recette que vous créerez vous-mêmes, basée sur le nombre de points de vie maximal du héros ainsi que la somme des prix de ses équipements) et d'un Hall of Fame ; on exigera de vous, lors de la soutenance, que vous nous montriez votre Hall of Fame avec vos trois meilleures parties ;
- Parmi les ennemis proposés, on doit désormais trouver le sorcier-grenouille, l'ombre vivante et la reine des abeilles ;
- La carte de chaque étage du donjon est générée aléatoirement, sous réserve que l'on puisse toujours se déplacer de n'importe quelle salle vers n'importe quelle autre (sauf qu'il peut y avoir des ennemis ou une grille fermée à clé sur le chemin).

Consignes de rendu

Attention :

Le calendrier détaillé du projet (dates de rendus, soutenances, etc.) n'est pas encore finalisé.

Les dates précises seront communiquées ultérieurement, ainsi que d'éventuelles consignes supplémentaires.

— Ce projet est à faire en **binôme** (c'est-à-dire exactement **deux** personnes) d'un **même groupe de TP**. Vous aurez jusqu'au **jeudi 13 novembre** à **23:59** pour indiquer, sur e-learning, le nom de votre partenaire de projet. Si vous avez des difficultés à trouver un partenaire de projet, approchez-vous sans attendre de votre chargé de TP, pour qu'il vous aide à ce propos.

Si vous n'avez pas indiqué votre choix de partenaire à la date demandée

- et si vous n'avez pas non plus participé au TP noté du **jeudi 13 novembre**, vous serez d'office considéré comme défaillant au projet (donc à l'année de L3) ;
- et si vous avez participé au TP noté du **jeudi 13 novembre**, un partenaire de projet vous sera attribué d'office, de manière arbitraire.

Il vous est **très vivement recommandé** de ne pas attendre le **jeudi 13 novembre** pour trouver un partenaire de projet.

— Une soutenance β sera organisée **fin novembre ou début décembre**. Pendant cette soutenance, vous ferez une démonstration sur une machine des salles de TP et serez interrogés sur le projet. Le but de cette soutenance est, entre autres, de permettre à votre enseignant de vous encourager si vous êtes sur la bonne voie, et de vous remettre dans le droit chemin sinon.

Il est attendu, à cette étape du projet, que la phase 1 soit terminée, et la phase 2 sérieusement entamée, si ce n'est achevée ; en effet, la phase 2 est usuellement l'une des plus difficiles pour les étudiant(e)s, mais si vous n'avez rien à nous montrer, nous n'aurons rien à vous suggérer.

Attention :

La soutenance β n'est pas notée : elle est là pour vous aider, profitez-en.

— Un premier travail intermédiaire, qui sera présenté lors de la soutenance β, doit être déposé sur e-learning **fin novembre ou début décembre**.

— Une soutenance finale sera organisée **courant janvier** Pendant cette soutenance,

vous ferez une démonstration sur une machine des salles de TP et serez interrogés sur le projet. Le but de cette soutenance est, pour vous, de nous montrer ce dont vous êtes fiers et, pour nous, de vous indiquer quels étaient les points positifs et les points négatifs de votre projet.

Il est attendu, à cette étape du projet, que les phases 1 à 3 soient achevées.

Attention :

Lors de la soutenance finale, chaque membre du binôme sera interrogé pour expliquer des parties du code. En cas d'incapacité à répondre, on considérera que ce membre n'a pas participé au projet, et recevra de ce fait la note de 0. Toute justification dans la veine de "je ne m'en souviens plus" ne sera pas acceptée. N'hésitez pas à préparer votre soutenance et réviser votre code au préalable avec votre binôme.

Rendu intermédiaire

Votre rendu intermédiaire devra consister en une archive au format `zip` : tout `rar`, `tar.gz`, `7z` ou autre ne sera pas ouvert. Cette archive contiendra :

- un répertoire `src` contenant les sources du projet ;
- un répertoire `docs` contenant un manuel de l'utilisateur (`user.pdf`) et un manuel expliquant l'architecture que vous avez choisie (`dev.pdf`) ; le manuel `user.pdf` doit être lisible par Bosphore, 11 ans, qui connaît le *gameplay* de Backpack Hero et essaiera d'utiliser votre programme sans avoir lu d'autre document, ni même l'énoncé du projet ;
- un répertoire `docs/doc` contenant la javadoc, écrite **en anglais** ;
- un répertoire `lib` contenant les librairies dont dépend l'application ;
- un `jar` exécutable `BackpackHero.jar`, qui fonctionne avec la commande `java -jar BackpackHero.jar`.

Cette archive aura pour nom `Nom1_Nom2_BackpackHero.zip`, où les noms sont ceux des membres du binôme par ordre alphabétique. L'extraction de cette archive devra créer un répertoire intitulé `Nom1_Nom2_BackpackHero` et contenant tous les éléments demandés ci-dessus.

Rendu final

Votre rendu final devra consister en une archive au format `zip` : tout `rar`, `tar.gz`, `7z` ou autre ne sera pas ouvert. Cette archive contiendra :

— un répertoire `src` contenant les sources du projet ;
— un répertoire `docs` contenant un manuel de l'utilisateur (`user.pdf`) et un manuel expliquant l'architecture que vous avez choisie (`dev.pdf`) ; le manuel `user.pdf` doit être lisible par Bosphore, 11 ans, qui connaît qui connaît le *gameplay* de Backpack Hero et essaiera d'utiliser votre programme sans avoir lu d'autre document, ni même l'énoncé du projet ; le manuel `dev.pdf` doit notamment inclure une section dédiée aux améliorations et corrections apportées depuis la soutenance β ;
— un répertoire `classes`, vide dans l'archive, et qui contiendra les classes une fois compilées ;
— un répertoire `lib` contenant les librairies dont dépend l'application ;
— un `jar` exécutable `BackpackHero.jar`, qui fonctionne avec la commande `java -jar BackpackHero.jar`, et qui possède donc un fichier `manifest` adéquat ;
— un fichier `build.xml`, écrit à la main, qui permet de

  — compiler des sources (`target compile`) ;
  — créer le `jar` exécutable (`target jar`) ; il devra s'agir de la `target` par défaut ;
  — générer la javadoc, écrite **en anglais**, dans le répertoire `docs/doc` (`target javadoc`) ;
  — nettoyer le projet pour qu'il ne rete plus que les éléments demandés (`target clean`).

Cette archive aura pour nom `Nom1_Nom2_BackpackHero.zip`, où les noms sont ceux des membres du binôme par ordre alphabétique. L'extraction de cette archive devra créer un répertoire intitulé `Nom1_Nom2_BackpackHero` et contenant tous les éléments demandés ci-dessus.

Critères de notation

— Bosphore, 11 ans, doit pouvoir jouer à votre jeu et y prendre plaisir ;
— la propreté et la lisibilité du code auront un poids très important dans la note ;
— l'architecture que vous aurez définie (interfaces, classes, etc) devra être donnée dans les documents PDF et aura également un poids très important dans la note ; ainsi, votre code devra être modulable, de manière à ce qu'ajouter d'autres extensions (par exemple davantage de cartes, un niveau de plus, ...) soit aussi facile que possible ;
— votre code ne devra pas contenir de méthodes de plus de 20 lignes ;
— pas de duplication de code, et respect des principes de programmation objet ;

— pas de variable globale ;
— pas de code inutile ;
— présence des différents rapports et, par conséquent, orthographe correcte !
— prise en considération des remarques faites lors de la soutenance β pour le rendu final.

De manière générale, si tout est parfait d'après votre formulaire d'auto-évaluation, et si celui-ci reflète fidèlement la réalité de votre projet, votre note devrait être excellente.

Règles à respecter impérativement – Mort subite

Voici une liste de règles qu'il vous faudra respecter impérativement. Si vous ne respectez pas ne serait-ce qu'une seule de ces règles, et en fonction de la situation et de vos explications, la conséquence pour vous pourra aller du retrait de quelques points sur votre note à un statut défaillant pour le projet, donc pour la L3.

— Vous **devez** participer à la soutenance β ; si un seul des deux membres d'un binôme participe à la soutenance β, le membre absent sera considéré comme défaillant pour le projet.
— Vous **devez** déposer un premier travail **sur e-learning** avant la soutenance β.
— Vous **devez** participer à la soutenance finale ; si un seul des deux membres d'un binôme participe à la soutenance finale, le membre absent sera considéré comme défaillant pour le projet.
— Vous **devez** déposer votre version finale **sur e-learning** avant la soutenance finale.
— Dans les deux cas (version intermédiaire et version finale), votre code **doit** compiler.
— Vous **devez** inclure une javadoc écrite **en anglais** et des fichiers `user.pdf` et `dev.pdf` avec votre version finale.
— Votre archive **devra** avoir le bon nom, être une archive `.zip`, et produire un répertoire qui a le bon nom.
— Le projet ne **devra pas** utiliser ou inclure de librairie externe autre que celles indiquées dans le sujet.
— Le projet ne **devra pas** contenir de code copié-collé du net. La présence d'un tel code sera interprétée comme une tentative de tricherie, et s'accompagnera donc d'une convocation devant le conseil de discipline de l'IGM.
— Le projet **devra** gérer les entrées/sorties conformément au cours ; par exemple, il ne **devra surtout pas** utiliser `java.io.File`.

— Le projet ne **devra pas** contenir de champ avec une visibilité autre que `private`, et toute méthode de visibilité `public` devra commencer par vérifier que ses arguments sont raisonnables. Par exemple, si une fonction lance une `NullPointerException`, celle-ci doit être due à un appel à `Objects.requireNonNull` qui aura détecté que l'argument proposé était nul.

Références

1. **Ant Manual** pour la construction du fichier `build.xml` ;
2. **How to create an executable jar ?**
3. La **JavaDoc** ;
4. Les **entrées/sorties sur fichier** ;
5. La bibliothèque graphique **Zen**, ses **sources** et sa **documentation** ;
6. Un **exemple de code** utilisant le modèle de développement Modèle-Vue-Contrôleur pour programmer avec Zen ;
7. Une **vidéo** indiquant comment intégrer Zen à un projet avec Eclipse ;