Jolliet Corentin, Colin Kevin,
El Kandoussi Adnan, Nore Vincent

Conduite de projet
Tactics Arena

L2 Informatique

Université du Mans

Lien du github : `https://github.com/VincentNore/Tactics-Arena`

Contents

1 Présentation

Notre projet avait pour but de créer un jeu dérivé d'un jeu existant appelé Tactics Arena.
C'est un jeu de stratégie en ligne gratuit qui se joue à deux joueurs en tour par tour, similaire aux échecs dans la forme, mais qui possède une dimension beaucoup plus stratégique que celui-ci. Il est sorti en 2003 (fermer le premier mai 2016) sur micro-ordinateur et développer par Vito "Seed" Sze au format Flash MX aujourd'hui Adobe Flash.
Les joueurs avaient pour but de détruire toutes les unités ennemis en effectuant des actions de déplacements ou d'attaques chacun leur tour.
Chaque joueur débute avec dix unités. Chaque unité possède des caractéristiques qui lui sont propre (blocage, points de vie, puissance, armure) suivant sa nature.
Le joueur avait la possibilité d'avoir un compte payant pour accède à un certains nombre de services comme la possibilité d'avoir des unités plus variés et également créer des clans avec d'autres joueurs.

2 Objectifs

La finalité de ce projet était de développer un jeu ,à deux joueurs ou seul contre une intelligence artificielle, sur ordinateur en langage C en reprenant les bases du jeu Tactics Arena puis en l'améliorant avec l'ajout de fonctionnalités diverses pour se démarquer du jeu de base.
Nous avons donc dans un premier temps défini des règles à respecter et des fonctionnalités secondaire.
Nous voulions au départ positionner les pions aléatoirement sur des lignes bien précise du plateau.
Le jeu se joue sur un plateau de taille fixe définie dans le code du programme.
L'affichage de décors divers et variés, comme de l'eau des rocher, sur le plateau pour rajouter une dimension stratégique plus importante.
Les joueurs choisissent le nom de l'unité sont symbole qui permet de l'identifier et où positionner leurs unités sur le plateau. Ils peuvent les positionner sur n'importe quelle case du plateau.
Un système d'inventaire avec des objets à utiliser, il est identique pour chaque joueur.
La possibilité d'utiliser des attaques spéciales qui infliges plus de dégâts sur l'ennemi que l'attaque classique.
Toutes ces règles permettent de s'écarter du jeu de base, sans pour autant créer un nouveau jeu. Nous voulions non plus créer un simple de jeu de stratégie exactement comme Tactics Arena, mais bien en faire également un jeu de rôle.

3 Organisation

3.1 Répartition des tâches

Nous avions commencés par l'analyse et la conception du jeu à l'oral ainsi que par écrit. De cette réunions, il en est ressortis 4 grande tâches à réaliser distribuer à chacun des membres du groupe :

- NORE Vincent - L'affichage de la matrice avec les décors et unités.
- COLIN Kevin - La création des structures unités et décors.
- JOLLIET Corentin - Les règles de combat et de déplacements pour les unités.
- EL KANDOUSSI Adnan - La sélections des unités.

Chacune de ces tâches ont été réalisés et modifiés en fonction de l'avancement du projet.
Par la suite, d'autres tâches ont vus le jour :

- COLIN Kevin - Création d'un inventaire.
- COLIN Kevin & EL KANDOUSSI Adnan - Assemblage fonctions de combat.
- EL KANDOUSSI Adnan & NORE Vincent - Organisation du dépôt GitHub.
- NORE Vincent - Documentation.
- NORE Vincent - Assemblage du projet.
- NORE Vincent - Débogage.

3.2 Outils de travail

Nous avons d'abord listé les tâches à réaliser, mais également les tâches futures sur un document en .odt que nous avons ensuite ajoutés sur notre dépôt pour qu'il puisse être modifié par tout les membres du groupe. Ce document nous a permis de voir et connaître l'avancement des tâches et plus globalement du projet en lui-même.

Afin d'organiser notre projet et nos fichiers de code nous avons choisis de travailler dans un dépôt Git, pour cela, nous avons utilisés la plate forme GitHub qui nous à permis de travailler en collaboration tous ensemble sans interférer avec le travail des autres membres du groupe. Pour générer la documentation de notre programme nous avons décidé de choisir Doxygen qui permet de générer une documentation au format HTML, PDF, Latex avec une grande facilité. Enfin pour effectuer la création du rapport de projet ainsi que de notre présentation, la plate forme OverLeaf nous à été vivement conseillé pour pouvoir travailler ensemble en même temps afin de générer nos documents au format Latex.

4 Conception

4.1 Règles

Le jeu utilise un tableau de onze cases par onze cases et cinq pions par joueur.
Le jeu se déroule en tour par tour avec deux joueurs.
À chacun de leur tour, les joueurs ont la possibilité d'effectuer une à plusieurs actions, à savoir attaquer, déplacer leur(s) pion(s), utiliser un inventaire contenant des objets pour ainsi augmenter les caractéristiques du pion comme la vie et la mana et également réfléchir à une stratégie à effectuer.
Un joueur peut se déplacer et/ou aller dans son inventaire lorsqu'il contrôle son pion, mais seulement une seule fois par tour.
Le tour du joueur s'arrête lorsque le joueur a attaqué ou libérer son pion.
Le jeu ne s'arrête que lorsque l'un des 2 joueurs ne possède plus de pion jouable.

4.2 Jouer une partie

Lors du lancement du programme, il est demandé aux joueurs un de créer sa première unité(sbire). Pour cela, le joueur doit saisir le nom du sbire ainsi que sont symbole qui permettre d'identifier le sbire sur le plateau. Une fois que le joueur un à créer ses cinq sbires c'est au tour du joueur deux.
Une fois les armées de chaque joueur créent, deux nombres aléatoires sont tirés pour définir le joueur qui commencera à jouer.
Le joueur qui joue en premier doit alors sélectionné qui devra effectuer une action. Pour cela, le programme lui liste les unités dont le joueur dispose.
Lorsque l'unité est sélectionnée un menu apparais pour lui demander l'action qu'il souhaite effectuer. Si le joueur choisit de déplacer sont unité alors le programme lui demande de choisir entre un déplacement horizontal ou vertical puis de choisir entre gauche et droite ou haut et bas.
Si une unité ennemie est à porté alors le sbire sélectionné pourra l'attaquer, dans ce cas-là, il est demandé au joueur de choisir le type d'attaque. Dans le cas ou le joueur ne souhaite ni se déplacer ni attaquer, il peut toujours accéder à l'inventaire ou observer les environs pour détecter si un ou des ennemi(s) sont à porté.
Une fois les actions terminées, c'est au tour du joueur suivant de jouer.

4.3 Fonctionnalités prévues

Généralités :
Le programme doit prendre en compte l'état de chacun des pions après chaque action.
Si la vie d'un pion a atteint zéro, il est alors considérer comme mort, dans ce cas, il est supprimé du plateau et ne doit plus être utilisable par le joueur propriétaire.
Si tous les pions d'un des joueurs sont mis mort, le programme s'arrête et affiche le vainqueur de la partie.
Les joueurs doivent toujours être capables de garder un œil sur le terrain après chacun de leur tour et durant leur tour.

Déplacements :
Le programme doit permettre de sélectionner un pion à déplacer et le déplacer dans toutes les directions possibles.
Un déplacement effectué doit pouvoir être annulé et le joueur doit pouvoir choisir de continuer à bouger son pion tant qu'il reste des points de déplacement au pion sélectionné.

Attaques :
Un pion qui attaque doit avoir le choix entre une attaque classique, une attaque magique ou une compétence, utilisant de la mana, ayant chacune une portée limitée.
La mana du pion utilisant une attaque magique doit être réduite en conséquence.
Le pion qui attaque doit pouvoir choisir sa cible parmi celles situées à sa portée.
Si aucune cible n'est à portée alors le pion ne peut pas attaquer.

Inventaire :
Le programme doit permettre au joueur de sélectionner un objet à utiliser depuis un inventaire, identique à chaque joueur.
L'objet choisi dans l'inventaire doit être utilisable sur le pion choisi et prendre en compte l'état de ce dernier.
Une fois l'objet utilisé, il ne doit plus apparaître dans l'inventaire.

5 Développement

5.1 Présentation des éléments du programme

Étant donné que Tactics Arena repose essentiellement sur de la gestion d'unités, la structure est l'élément principal du projet.

Le programme utilise donc deux structure.
Une structure appelée sbires_t qui contient les caractéristiques des unités :

- Le nom de l'unité
- Le symbole pour l'identifié
- La puissance d'attaque
- Les points de vie
- La défense
- L'esquive
- Les points de magie
- Le nombre de déplacement possible
- La position de l'unité sur le plateau

Une seconde structure appelée armée_t qui contient cinq structure sbires_t et qui permet de créer les armées de chaque joueur.

Le plateau de jeu sur lequel évoluent les unités, est crée à l'aide d'une matrice de taille fixe. Chaque case de cette matrice peut contenir soit une case vide, soit une unité.

5.2 Tests

Les tests réalisés lors de la partie développement de ce projet sont restés très simples.

Seulement des tests bruts ont été réalisés. Savoir si cela compile ou pas, en changeant les valeurs initialisées manuellement pour correspondre aux situations de test de telle ou telle fonction.

Lors de mise en commun de deux travaux, deux tâches, par exemple la création d'unités et les fonctions de choix du joueur, un main commun est crée avec tous les fichiers sources et headers nécessaires de toutes les fonctions des deux tâches. Un makefile qui sert à compiler tous les fichiers contenants les éléments des deux tâches et créer l'exécutable qui permettra de lancer le test d'une fonctionnalité du projet.

6 Résultat

6.1 Fonctionnalités réalisées

La majorité des fonctionnalités principales ont été réalisées dans les temps notamment :

- L'affichage du plateau de jeu avec les unités et des couleurs pour différencier les unités de chaque joueur.
- Les fonctions de calcul de déplacements, d'ennemis, de porté d'attaques.
- Les fonctions de combat avec l'attaque basique, l'attaque magique et les compétences et de déplacement.
- La fonction d'inventaire avec l'utilisation d'objet.
- La fonction de détection de mot pour vérifier si une unité est encore en vie.
- La Fonction de choix d'une unité.

6.2 Difficultés

Toutes les fonctionnalités du programme n'ont pas été développé, certaines ont étés simplement abandonnés par manque de temps et d'autres modifiées du fait d'une revue à la baisse des exigences du projet face aux différents problèmes rencontrés lors de leur développement. La SDL à été notamment abandonnée par manque de temps du fait de son démarrage tardif dans le projet. Les différentes structures ont étés revues à la baisse et simplifiées (type d'unités et compétences variées) compte tenue du manque de temps.

La répartition du travail qui aurait dut être découpée en fonction et non pas en grandes tâches qui génèrent beaucoup trop de travail.
Le respect difficile du planning contenu de la mauvaise analyse des tâches et suite à cela le manque de temps pour effectuer certaines tâches.

6.3 Planning et répartition

Liste des tâches	Date de début	Date de fin	Durée	Kevin	Adnan	Corentin	Vincent
Analyse	30/01/18	30/01/18	3	30,00	30,00	20,00	20,00
Conception	06/02/18	06/02/18	3	35,00	25,00	25,00	25,00
Structures	15/02/18	19/03/18	15	70,00	20,00	5,00	5,00
Règles combats et déplacements	15/02/18	27/03/18	18	15,00	10,00	70,00	5,00
Environnement(matrice)	15/02/18	27/03/18	18	5,00	5,00	10,00	80,00
Choix du joueur	15/02/18	27/02/18	9	0,00	100,00	0,00	0,00
Inventaire	15/02/18	22/02/18	6	100,00	0,00	0,00	0,00
Organisation du dépôt Git	30/01/18	04/04/18	1	5,00	45,00	5,00	45,00
Assemblage	30/03/18	04/04/18	9	20,00	20,00	10,00	50,00
Documentation	03/04/18	04/04/18	2	0,00	0,00	0,00	100,00
Débogage	30/03/18	04/04/18	6	10,00	10,00	10,00	60,00

7 Conclusion

7.1 Amélioration du projet

Afin d'améliorer le programme, l'ajout des fonctions non implémenté par manque de temps pourrons êtres ajoutés.
La création d'unités différentes avec des caractéristiques différentes pour chacune d'entre elle.
L'ajout de décors et d'obstacle sur la matrice.
Le développement du jeu en interface graphique avec SDL ou d'autres bibliothèque graphique. La possibilité de jouer contre une intelligence artificielle. Et pour finir donner la possibilité au joueur, s'il joue contre l'intelligence artificielle, de sauvegarder une partie et de pouvoir la charger ultérieurement.

7.2 Apports du projet

Ce projet à permis dans un premier temps d'apprendre à gérer un projet, plutôt conséquent, en groupe. D'analyser les besoins pour développer le programme.
Répartir la charges de travail en tâches équitable pour chaque membre du groupe de projet.
Savoir gérer le temps de travail.
Définir des conventions pour éviter les écarts au niveau du nommage des variable, de

la définition des structures... Apprendre à mieux communiquer et travailler ensembles pour réussir à créer un programme qui fonctionne selon les attentes du groupe.