Product

Manuel utilisateur

Action du joueur

Ce projet a pour objectif la création d'un jeu vidéo en 2D dont le but est de faire prospérer une ville. Pour ce faire, le joueur aura accès à un certain nombre d'éléments de gameplay ainsi qu'aux informations des différents quartiers.

La carte de jeu est en fait une grille de quartiers, lesquels peuvent chacun être de trois types différents (quartier d'affaire, quartier commercial et quartier de services publiques). Au début d'une partie la carte sera vierge de quartiers. Chaque quartier, qu'il soit résidentiel, commercial ou de services publics à le droit à la création d'une station de métro au sein de celui-ci, entraînant ainsi des coûts de construction. Si plusieurs stations sont créées dans différents quartiers le joueur aura la possibilité par la suite, si ses ressources financières le lui permettent, de relier ces stations par des lignes de Métro qui pourront desservir plusieurs quartiers.

Pour gérer les différentes lignes de métro, le joueur aura la possibilité de créer une ligne de métro. L'utilisateur n'aura alors plus qu'à choisir les stations par lesquelles il désire que la ligne passe et pourra choisir d'autres cases, qui ne possèdent pas nécessairement une station de métro. Le joueur sera notifié du coût de production et une fois qu'il aura confirmée, la ligne sera créée. Si, par la suite, l'utilisateur souhaite modifier son choix et effacer un lien entre deux stations, il lui sera possible de le faire mais cela engendrera des frais supplémentaires.

Informations

Le joueur pourra accéder aux différentes données caractérisant le quartier : son nom, le nombre d'habitants, le taux de contentement de la population, le temps de trajet moyen, le nombre de lignes de métro passant dans ce quartier s'il y a une station (et le cas échéant son nom), ainsi que les informations financières.

Il pourra également visualiser les ressources financières générales qu'il possède afin de pouvoir gérer son budget. De plus, il aura également accès aux données importantes liées au déroulement de la partie comme les recettes et les coûts de la ville en temps réel mais également le taux de satisfaction moyen de tous les habitants de la ville.

But du jeu

La prospérité des quartiers résidentiels et commerciaux dépend de l'accessibilité au métro, c'est à dire du temps de trajet moyen entre ces quartiers et ceux de services publics, lequel dépend du nombre d'habitants du quartier ainsi que de la proximité à une station de

métro. En effet, un quartier ne possède pas forcément une station de métro, la population de ce quartier doit donc se rendre à pied dans un quartier voisin afin de pouvoir utiliser le métro, mais cela augmente considérablement son temps de trajet. La population du quartier évolue en fonction de la prospérité, de sorte que ces deux paramètres jouent chacun l'un sur l'autre. Plus le quartier est prospère, plus il aura d'habitants mais cette augmentation aura pour conséquence d'augmenter le temps de trajet moyen et le taux de surpopulation du quartier, et donc de diminuer la prospérité.

L'équilibre financier permet au jeu un certain réalisme. Il oblige le joueur à faire des choix, lui rendant impossible le fait de construire des stations et des lignes de métro dans tous les coins de la carte. De ce fait, le développement de sa ville prend un certain temps et la stratégie de construction du joueur est récompensée. Cet équilibre est calculé par rapport aux coûts et recettes de la ville. Les coûts peuvent être générés par la construction de lignes de métro et de quartiers ainsi que par l'entretien des lignes de métro et des quartiers de services publics, mais des recettes sont générées par les impôts dans les quartiers résidentiels. Chaque type de quartier possède une formule propre permettant de calculer le coût / recette. Ces formules sont conçues pour que l'équilibre financier soit atteignable tout en étant délicat à maintenir.

Dynamiques techniques

Tout d'abord, les quartiers possèdent une variable binaire hasStation. Celle-ci permet d'apporter un "malus" financier aux villes possédant une station métro, dû à l'entretien de celle-ci. Ce malus financier est multiplié par deux fois le nombres d'habitants travaillant. De plus, les personnes ne travaillant pas apportent également un malus, car elles ne génèrent pas de revenu mais empruntent tout de même les stations de métro pour se rendre dans les quartiers publics. Cependant ces malus sont compensés par les impôts payées par les travailleurs sociaux ainsi que par la population travaillant dans les services publics. On peut alors calculer les revenus générés par les quartiers résidentiels par la formule suivante :

recettes = (-2) * hasStation * (populationTravailleurs) - populationNonTravailleurs + 2.5 * populationTravailleursQuartiersAffaires + 1.5 * populationTravailleursServices publics + populationTotale

Il en va de même pour les quartiers commerciaux ainsi que pour les quartiers de services publics, bien que les formules soient bien plus simples :

recettes (Commercial) = 3 * populationTravailleurs

recettes (Services publics) = -2 * populationTravailleurs

Nous avons jugé que ces chiffres permettraient de trouver un équilibre financier assez facilement en laissant place à l'imagination pour le joueur, tout en imposant une certaine rigueur et en punissant les erreurs commises par le joueur.

Chaque quartier possède un certains nombre d'habitants, et ceux-ci font tous partie de différentes démographies, c'est à dire qu'ils ont des fonctions différentes au sein du quartier. Les pourcentages population de chaque quartier sont décidées de manière aléatoire à la création de ceux-ci, ainsi que lors de l'expansion de la population d'un quartier. La répartition de ces populations suit les règles suivantes lors de la création d'un quartier :

- La population totale est comprise entre 10 000 et 15 000 habitants.
- La population de chômeurs est comprise entre 0.2 et 0.4 * la population totale du quartier.
- Le total de personnes travaillant dans les services publics est compris entre 0.3 et 0.4 la population totale du quartier.
- Le reste des habitants travaille dans les quartiers commerciaux.

Les quartiers possèdent également des dynamiques de déplacement de population spécifiques. Ainsi, chaque jour de la semaine, un neuvième de la population va se diriger vers les services publics, et chaque jour du week-end deux neuvième se dirigeront vers les services sociaux. Le calcul du temps moyen de transport par quartier se fait en plusieurs étapes. Tout d'abord, on trouve la station la plus proche, et à partir de cette station, si elle est dans une ligne, la station d'un quartier de service publique ou bien d'un quartier commercial la plus proche est trouvée et sauvegardée dans les paramètres du quartier. De même, on calcule le quartier de même type le plus proche à pied. Ensuite, on calcule le temps que prend un habitant à se rendre dans chaque quartier en comptant que traverser une case à pied prend 15 minutes, et traverser une case en métro prend 2 minutes. Les habitants se rendent alors dans leur quartier de travail respectifs avec le moyen le plus rapide. Ces statistiques vont permettre de calculer le temps moyen de voyage pour les habitants travaillant dans les services publics ainsi que pour ceux travaillant dans les quartiers d'affaires. La formule utilisée est relativement simple :

tpsMoyenTransport = (popTravaillantServicesPublics * tpsDeTransportServicesPublics + popTravaillantQuartierAffaire * tpsDeTransportQuartierPublics) / populationTravailleurs

Grâce aux deux paramètres précédents, on va pouvoir calculer un taux de satisfaction des quartiers, qui dépendra de plusieurs paramètres tels que le temps moyen de transport. De plus, chaque quartier possédera une valeur de population idéale, au dessus de laquelle le mécontentement des habitants du quartier augmentera. On peut alors calculer la satisfaction des habitants du quartier par la formule suivante :

happiness = averageCommuteTime * populationTravailleurs + (idealPop * totalPop)² * 0.6

Enfin, il existe un dernier paramètre qui correspond à la croissance de la population du quartier. Celle-ci est calculée afin d'être minimale lorsque le contentement de la population est au plus bas, sans pour autant être excessive lorsque celle-ci est au plus haut. Pour ce faire, on utilise la fonction logarithme pour obtenir ces propriétés.

Didacticiel

Lorsque le jeu se lance, une map quadrillée s'affiche. Seront également affichées les informations relatives à la ville, c'est à dire le taux de contentement de la ville, l'argent que possède le joueur mais aussi le nombre de quartier de chaque sorte ainsi que le nombre de stations et de lignes présentes dans la ville.

Pour commencer à jouer, le joueur pourra cliquer sur la case de son choix. Un menu va alors apparaître lui proposant de construire un quartier résidentiel, commercial ou de services publics. Il devra également donner un nom à ce quartier, qui permettra de l'identifier de manière générale. Cette construction pourra être effectuée seulement si le joueur possède

une réserve suffisante d'argent. Une fois ce dernier construit, l'utilisateur peut alors cliquer dessus ou sur une autre case pour en construire un nouveau. S'il choisit de cliquer sur le quartier, une nouvelle fenêtre va apparaître montrant toutes les informations relatives à celui-ci mais aussi qui lui permettra de bâtir une station de métro moyennant une certaine somme d'argent.

Si le joueur possède au moins deux quartiers contenants chacun une station alors il pourra, s'il le souhaite, les relier par une ligne de métro. Cette ligne doit obligatoirement partir d'un quartier ayant une station et se terminer dans un autre en possédant une également. Pour créer cette ligne, il suffit de cliquer sur une case juxtaposée à la précédente ou à la station. Une fois la ligne finie il ne reste plus qu'à cliquer sur un bouton pour terminer cette construction. Plus la ligne est longue, plus son coût de fabrication sera important, il faudra donc être économe dans l'utilisation des ressources et optimiser l'espace mis à disposition.

Manuel système

Articulation du programme

Le programme est composé de quatre paquets permettant de diviser les différentes classes par type et par thème. On possède donc :

- un paquet "test" permettant de gérer les différents modules de test du programme.
- un paquet "time" dans lequel on implémente un chronomètre, lequel va nous permettre de compter les jours et donc le système de temps du jeu.
- un paquet "gui", qui lui contient toutes les classes permettant de créer l'interface graphique du jeu. Cela passe par les panels d'option, de création de ligne, de quartiers et de stations de métro, mais également le panel de jeu en lui même ainsi que celui fournissant des informations sur la partie de manière générale.
- et un dernier paquet "core", contenant toutes les classes du moteur de jeu. C'est notamment dans ces classes que l'on va pouvoir retrouver les systèmes de gestion des différents quartiers, des stations ainsi que des lignes de métro.

Déroulement du code

La classe principale, Game, est créée depuis le main du programme. Ce choix a été fait afin de pouvoir implémenter un listener, ceci n'étant pas possible depuis le main car il est obligatoirement statique, depuis la classe principale, qui permettra d'identifier l'ensemble des actions du joueur, et donc d'agir en conséquence au sein du programme. Pour ce faire, les variables principales du core ainsi que de l'interface graphique sont instanciées, puis on rentre dans la boucle de jeu à proprement parler.

Celle-ci est relativement simpliste. Elle va tout d'abord vérifier sur un intervalle de temps donnée est passé depuis la dernière fois. Si c'est le cas, elle va alors mettre à jour l'interface graphique puis Celle-ci est relativement simpliste. Elle va tout d'abord vérifier sur un intervalle de temps donnée combien de temps est passé depuis la dernière fois. Si c'est le cas, elle va alors mettre à jour l'interface graphique puis incrémenter le compteur de temps.

Elle mettra également à jour les informations de chaque quartier en fonction des nouveaux quartiers ayant été créés tout au long du jeu ainsi que les dynamiques de populations qui auront évolué. Par exemple, chaque jour, la population augmentera ou diminuera en fonction de la satisfaction des habitants de chaque quartier.

Problèmes rencontrés et solutions apportées

Dans cette partie on évoquera quelques problèmes rencontrés au cours de la réalisation du programme et les solutions qu'on a pu mettre en place et celle qui ont étés prises pour le programme final.

- Problème d'innovation : Le tout premier problème auquel on a fait face fut comment commencer le projet ? La solution était vite apportée après réunion qui était de s'inspirer sur des solutions existantes notamment les Sims.

- Création de lignes: à la mise en place de la créations des lignes de métros au niveau du moteur on a réalisé solutions qui sont les suivantes

  - Lignes directes: Cette méthode consiste à sélectionner deux stations de métros et la ligne est construite en ligne droite avec le plus court chemin entre les deux stations.

  - Ligne choisie: Cette méthode en revanche donne le choix de pouvoir choisir comment placer la ligne c'est-à-dire en sélectionnant toutes les cases ou la ligne peut passer à condition que la case de début et de fin de création soit des quartiers muni d'une station de métro.

  Et c'est finalement cette dernière qui a été prise pour le programme final en vu de son large champ de possibilités qui permettra à l'utilisateur d'avoir plus de choix.

- Interface graphique: Concernant l'interface utilisateur, tout au début nous nous sommes orientés vers deux approches distinctes. D'une part, nous voulions créer une map quadrillée, ou chaque case serait un bouton. Et d'autre part, nous voulions dessiner une carte quadrillée sur laquelle chaque case (n'étant pas un bouton cette fois-ci mais simplement le sprite de fond) pourrait être cliquable par simple calcul de distinction de ces cases. Nous avons alors écarté la première approche car jugée pas assez optimale et laissé place à la deuxième.
- Problème de complexité: Définir les conditions de jeu était très complexe, vu que le contentement de la population par exemple peut dépendre de plusieurs variables sur lesquelles dépendent d'autres variables. Donc on a dû simplifier au maximum les variables du jeu pour diminuer la complexité de l'assemblage.

- Problème contextuel: Pour l'implémentation des conditions de jeu, en premiers on était partis juste sur la gestion des flux des naissances et personnes entrantes pour définir le contentement de l'ensemble de la population grâce au taux de chômage ainsi que le nombre de personnes sans logement(SDF) qui évoluer en fonction du temps. Mais pour concorder avec l'énoncé du projet et le cahier des charges, on a rajouté des variables plus précises sur le temps de déplacement moyens des résidents.

Points fort du système

- Concernant l'approche visuelle, nous avons opté pour l'utilisation de sprites, rappelant un peu l'univers des jeux rétros. Pour cela nous avons utilisé le site web Piskel pour les dessiner. Nous en avons créé une dizaine, notamment utilisés pour la représentation du fond de la map, pour rappeler un champ ou un terrain vague, les différents quartiers qu'ils soient Résidentiels, Commerciaux ou de Services publics. Nous en avons aussi fait pour les différentes orientations que peut prendre une ligne en fonction de l'action de l'utilisateur. Si ce dernier veut créer un virage lorsqu'il construit sa ligne alors elle va s'adapter et utiliser le sprite qui correspond.

Extensions possibles

- Améliorer l'interface graphique.
- Ajouter des conditions de jeu.
- Augmenter la complexité en tenant compte de chaque personne
- Varier les quantités d'accueil des quartiers résidentiels
- Varier les nombres de poste de travail des quartiers commerciaux et services publics

Plans des tests

Le test est une activité indispensable pour l'obtention d'application de qualité.

Dans notre application on a utilisé Les plateformes de test du langage utilisé (java), pour instrumenter des tests automatiques.

Tests

Problématique de l'organisation des tests:

Parfois au début, il nous arrivait à l'ajout d'une nouvelle fonctionnalité de découvrir qu'il faut toucher au code des fonctionnalités déjà existantes, ce qui affectait la lancée automatique des tests.

Après, on a commencé à suivre une méthodologie d'intégration continue, en veillant à rester à jour.

Plan des tests

Tests boite blanche :

- Tests unitaires : Pour chaque validation de transaction où on teste chaque fonction de chaque module.
- Tests d'intégration: Pour chaque partie d'avancement du projet où on test les interactions entre modules.

Tests Boite noire:

- Tests du Système : Tester l'efficacité des fonctionnalités de l'applications

Critères de qualité du produit : modularité, opérabilité, complétude.