Document d’itération

Echoes of the Engine

Fiche signalétique :

I. Genre : Textuel, Gestion.
II. Support : Windows.
III. Cible : Joueur de visual novel.
IV. Type de jeu : Solo.

Concept :

Vous conduisez un train qui part de la ville de Soleanna et doit arriver à la ville d’Hibernia, vous devrez gérer les réserves de nourriture, de charbon et d’eau de votre train afin de pouvoir continuer à avancer.

Pour conduire ton train tu as le choix entre 3 commandes :
“a” : La première consiste à faire avancer ton train à vitesse normale.
“b” : La seconde consiste à faire avancer ton train à une plus grande vitesse.
“c” : La dernière qui te permet d’afficher d’afficher le gestionnaire des ressources.

Mécanique principale :

L'interactivité du train avec le joueur.

Univers :

Le monde dans lequel vous vous trouvez a été détruit durant la dernière guerre.
Seuls vous et votre train êtes encore en vie.
Vous venez de la ville de Soleanna et votre voyage doit vous mener jusqu’à la ville d’Hibernia, connue comme étant la seule ville encore debout.

Gameplay :

I. Stats du train :

eau : 20 unités.
charbon : 20 unités.
nourriture : 20 unités.
distance : 2500 km.
distance parcourue : 0 km.

II. Interactivité avec le train :

Touche “a” : Le train avance de 20 km.

Touche “b” : Le train avance de 40 km.

Touche “c” : Permet d’ouvrir l’interface de gestion des ressources.

III. Gestion de l’état du train :

($”Eau restant : {eau} unités”) : Permet de garder en mémoire les unités d’eau du train.

($”Charbon restant : {charbon} unités”) : Permet de garder en mémoire les unités de charbon du train.

($”Nourriture restante : {nourriture} unités”) : Permet de garder en mémoire les unités de nourriture du train.

($”Distance restant : {distance} km”) : Permet de garder en mémoire la distance restante au train avant d’arriver à la ville d’Hibernia.

($”Distance parcourue : {distanceParcourue} km”) : Permet de garder en mémoire la distance parcourue par le train.

IV. Fonction nuit :

Tous les 50 km le train va demander au joueur s'il veut s'arrêter, pour réapprovisionner la réserve de charbon et d'eau ou s'il veut continuer de nuit.

Demande du train au moment où la nuit arrive :
("C'est la nuit.")
("Tu arrive à une gare.")
("Veux-tu t'arrêter ?")
("y : yes. ")
("n : no. ")

Fonction de Repos() quand la nuit l'appel :

Si le joueur appuie sur "y" le jeu appel la fonction Matin() :
- Le joueur perd 5 unités de nourriture.
- Le joueur gagne 10 unités d'eau mais ne peut pas dépasser les 20 unités d'eau.
- Le joueur gagne 10 unités de charbon mais ne peut pas dépasser les 20 unités de charbon.

Ensuite le train vérifie ce qu'il reste dans les ressources (VerifiedRessources())et renvoie à la fonction Command() seulement s'il reste assez de ressource dans la réserve.

Si le joueur appuie sur "n" le jeu donne le message suivant : ("Tu te décides à continuer ton chemin de nuit !") et appelle la fonction Command().

V. Événement aléatoire :

Tous les 30 km le train va rencontrer des événements aléatoires positif comme négatif qui vont impacter ses différentes ressources.

Un eventNum = random.Next(1, 5) se lance pour donner l'un des événements au hasard contenu dans le switch de la fonction Evenment().

Événement 1 : ProbEau(). Quand cette fonction est appelée un eventFuite = random.Next(0, 3) est lancé pour savoir s'il se passera quelque chose ou non, ensuite une fois la condition du résultat du random fait la condition Command() est appelée.

Si le random tombe sur 1 :
Le jeu donne les messages suivant :
("Oh non, le réservoir d'eau a une fuite, nous avons perdue 5 unités d'eau !")
("Regarde avec la touche "c" pour vérifier ce qu'il te reste en eau !")
- Le joueur perd 5 unités d'eau.

Si le random tombe sur 2 :
Aucunes stats ne change, le jeu donne le message suivant ("Pas de problème avec l'eau aujourd'hui !").

Événement 2 : ProbBan(). Quand cette fonction est appelée un eventBandie = random.Next(0, 3) est lancé pour savoir s'il se passera quelque chose ou non, ensuite une fois la condition du résultat du random fait la condition Command() est appelée.

Si le random tombe sur 1 :
Le jeu donne les messages suivant :
("Nous avons croisez des bandits et ils nous ont volé de la nourriture !")
("Ils nous ont volé 5 unités de nourritures")
- Le joueur perd 5 unités de nourriture.

Si le random tombe sur 2 :
Aucunes stats ne change, le jeu donne le message suivant ("Nous avons croisé des bandits mais heureusement ils ne nous ont rien voler !")

Evénement 3 : ProbNei(). Quand cette fonction est appelée un eventNeige = random.Next(0, 3) est lancé pour savoir s'il se passera quelque chose ou non, ensuite une fois la condition du résultat du random fait la condition Command() est appelée.

Si le random tombe sur 1 :
Le jeu donne les messages suivant :
("Nous allons traverser une tempête de neige, nous devons utiliser plus

de charbon pour garder la chaudière chaude !")
("Nous avons perdue 5 unités de charbon !")
- Le joueur perd 5 unités de charbon.

Si le random tombe sur 2 :
Aucunes stats ne change, le jeu donne le message suivant ("La tempête de neige est heureusement passée loin du train !").

Evénement 4 : Bouffe(). Quand cette fonction est appelée le jeu va poser la question au joueur s' il veut récupérer la caisse de nourriture ou non.
("Tu as trouvé une caisse de nourriture sur les rails.")
("Veux-tu la récupérer ?")
("y : yes.")
("n : no. ")

Si le joueur appuie sur "y" :
Le jeu donne le message :
("Tu récupère 5 unités de nourritures")
- Le joueur gagne 5 unités de nourriture.
Ensuite le train renvoie à la fonction Command().

Si le joueur appuie sur "n" : Aucune stats ne change, le jeu donne le message suivant ("Tu ne récupères pas de nourriture") et appelle la fonction Command().

VI. Gestion de la mort :

Quand le joueur perd toutes les unités d'une ressource, la fonction Die() est appelée.
Quand Die() est appelée le jeu donne les messages :
("Tu as perdu.")
("Veux-tu rejouer ?")
("y : yes")
("n : no")
et appelle la fonction Retry().

Fonction Retry() quand la fonction Die() l'appelle :

Si le joueur appuie sur "y" :
Le train renvoie à la fonction ResetGame() et à la fonction Main().

Si le joueur appuie sur "n" :
Le jeu donne le message ("Merci d'avoir jouer !") et ferme le terminal grâce à la commande Environment.Exit(0).

Fonction ResetGame() quand la fonction Retry() l'appelle :

```
eau = 20;
charbon = 20;
nourriture = 20;
distance = 2500;
distanceParcourue = 0;
```

VII. Gestion de la victoire :

La fonction Victory() n'est appelée que lorsque le joueur gagne le jeu en ayant parcouru les 2500 km qui le sépare de la ville d'arrivée.

Fonction Victory() :

Le jeu donne les messages :
("Bravo, tu as gagné !")
("Merci d'avoir jouer !")
et ferme le terminal grâce à la commande Environment.Exit(0).

USP :

Allez-vous vous risqué sur les rails vers Hibernia ?