v1.0 Projet CY-Valley

CLASSE préING1 • 2022-2023
AUTEUR Romuald GRIGNON
E-MAIL romuald.grignon@cyu.fr
D'APRES UNE IDEE ORIGINALE DE groupe MI5

DESCRIPTION

Ce projet est un jeu dans lequel vous déplacez un personnage sur une carte plane en 2 dimensions. Il n'y a pas de but imposé quant au but de ce jeu, c'est à vous de créer les mécaniques et l'ambiance du jeu en respectant les contraintes qui sont données ci-dessous.

[CARTE]
- La carte du jeu sera une matrice d'au moins 10000 cases (e.g. 100x100).
- La zone affichée à l'écran sera une zone de NxN cases centrées sur la position du personnage du joueur (avec N << 100, e.g. N=11)
- Il existera plusiers types de cases, à vous de décider lesquelles (eau, sable, herbe) avec chacune des propriétés purement esthétiques, ou des propriétés d'interaction avec le joueur.
- Il doit y avoir au moins 1 type d'obstacle indestructible et infranchissable (e.g. rocher. il est possible d'entourer TOUTE la carte d'une rangée de ce type d'obstacle pour limiter la zone de jeu par exemple).
- Il doit y avoir au moins 1 type d'obstacle destructible et infranchissable (e.g. caisse en bois trop lourde pour la déplacer mais que l'on peut casser avec un outil que l'on possèderai dans son inventaire).
- Il doit y avoir au moins 1 type d'obstacle destructible et franchissable (e.g. des fleurs que l'on peut ramasser lorsque l'on est positionné dessus, mais que l'on peut laisser par terre si l'on franchit simplement la case).
- Il doit y avoir au moins 1 type d'obstacle déplaçable (e.g. du mobilier que l'on pourrait pousser lorsqu'on se déplace contre cette case (Sokoban ?), ou bien lorsqu'on le récupère dans son inventaire et qu'on le redépose plus tard, au choix).
- La carte doit être générée "aléatoirement" au lancement de la partie (avec des probabilités pour chaque type de case : veillez à ne pas mettre des probabilités trop importantes pour les obstacles infranchissables car cela pourra générer une carte non jouable ou tout du moins non explorable totalement)

[DEPLACEMENTS/ACTIONS]
- Le déplacement du personnage sera à minima orthogonal, case par case (e.g. appui sur les touches 2,4,6,8 ou ZSQD).
- Vous pouvez définir toutes les actions que vous voulez et associer les

touches que vous voulez pour les réaliser.
Le programme vérifiera si le moment où vous appuyez sur les touches est propice à l'action associée (ex: appuyer sur E pour interagir avec un personnage ou ramasser un objet peut être suffisant, il suffit que le programme sache où se situe le personnage pour savoir si il peut interagir avec telle ou telle partie de l'environnement, ou bien vous associez des touches différentes pour chaque action possible du jeu).

[INVENTAIRE]
- Votre personnage doit posséder un inventaire avec au moins 3 types d'objets différents à stocker (le contenu de l'inventaire doit toujours être affiché à l'écran).
- Vous devez coder des mécaniques pour remplir tout ou partie de votre inventaire en interagissant avec l'environnement (e.g. ramasser une fleur qui se trouve sous vos pieds en appuyant sur une touche).
- Au moins 2 éléments de votre inventaire sont limités en quantité (e.g. impossible de stocker plus de 20 fleurs dans l'inventaire à la fois et pas plus de 10 carottes non plus).

[POINTS DE VIE]
- Votre personnage possède des points de vie (qui sont toujours affichés à l'écran).
- Vous devez coder au moins 1 manière de perdre des points de vie (marcher sur des pics, combattre des ennemis si il y a des ennemis et des mecaniques de combats, faire trop de déplacements, ...).

[SCORE]
- Votre programme doit poasséder un score qui sera toujours affiché à l'écran.
- Un maximum de mécaniques du jeu doivent influer sur le score (ramasser un objet, terminer une quete, terrasser un ennemi, se déplacer aussi pourquoi pas, ...).

[SAUVEGARDE]
- Votre programme doit proposer un système de sauvegarde du jeu en cours (soit à un moment précis, une position précise, ou bien à tout moment en choisissant l'option sauvegarder dans les choix d'actions). Au moins une sauvegarde doit être possible.
- Votre programme doit proposer un moyen de reprendre une ancienne sauvegarde, c'est à dire démarrer une partie depuis le moment où elle a été sauvegardée sur le disque dur dans un fichier.

[INTERACTIONS / QUETES]
- Votre programme doit proposer au moins 1 interaction avec un personnage non joueur pour lequel vous devez remplir une quête pour lui. La réalisation de cette quête doit générer plusieurs interactions avec ce personnage (e.g. : vous devez lui rapporter toutes les fleurs de la zone. Avec les exemples précédents, vous devez donc effectuer plusieurs aller-retours vers ce personnage pour transférer les fleurs depuis votre inventaire. Vous devez afficher au moins un message la première fois que vous le rencontrez et ensuite à chaque fois que vous interagissez avec, et enfin il faut au moins un message quand vous avez rempli la quete complètement).

[VICTOIRE/DEFAITE]

- Votre programme doit gérer au moins 3 conditions de victoire différentes (e.g. une quête terminée, venir à bout de tous les ennemis de la carte si vous décidez qu'il y a des ennemis, déplacer des ruches à des emplacements prévus (sokoban? … un jeu dans le jeu ?), …).
- Il faut au moins 1 condition de défaite quand le joueur n'a plus de points de vie.
- Il faut au moins 1 condition de défaite liée au temps (e.g. finir le jeu en un temps limité ou bien si une quête est lancée, il faut qu'elle soit terminée dans un certain temps imparti. Respectivement, le temps est mesuré depuis le début de la partie, ou depuis le début de la quête.)
- ATTENTION : cette fonctionnalité doit être compatible avec la sauvegarde/restauration du jeu.
- Le temps de jeu restant doit être affiché à l'écran (soit le temps global, soit le temps d'une quête en cours, …)
- Chaque condition de victoire ou de défaite remplie affiche un message correspondant à l'écran et termine le programme.

[MENU PRINCIPAL]

- Au demarrage du programme vous afficherez un menu qui demande à minima de lancer une nouvelle partie, de reprendre une partie sauvegardée, d'afficher les commandes du jeu (aide) ou bien de quitter tout simplement.

INFORMATIONS GENERALES

Taille de l'équipe

Ce projet est un travail d'équipe. Il est autorisé de se réunir en groupe de 3 personnes.

Démarrage du projet

Vous obtiendrez de plus amples informations quant aux dates précises de rendu, de soutenance, les critères d'évaluation, le contenu du livrable, …, quand le projet démarrera officiellement.

Dépôt de code

Vous devrez déposer la totalité des fichiers de votre projet sur un dépôt central Git. Il en existe plusieurs disponibles gratuitement sur des sites comme github ou gitlab.

Démonstration

Le jour de la présentation de votre projet, la version finale sur votre dépôt sera celle utilisée, même si vous avez ajouté des fonctionnalités ou corrigé des bugs entre temps. La démonstration se fera sur une machine de l'enseignant chargé de suivre votre groupe. C'est la date de commit sur votre dépôt qui fera foi.

Vous ferez votre démonstration, en fonction des exigences du cahier des charges de votre projet, et vous aurez à modifier légèrement votre application en direct en fonction de la requête de votre chargé de TD.

De plus des questions supplémentaires pourront être posées afin d'évaluer votre connaissance de l'implémentation de votre projet.

FONCTIONNALITES PRINCIPALES DU PROJET

- Le but principal du projet est de fournir une application **fonctionnelle**.
- Tous les **éléments de votre code** seront écrits en langue **anglaise** (structure, types, fonctions, fichiers, ...).
- Votre code sera généreusement **commenté** (langue française autorisée).
- Votre application ne doit **jamais** s'interrompre de manière intempestive (crash), quelle que soit la raison. Toutes les erreurs doivent être gérées correctement. Il est préférable de d'avoir une application stable avec moins de fonctionnalités qu'une application contenant toutes les exigences du cahier des charges mais qui plante trop souvent. Une application qui créée des fautes de segmentation par exemple sera très pénalisée.
- Votre application devra être **modulée** afion de ne pas avoir l'ensemble du code dans le même fichier.
- Votre **livrable** sera une **URL** d'un dépôt **Git public** envoyée à votre chargé de TD.

- L'ensemble du cahier des charges doit être respecté.
- Vous porterez une attention particulière à la gestion des structures, et aux interactions entre votre personnage et les cases de l'environnement, et enfin à l'affichage de la zone de jeu.
- Pour la partie graphique de votre projet, vous pouvez utiliser des emojis : sur les terminaux Unix, cela permet de rajouter beaucoup plus de lisibilité :

RESSOURCES UTILES

Github

- www.github.com
- https://docs.github.com/en/get-started/quickstart/hello-world

Modifier les couleurs du terminal Linux

- https://en.wikipedia.org/wiki/ANSI_escape_code#Colors
- http://sdz.tdct.org/sdz/des-couleurs-dans-la-console-linux.html

Contrôler le curseur du terminal

- https://en.wikipedia.org/wiki/ANSI_escape_code#CSI_(Control_Sequence_Introducer)_sequences