Candice Giami & Clément Szewczyk

Rapport du Projet d’Architecture Système

Le Gestionnaire de tâches différées

# Introduction :

À travers ce rapport, nous documentons en détail notre approche pour répondre à ce défi, mettant l'accent sur la modularité, la simplicité d'utilisation et la flexibilité. Nous explorerons les choix de conception effectués pour permettre une gestion efficace des tâches différées, en fournissant une analyse approfondie du problème et une solution pratique pour une planification dynamique.

En respectant les contraintes du langage C standard, nous avons développé des structures de données efficaces, des fonctions modulaires et des mécanismes de gestion des tâches, offrant ainsi une expérience utilisateur fluide et intuitive. Ce rapport servira de guide complet, fournissant une documentation détaillée de notre code, un mode d'emploi approfondi et des exemples d'utilisation pour aider les utilisateurs à exploiter pleinement les fonctionnalités offertes.

Rejoignez-nous dans cette aventure de développement, où chaque ligne de code contribue à façonner un environnement propice à des tâches différées efficaces et à des processus optimisés.

Une image contenant Dessin d’enfant, dessin, habits, clipart

Description générée automatiquement

## Objectif du Projet :

L'objectif principal de notre projet est de concevoir et de mettre en œuvre un gestionnaire de tâches différées en langage C. Cette application fournira aux utilisateurs un moyen simple et efficace de planifier et d'exécuter des tâches à des moments précis. L'interface utilisateur intuitive permettra aux utilisateurs de créer, visualiser et gérer facilement leurs tâches différées.

## Détails :

### Contraintes :

* Nous habitons très loin l’un de l’autre, nous avons donc dû nous organiser à distance via Git Hub.
* Nous étions ainsi obligées de se partager le travail plutôt que de travailler tous les 2 ensembles simultanément.

### L'équipe :

L’équipe pour réaliser ce projet est composé de Candice et de Clément.

### Mode de travail :

Nous avons commencé à réaliser le projet un peu après les vacances de février. Nous avons réalisé la partie minimum assez rapidement.

Nous nous sommes partagé le travail de façon équitable :

* Clément a réalisé la première version du code, non fonctionnelle donc Candice a pris le relais.
* Candice rédige le rapport.

De temps en temps sur Discord nous partagions nos problèmes et nous nous aidions mutuellement. Nous nous mettions d’accord sur ce que nous ferons au préalable pour que notre travail s’imbrique bien faute de se voir souvent à cause de nos emplois du temps respectif. Puis on s’envoyait le code mis à jour sur Git Hub.

## Présentation du plan :

​​

Table des matières

[Introduction : 1](#_Toc163555500)

[Objectif du Projet : 1](#_Toc163555501)

[Détails : 2](#_Toc163555502)

[Contraintes : 2](#_Toc163555503)

[L'équipe : 2](#_Toc163555504)

[Mode de travail : 2](#_Toc163555505)

[Présentation du plan : 2](#_Toc163555506)

[Développement : 3](#_Toc163555507)

[La réalisation : 3](#_Toc163555508)

[Choix et aléas : 7](#_Toc163555509)

[Avantages des choix : 9](#_Toc163555510)

[Vie du projet : 10](#_Toc163555511)

[Conclusion : 10](#_Toc163555512)

[a. Dans le Cadre du Projet 10](#_Toc163555513)

[b. Au-delà du Projet 13](#_Toc163555514)

​

​

​

## Développement :

### La réalisation :

#### Ce que nous devons faire :

Dans le cadre de notre projet, nous développons un logiciel de gestion de tâches différées en langage C. Ce logiciel offre une solution flexible pour la gestion des tâches informatiques en permettant l'exécution différée de commandes. Notre objectif principal consiste à développer une version minimale du programme qui permettra la planification des exécutions itérées d'une commande. L'utilisateur pourra dynamiquement spécifier les paramètres suivants lors de l'exécution du planificateur :

- La commande à exécuter ;

- Le délai entre chaque exécution ;

- Le nombre d'itérations de la commande.

Il est important de noter que ces paramètres ne sont pas prédéfinis lors de la compilation, mais sont fournis au programme lors de son exécution, offrant ainsi une grande flexibilité.

Dans une version plus avancée du programme, nous prévoyons d'ajouter des fonctionnalités supplémentaires pour enrichir l'expérience utilisateur. Parmi ces fonctionnalités, nous envisageons la possibilité de programmer une date précise pour l'exécution des tâches avant de lancer le planificateur. De plus, nous pourrions implémenter la capacité pour le planificateur de gérer un nombre infini d'itérations pour une plus grande adaptabilité aux besoins des utilisateurs.

#### La structure du code :

La version minimale de notre projet est composée de trois fonctions principales :

1. **L'exécuteur de commande :** La fonction **executer\_commande** est chargée de lancer la commande via le Shell. En cas d'échec de l'exécution, elle affiche un message d'erreur pour informer l'utilisateur.
2. **Le planificateur :** La fonction **planifier\_taches** gère la planification des tâches. Elle appelle l'exécuteur de commande le nombre de fois spécifié par l'utilisateur et attend le délai spécifié entre chaque exécution.
3. **Le main :** IL récupère les informations fournies par l'utilisateur via l'invite de commande. Il vérifie la conformité de ces informations et appelle ensuite la fonction **planifier\_taches** pour démarrer le processus de planification des tâches.

Haut du formulaire

### 

### Choix et aléas :

* **Structure du Donjon :**
* *Choix :* *:* Représentation du donjon sous forme d'une matrice de caractères.
* *Aléa :* *:* Les donjons très grands peuvent générer des problèmes de gestion de la mémoire, entraînant des ralentissements ou des erreurs d'exécution. Heureusement nous avons limité la taille de nos donjons à 70.

* **Représentation du Donjon :**
* **Avantages :** La représentation matricielle simplifie la manipulation et la visualisation du donjon. La limitation de taille évite les problèmes de gestion de mémoire pour des donjons trop grands.
* **Gestion des Erreurs :**
* **Avantages :** Signaler les erreurs pendant la création ou la sauvegarde améliore la robustesse du programme, aidant les utilisateurs à comprendre les problèmes.
* **Choix des Caractères pour Représenter les Éléments :**
* **Avantages :** L'utilisation de caractères spécifiques offre une visualisation claire et une compréhension facile du contenu du donjon. Les choix prédéfinis simplifient la lecture du code.
* **Utilisation des Caractères Minuscules pour les Choix de l'Utilisateur :**
* **Avantages :** Convertir automatiquement les choix de l'utilisateur en majuscules aide à éviter des erreurs potentielles et améliore la fiabilité des fonctions.
* **Placement de la Sortie et de l'Entrée :**
* **Avantages :** Le placement automatique simplifie l'interaction pour l'utilisateur, réduisant les risques d'erreurs de placement.
* **Limitation du Dimensionnement du Couloir :**
* **Avantages :** Limiter la taille des couloirs évite des problèmes potentiels d'adaptation aux donjons et contribue à la cohérence du design.
* **Choix de Représentation du Couloir :**
* **Avantages :** L'utilisation d'un tableau bidimensionnel offre une modélisation efficace malgré les inconvénients potentiels liés à la taille du couloir.

## Vie du projet :

Pour commencer, nous avons initialement envisagé de créer un outil générant des donjons de manière aléatoire. Cependant, nous avons rapidement réalisé que cela serait trop complexe et chronophage. Nous nous sommes retrouvés bloqués dans notre démarche. Finalement, nous avons pris la décision de solliciter l'utilisateur pour les valeurs à utiliser. Nous avons consacré beaucoup d'efforts à la création des salles, investissant un temps considérable pour anticiper et corriger toutes les erreurs potentielles afin d'éviter tout dysfonctionnement du programme.

Dans la conception des salles, nous avons rapidement décidé de représenter l'espace vide avec des '\*' pour simplifier le comptage et faciliter le placement des éléments aux coordonnées désirées. En ce qui concerne les couloirs, nous avons également investi du temps. Initialement, nous avions envisagé de permettre à l'utilisateur de spécifier une taille et une orientation, puis de continuer le couloir à partir du dernier caractère placé. Cependant, nous avons réalisé qu'il serait plus simple de créer des segments de couloirs que nous pourrions assembler lors de la création du donjon, offrant ainsi une approche plus simple avec le même rendu.

Nous avons opté pour des couloirs simples représentés par '#' car ils sont plus fins, offrant ainsi plus de possibilités de mouvement, même dans de petits donjons. Ensuite, nous avons mis en place le créateur de donjon, une tâche relativement simple où nous avons réutilisé le code des salles. Au lieu d'ajouter des éléments, nous avons importé nos fichiers texte dans le donjon.

L'une de nos difficultés majeures a été l'importation dans le donjon. Bien que Satya ait tenté de nous aider sans succès, Toby a finalement réussi à résoudre le problème. Notre défi était que nous importions tous les caractères du fichier, y compris les caractères invisibles. Nous avons trouvé une solution en limitant automatiquement l'importation aux dimensions du donjon, remplaçant les éléments en dessous, ce qui s'est révélé parfait.

En conclusion, nos principales difficultés ont résidé dans la création d'une suite au couloir et dans l'importation dans le donjon. La résolution des erreurs a été la partie la plus chronophage de notre projet.

## Conclusion :

### a. Dans le Cadre du Projet

#### i. Les attentes initiales et la réalisation finale

* **Création de Salles et de Couloirs :**
* *Attentes Initiales :* La consigne suggère de dimensionner d'abord la salle avant de placer des éléments à l'intérieur. Pour les couloirs, la consigne recommande de spécifier le nombre de déplacements nécessaires et de fournir la séquence de placements.
* *Réalisation du Projet :* La création de salles et de couloirs suive ces recommandations. Des fonctions spécifiques pour dimensionner, placer des éléments, et définir des déplacements ont été implémentées.
* **Création de Donjons :**
* *Attentes Initiales :* Les donjons peuvent être créés en assemblant des salles et des couloirs. Le recyclage du plan d'une salle est possible.
* *Réalisation du Projet :* Les fonctionnalités de création de donjons est opérationnel.
* **Fonctionnalités de l'Application :**
* *Attentes Initiales :* La consigne spécifie la visualisation d'un donjon en mode texte, la modification directe d'éléments, et la sauvegarde du donjon dans un fichier.
* *Réalisation du Projet :* Le donjon est bien visible en mode texte, les éléments peuvent bien être modifier et on peut bien sauvegarder le donjon.
* **Format de Données :**
* *Attentes Initiales :* La consigne mentionne l'utilisation d'un format de fichier texte pour représenter le donjon avec des symboles spécifiques pour chaque élément.
* *Réalisation du Projet :* Notre code enregistre correctement le donjon dans un fichier texte.
* **Évaluation et Propreté du Code :**
* *Attentes Initiales :* L'évaluation inclut des critères tels que la propreté des sources, la modularité, les commentaires, et le choix des noms de fonctions et de variables.
* *Réalisation du Projet :* Nous avons utilisé des noms de variable clair et simple, nous avons commenter chaque fonction avec leurs rôles dans le code et le code est propre.

#### ii. Idées pour l'Avenir

* **Feedback Utilisateur :**

Intégrer des messages d'erreur ou d'information détaillés dans l'interface utilisateur pour expliquer les raisons derrière les échecs de placement ou d'autres opérations.

* **Amélioration de l'Interface Textuelle :**

Explorer des options pour améliorer l'interface textuelle en utilisant des caractères spéciaux, des couleurs ou des mises en page pour rendre la visualisation plus intuitive et attrayante.

* **Sauvegarde Automatique :**

Pourquoi pas avoir un système qui sauvegarde nos salles, couloirs et donjons automatiquement avec un système d’historique.

* **Améliorations Visuelles Optionnelles :**

Envisager des améliorations visuelles optionnelles, telles que l'utilisation de bibliothèques graphiques simples pour créer une interface utilisateur plus visuelle, sans compromettre la conformité aux directives du projet.

* **Génération de Donjons Aléatoires :**

Implémentez la génération aléatoire de donjons pour diversifier l'expérience utilisateur.

* **Variété de Couloirs :**

Ajoutez la possibilité de créer différents types de couloirs (larges, étroits, sinueux, etc.) pour diversifier la conception du donjon.

* **Connexion des Couloirs :**

Implémentez une méthode pour connecter automatiquement les couloirs, assurant une structure cohérente du donjon.

* **Éléments Dynamiques :**

Introduisez la possibilité d'ajouter des éléments dynamiques dans la salle, tels que des pièges activables ou des monstres mobiles.

* **Personnalisation Avancée :**

Permettez à l'utilisateur de personnaliser davantage la salle en ajoutant des paramètres avancés, comme la densité des éléments.

* **Log de Gestion des Erreurs :**

Ajoutez un mécanisme pour enregistrer les erreurs dans un fichier journal, offrant une trace détaillée des problèmes rencontrés.

* **Optimisation de la Performance :**

Recherchez des opportunités d'optimisation pour améliorer les performances, en particulier si vous travaillez avec des structures de données volumineuses.

### b. Au-delà du Projet

#### i. Leçon d'Équipe

Nous avons traversé des montagnes russes d'apprentissage, tant du point de vue technique que collaboratif. Les défis ont été des tremplins pour renforcer nos compétences individuelles et notre cohésion en tant qu'équipe.

#### ii. Potentiel d'Application Étendu

Ce projet ne se limite pas à ses premiers contours. Il nous a permis de reprendre gouts au C. On s'est beaucoup améliorer tout en s’amusant et c’est le principal.

#### iii. L'Aventure Continue

En jetant un œil vers l'avenir, on imagine des ajouts excitants et des évolutions à venir. Cela pourrait inclure de nouvelles fonctionnalités, des optimisations ou même l'exploration de nouvelles technologies. Notre projet n'est pas statique ; il évoluera avec les besoins changeants et les opportunités qui se présenteront.