

|  |
| --- |
|  |
| Conception générale |
| Projet algorithmique semestre 4 |
| GEIGUER Pierre | FERNANDO Théo |LAMBALLAIS Hugo |

|  |  |
| --- | --- |
|  | Décoratif |
| Introduction |
| Ce document de conception générale vise à définir les différentes exigences, modules, fonctions et structures qui seront implémentés dans le projet. Il fournira une vue d'ensemble du système et de ses composants, ainsi que des spécifications détaillées pour chaque module et fonction. Ce document a pour but d'aider l’équipe de développement à comprendre les exigences et les spécifications du système, et à élaborer une conception solide et cohérente qui répond aux besoins du projet.  Ce document est un outil essentiel pour assurer la communication entre les membres de l'équipe de développement, ainsi qu'avec les parties prenantes du projet. En fournissant une vue d'ensemble claire du système, ce document permettra de minimiser les risques de malentendus ou de divergences de compréhension au sein de l'équipe, et aidera à garantir que tous les membres travaillent en harmonie vers des objectifs communs.  Enfin, ce document est conçu pour être évolutif et s'adapter à l'évolution du projet au fil du temps. Les spécifications des exigences, modules, fonctions et structures peuvent être mises à jour en fonction des besoins ou des difficultés. |
|  |
|  |

|  |  |
| --- | --- |
|  | Décoratif |
| Exigences generalesExigences d’initialisation :  1. Le programme initialisera la fenêtre 2. En cas d’erreur d’initialisation un message sera affiché 3. Le programme créera un pointeur de rendu graphique 4. En cas d’erreur de création un message sera affiché 5. Le programme créera des textures lors d’un niveau 6. Une surface est initialisée avec les données de l’image correspondantes 7. En cas d’erreur d’écriture un message sera affiché 8. Une texture sera créée à partir de la surface 9. En cas d’erreur d’écriture un message sera affiché 10. Le programme initialisera l’appel de création de texte 11. En cas d’erreur d’initialisation un message sera affiché 12. Le programme lira une police d’écriture 13. En cas d’erreur de lecture un message sera affiché  Exigences du système :  1. Le système sera un personnage 2. Le système sera un astronaute 3. Le système devra pouvoir se déplacer de gauche à droite 4. Le système devra pouvoir sauter 5. Le système ne devra pas sortir de l’écran 6. Le système devra évoluer sur une image de fond 7. Le système pourra se déplacer sur un sol 8. Le système sera soumis à des collisions 9. Le système devra évoluer dans plusieurs niveaux de type platformer 10. Le système rencontrera des ennemis 11. Les ennemis feront arrêter la partie en cas de collision avec le système 12. Le système pourra finir la partie en montant dans un vaisseau  Exigences de sauvegarde :  1. L’utilisateur devra pouvoir sauvegarder sa progression dans un fichier texte 2. Le système enregistrera le nombre de morts, les pièces, le niveau, les personnages 3. Le système lira le fichier texte 4. Le texte sera des nombres au format décimal 5. Les nombres du fichier seront séparés par des retours à la ligne et des espaces 6. En cas d’erreur de lecture un message sera affiché  Exigences des menus :  1. L’utilisateur aura accès à des menus 2. Les menus proposeront différents choix liés au système 3. Les menus vérifieront les coordonnées de la souris pour connaitre le choix 4. Les choix changeront de couleur lors du survol 5. Les menus se fermeront lors d’un clic gauche sur un choix  Exigences de la progression :  1. Le programme affichera le nombre de morts 2. Le programme affichera le meilleur temps du niveau 3. Le programme affichera le temps passé dans le niveau 4. Le programme affichera le nombre de pièces récoltées |

|  |  |
| --- | --- |
|  | Décoratif |
| modules et architectureArchitecture finale: |
|  |
|  | Description des modules :Main :  * Exécute la boucle principale du programme * Contient le menu qui permet d’accéder aux différents niveaux et fonctionnalités du programme  Save :  * Module responsable de la sauvegarde et du chargement des données de la partie * Ce module reçoit des données venant du main et les enregistre dans un fichier texte pour la sauvegarde * Ce module renvoie des données au main lors du chargement d’une partie  niveau :  * Ce module contient les informations d’un niveau sous forme d’une fonction appelée par le main * Le module est chargé d’appeler les fonctions détectant les collisions entre le personnage et les éléments du niveau * Ce module gère les déplacements du personnage  Fonctions  * Ce module contient le moteur du jeu * Ce module est chargé de détecter les collisions et de modifier la position du personnage en cas de collision  Affichage  * Ce module contient le moteur graphique du jeu * Toutes les fonctions chargées de l’affichage sont définies dans ce module et sont appelées par les autres modules  Structure  * Comme son nom l’indique ce module permet de définir toutes les structures utilisées par le programme de manière centralisée  Init  * Ce module contient les fonctions servant à initialiser correctement différents éléments |

|  |
| --- |
| Décoratif |
| Fonctions principales Les fonctions principales sont :   * Les niveaux qui rassemblent toutes les fonctions d’affichage de compteurs et de menus. Ils comportent également tout le code pour faire déplacer le personnage, le fond d’écran, les objets et les ennemis. * Les menus qui comportent toutes les boucles pour afficher les différents choix disponibles à l’utilisateur. Il y a les menus de pause en jeu, de mort, de victoire, d’affichage des commandes, de choix de personnages ainsi que le menu principal. * Sauvegarde / chargement qui enregistrent et récupèrent les données importantes du jeu. * Collisions pour détecter les collisions entre le bord de l’écran et le personnage, les blocs et le personnage, les ennemis et le personnage. |
|  |

|  |
| --- |
| Décoratif |
| Structures Une structure Astronaute qui contient toutes les textures d’un astronaute, ses positions, sa vitesse maximale, sa vitesse de saut.  Une structure Ennemi qui contient toutes les textures d’un ennemi et ses positions.  Une structure Plateforme qui contient la texture du bloc et ses positions.  Une structure Vaisseau qui contient ses positions et les textures correspondantes. |