Liste des fonctionnalités

Deux modules :

|  |  |
| --- | --- |
| **Création de labyrinthe** | **Localisation dans un labyrinthe** |
| **Créer un labyrinthe de taille AxB :**  Pour ce faire, l’utilisateur de l’application doit indiquer les dimensions A et B dans deux JSpinner distincts. Puis il doit valider son choix en cliquant sur un JButton « Créer »  Les dimensions A et B sont limitées à une taille maximale de 50x50 | **Positionner le robot :**  Initialement, le robot est positionné sur la case entrée du labyrinthe. |
| **Créer une case vide :**  Une fois le labyrinthe créé aux dimensions souhaitées, l’utilisateur doit construire le labyrinthe case par case.  S’il veut ajouter une case vide, c’est-à-dire une case où il est possible de marcher dessus, il doit cliquer sur le JButton « Vide » et ensuite sélectionner les cases qu’il souhaite. | **Déplacer le robot vers l’avant :**  Il est possible de faire avancer le robot de deux manières :  A l’aide d’un JButton « Avancer » qui met à jour la position du robot, ou à l’aide de la touche « Flèche du haut » du clavier. |
| **Créer une case Mur :**  Une fois le labyrinthe créé aux dimensions souhaitées, l’utilisateur doit construire le labyrinthe case par case.  S’il veut ajouter une case Mur, c’est-à-dire une case où il est impossible de marcher dessus, il doit cliquer sur le JButton « Mur » et ensuite sélectionner les cases qu’il souhaite. | **Rotation du robot de 90° vers la droite**  Le robot reste sur sa case, et il effectue une rotation de 90° vers la droite.  Deux possibilités d’effectuer la rotation :  A l’aide d’un JButton « Rotation droite », ou de la touche clavier « Flèche de droite » |
| **Créer une case entrée :**  Une fois le labyrinthe créé, l’utilisateur doit positionner une entrée. Il doit cliquer sur le bouton « entrée » et ensuite sélectionner une case du labyrinthe. Si une entrée a déjà été positionnée, celle-ci est déplacée à son nouvel emplacement. | **Rotation du robot de 90° vers la gauche**  Le robot reste sur sa case, et il effectue une rotation de 90° vers la gauche.  Deux possibilités d’effectuer la rotation :  A l’aide d’un JButton « Rotation droite », ou de la touche clavier « Flèche de droite » |
| **Créer une case sortie :**  Une fois le labyrinthe créé, l’utilisateur doit positionner une sortie. Il doit cliquer sur le bouton « entrée » et ensuite sélectionner une case du labyrinthe. Si une sortie a déjà été positionnée, celle-ci est déplacée à son nouvel emplacement. | **Afficher/Masquer vue globale**  Lorsque l’utilisateur clique sur le JButton « Masquer/Afficher globale », la vue du labyrinthe se masque si celle-ci est visible, ou s’affiche si celle-ci est masquée. |
| **Sauvegarder un labyrinthe :**  Une fois le labyrinthe terminé, il est possible de le sauvegarder pour le réutiliser plus tard en allant dans le menu Fichier puis Sauvegarder (il est également possible d’utiliser le raccourci clavier Ctrl+S). Cependant, la sauvegarde est possible seulement s’il y a une entrée et une sortie. La sauvegarde se fait sous la forme d’un fichier dont l’extension est .maze.  Exemple de fichier en .maze :  0 : Vide 1 : Mur 3 : Sortie 2 : Entrée    111111  200001  100003  111111 | **Afficher vue locale 3D**  Lorsque l’utilisateur clique sur le JButton « Vue 3D », la vue change et affiche la vue 3D.  Cette vue est une représentation 3D de ce que voit le robot à l’instant t. |
| **Afficher vue locale Cases**  Lorsque l’utilisateur clique sur le JButton « Vue cases », la vue change et affiche la vue cases.  Cette vue est une représentation 2D des cases adjacentes qui indique si une case voisine est une case Vide ou une case Mur. |
| **Charger un labyrinthe :**  Pour charger un labyrinthe, il faut aller dans le menu Fichier, puis Ouvrir (il est également possible d’utiliser le raccourci Ctrl+O). Ensuite, une fenêtre s’ouvre et nous demande l’emplacement du fichier .maze.  Lorsque l’utilisateur sectionne un fichier .maze, celui-ci est chargé dans l’application, il est alors possible de le modifier dans le panneau Création. | **Afficher/Masquer position du robot**  Lorsque l’utilisateur clique sur le JButton « Masquer/Afficher robot », le robot s’affiche sur la vue globale s’il est masqué, ou se masque s’il est visible.  Lorsque le robot est visible, sa position est actualisée lorsqu’il y a un déplacement effectué. |
| **Faire un nouveau labyrinthe :**  Pour faire un nouveau labyrinthe, il faut, soit lancer l’application, et dans ce cas, le processus de création est disponible, soit, si l’utilisateur a commencé à créer un labyrinthe, et qu’il souhaite en faire un nouveau, il clique sur Fichier, puis Nouveau (il est possible d’utiliser le raccourci clavier CTRL+N). Une fenêtre est alors affichée pour proposer à l’utilisateur de sauvegarder son labyrinthe en cours, et il peut ensuite commencer la création d’un nouveau labyrinthe. | **Lancer un algorithme de localisation :** |
| **Quitter l’application :**  Pour fermer l’application, l’utilisateur peut aller dans Fichier, puis Quitter (ou utiliser le raccourci Ctrl+Q) ou alors fermer la fenêtre. |  |
| **Remplir entièrement de cases :**  Si l’utilisateur souhaite remplir entièrement son labyrinthe de cases, il doit cliquer sur le bouton « Remplir entièrement ». |  |
| **Faire un ilot :** |  |
| **Creuser :**  Lorsque l’utilisateur clique sur le bouton « Creuser » le labyrinthe se remplit entièrement de case Vide. |  |
| **Générer un labyrinthe parfait :**  Un labyrinthe parfait est un labyrinthe où il existe uniquement un seul chemin pour aller d’une case à une autre.  Lorsque l’utilisateur clique sur le bouton « Générer parfait », un algorithme de génération est utilisé (Algorithme par fusion), et les cases du labyrinthe sont remplies en fonction de l’algorithme : le labyrinthe devient parfait. |  |
| **Générer un labyrinthe imparfait :**  Un labyrinthe imparfait est un labyrinthe où il existe plusieurs chemins pour aller d’une case à une autre.  Lorsque l’utilisateur clique sur le bouton « Générer imparfait », un algorithme de génération est utilisé (celui de génération d’algorithme parfait, où plusieurs murs sont détruits) et les cases du labyrinthe sont remplies en fonction de l’algorithme : le labyrinthe devient imparfait. |  |

**Globale**

Redimensionnement