Documentation technique

Mr.Bourdon&co

Produit : Le programme programmé programmant

Version : 2.8.3

Date : 17 Feb 2025 13:37:57

src1.c

Table des matières :

DEFINES

#define CORPS_SNAKE '-'	corps du snake
#define TETE_SNAKE 'B'	tete du snake
#define DU_MANGER '*'	pomme à manger pour le snake (score)
#define CASE_VIDE ' '	case vide

STRUCTURES

```
struct uneCellule {
     int ligne;                  coordonnées     de
     int colonne;           ligne de la grille
      struct uneCellule          coordonnées     de
* suiv;                     colonne de la grille
    };                           boucle
typedef          struct
uneCellule                       structure de grille
uneCellule;                 du jeu
```

```
struct unSnake {
        uneCellule     *         tete du snake
teteSnake;                       queue du snake
        uneCellule     *
queueSnake;                      structure
    };                      d'apparence          du
typedef          struct     snake
unSnake unSnake;
```

```
struct    uneDirection
{                                direction  de  ligne
     int ligne;             du snake
     int colonne;                direction          de
    };                      colonne du snake
typedef          struct
uneDirection                     structure          de
uneDirection;           direction du snake
```

GLOBALES

```
char    **    grille    =    grille
NULL;                        ligne de grille
int nbLignes = 0;            colonne de grille
int nbColonnes = 0;          mode lolilol
int inutile = 0;
```

FONCTIONS

unSnake creerSnake()	\brief crée le snake \detail donne une tete et une queue au snake et fait en sorte que la queue suivent la tete du snake \return unSnake le snake
void ajouterEnTete (unSnake * snake, int ligne, int colonne,int * aMange, int * fail)	\brief permet au snake de bouger \detail Redéfini la tête du snake aux coordonnées indiquées gère les collisions avec les bords (sauf si mode lolilol) et le snake vérifie si le snake mange quelque chose \param unSnake le snake \param int ligne de position du snake \param int colonne de position du snake \param int coordonner du manger \param int état de défaite

void supprimerQueue(unSnake * snake)	\brief Supprime la queue du snake à sa mort \detail Supprime la queue du snake à sa mort \param unSnake le snake
void initGrille()	\brief charge une grille vide \detail charge une grille vide
void afficherGrille(unSnake snake)	\brief affiche la grille \detail affiche la grille et le snake de base \param unSnake le snake

void gererEvenement(unSnake * snake, int touche, int * fail, uneDirection * direction,int * aMange)	\brief permet les controles de mouvement du snake \detail change de direction au snake en fonction des touche appuier \param unSnake le snake \param int touche appuier \param int état de défaite \param uneDirection direction vers laquelle le snake vas \param int cohordonnée du manger
void genererDuManger(char ** grille)	\brief genere une case contenent 'DU_MANGER' à des coordonnées aléatoires \detail donne de nouvelle cohordonnée au manger quand le précedant ce fait manger \param char grille de jeu
void printFail()	\brief affiche 'FAIL' a l'écran \detail affiche 'FAIL' a l'écran

int main (int argc, char * argv [])	\brief fonction principale du jeu \detail fonction principale du jeu \param int je sais pas \param char je sais pas \return 0

src2.c

Table des matières :

DEFINES

#define HAUTEUR 6	Hauteur du damier
#define LARGEUR 7	Largeur du damier

STRUCTURES

GLOBALES

char damier[LARGEUR] [HAUTEUR];	Tableau représentant le damier
int joueur = 1;	Variable pour suivre le joueur actuel
int nbtokens = HAUTEUR * LARGEUR;	Nombre de jetons restants
char symbols[2];	Symboles utilisés pour les joueurs

FONCTIONS

src3.c

Table des matières :

DEFINES

```
#define N 6
```

STRUCTURES

```
struct hello{
                    char        Un message venu
message[20];                de l'autre monde !
    };
typedef   struct   hello
hello;
```

GLOBALES

```
const int MAX = 30;      C'est juste le max qui
                         sert à rien.
```

FONCTIONS

void sayHello(hello hello)	\brief Une fonction qui demande et affiche un message. \detail Des messages venus des Enfers peuvent être envoyés ici. \param hello C'est la structure qui contient le message :O