

Jeu de Boggle

1

Généré par Doxygen 1.8.1.2

Mardi Avril 30 2013 10 :29 :28

Table des matières

1	Index des structures de données	1
1.1	Structures de données	1
2	Index des fichiers	3
2.1	Liste des fichiers	3
3	Documentation des structures de données	5
3.1	Référence de la structure Case	5
3.1.1	Description détaillée	5
3.1.2	Documentation des champs	5
3.1.2.1	i	5
3.1.2.2	j	5
3.2	Référence de la structure Dico	5
3.2.1	Description détaillée	6
3.2.2	Documentation des champs	6
3.2.2.1	dico	6
3.2.2.2	marqueurs	6
3.3	Référence de la structure Jeu	6
3.3.1	Description détaillée	6
3.3.2	Documentation des champs	6
3.3.2.1	dico	6
3.3.2.2	plateau	6
3.3.2.3	solutionUtilisateur	6
3.3.2.4	timestampDepart	7
3.4	Référence de la structure Plateau	7
3.4.1	Description détaillée	7
3.4.2	Documentation des champs	7
3.4.2.1	grille	7
3.4.2.2	solution	7
3.4.2.3	tailleGrille	7
3.5	Référence de la structure Solution	7
3.5.1	Description détaillée	8

3.5.2	Documentation des champs	8
3.5.2.1	mots	8
3.5.2.2	nbMots	8
4	Documentation des fichiers	9
4.1	Référence du fichier dictionnaire.c	9
4.1.1	Description détaillée	9
4.1.2	Documentation des fonctions	9
4.1.2.1	dictionnaire_nouveau	9
4.1.2.2	dictionnaire_rechercheDichotomique	10
4.1.2.3	dictionnaire_motDansDico	10
4.2	Référence du fichier interfaceNcurses.c	10
4.2.1	Description détaillée	10
4.3	Référence du fichier jeu.c	11
4.3.1	Description détaillée	11
4.3.2	Documentation des fonctions	11
4.3.2.1	jeu_compteurClaque	11
4.3.2.2	jeu_lancer	11
4.3.2.3	jeu_lancerModeTexte	12
4.3.2.4	jeu_nouveau	12
4.3.2.5	jeu_proposerMot	12
4.3.2.6	jeu_stopper	12
4.3.2.7	jeu_tempsRestant	12
4.4	Référence du fichier plateau.c	13
4.4.1	Description détaillée	13
4.4.2	Documentation des fonctions	13
4.4.2.1	plateau_choisirLettre	13
4.4.2.2	plateau_detruire	13
4.4.2.3	plateau_nouveau	13
4.4.2.4	plateau_probaLettre	14
4.4.2.5	plateau_remplirGrilleAleatoire	14
4.4.2.6	plateau_remplirGrillePredefinie	14
4.5	Référence du fichier resolveur.c	14
4.5.1	Description détaillée	14
4.5.2	Documentation des fonctions	15
4.5.2.1	recurse	15
4.5.2.2	resolveur	15
4.6	Référence du fichier util.c	15
4.6.1	Description détaillée	16
4.6.2	Documentation des fonctions	16

4.6.2.1	util_affichageDebug	16
4.6.2.2	util_conversionTemps	16
4.6.2.3	util_deplacerCurseurDunMot	16
4.6.2.4	util_echanger	16
4.6.2.5	util_isInArray	16
4.6.2.6	util_nbAleatoire	17
4.6.2.7	util_quickSort	17
4.6.2.8	util_substr	17
4.6.2.9	util_supprimerAccents	17
4.6.2.10	util_uppercase	18

Chapitre 1

Index des structures de données

1.1 Structures de données

Liste des structures de données avec une brève description :

Case	Une case de la grille	5
Dico	Dictionnaire Contient le dictionnaire du jeu	5
Jeu	Structure permettant de jouer au Boggle Structure contenant toutes les informations nécessaire au bon déroulement du jeu	6
Plateau	Grille du jeu Contient une grille permettant de jouer au Boggle	7
Solution	Solution du jeu Contient une solution pour une grille de Boggle	7

Chapitre 2

Index des fichiers

2.1 Liste des fichiers

Liste de tous les fichiers documentés avec une brève description :

case.h	??
dictionnaire.c	
Gestion du dictionnaire Fonctions de gestion du dictionnaire, le dictionnaire doit être dans un fichier texte avec un mot différent par ligne	9
dictionnaire.h	??
interfaceNcurses.c	
Affichage avec la bibliothèque Ncurses Affiche et permet de jouer au jeu de Boggle avec un affichage utilisant la bibliothèque Ncurses	10
interfaceNcurses.h	??
interfaceTexte.h	??
jeu.c	
Gestion du jeu de Boggle Les fonctions permettant de gérer le Jeu de Boggle	11
jeu.h	??
plateau.c	
Gestion du plateau de Jeu Fonctions se rapportant à un plateau de jeu, sera sert à gérer la grille de Boggle	13
plateau.h	??
resolveur.c	
Gestion du résolveur Permet de résoudre une grille de Boggle, une fois résolue, la solution se trouve dans le module Solution	14
resolveur.h	??
solution.h	??
util.c	
Fonctions utiles à l'ensemble du projet Toutes les fonctions de bases utiles au projet, ces fonctions travaillent sur des types de base et ne sont pas spécifiques au projet, ce module permet de mieux organiser le code	15
util.h	??

Chapitre 3

Documentation des structures de données

3.1 Référence de la structure Case

Une case de la grille.

```
#include <case.h>
```

Champs de données

- unsigned char `i`
- unsigned char `j`

3.1.1 Description détaillée

Une case de la grille.

3.1.2 Documentation des champs

3.1.2.1 unsigned char Case : `i`

Abscisse

3.1.2.2 unsigned char Case : `j`

ordonnée

La documentation de cette structure a été générée à partir du fichier suivant :

- case.h

3.2 Référence de la structure Dico

Dictionnaire Contient le dictionnaire du jeu.

```
#include <dictionnaire.h>
```

Champs de données

- FILE * `dico`
- long int `marqueurs` [32]

3.2.1 Description détaillée

Dictionnaire Contient le dictionnaire du jeu.

3.2.2 Documentation des champs

3.2.2.1 FILE* Dico : :dico

Fichier contenant le dictionnaire, celui-ci doit être un fichier texte avec un mot par ligne

3.2.2.2 long int Dico : :marqueurs[32]

Contient les positions de chacune des lettres de l'alphabet dans le fichier

La documentation de cette structure a été générée à partir du fichier suivant :

– dictionnaire.h

3.3 Référence de la structure Jeu

Structure permettant de jouer au Boggle Structure contenant toutes les informations nécessaire au bon déroulement du jeu.

```
#include <jeu.h>
```

Champs de données

- [Plateau](#) plateau
- [Dico](#) dico
- [time_t](#) timestampDepart
- [Solution](#) solutionUtilisateur

3.3.1 Description détaillée

Structure permettant de jouer au Boggle Structure contenant toutes les informations nécessaire au bon déroulement du jeu.

Voir également

[Dico](#), [Plateau](#), [Solution](#)

3.3.2 Documentation des champs

3.3.2.1 Dico Jeu : :dico

Le dictionnaire

3.3.2.2 Plateau Jeu : :plateau

Le plateau de [Jeu](#)

3.3.2.3 Solution Jeu : :solutionUtilisateur

La solution que propose l'utilisateur, celle-ci peut ne pas être complète, mais tous les mots présents dans cette solution sont présent dans la grille

3.3.2.4 time_t Jeu : :timestampDepart

Le timestamp auquel on a commencé à jouer

La documentation de cette structure a été générée à partir du fichier suivant :
– jeu.h

3.4 Référence de la structure Plateau

Grille du jeu Contient une grille permettant de jouer au Boggle.

```
#include <plateau.h>
```

Champs de données

- char ** grille
- unsigned int tailleGrille
- int ** grid
- Solution solution

3.4.1 Description détaillée

Grille du jeu Contient une grille permettant de jouer au Boggle.

3.4.2 Documentation des champs

3.4.2.1 char** Plateau : :grille

La grille de boggle

3.4.2.2 Solution Plateau : :solution

La solution complète de cette grille de Boggle

3.4.2.3 unsigned int Plateau : :tailleGrille

La taille de la grille

La documentation de cette structure a été générée à partir du fichier suivant :
– plateau.h

3.5 Référence de la structure Solution

Solution du jeu Contient une solution pour une grille de Boggle.

```
#include <solution.h>
```

Champs de données

- char ** mots
- unsigned int nbMots

3.5.1 Description détaillée

[Solution](#) du jeu Contient une solution pour une grille de Boggle.

3.5.2 Documentation des champs

3.5.2.1 `char**` Solution : :mots

Les mots présents dans la solution

3.5.2.2 `unsigned int` Solution : :nbMots

Le nombre de mots que contient la solution

La documentation de cette structure a été générée à partir du fichier suivant :

– solution.h

Chapitre 4

Documentation des fichiers

4.1 Référence du fichier dictionnaire.c

Gestion du dictionnaire Fonctions de gestion du dictionnaire, le dictionnaire doit être dans un fichier texte avec un mot différent par ligne.

```
#include <string.h>
#include "dictionnaire.h"
#include "util.h"
```

Fonctions

- [Dico dictionnaire_nouveau](#) (const char *pNomFichier)
Créer un nouveau dictionnaire à partir d'un fichier.
- int [dictionnaire_motDansDico](#) ([Dico](#) pDictionnaire, char *pMot)
Permet de savoir si un mot est présent dans le dictionnaire.
- void [dictionnaire_rechercheDichotomique](#) ([Dico](#) pDictionnaire, char *pMotAChercher, char *pMotLePlusProche)
Effectue une recherche dichotomique de pMotAChercher dans le dictionnaire.

4.1.1 Description détaillée

Gestion du dictionnaire Fonctions de gestion du dictionnaire, le dictionnaire doit être dans un fichier texte avec un mot différent par ligne.

4.1.2 Documentation des fonctions

4.1.2.1 [Dico dictionnaire_nouveau](#) (const char * *pNomFichier*)

Créer un nouveau dictionnaire à partir d'un fichier.

Paramètres

<i>pNomFichier</i>	Le fichier contenant les mots du dictionnaire.
--------------------	--

Renvoie

Le nouveau dictionnaire

4.1.2.2 void dictionnaire_rechercheDichotomique (Dico *pDictionnaire*, char * *pMotAChercher*, char * *pMotLePlusProche*)

Effectue une recherche dichotomique de *pMotAChercher* dans le dictionnaire.

Paramètres

<i>pDictionnaire</i>	Le dictionnaire dans lequel chercher
<i>pMotAChercher</i>	Le mot à chercher
<i>pMotLePlusProche</i>	Le mot le plus proche trouvé, si <i>pMotLePlusProche</i> == <i>pMotAChercher</i> , il est présent dans le dictionnaire

4.1.2.3 int dictionnaire_motDansDico (Dico *pDictionnaire*, char * *pMot*)

Permet de savoir si un mot est présent dans le dictionnaire.

Paramètres

<i>pDictionnaire</i>	Le dictionnaire dans lequel chercher
<i>pMot</i>	Le mot à chercher

Renvoie

0 si le mot est absent, 1 si des mots commencent par *pMot* et 10 si le mot exact est trouvé.

4.2 Référence du fichier interfaceNcurses.c

Affichage avec la bibliothèque Ncurses Affiche et permet de jouer au jeu de Boggle avec un affichage utilisant la bibliothèque Ncurses.

```
#include <ncurses.h>
#include <panel.h>
#include <stdlib.h>
#include <string.h>
#include "interfaceNcurses.h"
#include "plateau.h"
#include "jeu.h"
#include "util.h"
```

Fonctions

- void **interfaceNcurses_afficherSolution** (const [Solution](#) pSolution)
- WINDOW * **interfaceNcurses_initialiser** (void)
- void **interfaceNcurses_afficherGrille** (const [Plateau](#) pPlateau, const [Case](#) pSelectedCase, const [Case](#) *pUsedCase, const int pLgUsedCase)
- void **interfaceNcurses_terminer** (WINDOW *fenetre)
- void **jeu_lancerModeNcurses** ([Jeu](#) pJeu)

4.2.1 Description détaillée

Affichage avec la bibliothèque Ncurses Affiche et permet de jouer au jeu de Boggle avec un affichage utilisant la bibliothèque Ncurses.

4.3 Référence du fichier jeu.c

Gestion du jeu de Boggle Les fonctions permettant de gérer le [Jeu](#) de Boggle.

```
#include <stdlib.h>
#include <string.h>
#include "util.h"
#include "resolveur.h"
#include "jeu.h"
#include "plateau.h"
#include "dictionnaire.h"
#include "interfaceTexte.h"
```

Fonctions

- `time_t jeu_tempsRestant (const Jeu pJeu)`
Retourne le nombre de secondes restantes avant la fin du jeu.
- `bool jeu_compteurClaque (const Jeu pJeu)`
Retourne vrai si le temps est écoulé faux sinon.
- `Jeu jeu_nouveau (const char *pNomDico, const unsigned char pTaillePlateau)`
Créer un nouveau jeu.
- `void jeu_lancer (Jeu *pJeu)`
Lance le jeu.
- `_Bool jeu_proposerMot (Jeu *pJeu, const char *pMot)`
Proposer un mot.
- `void jeu_stopper (Jeu pJeu)`
Fin du jeu.
- `void jeu_lancerModeTexte (Jeu pJeu)`
Lance le jeu en mode texte.

4.3.1 Description détaillée

Gestion du jeu de Boggle Les fonctions permettant de gérer le [Jeu](#) de Boggle.

Voir également

[Plateau](#), [Resolveur](#), [Dico](#), [Solution](#)

4.3.2 Documentation des fonctions

4.3.2.1 `bool jeu_compteurClaque (const Jeu pJeu)`

Retourne vrai si le temps est écoulé faux sinon.

Paramètres

<code>pJeu</code>	Le jeu pour lequel on veut savoir si le temps est écoulé
-------------------	--

Renvoie

Vrai si le temps est écoulé faux sinon

4.3.2.2 `void jeu_lancer (Jeu * pJeu)`

Lance le jeu.

Paramètres

<i>pJeu</i>	Le jeu à lancer
-------------	-----------------

4.3.2.3 void jeu_lancerModeTexte (Jeu *pJeu*)

Lance le jeu en mode texte.

Paramètres

<i>pJeu</i>	le jeu à lancer
-------------	-----------------

4.3.2.4 Jeu jeu_nouveau (const char * *pNomDico*, const unsigned char *pTaillePlateau*)

Créer un nouveau jeu.

Paramètres

<i>pNomDico</i>	Le dictionnaire utilisé dans le Jeu
<i>pTaillePlateau</i>	La taille du plateau à créer

Renvoie

Le nouveau [Jeu](#)

4.3.2.5 _Bool jeu_proposerMot (Jeu * *pJeu*, const char * *pMot*)

Proposer un mot.

Paramètres

<i>pJeu</i>	Le jeu pour lequel on veut proposer un mot
<i>pMot</i>	Le mot proposé

Renvoie

Vrai si le mot est validé

4.3.2.6 void jeu_stopper (Jeu *pJeu*)

Fin du jeu.

Paramètres

<i>pJeu</i>	Le jeu à terminer
-------------	-------------------

4.3.2.7 time_t jeu_tempsRestant (const Jeu *pJeu*)

Retourne le nombre de secondes restantes avant la fin du jeu.

Paramètres

<i>pJeu</i>	Le jeu pour lequel on veut savoir le temps restant
-------------	--

Renvoie

Le nombre de secondes restantes

4.4 Référence du fichier plateau.c

Gestion du plateau de [Jeu](#) Fonctions se rapportant à un plateau de jeu, sera sert à gérer la grille de Boggle.

```
#include <stdlib.h>
#include "solution.h"
#include "plateau.h"
#include "util.h"
```

Fonctions

- [Plateau plateau_nouveau](#) (const unsigned char pTailleGrille)
Créer un nouveau plateau.
- void [plateau_detruire](#) ([Plateau](#) *pPlateau)
Détruit le plateau pPlateau.
- void [plateau_remplirGrilleAleatoire](#) ([Plateau](#) *pPlateau)
Rempli une grille aléatoirement dans le plateau.
- void [plateau_remplirGrillePredefinie](#) ([Plateau](#) *pPlateau)
*Rempli le plateau avec une grille prédéfinie de taille 4*4 E D R C A N V C I R Q A E B R U.*
- char [plateau_choisirLettre](#) (void)
Choisi une lettre avec la probabilité des lettres en fonctions de leurs importance d'apparition dans la langue française.
- double [plateau_probaLettre](#) (const char pLettre)
Retourne la probabilité d'apparition de la lettre passée en paramètre.

4.4.1 Description détaillée

Gestion du plateau de [Jeu](#) Fonctions se rapportant à un plateau de jeu, sera sert à gérer la grille de Boggle.

4.4.2 Documentation des fonctions

4.4.2.1 char plateau_choisirLettre (void)

Choisi une lettre avec la probabilité des lettres en fonctions de leurs importance d'apparition dans la langue française.

Renvoie

La lettre choisie

4.4.2.2 void plateau_detruire (Plateau * pPlateau)

Détruit le plateau pPlateau.

Paramètres

<i>pPlateau</i>	Le plateau à détruire
-----------------	-----------------------

4.4.2.3 Plateau plateau_nouveau (const unsigned char pTailleGrille)

Créer un nouveau plateau.

Paramètres

<i>pTailleGrille</i>	La taille de la grille à créer
----------------------	--------------------------------

Renvoie

Le nouveau plateau

4.4.2.4 double plateau_probaLettre (const char pLettre)

Retourne la probabilité d'apparition de la lettre passée en paramètre.

Paramètres

<i>pLettre</i>	La lettre pour laquelle retourner la probabilité
----------------	--

Renvoie

La probabilité

4.4.2.5 void plateau_remplirGrilleAleatoire (Plateau * pPlateau)

Rempli une grille aléatoirement dans le plateau.

Paramètres

<i>pPlateau</i>	Le plateau à remplir
-----------------	----------------------

4.4.2.6 void plateau_remplirGrillePredefinie (Plateau * pPlateau)

Rempli le plateau avec une grille prédéfinie de taille 4*4 E D R C A N V C I R Q A E B R U.

Paramètres

<i>pPlateau</i>	Le plateau à remplir
-----------------	----------------------

4.5 Référence du fichier resolveur.c

Gestion du résolveur Permet de résoudre une grille de Boggle, une fois résolue, la solution se trouve dans le module [Solution](#).

```
#include <string.h>
#include "dictionnaire.h"
#include "resolveur.h"
#include "solution.h"
```

Fonctions

- void [recurse](#) (Plateau *pPlateau, int x, int y, int depth, char *choices, [Dico](#) pDico)
Fonction privée récursive permettant de résoudre une grille de Boggle.
- void [resolveur](#) (Plateau *pPlateau, [Dico](#) pDico)
Résoud une grille de boggle.

4.5.1 Description détaillée

Gestion du résolveur Permet de résoudre une grille de Boggle, une fois résolue, la solution se trouve dans le module [Solution](#).

4.5.2 Documentation des fonctions

4.5.2.1 void recurse (Plateau * pPlateau, int x, int y, int depth, char * choices, Dico pDico)

Fonction privée récursive permettant de résoudre une grille de Boggle.

Paramètres

<i>pPlateau</i>	Le plateau à résoudre
<i>x</i>	L'abscisse de la lettre pour laquelle on part
<i>y</i>	L'ordonnée de la lettre pour laquelle on part
<i>depth</i>	La profondeur à laquelle on se trouve
<i>choices</i>	Le mot formé actuellement
<i>pDico</i>	Le dictionnaire dans lequel chercher les mots

4.5.2.2 void resolveur (Plateau * pPlateau, Dico pDico)

Résout une grille de boggle.

Paramètres

<i>pPlateau</i>	Le plateau à résoudre
<i>choices</i>	
<i>pDico</i>	le dictionnaire

4.6 Référence du fichier util.c

Fonctions utiles à l'ensemble du projet Toutes les fonctions de bases utiles au projet, ces fonctions travaillent sur des types de base et ne sont pas spécifiques au projet, ce module permet de mieux organiser le code.

```
#include <string.h>
#include <wchar.h>
#include <stdlib.h>
#include <stdbool.h>
#include "util.h"
#include "jeu.h"
#include "case.h"
```

Fonctions

- void [util_affichageDebug](#) (const char *pNomFonction, const char *pChaine)
Affiche une chaîne de caractère sur la console uniquement si le define MODE_DEBUG est vrai.
- void [util_afficherTableAscii](#) (void)
Affiche la tableau ASCII.
- char * [util_supprimerAccents](#) (const char *pChaine)
Supprime les accents d'une chaîne de caractère.
- void [util_uppercase](#) (char *pChaine)
Modifie la chaîne de caractère afin qu'elle soit en majuscule.
- char [util_nbAleatoire](#) (const char pDebut, const char pFin)
Retourne un nombre aléatoire entre pDebut et pFin.
- int [util_substr](#) (const char *chaîne, int debut, int fin, char *result)
Crée une sous-chaîne de caractère de chaîne depuis début jusqu'à fin.
- void [util_echanger](#) (char **tableau, int a, int b)
Echange deux variables dans un tableau de chaîne de caractères.
- void [util_quickSort](#) (char **tableau, int debut, int fin)
Effectue un tri sur une portion d'un tableau de chaîne de caractère s.
- void [util_deplacerCurseurDunMot](#) (FILE *pFichier, const int pSens)
Déplace le curseur dans le fichier au début du mot précédent ou suivant en fonction de pSens.

- `_Bool util_isInArray` (const `Case` *pTableau, const int pTaille, const `Case` pCase)
Retourne vrai si la case pCase est présente dans le tableau.
- `void util_conversionTemps` (const time_t pTimestamp, int *pMinutes, int *pSecondes)
Convertis un nombre de secondes en minutes et secondes.

4.6.1 Description détaillée

Fonctions utiles à l'ensemble du projet Toutes les fonctions de bases utiles au projet, ces fonctions travaillent sur des types de bae et ne sont pas spécifiques au projet, ce module permet de mieux organiser le code.

4.6.2 Documentation des fonctions

4.6.2.1 `void util_affichageDebug (const char * pNomFonction, const char * pChaine) [inline]`

Affiche une chaîne de caractère sur la console uniquement si le define `MODE_DEBUG` est vrai.

Paramètres

<code>pNomFonction</code>	Le nom de la fonction de laquelle est appelée cette fonction
<code>pChaine</code>	La chaîne de caractère à afficher

4.6.2.2 `void util_conversionTemps (const time_t pTimestamp, int * pMinutes, int * pSecondes)`

Convertis un nombre de secondes en minutes et secondes.

Paramètres

<code>pTimestamp</code>	Le nombre de secondes à convertir
<code>pMinutes</code>	Le nombre de minutes résultat
<code>pSecondes</code>	Le nombre de secondes résultat

4.6.2.3 `void util_deplacerCurseurDunMot (FILE * pFichier, const int pSens)`

Déplace le curseur dans le fichier au début du mot précédent ou suivant en fonction de pSens.

Paramètres

<code>pFichier</code>	Le fichier sur lequel s'applique le curseur. Celui-ci doit être ouvert en lecture
<code>pSens</code>	le sens du déplacement <code>MOT_PRECEDENT</code> ou <code>MOT_SUIVANT</code>

4.6.2.4 `void util_echanger (char ** tableau, int a, int b)`

Echange deux variables dans un tableau de chaîne de caractères.

Paramètres

<code>tableau</code>	Le tableau sur lequel on effectue l'échange
<code>a</code>	La première valeur à échanger
<code>b</code>	La seconde valeur

4.6.2.5 `_Bool util_isInArray (const Case * pTableau, const int pTaille, const Case pCase)`

Retourne vrai si la case pCase est présente dans le tableau.

Paramètres

<i>pTableau</i>	Le tableau sur lequele s'effectue le test
<i>pTaille</i>	La taille du tableau
<i>pCase</i>	La case à chercher

Renvoie

Vrai si la case existe dans le tableau.

4.6.2.6 char util_nbAleatoire (const char *pDebut*, const char *pFin*)

Retourne un nombre aléatoire entre *pDebut* et *pFin*.

Paramètres

<i>pDebut</i>	la borne inférieur
<i>pFin</i>	la borne supérieur

Renvoie

Le nombre généré

4.6.2.7 void util_quickSort (char ** *tableau*, int *debut*, int *fin*)

Effectue un tri sur une porition d'un tableau de chaine de caractère s.

Ce tri utilise l'algorithme du tri rapide.

Paramètres

<i>tableau</i>	Le tableau à trier
<i>debut</i>	Le début du tableau à trier
<i>fin</i>	La fin du tableau à trier

4.6.2.8 int util_substr (const char * *chaine*, int *debut*, int *fin*, char * *result*)

Créer une sous-chaine de caractère de chaine depuis début jusqu'à fin.

Paramètres

<i>chaine</i>	La chaine sur laquelle s'applique la fonction
<i>debut</i>	Le début du découpage
<i>fin</i>	La fin du découpage
<i>result</i>	La chaine résultat

Renvoie

La taille de la chaine résultat

4.6.2.9 char* util_supprimerAccents (const char * *pChaine*)

Supprime les accents d'une chaine de caratère.

Paramètres

<i>pChaine</i>	La chaine de caractère avec les accents
----------------	---

Renvoie

La nouvelle chaîne de caractère sans accents

4.6.2.10 void util_uppercase (char * *pChaine*)

Modifie la chaîne de caractère afin qu'elle soit en majuscule.

Paramètres

<i>pChaine</i>	La chaîne de caractère à modifier
----------------	-----------------------------------

Index

- Case, [5](#)
 - i, [5](#)
 - j, [5](#)
- Dico, [5](#)
 - dico, [6](#)
 - marqueurs, [6](#)
- dico
 - Dico, [6](#)
 - Jeu, [6](#)
- dictionnaire.c, [9](#)
 - dictionnaire_nouveau, [9](#)
 - dictionnaire_rechercheDichotomique, [10](#)
 - dictonnaire_motDansDico, [10](#)
- dictionnaire_nouveau
 - dictionnaire.c, [9](#)
- dictionnaire_rechercheDichotomique
 - dictionnaire.c, [10](#)
- dictonnaire_motDansDico
 - dictionnaire.c, [10](#)
- grille
 - Plateau, [7](#)
- i
 - Case, [5](#)
- interfaceNcurses.c, [10](#)
- j
 - Case, [5](#)
- Jeu, [6](#)
 - dico, [6](#)
 - plateau, [6](#)
 - solutionUtilisateur, [6](#)
 - timestampDepart, [6](#)
- jeu.c, [11](#)
 - jeu_compteurClaque, [11](#)
 - jeu_lancer, [11](#)
 - jeu_lancerModeTexte, [12](#)
 - jeu_nouveau, [12](#)
 - jeu_proposerMot, [12](#)
 - jeu_stopper, [12](#)
 - jeu_tempsRestant, [12](#)
- jeu_compteurClaque
 - jeu.c, [11](#)
- jeu_lancer
 - jeu.c, [11](#)
- jeu_lancerModeTexte
 - jeu.c, [12](#)
- jeu_nouveau
 - jeu.c, [12](#)
 - jeu_proposerMot
 - jeu.c, [12](#)
 - jeu_stopper
 - jeu.c, [12](#)
 - jeu_tempsRestant
 - jeu.c, [12](#)
 - marqueurs
 - Dico, [6](#)
 - mots
 - Solution, [8](#)
 - nbMots
 - Solution, [8](#)
 - Plateau, [7](#)
 - grille, [7](#)
 - solution, [7](#)
 - tailleGrille, [7](#)
 - plateau
 - Jeu, [6](#)
 - plateau.c, [13](#)
 - plateau_choisirLettre, [13](#)
 - plateau_detruire, [13](#)
 - plateau_nouveau, [13](#)
 - plateau_probaLettre, [14](#)
 - plateau_remplirGrilleAleatoire, [14](#)
 - plateau_remplirGrillePredefinie, [14](#)
 - plateau_choisirLettre
 - plateau.c, [13](#)
 - plateau_detruire
 - plateau.c, [13](#)
 - plateau_nouveau
 - plateau.c, [13](#)
 - plateau_probaLettre
 - plateau.c, [14](#)
 - plateau_remplirGrilleAleatoire
 - plateau.c, [14](#)
 - plateau_remplirGrillePredefinie
 - plateau.c, [14](#)
 - recurse
 - resolveur.c, [15](#)
 - resolveur
 - resolveur.c, [15](#)
 - resolveur.c, [14](#)
 - recurse, [15](#)
 - resolveur, [15](#)
 - Solution, [7](#)

- mots, [8](#)
 - nbMots, [8](#)
- solution
 - Plateau, [7](#)
- solutionUtilisateur
 - Jeu, [6](#)
- tailleGrille
 - Plateau, [7](#)
- timestampDepart
 - Jeu, [6](#)
- util.c, [15](#)
 - util_affichageDebug, [16](#)
 - util_conversionTemps, [16](#)
 - util_deplacerCurseurDunMot, [16](#)
 - util_echanger, [16](#)
 - util_isInArray, [16](#)
 - util_nbAleatoire, [17](#)
 - util_quickSort, [17](#)
 - util_substr, [17](#)
 - util_supprimerAccents, [17](#)
 - util_uppercase, [18](#)
- util_affichageDebug
 - util.c, [16](#)
- util_conversionTemps
 - util.c, [16](#)
- util_deplacerCurseurDunMot
 - util.c, [16](#)
- util_echanger
 - util.c, [16](#)
- util_isInArray
 - util.c, [16](#)
- util_nbAleatoire
 - util.c, [17](#)
- util_quickSort
 - util.c, [17](#)
- util_substr
 - util.c, [17](#)
- util_supprimerAccents
 - util.c, [17](#)
- util_uppercase
 - util.c, [18](#)