

My Project

Generated by Doxygen 1.8.13

Contents

1	Namespace Index	1
1.1	Namespace List	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Namespace Documentation	9
5.1	io Namespace Reference	9
5.1.1	Detailed Description	11
5.1.2	Function Documentation	11
5.1.2.1	afficher()	11
5.1.2.2	afficherCarte()	11
5.1.2.3	afficherMouvements() [1/2]	11
5.1.2.4	afficherMouvements() [2/2]	11
5.1.2.5	bienvenue()	12
5.1.2.6	ChangeTerminal()	12
5.1.2.7	checkInput()	12
5.1.2.8	checkTerminalSize()	13
5.1.2.9	choix_unique_element()	13

5.1.2.10	<code>clearScreen()</code>	13
5.1.2.11	<code>createCompetence()</code>	13
5.1.2.12	<code>createCompetenceMonstre()</code>	14
5.1.2.13	<code>createMonstre()</code>	14
5.1.2.14	<code>de()</code>	14
5.1.2.15	<code>getTerminalHeight()</code>	14
5.1.2.16	<code>getTerminalWidth()</code>	15
5.1.2.17	<code>liste_elements()</code>	15
5.1.2.18	<code>loadAllCarteFromFile()</code>	16
5.1.2.19	<code>loadAllEntiteFromFile()</code>	16
5.1.2.20	<code>loadCompetenceFromFile()</code>	16
5.1.2.21	<code>long_input()</code>	16
5.1.2.22	<code>removeLastChar()</code>	17
5.1.2.23	<code>ResetTerminal()</code>	17
5.1.2.24	<code>taille_str()</code>	17
5.1.3	Variable Documentation	18
5.1.3.1	BLANK	18
5.1.3.2	BLUE	18
5.1.3.3	currentPlayerPosition	18
5.1.3.4	GREEN	18
5.1.3.5	interactionsOverlayX	18
5.1.3.6	MAGENTA	19
5.1.3.7	mapPositionX	19
5.1.3.8	mapPositiony	19
5.1.3.9	RED	19
5.1.3.10	TermHeight	19
5.1.3.11	TermWidth	19
5.1.3.12	YELLOW	19

6 Class Documentation	21
6.1 Carte Class Reference	21
6.1.1 Constructor & Destructor Documentation	21
6.1.1.1 Carte() [1/2]	21
6.1.1.2 Carte() [2/2]	22
6.1.2 Member Function Documentation	22
6.1.2.1 affichage_normal()	22
6.1.2.2 coordonneejoueur()	22
6.1.2.3 coordonneemonstre()	22
6.1.2.4 coordonneeobstacle()	22
6.1.2.5 getDescription()	22
6.1.2.6 getName()	22
6.1.2.7 nbLigneFichier()	23
6.1.2.8 operator=()	23
6.1.2.9 sauvegarde()	23
6.1.2.10 setCase()	23
6.1.2.11 setDescription()	23
6.1.2.12 setName()	23
6.1.2.13 setPlateau()	23
6.1.2.14 verif_taille()	24
6.2 competence Class Reference	24
6.2.1 Constructor & Destructor Documentation	24
6.2.1.1 competence() [1/3]	24
6.2.1.2 competence() [2/3]	24
6.2.1.3 competence() [3/3]	25
6.2.1.4 ~competence()	25
6.2.2 Member Function Documentation	25
6.2.2.1 competenceString()	25
6.2.2.2 getDamage()	25
6.2.2.3 getDescription()	25

6.2.2.4	getManaCost()	25
6.2.2.5	getName()	25
6.2.2.6	printCompetence()	26
6.2.2.7	toString()	26
6.3	entite Class Reference	26
6.3.1	Constructor & Destructor Documentation	27
6.3.1.1	entite() [1/2]	27
6.3.1.2	entite() [2/2]	28
6.3.2	Member Function Documentation	28
6.3.2.1	enleverMana()	28
6.3.2.2	enleverVie()	28
6.3.2.3	entiteString()	29
6.3.2.4	getAlive()	29
6.3.2.5	getDescription()	29
6.3.2.6	getHpCurrent()	29
6.3.2.7	getHpMax()	29
6.3.2.8	getID()	29
6.3.2.9	getManaCurrent()	30
6.3.2.10	getManaMax()	30
6.3.2.11	getName()	30
6.3.2.12	getSkillVect()	30
6.3.2.13	getSpeed()	30
6.3.2.14	nbLigneFichier()	30
6.3.2.15	printEntite()	31
6.3.2.16	saveInFile()	31
6.3.2.17	toString()	31
6.3.3	Member Data Documentation	31
6.3.3.1	entiteAlive	31
6.3.3.2	entiteDescription	31
6.3.3.3	entiteHpCurrent	32

6.3.3.4	entiteHpMax	32
6.3.3.5	entiteId	32
6.3.3.6	entiteManaCurrent	32
6.3.3.7	entiteManaMax	32
6.3.3.8	entiteName	32
6.3.3.9	entiteSkillVect	32
6.3.3.10	entiteSpeed	33
6.4	jeu Class Reference	33
6.4.1	Detailed Description	33
6.4.2	Constructor & Destructor Documentation	33
6.4.2.1	jeu()	34
6.4.2.2	~jeu()	34
6.4.3	Member Function Documentation	34
6.4.3.1	getCarte()	34
6.4.3.2	getMonstres()	34
6.4.3.3	getNbMonstres()	34
6.4.3.4	getPerso()	35
6.4.3.5	preparation_partie()	35
6.5	monstre Class Reference	35
6.5.1	Constructor & Destructor Documentation	35
6.5.1.1	monstre() [1/2]	36
6.5.1.2	monstre() [2/2]	36
6.5.2	Member Function Documentation	36
6.5.2.1	printMonstre()	36
6.6	personnage Class Reference	37
6.6.1	Constructor & Destructor Documentation	37
6.6.1.1	personnage() [1/2]	37
6.6.1.2	personnage() [2/2]	37
6.6.2	Member Function Documentation	38
6.6.2.1	printPersonnage()	38

7 File Documentation	39
7.1 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/carte.h File Reference	39
7.2 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/competence.h File Reference	39
7.3 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/entite.h File Reference . . .	39
7.3.1 Macro Definition Documentation	40
7.3.1.1 ENTITE_H	40
7.4 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/fonctionsjeu.h File Reference	40
7.4.1 Macro Definition Documentation	40
7.4.1.1 FONCTIONSJEU_H	41
7.4.2 Function Documentation	41
7.4.2.1 sort_speed()	41
7.5 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/io.h File Reference	41
7.5.1 Macro Definition Documentation	43
7.5.1.1 IO_H	43
7.6 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/monstre.h File Reference . .	43
7.7 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/personnage.h File Reference	44
Index	45

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

[io](#)

Cet espace sera un espace permettant de définir un buffer custom pour les input, ainsi que de pouvoir afficher tout ce que l'on souhaite

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Carte	21
competence	24
entite	26
monstre	35
personnage	37
jeu	33

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Carte	21
competence	24
entite	26
jeu		
	Ceci sera la classe du jeu. Elle contient toutes les entités, la carte, ainsi que les fonctions nécessaires à la partie	33
monstre	35
personnage	37

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

/Users/thibault/GitHub/CERI_software_engineering_game_1/headers/ carte.h	39
/Users/thibault/GitHub/CERI_software_engineering_game_1/headers/ competence.h	39
/Users/thibault/GitHub/CERI_software_engineering_game_1/headers/ entite.h	39
/Users/thibault/GitHub/CERI_software_engineering_game_1/headers/ fonctionsjeu.h	40
/Users/thibault/GitHub/CERI_software_engineering_game_1/headers/ io.h	41
/Users/thibault/GitHub/CERI_software_engineering_game_1/headers/ monstre.h	43
/Users/thibault/GitHub/CERI_software_engineering_game_1/headers/ personnage.h	44

Chapter 5

Namespace Documentation

5.1 io Namespace Reference

Cet espace sera un espace permettant de définir un buffer custom pour les input, ainsi que de pouvoir afficher tout ce que l'on souhaite.

Functions

- void [ChangeTerminal](#) (bool Ech=0)
Changement des paramètres du terminal.
- void [ResetTerminal](#) ()
Remet le terminal à zero.
- char [de](#) ()
Input.
- void [removeLastChar](#) (std::stringstream &i)
Enlève le dernier caractère d'un stringstream.
- std::string [long_input](#) ()
Long input.
- int [getTerminalWidth](#) ()
Retourne la largeur du terminal.
- int [getTerminalHeight](#) ()
Retourne la hauteur du terminal.
- void [bienvenue](#) ()
Message d'accueil.
- bool [checkInput](#) (int x)
Vérifie que l'user entre des entier.
- [competence](#) [createCompetence](#) ()
Creer une competence.
- [competence](#) [createCompetenceMonstre](#) ()
Créer une compétence pour monstre (sans mana)
- [monstre](#) [createMonstre](#) ()
Créer un monstre.
- std::vector< [competence](#) > [loadCompetenceFromFile](#) (std::string nomFichier, int numLigne)
Récupérer les compétences d'un monstre dans le .txt.
- void [clearScreen](#) ()

- *Efface l'écran.*
- void [afficherCarte](#) ([Carte](#) &, [personnage](#) &, int)
 - *Affichage de la carte.*
- void [afficherMouvements](#) ()
- void [afficherMouvements](#) (std::string)
- int [taille_str](#) (std::string)
 - *Compte la taille d'une string mieux que la fonction std::string::size(), car elle ne compte pas les accents comme deux caractères.*
- void [checkTerminalSize](#) ()
- std::vector< [Carte](#) > [loadAllCarteFromFile](#) (std::string nomFichier)
 - *Récupérer les cartes dans le .txt.*
- template<typename T >
 - void [afficher](#) (T object)
 - *Affichage d'objet.*
- template<typename T >
 - void [liste_elements](#) (std::vector< T > vect_element)
 - *Affichage d'un ensemble d'objets.*
- template<typename T >
 - T [choix_unique_element](#) (std::vector< T > vect_element)
 - *Choix d'un élément unique.*
- template<typename T >
 - std::vector< T > [loadAllEntiteFromFile](#) (T temp, std::string nomFichier)

Variables

- int [TermWidth](#)
 - *Variable retenant la valeur de la largeur de la fenêtre du terminal. Elle permet de réduire le nombre de calculs à faire (étant donné que cette valeur est obtenue avec l'ouverture d'un fichier, son calcul prends donc quelques temps).*
- int [TermHeight](#)
 - *Variable retenant la valeur de la hauteur de la fenêtre du terminal. Elle permet de réduire le nombre de calculs à faire (étant donné que cette valeur est obtenue avec l'ouverture d'un fichier, son calcul prends donc quelques temps).*
- std::string [BLANK](#)
 - *Chaîne de caractères permettant de remettre à zéro la couleur du texte.*
- std::string [RED](#)
 - *Chaîne de caractères permettant de rendre le texte affiché de couleur rouge.*
- std::string [GREEN](#)
 - *Chaîne de caractères permettant de rendre le texte affiché de couleur verte.*
- std::string [YELLOW](#)
 - *Chaîne de caractères permettant de rendre le texte affiché de couleur jaune.*
- std::string [BLUE](#)
 - *Chaîne de caractères permettant de rendre le texte affiché de couleur bleue.*
- std::string [MAGENTA](#)
 - *Chaîne de caractères permettant de rendre le texte affiché de couleur magenta.*
- int [mapPositionX](#)
 - *Variable permettant de retenir à partir de quelle coordonnée "x" la carte est affichée (si la carte est plus grande que la fenêtre de terminal, cette valeur ne sera pas toujours à 0 ...)*
- int [mapPositiony](#)
 - *Variable permettant de retenir à partir de quelle coordonnée "y" la carte est affichée (si la carte est plus grande que la fenêtre de terminal, cette valeur ne sera pas toujours à 0 ...)*
- int [interactionsOverlayX](#)
 - *Stocke la position (x) de l'affichage de l'overlay des actions. Nous n'avons pas besoin du Y car l'overlay prends toute la largeur quoi qu'il arrive.*
- std::pair< int, int > [currentPlayerPosition](#)
 - *Paire de valeurs (std::pair) gardant la position actuelle du joueur dans.*

5.1.1 Detailed Description

Cet espace sera un espace permettant de définir un buffer custom pour les input, ainsi que de pouvoir afficher tout ce que l'on souhaite.

5.1.2 Function Documentation

5.1.2.1 afficher()

```
template<typename T >
void io::afficher (
    T object )
```

Affichage d'objet.

Affiche le nom et la description d'un objet.

Parameters

<i>object</i>	Objet à afficher.
---------------	-------------------

5.1.2.2 afficherCarte()

```
void io::afficherCarte (
    Carte & ,
    personnage & ,
    int )
```

Affichage de la carte.

5.1.2.3 afficherMouvements() [1/2]

```
void io::afficherMouvements ( )
```

5.1.2.4 afficherMouvements() [2/2]

```
void io::afficherMouvements (
    std::string )
```

5.1.2.5 bienvenue()

```
void io::bienvenue ( )
```

Message d'accueil.

Affiche un message de bienvenue.

5.1.2.6 ChangeTerminal()

```
void io::ChangeTerminal (
    bool Ech = 0 )
```

Changement des paramètres du terminal.

Permet de changer le mode d'entrée de stdin du terminal. Les paramètres présents auparavant sont sauvegardés.

Parameters

<i>Ech</i>	Détermine si on veut que l'entrée utilisateur soit affichée ou pas.
------------	---

See also

[de\(\)](#), [long_input\(\)](#)

5.1.2.7 checkInput()

```
bool io::checkInput (
    int x )
```

Vérifie que l'user entre des entier.

Cette fonction vérifie que l'entrée utilisateur est bien un entier.

Mode opératoire :

- Vérification du failbit de l'entrée utilisateur (std::cin::failbit)
 1. Vidage du buffer
 2. Ignore 256 caractères ou jusqu'a
 3. Affichage d'un message d'erreur d'entrée utilisateur.
 4. Retourne faux
- Sinon retourne vrai

Parameters

<i>x</i>	on sait pas ce qu'il fait là, mais il est là.
----------	---

5.1.2.8 checkTerminalSize()

```
void io::checkTerminalSize ( )
```

5.1.2.9 choix_unique_element()

```
template<typename T >  
T io::choix_unique_element (   
    std::vector< T > vect_element )
```

Choix d'un élément unique.

Fonction qui prend un vecteur d'éléments en entrée ainsi qu'un booléen, et affiche puis renvoie l'élément choisi.

Parameters

<i>vect_element</i>	Vecteur de l'élément à choisir.
<i>need_desc</i>	Nécessité de description ou non.

Returns

L'élément choisi.

See also

[liste_elements\(\)](#), [afficher\(\)](#)

5.1.2.10 clearScreen()

```
void io::clearScreen ( )
```

Efface l'écran.

5.1.2.11 createCompetence()

```
competence io::createCompetence ( )
```

Créer une compétence.

Cette fonction permet de créer rapidement une compétence pour pouvoir l'utiliser facilement après.

Mode opératoire :

- On crée les variables qui vont tenir les infos rentrées (skillName, skillDamage, skillManaCost)
- On rentre

5.1.2.12 createCompetenceMonstre()

```
competence io::createCompetenceMonstre ( )
```

Créer une compétence pour monstre (sans mana)

5.1.2.13 createMonstre()

```
monstre io::createMonstre ( )
```

Créer un monstre.

5.1.2.14 de()

```
char io::de ( )
```

Input.

Gestion des entrées utilisateur, ne prends qu'un seul caractère à la fois.

Voici son mode opératoire :

1. On crée une variable (char)
2. On change la façon dont le terminal gère l'entrée utilisateur avec [ChangeTerminal\(\)](#)
3. On utilise la fonction `std::getchar()` (qui ne prends maintenant qu'un seul caractère sans avoir besoin d'appuyer sur entrée, grâce à [ChangeTerminal\(\)](#))
4. On remet les paramètres du terminal comme avant avec [ResetTerminal\(\)](#)
5. On retourne l'entrée utilisateur

See also

[ChangeTerminal\(\)](#); [ResetTerminal\(\)](#); [long_input\(\)](#)

5.1.2.15 getTerminalHeight()

```
int io::getTerminalHeight ( )
```

Retourne la hauteur du terminal.

5.1.2.16 getTerminalWidth()

```
int io::getTerminalWidth ( )
```

Retourne la largeur du terminal.

5.1.2.17 liste_elements()

```
template<typename T >  
void io::liste_elements (   
    std::vector< T > vect_element )
```

Affichage d'un ensemble d'objets.

Parcourt le vecteur de stockage des objets chargés, et les affiche.

Parameters

<i>vect_element</i>	Vecteur d'éléments.
<i>need_desc</i>	description ou non.

See also

[afficher\(\)](#)**5.1.2.18 loadAllCarteFromFile()**

```
std::vector<Carte> io::loadAllCarteFromFile (
    std::string nomFichier )
```

Récupérer les cartes dans le .txt.

5.1.2.19 loadAllEntiteFromFile()

```
template<typename T >
std::vector<T> io::loadAllEntiteFromFile (
    T temp,
    std::string nomFichier )
```

5.1.2.20 loadCompetenceFromFile()

```
std::vector<competence> io::loadCompetenceFromFile (
    std::string nomFichier,
    int numLigne )
```

Récupérer les compétences d'un monstre dans le .txt.

5.1.2.21 long_input()

```
std::string io::long_input ( )
```

Long input.

magic.gif

5.1.2.22 removeLastChar()

```
void io::removeLastChar (
    std::stringstream & i )
```

Enlève le dernier caractère d'un stringstream.

Le but de cette fonction est d'enlever le dernier caractère d'un flux de caractères (std::stringstream) étant donné que le C++ ne propose pas de fonction par défaut pour cette fonctionnalité.

Voici son mode opératoire :

1. On prends tout le contenu du stringstream et on le met dans une chaîne de caractères (std::string)
2. Si la chaîne de caractère contient au moins 1 caractère :
 - (a) On enlève le dernier caractère affiché sur stdout (en déplaçant le curseur vers la droite après avoir affiché un espace)
 - (b) Alors on utilise la fonction std::string::erase(std::string::iterator) pour enlever le dernier caractère
 - (c) On remplace le contenu du flux de caractère par du vide
 - (d) On remet la chaîne de caractère coupée dans le flux.

Precondition

La fonction recevra un stringstream d'entrée utilisateur. Son but est d'enlever le dernier caractère entré (cette fonction est appelée dans [long_input\(\)](#) dans une condition si le caractère rentré est 127, aussi connu sous le nom de DEL ASCII).

Postcondition

La fonction ne retourne rien, car le seul argument est passé **par argument** et est donc automatiquement modifié.

Parameters

<i>i</i>	C'est un flux de caractères (std::stringstream) à partir duquel il faudra enlever le dernier caractère.
----------	---

5.1.2.23 ResetTerminal()

```
void io::ResetTerminal ( )
```

Remet le terminal à zero.

5.1.2.24 taille_str()

```
int io::taille_str (
    std::string )
```

Compte la taille d'une string mieux que la fonction std::string::size(), car elle ne compte pas les accents comme deux caractères.

5.1.3 Variable Documentation

5.1.3.1 BLANK

```
std::string io::BLANK
```

Chaîne de caractères permettant de remettre à zéro la couleur du texte.

5.1.3.2 BLUE

```
std::string io::BLUE
```

Chaîne de caractères permettant de rendre le texte affiché de couleur bleue.

5.1.3.3 currentPlayerPosition

```
std::pair<int,int> io::currentPlayerPosition
```

Paire de valeurs (std::pair) gardant la position actuelle du joueur dans.

5.1.3.4 GREEN

```
std::string io::GREEN
```

Chaîne de caractères permettant de rendre le texte affiché de couleur verte.

5.1.3.5 interactionsOverlayX

```
int io::interactionsOverlayX
```

Stocke la position (x) de l'affichage de l'overlay des actions. Nous n'avons pas besoin du Y car l'overlay prends toute la largeur quoi qu'il arrive.

5.1.3.6 MAGENTA

```
std::string io::MAGENTA
```

Chaîne de caractères permettant de rendre le texte affiché de couleur magenta.

5.1.3.7 mapPositionX

```
int io::mapPositionX
```

Variable permettant de retenir à partir de quelle coordonnée "x" la carte est affichée (si la carte est plus grande que la fenêtre de terminal, cette valeur ne sera pas toujours à 0 ...)

5.1.3.8 mapPositiony

```
int io::mapPositiony
```

Variable permettant de retenir à partir de quelle coordonnée "y" la carte est affichée (si la carte est plus grande que la fenêtre de terminal, cette valeur ne sera pas toujours à 0 ...)

5.1.3.9 RED

```
std::string io::RED
```

Chaîne de caractères permettant de rendre le texte affiché de couleur rouge.

5.1.3.10 TermHeight

```
int io::TermHeight
```

Variable retenant la valeur de la hauteur de la fenêtre du terminal. Elle permet de réduire le nombre de calculs à faire (étant donné que cette valeur est obtenue avec l'ouverture d'un fichier, son calcul prends donc quelques temps).

5.1.3.11 TermWidth

```
int io::TermWidth
```

Variable retenant la valeur de la largeur de la fenêtre du terminal. Elle permet de réduire le nombre de calculs à faire (étant donné que cette valeur est obtenue avec l'ouverture d'un fichier, son calcul prends donc quelques temps).

5.1.3.12 YELLOW

```
std::string io::YELLOW
```

Chaîne de caractères permettant de rendre le texte affiché de couleur jaune.

Chapter 6

Class Documentation

6.1 Carte Class Reference

```
#include <carte.h>
```

Public Member Functions

- [Carte](#) ()
- [Carte](#) (int taille, std::string name, std::string description)
- int [verif_taille](#) (int taille)
- void [coordonneejoueur](#) ()
- void [coordonneeobstacle](#) ()
- void [coordonneemonstre](#) ()
- void [affichage_normal](#) ()
- int [nbLigneFichier](#) (std::string nomFichier)
- void [sauvegarde](#) ()
- std::string [getName](#) ()
- std::string [getDescription](#) ()
- void [setName](#) (std::string name)
- void [setDescription](#) (std::string desc)
- void [setPlateau](#) (int taille)
- void [setCase](#) (int i, int j, std::string value)
- [Carte operator=](#) (const [Carte](#) &a_copier)

6.1.1 Constructor & Destructor Documentation

6.1.1.1 [Carte](#)() [1/2]

```
Carte::Carte ( )
```

6.1.1.2 Carte() [2/2]

```
Carte::Carte (
    int taille,
    std::string name,
    std::string description )
```

6.1.2 Member Function Documentation

6.1.2.1 affichage_normal()

```
void Carte::affichage_normal ( )
```

6.1.2.2 coordonneejoueur()

```
void Carte::coordonneejoueur ( )
```

6.1.2.3 coordonneemonstre()

```
void Carte::coordonneemonstre ( )
```

6.1.2.4 coordonneeobstacle()

```
void Carte::coordonneeobstacle ( )
```

6.1.2.5 getDescription()

```
std::string Carte::getDescription ( )
```

6.1.2.6 getName()

```
std::string Carte::getName ( )
```

6.1.2.7 nbLigneFichier()

```
int Carte::nbLigneFichier (
    std::string nomFichier )
```

6.1.2.8 operator=()

```
Carte Carte::operator= (
    const Carte & a_copier )
```

6.1.2.9 sauvegarde()

```
void Carte::sauvegarde ( )
```

6.1.2.10 setCase()

```
void Carte::setCase (
    int i,
    int j,
    std::string value )
```

6.1.2.11 setDescription()

```
void Carte::setDescription (
    std::string desc )
```

6.1.2.12 setName()

```
void Carte::setName (
    std::string name )
```

6.1.2.13 setPlateau()

```
void Carte::setPlateau (
    int taille )
```

6.1.2.14 `verif_taille()`

```
int Carte::verif_taille (
    int taille )
```

The documentation for this class was generated from the following file:

- `/Users/thibault/GitHub/CERI_software_engineering_game_1/headers/carte.h`

6.2 competence Class Reference

```
#include <competence.h>
```

Public Member Functions

- `competence` ()
- `competence` (std::string skillName, int skillDamage, int skillManaCost)
- `competence` (std::string skillName, int skillDamage)
- `~competence` ()
- std::string `getName` ()
- std::string `getDescription` ()
- int `getDamage` ()
- int `getManaCost` ()
- template<typename T >
std::string `toString` (const T &valeur)
- void `printCompetence` ()
- std::string `competenceString` ()

6.2.1 Constructor & Destructor Documentation

6.2.1.1 `competence()` [1/3]

```
competence::competence ( )
```

6.2.1.2 `competence()` [2/3]

```
competence::competence (
    std::string skillName,
    int skillDamage,
    int skillManaCost )
```


6.2.1.3 competence() [3/3]

```
competence::competence (
    std::string skillName,
    int skillDamage )
```

6.2.1.4 ~competence()

```
competence::~~competence ( )
```

6.2.2 Member Function Documentation

6.2.2.1 competenceString()

```
std::string competence::competenceString ( )
```

6.2.2.2 getDamage()

```
int competence::getDamage ( )
```

6.2.2.3 getDescription()

```
std::string competence::getDescription ( )
```

6.2.2.4 getManaCost()

```
int competence::getManaCost ( )
```

6.2.2.5 getName()

```
std::string competence::getName ( )
```

6.2.2.6 printCompetence()

```
void competence::printCompetence ( )
```

6.2.2.7 toString()

```
template<typename T >
std::string competence::toString (
    const T & valeur )
```

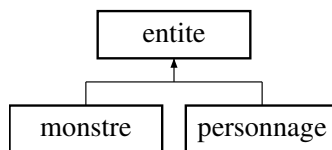
The documentation for this class was generated from the following file:

- /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/competence.h

6.3 entite Class Reference

```
#include <entite.h>
```

Inheritance diagram for entite:



Public Member Functions

- [entite](#) ()
Constructeur vide.
- [entite](#) (std::string [entiteId](#), std::string [entiteName](#), int [entiteHpMax](#), int [entiteSpeed](#), int [entiteManaMax](#), std::string [entiteDescription](#), std::vector< [competence](#) > allSkills)
Constructeur avec tout.
- template<typename T >
std::string [toString](#) (const T &valeur)
- std::string [getID](#) ()
Getter pour l'id.
- std::string [getName](#) ()
Getter pour le nom.
- std::string [getDescription](#) ()
Getter pour la description.
- int [getHpMax](#) ()
Getter pour le nombre de points de vie max.
- int [getHpCurrent](#) ()
Getter pour le nombre de points de vie actuels.
- int [getSpeed](#) ()

- Getter pour la vitesse d'attaque de l'entite.*
- bool `getAlive` ()
- Getter qui permet de savoir si l'entite est en vie.*
- int `getManaMax` ()
- Getter pour la mana maximum de l'entite.*
- int `getManaCurrent` ()
- Getter pour la mana actuelle de l'entite.*
- std::vector< `competence` > `getSkillVect` ()
- Getter qui renvoie un vecteur (std::vector) de compétences.*
- int `nbLigneFichier` (std::string nomFichier)
- Retour d'une string représentant un entite.*
- std::string `entiteString` (std::string lettreEntite, std::string nomFichier)
- void `saveInFile` (std::string lettreEntite, std::string nomFichier)
- Permet d'écrire l'entite dans un fichier de sauvegarde.*
- void `printEntite` ()
- Pour tester.*
- bool `enleverVie` (int degats)
- Enlève x points de vie a l'entite.*
- bool `enleverMana` (int skillManaCost)
- Enlève x points de mana a l'entite.*

Protected Attributes

- std::string `entiteId`
- std::string `entiteName`
- std::string `entiteDescription`
- int `entiteHpMax`
- int `entiteHpCurrent`
- int `entiteManaMax`
- int `entiteManaCurrent`
- int `entiteSpeed`
- bool `entiteAlive`
- std::vector< `competence` > `entiteSkillVect`

6.3.1 Constructor & Destructor Documentation

6.3.1.1 `entite()` [1/2]

`entite::entite` ()

Constructeur vide.

6.3.1.2 entite() [2/2]

```
entite::entite (
    std::string entiteId,
    std::string entiteName,
    int entiteHpMax,
    int entiteSpeed,
    int entiteManaMax,
    std::string entiteDescription,
    std::vector< competence > allSkills )
```

Constructeur avec tout.

Parameters

<i>entiteId</i>	L'identifiant de l'entite
<i>entiteName</i>	Le nom de l'entite
<i>entiteHpMax</i>	Les points de vie max de l'entite
<i>entiteSpeed</i>	La vitesse de l'entite
<i>entiteManaMax</i>	Les points de mana max de l'entite
<i>entiteDescription</i>	La description de l'entite
<i>allSkills</i>	Un vecteur (std::vector) contenant toutes les compétences de cette entite.

6.3.2 Member Function Documentation

6.3.2.1 enleverMana()

```
bool entite::enleverMana (
    int skillManaCost )
```

Enlève x points de mana a l'entite.

Cette fonction ne sert à rien, à part ne pas faire bugger les autres.

Returns

Un booléen vérifiant la capacité à dépenser la mana.

6.3.2.2 enleverVie()

```
bool entite::enleverVie (
    int degats )
```

Enlève x points de vie a l'entite.

Cette fonction permet d'enlever des points de vie. Elle permet aussi de savoir si une entite est en vie (ptsVie < 0) ou si elle est morte.

Returns

Un booléen qui est égal à `true` si le entite est mort, `false` sinon.

6.3.2.3 entiteString()

```
std::string entite::entiteString (
    std::string lettreEntite,
    std::string nomFichier )
```

6.3.2.4 getAlive()

```
bool entite::getAlive ( )
```

Getter qui permet de savoir si l'entite est en vie.

6.3.2.5 getDescription()

```
std::string entite::getDescription ( )
```

Getter pour la description.

6.3.2.6 getHpCurrent()

```
int entite::getHpCurrent ( )
```

Getter pour le nombre de points de vie actuels.

6.3.2.7 getHpMax()

```
int entite::getHpMax ( )
```

Getter pour le nombre de points de vie max.

6.3.2.8 getID()

```
std::string entite::getID ( )
```

Getter pour l'id.

6.3.2.9 getManaCurrent()

```
int entite::getManaCurrent ( )
```

Getter pour la mana actuelle de l'entite.

6.3.2.10 getManaMax()

```
int entite::getManaMax ( )
```

Getter pour la mana maximum de l'entite.

6.3.2.11 getName()

```
std::string entite::getName ( )
```

Getter pour le nom.

6.3.2.12 getSkillVect()

```
std::vector<competence> entite::getSkillVect ( )
```

Getter qui renvoie un vecteur (std::vector) de compétences.

6.3.2.13 getSpeed()

```
int entite::getSpeed ( )
```

Getter pour la vitesse d'attaque de l'entite.

6.3.2.14 nbLigneFichier()

```
int entite::nbLigneFichier (
    std::string nomFichier )
```

Retour d'une string représentant un entite.

Convertit un objet entite en une ligne de string.

Postcondition

La string contiendra les infos dans cet ordre :

- entiteIdentifiant (type m<entier>)
- nom de l'entite
- nombre de points de vie
- vitesse d'attaque
- toutes les compétences , séparées par des : Retourne le nombre de lignes d'un fichier.

Compte le nb de lignes du fichier pour créer l'identifiant unique d'un entite. L'identifiant sera nbLignes + 1

Returns

Un entier représentant le nombre de lignes.

Parameters

<i>nomFichier</i>	Une string (std::string) qui sera le nom du fichier à ouvrir.
-------------------	---

6.3.2.15 printEntite()

```
void entite::printEntite ( )
```

Pour tester.

6.3.2.16 saveInFile()

```
void entite::saveInFile (
    std::string lettreEntite,
    std::string nomFichier )
```

Permet d'écrire l'entite dans un fichier de sauvegarde.

6.3.2.17 toString()

```
template<typename T >
std::string entite::toString (
    const T & valeur )
```

6.3.3 Member Data Documentation**6.3.3.1 entiteAlive**

```
bool entite::entiteAlive [protected]
```

6.3.3.2 entiteDescription

```
std::string entite::entiteDescription [protected]
```

6.3.3.3 entiteHpCurrent

```
int entite::entiteHpCurrent [protected]
```

6.3.3.4 entiteHpMax

```
int entite::entiteHpMax [protected]
```

6.3.3.5 entiteId

```
std::string entite::entiteId [protected]
```

6.3.3.6 entiteManaCurrent

```
int entite::entiteManaCurrent [protected]
```

6.3.3.7 entiteManaMax

```
int entite::entiteManaMax [protected]
```

6.3.3.8 entiteName

```
std::string entite::entiteName [protected]
```

6.3.3.9 entiteSkillVect

```
std::vector<competence> entite::entiteSkillVect [protected]
```


6.3.3.10 entiteSpeed

```
int entite::entiteSpeed [protected]
```

The documentation for this class was generated from the following file:

- /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/[entite.h](#)

6.4 jeu Class Reference

Ceci sera la classe du jeu. Elle contient toutes les entités, la carte, ainsi que les fonctions nécessaires à la partie.

```
#include <fonctionsjeu.h>
```

Public Member Functions

- [jeu](#) ()
Constructeur par défaut sans argument.
- [~jeu](#) ()
Destructeur par défaut.
- [Carte](#) [getCarte](#) ()
- [personnage](#) [getPerso](#) ()
- [std::vector< monstre >](#) [getMonstres](#) ()
- [int](#) [getNbMonstres](#) ()
- [void](#) [preparation_partie](#) ()
Fonction permettant de déterminer comment va démarrer la partie.

6.4.1 Detailed Description

Ceci sera la classe du jeu. Elle contient toutes les entités, la carte, ainsi que les fonctions nécessaires à la partie.

Cette classe contient les fonctions nécessaires au démarrage de la partie, au combat, ainsi que toutes les fonctions intermédiaires nécessaires au bon fonctionnement de celles-ci.

Librairies incluses :

- [std::stack](#) ,
- [io](#) (depuis [io.h](#))

6.4.2 Constructor & Destructor Documentation

6.4.2.1 jeu()

```
jeu::jeu ( )
```

Constructeur par défaut sans argument.

Avec ce constructeur, on peut créer toutes les entités du jeu.

- Chargement de la carte,
- Création d'un personnage,
- Création de tous les monstres.

See also

`perso()`, `carte()`, [monstre\(\)](#)

6.4.2.2 ~jeu()

```
jeu::~~jeu ( )
```

Destructeur par défaut.

6.4.3 Member Function Documentation

6.4.3.1 getCarte()

```
Carte jeu::getCarte ( )
```

6.4.3.2 getMonstres()

```
std::vector<monstre> jeu::getMonstres ( )
```

6.4.3.3 getNbMonstres()

```
int jeu::getNbMonstres ( )
```

6.4.3.4 getPerso()

```
personnage jeu::getPerso ( )
```

6.4.3.5 preparation_partie()

```
void jeu::preparation_partie ( )
```

Fonction permettant de déterminer comment va démarrer la partie.

Affichage d'un message de bienvenue. Choix du personnage. Choix de la carte. Chargement des monstres.

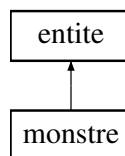
The documentation for this class was generated from the following file:

- /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/fonctionsjeu.h

6.5 monstre Class Reference

```
#include <monstre.h>
```

Inheritance diagram for monstre:



Public Member Functions

- [monstre](#) ()
Constructeur vide.
- [monstre](#) (std::string [entiteId](#), std::string [entiteName](#), int [entiteHpMax](#), int [entiteSpeed](#), int [entiteManaMax](#), std::string [entiteDescription](#), std::vector< [competence](#) > allSkills)
Constructeur avec tout.
- void [printMonstre](#) ()
Pour tester.

Additional Inherited Members

6.5.1 Constructor & Destructor Documentation

6.5.1.1 `monstre()` [1/2]

```
monstre::monstre ( ) [inline]
```

Constructeur vide.

6.5.1.2 `monstre()` [2/2]

```
monstre::monstre (
    std::string entiteId,
    std::string entiteName,
    int entiteHpMax,
    int entiteSpeed,
    int entiteManaMax,
    std::string entiteDescription,
    std::vector< competence > allSkills ) [inline]
```

Constructeur avec tout.

Parameters

<i>entiteId</i>	L'identifiant du monstre
<i>entiteName</i>	Le nom du monstre
<i>entiteHpMax</i>	Les points de vie max du monstre
<i>entiteSpeed</i>	La vitesse du monstre
<i>entiteManaMax</i>	Les points de mana max du monstre
<i>entiteDescription</i>	La entiteDescription du monstre
<i>allSkills</i>	Un vecteur (std::vector) contenant toutes les compétences de ce monstre.

6.5.2 Member Function Documentation

6.5.2.1 `printMonstre()`

```
void monstre::printMonstre ( )
```

Pour tester.

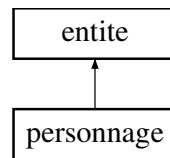
The documentation for this class was generated from the following file:

- `/Users/thibault/GitHub/CERI_software_engineering_game_1/headers/monstre.h`

6.6 personnage Class Reference

```
#include <personnage.h>
```

Inheritance diagram for personnage:



Public Member Functions

- `personnage ()`
Constructeur vide.
- `personnage (std::string entiteId, std::string entiteName, int entiteHpMax, int entiteSpeed, int entiteManaMax, std::string entiteDescription, std::vector< competence > allSkills)`
- `void printPersonnage ()`
Fonction de test.

Additional Inherited Members

6.6.1 Constructor & Destructor Documentation

6.6.1.1 `personnage()` [1/2]

```
personnage::personnage ( ) [inline]
```

Constructeur vide.

Le personnage créé aura 0 de mana, et n'aura aucune description. Mais il sera créé.

6.6.1.2 `personnage()` [2/2]

```

personnage::personnage (
    std::string entiteId,
    std::string entiteName,
    int entiteHpMax,
    int entiteSpeed,
    int entiteManaMax,
    std::string entiteDescription,
    std::vector< competence > allSkills ) [inline]
  
```

6.6.2 Member Function Documentation

6.6.2.1 printPersonnage()

```
void personnage::printPersonnage ( )
```

Fonction de test.

The documentation for this class was generated from the following file:

- /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/[personnage.h](#)

Chapter 7

File Documentation

7.1 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/carte.h File Reference

```
#include <iostream>
#include <string>
#include <vector>
```

Classes

- class [Carte](#)

7.2 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/competence.h File Reference

```
#include <string>
#include <iostream>
#include <sstream>
```

Classes

- class [competence](#)

7.3 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/entite.h File Reference

```
#include <string>
#include <vector>
#include "competence.h"
```

Classes

- class [entite](#)

Macros

- #define [ENTITE_H](#)

7.3.1 Macro Definition Documentation

7.3.1.1 ENTITE_H

```
#define ENTITE_H
```

7.4 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/fonctionsjeu.h File Reference

```
#include "../headers/io.h"  
#include "../headers/carte.h"  
#include "../headers/monstre.h"  
#include "../headers/competence.h"  
#include "../headers/personnage.h"  
#include <stack>  
#include <vector>
```

Classes

- class [jeu](#)

Ceci sera la classe du jeu. Elle contient toutes les entités, la carte, ainsi que les fonctions nécessaires à la partie.

Macros

- #define [FONCTIONSJEU_H](#)

Functions

- bool [sort_speed](#) ([entite](#) a, [entite](#) b)

7.4.1 Macro Definition Documentation

7.4.1.1 FONCTIONSJEU_H

```
#define FONCTIONSJEU_H
```

7.4.2 Function Documentation

7.4.2.1 sort_speed()

```
bool sort_speed (
    entite a,
    entite b )
```

7.5 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/io.h File Reference

```
#include <algorithm>
#include <fstream>
#include <iostream>
#include <sstream>
#include <stdio.h>
#include <termios.h>
#include <typeinfo>
#include <vector>
#include "../headers/carte.h"
#include "../headers/competence.h"
#include "../headers/monstre.h"
#include "../headers/personnage.h"
```

Namespaces

- [io](#)

Cet espace sera un espace permettant de définir un buffer custom pour les input, ainsi que de pouvoir afficher tout ce que l'on souhaite.

Macros

- `#define` [IO_H](#)

Functions

- void [io::ChangeTerminal](#) (bool Ech=0)
Changement des paramètres du terminal.
- void [io::ResetTerminal](#) ()
Remet le terminal à zero.
- char [io::de](#) ()
Input.
- void [io::removeLastChar](#) (std::stringstream &i)
Enlève le dernier caractère d'un stringstream.
- std::string [io::long_input](#) ()
Long input.
- int [io::getTerminalWidth](#) ()
Retourne la largeur du terminal.
- int [io::getTerminalHeight](#) ()
Retourne la hauteur du terminal.
- void [io::bienvenue](#) ()
Message d'accueil.
- bool [io::checkInput](#) (int x)
Vérifie que l'user entre des entier.
- [competence](#) [io::createCompetence](#) ()
Créer une competence.
- [competence](#) [io::createCompetenceMonstre](#) ()
Créer une compétence pour monstre (sans mana)
- [monstre](#) [io::createMonstre](#) ()
Créer un monstre.
- std::vector< [competence](#) > [io::loadCompetenceFromFile](#) (std::string nomFichier, int numLigne)
Récupérer les compétences d'un monstre dans le .txt.
- void [io::clearScreen](#) ()
Efface l'écran.
- void [io::afficherCarte](#) ([Carte](#) &, [personnage](#) &, int)
Affichage de la carte.
- void [io::afficherMouvements](#) ()
- void [io::afficherMouvements](#) (std::string)
- int [io::taille_str](#) (std::string)
Compte la taille d'une string mieux que la fonction std::string::size(), car elle ne compte pas les accents comme deux caractères.
- void [io::checkTerminalSize](#) ()
- std::vector< [Carte](#) > [io::loadAllCarteFromFile](#) (std::string nomFichier)
Récupérer les cartes dans le .txt.
- template<typename T >
void [io::afficher](#) (T object)
Affichage d'objet.
- template<typename T >
void [io::liste_elements](#) (std::vector< T > vect_element)
Affichage d'un ensemble d'objets.
- template<typename T >
T [io::choix_unique_element](#) (std::vector< T > vect_element)
Choix d'un élément unique.
- template<typename T >
std::vector< T > [io::loadAllEntiteFromFile](#) (T temp, std::string nomFichier)

Variables

- int [io::TermWidth](#)
Variable retenant la valeur de la largeur de la fenêtre du terminal. Elle permet de réduire le nombre de calculs à faire (étant donné que cette valeur est obtenue avec l'ouverture d'un fichier, son calcul prends donc quelques temps).
- int [io::TermHeight](#)
Variable retenant la valeur de la hauteur de la fenêtre du terminal. Elle permet de réduire le nombre de calculs à faire (étant donné que cette valeur est obtenue avec l'ouverture d'un fichier, son calcul prends donc quelques temps).
- std::string [io::BLANK](#)
Chaîne de caractères permettant de remettre à zéro la couleur du texte.
- std::string [io::RED](#)
Chaîne de caractères permettant de rendre le texte affiché de couleur rouge.
- std::string [io::GREEN](#)
Chaîne de caractères permettant de rendre le texte affiché de couleur verte.
- std::string [io::YELLOW](#)
Chaîne de caractères permettant de rendre le texte affiché de couleur jaune.
- std::string [io::BLUE](#)
Chaîne de caractères permettant de rendre le texte affiché de couleur bleue.
- std::string [io::MAGENTA](#)
Chaîne de caractères permettant de rendre le texte affiché de couleur magenta.
- int [io::mapPositionX](#)
Variable permettant de retenir à partir de quelle coordonnée "x" la carte est affichée (si la carte est plus grande que la fenêtre de terminal, cette valeur ne sera pas toujours à 0 ...)
- int [io::mapPositiony](#)
Variable permettant de retenir à partir de quelle coordonnée "y" la carte est affichée (si la carte est plus grande que la fenêtre de terminal, cette valeur ne sera pas toujours à 0 ...)
- int [io::interactionsOverlayX](#)
Stocke la position (x) de l'affichage de l'overlay des actions. Nous n'avons pas besoin du Y car l'overlay prends toute la largeur quoi qu'il arrive.
- std::pair< int, int > [io::currentPlayerPosition](#)
Paire de valeurs (std::pair) gardant la position actuelle du joueur dans.

7.5.1 Macro Definition Documentation

7.5.1.1 IO_H

```
#define IO_H
```

7.6 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/monstre.h File Reference

```
#include "../headers/entite.h"
```

Classes

- class [monstre](#)

7.7 /Users/thibault/GitHub/CERI_software_engineering_game_1/headers/personnage.h File Reference

```
#include "../headers/entite.h"
```

Classes

- class [personnage](#)

Index

/Users/thibault/GitHub/CERI_software_engineering_↵
game_1/headers/carte.h, 39
/Users/thibault/GitHub/CERI_software_engineering_↵
game_1/headers/competence.h, 39
/Users/thibault/GitHub/CERI_software_engineering_↵
game_1/headers/entite.h, 39
/Users/thibault/GitHub/CERI_software_engineering_↵
game_1/headers/fonctionsjeu.h, 40
/Users/thibault/GitHub/CERI_software_engineering_↵
game_1/headers/io.h, 41
/Users/thibault/GitHub/CERI_software_engineering_↵
game_1/headers/monstre.h, 43
/Users/thibault/GitHub/CERI_software_engineering_↵
game_1/headers/personnage.h, 44
~competence
competence, 25
~jeu
jeu, 34

affichage_normal
Carte, 22
afficher
io, 11
afficherCarte
io, 11
afficherMouvements
io, 11

BLANK
io, 18
BLUE
io, 18
bienvenue
io, 11

Carte, 21
affichage_normal, 22
Carte, 21
coordonneejoueur, 22
coordonneemonstre, 22
coordonneeobstacle, 22
getDescription, 22
getName, 22
nbLigneFichier, 22
operator=, 23
sauvegarde, 23
setCase, 23
setDescription, 23
setName, 23
setPlateau, 23
verif_taille, 23

ChangeTerminal
io, 12
checkInput
io, 12
checkTerminalSize
io, 13
choix_unique_element
io, 13
clearScreen
io, 13
competence, 24
~competence, 25
competence, 24
competenceString, 25
getDamage, 25
getDescription, 25
getManaCost, 25
getName, 25
printCompetence, 25
toString, 26
competenceString
competence, 25
coordonneejoueur
Carte, 22
coordonneemonstre
Carte, 22
coordonneeobstacle
Carte, 22
createCompetence
io, 13
createCompetenceMonstre
io, 13
createMonstre
io, 14
currentPlayerPosition
io, 18

de
io, 14

ENTITE_H
entite.h, 40
enleverMana
entite, 28
enleverVie
entite, 28
entite, 26
enleverMana, 28
enleverVie, 28

- entite, 27
- entiteAlive, 31
- entiteDescription, 31
- entiteHpCurrent, 31
- entiteHpMax, 32
- entiteld, 32
- entiteManaCurrent, 32
- entiteManaMax, 32
- entiteName, 32
- entiteSkillVect, 32
- entiteSpeed, 32
- entiteString, 28
- getAlive, 29
- getDescription, 29
- getHpCurrent, 29
- getHpMax, 29
- getID, 29
- getManaCurrent, 29
- getManaMax, 30
- getName, 30
- getSkillVect, 30
- getSpeed, 30
- nbLigneFichier, 30
- printEntite, 31
- saveInFile, 31
- toString, 31
- entite.h
 - ENTITE_H, 40
- entiteAlive
 - entite, 31
- entiteDescription
 - entite, 31
- entiteHpCurrent
 - entite, 31
- entiteHpMax
 - entite, 32
- entiteld
 - entite, 32
- entiteManaCurrent
 - entite, 32
- entiteManaMax
 - entite, 32
- entiteName
 - entite, 32
- entiteSkillVect
 - entite, 32
- entiteSpeed
 - entite, 32
- entiteString
 - entite, 28
- FONCTIONSJEU_H
 - fonctionsjeu.h, 40
- fonctionsjeu.h
 - FONCTIONSJEU_H, 40
 - sort_speed, 41
- GREEN
 - io, 18
- getAlive
 - entite, 29
- getCarte
 - jeu, 34
- getDamage
 - competence, 25
- getDescription
 - Carte, 22
 - competence, 25
 - entite, 29
- getHpCurrent
 - entite, 29
- getHpMax
 - entite, 29
- getID
 - entite, 29
- getManaCost
 - competence, 25
- getManaCurrent
 - entite, 29
- getManaMax
 - entite, 30
- getMonstres
 - jeu, 34
- getName
 - Carte, 22
 - competence, 25
 - entite, 30
- getNbMonstres
 - jeu, 34
- getPerso
 - jeu, 34
- getSkillVect
 - entite, 30
- getSpeed
 - entite, 30
- getTerminalHeight
 - io, 14
- getTerminalWidth
 - io, 14
- IO_H
 - io.h, 43
- interactionsOverlayX
 - io, 18
- io, 9
 - afficher, 11
 - afficherCarte, 11
 - afficherMouvements, 11
 - BLANK, 18
 - BLUE, 18
 - bienvenue, 11
 - ChangeTerminal, 12
 - checkInput, 12
 - checkTerminalSize, 13
 - choix_unique_element, 13
 - clearScreen, 13
 - createCompetence, 13
 - createCompetenceMonstre, 13

- createMonstre, 14
- currentPlayerPosition, 18
- de, 14
- GREEN, 18
- getTerminalHeight, 14
- getTerminalWidth, 14
- interactionsOverlayX, 18
- liste_elements, 15
- loadAllCarteFromFile, 16
- loadAllEntiteFromFile, 16
- loadCompetenceFromFile, 16
- long_input, 16
- MAGENTA, 18
- mapPositionX, 19
- mapPositiony, 19
- RED, 19
- removeLastChar, 16
- ResetTerminal, 17
- taille_str, 17
- TermHeight, 19
- TermWidth, 19
- YELLOW, 19
- io.h
 - IO_H, 43
- jeu, 33
 - ~jeu, 34
 - getCarte, 34
 - getMonstres, 34
 - getNbMonstres, 34
 - getPerso, 34
 - jeu, 33
 - preparation_partie, 35
- liste_elements
 - io, 15
- loadAllCarteFromFile
 - io, 16
- loadAllEntiteFromFile
 - io, 16
- loadCompetenceFromFile
 - io, 16
- long_input
 - io, 16
- MAGENTA
 - io, 18
- mapPositionX
 - io, 19
- mapPositiony
 - io, 19
- monstre, 35
 - monstre, 35, 36
 - printMonstre, 36
- nbLigneFichier
 - Carte, 22
 - entite, 30
- operator=
 - Carte, 23
 - personnage, 37
 - personnage, 37
 - printPersonnage, 38
 - preparation_partie
 - jeu, 35
 - printCompetence
 - competence, 25
 - printEntite
 - entite, 31
 - printMonstre
 - monstre, 36
 - printPersonnage
 - personnage, 38
- RED
 - io, 19
- removeLastChar
 - io, 16
- ResetTerminal
 - io, 17
- sauvegarde
 - Carte, 23
- saveInFile
 - entite, 31
- setCase
 - Carte, 23
- setDescription
 - Carte, 23
- setName
 - Carte, 23
- setPlateau
 - Carte, 23
- sort_speed
 - fonctionsjeu.h, 41
- taille_str
 - io, 17
- TermHeight
 - io, 19
- TermWidth
 - io, 19
- toString
 - competence, 26
 - entite, 31
- verif_taille
 - Carte, 23
- YELLOW
 - io, 19