Snake

Created by: Alessandro Beltramo
Version 1.0
07/06/2012

Indice dei tipi composti

Elenco dei tipi composti

Queste sono le classi, le struct, le union e le interfacce con una loro breve descrizione:

Documentazione delle classi

Riferimenti per la classe ***cDAT***

Membri pubblici

- **cDAT** (void)

- bool **Create** (std::vector< std::string > files, std::string destination)
  Crea il file destination inserendogli all'interno tutti i files indicati nel vettore di stringhe.

- void **Read** (std::string source)
  Legge il file .dat chiamato source

- char * **GetFile** (std::string filename)
  Recupera filename dal file .dat.
  Per richiamare questa funziona bisogna aver precedentemente letto il file con la funzione Read

- long int **GetFileSize** (std::string filename)
  Restituisce la dimensione del file in byte.
  Per richiamare questa funziona bisogna aver precedentemente letto il file con la funzione Read

Esempio:

```
cDAT DataFile;
std::vector<std::string> FileList;
FileList.push_back("Img\\Background.png");
if (!DataFile.Create(FileList, "resources.dat"))
	return EXIT_ERROR;
//LEGGO IL FILE
DataFile.Read("Resources.dat");
char* buffer;
buffer = DataFile.GetFile("FileName");
if(buffer == NULL)
	return EXIT_ERROR;
```

La documentazione per questa classe è stata generata a partire dai seguenti file:

- C:/Users/Ale/Documents/Visual Studio 2010/Projects/SnakeGrafico/SnakeGrafico/DatFileClass.h
- C:/Users/Ale/Documents/Visual Studio 2010/Projects/SnakeGrafico/SnakeGrafico/DatFile.cpp

Riferimenti per la classe Controller360

Membri pubblici

- **Controller360** (int NumPad)
  Inizializza l'oggetto leggendo lo stato del pad.
  NumPad è un numero compreso tra 0 e 3 e definisce quale controller si vuole gestire

- bool **CheckControllerStatus** (void)
  Restituisce true se il controller è connesso al computer altrimenti false.

- bool **CheckButton** (char *Btn)
  Restituisce true se il Btn è stato premuto.
  I possibili Btn sono:
  "A","B","X","Y";"BACK","START","PADUP","PADDOWN","PADLEFT",PADRIGHT","RB","LB",
  "BTNDX",BTNRX"

- int **CheckTrigger** (char *Btn)
  Restituisce il valore del Trigger: esso è un numero compreso tra 0 e 255
  I possibili Btn sono:
  "LT","RT".

- float **CheckThumb** (char *Btn)
  Restituisce il valore dei pad analogici: esso è un numero compreso tra -32768 e 32767
  I possibili Btn sono:
  "LX","LY","RX","RY"
- void **SetVibrations** (char *Pos, unsigned int Value)
  Imposta una vibrazione sul joystick. Value è un numero compreso tra 0 e 100.
  Le possibili Pos sono:
  "DX","SX"

Esempio:

```
Controller360 Pad(0);
int HowMuchPress = Pad.CheckTrigger("RT"); //Leggo quanto è stato premuto il trigger
```

La documentazione per questa classe è stata generata a partire dai seguenti file:

- C:/Users/Ale/Documents/Visual Studio 2010/Projects/SnakeGrafico/SnakeGrafico/Controller360.h
- C:/Users/Ale/Documents/Visual Studio 2010/Projects/SnakeGrafico/SnakeGrafico/Controller360.cpp

Riferimenti per la classe Fruit

Membri pubblici

- void **AddFruit** (int PosX, int PosY)
  Aggiunge un frutto alla posizione PosX,PosY

- int **GetNumFruits** (void)
  Restituisce il numero di frutti che sono presenti sul campo.

- bool **IsFruitPresent** (int PosX, int PosY)
  Restituisce true se alla posizione PosX,PosY è presente un frutto.

- bool **EraseFruit** (int PosX, int PosY)
  Elimina un frutto alla posizione PosX,PosY

- void **DrawFruits** (int DimFruit, sf::RenderTarget &target, sf::Texture *SnakeTexture)
  Disegna un frutto sul target utilizzando la texture SnakeTexture.
  Il nome della texture è fuorviante è possibile passare qualsiasi texture al metodo.

La documentazione per questa classe è stata generata a partire dai seguenti file:

- C:/Users/Ale/Documents/Visual Studio 2010/Projects/SnakeGrafico/SnakeGrafico/FruitClass.h
- C:/Users/Ale/Documents/Visual Studio 2010/Projects/SnakeGrafico/SnakeGrafico/Fruit.cpp

Riferimenti per la classe Snake

Membri pubblici

- **Snake** (int Rig, int Col, **Fruit** *Frutto)
  Costruttore: Inizializza un campo alto Rig e lungo Col e salva l'indirizzo dell'oggetto frutto.

- **Snake** (int Rig, int Col, int LunIniSnake, **Fruit** *Frutto)
  Oltre al costruttore precedente imposta una lunghezza iniziale dello snake

- **Snake** (int Rig, int Col, int LunIniSnake, int PosIniX, int PosIniY, **Fruit** *Frutto)
  Oltre al costruttore precedente imposta una posizione iniziale dello snake

- void **Print_All** (void)
  Stampa tramite cout come se fosse una matrice. (inutile con sfml)

- bool **Move_Next** (int Direction)
  Muove lo snake nella nuova direzione. Restituisce true se si tocca un muro (dead = true). Controlla in automatico il punteggio dello snake, se è possibile spostarsi nella nuova direzione e se tocca se stesso.

- bool **Move_Next** (int Direction, **Snake** *Player2)
  Come sopra la differenza è che controlla anche se tocca il Player2 (**NON IMPLEMENTATA ANCORA!)**

- void **DrawSnake** (sf::RenderTarget &target, int DimSnake, sf::Texture *SnakeTexture, sf::Texture *HeadTexture)
  Disegna lo Snake sul target, impostandogli una dimensione uguale a DimSnake ed assegnando SnakeTexture come texture del corpo e HeadTexture come texture della testa.

- void **DrawBorder** (sf::RenderTarget &target, int DimSnake, sf::Texture *Background)
  Disegna sul targhet lo sfondo definito dalla texture Background

- void **IncrementSnake** (int N)
  Aumenta le dimensioni dello snake di N

- bool **FindSnake** (**Coordinate** Pos)
  Restituisce true se lo snake si trova nella posizione Pos

- void **AddPoint** (int Point)
  Aggiunge Point al currente punteggio dello snake.

- int **GetPoint** (void)
  Restituisce i punti che sono stati totalizzati dallo snake.

La documentazione per questa classe è stata generata a partire dai seguenti file:

- C:/Users/Ale/Documents/Visual Studio 2010/Projects/SnakeGrafico/SnakeGrafico/snakeClass.h
- C:/Users/Ale/Documents/Visual Studio 2010/Projects/SnakeGrafico/SnakeGrafico/Snake.cpp