MulticocoSDL

1,0

Generated by Doxygen 1.8.3.1

Fri May 17 2013 21:41:03

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

Class Documentation

3.1 CollisionBox Class Reference

Public Member Functions

- **CollisionBox** (Vector2D &position, unsigned int width, unsigned int height, float rFactor)
- bool collides (CollisionBox &other)

  Comprueba si esta caja colision con otra.
- Vector2D **position** ()
- unsigned int **width** ()
- unsigned int **height** ()
- void **render** (SDL_Surface *screen, unsigned int r=0, unsigned int g=255, unsigned int b=0)
- void updatePosition ()

  Actualiza la posicion de la caja de colisiones.
- void **updatePosition** (Vector2D &pos)

3.1.1 Member Function Documentation

3.1.1.1 bool CollisionBox::collides (CollisionBox & *other*)

Comprueba si esta caja colision con otra.

Parameters

other	La otra caja de colision con la que comprobar.

Returns

True si las cajas colisionan. False en caso contrario.

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

- MulticocoSDL/MulticocoSDL/collisionbox.h
- MulticocoSDL/MulticocoSDL/collisionbox.cpp

3.2 CPSProcessSerNum Struct Reference

Protected Attributes

- UInt32 **lo**

- UInt32 **hi**

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

- MulticocoSDL/MulticocoSDL/SDLMain.m

3.3 Enemy Class Reference

Inheritance diagram for Enemy:

Collaboration diagram for Enemy:

Public Types

- enum **Type** { **FAST**, **NORMAL**, **PREDICTION**, **RANDOM** }

Public Member Functions

- **Enemy** (Type type, Scenario ∗scenario)
- void **update** ()
- bool isAlive ()
- bool isVulnerable ()
- Vector2D **positionCentered** ()
- void recordPlayerPosition (Vector2D pos)

  Guarda la posición del jugador en la lista.
- Vector2D **destinationCell** ()

Additional Inherited Members

3.3.1 Member Function Documentation

3.3.1.1 bool Enemy::isAlive ( )

Returns

3.3.1.2 bool Enemy::isVulnerable ( )

Returns

3.3.1.3 void Enemy::recordPlayerPosition (Vector2D *pos*)

Guarda la posición del jugador en la lista.

Parameters

pos	posición del jugador

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

- MulticocoSDL/MulticocoSDL/enemy.h
- MulticocoSDL/MulticocoSDL/enemy.cpp

3.4 Entity Class Reference

Inheritance diagram for Entity:

Collaboration diagram for Entity:

Public Member Functions

- void **setPosition** (float x, float y)
- void **setPosition** (Vector2D v)
- void **setDirection** (float x, float y)
- void **setDirection** (Vector2D &v)
- void **setVisible** (bool v)
- void **setMoving** (bool m)
- void **setSpriteSheet** (const char ∗file, int w, int h, int ∗animations, int nAnimations)
- SpriteSheet & **spriteSheet** ()
- CollisionBox & **collisionBox** ()
- void **setAnimation** (const char ∗name)
- bool **isVisible** ()
- bool **isMoving** ()
- Vector2D & **position** ()
- Vector2D & **direction** ()
- virtual void **update** ()
- void **move** ()
- void **moveToPreviousPosition** ()
- void **render** (SDL_Surface ∗screen, bool showDebugGraphics=false)

Protected Attributes

- Vector2D **_position**
- Vector2D **_previousPosition**
- Vector2D **_direction**
- bool **_visible**
- bool **_moving**
- SpriteSheet ∗ **_sprite**
- CollisionBox ∗ **_collisionBox**

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

- MulticocoSDL/MulticocoSDL/entity.h

3.5 Music Class Reference

Public Member Functions

- **Music** (const char ∗file)
- void **play** ()
- void **playLoop** ()

- void **stop** ()

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

- MulticocoSDL/MulticocoSDL/music.h
- MulticocoSDL/MulticocoSDL/music.cpp

3.6 NSApplication(SDL_Missing_Methods) Category Reference

Instance Methods

- (void) - **setAppleMenu:**

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

- MulticocoSDL/MulticocoSDL/SDLMain.m

3.7 NSApplication(SDLApplication) Category Reference

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

- MulticocoSDL/MulticocoSDL/SDLMain.m

3.8 NSString(ReplaceSubString) Category Reference

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

- MulticocoSDL/MulticocoSDL/SDLMain.m

3.9 Scenario Class Reference

Public Member Functions

- **Scenario** (unsigned int hSize, unsigned int vSize, Vector2D position)
- **Scenario** (const char *file)
- unsigned int horizontalSize ()

  Numero de casillas en horizontal del escenario.
- unsigned int verticalSize ()

  Numero de casillas en vertical.
- unsigned int width ()

  Ancho total del escenario en pixeles.
- unsigned int height ()

  Altura total del escenario en pixeles.
- Vector2D **enemySpawningCell** ()
- Vector2D **playerSpawningCell** ()
- SpriteSheet & **spriteSheet** ()
- Vector2D **playerSpawningPosition** ()
- Vector2D **enemySpawningPosition** ()
- void **setEnemySpawningCell** (unsigned int x, unsigned int y)

- void **setPlayerSpawningCell** (unsigned int x, unsigned int y)
- void **setRandomPlayerSpawningCell** ()
- void **setSpriteSheet** (const char *file, int w, int h, int *animations, int nAnimations)
- void setWallSprite (unsigned int pos)

  Indica qué fila de la plantilla asignada corresponde al sprite del muro.
- void setCorridorSprite (unsigned int pos)

  Indica qué fila de la plantilla asignada corresponde al sprite del pasillo.
- bool **save** (const char *file)
- bool **isWall** (int x, int y)
- bool **isCorridor** (int x, int y)
- Vector2D cell (int x, int y)

  Calcula que casilla del escenario corresponde con dichas coordenadas.
- Vector2D **cell** (Vector2D pos)
- Vector2D cellPosition (unsigned int x, unsigned int y)

  Posicion en pixeles de una celda cada.
- std::vector< Vector2D > **avalibleDirections** (int posX, int posY)
- bool **collides** (Entity &object)
- std::list< Vector2D > corridorCells ()

  Lista de celdas del escenario que no son pared.
- std::list< Vector2D > **corridorPositions** ()
- std::list< Vector2D > **corridorPositionsWithoutGhostHouse** ()
- void render (SDL_Surface *screen, bool showDebugGraphics=false)

  Dibuja el escenario en la pantalla en la posicion indicada.

3.9.1 Member Function Documentation

3.9.1.1 Vector2D Scenario::cell (int *x*, int *y*)

Calcula que casilla del escenario corresponde con dichas coordenadas.

Parameters

x	Coordenada X.
y	Coordenada Y.

Returns

Celda que corresponde a las coordenadas x e y.

3.9.1.2 void Scenario::render (SDL_Surface * *screen*, bool *showDebugGraphics* = `false`)

Dibuja el escenario en la pantalla en la posicion indicada.

Parameters

screen	Pantalla en la que dibujar el escenario.
position	Punto central donde dibujar el laberinto.

Precondition

Se debe haber especificado la plantilla mediante setSpriteSheet y vincular las imagenes de la pared y el pasillo mediante setWallSprite y setCorridorSprite.

3.9.1.3 void Scenario::setCorridorSprite (unsigned int *pos*)

Indica qué fila de la plantilla asignada corresponde al sprite del pasillo.

Parameters

pos	Posicion de la fila en la plantilla.

3.9.1.4 void Scenario::setWallSprite (unsigned int *pos*)

Indica qué fila de la plantilla asignada corresponde al sprite del muro.

Parameters

pos	Posicion de la fila en la plantilla.

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

- MulticocoSDL/MulticocoSDL/scenario.h
- MulticocoSDL/MulticocoSDL/scenario.cpp

3.10 SDLMain Class Reference

Inheritance diagram for SDLMain:

Collaboration diagram for SDLMain:

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

- MulticocoSDL/MulticocoSDL/SDLMain.h

3.11 Sound Class Reference

Public Member Functions

- **Sound** (const char ∗file)
- bool **play** ()
- bool **playLoop** ()
- bool **isPlaying** ()
- void **stop** ()

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

- MulticocoSDL/MulticocoSDL/sound.h
- MulticocoSDL/MulticocoSDL/sound.cpp

3.12 Sprite Class Reference

Public Member Functions

- Sprite (SDL_Surface ∗img, int animations, int w, int h)

  Construye un Sprite animado a partir de una imagen con varios frames.

- void **render** (SDL_Surface *screen, Vector2D &pos)
- void **nextFrame** ()
- void **setFrameSkip** (unsigned int f)

3.12.1 Constructor & Destructor Documentation

3.12.1.1 Sprite::Sprite (SDL_Surface * *img*, int *animations*, int *w*, int *h*)

Construye un Sprite animado a partir de una imagen con varios frames.

Parameters

img	Sprite previamente cargado. Suele ser la fila de un SpriteSheet.
animations	Numero de frames de la animacion.
w	Ancho de cada frame del sprite.
h	Altura de cada frame del sprite.

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

- MulticocoSDL/MulticocoSDL/sprite.h
- MulticocoSDL/MulticocoSDL/sprite.cpp

3.13 SpriteSheet Class Reference

Public Member Functions

- SpriteSheet (const char *img, int w, int h, int *animations, int nAnimations)

  Constructor de la plantilla de sprites.
- unsigned int spriteWidth ()

  Ancho de cada frame individual del sprite.
- unsigned int spriteHeight ()

  Alto de cada frame individual del sprite.
- void bindAnimation (unsigned int pos, const char *name)

  Vincula un nombre de animacion con una fila de la plantilla de animaciones. De esta forma es mas sencillo trabajar con distintas animaciones sin tener que saber el orden en que se encuentran en la plantilla. Por ejemplo, la animacion de moverse hacia la derecha esta en la primera fila de la plantilla, por lo que podemos vincular la posicion 0 de la plantilla con el nombre de animacion 'RIGHT' mediante blinAnimation(0,"RIGHT"). De esta forma nos podremos referir a dicha animacion mas adelante mediante el nombre "RIGHT".
- void setAnimation (const std::string name)

  Cambia la animacion actual que se renderiza.
- void nextFrame ()

  Pasa al siguiente frame de la animacion actual (si el frameskip lo permite).
- void setFrameSkip (const std::string name, unsigned int value)

  Ajusta el frame skip de una animacion.
- void setFrameSkip (const unsigned int value)

  Ajusta el frameskip de todas las animaciones.
- void render (SDL_Surface *screen, Vector2D &pos)

  Dibuja la imagen en la posicion indicada.
- void pause ()

  Pausa la animacion actual.
- void resume ()

  Reanuda la animacion actual.

3.13.1 Constructor & Destructor Documentation

3.13.1.1 SpriteSheet::SpriteSheet (const char * *img,* int *w,* int *h,* int * *animations,* int *nAnimations*)

Constructor de la plantilla de sprites.

Parameters

img	Ruta hasta el archivo de plantilla.
w	Ancho individual de cada elemento de la plantilla.
h	Alto individual de cada elemento de la plantilla, que coincide con el alto de cada fila.
animations	Vector con el numero de frames de cada animacion (columnas de cada fila).
nAnimations	Numero de animaciones distintas que tiene la plantilla (numero de filas).

3.13.2 Member Function Documentation

3.13.2.1 void SpriteSheet::bindAnimation (unsigned int *pos,* const char * *name*)

Vincula un nombre de animacion con una fila de la plantilla de animaciones. De esta forma es mas sencillo trabajar con distintas animaciones sin tener que saber el orden en que se encuentran en la plantilla. Por ejemplo, la animacion de moverse hacia la derecha esta en la primera fila de la plantilla, por lo que podemos vincular la posicion 0 de la plantilla con el nombre de animacion 'RIGHT' mediante blinAnimation(0,"RIGHT"). De esta forma nos podremos referir a dicha animacion mas adelante mediante el nombre "RIGHT".

Parameters

pos	Fila de la plantilla a vincular.
name	Nombre de animacion que queremos vincular con la fila.

Postcondition

Si se vincular una animacion ya vinculada anteriormente, el anterior nombre deja de ser valido.

3.13.2.2 void SpriteSheet::render (SDL_Surface * *screen,* Vector2D & *pos*)

Dibuja la imagen en la posicion indicada.

Parameters

screen	Ventana en la que dibujar la imagen.
pos	Posicion en la ventana donde dibujar.

3.13.2.3 void SpriteSheet::setAnimation (const std::string *name*)

Cambia la animacion actual que se renderiza.

Parameters

name	Nombre de la nueva animacion a renderizar.

Precondition

La animacion debe estar vinculada con una fila de la plantilla mediante bindAnimation(...).

3.13.2.4 void SpriteSheet::setFrameSkip (const std::string *name,* unsigned int *value*)

Ajusta el frame skip de una animacion.

Parameters

name	Nombre de la animacion.
value	Cantidad de actualizaciones a ignorar antes de pasar al siguiente frame de la animacion.

Precondition

La animacion debe estar vinculada con una fila de la plantilla mediante bindAnimation(...).

3.13.2.5 void SpriteSheet::setFrameSkip (const unsigned int *value*)

Ajusta el frameskip de todas las animaciones.

Parameters

value	Cantidad de actualzaciones a ignorar antes de pasar al siguiente frame de la animacion.

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

- MulticocoSDL/MulticocoSDL/spritesheet.h
- MulticocoSDL/MulticocoSDL/spritesheet.cpp

3.14 Vector2D Class Reference

Public Member Functions

- Vector2D ()
- Vector2D (float x, float y)
- Vector2D (const Vector2D &orig)
- ~Vector2D ()
- Vector2D & operator= (const Vector2D &vector)
- Vector2D operator+ (const Vector2D &vector)
- Vector2D operator* (float f)
- Vector2D **operator-** (const Vector2D &vector)
- bool **operator!=** (const Vector2D &vector)
- bool **operator==** (const Vector2D &vector)
- float **x** ()
- float **y** ()
- void **setX** (int x)
- void **setY** (int y)
- float lengthSquared ()
- float length ()
- float distanceSquared (const Vector2D &vector)
- float distance (const Vector2D &vector)
- Vector2D & normalize ()
- std::string **toString** ()

3.14.1 Constructor & Destructor Documentation

3.14.1.1 Vector2D::Vector2D ( )

Constructor por defecto. Inicializa los valores de ambas coordenadas a 0.

3.14.1.2 Vector2D::Vector2D (float *x*, float *y*)

Constructor por parametros.

Parameters

x	Coordenada en el eje x.
y	Coordenada en el eje y.

3.14.1.3 Vector2D::Vector2D (const Vector2D & *orig*)

Constructor copia.

Parameters

orig	Vector a copiar.

3.14.1.4 Vector2D::~Vector2D ( )

Destructor.

3.14.2 Member Function Documentation

3.14.2.1 float Vector2D::distance (const Vector2D & *vector*)

Calcula la distancia hasta otro vector.

Parameters

vector	Vector respecto al cual calcular la distancia.

Returns

Distancia hasta el vector indicado.

3.14.2.2 float Vector2D::distanceSquared (const Vector2D & *vector*)

Calcula la distancia cuadratica hasta otro vector.

Parameters

vector	Vector respecto al cual calcular la distancia.

Returns

Distancia cuadratica hasta el vector indicado.

3.14.2.3 float Vector2D::length ( )

Calcula la longitud del vector.

Returns

Longitud del vector.

3.14.2.4 float Vector2D::lengthSquared ( )

Calcula la longitud cuadratica del vector.

Returns

Longitud cuadratica (sin aplicar la raiz) del vector.

3.14.2.5 Vector2D & Vector2D::normalize ( )

Normaliza el vector.

3.14.2.6 Vector2D Vector2D::operator∗ (float *f*)

Multiplicacion por un factor.

Parameters

f	Factor de escala del vector.

Returns

Vector resultado de la escala por el factor.

3.14.2.7 Vector2D Vector2D::operator+ (const Vector2D & *vector*)

Operador de suma.

Parameters

vector	Vector a sumar con el actual.

Returns

Nuevo vector resultado de la suma de los dos.

3.14.2.8 Vector2D & Vector2D::operator= (const Vector2D & *vector*)

Operador de asignacion.

Parameters

vector	Vector que asignar al actual.

Returns

Referencia al vector ya asignado.

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

- MulticocoSDL/MulticocoSDL/vector2d.h
- MulticocoSDL/MulticocoSDL/vector2d.cpp

3.15 Window Class Reference

Public Member Functions

- **Window** (int w, int h, string title)
- void **startMainLoop** ()
- void **setFullScreen** (bool full)

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

- MulticocoSDL/MulticocoSDL/window.h
- MulticocoSDL/MulticocoSDL/window.cpp

Index