Sprites

Ventanas

Reglasdeljuego

Rocas

Enemigo

Cañon

Disparo

Registro:  
  
  
  
  
class reglas\_juego : public QObject

{

Q\_OBJECT

public:

reglas\_juego(QGraphicsView \*graph, QVector<QLabel \*> game\_labels);

~reglas\_juego();

void key\_event(QKeyEvent \*event);

public slots:

void move\_enemy();  
  
private:   
void set\_canon\_keys();

void generate\_map();

void setup\_scene();

void setup\_canon();

void setup\_enemigo();

void setup\_disparo();

void pocision\_disparo();  
  
bool Bernoulli\_events(); //numero aleatorio de rocas  
  
bool object\_right\_movement(QGraphicsPixmapItem \*item, unsigned int speed);

bool object\_left\_movement(QGraphicsPixmapItem \*item, unsigned int speed);

bool object\_up\_movement(QGraphicsPixmapItem \*item, unsigned int speed);

bool object\_down\_movement(QGraphicsPixmapItem \*item, unsigned int speed);  
  
signals:

void game\_scene\_changed();

};  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
#ifndefe ROCAS\_H

#define ROCAS\_H

#include <QObject>

#include <QGraphicsPixmapItem>

#include "sprites.h"

#define blocks\_pixel\_x\_size 16

#define blocks\_pixel\_y\_size 16

class rocas : public QObject, public QGraphicsPixmapItem

{

Q\_OBJECT

public:

escenario(unsigned int scale, unsigned int type);

~escenario();

unsigned int get\_type();

private:

sprites \*pixmap;

unsigned int scale, type;

QRect set\_complete\_sprites();

};

#endif // ESCENARIO\_H

#ifndef ENEMIGO\_H

#define ENEMIGO\_H

#include <QObject>

#include <QGraphicsPixmapItem>

#include <QTimer>

#include "sprites.h"

#define enemigo\_pixel\_x\_size 16

#define enemigo\_pixel\_y\_size 16

#define enemigo\_speed 10

class enemigo : public QObject, public QGraphicsPixmapItem

{

Q\_OBJECT

public:

enemigo(unsigned int scale);

~enemigo();

public slots:

void mover(int direccion, bool is\_valid); // Slot para mover al enemigo con movimiento ondulatorio

private:

QRect set\_complete\_sprites();

void set\_animations();

void set\_left\_animation();

void set\_right\_animation();

void set\_up\_animation();

void set\_down\_animation();

void set\_death\_animation();

sprites \*pixmap\_manag; // Clase para manejar los gráficos del enemigo

};

#endif// ENEMIGO\_H

#ifndef CANON\_H

#define CANON\_H

#include <QObject>

#include <QGraphicsPixmapItem>

#include <QTimer>

#include "sprites.h"

#define canon\_pixel\_x\_size 16

#define canon\_pixel\_y\_size 16

Class canon : public QObject, public QGraphicsPixmapItem

{

Q\_OBJECT

Signals:  
 void aPuntodeDisparar(); //definicion de señal

public:

canon(unsigned int scale);

~canon();

private slots:

void changeCanonSprite();

private:

QRect set\_complete\_sprites();

void set\_animations();

void set\_activa\_animation();

sprites \*pixmap\_mana; // Clase para manejar los gráficos del enemigo

QTimer \*timer;

int frame\_index;

};

#endif DISPARO\_H  
  
#ifndef DISPARO\_H

#define DISPARO\_H

#include <QObject>

#include <QGraphicsPixmapItem>

#include <QTimer>

#include "sprites.h"

#define disparo\_pixel\_x\_size 16

#define disparo\_pixel\_y\_size 16

class disparo : public QObject, public QGraphicsPixmapItem

{

Q\_OBJECT

public:

disparo(unsigned int scale);

~disparo();

private slots:

void changeDisparoSprite();

private:

QRect set\_complete\_sprites();

void set\_animations();

void set\_activa\_animation();

sprites \*pixmap\_mana; // Clase para manejar los gráficos del enemigo

QTimer \*timer;

int frame\_index;

};

#ifndef SPRITES\_H

#define SPRITES\_H

#include <QRect>

#include <QPixmap>

#include <QVector>

class sprites

{

public:

sprites(QString main\_pixmap, unsigned int scale);

void set\_design\_size(unsigned int x, unsigned int y);

void cut\_character\_pixmap(QRect size);

void add\_new\_animation(QRect size, unsigned int number);

QPixmap get\_current\_pixmap(unsigned int animation);

QPixmap get\_fixed\_image(QRect size);

private:

QPixmap \*main\_pixmap, \*character\_pixmap;

QVector<QRect> animations;

QVector<unsigned int> animations\_size;

unsigned int width, height, animation\_counter, scale;

};

#endif // SPRITES\_H

#ifndef VENTANAS\_H

#define VENTANAS\_H

#include <QMainWindow>

#include "reglas\_juego.h"

#include <QKeyEvent>

QT\_BEGIN\_NAMESPACE

namespace Ui { class ventanas; }

QT\_END\_NAMESPACE

class ventanas : public QMainWindow

{

Q\_OBJECT

public:

ventanas(QWidget \*parent = nullptr);

~ventanas();

void keyPressEvent(QKeyEvent \*event);

private:

Ui::ventanas \*ui;

reglas\_juego \*game;

void setup\_game\_rules();

private slots:

void set\_mainwindow();

};

#endif // VENTANAS\_H

class Registro {

private:

std::ifstream archivo; // Objeto ifstream para lectura.

public:

ManejadorArchivo(const char\* nombreArchivo);

void leerArchivo() ;  
  
void escribirArchivo();

void cerrarArchivo() ;

~ManejadorArchivo() ;