```
#ifndef _VIEW_
#define _VIEW_

#include <SFML/Graphics.hpp>
#include <SFML/Audio.hpp>
#include "SlidingBackground.h"
#include "AutonomousElement.h"
#include "Animatedgraphicelement.h"
#include "Constantes.h"
#include "MenuFixed.h"
#include "MenuSliding.h"
#include "ClickableElement.h"

/*
 * ----------------------------------------------------------------------------
 * "THE BEER-WARE LICENSE" (Revision 42):
 * Antoine BOULANGÃM-^I & Pierre_Elliot CABRERA wrote this file. As long as you
retain this notice you
 * can do whatever you want with this stuff. If we meet some day, and you think
 * this stuff is worth it, you can buy me a beer in return
 * ----------------------------------------------------------------------------
 */


class Model;

class GraphicElement;

class MovableElement;

class View {
private:

    int _w, _h;
    int _positionScore;

    LANGUAGE _language;

    sf::Music _music;

    sf::RenderWindow * _window;
    Model * _model;

    sf::Texture _backgroundAvant;
    sf::Texture _backgroundArriere;
    SlidingBackground * _slidingBackgroundArriere;
    SlidingBackground * _slidingBackgroundAvant;

    sf::Texture _textureNinja;
    AnimatedGraphicElement * _ninjaSprite;

    sf::Texture _hpBarTexture;
    GraphicElement * _hpBar;
    sf::Texture _containerTexture;
    GraphicElement * _container;

    sf::Texture _textureExplosion;
    AnimatedGraphicElement * _explosionSpriteTall;
    AnimatedGraphicElement * _explosionSpriteShort;
    AnimatedGraphicElement * _explosionSpriteFlying;


    sf::Texture _textureBouton;
    GraphicElement * _buttonSprite;


    sf::Texture _textureObstacleTall;
    GraphicElement * _obstacleTallSprite;
    sf::Texture _textureObstacleShort;
```



```
    GraphicElement * _obstacleShortSprite;
    sf::Texture _textureObstacleFlying;
    GraphicElement * _obstacleFlyingSprite;

    sf::Texture _textureBonus;
    GraphicElement * _bonusSprite;

    std::map<const AutonomousElement *, GraphicElement *> _elementToGraphicEleme
nt;
    std::map<const AutonomousElement *, AnimatedGraphicElement *> _elementToAnim
atedGraphicElement;

    sf::Font * _fontScore;
    sf::Text _textScore;

    sf::Font * _fontClassic;
    sf::Text _textClassic;

    sf::Clock _clockGeneral;
    sf::Time _elapsedEvolution;

    std::pair<std::string, int> _highscore1;
    std::pair<std::string, int> _highscore2;
    std::pair<std::string, int> _highscore3;
    char _first, _second, _third;

    sf::Texture _textureButtonMenu;
    ClickableElement *_play;
    ClickableElement *_quit;
    ClickableElement *_highscore;
    sf::Texture _textureSound;
    ClickableElement *_sound;
    ClickableElement *_back;

    MenuSliding *_menuSlidingPrincipal;
    MenuSliding *_menuSlidingHighscore;

    SlidingBackground * _slidingBackgroundArriereMenu;
    SlidingBackground * _slidingBackgroundAvantMenu;

    sf::Texture _textureIntroduction;
    GraphicElement * _introduction;

public:

    //Constructeur
    View(int w, int h);

    //Destructeur
    ~View();

    //MÃ©thodes
    void setModel(Model * model);
    void draw();
    void drawColoredBackground();
    void drawHp();
    void synchronize();
    bool treatEvents();
    void readFile();
    void synchronizeHighscores(std::string name);
    void writeFile(std::string name);
    void registerHighscore();
};
#endif
```