Aplication.h

1. #pragma once
2. #include <SFML/Graphics.hpp>
3. #include "Map.h"
4. #include "Player.h"
5. #include "Enemy.h"
6. #include "Base.h"
7. #include "Message.h"
8. class Application : private sf::NonCopyable {
9. private:
10. bool gameOver;//начало игры
11. bool gameStarted;//конец игры
12. int frags;
13. sf::RenderWindow mWindow;
14. Base mBase;
15. Map map;
16. Player mPlayer;
17. Enemy \*packOfEnemies;
18. //функции вывода сообщений
19. Message msgStart;//старт
20. Message msgOver;//конец
21. Message msgLost;
22. Message msgWon;//победа
23. public:
24. Application();
25. void process\_events();
26. void update(const sf::Int64 &time);
27. void render();
28. };

Aplication.cpp

1. #include <SFML/Graphics.hpp>
2. #include "Application.h"
3. Application::Application()
4. : mWindow(sf::VideoMode(720, 672), "Battle City"), gameOver(false), gameStarted(false),//основное окно
5. msgStart(90, 330, "Press \'Enter\' to start"), msgOver(250, 300, "Game over"),
6. msgLost(260, 350, "You lost"), msgWon(265, 350, "You won"), frags(0) {
7. sf::Clock clock;
8. packOfEnemies = new Enemy[4]{ Enemy(52,31), Enemy(147,391), Enemy(532,391), Enemy(628,31) };
9. while (mWindow.isOpen()) {
10. sf::Int64 time = clock.getElapsedTime().asMicroseconds();
11. clock.restart();
12. time /= 800;
13. process\_events();
14. if (sf::Keyboard::isKeyPressed(sf::Keyboard::Enter))
15. gameStarted = true;
16. if (gameStarted && !gameOver)//если начало и не конец
17. update(time);
18. render();
19. }
20. }
21. void Application::process\_events() {
22. sf::Event event;
23. while (mWindow.pollEvent(event)) {
24. switch (event.type) {
25. case sf::Event::Closed:
26. mWindow.close();
27. break;
28. }
29. }
30. }
31. void Application::update(const sf::Int64 &time) {
32. for (int i(0); i < 4; ++i)
33. if (!packOfEnemies[i].life)//если все враги мертвы
34. ++frags;
35. else {
36. frags = 0;
37. break;
38. }
39. if (frags == 4)
40. gameOver = true;//конец игры
41. if (!mPlayer.life)//если мертв игрок
42. gameOver = true;//конец игры
43. bool collision;
44. for (int i(0); i < 4; ++i) {
45. collision = mPlayer.mSprite.getGlobalBounds().intersects(packOfEnemies[i].mSprite.getGlobalBounds());
46. if (collision)
47. break;
48. }
49. if (mPlayer.life)
50. mPlayer.update(time, map, collision);
51. for (int i(0); i < 4; ++i) {
52. if (packOfEnemies[i].life) {//если есть в наличии враги
53. packOfEnemies[i].update(time, map, collision);//инициализируем врага
54. if (packOfEnemies[i].bullet.mSprite.getGlobalBounds().intersects(mPlayer.mSprite.getGlobalBounds())
55. && packOfEnemies[i].bullet.present) {
56. mPlayer.collapse();//- жизнь игрока
57. packOfEnemies[i].bullet.present = false;//если пуля врага пересечется с игроком, то пуля мертва
58. }
59. if (packOfEnemies[i].bullet.mSprite.getGlobalBounds().intersects(mBase.mSprite.getGlobalBounds())
60. && packOfEnemies[i].bullet.present) {
61. mBase.life = false;//если пуля попадет в штаб, то штаб мертв
62. gameOver = true;//игра окончена
63. }
64. if (mPlayer.bullet.mSprite.getGlobalBounds().intersects(packOfEnemies[i].mSprite.getGlobalBounds())//если пуля игрока попадет во врага
65. && mPlayer.bullet.present) {
66. packOfEnemies[i].collapse();//минус один враг
67. mPlayer.bullet.present = false;//пуля мертва
68. }
69. }
70. }
71. if (mPlayer.bullet.mSprite.getGlobalBounds().intersects(mBase.mSprite.getGlobalBounds())//если пуля игрока попадет в штаб
72. && mPlayer.bullet.present) {
73. mBase.life = false;//база мертва
74. gameOver = true;//конец игры
75. }
76. }
77. void Application::render() {//визуализация приложения
78. mWindow.clear();//очистка окна
79. map.draw(mWindow);//рисуем карту
80. if (mPlayer.life)//если игрок жив
81. mWindow.draw(mPlayer.mSprite);//рисуем игрока
82. if (mPlayer.bullet.present) mWindow.draw(mPlayer.bullet.mSprite);//если пули есть- рисуем пули
83. for (int i(0); i < 4; ++i) {//цикл от 0 до 4
84. if (packOfEnemies[i].bullet.present)// если у врагов есть пули
85. mWindow.draw(packOfEnemies[i].bullet.mSprite);//то рисуем пули
86. if (packOfEnemies[i].life)//если враг жив
87. mWindow.draw(packOfEnemies[i].mSprite);//рисуем врага
88. }
89. if (mBase.life)//если база жива
90. mWindow.draw(mBase.mSprite);//рисуем базу
91. if (!gameStarted)//если не начало игры
92. msgStart.print(mWindow);//окно о запуске игры
93. if (gameOver) {//если конец игры
94. msgOver.print(mWindow);//тоже какое-то окно
95. if (!mBase.life || !mPlayer.life)//если база мертва или мертв игрок
96. msgLost.print(mWindow);
97. else
98. msgWon.print(mWindow);
99. }
100. mWindow.display();//отрисовка окна
101. }

Tank.h

1. #pragma once
2. #include <SFML/System/String.hpp>
3. #include <SFML/Graphics/Texture.hpp>
4. #include <SFML/Graphics/Sprite.hpp>
5. #include "Bullet.h"
6. #include <utility>
7. class Map;
8. class Tank : private sf::NonCopyable {
9. protected:
10. float mX, mY, mWidth, mHeight, mDx, mDy, mSpeed;//координаты ч,у, ширина, высота, ускорение по х, у, скорость
11. int mDir;//направление
12. float mCurrentFrame;//текущий кадр
13. bool mCollision;//столкновение
14. sf::String mFile;//файл
15. sf::Texture mTexture;//текстура
16. public:
17. explicit Tank(const float &x, const float &y, const float &width, const float &height, const sf::String &file);//явный конструктор(защищает от переопределения типов: int в float и тд)
18. bool life;//жизнь
19. Bullet bullet;//класс пули
20. sf::Sprite mSprite;//спрайт
21. void animate(const sf::Int64 &time);//функция анимации
22. void collapse();//функция попадиния
23. virtual void move(const sf::Int64 &time) {}//переопределяемая функция движения
24. void map\_interaction(Map &map);//взаимодействие с картой
25. void tank\_interaction();//взаимодействие с танком
26. };

Tank.cpp

1. #include "Tank.h"
2. #include <Map.h>
3. Tank::Tank(const float &x, const float &y, const float &width, const float &height, const sf::String &file)
4. : life(true), mX(x), mY(y), mDx(0.f), mDy(0.f), mFile(file), mDir(0), mCollision(false),
5. mCurrentFrame(0.f), mSpeed(0.f), mWidth(width), mHeight(height),
6. bullet(x, y) {//координаты пули
7. mTexture.loadFromFile(mFile);
8. mSprite.setTexture(mTexture);
9. mSprite.setTextureRect(sf::IntRect(78, 39, mWidth, mHeight));
10. mSprite.setPosition(x, y);
11. }
12. void Tank::animate(const sf::Int64 &time) {//функция анимации танка
13. mCurrentFrame += 0.005f \* time;
14. if (mCurrentFrame >= 2)//если текущий кадр больше либо равно 2
15. mCurrentFrame -= 2;//то возвращаемся в начало
16. if (mDx > 0.f)
17. mSprite.setTextureRect(sf::IntRect(0 + 39 \* (int)mCurrentFrame, 0, 39, 39));//едем вправо
18. if (mDx < 0.f)
19. mSprite.setTextureRect(sf::IntRect(78 + 39 \* (int)mCurrentFrame, 0, 39, 39));//влево
20. if (mDy > 0.0f)
21. mSprite.setTextureRect(sf::IntRect(0 + 39 \* (int)mCurrentFrame, 39, 39, 39));//вверх
22. if (mDy < 0.0f)
23. mSprite.setTextureRect(sf::IntRect(78 + 39 \* (int)mCurrentFrame, 39, 39, 39));//вниз
24. }
25. void Tank::collapse() {//в танк попала пуля
26. life = false;//танк мертв
27. bullet.present = false;//пуля мертва
28. mSprite.setPosition(0, 0);//танк в начала
29. }
30. void Tank::map\_interaction(Map &map) {//взаимодействие с картой
31. for (int i = mY / 24; i < (mY + mHeight) / 24; ++i)
32. for (int j = mX / 24; j < (mX + mWidth) / 24; ++j) {
33. char tile = map.get\_tile(i, j);
34. if (tile >= '0' && tile <= '2') {//если символ карты "0" и "2"
35. if (mDy > 0.f)//если едем вверх
36. mY = i \* 24 - mHeight;//не знаю точно что это( наверное, обновление координаты У)
37. if (mDy < 0.f)
38. mY = (float)i \* 24 + 24;
39. if (mDx > 0.f)
40. mX = j \* 24 - mWidth;
41. if (mDx < 0.f)
42. mX = (float)j \* 24 + 24;
43. }
44. }
45. }
46. void Tank::tank\_interaction() {//взаимодействие с танком
47. if (mDy > 0.f)
48. mY -= 0.3f;
49. if (mDy < 0.f)
50. mY += 0.3f;
51. if (mDx > 0.f)
52. mX -= 0.3f;
53. if (mDx < 0.f)
54. mX += 0.3f;
55. }

Player.h

1. #pragma once
2. #include <SFML/Window/Keyboard.hpp>
3. #include "Tank.h"
4. class Player : public Tank {//дочерний класс плеер(родитель "танк")
5. public:
6. Player();
7. void move(const sf::Int64 &time);//функция движения
8. void update(const sf::Int64 &time, Map &map, const bool &collision);//инициализация
9. };

Player.cpp

1. #include <SFML/Window/Keyboard.hpp>
2. #include "Player.h"
3. #include <Map.h>
4. using sf::Keyboard;
5. Player::Player()
6. : Tank(244, 600, 39, 39, "battle-city-src/media/playerSprites.png") {//инициализация игрока
7. }
8. void Player::move(const sf::Int64 &time) {
9. if (Keyboard::isKeyPressed(Keyboard::Left)) {
10. mDir = 1;
11. mSpeed = 0.11f;
12. }
13. if (Keyboard::isKeyPressed(Keyboard::Right)) {
14. mDir = 0;
15. mSpeed = 0.11f;
16. }
17. if (Keyboard::isKeyPressed(Keyboard::Up)) {
18. mDir = 3;
19. mSpeed = 0.11f;
20. }
21. if (Keyboard::isKeyPressed(Keyboard::Down)) {
22. mDir = 2;
23. mSpeed = 0.11f;
24. }
25. switch (mDir) {
26. case 0:
27. mDx = mSpeed;//направление вправо
28. mDy = 0;
29. break;
30. case 1:
31. mDx = -mSpeed;//влево
32. mDy = 0;
33. break;
34. case 2:
35. mDx = 0;
36. mDy = mSpeed//вверх;
37. break;
38. case 3:
39. mDx = 0;
40. mDy = -mSpeed;//вниз
41. break;
42. }
43. if (mCollision) tank\_interaction();//если столкновение с танком
44. mX += mDx \* time;
45. mY += mDy \* time;
46. }
47. void Player::update(const sf::Int64 &time, Map &map, const bool &collision) {
48. mCollision = collision;
49. move(time);
50. mSpeed = 0.f;//скорость
51. mSprite.setPosition(mX, mY);//начальная позиция
52. animate(time);//анимация
53. map\_interaction(map);//взаимодействие с картой
54. bullet.update(map, time, mX, mY, mDir);//инициализация пули
55. if (!bullet.present)//если не стреляем
56. if (Keyboard::isKeyPressed(Keyboard::Space)) {//если нажат пробел
57. bullet.timeBeforeShot += time;
58. if (bullet.timeBeforeShot > 10.f) {//задержка между пулями
59. bullet.present = true;//стреляем
60. bullet.timeBeforeShot = 0.f;//перезаряжаем))
61. }
62. }
63. }

Enemy.h

1. #pragma once
2. #include "Tank.h"
3. class Enemy : public Tank {
4. private:
5. float timeBeforeMoving;//время до переезда
6. float timeBeforeShot;//время до стрельбы
7. public:
8. Enemy(const float &x, const float &y);//координаты
9. Enemy(const Enemy &copy) : Enemy(copy.mX, copy.mY){}
10. void move(const sf::Int64 &time);//движение
11. void shoot(const float &time);//стрельба
12. void update(const sf::Int64 &time, Map &map, const bool &collision);//обновление
13. };

Enemy.cpp

1. #include <SFML/Graphics.hpp>
2. Enemy::Enemy(const float &x, const float &y)
3. : Tank(x, y, 39, 39, "battle-city-src/media/enemySprites.png"), timeBeforeMoving(0.f), timeBeforeShot(0.f) {
4. }
5. void Enemy::move(const sf::Int64 &time) {
6. mSpeed = 0.1f;//скорость
7. timeBeforeMoving += time;//???
8. if (timeBeforeMoving > 1000 + rand() % 501) {//отрезок времени
9. mDir = rand() % 4;//рандомно выбирается направление
10. timeBeforeMoving = 0.f;//обновляется переменная времени
11. }
12. switch (mDir) {//направление
13. case 0:
14. mDx = mSpeed;//вправо
15. mDy = 0;
16. break;
17. case 1:
18. mDx = -mSpeed;//влево
19. mDy = 0;
20. break;
21. case 2:
22. mDx = 0;
23. mDy = mSpeed;//вверх
24. break;
25. case 3:
26. mDx = 0;
27. mDy = -mSpeed;//вниз
28. break;
29. }
30. if (mCollision) tank\_interaction();//если столкновение с танком
31. mX += mDx \* time;//не знаю, что происходит... наверное, убегает)))
32. mY += mDy \* time;
33. }
34. void Enemy::shoot(const float &time) {
35. timeBeforeShot += time;//время между пулями
36. if (timeBeforeShot > 1000.f) {
37. if (!bullet.present)
38. if (rand() % 2)
39. bullet.present = true;//стреляет
40. timeBeforeShot = 0.f;
41. }
42. }
43. void Enemy::update(const sf::Int64 &time, Map &map, const bool &collision) {//инициализация врага
44. mCollision = collision;
45. move(time);
46. mSpeed = 0.f;//скорость
47. mSprite.setPosition(mX, mY);//место "рождения"
48. animate(time);//анимация рождения
49. map\_interaction(map);//взаимодействие с картой
50. bullet.update(map, time, mX, mY, mDir);//инициализация пули
51. shoot(time);//стрельба

map.h

1. #pragma once
2. #include <SFML/System/String.hpp>
3. #include <SFML/Graphics/Texture.hpp>
4. #include <SFML/Graphics/Sprite.hpp>
5. #include <SFML/Graphics/RenderWindow.hpp>
6. class Map : private sf::NonCopyable {
7. private:
8. static const int HEIGHT\_MAP = 28;
9. static const int WIDTH\_MAP = 30;
10. sf::String \*TileMap;
11. sf::Texture mTexture;
12. sf::Sprite mSprite;
13. public:
14. Map();
15. void break\_wall(const int &i, const int &j);//ломать стену
16. const char get\_tile(const int &i, const int &j) { return TileMap[i][j]; }// функция получения тайла
17. void draw(sf::RenderWindow &window);//рисует окно
18. };

Map.cpp

1. #include <Map.h>
2. Map::Map() {
3. TileMap = new sf::String[HEIGHT\_MAP] {
4. "000000000000000000000000000000",
5. "00 00",
6. "00 00",
7. "00 11 11 11 11 11 11 00",
8. "00 11 11 11 11 11 11 00",
9. "00 11 11 11 11 11 11 00",
10. "00 11 11 11 11 11 11 00",
11. "00 11 11 112211 11 11 00",
12. "00 11 11 112211 11 11 00",
13. "00 11 11 11 11 11 11 00",
14. "00 11 11 11 11 00",
15. "00 11 11 11 11 00",
16. "00 11 11 00",
17. "00 11 11 00",
18. "0011 1111 1111 1100",
19. "0022 1111 1111 2200",
20. "00 11 11 00",
21. "00 111111 00",
22. "00 11 11 111111 11 11 00",
23. "00 11 11 11 11 11 11 00",
24. "00 11 11 11 11 11 11 00",
25. "00 11 11 11 11 11 11 00",
26. "00 11 11 11 11 00",
27. "00 11 11 11 11 00",
28. "00 11 11 1111 11 11 00",
29. "00 1 1 00",
30. "00 1 1 00",
31. "000000000000000000000000000000",
32. };
33. mTexture.loadFromFile("battle-city-src/media/groundSprites.png");
34. mTexture.setSmooth(true);
35. mSprite.setTexture(mTexture);
36. }
37. void Map::break\_wall(const int &i, const int &j) {
38. TileMap[i][j] = ' ';
39. }
40. void Map::draw(sf::RenderWindow &window) {//функция отрисовки карты
41. for (int i(0); i < HEIGHT\_MAP; i++)
42. for (int j(0); j < WIDTH\_MAP; j++) {//проходим по всей карте
43. switch (TileMap[i][j]) {
44. case ' ':
45. mSprite.setTextureRect(sf::IntRect(0, 0, 24, 24));//рисуем тайлы
46. break;
47. case '0':
48. mSprite.setTextureRect(sf::IntRect(72, 0, 24, 24));
49. break;
50. case '1':
51. mSprite.setTextureRect(sf::IntRect(24, 0, 24, 24));
52. break;
53. case '2':
54. mSprite.setTextureRect(sf::IntRect(48, 0, 24, 24));
55. break;
56. }
57. mSprite.setPosition(j \* 24, i \* 24);//устанавливаем позиции
58. window.draw(mSprite);//рисуем карту
59. }
60. }

Bullet.h

1. #pragma once
2. #include <SFML/Graphics/Texture.hpp>
3. #include <SFML/Graphics/Sprite.hpp>
4. class Map;
5. class Bullet {//класс пули
6. private:
7. float mX, mY, mDx, mDy;//координаты, ускорение
8. int mDir;//направление
9. sf::Texture mTexture;//текстура
10. public:
11. bool present; //стреляем?
12. float timeBeforeShot;//время до выстрела
13. sf::Sprite mSprite;
14. explicit Bullet(const float &x, const float &y);//непереопределяемая функция пули( параметры: координаты)
15. void shoot() { present = true; }//функция стрельбы
16. void move(const sf::Int64 &time);
17. void update(Map &map, const sf::Int64 &time, float &x, const float &y, const int &dir);//инициализация пули
18. void map\_interaction(Map &map);//взаимодействие с картой
19. };

Bullet.cpp

1. #include <Bullet.h>
2. #include <Map.h>
3. Bullet::Bullet(const float &x, const float &y)
4. : present(false), mX(x), mY(y), mDir(0), mDx(0.f), mDy(0.f), timeBeforeShot(0.f) {
5. mTexture.loadFromFile("battle-city-src/media/bulletSprite.png");//загружаем текстуру
6. mSprite.setTexture(mTexture);
7. mSprite.setTextureRect(sf::IntRect(0, 0, 9, 12));//получаем картинку
8. mSprite.setPosition(x, y);//устанавливаем позицию
9. }
10. void Bullet::move(const sf::Int64 &time) {//функция движения пули
11. switch (mDir) {//направление
12. case 0:
13. mDx = 0.3f;//вправо
14. mDy = 0;
15. mSprite.setRotation(90.f);//спрайт поворачиваем на 90 градусов
16. break;
17. case 1:
18. mDx = -0.3f;//влево
19. mDy = 0;
20. mSprite.setRotation(-90.f);
21. break;
22. case 2:
23. mDx = 0;
24. mDy = 0.3f;//вверх
25. mSprite.setRotation(180.f);
26. break;
27. case 3:
28. mDx = 0;
29. mDy = -0.3f;//вниз
30. mSprite.setRotation(0.f);
31. break;
32. }
33. mX += mDx \* time;//?? задаем движение по иксу??
34. mY += mDy \* time;// то же по игрику
35. mSprite.setPosition(mX, mY);//устанавливаем позицию
36. }
37. void Bullet::update(Map &map, const sf::Int64 &time, float &x, const float &y, const int &dir) {//инициализация пули
38. if (present) {
39. move(time);//если пуля движется
40. map\_interaction(map);//взаимодействует с картой
41. }
42. else {
43. mDir = dir;//определяем направление
44. switch (mDir) {
45. case 0:
46. mX = x + 35.f;//пока не знаю, что это
47. mY = y + 15.f;
48. break;
49. case 1:
50. mX = x + 7.f;
51. mY = y + 23.f;
52. break;
53. case 2:
54. mX = x + 23.f;
55. mY = y + 32.f;
56. break;
57. case 3:
58. mX = x + 15.f;
59. mY = y + 5.f;
60. break;
61. }
62. mSprite.setPosition(mX, mY);//задаем позицию
63. }
64. }
65. void Bullet::map\_interaction(Map &map) {//взаимодействие с картой
66. for (int i = mY / 24; i < (mY + 12) / 24; ++i)
67. for (int j = mX / 24; j < (mX + 9) / 24; ++j) {//проходим все тайлы
68. char tile = map.get\_tile(i, j);//получить символ тайла
69. if (tile == '0' || tile == '2')//если тайл "0" или "2"
70. present = false;//ничего не происходит
71. if (tile == '1') {//если "1"
72. map.break\_wall(i, j);//стомать стену
73. present = false;// тайла нет
74. }
75. }
76. }

Base.h

1. #pragma once
2. #include <SFML/Graphics/Texture.hpp>
3. #include <SFML/Graphics/Sprite.hpp>
4. class Base : private sf::NonCopyable {
5. private:
6. sf::Texture mTexture;//текстура
7. public:
8. sf::Sprite mSprite;//спрайт
9. bool life;//жизнь
10. Base();
11. };

Base.cpp

1. #include "Base.h"
2. Base::Base()
3. : life(true) {
4. mTexture.loadFromFile("battle-city-src/media/baseSprite.png");//загружаем текстуру
5. mSprite.setTexture(mTexture);//
6. mSprite.setTextureRect(sf::IntRect(0, 0, 48, 48));//загружаем картинку
7. mSprite.setPosition(336, 600);//определяем позицию
8. }

Message.h

1. #pragma once
2. #include <SFML/Graphics/Text.hpp>
3. #include <SFML/Graphics/RenderWindow.hpp>
4. class Message {
5. private:
6. float mX, mY;//координаты окна
7. sf::Font mFont;//класс шрифта
8. sf::Text mStr;//текстовая строка
9. public:
10. explicit Message(const float &x, const float &y, const sf::String &str);//инициализация вывода сообщения
11. void print(sf::RenderWindow &window) { window.draw(mStr); }//функция печати
12. };

Message.cpp

1. //#include <SFML/System/String.hpp>
2. #include "Message.h"
3. Message::Message(const float &x, const float &y, const sf::String &str)
4. : mX(x), mY(y) {//инициализируем окно
5. mFont.loadFromFile("battle-city-src/media/PressStart2P.ttf");//загружаем шрифты
6. mStr.setFont(mFont);//отправляем шрифт в строку
7. mStr.setString(str);//отправляем строку
8. mStr.setPosition(mX, mY);//устанавливаем координаты
9. mStr.setCharacterSize(25);//установить размер символов
10. }