Федеральное государственное бюджетное образовательное учреждение высшего образования «Саратовский государственный технический университет имени Гагарина Ю.А.»

Институт прикладных информационных технологий и коммуникаций

Кафедра Информационная безопасность автоматизированных систем

Направление подготовки 10.03.01 Информационная безопасность

Расчётно-графическая работа по дисциплине «Языки программирования»

«Тема: Игра Digger»

Выполнил: студент 1 курса

учебной группы с-ИБС12

очной формы обучения

Ноздрин Алексей Владимирович

Проверил: ассистент каф. ИБС

Романчук С. П.

Саратов 2020

Аннотация:

В данной работе представлена игра диггер. В ней включалась работа с библиотекой SFML, а так же с классами и функциями. В работе используется простая пиксельная графика и банальные методы управления но это не делает ее программирование простым, для этого мне пришлось изучить пару нестандартных библиотек. Работа использует карту из массива строк и самых обычных монстров которые пытаться преследовать главного героя.

Содержание

1. Введение………………………………………………………………..4
2. Теоретическая часть………………………………………………..….5
3. Практическая часть…………………………………………………..6-8
4. Заключение…………………………………………………….……….9
5. Приложения…………………………………………………………10-16
6. Литература…………………………………………………….……….17

**Введение**

**Задание 8. Игра Digger**

Целью было создать игру, в которой главный герой можно передвигаться, копать в шахтах, собирать золото набирая очки, убегать от мобов, которые при соприкосновении с героем его убивают.

Я захотел немного усложнить и сделал игру не в консоли при помощи стандартных библиотек, а в отдельном окне с пиксельной графикой с помощью библиотеки SFML.

**Описание**

**1-среда разработки**

**2-язык разработки**

**3-библиотеки**

1) Программирование и компиляцию произвожу в программе Visual Studio 2019.

**Visual Studio** включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).

2) Использован язык программирования С++ а также библиотеки <SFML/Graphics.hpp> и <sstream>

**C++** широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создание операционных систем, разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также игр. Существует множество реализаций языка C++, как бесплатных, так и коммерческих и для различных платформ. Например, на платформе x86 это GCC, Visual C++, Intel C++ Compiler, Embarcadero (Borland) C++ Builder и другие. C++ оказал огромное влияние на другие языки программирования, в первую очередь на Java,C#, Golang, Solidity, PHP.

Так же для улучшения вида программы были добавлены некоторые изображения и шрифт.

**3) SFML** (англ. *Simple and Fast Multimedia Library* — простая и быстрая мультимедийная библиотека) — свободная кроссплатформенная мультимедийная библиотека. Написана на C++, но доступна также для C, C#, .Net, D, Java, Python, Ruby, OCaml, Go и Rust. Представляет собой объектно-ориентированный аналог SDL.

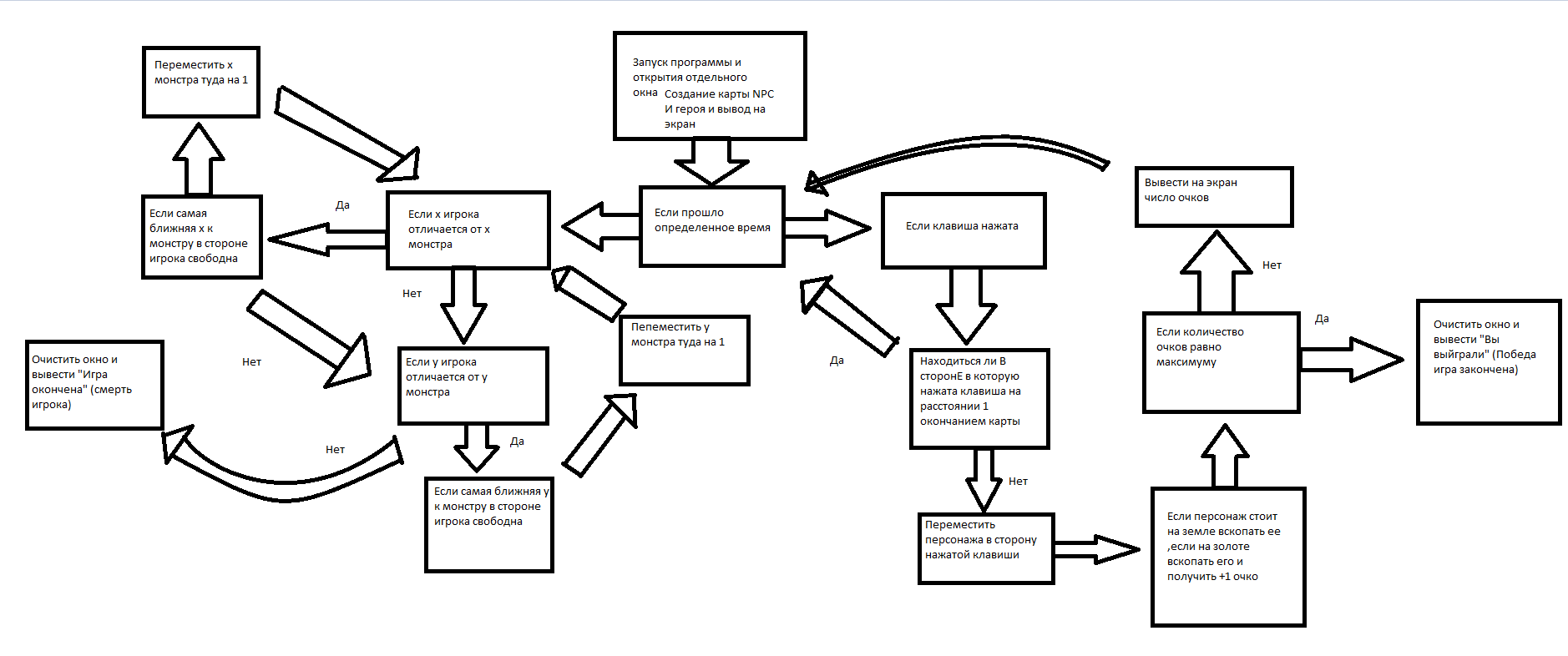
SFML содержит ряд модулей для простого программирования игр и мультимедиа приложений. Исходный код библиотеки предоставляется под лицензией zlib/png license.[3]

**Описание функций и программы в деле**

**Библиотека SFML**

В основном используется для создания простых игр и программ выведенных в отдельное окно, довольно простая библиотека так как все ключевые слова использованные в приложении говорят о своей функцией своим названием например слово Window –окно – открывает новое окно и тд.

**Блок схема:**



Описание алгоритмов

String Map[H]{}; -создание карты, каждый элемент которой потом будет заменен текстурой.

Класс Player отвечает за действия персонажа, имеет переменные x, y – координаты игрока , speed – скорость перемещения по карте , score – его очки, так же текстуру Miner и спрайт от неё miner так же имеет свои часы и таймер для задержки между его действиями

Конструктор игрока задает ему начальные координаты, скорость, создает модельку и устанавливает его в нужную часть карты.

Функция void control () {} отвечает за установку времени и перемещение персонажа по координатам x, y в зависимости от нажатой клавиши времени и условий на карте

Void draw () {} отвечает за установку персонажа в позицию на карте после вызова функции control () а так же убирает блоки на которые заходит персонаж, якобы “Копая” и увеличивает score если прокопано золото.

Класс Mob отвечает за движение мобов имеет также переменные x, y , speed отвечающие за координаты и скорость монстра, имеет текстуру и спрайт.

В конструкторе создается моделька, задается скорость, начальные координаты и начальное положение на карте

Функция void move () отвечает за перемещение монстра, она устроена так что если координаты x или y игрока не совпадает с координатами монстра, то он пытается их сократить, если прошло определенное время и путь монстра пуст.

Функция void draw () отвечает за установку монстра на координаты после функции move ().

После чего в главной функции int main () {} создаются текстуры блоков и фона, а также шрифт.

После открывается окно 1920 на 1080 пикселей

И создается 3 монстра

Далее начинается цикл while(window.isOpen()) {}

Который будет работать пока не закроется окно так же в нем описано стандартное условие закрывания окна, что бы при попытке его закрыть игра не крашилась. Это главная часть программы

В этом цикле создается еще один цикл while(Life)

Созданный для проверки наличия жизни главного игрока что бы после если по завершению игрок остался бы жив вывелась строчка “You win” а если бы он умер в ходе игра вывелось бы “You lose"

В этом цикле тоже есть цикл while (onGame) {}

Он обязан проверять не завершили ли вы игру (в нашем случае набрав 21 золото)

В нем рисуется на определенных координатах игрок, монстр, и просто текстуры мобов, а также указывается ваше количество очков в определенной координате на карте.

Отрисовкой текстур занимается цикл for(int i=0;i<H;i++){for(int j=0;j<W;j++) {}}

Которая в зависимости от знака на карте написанной изначально выводит в окно определенный спрайт

Так же отрисовывает персонажа и монстра и проверяет их координаты на совпадение для проверки на смерть игрока

Библиотека SFML/Graphics.hpp, главная библиотека программы отвечала в программе за текстуры, отдельное окно, время и за отслеживание нажатия клавиш.

**Заключение**

В итоге у меня получилась простенькая игра жанра Digger в написании которой я обучился библиотеке SFML, а также отработал свои навыки в знании массивов, циклов и функций.

Так же можно сделать разнообразную систему создания карт, добавить возможность убивать монстров, добавить новых монстров и ресурсов с бонусами, так же можно сделать главное меню игры и функционал выхода и перезапуска после окончания игры.

**Приложение**

#include <SFML/Graphics.hpp>

#include <sstream>

using namespace sf;

const int H = 17;

const int W = 33;

String Map[H]{

"NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN",

"NBBBBBBBBBBB BBBBBBBB0BBBBBBN",

"NBBBBB BBBBBBBBBBBBBBBBBBN",

"NBBBBBBBBB00BBB0BBBBBBBBB000BBBN",

"NBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBN",

"NBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBN",

"NBBBBBBBBBBBBBBBBBBBBBBBB000000N",

"NBBBBBB B BBBBBBB BBBBN",

"NBBBBBBBBBBBBB B BBBBBBBBBN",

"NBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBN",

"NBBBB0BBBBBBBBBBBBBBBBBBB000BBBN",

"NBBBBBBBB0BBBBBBBBBBBBBBBBBBBBBN",

"NBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBN",

"NBBBBBBBBBBBBB BBB B BBBBBN",

"NBBB BBBBBBBBBBBBBBBBB0BBBBBBBN",

"NBBBBBBBBBBB0BBBBBBBBBB0BBBBBBBN",

"NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN"

};

class Player

{

public:

int x, y, speed, score;// o-начальная координата по x p- начальная координата по y

Texture Miner;

Sprite miner;

bool timer = true, life = true;

Clock TIME;

Player(int x, int y, int speed)

{

score = 0; this->x = x; this->y = y;

this->speed = 1000 - speed;

Miner.loadFromFile("1.png");

miner.setTexture(Miner);

miner.setTextureRect(IntRect(0, 0, 45, 60));

miner.setPosition(x \* 60, y \* 60);

}

int GetScore()

{

return score;

}

void control()

{

float time = TIME.getElapsedTime().asMilliseconds();

if (time > speed)

{

timer = true;

TIME.restart();

}

if (Keyboard::isKeyPressed(Keyboard::Right) && timer && Map[y][x + 1] != 'N')

{

x++;

timer = false;

}

if (Keyboard::isKeyPressed(Keyboard::Up) && timer && Map[y + -1][x] != 'N')

{

y--;

timer = false;

}

if (Keyboard::isKeyPressed(Keyboard::Left) && timer && Map[y][x - 1] != 'N')

{

x--;

timer = false;

}

if (Keyboard::isKeyPressed(Keyboard::Down) && timer && Map[y + 1][x] != 'N')

{

y++;

timer = false;

}

}

void draw()

{

control();

miner.setPosition(x \* 60, y \* 60);

if (Map[y][x] == 'B')

{

Map[y][x] = ' ';

}

if (Map[y][x] == '0')

{

Map[y][x] = ' ';

score++;

}

}

};

Player Hero(7, 7, 500);

class Mob

{

public:

int x, y, speed;

bool timer=true ;

Texture MOB;

Sprite mob;

Clock TIME;

Mob(int x, int y, int speed)

{

this->x = x; this->y = y; this->speed = -speed;

MOB.loadFromFile("Mob.png");

mob.setTexture(MOB);

mob.setTextureRect(IntRect(0, 0, 60, 60));

mob.setPosition(x \* 60, y \* 60);

}

void move()

{

float time = TIME.getElapsedTime().asSeconds();

if (time >1)

{

timer = true;

TIME.restart();

}

if (Hero.x<x&&Map[y][x-1]==' '&&timer)

{

x--;

timer = false;

}

if (Hero.y<y&&Map[y-1][x] == ' '&&timer)

{

y--;

timer = false;

}

if (Hero.x>x&&Map[y][x+1] == ' '&&timer)

{

x++;

timer = false;

}

if (Hero.y>y&&Map[y+1][x] == ' '&&timer)

{

y++;

timer = false;

}

}

void draw()

{

move();

mob.setPosition(x \* 60, y \* 60);

}

};

int main()

{

Texture Block;

Block.loadFromFile("Kamen.png");

Texture FON;

FON.loadFromFile("FON.jpg");

Sprite fon;

fon.setTexture(FON);

Texture KRAI;

KRAI.loadFromFile("KRAI.jpg");

Texture GOLD;

GOLD.loadFromFile("Gold.png");

Font font;

font.loadFromFile("Q.ttf");

Text text;

RenderWindow window(VideoMode(1920, 1080), "Diggerц");

Mob Vrag(25, 7, 1);

Mob Vrag2(6, 2, 1);

Mob Vrag3(5, 15, 1);

bool onGame = true;

bool Life = true;

while (window.isOpen())

{

Event event;

while (window.pollEvent(event))

{

if (event.type == Event::Closed)

window.close();

}

while (Life)

{

Event event;

while (window.pollEvent(event))

{

if (event.type == Event::Closed)

window.close();

}

while (onGame)

{

Event event;

while (window.pollEvent(event))

{

if (event.type == Event::Closed)

window.close();

}

window.draw(fon);

for (int i = 0; i < H; i++)

for (int j = 0; j < W; j++)

{

if (Map[i][j] == 'B') {

Sprite block;

block.setTexture(Block);

block.setTextureRect(IntRect(0, 0, 60, 60));

block.setPosition(j \* 60, i \* 60);

window.draw(block);

}

if (Map[i][j] == '0') {

Sprite gold;

gold.setTexture(GOLD);

gold.setTextureRect(IntRect(0, 0, 60, 60));

gold.setPosition(j \* 60, i \* 60);

window.draw(gold);

}

if (Map[i][j] == 'N') {

Sprite krai;

krai.setTexture(KRAI);

krai.setTextureRect(IntRect(0, 0, 60, 60));

krai.setPosition(j \* 60, i \* 60);

window.draw(krai);

}

Hero.draw();

Vrag.draw(); Vrag2.draw(); Vrag3.draw();

if (Hero.x == Vrag3.x && Hero.y == Vrag3.y) {

onGame = false;

Life = false;

}

if (Hero.x == Vrag2.x && Hero.y == Vrag2.y) {

onGame = false;

Life = false;

}

if (Hero.x == Vrag.x && Hero.y == Vrag.y) {

onGame = false;

Life = false;

}

}

if (Hero.score == 21)

{

onGame = false;

}

window.draw(Vrag.mob); window.draw(Vrag2.mob); window.draw(Vrag3.mob);window.draw(Hero.miner);

std::ostringstream playerScoreString;

playerScoreString << Hero.score;

text.setFont(font);

text.setString("Score:" + playerScoreString.str());

text.setCharacterSize(25);

text.setFillColor(Color::Red);

text.setStyle(Text::Bold | Text::Underlined);

text.setPosition(20, 20);

window.draw(text);

window.display();

}

window.clear(Color::White);

text.setFont(font);

text.setString("You win");

text.setCharacterSize(150);

text.setFillColor(Color::Blue);

text.setStyle(Text::Bold | Text::Underlined);

text.setPosition(960, 460);

window.draw(text);

window.display();

}

window.clear(Color::White);

text.setFont(font);

text.setString("You lose");

text.setCharacterSize(150);

text.setFillColor(Color::Red);

text.setStyle(Text::Bold | Text::Underlined);

text.setPosition(960, 460);

window.draw(text);

window.display();

}

return 0;

}

**Литература**

1. [**https://kychka-pc.ru/sfml/urok-1-podklyuchenie-biblioteki-k-srede-razrabotki-visual-studio-2013.html**](https://kychka-pc.ru/sfml/urok-1-podklyuchenie-biblioteki-k-srede-razrabotki-visual-studio-2013.html)
2. [**https://www.youtube.com/playlist?list=PLQOaTSbfxUtCrKs0nicOg2npJQYSPGO9r**](https://www.youtube.com/playlist?list=PLQOaTSbfxUtCrKs0nicOg2npJQYSPGO9r)
3. [**https://ru.wikipedia.org/wiki/SFML**](https://ru.wikipedia.org/wiki/SFML)
4. [**https://ru.wikipedia.org/wiki/Microsoft\_Visual\_Studio**](https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio)
5. [**https://ru.wikipedia.org/wiki/C%2B%2B**](https://ru.wikipedia.org/wiki/C%2B%2B)