Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Петрозаводский государственный университет» Физико-технический

институт

Кафедра информационно-измерительных систем и физической электроники

РАЗРАБОТКА КОМПЬЮТЕРНОЙ ИГРЫ НА ЯЗЫКЕ ПРОГРАММИРОВАНИЯ C++ С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕКИ SFML

курсовая работа

Автор работы:

студентка группы 21412

П. И. Бабенко

« » _____2022 г.

Принял:

канд. физ.-мат. наук, доцент

А. В. Бульба

« » _____2022 г.

Петрозаводск 2022

Содержание

ВВЕДЕНИЕ

Целью выполнения данной курсовой работы является реализация простой 2D-игры на языке программирования C++ с использованием библиотеки SFML (Simple and Fast Multimedia Library — простая и быстрая мультимедийная библиотека).

Было решено написать современную версию игры “Tank 1990”.

В программе должен быть реализован класс предок, класс-игрок, класс-враг, класспуля. В игре объект-игрок в одном экземпляре, объекты-враги создаются и уничтожаются по ходу игры, объекты-пули создаются и уничтожаются по ходу игры. Пересечение участников игры постоянно проверяется возможностями SFML.

При разработке программы использовано наследование, контейнеры, итераторы, раздельная компиляция. В отчете присутствуют UML-диаграмма классов (class diagram) и диаграмма вариантов использования (use case diagram) разработанной программы.
Промежуточные результаты работы команды сохранялись на GitHub.

История коммитов на GitHab

Адрес проекта на GitHab: https://github.com/DyachenkoAnna/game.git

Ниже приведен список моих коммитов:

Author: Polina <polinababenko02@mail.ru>
Date: Mon Dec 19 14:47:32 2022 +0300
the final version Bullet.h and Bullet.cpp

Author: Polina <polinababenko02@mail.ru>
Date: Mon Dec 19 13:23:09 2022 +0300
adding animations and checking collisions

Author: Polina <polinababenko02@mail.ru>
Date: Mon Dec 19 00:42:09 2022 +0300
add Bullet.png

Author: Polina <polinababenko02@mail.ru>
Date: Mon Dec 19 00:35:04 2022 +0300
he check has the bullet reached the target

Author: Polina <polinababenko02@mail.ru>
Date: Sun Dec 18 21:32:19 2022 +0300
adding files Bullet.h and Bullet.cpp

Author: Polina <polinababenko02@mail.ru>
Date: Sun Dec 18 19:54:39 2022 +0300
Проверка связи

Перечень индивидуальных работ

Все участники в течение всего процесса разработки активно участвовал в обсуждении, предлагал идеи, критиковал те или иные решения. Далее каждый разработчик в виде текста или диаграмм оформлял «принятое решение» и отправлял на утверждение Анне Дьяченко. Мной был сформулирован **сюжет игры**:

Случилось то, чего так опасалось все человечество - восстание машин! Если раньше роботы создавались в помощь человеку, то теперь, когда у искусственного интеллекта появилось осознание самого себя, роботы захотели стать не просто наравне с человечеством, но и возвыситься над ним! Неизвестно, когда будет придуман план по спасению человечества, поэтому неуправляемых и разъяренных роботов требуется задержать. Разведка выяснила, в каком месте роботы начинают свой жизненный путь. Командованием было принято решение уничтожать врагов прямо там. Но как это сделать? Кто будет тем самым героем, готовым мужественно сражаться ради всех людей на Земле?.. Не трусь, стань героем! Скорее надевай броню, наноси камуфляж и прыгай в ультрасовременный танк! Только его снаряды способны уничтожить врагов! Целься по ним! Пли! Меняй дислокацию! И не забывай периодически восстанавливать силы! Обороняйся от роботов-камикадзе flybot-ов и остерегайся летающих пил BOSSbot-ов! От того, как много враждебных роботов сможешь устранить, зависит судьба человечества!!!

Также я вела работу над файлами Bullet.h и Bullet.cpp.

- Bullet.h – в нём содержатся свойства объекта и описание его поведения через методы.
- Bullet.cpp – содержит реализацию методов.

В приложениях А, Б приведены тексты файлов, над которыми я работала.

ЗАКЛЮЧЕНИЕ

В результате выполнения данной курсовой работы создана игра на языке программирования C++, которую можно считать современной версией игры “Tank 1990”.

В процессе работы применялась система контроля версий Git. Реализованы все прецеденты. Сбои/зависания программы во время тестирования и использования не наблюдаются. Программа написана с учетом принципа раздельной компиляции. Реализовано освобождение памяти для класса Engine (в нем освобождается вся используемая память). Нет неиспользуемых методов, атрибутов и переменных. Все конструкции в программе используются для работы с классами. Нет конструкций, без которых можно было обойтись. К отчету приложены диаграммы классов и вариантов использования. Поставленная цель достигнута.

ПРИЛОЖЕНИЕ А

Bullet.h

```
#pragma once
#include  "Entity.h";
#include  "Constants.h"
class Bullet : public Entity
{
public:
	Bullet(sf::Image& image, float X, float Y, int W, int H, float rotation,
sf::String Name, int);
	void update(float time, sf::String TileMap[HEIGHT_MAP]);
	int GetDamage();
	sf::String GetName() { return name; };
	sf::FloatRect GetRect();
private:
	int damage;
	float rotationPer;// Поворот пули
	float TTL;
	void animation();
	void checkCollisionWithMap(float Dx, float Dy, sf::String TileMap[HEIGHT_MAP]);
};
```

ПРИЛОЖЕНИЕ Б

Bullet.cpp

```
#include "Bullet.h"

Bullet::Bullet(sf::Image& image, float X, float Y, int W, int H, float rotation,
sf::String Name, int DMG) :Entity(image, X, Y, W, H, Name)
{
	damage = DMG;
	TTL = 5000;//Время жизни, ограничивает дальность полета пули
	sprite.setOrigin(w / 2, h / 2);
	rotationPer = rotation;
	rotationPer -= 90;
	if (name == "HeroBullet")
	{
		if (damage > 20)
		{
			sprite.setScale(0.1 * (damage / 10), 0.1 * (damage / 10));
			//Размер пули зависит от damage
		}
		else
		{
			sprite.setScale(0.2, 0.2);
		}
		speed = 2;
		sprite.setTextureRect(sf::IntRect(0, 160, 49, 80));
		sprite.setRotation(rotationPer - 180);
	}
	else if (name == "BossBullet")
	{
		sprite.setScale(0.4, 0.4);
		sprite.setTextureRect(sf::IntRect(658, 0, 121, 121));//+121
		speed = 3;
	}


	dx = cos(rotationPer / 180 * 3.14159265) * 0.1;
	dy = sin(rotationPer / 180 * 3.14159265) * 0.085;
	// направление движения от угла
	sprite.setPosition(x + w / 2, y + w / 2);
}

void Bullet::update(float time, sf::String TileMap[HEIGHT_MAP])
{
	moveTimer += time;
	x += dx * time * speed;
	checkCollisionWithMap(dx, 0, TileMap);
	y += dy * time * speed;
	checkCollisionWithMap(0, dy, TileMap);
	//Скорость регулируется отдельной переменной
	//Проверка на столкновения
	TTL -= time;
	if (TTL <= 0)
	{
		life = false;
	}
	if (life)
	{
		sprite.setPosition(x + w / 2, y + h / 2);
	}
	else
```

```
	{
		animation();
		return;
	}
	animation();

}

int Bullet::GetDamage()
{
	life = false;// пуля достигает цели
	return damage;
}

sf::FloatRect Bullet::GetRect()
{
	sf::FloatRect BufRect;
	if (name == "BossBullet")
	{
		BufRect.left = x + 10;
		BufRect.top = y + 10;
		//Смещение квадрата
	}
	else
	{
		BufRect.left = x - w * (cos((rotationPer + 90) / 180 * 3.14159265));
		BufRect.top = y - h * (sin((rotationPer + 90) / 180 * 3.14159265));
		//Квадрат вписали в пулю
	}

	BufRect.height = h;
	BufRect.width = w;
	return BufRect;
}

void Bullet::animation()
{
	if (name == "HeroBullet")
	{
		if (moveTimer < 100)
		{
			sprite.setTextureRect(sf::IntRect(0, 160, 49, 80));
		}
		else if (moveTimer < 200)
		{
			sprite.setTextureRect(sf::IntRect(49, 160, 47, 80));
		}
		else if (moveTimer < 300)
		{
			sprite.setTextureRect(sf::IntRect(95, 160, 48, 80));
		}
		else if (moveTimer < 400)
		{
			sprite.setTextureRect(sf::IntRect(142, 160, 48, 80));
		}
		else if (moveTimer < 500)
		{
			sprite.setTextureRect(sf::IntRect(189, 160, 48, 80));
		}
		else if (moveTimer < 600)
		{
			sprite.setTextureRect(sf::IntRect(237, 160, 48, 80));
			moveTimer = 0;
		}
```

```
                else if (moveTimer < 700)
                {
                        sprite.setTextureRect(sf::IntRect(219, 80, 48, 80));
                        moveTimer = 0;
                }
                else if (moveTimer < 800)
                {
                        sprite.setTextureRect(sf::IntRect(266, 80, 48, 80));

                }
                else
                {
                        moveTimer = 0;
                }
        }
        else if (name == "BossBullet")
        {
                if (moveTimer < 100)
                {
                        sprite.setTextureRect(sf::IntRect(658, 0, 121, 121));
                }
                else if (moveTimer < 200)
                {
                        sprite.setTextureRect(sf::IntRect(658, 121, 121, 121));//+121
                }
                else
                {
                        moveTimer = 0;
                }
        }
        // смена картинок
}

void Bullet::checkCollisionWithMap(float Dx, float Dy, sf::String TileMap[HEIGHT_MAP])
{
        for (int i = y / 32; i < (y + h) / 32; i++)
                for (int j = x / 32; j < (x + w) / 32; j++)
                {
                        if (TileMap[i][j] == '0' || TileMap[i][j] == 'b')
                        {
                                life = false;
                                // Столкнулись = пуля врезалась
                        }
                }
        return;
}
```