|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |
| Институт кибернетики | | |
| Кафедра программного обеспечения систем радиоэлектронной аппаратуры | | |

Курсовой проект по дисциплине

«Методы и стандарты программирования»

на тему: Клон игры Worms

|  |  |  |  |
| --- | --- | --- | --- |
| Обучающийся | \_\_\_\_\_\_\_\_\_\_\_\_\_ | Тришин Никита Андреевич*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* | |
|  | *Подпись* | *Фамилия Имя Отчество* | |
| Шифр | 19К0219 |  |  |
| Группа | КМБО-02-19 |  |  |
|  |  |  |  |
| Руководитель  работы | \_\_\_\_\_\_\_\_\_\_\_\_\_ | *\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_* | |
|  | *Подпись* | *Фамилия Имя Отчество* | |

Москва 2019

**Оглавление**

[Введение 2](#_Toc59249450)

[Описание задачи и её решения 2](#_Toc59249451)

[Руководство пользователя 2](#_Toc59249452)

[Запуск 2](#_Toc59249453)

[Управление 3](#_Toc59249454)

[Механика игры 4](#_Toc59249455)

[Документация 4](#_Toc59249456)

[Сборка программы 4](#_Toc59249457)

[Архитектура программы 4](#_Toc59249458)

[Главное меню 4](#_Toc59249459)

[Игра 5](#_Toc59249460)

[Заключение 6](#_Toc59249461)

# 

# **Введение**

Темой этой курсовой работы является создание клона игры “Worms”. Для осуществления проекта необходимо было проанализировать механику игры и выбрать инструменты для создания.

В данном отчете описан процесс создания игры, руководство пользователя и документация кода игры.

# **Описание задачи и её решения**

Основная концепция игры довольно проста – имеется определенная карта (по сути, графический файл в формате png), на которой находятся определенные юниты.

Каждый юнит принадлежит к определенной команде. Всего юнитов по 4 в каждой команде. Количество команд пользователь может выбирать в главном меню.

Когда игра начинается, игрок может управлять одним юнитом. Его задача – поразить ракетой вражеских юнитов (и желательно не задеть своих). После того, как игрок выстрелил (или упал за пределы карты) все юниты, попавшие в радиус взрыва, получают урон, а земля разрушается. Затем ход передается следующему игроку, и так пока на карте не останутся только юниты одной команды.

Игра была написана на языке C++ 20-ого стандарта с использованием библиотеки SFML версии 2.5.1. Код был написан в IDE Clion.

# **Руководство пользователя**

## **Запуск**

Чтобы запустить игру, нужно открыть исполняемый файл(exe) под именем “Penguin Warfare” в папке игры.

## **Управление**

После запуска игры откроется главное меню. Главное меню состоит из 3-х пунктов: игра, настройки, выход. Перемещаться по главному меню можно с помощью клавиш “W” и “S”, “Enter” – выбрать пункт меню, “Esc” - вернуться в главное меню.

Если пользовать выберет пункт “Выход” – игра закроется.

Если пользователь откроет настройки – ему будет предложено выбрать режим экрана – либо полноэкранный, либо в окне. Чтобы поменять режим, нужно нажать “Enter”.

Если пользователь выберет пункт “Игра” – ему откроется меню, в котором он может:

* Начать игру (нажать Enter)
* Поменять количество команд (“A” чтобы уменьшить, “D” чтобы увеличить)
* Поменять карту (Пока в разработке)
* Поменять количество оружия (Пока в разработке)

Когда пользователь начнет игру, он увидит в центре экрана юнита, которым он в данный момент управляет. Пользователь может:

* Ходить (“A” влево, “D” вправо)
* Прыгать (“Q” прыжок влево, “E” прыжок вправо. Так же стоит заметить, что прыжок “вперед” и “спиной” отличаются)
* Целиться (“W” прицел вверх, “S” прицел вниз)
* Стрелять (Удерживать “Пробел”. Чем дольше пользователь удерживает кнопку, тем сильнее выстрелит оружие юнита)
* Вернуться в главное меню (“Escape”)
* Осмотреться (“Enter”. В режиме осмотра “A” влево, “D” вправо, “Enter” вернутся)

## **Механика игры**

Игрок управляет одним из своих юнитов. Его цель – поразить юнитов из другой команды.

В начале игры у каждого юнита 100 очков здоровья. Если юнит оказался в радиусе взрыва, то эти очки здоровья уменьшатся. Если очки здоровья у юнита упали до 0 или ниже, то юнит погибает. Также юнит погибнет, если вылетел с карты.

Если остались в живых лишь юниты одной команды, то игра заканчивается, и эта команда побеждает.

# **Документация**

## **Сборка программы**

Перед сборкой у вас должны быть установлены следующие программы:

* Cmake
* Visual Studio 15 2017 года и новее.

Если у вас имеются эти программы, то вам необходимо прописать команду cmake и указать путь до папки с исходными файлами.

После завершения сборки вы можете запустить программу, запустив исполняемый файл Penguin Warfare.

## **Архитектура программы**

### **Главное меню**

Программа начинает работу из файла main.cpp. В этом файле и находится главный цикл. Работает все по следующему принципу: у нас имеется переменная currentClass, с помощью которой мы понимаем, какой класс у нас запущен в текущий момент.

Сначала, currentClass равен значению menu, то есть сначала у нас открыто меню. Значит, мы работаем с объектом класса Menu, который мы предварительно создали. Если нажал Play, то currentClass становится равен Game, где мы удаляем объект класса меню, и создаем объект класса GameWorld. Именно тут и начинается игра.

### **Игра**

Класс GameWorld довольно комплексный и требует более детального обзора. В данном классе содержатся следующие классы:

* Terrain – это земля, по которой двигаются юниты. В процессе игры земля будет деформироваться (вся земля будет уничтожаться в определенном радиусе вокруг попадания снаряда). Сама же земля представляет собой png файл. Принцип работы довольно прост. Если пиксель в этом изображении прозрачен – то это воздух, если нет, то это земля. Нетрудно догадаться, что, когда игрок разрушает землю, по факту программа просто перекрашивает пиксели в прозрачные, и таким образом появляется разрушение.
* Player – Вектор указателей на игроков. По сути, это и есть наши команды. Количество элементов этого вектора зависит от того, сколько команд выбрал пользователь в меню. В каждом “игроке” хранятся юниты и оружие, которые принадлежат команде игрока.
  + Unit – это и есть пингвины, которые ходят, стреляют и получают урон. Так же в каждом юните есть объект анимации, который отвечает непосредственно за анимацию юнитов.
  + Weapon – набор параметров оружия (радиус взрыва, урон), масса снаряда, текстуры.
* Projectile – снаряд, который выстреливают юниты. При столкновении с землей “взрывается”
* Crosshair – это прицел, с помощью которого игрок задает направление выстрела.

Заметим, что классы Unit и Projectile унаследованы от класса Entity. Класс Entity отвечает за физические особенности объекта. Его координаты, импульс, и притяжение к земле.

Сама же физика была реализована следующим образом: у каждого объекта, унаследованного от Entity, имеются определенные “точки коллизии”. Если пиксель текстуры в некой точке коллизии непрозрачен, то происходит коллизия. В случае юнитов это значит, что игра перестает притягивать его к земле. В случае снаряда это значит, что снаряду пора взрываться. Также, если физический объект(юнит или снаряд) окажутся вне карты, то снаряд исчезнет, и игра пойдет дальше, а юнит перестанет отрисовываться и будет считаться мертвым.

Если одна из команд победит (или пользователь нажмет “Escape”), то объект класса GameWorld будет удален, объект класса Menu будет создан, и в игровом окне пользователь снова увидит главное меню.

# **Заключение**

В этой работе удалось выполнить главную задачу – сделать клон игры “Worms”. Во время работы удалось в определенной степени изучить SFML, а также получить огромный опыт в сфере разработки игр