**Введение**

Развлечения всегда были неотъемлемой частью жизни человека. По этой причине в данной работе я хотел бы разобрать одно из самых передовых и быстро развивающихся технологий в области развлечений – «лазертаг», а именно программное обеспечение для клиентской части.

**Анализ проблемы**

В конце 70-х годов была представлена одна из первых версий взаимодействия устройств по ИК сигналу. Для обмена данными был придуман устоявшийся протокол передачи – «Miles tag», который дошел до современного вида лазертага. По этому протоколу построены все правила передачи информации между устройствами. Устройствами являются муляжи оружий, повязки на голову и дополнительные устройства для разнообразия сценариев игры. Муляж оружия – «тагер», умеет излучать ИК сигнал для имитации выстрела. Повязка на голову или жилет имеют несколько модулей для приёма ИК сигнала, которые оснащены световыми индикаторами. Радиобаза – устройство для передачи информацией между игроками и компьютером со специальным программным обеспечением. Программное обеспечение является способом взаимодействия клиента с оборудованием для настройки игровых параметров и введения статистики игры.

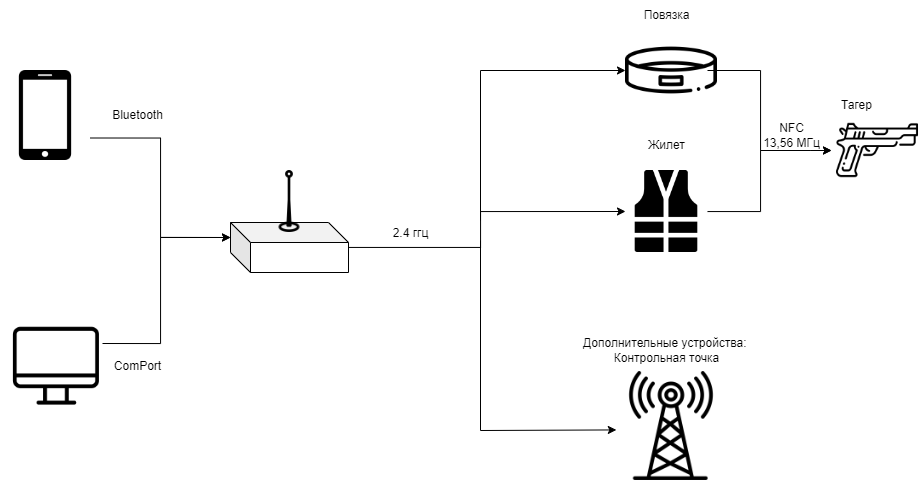


Рисунок 1 – Абстрактное представление работы оборудования.

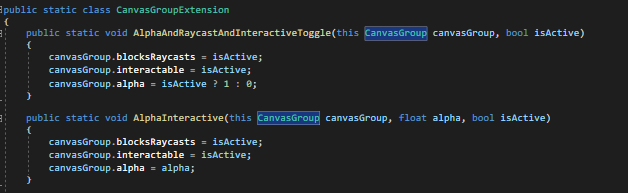
Целью данной работы является разработка вышеупомянутого программного обеспечения, для настройки всех типов устройств специального лазертаг оборудования, учёта статистики и автоматизации игровых событий. Существует множество способов подключения радиобазы для дальнейшей передачи информации, но нужно реализовать 2 – соединение с радиобазой по COMPORT и Bluetooth, для компьютеров и телефонов соответственно.

**3 Разработка**

В данной главе рассмотрена разработка всех классов и методов для построения основного функционала приложения.

**3.1 Создание проекта и сцены под Desktop**

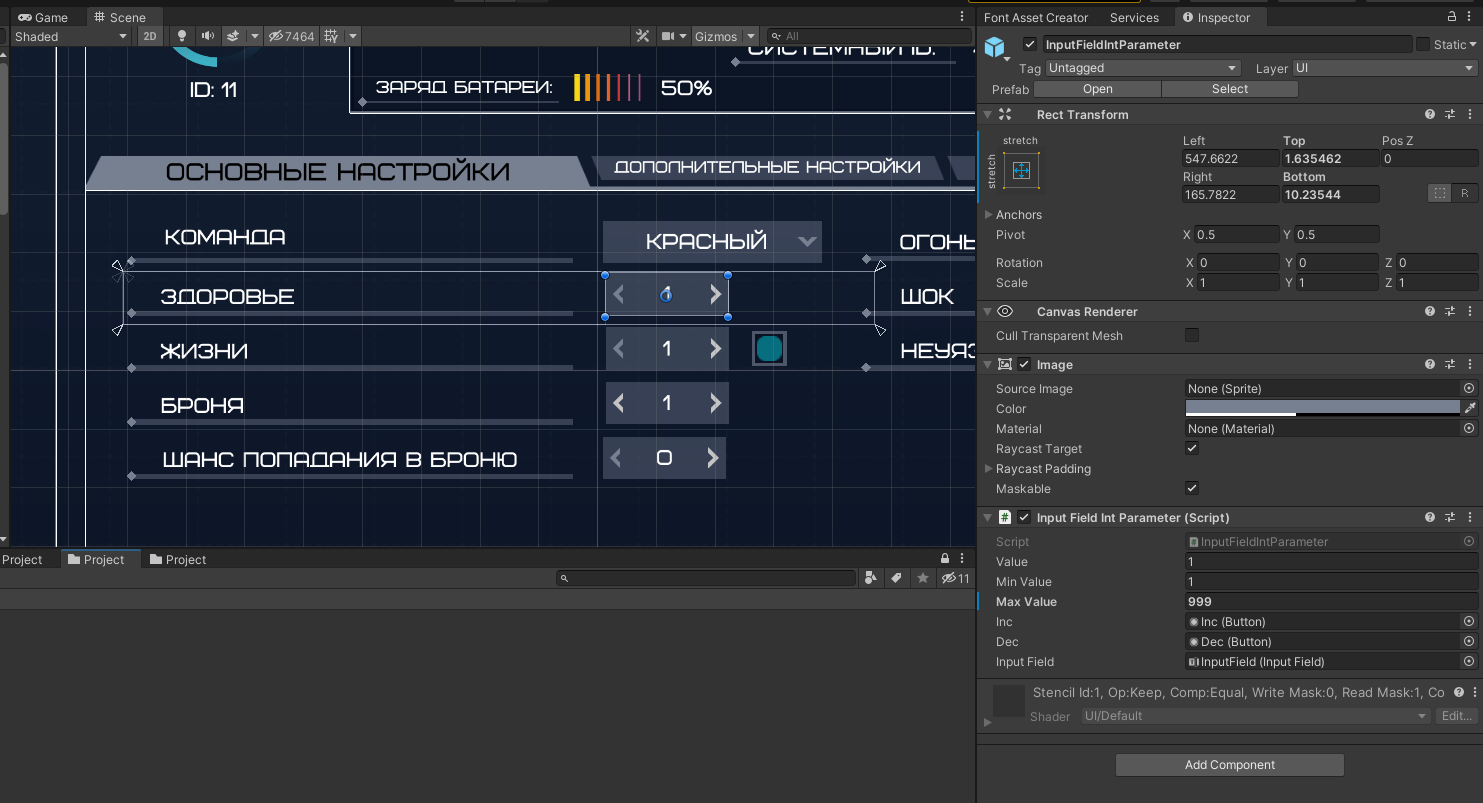
Меню конфигуратора включает в себя множество параметров, которые были утверждены и расставлены в дизайне приложения. Под все основные меню выделены объекты, включающие в себя компонент «CanvasGroup». Данное решение было принято за счёт удобства взаимодействия с этим компонентом. Встроенный класс был расширен двумя методами, отвечающими за прозрачность, с помощью которых реализовано открытие и закрытие основных меню.

  
Рисунок 0 – Методы расширения CanvasGroup

Все элементы конфигуратора были и разделены на «настройки повязки», «настройки тагера», а они, разбиты на «основные настройки» и «дополнительные настройки», согласно дизайну.

  
Рисунок 0 – Пример расположения элементов конфигуратора на сцене

Каждый параметр – объект с наследником класса «MultyTypeIntParameter», который содержит в себе встроенный InputField, нижний и верхний порог значения и событие изменения параметра. Каждый наследник переопределяет метод разбора внесенного пользователем значения. Например, в поле ввода «здоровье», пользователь может занести только целое число не меньше 1 и не больше 999.

  
Рисунок 0 – Параметр конфигуратора

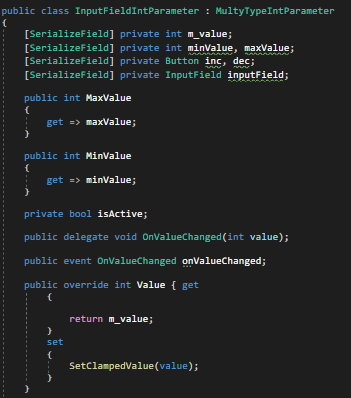
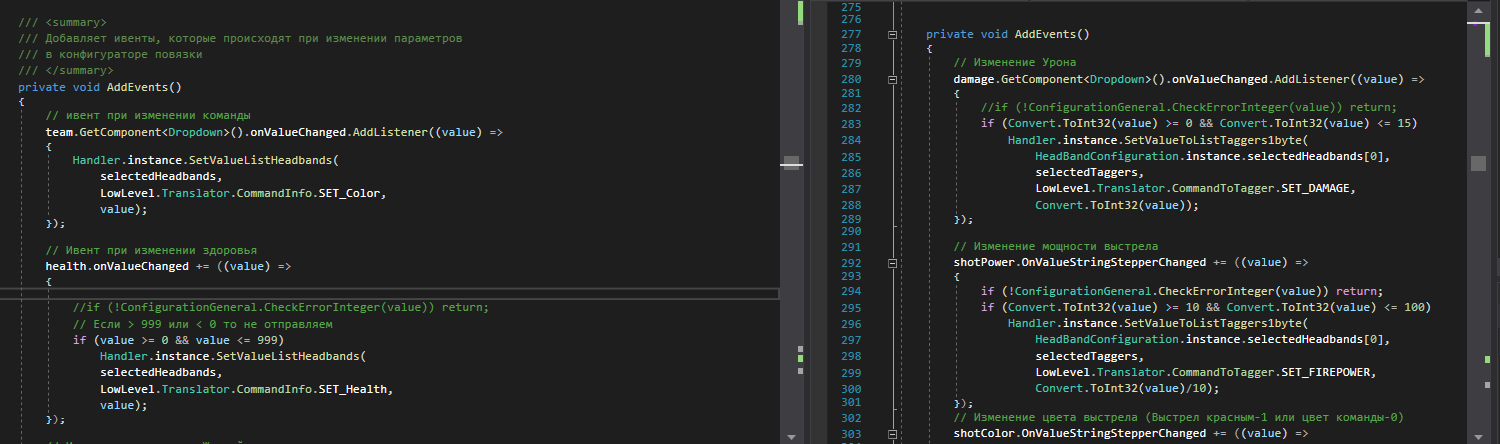


Рисунок 0 – класс наследник MultyTypeIntParameter

**3.2 Разработка классов конфигуратора**

Классы конфигуратора, как и UI, разделены на «конфигуратор повязки» и «конфигуратор тагера» - «HeadBandConfiguration», «TaggerConfiguration» соответственно. Каждый содержит в себе соответствующие параметры, методы инициализации девайса и добавление слушателей на события изменения параметров.

  
Рисунок 0 – Обработка событий изменения параметров

Каждое изменение параметра обращается к классу одиночке Handler, через который передаются все команды на COMPORT. Он содержит в себе объект типа ICommander, через который идёт основной обмен данными.