**Создание новой игры:**

* После накопления 8 игроков очередь автоматически создает новую комнату с интерфейсом RoyalRoom и передает ей свойства объекта виде QueueGamer (в базовом виде ссылка на ClientServer ник и пароль для возвращения в аккаунт)
* Передает созданную комнату в хранилище активных комнат Rooms
* Запускает комнату вызовом метода StartRoom

**Конструктор IRoom**

* Создает объект IRoomLogic
* Передаем объект IRoomLogic в качестве параметра конструктора INetwork

Класс RoomNetwork – управляет отправкой данных клиенту, содержит в себе список клиентов

**Движение игрока:**

* Начинается с получения сообщения GoToMsg (содержит в себе направление движения)
* При получении GoToMsg оно трансформируется в StartMoveGamer содержащее направление движения и добавляется в очередь сообщений объекта игрока
* После истечения кванта времени оно передается компоненту Movement
* Обработчик этого сообщения в Movement добавляет направление переданное в StartMoveGamer в currentDirection, компоненте переходим в состоянии active
* После перехода в активное состоянии, компонент на каждое сообщение TimeQuantPassed передвигает игрока на некоторое расстояние согласно текущему выбранному направлению.
* При получении сообщения EndMoveGamer компонент Movement переводит active в false(движение игрока прекращается)

**Обработка столкновений с предметами в процесс движения:**

* Получаем сообщение CollisionObjects
* Отодвигаем игрока на необходимое расстояние, чтобы не нарушать границы другого объекта

Каждый 2 ход (10 раз в секунду) в очередь сообщения об произошедших событиях в объекте RoyalGameModel должно добавляться сообщение об изменении координат игрока (сообщение содержит ссылку на объект игрока и точку в которую переместился игрок), которое затем конвертируется (при его обработке в NetworkRoom) в сообщение об изменении координат игрока (содержит координаты и идентификатор игрока (уникален только в рамках игровой комнаты)) – рассылается всем игрокам, попадающее в область видимости этого события

**Механизм работы срочных сообщений**

* В модели хранится коллекция HappenedEvents, в которую добавляются объекты событий, произошедших в игре. Объекты событий создаются в компонентах игровых объектов.
* Так как коллекция ObservalableCollection, то на нее можно подписаться, чтобы узнать о том, что она изменилась. На нее подписывается метод класса RoomNetwork
* При получении такого сообщения INetwork определяет, игроков, которые оказались в границах видимости данного события и уведомляет их об этом событии (конвертируя в нужный тип IMessage)

**Инвентарь. Поднятие предметов**

* Представлен компонентом Collector
* Содержит в себе две коллекции Модификаторы и Оружие
* При получении сообщения PickUpLootMsg мы обращаемся к клеткам карты на которых находится объект (для этого нам нужна ссылка на SolidBody этого объекта) – идем по списку объектов на всех клетках, и определяем:
  + - Является ли данный объект поднимаемым
    - Входит ли игрок в область поднятия игрового предмета
* Мы добавляем его в инвентарь и удаляем с карты. Отправляем этому объекту сообщение о том, что он подобран (обрабатывать его не обязательно)

Реализация проверки на возможность подбора игроком осуществляется с помощью метода CheckCollision

**Оружие**

Содержит следующие компоненты:

* SolidBody – прозрачный
* Shot для создания выстрела(добавляет пулю в коллекцию игровых объектов) содержит в себе компонент Magazin
* Magazin – выполняет все действия происходящие с магазином

**Реализация создания игровой комнаты:**

* Справочная информация по работе комнаты: <https://habr.com/ru/post/147431/>
* Создать объект RoyalRoom и передать ему список подключившихся игроков
* Удалить ссылки на игроков в очереди
* Добавить объект в Rooms

**При создании RoyalRoom должны выполняться следующие действия:**

* Создание объекта RoyalRoomLogic
* Создание RoomNetwork

**При создании RoyalRoomLogic должны выполняться следующие действия:**

* Создание контекста данных RoyalGameModel
* Создание таймера запускающегося каждые 50 милисекунд и вызывающего метод TickQuantTimer
* Cоздание класса для хранения времени прошедшего с предыдущего запуска TickQuantTimer
* Реализовать метод TickQuantTimer

**Создание контекста данных RoyalGameModel:**

* Создаем массив клеток размером lengthOfSide. В каждую клетку помещаем ссылку на CellField
* Запускаем метод CreateGameObject (необходимо реализовать по умолчанию) – создаем объекты и добавляет их в список объектов и на игровую карту в соответствии с координатами
* Создаем метод CreatePlayers() – добавляет игроков в спиcки Players, GameObjects и на игровую карту

**При создании RoomNetwork должны выполняться следующие действия**:

* Сохранение ссылки на игровую логику
* Запуск метода CreateClients – создает INetworkClient и записывает в него ссылку на игрока из RoyalRoomLogic
* Создание таймер, который срабатывает каждые 3 сек и вызывает метод HandlerTotalSinch

**Работа механизма столкновений физических тел:**

* При движении игрока или другого физического предмета(пули), вызывается метод AppendCoords компонента SolidBody, в данном методе:
  + - Меняются координаты SolidBody
    - Меняются координаты объекта на карте
    - Запускается метод определения возможности столкновения: Определяются все тела в области данного объекта. Если на области тела находят другие тела, то происходит столкновение. В случае столкновения тело сдвигается в противоположном перемещению направлении, на расстояние достаточное для выхода из области наложения тел предметов.
    - Все столкнувшимся объектам рассылаются сообщения об этом

На карте располагаются не сами объекты, а их компоненты SolidBody, которые наследуют интерфес IFieldObject