**Форма Autorization:**

Работа клиента начинается с запуска формы Autorization, которая имеет 2 поля логин и пароль и кнопку войти, при изменении которых форма уведомляет свой контроллер AutorizationController, а также ссылку на контроллер AutorizationController.

AutorizationController содержит в себе простейшую логику проверки на валидность (проверка существования логина или пароля). Содержит в себе ссылку на вид(форму Autorization).

Содержит в модель AutorizationModel содержит в себе свойства для хранения пароля, логина и состояния (правильный пароль или нет) (смотри схему работы механизма авторитизации), а также событие уведомляющее, что модель изменилась.

При первом нажатии кнопки войти контроллер создает BaseClient библиотеки CSInteraction и подписывается на его события, после чего создает запрос на вход в аккаунт AuthorizationMessage.

Если получено сообщение ErrorAuthorization, то меняет состояние у модели на ошибка авторитизации, иначе (SuccessfulAuthorization) переходим на форму Account.

При повторной попытке войти в аккаунт используется уже созданный BaseClient.

Для подключения к серверу контроллер использует данные заданные в форме Настройки и сохранные в файле(пока не будет реализовано)

**Форма Account:**

Получает в качестве аргументов объект BaseClient, логин и пароль игрока, которая передает их в контроллер AccountController, который добавляет подписывается на события BaseClient и сохраяент логин и пароль в AccountModel.

AccountController содержит в себе AccountModel, которая также хранит количество килов, смертей, битв, игрового времени.

После загрузки формы отправляет сообщение серверу, что форма загружена

Получает данные от сервера и на основе их инициализирует модель

При нажатии на кнопку в бой создает форму Queue

**Форма Queue:**

Получает в качестве аргумента BaseClient. Account закрывается, однако при обычном выходе будет выполнен переход в Account с повторным запросов данных аккаунта. Форма содержит надпись о количестве игроков в сети и кнопку «выйти из очереди». QueueController содержит в себе QueueModel содержащая в себе количество игроков онлайн на данный момент. Переход к форме осуществляется после получения сообщения AddInBattle

**Форма Battle:**

Имеет 2 контроллера UserActionController – обрабатывает действия пользователя для их отправки серверу и GameActionController - обрабатывает сообщения пришедшие с сервера только GameActionСontroller имеет прямой доступ к игровой модели.

BattleModel хранит объекты игры – список объектов. На карте отрисовываются только те объекты, которые входят в экран игрока. В списке хранятся не все объекты – а только те которые могут перемещаться. Отрисовка статических объектов производится только один раз при создании карты.

Карта представляет собой Bitmap. Который обрезается до нужных размеров, чтобы отрисоваться на экране