Пакет romanow.domino.client

|  |  |
| --- | --- |
| Класс | Описание |
| GameController | Класс игрового контроллера клиентской части.  Реализует интерфейсы ViewEvents и NetworkEvents  **Поля класса**:  *private NetworkCommands* **–** поле, хранящее ссылку на сетевые команды  *private ViewCommands –* поле, хранящее ссылку на команды user-интерфейса  *private String name –* имя текущего игрока  *private boolean isMarketEmpty –* поле, определяющее пуст ли магазин  *private boolean isMyTurn –* поле, определяющее, что сейчас ход текущего игрока  *private boolean isAuthorize –* поле, определяющее, что текущий пользователь авторизован в игре  *private Model model –* поле, хранящее ссылку на клиентскую модель игры  **Конструктор:** *public GameController(NetworkCommands, ViewCommands, Model)*.  **Методы:**  *getIsAuthorize(), setIsAuthorize(boolean), getName(), setName(String), getModel(), setModel() –* геттеры и сеттеры для полей  *void playGame() – метод для запуска игры*  *void waitForAuthorization() – метод, для ожидания авторизации текущего пользователя.*  ***Реализация интерфейса NetworkEvents:***  *void onDominoPlaced(String name, Domino tile, boolean isLeft) –* метод для обработки события с сервера о том, что было поставлено домино (tile) на доску игроком (name) слева (isLeft – true) или справа (isLeft – false)  *void onNextTurn(String name) –* метод для обработки сообщения с сервера о том, чей сейчас ход (name)  *void onDrawFromMarket(String name, Domino tile) –* метод для обработки сообщения с сервера о том, что кто-то (name) взял домино с магазина (tile)  *void onTextMessage(String message) –* метод, для обработки запроса с сервера об отправленном сообщении (message)  *void onGameStart(HashMap<String, ArrayList<Domino>> hands) –* обработка запроса с сервера о начале игры  *void onGameAbort(String message) –* конец игры  *void onAuthorizationStatusReceived(String name, boolean isAuthorized, String message) –* подтверждения статуса авторизации  *void onConnectionEstablished() –* обработка запроса с сервера об установленном соединении  *void onConnectionFailed() –* обработка запроса с сервера об ошибке соединения  *void onConnectionLost() –* обработка запроса с сервера об утерянном соединении  **Реализация интерфейса *ViewEvents*:**  *void onAbortGame() –* попытка текущим клиентом завершить игру  *void onTryDrawDominoFromMarket() –* взятие домино из магазина  void onTryPass() – попытка сделать pass  *void onGameStartRequest() –* запрос начала игры  *void onTryPlaceDomino(Domino tile, boolean isLeft) –* попытка положить домино (tile) слева (isLeft = true) или справа (isLeft = false)  *void onTryConnect(String ip) –* подключение к серверу  *void onTryAuthorize(String name) –* попытка авторизации в игре |
| NetworkEvents | Интерфейс для событий сети (от сервера)  *void onDominoPlaced(String name, Domino tile, boolean isLeft) –* метод для обработки события с сервера о том, что было поставлено домино (tile) на доску игроком (name) слева (isLeft – true) или справа (isLeft – false)  *void onNextTurn(String name) –* метод для обработки сообщения с сервера о том, чей сейчас ход (name)  *void onDrawFromMarket(String name, Domino tile) –* метод для обработки сообщения с сервера о том, что кто-то (name) взял домино с магазина (tile)  *void onTextMessage(String message) –* метод, для обработки запроса с сервера об отправленном сообщении (message)  *void onGameStart(HashMap<String, ArrayList<Domino>> hands) –* обработка запроса с сервера о начале игры  *void onGameAbort(String message) –* конец игры  *void onAuthorizationStatusReceived(String name, boolean isAuthorized, String message) –* подтверждения статуса авторизации  *void onConnectionEstablished() –* обработка запроса с сервера об установленном соединении  *void onConnectionFailed() –* обработка запроса с сервера об ошибке соединения  *void onConnectionLost() –* обработка запроса с сервера об утерянном соединении |

Пакет romanow.domino.server;

|  |  |
| --- | --- |
| RemoteUser | Класс на севере, отвечающий за удалённого игрока  Наследуется от *BaseUser*, реализует интерфейс *NetworkEvents*.  **Поля класса:**  *private NetworkCommands networkCommands* – поле, хранящее ссылку на сетевые команды  *private String name* – имя игрока  **Конструктор:** *RemoteUser(GameCommands, NetworkCommands, String name, Model)*  **Методы:**  **Реализация абстрактных методов BaseUser:**  *void onPlaceDomino(BaseUser user, Domino tile, boolean isLeft)* – постав – кто-то (user) поставил домино (tile) слева (isLeft = true) или справа (isLeft=false)  *void onNextTurn(BaseUser user)* – метод, который сообщает пользователю чей (user) сейчас ход  *void onDrawFromMarket(BaseUser who, Domino tile*) – метод, сообщающий пользователю кто (who) взял домино (tile) из магазина  *void onTextMessage(BaseUser user, String message)* – метод, сообщающий пользователю, что пользователь (user) написал сообщение (message)  *void onGameStart(HashMap<String, ArrayList<Domino>> hands*) – начало игры, где hands – словарь домино пользователей (рук)  *void onGameAbort(String message)* – окончание игры  **Реализация интерфейса NetworkEvents:**  *void onTryAbortGame()* – попытка закончить игру  *void onTryDrawDominoFromMarket()* – попытка взять домино из магазина  *void onTryPass()* – попытка сделать pass  *void onTryStartGame()* – попытка начать игру  *void onTryPlaceDomino(Domino tile, boolean isLeft)* – попытка положить домино  *void onDebugMessage(String name, String message)* - отлаживание |
| NetworkEvents | Интерфейс для событий сети (от сервера)  *void onTryAbortGame()* – попытка закончить игру  *void onTryDrawDominoFromMarket()* – попытка взять домино из магазина  *void onTryPass()* – попытка сделать pass  *void onTryStartGame()* – попытка начать игру  *void onTryPlaceDomino(Domino tile, boolean isLeft)* – попытка положить домино  *void onDebugMessage(String name, String message)* - отлаживание |

Анализ результатов и недостатки:

Недостатки:

* Добавить валидацию входных данных: проверки на null и пустые строки
* Удалить неиспользуемые методы геттеры и сеттеры
* Метод onGameStart в интерфейсе NetworkEvents принимает руки всех пользователей, а должен только свою и количество домино в каждой *не своей* руке