OLD VERSION: CREATE GAME

sd enterGame

Client → **Server**

1: Ping()

Ack()

> The Ping message is used to inform the Server of the effective presence of the client. The "acknowledgment message" is used to inform the client whether it has successfully established a connection or not.

loop : ifFailed [Guard]

2: SetNickname()

2.1: nameVerify()

alt

NotValidNickname

[Else]

3: SetNickname()

validNickname

> Using the setNickname method, the client sends its nickname to the server, which will verify (nameVerify())if this nickname has been used by other players in the lobby or not. . Following this, an alternative condition occurs (alt): if the nickname is invalid, the client can reattempt setting its nickname using a while loop; otherwise, the server will send a message to notify the client that the nickname has been successfully added to the list of players.

4: MatchList()

> The server sends a MatchList to the client so that the latter can choose which game to join.

alt [if Client want a new match or there is no match]

5: createNewMatch()()

> Thanks to the alternative condition, the client can choose between creating a new lobby by requesting createNewMatch() to the server or joining an existing lobby with joinMatch(). If there is no existing lobby, the client must create a new lobby.

[join a already created match]

6: joinMatch()

loop : Until all the players are ready

7: playerReadyToStart()

alt [ready]

allReady

[new Player In]

8: newPlayerInMatchMessage()

[notReady]

9: notReady()

> The client signals its readiness to start a game by invoking the method playerReadyToStart() on the server. The server can respond in three different ways:
>
> 1.If all players are ready, the server will send an AllReadyMessage() to the client.
> 2.If a new player joins the lobby while the process is ongoing, the server will inform the client using the method newPlayerInMatchMessage().
> 3.If any member of the group is not prepared, the server will notify the client by sending a message using notReady().
>
> The while loop will continue until all players are ready.

NEW VERSION: CREATE GAME

Actor	AllMatchesController	SingleMatchController	Match	CardParsing	PointTable	PlayerColor	Player	Board	joinFailMsg	newPlayerInMsg	ActionSuccessMsg	ViewModel	Listener	gameStartMsg	joinSuccessMsg

1:createNewMatch

1.1:<<create>>

1.1.1:<<create>>

1.1.1.1:<<create>>

1.1.1.2:<<create>>

1.1.1.3:loadInitialCards

1.1.1.4:loadGoldCards

1.1.1.5:loadTargetCards

1.1.1.6:loadResourceCards

1.1.1.7:shuffleAll

1.1.1.8:values

1.2:joinMatch

1.2.1:<<create>>

1.2.1.1:<<create>>

1.2.2:<<create>>

1.2.3:<<create>>

1.2.4:addPlayer

1.2.4.1:addPlayer

1.2.4.2:<<create>>

1.2.4.2.1:<<create>>

1.2.4.2.1.1:<<create>>

1.2.4.3:notifyAllListeners

1.2.4.3.1:update

1.2.4.4:addListener

1.2.4.4.1:addListener

1.2.4.5:startGame

1.2.4.5.1:extractCommonTargetCard

1.2.4.5.2:distributeCardsAndSetBoards

1.2.4.5.2.1:getAResourceCard

1.2.4.5.2.2:getAResourceCard

1.2.4.5.2.3:getAGoldCard

1.2.4.5.3:extractFirstPlayer

1.2.4.5.4:<<create>>

1.2.4.5.4.1:<<create>>

1.2.4.5.4.1.1:<<create>>

1.2.4.5.4.2:<<create>>

1.2.4.5.5:notifyAllListeners

1.2.4.5.5.1:update

1.2.5:<<create>>

1.2.6:<<create>>

1.2.6.1:<<create>>

1.2.6.1.1:<<create>>

1.2.7:<<create>>

OLD VERSION: PLAY A CARD

sd Sequence Diagram0

Client	Server

loop : until the card is put [Guard]

alt

1: PlayCard(Card, boolean, x,y)

1.1: VerifyPossibility(Card, boolean, x,y)

AckMessage(true)

2: UpdateBoard()

3: UpdateCardOnHand()

4: UpdateScore()

[else]

5: AckMessage(false)

The playCard method has four arguments :a card, a boolean value, x and y coordinates. The boolean indicates whether the card is facing front or back, and the coordinates indicate the card's position on the board. Using this method, the client requests the server's permission to put a card on the board. The server can respond in two ways:

1. If the server replies with AckMessage(true), it means the card has been successfully placed on the board, satisfying the card's requirements and the rules for card placement. Following this, the server updates the board using UpdateBoard(), the player's hand cards with UpdateCardOnHand(), and the player's score with UpdateScore(). Finally, the while loop ends.

2.If the server responds with AckMessage(false), it indicates that it was unable to place the card on the board. In this case, the client needs to repeat the process within the while loop.

NEW VERSION: PLAY A CARD

Actor | SingleMatchController | ResourceCard | Match | GoldCard | PointTable | playCardSuccess | ActionSuccessMsg | ViewModel | Listener | ActionNotRecognize

1:playACardOnHand

1.1:checkRequirements

1.2:updatePoint

1.2.1:getGoalPoint

1.2.1.1:goalCount

1.2.2:updatePoint

1.3:nextPlayer

1.4:<<create>>

1.4.1:<<create>>

1.4.1.1:<<create>>

1.5:notifyAllListeners

1.5.1:update

1.6:getListenerOf

1.7:<<create>>

1.8:update

1.9:getListenerOf

1.10:<<create>>

1.11:update

OLD VERSION: GET A CARD

Client	Server

loop : IsYourTurn [isYourTurn]

1: ScegliCarta(Deck)()

Using the "chooseCard" method, the client will request a permission from the server to select either one of the first three cards from the Resource Deck or one of the first three cards from the GoldDeck. It is always allowed to reveal the first two cards of the decks.

alt

[Guard]

AckMessage(true)

2: UpdateDeck()

If both the ResourceDeck and the GoldDeck are not empty, the server will send an AckMessage to the client, indicating that the request has been accepted. Afterwards, it will update the decks and the player's hand cards using the UpdateDeck and UpdateCardOnHand methods.

3: UpdateCardOn()()

[Else]

4: AckMassage(false)

If both decks are empty, the server will send an AckMessage notifying the client that the request has been denied.

NEW VERSION: GET A CARD

Actor	SingleMatchController	Player	Board	ActionNotRecognize	Listener	Match	PointTable	drawCardSuccess	ActionSuccessMsg	ViewModel	NowIsYourRoundMsg

1:getACard

1.1:<<create>>

1.1.1:<<create>>

1.2:getListenerOf

1.3:<<create>>

1.4:update

1.5:getAGoldCard

1.6:getAResourceCard

1.7:ifLastTurn

1.7.1:findMaxPoint

1.7.2:notifyAllListeners

1.7.2.1:update

1.8:nextPlayer

1.9:<<create>>

1.9.1:<<create>>

1.9.1.1:<<create>>

1.10:notifyAllListeners

1.10.1:update

1.11:getListenerOf

1.12:<<create>>

1.13:update

1.14:getListenerOf

1.15:<<create>>

1.16:update

1.17:getListenerOf

1.18:<<create>>

1.19:update

1.20:getListenerOf

1.21:<<create>>

1.22:update