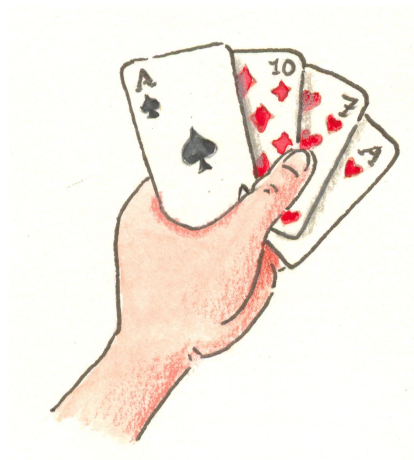


SPEZIFIKATION

7. November 2013



NET-WIZHEARTS

Phase	Verantwortlicher	E-Mail
Pflichtenheft	Alina Meixl	alina@meixl.de
Entwurf	Viktoria Witka	witkaviktoria@freenet.de
Spezifikation	Daniel Riedl	dariedl14@yahoo.de
Implementation	Andreas Altenbuchner	a.andi007@gmail.com
Verifikation	Patrick Kubin	kubin@fim.uni-passau.de
Präsentation	w	w

Teilweise erzeugt von Doxygen 1.8.5

Inhaltsverzeichnis

1	Systemarchitektur	2
2	Klassendiagramm	3
2.1	Packages	3
3	Änderungen am Entwurf	4
3.1	ComObjects	4
3.2	Server	4
3.2.1	Server-Interface	4
3.3	Client-Model	4
3.3.1	MVMessages	4
4	JUnit-Tests	5
5	Exceptions	5
6	Hierarchie-Verzeichnis	5
6.1	Klassenhierarchie	5
7	Klassen-Verzeichnis	8
7.1	Auflistung der Klassen	8
8	Klassen-Dokumentation	12
8.1	Client.View.Card Klassenreferenz	12
8.1.1	Ausführliche Beschreibung	13
8.1.2	Beschreibung der Konstruktoren und Destruktoren	13
8.2	Ruleset.Card Klassenreferenz	13
8.2.1	Ausführliche Beschreibung	13
8.2.2	Dokumentation der Elementfunktionen	14
8.3	Ruleset.CardDeck Klassenreferenz	15
8.4	Ruleset.CardDeckBuilder Klassenreferenz	15
8.5	Client.CardID Enum-Referenz	15
8.6	Client.View.ChooseCards Klassenreferenz	15
8.6.1	Dokumentation der Elementfunktionen	15
8.7	Client.View.Chooseltem Klassenreferenz	15
8.7.1	Ausführliche Beschreibung	16
8.7.2	Dokumentation der Elementfunktionen	16
8.8	Client.ClientController Klassenreferenz	16
8.9	Ruleset.ClientHearts Klassenreferenz	16
8.9.1	Dokumentation der Elementfunktionen	16
8.10	Server.ClientListenerThread Klassenreferenz	17

8.11	Server.LobbyServer.ClientListenerThread Klassenreferenz	17
8.11.1	Ausführliche Beschreibung	17
8.12	Client.ClientMain Klassenreferenz	17
8.12.1	Dokumentation der Elementfunktionen	17
8.13	Client.ClientModel Klassenreferenz	17
8.13.1	Ausführliche Beschreibung	18
8.13.2	Dokumentation der Elementfunktionen	18
8.14	Ruleset.ClientRuleset Klassenreferenz	20
8.14.1	Ausführliche Beschreibung	20
8.14.2	Dokumentation der Elementfunktionen	20
8.15	Client.ClientState Enum-Referenz	21
8.16	Ruleset.ClientWizard Klassenreferenz	21
8.16.1	Dokumentation der Elementfunktionen	22
8.17	Ruleset.Colour Enum-Referenz	22
8.18	ComObjects.ComBeenKicked Klassenreferenz	22
8.18.1	Ausführliche Beschreibung	22
8.18.2	Beschreibung der Konstruktoren und Destruktoren	22
8.18.3	Dokumentation der Elementfunktionen	22
8.19	ComObjects.ComChatMessage Klassenreferenz	23
8.19.1	Ausführliche Beschreibung	23
8.19.2	Beschreibung der Konstruktoren und Destruktoren	23
8.19.3	Dokumentation der Elementfunktionen	23
8.20	ComObjects.ComClientLeave Klassenreferenz	23
8.20.1	Ausführliche Beschreibung	24
8.21	ComObjects.ComClientQuit Klassenreferenz	24
8.21.1	Ausführliche Beschreibung	24
8.22	ComObjects.ComCreateGameRequest Klassenreferenz	24
8.22.1	Ausführliche Beschreibung	24
8.22.2	Beschreibung der Konstruktoren und Destruktoren	24
8.22.3	Dokumentation der Elementfunktionen	25
8.23	ComObjects.ComInitGameLobby Klassenreferenz	25
8.23.1	Ausführliche Beschreibung	26
8.23.2	Beschreibung der Konstruktoren und Destruktoren	26
8.23.3	Dokumentation der Elementfunktionen	26
8.24	ComObjects.ComInitLobby Klassenreferenz	26
8.24.1	Ausführliche Beschreibung	26
8.24.2	Beschreibung der Konstruktoren und Destruktoren	27
8.24.3	Dokumentation der Elementfunktionen	28
8.25	ComObjects.ComJoinRequest Klassenreferenz	28
8.25.1	Ausführliche Beschreibung	28

8.25.2	Beschreibung der Konstruktoren und Destrukturen	28
8.25.3	Dokumentation der Elementfunktionen	29
8.26	ComObjects.ComKickPlayerRequest Klassenreferenz	29
8.26.1	Ausführliche Beschreibung	29
8.26.2	Beschreibung der Konstruktoren und Destrukturen	29
8.26.3	Dokumentation der Elementfunktionen	29
8.27	ComObjects.ComLobbyUpdateGamelist Klassenreferenz	30
8.27.1	Ausführliche Beschreibung	30
8.27.2	Beschreibung der Konstruktoren und Destrukturen	30
8.27.3	Dokumentation der Elementfunktionen	30
8.28	ComObjects.ComLoginRequest Klassenreferenz	31
8.28.1	Ausführliche Beschreibung	31
8.28.2	Beschreibung der Konstruktoren und Destrukturen	31
8.28.3	Dokumentation der Elementfunktionen	31
8.29	ComObjects.ComObject Klassenreferenz	31
8.30	ComObjects.ComRuleset Klassenreferenz	32
8.30.1	Ausführliche Beschreibung	32
8.30.2	Beschreibung der Konstruktoren und Destrukturen	32
8.30.3	Dokumentation der Elementfunktionen	34
8.31	ComObjects.ComServerAcknowledgement Klassenreferenz	34
8.32	ComObjects.ComStartGame Klassenreferenz	34
8.32.1	Ausführliche Beschreibung	34
8.33	ComObjects.ComUpdatePlayerlist Klassenreferenz	34
8.33.1	Ausführliche Beschreibung	35
8.33.2	Beschreibung der Konstruktoren und Destrukturen	35
8.33.3	Dokumentation der Elementfunktionen	35
8.34	ComObjects.ComWarning Klassenreferenz	35
8.34.1	Ausführliche Beschreibung	36
8.34.2	Beschreibung der Konstruktoren und Destrukturen	36
8.34.3	Dokumentation der Elementfunktionen	36
8.35	Client.View.CreateGame Klassenreferenz	36
8.35.1	Ausführliche Beschreibung	37
8.35.2	Beschreibung der Konstruktoren und Destrukturen	37
8.35.3	Dokumentation der Elementfunktionen	37
8.36	Client.View.DiscardPile Klassenreferenz	37
8.36.1	Ausführliche Beschreibung	37
8.37	Client.View.DrawDeck Klassenreferenz	37
8.37.1	Ausführliche Beschreibung	37
8.38	Client.View.Game Klassenreferenz	37
8.38.1	Ausführliche Beschreibung	38

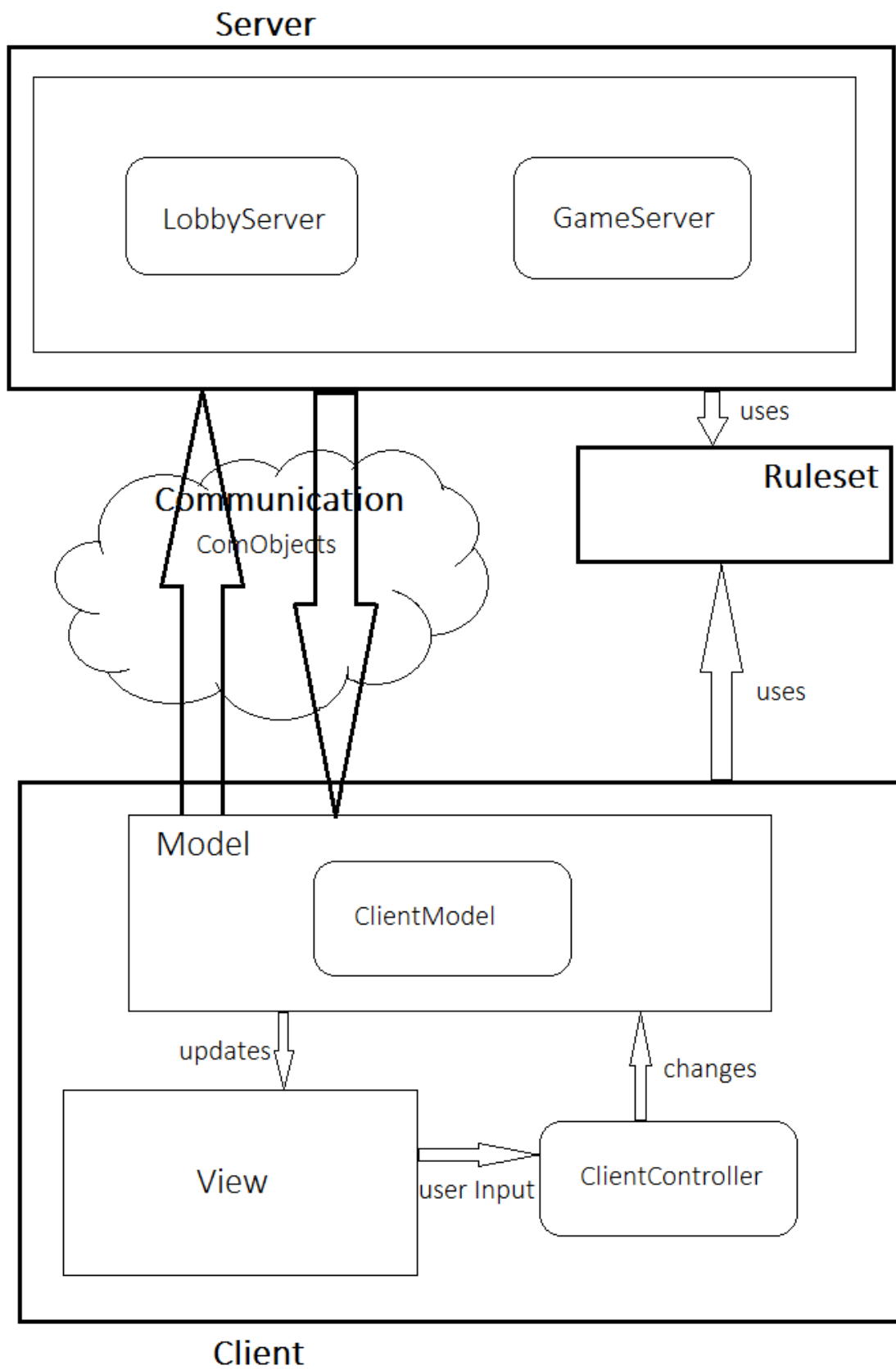
8.38.2	Beschreibung der Konstruktoren und Destruktoren	38
8.38.3	Dokumentation der Elementfunktionen	38
8.39	Ruleset.GameClientUpdate Klassenreferenz	38
8.39.1	Ausführliche Beschreibung	39
8.39.2	Dokumentation der Elementfunktionen	39
8.40	Client.View.GameLobby Klassenreferenz	39
8.40.1	Ausführliche Beschreibung	40
8.40.2	Dokumentation der Elementfunktionen	40
8.41	Client.View.GamePanel Klassenreferenz	40
8.41.1	Ausführliche Beschreibung	40
8.42	Ruleset.GamePhase Enum-Referenz	40
8.43	Server.GameServer Klassenreferenz	40
8.43.1	Ausführliche Beschreibung	41
8.43.2	Beschreibung der Konstruktoren und Destruktoren	41
8.43.3	Dokumentation der Elementfunktionen	41
8.44	Server.GameServerRepresentation Klassenreferenz	43
8.44.1	Ausführliche Beschreibung	43
8.44.2	Beschreibung der Konstruktoren und Destruktoren	43
8.45	Ruleset.GameState Klassenreferenz	44
8.45.1	Ausführliche Beschreibung	44
8.45.2	Dokumentation der Elementfunktionen	44
8.46	Ruleset.HearthsDeck Klassenreferenz	45
8.47	Ruleset.HeartsCard Klassenreferenz	46
8.47.1	Dokumentation der Elementfunktionen	46
8.48	Client.View.HeartsCard Klassenreferenz	46
8.48.1	Ausführliche Beschreibung	47
8.48.2	Beschreibung der Konstruktoren und Destruktoren	47
8.48.3	Dokumentation der Elementfunktionen	47
8.49	Ruleset.HeartsData Klassenreferenz	47
8.49.1	Dokumentation der Elementfunktionen	47
8.50	Ruleset.HeartsID Enum-Referenz	48
8.51	Client.View.InputNumber Klassenreferenz	48
8.51.1	Ausführliche Beschreibung	48
8.51.2	Dokumentation der Elementfunktionen	48
8.52	Client.View.Language Enum-Referenz	48
8.52.1	Ausführliche Beschreibung	48
8.53	Client.View.Lobby Klassenreferenz	48
8.53.1	Ausführliche Beschreibung	49
8.53.2	Dokumentation der Elementfunktionen	49
8.54	Server.LobbyServer Klassenreferenz	49

8.54.1 Ausführliche Beschreibung	50
8.54.2 Dokumentation der Elementfunktionen	50
8.55 Client.View.Login Klassenreferenz	51
8.55.1 Ausführliche Beschreibung	51
8.55.2 Dokumentation der Elementfunktionen	51
8.56 Client.MessageListenerThread Klassenreferenz	52
8.57 ComObjects.MsgCard Klassenreferenz	52
8.57.1 Ausführliche Beschreibung	52
8.57.2 Beschreibung der Konstruktoren und Destruktoren	52
8.57.3 Dokumentation der Elementfunktionen	52
8.58 ComObjects.MsgCardRequest Klassenreferenz	53
8.58.1 Ausführliche Beschreibung	53
8.59 ComObjects.MsgGameEnd Klassenreferenz	53
8.59.1 Ausführliche Beschreibung	53
8.60 ComObjects.MsgMultiCards Klassenreferenz	53
8.60.1 Ausführliche Beschreibung	53
8.60.2 Beschreibung der Konstruktoren und Destruktoren	54
8.60.3 Dokumentation der Elementfunktionen	55
8.61 ComObjects.MsgMultiCardsRequest Klassenreferenz	55
8.61.1 Dokumentation der Elementfunktionen	55
8.62 ComObjects.MsgMultipleCardsRequest Klassenreferenz	55
8.63 ComObjects.MsgNumber Klassenreferenz	56
8.63.1 Ausführliche Beschreibung	56
8.63.2 Beschreibung der Konstruktoren und Destruktoren	56
8.63.3 Dokumentation der Elementfunktionen	56
8.64 ComObjects.MsgNumberRequest Klassenreferenz	56
8.65 ComObjects.MsgSelection Klassenreferenz	57
8.65.1 Ausführliche Beschreibung	57
8.65.2 Beschreibung der Konstruktoren und Destruktoren	57
8.65.3 Dokumentation der Elementfunktionen	57
8.66 ComObjects.MsgSelectionRequest Klassenreferenz	57
8.66.1 Ausführliche Beschreibung	58
8.67 ComObjects.MsgUser Klassenreferenz	58
8.67.1 Ausführliche Beschreibung	58
8.67.2 Beschreibung der Konstruktoren und Destruktoren	58
8.67.3 Dokumentation der Elementfunktionen	58
8.68 Client.MVMessages Schnittstellenreferenz	58
8.69 Ruleset.OtherData Klassenreferenz	58
8.69.1 Dokumentation der Elementfunktionen	59
8.70 Client.View.OtherPlayer Klassenreferenz	59

8.70.1 Ausführliche Beschreibung	59
8.71 Client.View.OwnHand Klassenreferenz	59
8.71.1 Ausführliche Beschreibung	59
8.72 Client.View.Password Klassenreferenz	59
8.72.1 Ausführliche Beschreibung	60
8.72.2 Dokumentation der Elementfunktionen	60
8.73 Server.Player Klassenreferenz	60
8.73.1 Ausführliche Beschreibung	60
8.73.2 Beschreibung der Konstruktoren und Destruktoren	60
8.73.3 Dokumentation der Elementfunktionen	61
8.74 Ruleset.PlayerState Klassenreferenz	61
8.74.1 Ausführliche Beschreibung	61
8.74.2 Dokumentation der Elementfunktionen	62
8.75 ComObjects.RulesetMessage Klassenreferenz	63
8.75.1 Ausführliche Beschreibung	63
8.76 Ruleset.RulesetType Enum-Referenz	63
8.77 Client.View.ScoreWindow Klassenreferenz	63
8.77.1 Ausführliche Beschreibung	64
8.77.2 Dokumentation der Elementfunktionen	64
8.78 Server.Server Klassenreferenz	64
8.78.1 Ausführliche Beschreibung	64
8.78.2 Dokumentation der Elementfunktionen	64
8.79 Ruleset.ServerHearts Klassenreferenz	65
8.79.1 Ausführliche Beschreibung	66
8.79.2 Dokumentation der Elementfunktionen	66
8.80 Server.ServerMain Klassenreferenz	66
8.80.1 Ausführliche Beschreibung	66
8.80.2 Dokumentation der Elementfunktionen	66
8.81 Ruleset.ServerRuleset Klassenreferenz	66
8.81.1 Ausführliche Beschreibung	67
8.81.2 Dokumentation der Elementfunktionen	67
8.82 Ruleset.ServerWizard Klassenreferenz	68
8.82.1 Ausführliche Beschreibung	68
8.82.2 Dokumentation der Elementfunktionen	68
8.83 Client.View.Notification Enum-Referenz	69
8.84 Client.View.Warning Klassenreferenz	69
8.84.1 Ausführliche Beschreibung	69
8.84.2 Dokumentation der Elementfunktionen	69
8.85 Ruleset.WizardCard Klassenreferenz	69
8.86 Ruleset.WizardDeck Klassenreferenz	70

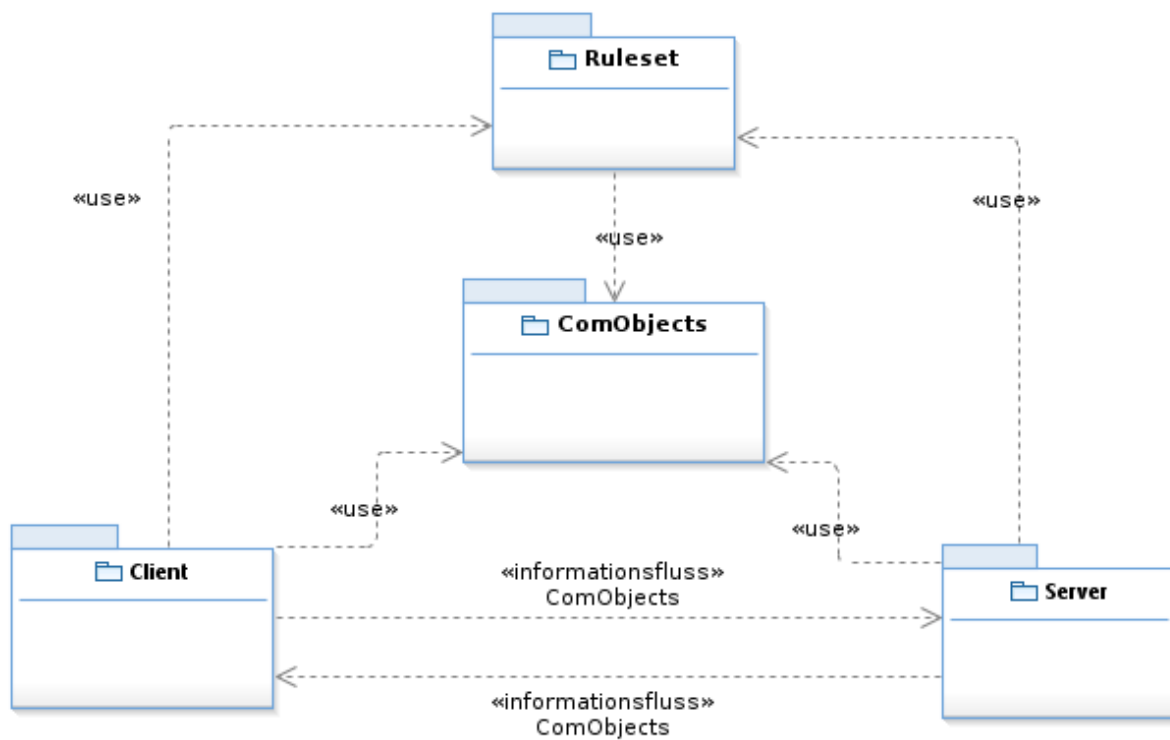
8.87 Client.View.WizCard Klassenreferenz	70
8.87.1 Ausführliche Beschreibung	70
8.87.2 Beschreibung der Konstruktoren und Destrukturen	70
8.87.3 Dokumentation der Elementfunktionen	70
8.88 Ruleset.WizData Klassenreferenz	71
8.88.1 Dokumentation der Elementfunktionen	71
8.89 Ruleset.WizID Enum-Referenz	73

1 Systemarchitektur



2 Klassendiagramm

2.1 Packages



3 Änderungen am Entwurf

3.1 ComObjects

3.2 Server

3.2.1 Server-Interface

3.3 Client-Model

3.3.1 MVMessages

löschen von Interfaces??

4 JUnit-Tests

5 Exceptions

6 Hierarchie-Verzeichnis

6.1 Klassenhierarchie

Die Liste der Ableitungen ist -mit Einschränkungen- alphabetisch sortiert:

Ruleset.Card	13
Ruleset.HeartsCard	46
Ruleset.WizardCard	69
Ruleset.CardDeck	15
Ruleset.CardDeckBuilder	15
Client.CardID	15
Client.ClientController	16
Client.ClientMain	17
Ruleset.ClientRuleset	20
Ruleset.ClientHearts	16
Ruleset.ClientWizard	21
Client.ClientState	21
Ruleset.Colour	22
ComObjects.ComBeenKicked	22
Client.View.DiscardPile	37
Client.View.DrawDeck	37
Ruleset.GameClientUpdate	38
Ruleset.GamePhase	40
Server.GameServerRepresentation	43
Ruleset.GameState	44
Ruleset.HearthsDeck	45
Ruleset.HeartsID	48
Client.View.Language	48
Client.MessageListenerThread	52
ComObjects.MsgCardRequest	53

Client.MVMessages	58
Ruleset.OtherData	58
Ruleset.HeartsData	47
Ruleset.WizData	71
Client.View.OtherPlayer	59
Client.View.OwnHand	59
Ruleset.PlayerState	61
Ruleset.RulesetType	63
Runnable	
Server.ClientListenerThread	17
Server.LobbyServer.ClientListenerThread	17
Server.Player	60
Server.Server	64
Server.GameServer	40
Server.LobbyServer	49
Server.ServerMain	66
Ruleset.ServerRuleset	66
Ruleset.ServerHearts	65
Ruleset.ServerWizard	68
Client.ViewNotification	69
Ruleset.WizardDeck	70
Ruleset.WizID	73
JFrame	
Client.View.CreateGame	36
Client.View.Game	37
Client.View.GameLobby	39
Client.View.Lobby	48
Client.View.Login	51
Client.View.Password	59
JLabel	
Client.View.Card	12
Client.View.HeartsCard	46
Client.View.WizCard	70
JPanel	

Client.View.GamePanel	40
Observable	
Client.ClientModel	17
Observer	
Client.View.ChooseCards	15
Client.View.ChooseItem	15
Client.View.CreateGame	36
Client.View.Game	37
Client.View.GameLobby	39
Client.View.InputNumber	48
Client.View.Lobby	48
Client.View.Login	51
Client.View.Password	59
Client.View.ScoreWindow	63
Client.View.Warning	69
Serializable	
ComObjects.ComObject	31
ComObjects.ComChatMessage	23
ComObjects.ComClientLeave	23
ComObjects.ComClientQuit	24
ComObjects.ComCreateGameRequest	24
ComObjects.ComInitGameLobby	25
ComObjects.ComInitLobby	26
ComObjects.ComJoinRequest	28
ComObjects.ComKickPlayerRequest	29
ComObjects.ComLobbyUpdateGamelist	30
ComObjects.ComLoginRequest	31
ComObjects.ComRuleset	32
ComObjects.ComServerAcknowledgement	34
ComObjects.ComStartGame	34
ComObjects.ComUpdatePlayerlist	34
ComObjects.ComWarning	35
ComObjects.RulesetMessage	63

ComObjects.MsgCard	52
ComObjects.MsgGameEnd	53
ComObjects.MsgMultiCards	53
ComObjects.MsgMultiCardsRequest	55
ComObjects.MsgMultipleCardsRequest	55
ComObjects.MsgNumber	56
ComObjects.MsgNumberRequest	56
ComObjects.MsgSelection	57
ComObjects.MsgSelectionRequest	57
ComObjects.MsgUser	58

7 Klassen-Verzeichnis

7.1 Auflistung der Klassen

Hier folgt die Aufzählung aller Klassen, Strukturen, Varianten und Schnittstellen mit einer Kurzbeschreibung:

Client.View.Card	
Card ist die View-seitige Repräsentation einer Karte	12
Ruleset.Card	
Diese Klasse modelliert eine Spielkarte	13
Ruleset.CardDeck	15
Ruleset.CardDeckBuilder	15
Client.CardID	15
Client.View.ChooseCards	15
Client.View.ChooseItem	
Dieses Fenster ermöglicht es dem Spieler aus einer Liste von Items eines auszuwählen	15
Client.ClientController	16
Ruleset.ClientHearts	
Diese Klasse bildet das Regelwerk für den Client bei einer Partie Hearts	16
Server.ClientListenerThread	17
Server.LobbyServer.ClientListenerThread	
Diese Klasse ist für das Zustandekommen von Clientverbindungen zuständig	17
Client.ClientMain	
Die ClientMain Klasse startet den Spielclient und initialisiert dessen Komponenten	17
Client.ClientModel	
Implementiert das Client Model	17

Ruleset.ClientRuleset	
ClientRuleset ist eine abstrakte Klasse und wird zur Regelvorauswertung im Client verwendet	20
Client.ClientState	
Dieser Enumerator enthält alle Zustände in denen sich der Client befinden kann	21
Ruleset.ClientWizard	
Diese Klasse bildet das Regelwerk für den Client bei einer Partie Wizard	21
Ruleset.Colour	
Repräsentiert die Farbe einer Karte	22
ComObjects.ComBeenKicked	
Diese Klasse ist ein spezielles Kommunikations-Objekt	22
ComObjects.ComChatMessage	
Diese Klasse ist ein spezielles Kommunikations-Objekt	23
ComObjects.ComClientLeave	
Diese Klasse ist ein spezielles Kommunikations-Objekt	23
ComObjects.ComClientQuit	
Diese Klasse ist ein spezielles Kommunikations-Objekt	24
ComObjects.ComCreateGameRequest	
Diese Klasse ist ein spezielles Kommunikations-Objekt	24
ComObjects.ComInitGameLobby	
Diese Klasse ist ein spezielles Kommunikations-Objekt	25
ComObjects.ComInitLobby	
Diese Klasse ist ein spezielles Kommunikations-Objekt	26
ComObjects.ComJoinRequest	
Diese Klasse ist ein spezielles Kommunikations-Objekt	28
ComObjects.ComKickPlayerRequest	
Diese Klasse ist ein spezielles Kommunikations-Objekt	29
ComObjects.ComLobbyUpdateGamelist	
Diese Klasse ist ein spezielles Kommunikations-Objekt	30
ComObjects.ComLoginRequest	
Diese Klasse ist ein spezielles Kommunikations-Objekt	31
ComObjects.ComObject	31
ComObjects.ComRuleset	
Diese Klasse ist ein spezielles Kommunikations-Objekt	32
ComObjects.ComServerAcknowledgement	
Diese Klasse ist ein spezielles Kommunikations-Objekt	34
ComObjects.ComStartGame	
Diese Klasse ist ein spezielles Kommunikations-Objekt	34
ComObjects.ComUpdatePlayerlist	
Diese Klasse ist ein spezielles Kommunikations-Objekt	34
ComObjects.ComWarning	
Diese Klasse ist ein spezielles Kommunikations-Objekt	35

Client.View.CreateGame	
Das Fenster CreateGame dient dem Benutzer zur Erstellung eines neuen Spieles	36
Client.View.DiscardPile	
Stellt einen Ablagestapel dar, dieser kann sowohl für jeden Spieler einzeln oder für alle Spieler gemeinsam in der Mitte des Spielfeldes angezeigt werden	37
Client.View.DrawDeck	
Stellt einen Aufnahmestapel dar	37
Client.View.Game	
Im Game Fenster läuft das Spiel ab. Es enthält den Spielchat und ein GamePanel	37
Ruleset.GameClientUpdate	
Das GameClientUpdate wird vom RuleSet aber den GameServer an den Client geschickt und enthält alle Änderungen des GameState , die für den Client relevant sind	38
Client.View.GameLobby	
Die GameLobby modelliert das Wartefenster, in dem beigetretene Spieler auf den Start des Spieles durch den Spielleiter warten	39
Client.View.GamePanel	
Das Panel ist die Komponente des Game-Fensters, welche das eigentliche Spiel darstellt	40
Ruleset.GamePhase	
Die GamePhase modelliert die verschiedenen Zustände des Spiels im GameState	40
Server.GameServer	
Diese Klasse ist für die Spielverwaltung zuständig	40
Server.GameServerRepresentation	
Dies eine Klasse, die Informationen über den Zustand eines Spielers bereithält	43
Ruleset.GameState	
Das GameState modelliert einen aktuellen Spielzustand, es wird vom GameServer instanziiert und vom RuleSet bearbeitet	44
Ruleset.HeartsDeck	45
Ruleset.HeartsCard	
Modelliert eine Heartskarte	46
Client.View.HeartsCard	
HeartsCard ist die View-seitige Repräsentation einer Hearts-Karte	46
Ruleset.HeartsData	
Die zusätzlichen Informationen eines Spielers zum Spiel Hearts	47
Ruleset.HeartsID	
Die eindeutigen IDs zu jeder Heartskarte	48
Client.View.InputNumber	
In diesem Fenster, kann der Benutzer eine Zahl eingeben	48
Client.View.Language	
Language stellt Repräsentationen verschiedener Sprachen dar, die von der GUI verwendet werden, um festzustellen welche Anzeigesprache verwendet werden soll	48
Client.View.Lobby	
Diese Klasse erzeugt die Ansicht der ServerLobby auf der Client Seite, in der die Spieler neue Spiele erstellen oder offenen beitreten können	48

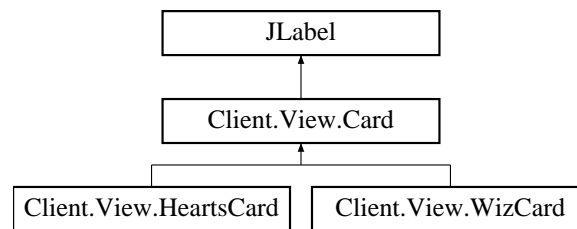
Server.LobbyServer	
Diese Klasse ist für die Verwaltung der Spiellobby auf dem Server verantwortlich	49
Client.View.Login	
Das Login-Fenster repräsentiert den initialen Dialog zwischen Benutzer und Client	51
Client.MessageListenerThread	52
ComObjects.MsgCard	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	52
ComObjects.MsgCardRequest	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	53
ComObjects.MsgGameEnd	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	53
ComObjects.MsgMultiCards	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	53
ComObjects.MsgMultiCardsRequest	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	55
ComObjects.MsgMultipleCardsRequest	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	55
ComObjects.MsgNumber	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	56
ComObjects.MsgNumberRequest	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	56
ComObjects.MsgSelection	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	57
ComObjects.MsgSelectionRequest	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	57
ComObjects.MsgUser	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	58
Client.MVMessages	58
Ruleset.OtherData	
OtherData ist abstract und speichert die zusätzlichen Informationen eines Spielers	58
Client.View.OtherPlayer	
Zeigt die Informationen über die anderen Spieler an, also den Namen, ein Symbol für die verdeckte Hand und das Label für zusätzliche Angaben	59
Client.View.OwnHand	
Stellt die Karten dar, die der Spieler auf der Hand hat	59
Client.View.Password	
Dieses Fenster ermöglicht die Eingabe eines Passwortes um einem Passwortgeschütztem Spiel beizutreten oder per 'Leave' wieder in die Lobby zurückzukehren	59
Server.Player	
Die Player-Klasse wird zum Versenden von Java Serializable Objects verwendet	60

Ruleset.PlayerState	Repräsentiert den Spielzustand eines Spielers, und wird unter anderem im GameState gespeichert	61
ComObjects.RulesetMessage	Diese Klasse ist eine Verfeinerung der ComRuleset-Klasse	63
Ruleset.RulesetType	Die verschiedenen Regelwerke	63
Client.View.ScoreWindow	Dieses Fenster zeigt den momentanen Punktestand nach jeder Runde und den Gesamtpunktestand am Ende des Spieles an	63
Server.Server	Ist ein abstrakte Klasse, von der die Klassen LobbyServer und GameServer erben	64
Ruleset.ServerHearts	Diese Klasse erstellt das Regelwerk zum Spiel Hearts	65
Server.ServerMain	Diese Klasse startet den Server und ist für die Konfiguration und Wartung des Servers verantwortlich	66
Ruleset.ServerRuleset	Das ServerRuleset ist eine abstrakte Klasse und für den Ablauf und die Einhaltung der Regeln eines Spiels zuständig (/L280/)	66
Ruleset.ServerWizard	Diese Klasse erstellt das Regelwerk zum Spiel Wizard	68
Client.ViewNotification		69
Client.View.Warning	Das Warning-Fenster zeigt dem Benutzer Fehlermeldungen bzw	69
Ruleset.WizardCard	Modelliert eine Wizardkarte	69
Ruleset.WizardDeck		70
Client.View.WizCard	WizCard ist die View-seitige Repräsentation einer Wizard-Karte	70
Ruleset.WizData	Die zusätzlichen Informationen eines Spielers zum Spiel Wizard	71
Ruleset.WizID	Die eindeutigen IDs zu jeder Wizardkarte	73

8 Klassen-Dokumentation

8.1 Client.View.Card Klassenreferenz

Klassendiagramm für Client.View.Card:



Öffentliche Methoden

- [Card](#) (String s)

8.1.1 Ausführliche Beschreibung

Sie wird verwendet um einzelne Karten auf das Spielfeld zu zeichnen. Dazu enthält sie die Pfadangabe zu dem Ordner, in dem die Bilder der Karten gespeichert sind, und eine ID, um das genaue Bild zu spezifizieren.

8.1.2 Beschreibung der Konstruktoren und Destruktoren

8.1.2.1 Client.View.Card.Card (String s)

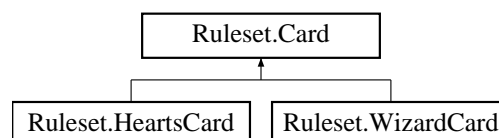
Erstellt eine neue Karte für die Anzeige und zeichnet dafür das Bild, das durch die Pfadangabe s angegeben ist.

Parameter

s	Pfadangabe zum zu zeichnenden Bild
---	------------------------------------

8.2 Ruleset.Card Klassenreferenz

Klassendiagramm für Ruleset.Card:



Öffentliche Methoden

- int [getValue](#) ()
- [Colour](#) [getColour](#) ()

Geschützte Methoden

- int [getIDValue](#) (WizID id)
- int [getIDValue](#) (HeartsID id)
- [Colour](#) [getIDColour](#) (WizID id)
- [Colour](#) [getIDColour](#) (HeartsID id)

8.2.1 Ausführliche Beschreibung

Jede Karte besitzt als Attribute einen Wert und eine Farbe.

8.2.2 Dokumentation der Elementfunktionen

8.2.2.1 Colour Ruleset.Card.getColour ()

Holt die Farbe der Karte.

Rückgabe

Die Farbe der Karte

8.2.2.2 Colour Ruleset.Card.getIDColour (WizID *id*) [protected]

Holt die Farbe einer Wizardkarte aufgrund seiner ID zurück

Parameter

<i>id</i>	Die WizardID
-----------	--------------

Rückgabe

Gibt die Farbe zurück

8.2.2.3 Colour Ruleset.Card.getIDColour (HeartsID *id*) [protected]

Holt die Farbe einer Heartskarte aufgrund seiner ID zurück

Parameter

<i>id</i>	Die HeartsID
-----------	------------------------------

Rückgabe

Gibt die Farbe zurück

8.2.2.4 int Ruleset.Card.getIDValue (WizID *id*) [protected]

Holt den Wert einer Wizardkarte aufgrund seiner ID zurück

Parameter

<i>id</i>	Die WizardID
-----------	--------------

Rückgabe

Gibt den Wert zurück

8.2.2.5 int Ruleset.Card.getIDValue (HeartsID *id*) [protected]

Holt den Wert einer Heartskarte aufgrund seiner ID zurück

Parameter

<i>id</i>	Die HeartsID
-----------	------------------------------

Rückgabe

Gibt den Wert zurück

8.2.2.6 int Ruleset.Card.getValue ()

Holt den logischen Wert der Karte.

Rückgabe

Der logische Wert der Karte

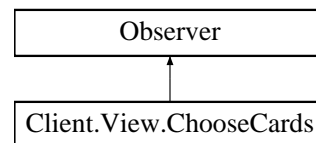
8.3 Ruleset.CardDeck Klassenreferenz

8.4 Ruleset.CardDeckBuilder Klassenreferenz

8.5 Client.CardID Enum-Referenz

8.6 Client.View.ChooseCards Klassenreferenz

Klassendiagramm für Client.View.ChooseCards:



Öffentliche Methoden

- void [update](#) (Observable o, Object arg)

8.6.1 Dokumentation der Elementfunktionen

8.6.1.1 void Client.View.ChooseCards.update (Observable o, Object arg)

Wird durch notify() im [ClientModel](#) aufgerufen.

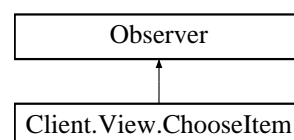
Je nach dem in arg übergebenen Befehl wird ein Update des Fensters ausgeführt oder eine Fehlermeldung angezeigt.

Parameter

<i>o</i>	erwartet ein Objekt von der Klasse ClientModel
<i>arg</i>	erwartet: openChooseCards

8.7 Client.View.ChooseItem Klassenreferenz

Klassendiagramm für Client.View.ChooseItem:



Öffentliche Methoden

- void [update](#) (Observable arg0, Object arg1)

8.7.1 Ausführliche Beschreibung

Autor

m4nkey

8.7.2 Dokumentation der Elementfunktionen

8.7.2.1 void Client.View.Chooseltem.update (Observable *arg0*, Object *arg1*)

Wird durch notify() im [ClientModel](#) aufgerufen.

Je nach dem in arg übergebenen Befehl wird ein Update des Fensters ausgeführt oder eine Fehlermeldung angezeigt.

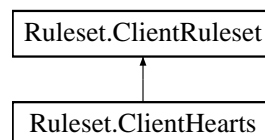
Parameter

<i>o</i>	erwartet ein Objekt von der Klasse ClientModel
<i>arg</i>	erwartet: openChooseltem

8.8 Client.ClientController Klassenreferenz

8.9 Ruleset.ClientHearts Klassenreferenz

Klassendiagramm für Ruleset.ClientHearts:



Öffentliche Methoden

- boolean [isValidMove](#) ([Card](#) card)

Weitere Geerbte Elemente

8.9.1 Dokumentation der Elementfunktionen

8.9.1.1 boolean Ruleset.ClientHearts.isValidMove ([Card](#) *card*) [virtual]

überprüft ob ein gemachter Zug zu dem Spiel Hearts gültig ist.

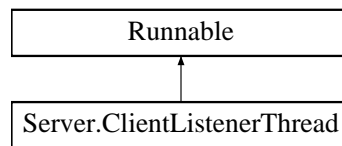
Rückgabe

isValid true falls Zug gültig, false wenn nicht

Implementiert [Ruleset.ClientRuleset](#).

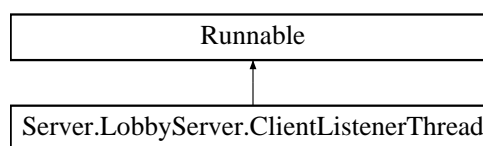
8.10 Server.ClientListenerThread Klassenreferenz

Klassendiagramm für Server.ClientListenerThread:



8.11 Server.LobbyServer.ClientListenerThread Klassenreferenz

Klassendiagramm für Server.LobbyServer.ClientListenerThread:



8.11.1 Ausführliche Beschreibung

Der Thread auf eingehende Clientverbindungen, stellt diese her und instanziiert für jede Verbindung eine Klasse [Player](#). Dieser wird dann dem [LobbyServer](#) übergeben.

Autor

Viktoria

8.12 Client.ClientMain Klassenreferenz

Öffentliche, statische Methoden

- static void [main](#) (final String[] args)

8.12.1 Dokumentation der Elementfunktionen

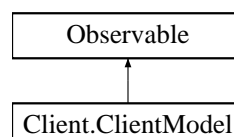
8.12.1.1 static void Client.ClientMain.main (final String[] args) [static]

Parameter

<i>args</i>	
-------------	--

8.13 Client.ClientModel Klassenreferenz

Klassendiagramm für Client.ClientModel:



Klassen

- class **MessageListenerThread**

Öffentliche Methoden

- [ComInitGameLobby](#) [getGameLobbyInit](#) ()
- [ComInitLobby](#) [getServerLobbyInit](#) ()
- [ComLobbyUpdateGamelist](#) [getServerLobbyGamelistUpdate](#) ()
- [ComUpdatePlayerlist](#) [getPlayerlistUpdate](#) ()
- [ComChatMessage](#) [getChatMessage](#) ()
- [CardID](#) [getPlayedCard](#) ()
- void [setLanguage](#) (final [Language](#) language)
- [Language](#) [getLanguage](#) ()
- void [kickPlayer](#) (final String name)
- void [joinGame](#) (final String name)
- void [makeMove](#) ([CardID](#) id)
- void [createConnection](#) (final String username, final String serverAdress, final int port)

8.13.1 Ausführliche Beschreibung

Das Model bedient den Server durch den ListenerThread und leitet Daten an das Regelwerk und View weiter.

8.13.2 Dokumentation der Elementfunktionen

8.13.2.1 void Client.ClientModel.createConnection (final String *username*, final String *serverAdress*, final int *port*)

Erstellt den [MessageListenerThread](#) und führt den Benutzerlogin durch.

Parameter

<i>username</i>	String der eindeutige Benutzername der für den Login verwendet wird.
<i>serverAdress</i>	String die Adresse des spielservers.
<i>port</i>	Integer der Port des Spielservers.

8.13.2.2 ComChatMessage Client.ClientModel.getChatMessage ()

Diese Methode wird von der View aufgerufen um eine neue Chatnachricht abzuholen.

Rückgabe

String die Chatnachricht.

8.13.2.3 ComInitGameLobby Client.ClientModel.getGameLobbyInit ()

Diese Methode wird von der View beim betreten der Spiellobby aufgerufen und liefert eine Liste von Spielern in der Spiellobby.

Rückgabe

ComInitGameLobby Voller Datensatz für die Spiellobby.

8.13.2.4 Language Client.ClientModel.getLanguage ()

Liefert die Sprache der GUI.

Rückgabe

language Enumerator der die Spielsprache anzeigt.

8.13.2.5 CardID Client.ClientModel.getPlayedCard ()

Gibt der View die gespielte Karte eines anderen Spielers zurück.

Rückgabe

enum [CardID](#). Die Id der Karte

8.13.2.6 ComUpdatePlayerlist Client.ClientModel.getPlayerlistUpdate ()

Diese Methode wird von der View aufgerufen um die Liste der Spieler zu aktualisieren.

Rückgabe

ComUpdatePlayerlist Update für die aktuelle Spielerliste.

8.13.2.7 ComLobbyUpdateGamelist Client.ClientModel.getServerLobbyGamelistUpdate ()

Diese Methode wird von der View aufgerufen und aktualisiert einzelne kommende und abgehende Spieler in den Listen der View.

Rückgabe

ComLobbyUpdateGamelist

8.13.2.8 ComInitLobby Client.ClientModel.getServerLobbyInit ()

Diese Methode wird von der View beim betreten der Serverlobby aufgerufen und liefert eine Liste von Spielern und Spielen in der Serverlobby.

Rückgabe

ComInitLobby Voller Datensatz für die ServerLobby.

8.13.2.9 void Client.ClientModel.joinGame (final String name)

Diese Methode wird von dem [ClientController](#) aufgerufen um einem bereits erstelltem Spiel beizutreten.

Parameter

<i>name</i>	String Der Name des Spiels.
-------------	-----------------------------

8.13.2.10 void Client.ClientModel.kickPlayer (final String name)

Wird vom Controller aufgerufen um einen Spieler aus der Spiellobby zu entfernen.

Parameter

<i>name</i>	des Spielerst welcher entfernt werden soll.
-------------	---

8.13.2.11 void Client.ClientModel.makeMove (CardID id)

Wird vom ClientConroller aufgerufen um eine Karte auszuspielen.

Parameter

<i>id</i>	Die id der gespielten Karte um sie einer logischen Karte zuordnen zu können.
-----------	--

8.13.2.12 void Client.ClientModel.setLanguage (final Language language)

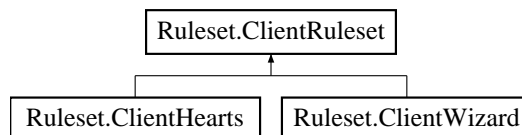
Setzt die Sprache der GUI.

Parameter

<i>language</i>	Enumerator der die Spielsprache anzeigt.
-----------------	--

8.14 Ruleset.ClientRuleset Klassenreferenz

Klassendiagramm für Ruleset.ClientRuleset:

**Öffentliche Methoden**

- void [resolveMessage](#) (MsgUser clientUpdate)
- void [resolveMessage](#) (MsgCardRequest msgCardRequest)
- void [resolveMessage](#) (MsgMultipleCardsRequest msgMultiCardsRequest)
- void [resolveMessage](#) (MsgNumberRequest msgNumber)
- void [resolveMessage](#) (MsgSelectionRequest msgSelection)

Geschützte Methoden

- void [send](#) (RulesetMessage message)

8.14.1 Ausführliche Beschreibung

Dazu benutzt es die [isValidMove\(\)](#) Methode. Des Weiteren kann es vom ClientModel erhaltene RulesetMessages mit der [resolveMessage\(\)](#) Methode behandeln.

8.14.2 Dokumentation der Elementfunktionen**8.14.2.1 void Ruleset.ClientRuleset.resolveMessage (MsgUser clientUpdate)**

Verarbeitet die RulesetMessage dass der Server ein Spielupdate an den Client schickt.

Parameter

<i>clientUpdate</i>	Die Nachricht vom Server
---------------------	--------------------------

8.14.2.2 void Ruleset.ClientRuleset.resolveMessage (**MsgCardRequest** *msgCardRequest*)

Verarbeitet die RulesetMessage dass der Server von dem Spieler verlangt eine Karte zu spielen.

Parameter

<i>msgCard-Request</i>	Die Nachricht vom Server
------------------------	--------------------------

8.14.2.3 void Ruleset.ClientRuleset.resolveMessage (**MsgMultipleCardsRequest** *msgMultiCardsRequest*)

Verarbeitet die RulesetMessage dass der Server von dem Spieler verlangt mehrere Karten anzugeben.

Parameter

<i>msgMultiCards-Request</i>	Die Nachricht vom Server
------------------------------	--------------------------

8.14.2.4 void Ruleset.ClientRuleset.resolveMessage (**MsgNumberRequest** *msgNumber*)

Verarbeitet die RulesetMessage dass der Server von dem Spieler verlangt eine Stichanzahl anzugeben.

Parameter

<i>msgNumber</i>	Die Nachricht vom Server
------------------	--------------------------

8.14.2.5 void Ruleset.ClientRuleset.resolveMessage (**MsgSelectionRequest** *msgSelection*)

Verarbeitet die RulesetMessage dass der Server von dem Spieler verlangt eine Farbe auszuwählen.

Parameter

<i>msgSelection</i>	Die Nachricht vom Server
---------------------	--------------------------

8.14.2.6 void Ruleset.ClientRuleset.send (**RulesetMessage** *message*) [protected]

Schickt eine Nachricht übers Model an den Server.

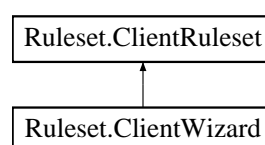
Parameter

<i>message</i>	Die Nachricht
----------------	---------------

8.15 Client.ClientState Enum-Referenz

8.16 Ruleset.ClientWizard Klassenreferenz

Klassendiagramm für Ruleset.ClientWizard:



Öffentliche Methoden

- boolean [isValidMove](#) ([Card](#) card)

Weitere Geerbte Elemente

8.16.1 Dokumentation der Elementfunktionen

8.16.1.1 boolean Ruleset.ClientWizard.isValidMove ([Card](#) card) [virtual]

Prüft ob ein gemachter Zug zum Spiel Wizard gültig ist.

Rückgabe

isValid true falls Zug gültig, false wenn nicht

Implementiert [Ruleset.ClientRuleset](#).

8.17 Ruleset.Colour Enum-Referenz

8.18 ComObjects.ComBeenKicked Klassenreferenz

Öffentliche Methoden

- [ComBeenKicked](#) (String message)
- String [getMessage](#) ()

8.18.1 Ausführliche Beschreibung

Die Nachricht wird an einen Spieler gesendet, wenn er aus einem Spiel entfernt wurde. Dies geschieht, wenn ein Spieler ein Spiel verlässt oder wenn der Spielleiter das Wartefenster verlässt.

8.18.2 Beschreibung der Konstruktoren und Destruktoren

8.18.2.1 ComObjects.ComBeenKicked.ComBeenKicked (String message)

Dies ist der Kontruktor für eine neue ComBeenKicked-Nachricht.

Parameter

<i>message</i>	ist die Nachricht.
----------------	--------------------

8.18.3 Dokumentation der Elementfunktionen

8.18.3.1 String ComObjects.ComBeenKicked.getMessage ()

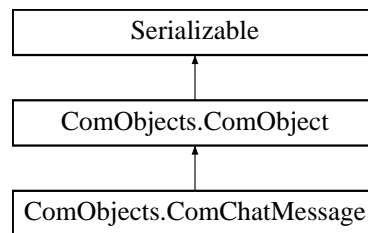
Diese Methode liefert die Nachricht, die an den Spieler gesendet wird, wenn er entfernt wird.

Rückgabe

die Nachricht.

8.19 ComObjects.ComChatMessage Klassenreferenz

Klassendiagramm für ComObjects.ComChatMessage:



Öffentliche Methoden

- [ComChatMessage](#) (String message)
- String [getChatMessage](#) ()

8.19.1 Ausführliche Beschreibung

Sie enthält eine Chatnachricht in Form eines Strings.

8.19.2 Beschreibung der Konstruktoren und Destruktoren

8.19.2.1 ComObjects.ComChatMessage.ComChatMessage (String message)

Dies ist der Kontruktor für eine neue ComChatMessage-Nachricht.

Parameter

<i>message</i>	ist die Chatnachricht, die versendet wird.
----------------	--

8.19.3 Dokumentation der Elementfunktionen

8.19.3.1 String ComObjects.ComChatMessage.getChatMessage ()

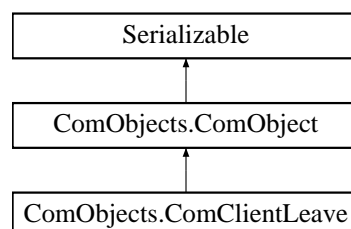
Hier kann die versendete Nachricht von anderen Klassen ausgelesen werden.

Rückgabe

die Chatnachricht, die versendet wurde.

8.20 ComObjects.ComClientLeave Klassenreferenz

Klassendiagramm für ComObjects.ComClientLeave:

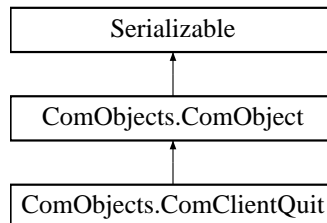


8.20.1 Ausführliche Beschreibung

Sie wird zur Benachrichtigung gesendet, wenn ein Spieler ins nächste Fenster möchte und aus dem alten entfernt werden soll.

8.21 ComObjects.ComClientQuit Klassenreferenz

Klassendiagramm für ComObjects.ComClientQuit:

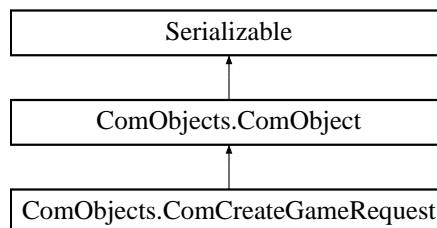


8.21.1 Ausführliche Beschreibung

Die Nachricht wird verschickt, wenn der Spieler ein Fenster schließt.

8.22 ComObjects.ComCreateGameRequest Klassenreferenz

Klassendiagramm für ComObjects.ComCreateGameRequest:



Öffentliche Methoden

- [ComCreateGameRequest](#) (String name, Enum ruleset, boolean hasPassword, String password)
- String [getGameName](#) ()
- Enum [getRuleset](#) ()
- boolean [hasPassword](#) ()
- String [getPassword](#) ()

8.22.1 Ausführliche Beschreibung

Diese Nachricht wird versendet, wenn ein neues Spiel erstellt werden soll.

8.22.2 Beschreibung der Konstruktoren und Destruktoren

8.22.2.1 ComObjects.ComCreateGameRequest.ComCreateGameRequest (String name, Enum ruleset, boolean hasPassword, String password)

Dies ist der Kontruktor für eine neue ComCreateGameRequest-Nachricht.

Parameter

<i>name</i>	ist der Name des Spiels.
<i>ruleset</i>	ist die der Spieltyp, der erstellt werden soll.
<i>hasPassword</i>	sagt, ob ein Passwort gesetzt wurde.
<i>password</i>	ist das Passwort, das gesetzt wurde.

Benutzt ComObjects.ComCreateGameRequest.hasPassword().

8.22.3 Dokumentation der Elementfunktionen

8.22.3.1 String ComObjects.ComCreateGameRequest.getGameName ()

Diese Methode gibt den Namen des Spiels zurück.

Rückgabe

den Spielnamen.

8.22.3.2 String ComObjects.ComCreateGameRequest.getPassword ()

Gibt das Passwort zurück.

Sollte keines gesetzt sein, wird null zurück gegeben.

Rückgabe

das Passwort.

8.22.3.3 Enum ComObjects.ComCreateGameRequest.getRuleset ()

Diese Methode gibt das Regelwerk zurück, das benutzt werden soll.

Rückgabe

das Regelwerk, welches benutzt wird.

8.22.3.4 boolean ComObjects.ComCreateGameRequest.hasPassword ()

Diese Methode gibt an, ob eine Passwort für ein Spiel gesetzt wurde.

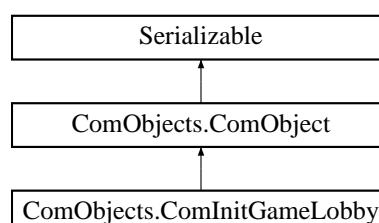
Rückgabe

ob es ein Passwort gibt.

Wird benutzt von ComObjects.ComCreateGameRequest.ComCreateGameRequest().

8.23 ComObjects.ComInitGameLobby Klassenreferenz

Klassendiagramm für ComObjects.ComInitGameLobby:



Öffentliche Methoden

- [ComInitGameLobby](#) (List *playerList*)
- Object [getPlayerList](#) ()

8.23.1 Ausführliche Beschreibung

Sie liefert die Liste der Spieler, die sich bereits beim Betreten des Wartefensters darin befinden.

8.23.2 Beschreibung der Konstruktoren und Destruktoren

8.23.2.1 ComObjects.ComInitGameLobby.ComInitGameLobby (List *playerList*)

Dies ist der Kontruktor für eine neue ComInitGameLobby-Nachricht.

Parameter

<i>playerList</i>	ist die Liste aller Player, die sich im Wartefenster befinden.
-------------------	--

8.23.3 Dokumentation der Elementfunktionen

8.23.3.1 Object ComObjects.ComInitGameLobby.getPlayerList ()

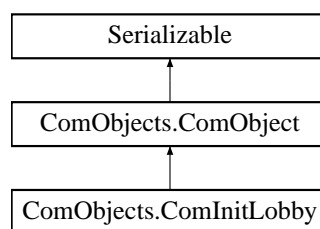
Diese Methode gibt die Liste der Player zurück, die sich momentan inm Wartefenster befinden.

Rückgabe

die Liste der Spieler.

8.24 ComObjects.ComInitLobby Klassenreferenz

Klassendiagramm für ComObjects.ComInitLobby:



Öffentliche Methoden

- [ComInitLobby](#) (List *playerList*, Set *gameList*)
- List [getPlayerList](#) ()
- Set< [GameServerRepresentation](#) > [getGameList](#) ()

8.24.1 Ausführliche Beschreibung

Sie synchronisiert den Client mit der Lobby, wenn er sich mit dem Server verbindet oder nach einem Spiel in die Lobby zurückkehrt. Dazu enthält sie sowohl die *playerList*, als auch die *gameList*.

8.24.2 Beschreibung der Konstruktoren und Destrukturen

8.24.2.1 ComObjects.ComInitLobby.ComInitLobby (List *playerList*, Set *gameList*)

Dies ist der Kontruktor für eine neue ComInitLobby-Nachricht.

Parameter

<i>playerList</i>	ist die Liste der Spieler, die sich in der Lobby befinden.
<i>gameList</i>	ist die Liste der Spiele, die existieren und in der Lobby angezeigt werden.

8.24.3 Dokumentation der Elementfunktionen**8.24.3.1 Set<GameServerRepresentation> ComObjects.ComInitLobby.getGameList ()**

Diese Methode liefert eine Liste aller Spiele, die erstellt wurden, damit sie in der Lobby angezeigt werden können.

Rückgabe

die Liste der Spiele.

8.24.3.2 List ComObjects.ComInitLobby.getPlayerList ()

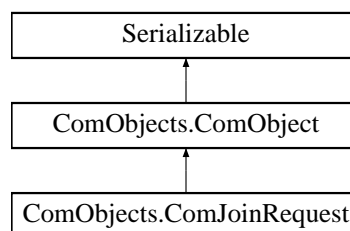
Die Methode liefert die Liste aller Spieler, die in der Lobby sind.

Rückgabe

die Liste der Spieler.

8.25 ComObjects.ComJoinRequest Klassenreferenz

Klassendiagramm für ComObjects.ComJoinRequest:

**Öffentliche Methoden**

- [ComJoinRequest](#) (String gameMasterName)
- String [getGameMasterName](#) ()

8.25.1 Ausführliche Beschreibung

Sie ist eine Nachricht, die an den Server gesendet wird, wenn der Spieler einem bestimmten Spiel beitreten will. Dazu enthält es den Namen des Spielleiters als String.

8.25.2 Beschreibung der Konstruktoren und Destruktoren**8.25.2.1 ComObjects.ComJoinRequest.ComJoinRequest (String gameMasterName)**

Dies ist der Kontruktor für eine neue ConJoinRequest-Nachricht.

Ein Spiel kann durch den eindeutigen Namen der Spielleiters identifiziert werden.

Parameter

<i>gameMaster-Name</i>	ist der Name der Spielleiters.
------------------------	--------------------------------

8.25.3 Dokumentation der Elementfunktionen

8.25.3.1 String ComObjects.ComJoinRequest.getGameMasterName ()

Diese Methode gibt den Namen des Spielleiters zurück.

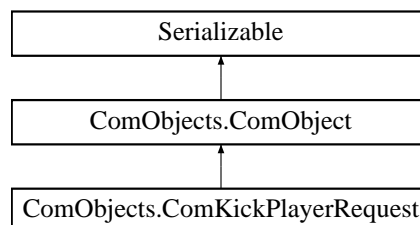
Dieser ist eindeutig, so kann ein bestimmtes Spiel identifiziert werden.

Rückgabe

den Namen des Spielleiters.

8.26 ComObjects.ComKickPlayerRequest Klassenreferenz

Klassendiagramm für ComObjects.ComKickPlayerRequest:



Öffentliche Methoden

- [ComKickPlayerRequest](#) (String playerName)
- String [getPlayerName](#) ()

8.26.1 Ausführliche Beschreibung

Sie ist eine Nachricht an den Server, die angibt einen Spieler vom Spiel zu entfernen. Dazu enthält es einen String, der den Namen des Spielers enthält.

8.26.2 Beschreibung der Konstruktoren und Destruktoren

8.26.2.1 ComObjects.ComKickPlayerRequest.ComKickPlayerRequest (String playerName)

Dies ist der Kontruktor für eine neue ComKickPlayerRequest-Nachricht.

Diese enthält den Namen des Spielers, der aus den Spiel gelöscht werden soll.

Parameter

<i>playerName</i>	ist der Name des Spielers.
-------------------	----------------------------

8.26.3 Dokumentation der Elementfunktionen

8.26.3.1 String ComObjects.ComKickPlayerRequest.getPlayerName ()

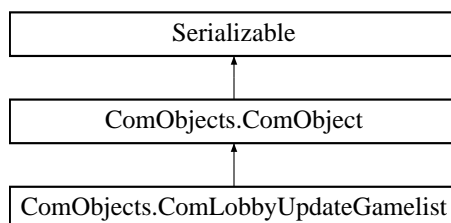
Diese Methode liefert den Namen des Spielers, der aus dem Spiel entfernt werden soll.

Rückgabe

den Spielernamen.

8.27 ComObjects.ComLobbyUpdateGamelist Klassenreferenz

Klassendiagramm für ComObjects.ComLobbyUpdateGamelist:

**Öffentliche Methoden**

- [ComLobbyUpdateGamelist](#) (boolean removeFlag, [GameServerRepresentation](#) gameServer)
- boolean [isRemoveFlag](#) ()
- [GameServerRepresentation](#) [getGameServer](#) ()

8.27.1 Ausführliche Beschreibung

Sie aktualisiert die Gameliste in der Lobby. Dazu enthält sie den GameServer und ein RemoveFlag.

8.27.2 Beschreibung der Konstruktoren und Destruktoren**8.27.2.1 ComObjects.ComLobbyUpdateGamelist.ComLobbyUpdateGamelist (boolean removeFlag, GameServerRepresentation gameServer)**

Dies ist der Kontruktor für eine neue ComLobbyUpdateGamelist-Nachricht.

Parameter

<i>removeFlag</i>	zeigt an, ob das Spiel gelöscht werden soll.
<i>gameServer</i>	ist das Spiel.

8.27.3 Dokumentation der Elementfunktionen**8.27.3.1 GameServerRepresentation ComObjects.ComLobbyUpdateGamelist.getGameServer ()**

Diese Methode liefert das Spiel, das geupdated werden soll.

Rückgabe

das Spiel.

8.27.3.2 boolean ComObjects.ComLobbyUpdateGamelist.isRemoveFlag ()

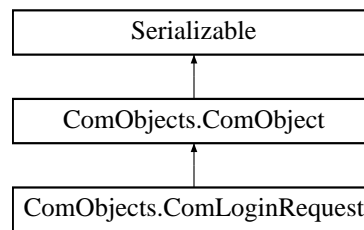
Diese Methode liefert, ob ein Spiel gelöscht werden soll oder nicht.

Rückgabe

ob das Spiel gelöscht wird.

8.28 ComObjects.ComLoginRequest Klassenreferenz

Klassendiagramm für ComObjects.ComLoginRequest:



Öffentliche Methoden

- [ComLoginRequest](#) (String name)
- String [getPlayerName](#) ()

8.28.1 Ausführliche Beschreibung

Sie ist eine Nachricht, die beim Login an den Server gesendet wird. Dazu enthält sie den Namen des Spielers, der sich einloggen möchte.

8.28.2 Beschreibung der Konstruktoren und Destruktoren

8.28.2.1 ComObjects.ComLoginRequest.ComLoginRequest (String name)

Dies ist der Kontruktor für eine neue ComLoginRequest-Nachricht.

Parameter

<i>name</i>	ist der Name des Spielers, des sich einloggen möchte.
-------------	---

8.28.3 Dokumentation der Elementfunktionen

8.28.3.1 String ComObjects.ComLoginRequest.getPlayerName ()

Diese Methode liefert den Namen des Spielers, des sich einloggen möchte.

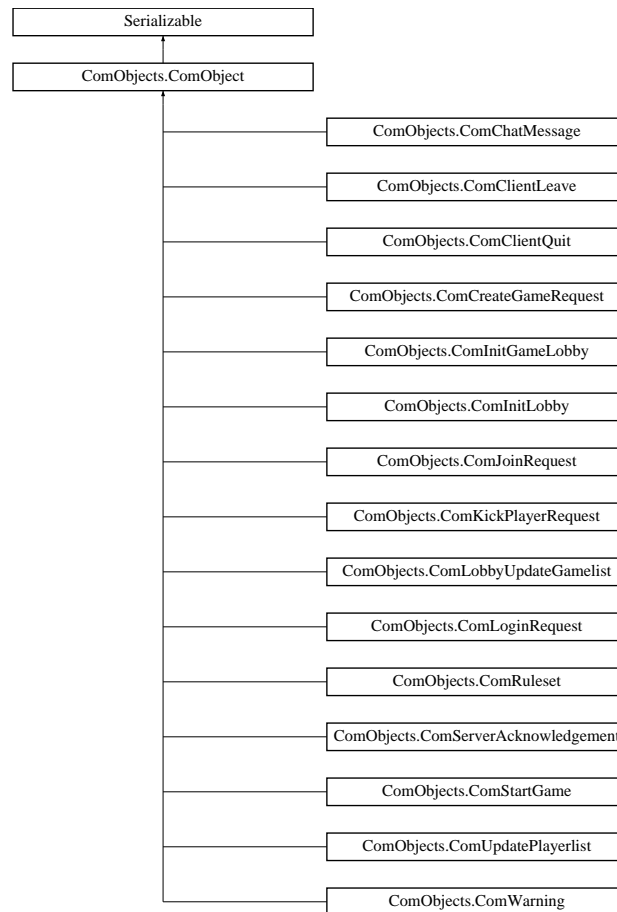
Dieser muss auf Eindeutigkeit geprüft werden.

Rückgabe

den Spielernamen.

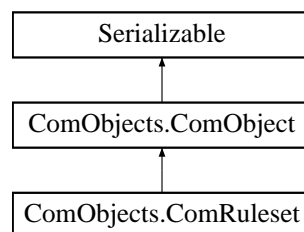
8.29 ComObjects.ComObject Klassenreferenz

Klassendiagramm für ComObjects.ComObject:



8.30 ComObjects.ComRuleset Klassenreferenz

Klassendiagramm für ComObjects.ComRuleset:



Öffentliche Methoden

- [ComRuleset](#) ([RulesetMessage](#) rulesetMessage)
- [RulesetMessage](#) getRulesetMessage ()

8.30.1 Ausführliche Beschreibung

Sie ist die grundlegende Nachricht eines Regelwerkaufrufes und enthält eine verfeinerte Nachricht mit weiteren Informationen, die [RulesetMessage](#).

8.30.2 Beschreibung der Konstruktoren und Destruktoren

8.30.2.1 ComObjects.ComRuleset.ComRuleset (RulesetMessage *rulesetMessage*)

Dies ist der Kontruktor für eine neue ComResult-Nachricht.

Parameter

<i>rulesetMessage</i>	ist eine Nachricht, die ans Ruleset gesendet werden soll.
-----------------------	---

8.30.3 Dokumentation der Elementfunktionen**8.30.3.1 RulesetMessage ComObjects.ComRuleset.getRulesetMessage ()**

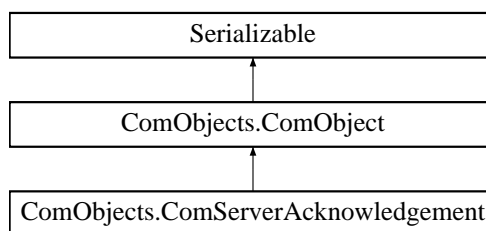
Diese Methode gibt die Nachricht zurück, die ans Ruleset gesendet werden soll.

Rückgabe

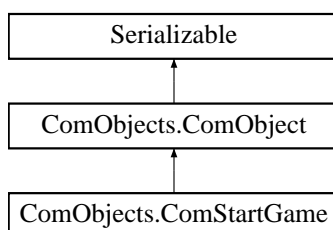
die Nachricht.

8.31 ComObjects.ComServerAcknowledgement Klassenreferenz

Klassendiagramm für ComObjects.ComServerAcknowledgement:

**8.32 ComObjects.ComStartGame Klassenreferenz**

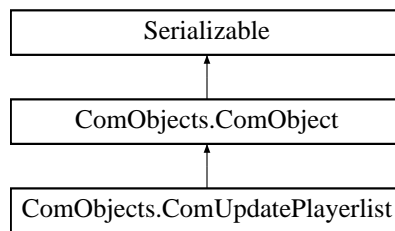
Klassendiagramm für ComObjects.ComStartGame:

**8.32.1 Ausführliche Beschreibung**

Sie wird versendet, wenn ein Spiel gestartet werden soll.

8.33 ComObjects.ComUpdatePlayerlist Klassenreferenz

Klassendiagramm für ComObjects.ComUpdatePlayerlist:



Öffentliche Methoden

- [ComUpdatePlayerlist](#) (String playerName, boolean removeFlag)
- String [getPlayerName](#) ()
- boolean [isRemoveFlag](#) ()

8.33.1 Ausführliche Beschreibung

Sie sendet eine Nachricht zum Update der Playerliste in der Lobby und Spiellobby. Dazu enthält sie den Player und ein removeFlag.

8.33.2 Beschreibung der Konstruktoren und Destruktoren

8.33.2.1 ComObjects.ComUpdatePlayerlist.ComUpdatePlayerlist (String playerName, boolean removeFlag)

Dies ist der Kontruktor für eine neue ComUpdatePlayerlist-Nachricht.

Diese beinhaltet den Namen des Spielers und die Angabe ob er gelöscht werden soll.

Parameter

<i>playerName</i>	ist der Name der Spielers.
<i>removeFlag</i>	zeigt, ob der Spieler gelöscht werden soll.

8.33.3 Dokumentation der Elementfunktionen

8.33.3.1 String ComObjects.ComUpdatePlayerlist.getPlayerName ()

Diese Methode gibt den Namen des Spielers zurück.

Rückgabe

den Spielernamen.

8.33.3.2 boolean ComObjects.ComUpdatePlayerlist.isRemoveFlag ()

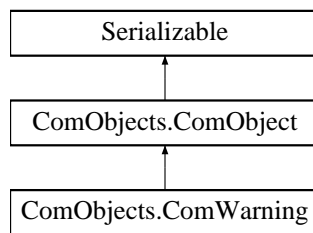
Diese Methode gibt zurück, ob der Spieler aus der Liste gelöscht werden soll oder nicht.

Rückgabe

ob der Spieler gelöscht werden soll.

8.34 ComObjects.ComWarning Klassenreferenz

Klassendiagramm für ComObjects.ComWarning:



Öffentliche Methoden

- [ComWarning](#) (String warning)
- String [getWarning](#) ()

8.34.1 Ausführliche Beschreibung

Sie soll dem Spieler eine Mitteilung senden und so über ein Fehlerevent informieren.

8.34.2 Beschreibung der Konstruktoren und Destruktoren

8.34.2.1 ComObjects.ComWarning.ComWarning (String warning)

Dies ist der Konstruktor einer neuen ComWarning-Nachricht.

Er enthält eine Warnung an den Spieler, wenn ein Fehler passiert.

Parameter

<i>warning</i>	ist die Warnung, die der Spieler erhält.
----------------	--

8.34.3 Dokumentation der Elementfunktionen

8.34.3.1 String ComObjects.ComWarning.getWarning ()

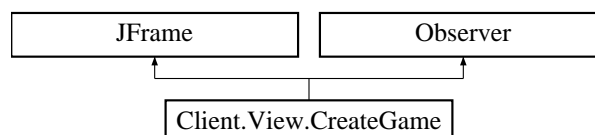
Diese Methode gibt die Nachricht zurück, die dem Spieler den Fehler mitteilt.

Rückgabe

die Warnnachricht.

8.35 Client.View.CreateGame Klassenreferenz

Klassendiagramm für Client.View.CreateGame:



Öffentliche Methoden

- [CreateGame](#) () throws IOException
- void [update](#) (Observable o, Object arg)

8.35.1 Ausführliche Beschreibung

Es bietet alle Komponenten, um ein Regelwerk zu wählen, einen Spielnamen festzulegen und das Spiel durch ein Passwort zu schätzen. In der Spielerstellung wird ein Titelbild des ausgewählten Spiels und eine kurze Beschreibung angezeigt. über 'Leave' kehrt der Spieler in die [Lobby](#) zurück und mit 'Create' wird das Spiel erstellt.

Autor

M4nkey

8.35.2 Beschreibung der Konstruktoren und Destruktoren

8.35.2.1 Client.View.CreateGame.CreateGame () throws IOException

Erstellt das [CreateGame](#) Fenster.

Ausnahmebehandlung

<i>IOException</i>	
--------------------	--

8.35.3 Dokumentation der Elementfunktionen

8.35.3.1 void Client.View.CreateGame.update (Observable o, Object arg)

Wird durch notify() im [ClientModel](#) aufgerufen.

Je nach dem in arg übergebenen Befehl wird ein Update des Fensters ausgeführt oder eine Fehlermeldung angezeigt.

Parameter

<i>o</i>	erwartet ein Objekt von der Klasse ClientModel
<i>arg</i>	erwartet: windowChangeAcknowledged, windowChangeDenied

8.36 Client.View.DiscardPile Klassenreferenz

8.36.1 Ausführliche Beschreibung

Autor

m4nkey

8.37 Client.View.DrawDeck Klassenreferenz

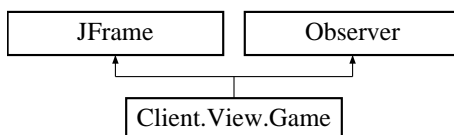
8.37.1 Ausführliche Beschreibung

Autor

m4nkey

8.38 Client.View.Game Klassenreferenz

Klassendiagramm für Client.View.Game:



Öffentliche Methoden

- [Game](#) () throws IOException
- void [update](#) (Observable o, Object arg)

8.38.1 Ausführliche Beschreibung

Außerdem können über ein Dropdown-Menü Änderungen an Hintergrundbild und Kartenhintergründen vorgenommen werden. Schließen beendet das Spiel und der Spieler wird in die [Lobby](#) zurückgeleitet.

Autor

M4nkey

8.38.2 Beschreibung der Konstruktoren und Destruktoren

8.38.2.1 Client.View.Game.Game () throws IOException

Erstellt das [Game](#) Fenster.

Ausnahmebehandlung

<i>IOException</i>	
--------------------	--

8.38.3 Dokumentation der Elementfunktionen

8.38.3.1 void Client.View.Game.update (Observable o, Object arg)

Wird durch notify() im [ClientModel](#) aufgerufen.

Je nach dem in arg übergebenen Befehl wird ein Update des Fensters ausgeführt oder eine Fehlermeldung angezeigt.

Parameter

<i>o</i>	erwartet ein Objekt von der Klasse ClientModel
<i>arg</i>	erwartet: chatMessage, playedCardsUpdate, otherDataUpdate

8.39 Ruleset.GameClientUpdate Klassenreferenz

Öffentliche Methoden

- ArrayList< [Card](#) > [getOwnHand](#) ()
- Map< String, [Card](#) > [getPlayedCards](#) ()
- [OtherData](#) [getOwnData](#) ()
- ArrayList< [OtherData](#) > [getOtherPlayerData](#) ()

8.39.1 Ausführliche Beschreibung

Das wären seine Spielhand, der Ablagestapel sowie die Otherdata von allen Spielern. Bei Wizard enthält es auch die momentane Trumpfkarte.

8.39.2 Dokumentation der Elementfunktionen

8.39.2.1 `ArrayList<OtherData> Ruleset.GameClientUpdate.getOtherPlayerData ()`

Holt die Spieldaten der anderen Spieler.

Rückgabe

`otherPlayerData` Die Spieldaten der anderen Spieler

8.39.2.2 `OtherData Ruleset.GameClientUpdate.getOwnData ()`

Holt die zusätzlichen Spieldaten des Client.

Rückgabe

`ownData` Die Spieldaten des Clients

8.39.2.3 `ArrayList<Card> Ruleset.GameClientUpdate.getOwnHand ()`

Holt die Karten die der Client auf der Hand hat.

Rückgabe

`ownHand` Die Hand des Clients

8.39.2.4 `Map<String, Card> Ruleset.GameClientUpdate.getPlayedCards ()`

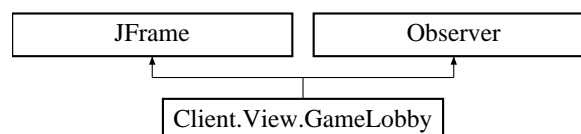
Holt die gespielten Karten auf dem Ablagestapel.

Rückgabe

`discardPile` Die gespielten Karten

8.40 Client.View.GameLobby Klassenreferenz

Klassendiagramm für Client.View.GameLobby:



Öffentliche Methoden

- void `update` (Observable o, Object arg)

8.40.1 Ausführliche Beschreibung

Der Spielleiter kann Spieler mit dem Remove Player Button entfernen. Über Leave kehren die Spieler in die [Lobby](#) zurück. Der spielinterne Chat ist ab hier verfügbar.

Autor

M4nkey

8.40.2 Dokumentation der Elementfunktionen

8.40.2.1 void Client.View.GameLobby.update (Observable o, Object arg)

Wird durch notify() im [ClientModel](#) aufgerufen.

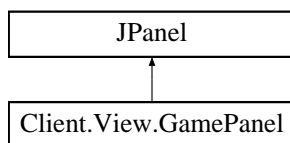
Je nach dem in arg übergebenen Befehl wird ein Update des Fensters ausgeführt oder eine Fehlermeldung angezeigt.

Parameter

<i>o</i>	erwartet ein Objekt von der Klasse ClientModel
<i>arg</i>	erwartet: windowChangeAcknowledged, windowChangeDenied, playerListUpdate, windowChangeForced, chatMessage

8.41 Client.View.GamePanel Klassenreferenz

Klassendiagramm für Client.View.GamePanel:



8.41.1 Ausführliche Beschreibung

Es besteht aus verschiedenen Panelobjekten, welche je nach Regelwerk auf das Spielfeld gezeichnet werden. Dazu gehören die eigenen Karten, eventuell ausgewählte Karten, ein Textfeld z.B. zur Anzeige der Anzahl der restlichen Karten der Mitspieler und den Ablagestapel (/L194/). Nach jeder Runde wird der Punktestand aktualisiert.

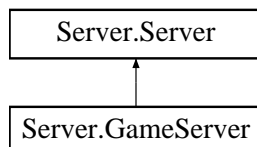
Autor

m4nkey

8.42 Ruleset.GamePhase Enum-Referenz

8.43 Server.GameServer Klassenreferenz

Klassendiagramm für Server.GameServer:



Öffentliche Methoden

- **GameServer** (**LobbyServer** server, **Player** gameMaster, String GameName, **RulesetType** ruleset, String password, boolean hasPassword)
- synchronized void **addPlayer** (**Player** player)
- synchronized void **removePlayer** (**Player** player)
- void **receiveMessage** (**Player** player, ComKickPlayerRequest kickPlayer)
- void **receiveMessage** (**Player** player, ComChatMessage chat)
- void **receiveMessage** (**Player** player, ComClientLeave leave)
- void **receiveMessage** (**Player** player, ComClientQuit quit)
- void **receiveMessage** (**Player** player, ComStartGame start)
- void **receiveMessage** (**Player** player, ComRuleset ruleset)
- ComInitGameLobby **initLobby** ()

8.43.1 Ausführliche Beschreibung

Sie verwaltet die Kommunikation zwischen den Clients während eines Spieles. Die GameServer-Klasse erbt Methoden zur Kommunikation vom **Server**. Der **GameServer** tauscht Nachrichten zwischen Ruleset und **Player** aus, um so den Spielablauf zu koordinieren.

Autor

Viktoria

8.43.2 Beschreibung der Konstruktoren und Destruktoren

8.43.2.1 **Server.GameServer.GameServer (LobbyServer server, Player gameMaster, String GameName, RulesetType ruleset, String password, boolean hasPassword)**

Konstruktor des GameServers.

Setzt die Attribute lobbyServer, name, password, hasPasword und rulesetType auf die übergebenen Werte. Setzt den gameMasterName auf den Namen des gameMaster und fügt den gameMaster dem Set an Spielern hinzu. Bestimmt mithilfe des Enums RulesetType das Ruleset und erstellt es. Setzt currentPlayers auf eins und maxPlayers je nach Ruleset.

Parameter

<i>server</i>	ist der LobbyServer der den GameServer erstellt hat.
<i>gameMaster</i>	ist der Name des Spielleiters
<i>GameName</i>	ist der Name des Spiels
<i>ruleset</i>	gibt an, welches Ruleset verwendet wird
<i>password</i>	speichert das Passwort des Spiels
<i>hasPassword</i>	gibt an, ob das Spiel ein Passwort hat

8.43.3 Dokumentation der Elementfunktionen

8.43.3.1 synchronized void **Server.GameServer.addPlayer (Player player)**

Diese Methode wird vom abstrakten **Server** vererbt.

Zusätzlich wird die Zahl der currentPlayers um eins Erhöht.

Parameter

<i>player</i>	ist der Player , der hinzugefügt wird
---------------	---

8.43.3.2 ComInitGameLobby Server.GameServer.initLobby ()

Baut ein neues ComInitGameLobby Objekt und gibt es zurück.

Rückgabe

Gibt das ComInitGameLobby Objekt zurück

8.43.3.3 void Server.GameServer.receiveMessage ([Player](#) *player*, ComKickPlayerRequest *kickPlayer*)

Diese Methode ist dafür zuständig zu ermitteln, was passiert wenn ein Spieler aus der GameLobby geworfen wird.

Der [Player](#) wird durch Aufruf von changeServer an die Lobby zurückgegeben. An diesen Spieler wird ein ComWarning und ein ComInitLobby geschickt. Danach wird ein ComUpdatePlayerlist Objekt mit broadcast an alle Client im Spiel verschickt.

Parameter

<i>player</i>	ist der Threat der die Nachricht erhalten hat
<i>kicked</i>	ist das ComObject, das verarbeitet wird

8.43.3.4 void Server.GameServer.receiveMessage ([Player](#) *player*, ComChatMessage *chat*)

Diese Methode ist dafür zuständig eine Chatnachricht an alle Clients im Spiel zu verschicken.

Dafür wird die ComChatMessage mit broadcast an alle Spieler im playerSet verteilt.

Parameter

<i>player</i>	ist der Threat der die Nachricht erhalten hat
<i>chat</i>	ist das ComObject, das die Chatnachricht enthält

8.43.3.5 void Server.GameServer.receiveMessage ([Player](#) *player*, ComClientLeave *leave*)

Diese Methode gibt einen [Player](#), der die GameLobby verlassen will, durch Aufruf von changeServer an die ServerLobby zurück und schickt ihm ein ComInitLobby.

Danach wird ein ComUpdatePlayerlist Objekt mit broadcast an alle Clients im Spiel verschickt.

Parameter

<i>player</i>	ist der Threat der die Nachricht erhalten hat
<i>leave</i>	ist das ComObject, welches angibt, dass der Spieler in die Lobby zurückkehrt

8.43.3.6 void Server.GameServer.receiveMessage ([Player](#) *player*, ComClientQuit *quit*)

Diese Methode behandelt den Fall, dass ein Spieler das laufende Spiel verlässt.

Sie gibt einen [Player](#), der das Spiel verlassen will, Aufruf von changeServer an die ServerLobby zurück und schickt ihm ein ComInitLobby. Alle Spieler, die sich im Spiel befinden bekommen ein ComWarning und ein ComInitLobby und werden durch Aufruf von changeServer an die Lobby zurückgegeben. Das Spiel wird aufgelöst.

Parameter

<i>player</i>	ist der Threat der die Nachricht erhalten hat
---------------	---

<i>quit</i>	ist das ComObject, welches angibt, dass der Spieler das Spiel verlässt
-------------	--

8.43.3.7 void Server.GameServer.receiveMessage (Player *player*, ComStartGame *start*)

Diese Methode sagt dem Ruleset, dass ein neues Spiel gestartet werden soll.

Parameter

<i>player</i>	ist der Threat der die Nachricht erhalten hat
<i>start</i>	ist das ComObject, dass angibt, dass das Spiel gestartet werden soll

8.43.3.8 void Server.GameServer.receiveMessage (Player *player*, ComRuleset *ruleset*)

Diese Methode gibt das erhaltene ComRuleset durch einen Aufruf von resolveMessage an das Ruleset weiter.

Parameter

<i>player</i>	ist der Threat der die Nachricht erhalten hat
<i>ruleset</i>	ist das ComObject, das zeigt, dass das Object vom Ruleset bearbeitet werden muss

8.43.3.9 synchronized void Server.GameServer.removePlayer (Player *player*)

Diese Methode wird vom abstrakten [Server](#) vererbt.

Zusätzlich wird die Zahl der currentPlayers um eins Verringert.

Parameter

<i>player</i>	ist der Player , der entfernt wird
---------------	--

8.44 Server.GameServerRepresentation Klassenreferenz

Öffentliche Methoden

- [GameServerRepresentation](#) (String gameMaster, String gameName, int max, int current, [RulesetType](#) type, boolean password)

8.44.1 Ausführliche Beschreibung

Sie wird dem ComObjekt ComLobbyUpdateGameList angehängt, um die Spielliste in der GameLobby aktualisieren zu können

Autor

Viktoria

8.44.2 Beschreibung der Konstruktoren und Destruktoren

8.44.2.1 Server.GameServerRepresentation.GameServerRepresentation (String *gameMaster*, String *gameName*, int *max*, int *current*, RulesetType *type*, boolean *password*)

Der Konstruktor der Klasse [GameServerRepresentation](#) initialisiert die Attribute mit den vom [GameServer](#) übergebenen Werten.

Parameter

<i>gameMaster</i>	der Name des Spielleiters
<i>gameName</i>	der Name des Spiels
<i>max</i>	Maximal mögliche Anzahl teilnehmender Spieler
<i>current</i>	Anzahl momentaner Spieler
<i>type</i>	Welches Ruleset verwendet wird
<i>password</i>	ob das Spiel ein Passwort hat

8.45 Ruleset.GameState Klassenreferenz

Öffentliche Methoden

- void [setCurrentPlayer](#) ([PlayerState](#) player)
- [PlayerState](#) [getCurrentPlayer](#) ()
- ArrayList< [Card](#) > [getCardsLeftInDeck](#) ()
- Map< String, [Card](#) > [getPlayedCards](#) ()
- [PlayerState](#) [getPlayer](#) (String name)
- void [setTrumpCard](#) ([Card](#) trumpCard)
- [Card](#) [getTrumpCard](#) ()
- int [getNumberOfPlayedCards](#) ()
- boolean [playCard](#) ([Card](#) card)

8.45.1 Ausführliche Beschreibung

Es enthält die einzelnen PlayerStates, sowie Informationen zum Ablage-, Aufnahmestapel, Rundenanzahl, den momentan aktiven Spieler sowie [GamePhase](#).

8.45.2 Dokumentation der Elementfunktionen

8.45.2.1 ArrayList<Card> Ruleset.GameState.getCardsLeftInDeck ()

Holt die Karten die noch im Aufnahmestapel sind.

Rückgabe

`cardsLeftInDeck` Holt die Karten die noch im Aufnahmestapel sind

8.45.2.2 PlayerState Ruleset.GameState.getCurrentPlayer ()

Holt den Spieler der momentan am Zug ist.

Rückgabe

`currentPlayer` Der Spielzustand des Spielers der grad am Zug ist

8.45.2.3 int Ruleset.GameState.getNumberOfPlayedCards ()

Holt die Anzahl der gespielten Karten.

Rückgabe

Die Anzahl der gespielten Karten

8.45.2.4 Map<String,Card> Ruleset.GameState.getPlayedCards ()

Holt die gespielten Karten im Ablagestapel.

Rückgabe

discardPile Die gespielten Karten

8.45.2.5 PlayerState Ruleset.GameState.getPlayer (String name)

Holt einen bestimmten Spieler.

Parameter

<i>name</i>	Der Name des Spielers
-------------	-----------------------

Rückgabe

player Der Spielzustand des Spielers

8.45.2.6 Card Ruleset.GameState.getTrumpCard ()

Holt die momentane Trumpfkarte im Spiel.

Rückgabe

trumpCard Die momentane Trumpfkarte

8.45.2.7 boolean Ruleset.GameState.playCard (Card card)

Entfernt eine Karte aus der Hand des currentPlayer und legt sie auf dem Ablagestapel.

Parameter

<i>card</i>	Die gespielte Karte
-------------	---------------------

Rückgabe

isInHand Gibt true zurück wenn die gespielte Karte auf der Hand vom Spieler liegt und false sonst

8.45.2.8 void Ruleset.GameState.setCurrentPlayer (PlayerState player)

Setzt einen neuen Spieler als currentPlayer.

Parameter

<i>player</i>	Der neue currentPlayer
---------------	------------------------

8.45.2.9 void Ruleset.GameState.setTrumpCard (Card trumpCard)

Setzt die Trumpfkarte.

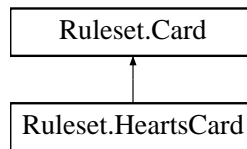
Parameter

<i>trumpCard</i>	Die Trumpfkarte
------------------	-----------------

8.46 Ruleset.HearthsDeck Klassenreferenz

8.47 Ruleset.HeartsCard Klassenreferenz

Klassendiagramm für Ruleset.HeartsCard:



Öffentliche Methoden

- `int getPoints ()`
- `HeartsID getID ()`

Weitere Geerbte Elemente

8.47.1 Dokumentation der Elementfunktionen

8.47.1.1 HeartsID Ruleset.HeartsCard.getID ()

Holt die ID der Karte.

Rückgabe

points Die ID der Karte

8.47.1.2 int Ruleset.HeartsCard.getPoints ()

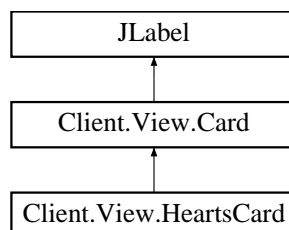
Holt die Punkte der Karte.

Rückgabe

points Die Punkte der Karte

8.48 Client.View.HeartsCard Klassenreferenz

Klassendiagramm für Client.View.HeartsCard:



Öffentliche Methoden

- `HeartsCard (HeartsID id)`
- `HeartsID getCardID ()`

8.48.1 Ausführliche Beschreibung

Sie wird verwendet um einzelne Karten auf das Spielfeld zu zeichnen. Dazu enthält sie die Pfadangabe zu dem Ordner, in dem die Bilder der Karten gespeichert sind, und eine ID, um das genaue Bild zu spezifizieren.

Autor

m4nkey

8.48.2 Beschreibung der Konstruktoren und Destruktoren

8.48.2.1 Client.View.HeartsCard.HeartsCard (HeartsID *id*)

Erstellt eine neue Hearts Karte für die Anzeige und zeichnet das Bild, das durch *id* spezifiziert ist.

Parameter

<i>id</i>	HeartsID der Karte
-----------	--------------------

8.48.3 Dokumentation der Elementfunktionen

8.48.3.1 HeartsID Client.View.HeartsCard.getCardID ()

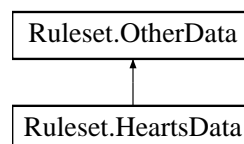
Gibt die HeartsID der Karte zurück

Rückgabe

HeartsID der Karte

8.49 Ruleset.HeartsData Klassenreferenz

Klassendiagramm für Ruleset.HeartsData:



Öffentliche Methoden

- int [getCompletePoints](#) ()
- int [getCurrentPoints](#) ()

8.49.1 Dokumentation der Elementfunktionen

8.49.1.1 int Ruleset.HeartsData.getCompletePoints ()

Holt den gesamten Punktestand eines Spielers.

Rückgabe

Der Gesamtpunktstand eines Spielers

8.49.1.2 `int Ruleset.HeartsData.getCurrentPoints ()`

Holt den momentanen Punktestand eines Spielers.

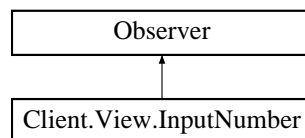
Rückgabe

Der momentane Punktestand eines Spielers

8.50 `Ruleset.HeartsID` Enum-Referenz

8.51 `Client.View.InputNumber` Klassenreferenz

Klassendiagramm für `Client.View.InputNumber`:



Öffentliche Methoden

- void `update` (`Observable o`, `Object arg`)

8.51.1 Ausführliche Beschreibung

Autor

m4nkey

8.51.2 Dokumentation der Elementfunktionen

8.51.2.1 `void Client.View.InputNumber.update (Observable o, Object arg)`

Wird durch `notify()` im `ClientModel` aufgerufen.

Je nach dem in `arg` übergebenen Befehl wird ein Update des Fensters ausgeführt oder eine Fehlermeldung angezeigt.

Parameter

<code>o</code>	erwartet ein Objekt von der Klasse <code>ClientModel</code>
<code>arg</code>	erwartet: <code>openInputNumber</code>

8.52 `Client.View.Language` Enum-Referenz

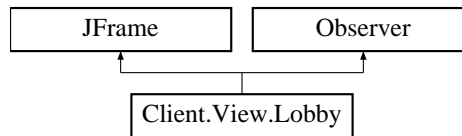
8.52.1 Ausführliche Beschreibung

Autor

m4nkey

8.53 `Client.View.Lobby` Klassenreferenz

Klassendiagramm für `Client.View.Lobby`:



Öffentliche Methoden

- void [update](#) (Observable o, Object arg)

8.53.1 Ausführliche Beschreibung

In der [Lobby](#) werden die Benutzernamen der sich in der [Lobby](#) befindenden Spieler, sowie offene Spiele angezeigt. In der [Lobby](#) können Chatnachrichten gesendet und empfangen werden. über 'Leave' verlässt der Spieler das Spiel. über 'Host [Game](#)' wird der Spieler zum CreateGame-Fenster weiter geleitet und mit 'Join [Game](#)' kann einem bereits erstellten Spiel beigetreten werden.

Autor

M4nkey

8.53.2 Dokumentation der Elementfunktionen

8.53.2.1 void Client.View.Lobby.update (Observable o, Object arg)

Wird durch notify() im [ClientModel](#) aufgerufen.

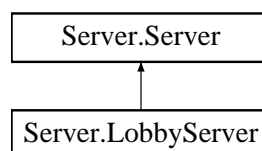
Je nach dem in arg übergebenen Befehl wird ein Update des Fensters ausgeführt oder eine Fehlermeldung angezeigt.

Parameter

<i>o</i>	erwartet ein Objekt von der Klasse ClientModel
<i>arg</i>	erwartet: windowChangeAcknowledged, windowChangeDenied, playerListUpdate, gameListUpdate, chatMessage

8.54 Server.LobbyServer Klassenreferenz

Klassendiagramm für Server.LobbyServer:



Klassen

- class [ClientListenerThread](#)

Öffentliche Methoden

- void [receiveMessage](#) ([Player](#) player, ComChatMessage chat)
- void [receiveMessage](#) ([Player](#) player, ComClientQuit quit)

- void [receiveMessage](#) ([Player](#) player, ComCreateGameRequest create)
- void [receiveMessage](#) ([Player](#) player, ComJoinRequest join)
- void [receiveMessage](#) ([Player](#) player, ComLoginRequest login)
- ComInitLobby [initLobby](#) ()

8.54.1 Ausführliche Beschreibung

Sie erstellt neue Spiele und verwaltet laufende Spiele. Auch wird der Chatverkehr über sie an die verbundenen Spieler weitergeleitet. Die LobbyServer-Klasse erbt Methoden zur Kommunikation vom [Server](#).

Autor

Viktoria

8.54.2 Dokumentation der Elementfunktionen

8.54.2.1 ComInitLobby Server.LobbyServer.initLobby ()

Baut ein neues ComInitLobby Objekt und gibt es zurück.

Rückgabe

Gibt das ComInitLobby Objekt zurück

8.54.2.2 void Server.LobbyServer.receiveMessage ([Player](#) *player*, ComChatMessage *chat*)

Diese Methode ist dafür zuständig eine Chatnachricht an alle Clients im Spiel zu verschicken.

Dafür wird die ComChatMessage mit broadcast an alle Spieler im playerSet verteilt.

Parameter

<i>player</i>	ist der Threat der die Nachricht erhalten hat
<i>chat</i>	ist das ComObject, das die Chatnachricht enthält

8.54.2.3 void Server.LobbyServer.receiveMessage ([Player](#) *player*, ComClientQuit *quit*)

Diese Methode schließt die Verbindung, der [Player](#) wird aus dem playerSet (bzw.

noNames Set) entfernt, der Name des Players wird aus dem Set names entfernt. War der Spieler im playerSet, wird ein ComUpdatePlayerlist mit broadcast an alle Clients verschickt.

Parameter

<i>player</i>	ist der Threat der die Nachricht erhalten hat
<i>quit</i>	ist das ComObject, welches angibt, dass der Spieler das Spiel vollständig verlässt

8.54.2.4 void Server.LobbyServer.receiveMessage ([Player](#) *player*, ComCreateGameRequest *create*)

Diese Methode erstellt einen neuen [GameServer](#) fügt ihm den [Player](#) hinzu.

Durch broadcast wird sowohl im [LobbyServer](#) als auch im [GameServer](#) ein ComUpdatePlayerlist verschickt. Zusätzlich wird dem Client mit sendToPlayer ein ComInitGameLobby geschickt.

Parameter

<i>player</i>	ist der Threat der die Nachricht erhalten hat
<i>create</i>	ist das ComObject, welches angibt, dass der Player ein neues Spiel erstellt hat

8.54.2.5 void Server.LobbyServer.receiveMessage ([Player](#) *player*, ComJoinRequest *join*)

Diese Methode fügt einen [Player](#) dem entsprechenden [GameServer](#) hinzu.

Falls das Passwort nicht leer ist wird geprüft, ob es mit dem Passwort des Spieles übereinstimmt, wenn nicht, wird ein ComWarning an den Client geschickt. Ansonsten wird und der [Player](#) dem, durch Namen des Spielers identifizierten, durch Aufruf von changeServer Gameserver übergeben. Durch broadcast wird sowohl im [Lobby-Server](#) als auch im [GameServer](#) ein ComUpdatePlayerlist verschickt. Zusätzlich wird dem joinendenClient mit send-ToPlayer ein ComInitGameLobby geschickt.

Parameter

<i>player</i>	ist der Threat der die Nachricht erhalten hat
<i>join</i>	ist das ComObject, welches angibt, dass der Player einem Spiel beitreten will

8.54.2.6 void Server.LobbyServer.receiveMessage ([Player](#) *player*, ComLoginRequest *login*)

Diese Methode überprüft, ob der Name im Set names vorhanden ist, falls ja, wird ein ComWarning an den Client geschickt, falls nein, wird im [Player](#) setName aufgerufen.

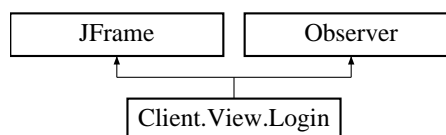
Der [Player](#) wird aus dem noNames Set entfernt und in das playerSet eingefügt. Der Name wird in das Set names eingefügt. Dem Client wird ein ComServerAcknowledgement geschickt.

Parameter

<i>player</i>	ist der Threat der die Nachricht erhalten hat
<i>login</i>	ist das ComObject, dass den Benutzernamen des Clients enthält

8.55 Client.View.Login Klassenreferenz

Klassendiagramm für Client.View.Login:



Öffentliche Methoden

- void [update](#) (Observable o, Object arg)

8.55.1 Ausführliche Beschreibung

In diesem Fenster kann der Benutzer seinen Namen und die Adresse des Servers eingeben. Außerdem ist über den [Login](#) die Auswahl der Sprache möglich. Über den Login-Button wird die Verbindung zum Server hergestellt.

Autor

M4nkey

8.55.2 Dokumentation der Elementfunktionen

8.55.2.1 void Client.View.Login.update (Observable o, Object arg)

Wird durch notify() im [ClientModel](#) aufgerufen.

Je nach dem in arg übergebenen Befehl wird ein Update des Fensters ausgeführt oder eine Fehlermeldung angezeigt.

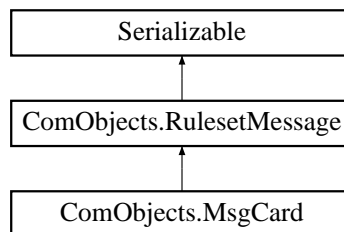
Parameter

<i>o</i>	erwartet ein Objekt von der Klasse ClientModel
<i>arg</i>	erwartet: windowChangeAcknowledged, windowChangeDenied

8.56 Client.MessageListenerThread Klassenreferenz

8.57 ComObjects.MsgCard Klassenreferenz

Klassendiagramm für ComObjects.MsgCard:



Öffentliche Methoden

- [MsgCard](#) ([Card](#) card)
- [Card](#) [getCard](#) ()

8.57.1 Ausführliche Beschreibung

Sie beinhaltet die ausgespielte Karte eines Spielers.

8.57.2 Beschreibung der Konstruktoren und Destruktoren

8.57.2.1 ComObjects.MsgCard.MsgCard (Card card)

Dies ist der Kontruktor für eine neue MsgCard-Nachricht.

Diese enthält die Information, welche Karte von einem Spieler gespielt wurde.

Parameter

<i>card</i>	ist die Karte.
-------------	----------------

8.57.3 Dokumentation der Elementfunktionen

8.57.3.1 Card ComObjects.MsgCard.getCard ()

Diese Methode gibt die ausgespielte Karte des Spielers zurück.

Rückgabe

die Karte.

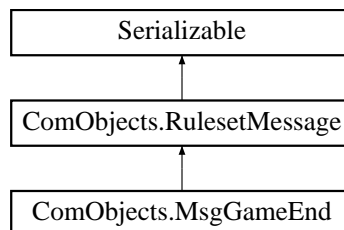
8.58 ComObjects.MsgCardRequest Klassenreferenz

8.58.1 Ausführliche Beschreibung

Diese Nachricht wird von Server gesendet, um einem Spieler mitzuteilen, dass er das Spielen einer Karte erwartet.

8.59 ComObjects.MsgGameEnd Klassenreferenz

Klassendiagramm für ComObjects.MsgGameEnd:

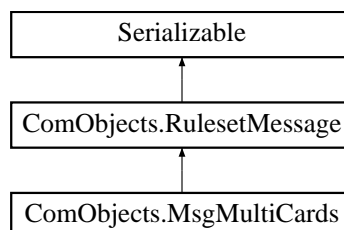


8.59.1 Ausführliche Beschreibung

Sie signalisiert dem ClientRuleset, dass das Spiel zu Ende ist.

8.60 ComObjects.MsgMultiCards Klassenreferenz

Klassendiagramm für ComObjects.MsgMultiCards:



Öffentliche Methoden

- [MsgMultiCards](#) (Set cardList)
- Set< [Card](#) > [getCardList](#) ()

8.60.1 Ausführliche Beschreibung

Sie liefert mehrere Karten zum Tausch für das Regelwerk Hearts.

8.60.2 Beschreibung der Konstruktoren und Destruktoren

8.60.2.1 ComObjects.MsgMultiCards.MsgMultiCards (Set *cardList*)

Dies ist der Kontruktor für eine neue MsgMultiCards-Nachricht.

Parameter

<i>cardList</i>	ist die Liste der ausgewählten Karten.
-----------------	--

8.60.3 Dokumentation der Elementfunktionen

8.60.3.1 Set<Card> ComObjects.MsgMultiCards.getCardList ()

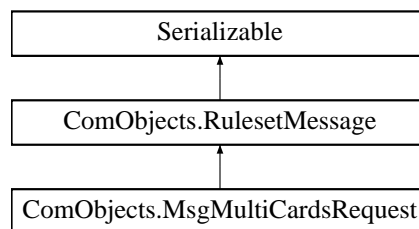
Gibt die Liste der gewählten Karten zurück.

Rückgabe

die Liste der Karten.

8.61 ComObjects.MsgMultiCardsRequest Klassenreferenz

Klassendiagramm für ComObjects.MsgMultiCardsRequest:



Öffentliche Methoden

- int [getCount](#) ()

8.61.1 Dokumentation der Elementfunktionen

8.61.1.1 int ComObjects.MsgMultiCardsRequest.getCount ()

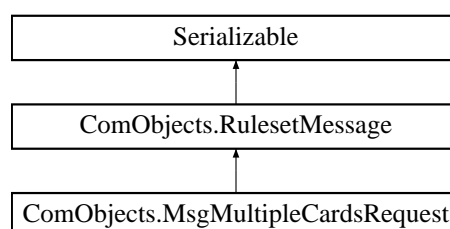
Diese Methode gibt die Anzahl der Karten zurück, die der Server vom Spieler erwartet.

Rückgabe

die Anzahl der Karten.

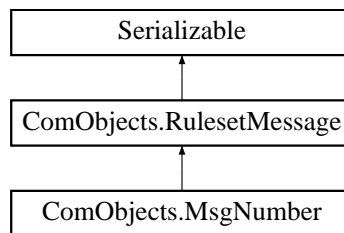
8.62 ComObjects.MsgMultipleCardsRequest Klassenreferenz

Klassendiagramm für ComObjects.MsgMultipleCardsRequest:



8.63 ComObjects.MsgNumber Klassenreferenz

Klassendiagramm für ComObjects.MsgNumber:



Öffentliche Methoden

- [MsgNumber](#) (int number)
- int [getNumber](#) ()

8.63.1 Ausführliche Beschreibung

Sie enthält eine Zahl.

8.63.2 Beschreibung der Konstruktoren und Destruktoren

8.63.2.1 ComObjects.MsgNumber.MsgNumber (int *number*)

Dies ist der Kontruktor für eine neue MsgNumber-Nachricht.

Parameter

<i>number</i>	ist eine Eingabe eines Spielers
---------------	---------------------------------

8.63.3 Dokumentation der Elementfunktionen

8.63.3.1 int ComObjects.MsgNumber.getNumber ()

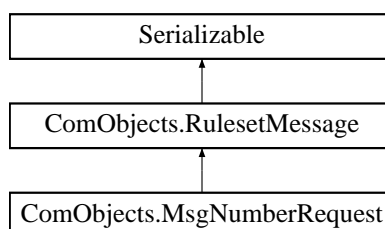
Diese Methode liefert die Eingabe eines Spielers.

Rückgabe

eine Zahl, die Eingabe des Spielers.

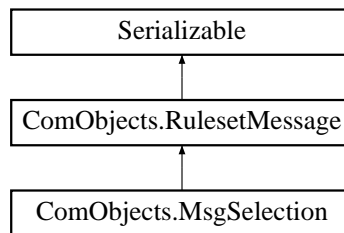
8.64 ComObjects.MsgNumberRequest Klassenreferenz

Klassendiagramm für ComObjects.MsgNumberRequest:



8.65 ComObjects.MsgSelection Klassenreferenz

Klassendiagramm für ComObjects.MsgSelection:



Öffentliche Methoden

- [MsgSelection](#) (int selection)
- int [getSelection](#) ()

8.65.1 Ausführliche Beschreibung

Diese Nachricht enthält Information über eine Auswahl, die der Spieler getroffen hat. Die Wahlmöglichkeiten werden durch Integer repräsentiert.

8.65.2 Beschreibung der Konstruktoren und Destruktoren

8.65.2.1 ComObjects.MsgSelection.MsgSelection (int selection)

Dies ist der Kontruktor für eine neue MsgSelection-Nachricht.

Parameter

<i>selection</i>	ist die getroffene Auswahl, repräsentiert durch einen Integer.
------------------	--

8.65.3 Dokumentation der Elementfunktionen

8.65.3.1 int ComObjects.MsgSelection.getSelection ()

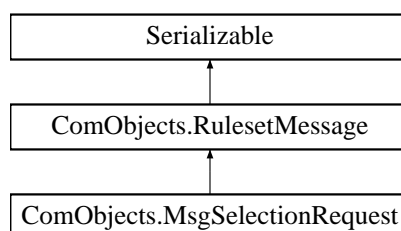
Diese Methode gibt die Auswahl des Spielers zurück, die er gemacht hat.

Rückgabe

die Auswahl.

8.66 ComObjects.MsgSelectionRequest Klassenreferenz

Klassendiagramm für ComObjects.MsgSelectionRequest:

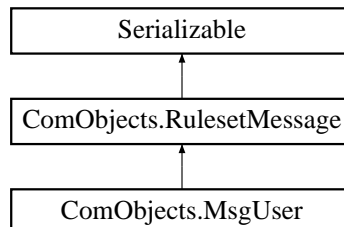


8.66.1 Ausführliche Beschreibung

Diese Nachricht sendet der Server an einen Spieler, wenn er eine Auswahl von diesem erwartet.

8.67 ComObjects.MsgUser Klassenreferenz

Klassendiagramm für ComObjects.MsgUser:



Öffentliche Methoden

- [MsgUser](#) ([GameClientUpdate](#) gameClientUpdate)
- [GameClientUpdate](#) [getGameClientUpdate](#) ()

8.67.1 Ausführliche Beschreibung

Sie wird dem Client gesendet, um dem ClientRuleset den aktuellen Spielzustand in Form eines GameClientUpdate zu übermitteln.

8.67.2 Beschreibung der Konstruktoren und Destruktoren

8.67.2.1 ComObjects.MsgUser.MsgUser ([GameClientUpdate](#) *gameClientUpdate*)

Dies ist der Konstruktor einer neuen MsgUser-Nachricht.

Parameter

<i>gameClient-Update</i>	ist der aktuelle Spielstand.
--------------------------	------------------------------

8.67.3 Dokumentation der Elementfunktionen

8.67.3.1 [GameClientUpdate](#) [ComObjects.MsgUser.getGameClientUpdate](#) ()

Diese Methode liefert den den aktuellen Spielzustand, der für ein Update benötigt wird.

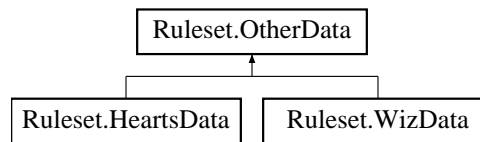
Rückgabe

den aktuellen Spielzustand.

8.68 Client.MVMessages Schnittstellenreferenz

8.69 Ruleset.OtherData Klassenreferenz

Klassendiagramm für Ruleset.OtherData:



Öffentliche Methoden

- String `getName()`

8.69.1 Dokumentation der Elementfunktionen

8.69.1.1 String Ruleset.OtherData.getName()

Holt den Namen des Spielers.

Rückgabe

name Der Name des Spielers

8.70 Client.View.OtherPlayer Klassenreferenz

8.70.1 Ausführliche Beschreibung

Autor

m4nkey

8.71 Client.View.OwnHand Klassenreferenz

8.71.1 Ausführliche Beschreibung

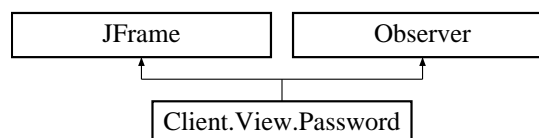
Der Spieler kann eine Karte durch Anklicken auswählen und durch einen zweiten Klick ausspielen.

Autor

m4nkey

8.72 Client.View.Password Klassenreferenz

Klassendiagramm für Client.View.Password:



Öffentliche Methoden

- void `update(Observable o, Object arg)`

8.72.1 Ausführliche Beschreibung

Autor

M4nkey

8.72.2 Dokumentation der Elementfunktionen

8.72.2.1 void Client.View.Password.update (Observable o, Object arg)

Wird durch notify() im [ClientModel](#) aufgerufen.

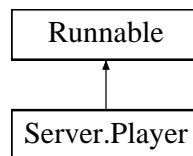
Je nach dem in arg übergebenen Befehl wird ein Update des Fensters ausgeführt oder eine Fehlermeldung angezeigt.

Parameter

<i>o</i>	erwartet ein Objekt von der Klasse ClientModel
<i>arg</i>	erwartet: windowChangeAcknowledged, windowChangeDenied

8.73 Server.Player Klassenreferenz

Klassendiagramm für Server.Player:



Öffentliche Methoden

- [Player](#) ([Server](#) lobbyServer, `ObjectOutput` output, `ObjectInput` input)
- void [send](#) (`ComObject` com)
- void [changeServer](#) ([Server](#) newServer)
- String [getName](#) ()
- void [setName](#) (String newName)

8.73.1 Ausführliche Beschreibung

Sie verwaltet für die Dauer einer Serververbindung die Verbindung zum Client

Autor

Viktoria

8.73.2 Beschreibung der Konstruktoren und Destruktoren

8.73.2.1 Server.Player.Player (Server lobbyServer, ObjectOutput output, ObjectInput input)

Konstruktor des Players, in ihm werden die Attribute server, comOut und comIn mit vom ClientListenerThret übergebenen werten Instanziiert.

Parameter

<i>lobbyServer</i>	ist der LobbyServer , der zu Beginn den Player verwaltet.
<i>output</i>	ist der ObjectOutput an den entsprechenden Client
<i>input</i>	ist der ObjectInput vom entsprechenden Client

8.73.3 Dokumentation der Elementfunktionen

8.73.3.1 void Server.Player.changeServer ([Server](#) *newServer*)

Diese Methode wechselt beim [Player](#) den [Server](#) an den er comObjects weiterleiten soll.

Dabei wird er aus dem playerSet des alten Servers entfernt und in das playerSet des neuen Players eingefügt. Danach wird vom neuen [Server](#) ein ComUpdatePlayerlist Objekt mit broadcast an alle Clients, die vom [Server](#) verwaltet werden, verschickt.

Parameter

<i>newServer</i>	ist der neue Server
------------------	-------------------------------------

8.73.3.2 String Server.Player.getName ()

Getter-Methode für den Benutzernamen.

Rückgabe

gibt den Benutzernamen des Spielers zurück

8.73.3.3 void Server.Player.send ([ComObject](#) *com*)

Diese Methode schickt ein ComObjekt an den Client.

Parameter

<i>com</i>	ist das ComObject das verschickt wird
------------	---------------------------------------

8.73.3.4 void Server.Player.setName ([String](#) *newName*)

Setter-Methode für den Benutzernamen.

Parameter

<i>newName</i>	ist der neue Name
----------------	-------------------

8.74 Ruleset.PlayerState Klassenreferenz

Öffentliche Methoden

- [String](#) [getName](#) ()
- [ArrayList](#)< [Card](#) > [getHand](#) ()
- [OtherData](#) [getOtherData](#) ()
- void [addCard](#) ([Card](#) card)
- boolean [removeCard](#) ([Card](#) card)

8.74.1 Ausführliche Beschreibung

Sie enthält den Namen des Spielers, seine Handkarten und [OtherData](#).

8.74.2 Dokumentation der Elementfunktionen

8.74.2.1 void Ruleset.PlayerState.addCard (Card card)

Gibt dem Spieler eine Karte.

Parameter

<i>card</i>	Die Karte die dem Spieler gegeben wird
-------------	--

8.74.2.2 ArrayList<Card> Ruleset.PlayerState.getHand ()

Holt die Kartenhand des Spielers.

Rückgabe

ownHand Die Kartenhand des Spielers

8.74.2.3 String Ruleset.PlayerState.getName ()

Holt den namen eines Spielers.

Rückgabe

name Der Name des Spielers

8.74.2.4 OtherData Ruleset.PlayerState.getOtherData ()

Holt die zusätzlichen Informationen des Spielers.

Rückgabe

ownHand Die zusätzlichen Informationen des Spielers

8.74.2.5 boolean Ruleset.PlayerState.removeCard (Card card)

Entfernt eine Karte aus der Hand des Spielers.

Parameter

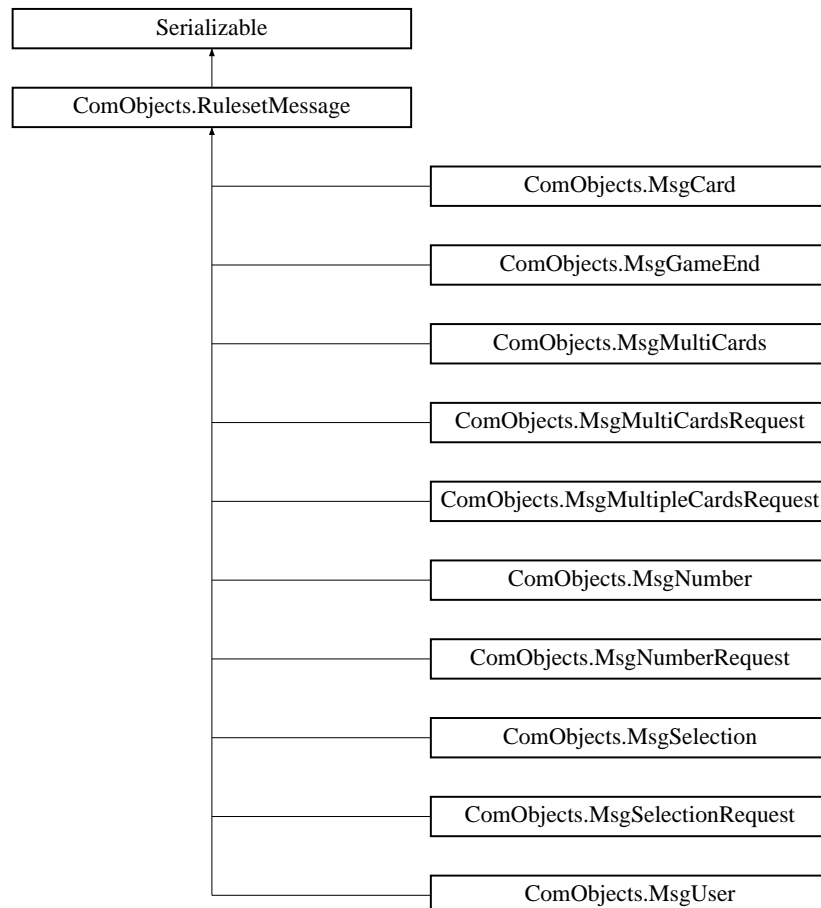
<i>card</i>	
-------------	--

Rückgabe

ownHand.remove(card) Gibt true zurück wenn die Karte in der Hand ist und false sonst

8.75 ComObjects.RulesetMessage Klassenreferenz

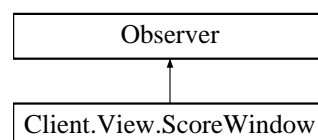
Klassendiagramm für ComObjects.RulesetMessage:

**8.75.1 Ausführliche Beschreibung**

Sie enthält einen Nachrichtentyp und vererbt an alle Nachrichten für das Regelwerk.

8.76 Ruleset.RulesetType Enum-Referenz**8.77 Client.View.ScoreWindow Klassenreferenz**

Klassendiagramm für Client.View.ScoreWindow:



Öffentliche Methoden

- void [update](#) (Observable o, Object arg)

8.77.1 Ausführliche Beschreibung

Autor

m4nkey

8.77.2 Dokumentation der Elementfunktionen

8.77.2.1 void Client.View.ScoreWindow.update (Observable o, Object arg)

Wird durch notify() im [ClientModel](#) aufgerufen.

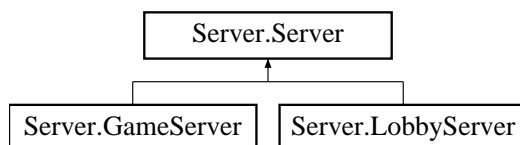
Je nach dem in arg übergebenen Befehl wird ein Update des Fensters ausgeführt oder eine Fehlermeldung angezeigt.

Parameter

<i>o</i>	erwartet ein Objekt von der Klasse ClientModel
<i>arg</i>	erwartet: showScore

8.78 Server.Server Klassenreferenz

Klassendiagramm für Server.Server:



Öffentliche Methoden

- void [receiveMessage](#) ([Player](#) player, [ComObject](#) com)
- synchronized void [sendToPlayer](#) (String name, [ComObject](#) com)
- synchronized void [addPlayer](#) ([Player](#) player)
- synchronized void [removePlayer](#) ([Player](#) player)
- synchronized void [broadcast](#) ([ComObject](#) com)

8.78.1 Ausführliche Beschreibung

Es stellt Methoden zur Nachrichtenversendung und -verarbeitung bereit, sowie zur Verwaltung von Playern

Autor

Viktoria

8.78.2 Dokumentation der Elementfunktionen

8.78.2.1 synchronized void Server.Server.addPlayer ([Player](#) player)

Diese Methode fügt einen [Player](#) dem Set an Playern hinzu, welche der [Server](#) verwaltet.

Parameter

<i>player</i>	ist der Player , der hinzugefügt wird
---------------	---

8.78.2.2 synchronized void Server.Server.broadcast ([ComObject com](#))

Diese Methode wird genutzt, um ein ComObject an alle Clients, die vom [Server](#) verwaltet werden, zu schicken.

Parameter

<i>com</i>	ist das ComObject, dass verschickt werden soll
------------	--

8.78.2.3 void Server.Server.receiveMessage ([Player player](#), [ComObject com](#))

Diese Methode dient zur Verarbeitung von eingehenden ComObjects.

Parameter

<i>player</i>	ist der Player von dem die Nachricht kommt
<i>com</i>	ist das ComObjekt vom Client verschickt wurde

8.78.2.4 synchronized void Server.Server.removePlayer ([Player player](#))

Diese Methode entfernt einen [Player](#) aus dem Set an Playern, welche der [Server](#) verwaltet.

Parameter

<i>player</i>	ist der Player , der entfernt wird
---------------	--

8.78.2.5 synchronized void Server.Server.sendToPlayer ([String name](#), [ComObject com](#))

Diese Methode wird genutzt, um ein ComObject an einen einzigen Client zu verschicken.

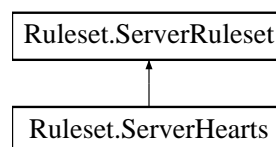
Der [Player](#) der die Nachricht verschicken soll wird Anhand des übergebenen Benutzernamens identifiziert.

Parameter

<i>name</i>	ist der Name des Clients, an den der Player die Nachricht verschicken soll
<i>c</i>	ist das ComObject, dass verschickt werden soll

8.79 Ruleset.ServerHearts Klassenreferenz

Klassendiagramm für Ruleset.ServerHearts:



Geschützte Methoden

- boolean [isValidMove](#) ([Card card](#))

Weitere Geerbte Elemente

8.79.1 Ausführliche Beschreibung

Sie enthält zudem weitere Methoden, welche für das Spiel Hearts spezifisch benötigt werden, wie die Regelung zum Tausch von Karten und die Berechnung der Stichpunkten.

8.79.2 Dokumentation der Elementfunktionen

8.79.2.1 `boolean Ruleset.ServerHearts.isValidMove (Card card)` `[protected]`, `[virtual]`

Prüft ob ein gemachter Zug in einem Hearts Spiel gültig ist.

Rückgabe

`isValid` true falls Zug gültig, false wenn nicht

Implementiert [Ruleset.ServerRuleset](#).

8.80 Server.ServerMain Klassenreferenz

Öffentliche, statische Methoden

- static void [main](#) (String[] args)

8.80.1 Ausführliche Beschreibung

Autor

Viktoria

8.80.2 Dokumentation der Elementfunktionen

8.80.2.1 `static void Server.ServerMain.main (String[] args)` `[static]`

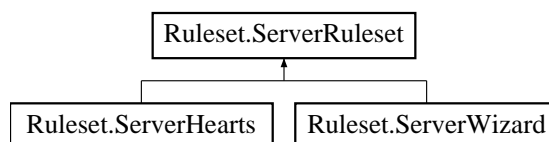
Die main-Methode erstellt einen neuen [LobbyServer](#).

Parameter

<code>args</code>	
-------------------	--

8.81 Ruleset.ServerRuleset Klassenreferenz

Klassendiagramm für Ruleset.ServerRuleset:



Öffentliche Methoden

- void [resolveMessage](#) ([MsgCard](#) msgCard, String name)
- void [resolveMessage](#) ([MsgMultiCards](#) msgMultiCard, String name)
- void [resolveMessage](#) ([MsgNumber](#) msgNumber, String name)
- void [resolveMessage](#) ([MsgSelection](#) msgSelection, String name)

Geschützte Methoden

- [PlayerState](#) [getCurrentPlayer](#) ()
- [PlayerState](#) [getPlayerState](#) (String name)
- void [send](#) ([RulesetMessage](#) message, String name)
- void [broadcast](#) ([RulesetMessage](#) message)

8.81.1 Ausführliche Beschreibung

Das [ServerRuleset](#) wird im [GameServer](#) instanziiert und verwaltet die Zustände des [GameStates](#) im Server. Mit der Methode [isValidMove\(\)](#) wird eine Eingabe eines Clients auf Regelkonformität überprüft und dann im [GameServer](#) das [GameState](#) verändert. Über [resolveMessage\(\)](#) kann eine [GameServer](#)instanz eine [RulesetMessage](#) vom Player an das Ruleset weiterleiten.

8.81.2 Dokumentation der Elementfunktionen

8.81.2.1 void Ruleset.ServerRuleset.broadcast (RulesetMessage message) [protected]

Schickt eine Nachricht an alle Spieler.

Parameter

<i>message</i>	Die Nachricht
----------------	---------------

8.81.2.2 PlayerState Ruleset.ServerRuleset.getCurrentPlayer () [protected]

Holt den Spieler der gerade am Zug ist.

Rückgabe

currentPlayer Der Spielzustand des Spielers der grad am Zug ist

8.81.2.3 PlayerState Ruleset.ServerRuleset.getPlayerState (String name) [protected]

Holt den Spielerzustand eines Spielers.

Parameter

<i>name</i>	Der Name des Spielers
-------------	-----------------------

Rückgabe

playerState Spielzustand eines Spielers

8.81.2.4 void Ruleset.ServerRuleset.resolveMessage (MsgCard msgCard, String name)

Verarbeitet die RulesetMessage dass eine Karte vom Spieler gespielt.

Parameter

<i>msgCard</i>	Die Nachricht vom Client welche Karte gespielt wurde
<i>name</i>	Der Name des Spielers

8.81.2.5 void Ruleset.ServerRuleset.resolveMessage (MsgMultiCards msgMultiCard, String name)

Verarbeitet die RulesetMessage dass mehrere Karten von einem Spieler übergeben wurden.

Parameter

<i>msgMultiCard</i>	Die Nachricht vom Client
<i>name</i>	Der Name des Spielers

8.81.2.6 void Ruleset.ServerRuleset.resolveMessage (MsgNumber msgNumber, String name)

Verarbeitet die RulesetMessage dass ein Spieler eine Stichangabe gemacht hat.

Parameter

<i>msgNumber</i>	Die Nachricht vom Client
<i>name</i>	Der Name des Spielers

8.81.2.7 void Ruleset.ServerRuleset.resolveMessage (MsgSelection msgSelection, String name)

Verarbeitet die RulesetMessage dass ein Spieler eine Farbe ausgewählt hat.

Parameter

<i>msgSelection</i>	Die Nachricht vom Client
<i>name</i>	Der Name des Spielers

8.81.2.8 void Ruleset.ServerRuleset.send (RulesetMessage message, String name) [protected]

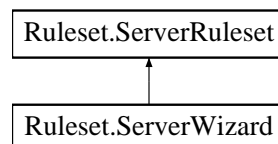
Schickt eine Nachricht an einen Spieler.

Parameter

<i>message</i>	Die Nachricht vom Typ RulesetMessage
<i>name</i>	Der Name vom Spieler

8.82 Ruleset.ServerWizard Klassenreferenz

Klassendiagramm für Ruleset.ServerWizard:

**Geschützte Methoden**

- boolean isValidMove (Card card)

Weitere Geerbte Elemente**8.82.1 Ausführliche Beschreibung**

Sie enthält zudem weitere Methoden, welche für das Spiel Wizard spezifisch benötigt werden, wie das Bestimmen einer Trumpffarbe und die Bestimmung der Rundenanzahl.

8.82.2 Dokumentation der Elementfunktionen

8.82.2.1 boolean Ruleset.ServerWizard.isValidMove (Card card) [protected],[virtual]

Prüft ob ein gemachter Zug in einem Wizard Spiel gültig ist.

Rückgabe

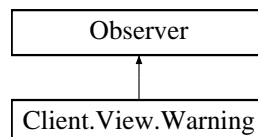
isValid true falls Zug gültig, false wenn nicht

Implementiert [Ruleset.ServerRuleset](#).

8.83 Client.ViewNotification Enum-Referenz

8.84 Client.View.Warning Klassenreferenz

Klassendiagramm für Client.View.Warning:



Öffentliche Methoden

- void [update](#) (Observable o, Object arg)

8.84.1 Ausführliche Beschreibung

Hinweise an, welche vom [ClientModel](#) übergeben wurden. Es wird nur im Fehlerfall angezeigt.

Autor

m4nkey

8.84.2 Dokumentation der Elementfunktionen

8.84.2.1 void Client.View.Warning.update (Observable o, Object arg)

Wird durch notify() im [ClientModel](#) aufgerufen.

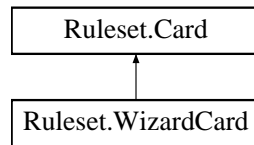
Je nach dem in arg übergebenen Befehl wird ein Update des Fensters ausgeführt oder eine Fehlermeldung angezeigt.

Parameter

<i>o</i>	erwartet ein Objekt von der Klasse ClientModel
<i>arg</i>	erwartet: openWarning

8.85 Ruleset.WizardCard Klassenreferenz

Klassendiagramm für Ruleset.WizardCard:

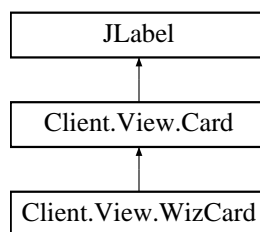


Weitere Geerbte Elemente

8.86 Ruleset.WizardDeck Klassenreferenz

8.87 Client.View.WizCard Klassenreferenz

Klassendiagramm für Client.View.WizCard:



Öffentliche Methoden

- [WizCard](#) (WizID id)
- [WizID](#) getCardID ()

8.87.1 Ausführliche Beschreibung

Sie wird verwendet um einzelne Karten auf das Spielfeld zu zeichnen. Dazu enthält sie die Pfadangabe zu dem Ordner, in dem die Bilder der Karten gespeichert sind, und eine ID, um das genaue Bild zu spezifizieren.

Autor

m4nkey

8.87.2 Beschreibung der Konstruktoren und Destruktoren

8.87.2.1 Client.View.WizCard.WizCard (WizID id)

Erstellt eine neue Wizard Karte für die Anzeige und zeichnet das Bild, das durch id spezifiziert ist.

Parameter

<i>id</i>	WizID der Karte
-----------	-----------------

8.87.3 Dokumentation der Elementfunktionen

8.87.3.1 WizID Client.View.WizCard.getCardID ()

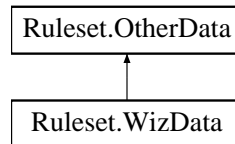
Gibt die WizID der Karte zurück

Rückgabe

WizID der Karte

8.88 Ruleset.WizData Klassenreferenz

Klassendiagramm für Ruleset.WizData:

**Öffentliche Methoden**

- int [getAnnouncedTricks](#) ()
- int [getAchievedTricks](#) ()
- int [getPoints](#) ()
- void [announceTricks](#) (int annouceTricks)
- void [setPoints](#) (int points)

8.88.1 Dokumentation der Elementfunktionen**8.88.1.1 void Ruleset.WizData.announceTricks (int *annouceTricks*)**

Beim Spielstart werden die vorausgesagten Stiche des Spieler gespeichert und die gemachten Stiche zurückgesetzt.

Parameter

<i>annouceTricks</i>	Die vorausgesagten Stiche des Spielers
----------------------	--

8.88.1.2 int Ruleset.WizData.getAchievedTricks ()

Holt die erreichten Stiche des Spielers.

Rückgabe

achievedTricks Die gemachten Stiche eines Spielers

8.88.1.3 int Ruleset.WizData.getAnnouncedTricks ()

Holt die angesagten Stiche des Spielers.

Rückgabe

announcedTricks Die angesagten Stiche

8.88.1.4 int Ruleset.WizData.getPoints ()

Holt den Punktestand des Spielers.

Rückgabe

points Der Punktestand des Spielers

8.88.1.5 void Ruleset.WizData.setPoints (int *points*)

Setzt den Punktestand eines Spielers.

Parameter

<i>points</i>	Der Punktestand eines Spielers
---------------	--------------------------------

8.89 Ruleset.WizID Enum-Referenz