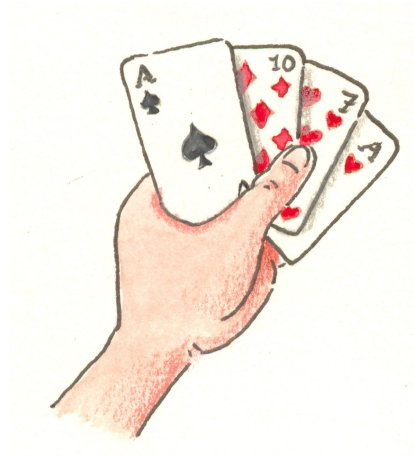


SPEZIFIKATION

8. 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

Inhaltsverzeichnis

1	Hierarchie-Verzeichnis	2
1.1	Klassenhierarchie	2
2	Klassen-Verzeichnis	5
2.1	Auflistung der Klassen	5
3	Klassen-Dokumentation	9
3.1	Client.CardID Enum-Referenz	9
3.2	Client.ClientController Klassenreferenz	9
3.3	Client.ClientMain Klassenreferenz	9
3.3.1	Dokumentation der Elementfunktionen	9
3.4	Client.ClientModel Klassenreferenz	10
3.4.1	Ausführliche Beschreibung	10
3.4.2	Dokumentation der Elementfunktionen	10
3.5	Client.ClientState Enum-Referenz	12
3.6	Client.MessageListenerThread Klassenreferenz	12
3.7	Client.MVMessages Schnittstellenreferenz	12
3.8	Client.View.Card Klassenreferenz	12
3.8.1	Ausführliche Beschreibung	13
3.8.2	Beschreibung der Konstruktoren und Destruktoren	13
3.9	Client.View.ChooseCards Klassenreferenz	13
3.9.1	Dokumentation der Elementfunktionen	13
3.10	Client.View.ChooseItem Klassenreferenz	13
3.10.1	Ausführliche Beschreibung	13
3.10.2	Dokumentation der Elementfunktionen	14
3.11	Client.View.CreateGame Klassenreferenz	14
3.11.1	Ausführliche Beschreibung	14
3.11.2	Beschreibung der Konstruktoren und Destruktoren	14
3.11.3	Dokumentation der Elementfunktionen	14
3.12	Client.View.DiscardPile Klassenreferenz	15
3.12.1	Ausführliche Beschreibung	15
3.13	Client.View.DrawDeck Klassenreferenz	15
3.13.1	Ausführliche Beschreibung	15
3.14	Client.View.Game Klassenreferenz	15
3.14.1	Ausführliche Beschreibung	15
3.14.2	Beschreibung der Konstruktoren und Destruktoren	15
3.14.3	Dokumentation der Elementfunktionen	15
3.15	Client.View.GameLobby Klassenreferenz	17
3.15.1	Ausführliche Beschreibung	17

3.15.2 Dokumentation der Elementfunktionen	17
3.16 Client.View.GamePanel Klassenreferenz	17
3.16.1 Ausführliche Beschreibung	17
3.17 Client.View.HeartsCard Klassenreferenz	18
3.17.1 Ausführliche Beschreibung	18
3.17.2 Beschreibung der Konstruktoren und Destrukturen	18
3.17.3 Dokumentation der Elementfunktionen	18
3.18 Client.View.InputNumber Klassenreferenz	18
3.18.1 Ausführliche Beschreibung	18
3.18.2 Dokumentation der Elementfunktionen	19
3.19 Client.View.Language Enum-Referenz	19
3.19.1 Ausführliche Beschreibung	19
3.20 Client.View.Lobby Klassenreferenz	19
3.20.1 Ausführliche Beschreibung	19
3.20.2 Dokumentation der Elementfunktionen	19
3.21 Client.View.Login Klassenreferenz	20
3.21.1 Ausführliche Beschreibung	21
3.21.2 Dokumentation der Elementfunktionen	21
3.22 Client.View.OtherPlayer Klassenreferenz	21
3.22.1 Ausführliche Beschreibung	21
3.23 Client.View.OwnHand Klassenreferenz	21
3.23.1 Ausführliche Beschreibung	22
3.24 Client.View.Password Klassenreferenz	22
3.24.1 Ausführliche Beschreibung	22
3.24.2 Dokumentation der Elementfunktionen	22
3.25 Client.View.ScoreWindow Klassenreferenz	22
3.25.1 Ausführliche Beschreibung	22
3.25.2 Dokumentation der Elementfunktionen	22
3.26 Client.View.Warning Klassenreferenz	23
3.26.1 Ausführliche Beschreibung	23
3.26.2 Dokumentation der Elementfunktionen	23
3.27 Client.View.WizCard Klassenreferenz	23
3.27.1 Ausführliche Beschreibung	23
3.27.2 Beschreibung der Konstruktoren und Destrukturen	24
3.27.3 Dokumentation der Elementfunktionen	24
3.28 Client.ViewNotification Enum-Referenz	24
3.29 ComObjects.ComBeenKicked Klassenreferenz	24
3.29.1 Ausführliche Beschreibung	24
3.29.2 Beschreibung der Konstruktoren und Destrukturen	24
3.29.3 Dokumentation der Elementfunktionen	24

3.30 ComObjects.ComChatMessage Klassenreferenz	24
3.30.1 Ausführliche Beschreibung	25
3.30.2 Beschreibung der Konstruktoren und Destruktoren	25
3.30.3 Dokumentation der Elementfunktionen	25
3.31 ComObjects.ComClientLeave Klassenreferenz	25
3.31.1 Ausführliche Beschreibung	25
3.32 ComObjects.ComClientQuit Klassenreferenz	25
3.32.1 Ausführliche Beschreibung	25
3.33 ComObjects.ComCreateGameRequest Klassenreferenz	25
3.33.1 Ausführliche Beschreibung	26
3.33.2 Beschreibung der Konstruktoren und Destruktoren	26
3.33.3 Dokumentation der Elementfunktionen	26
3.34 ComObjects.ComInitGameLobby Klassenreferenz	27
3.34.1 Ausführliche Beschreibung	27
3.34.2 Beschreibung der Konstruktoren und Destruktoren	27
3.34.3 Dokumentation der Elementfunktionen	27
3.35 ComObjects.ComInitLobby Klassenreferenz	27
3.35.1 Ausführliche Beschreibung	28
3.35.2 Beschreibung der Konstruktoren und Destruktoren	28
3.35.3 Dokumentation der Elementfunktionen	28
3.36 ComObjects.ComJoinRequest Klassenreferenz	28
3.36.1 Ausführliche Beschreibung	28
3.36.2 Beschreibung der Konstruktoren und Destruktoren	28
3.36.3 Dokumentation der Elementfunktionen	29
3.37 ComObjects.ComKickPlayerRequest Klassenreferenz	29
3.37.1 Ausführliche Beschreibung	29
3.37.2 Beschreibung der Konstruktoren und Destruktoren	29
3.37.3 Dokumentation der Elementfunktionen	29
3.38 ComObjects.ComLobbyUpdateGamelist Klassenreferenz	29
3.38.1 Ausführliche Beschreibung	30
3.38.2 Beschreibung der Konstruktoren und Destruktoren	30
3.38.3 Dokumentation der Elementfunktionen	30
3.39 ComObjects.ComLoginRequest Klassenreferenz	30
3.39.1 Ausführliche Beschreibung	30
3.39.2 Beschreibung der Konstruktoren und Destruktoren	31
3.39.3 Dokumentation der Elementfunktionen	32
3.40 ComObjects.ComObject Klassenreferenz	32
3.41 ComObjects.ComRuleset Klassenreferenz	32
3.41.1 Ausführliche Beschreibung	32
3.41.2 Beschreibung der Konstruktoren und Destruktoren	32

3.41.3	Dokumentation der Elementfunktionen	32
3.42	ComObjects.ComServerAcknowledgement Klassenreferenz	33
3.43	ComObjects.ComStartGame Klassenreferenz	33
3.43.1	Ausführliche Beschreibung	33
3.44	ComObjects.ComUpdatePlayerlist Klassenreferenz	33
3.44.1	Ausführliche Beschreibung	33
3.44.2	Beschreibung der Konstruktoren und Destruktoren	33
3.44.3	Dokumentation der Elementfunktionen	33
3.45	ComObjects.ComWarning Klassenreferenz	34
3.45.1	Ausführliche Beschreibung	34
3.45.2	Beschreibung der Konstruktoren und Destruktoren	34
3.45.3	Dokumentation der Elementfunktionen	34
3.46	ComObjects.MsgCard Klassenreferenz	34
3.46.1	Ausführliche Beschreibung	34
3.46.2	Beschreibung der Konstruktoren und Destruktoren	35
3.46.3	Dokumentation der Elementfunktionen	35
3.47	ComObjects.MsgCardRequest Klassenreferenz	35
3.47.1	Ausführliche Beschreibung	35
3.48	ComObjects.MsgGameEnd Klassenreferenz	35
3.48.1	Ausführliche Beschreibung	35
3.49	ComObjects.MsgMultiCards Klassenreferenz	35
3.49.1	Ausführliche Beschreibung	35
3.49.2	Beschreibung der Konstruktoren und Destruktoren	36
3.49.3	Dokumentation der Elementfunktionen	37
3.50	ComObjects.MsgMultiCardsRequest Klassenreferenz	37
3.50.1	Dokumentation der Elementfunktionen	37
3.51	ComObjects.MsgMultipleCardsRequest Klassenreferenz	37
3.52	ComObjects.MsgNumber Klassenreferenz	37
3.52.1	Ausführliche Beschreibung	37
3.52.2	Beschreibung der Konstruktoren und Destruktoren	38
3.52.3	Dokumentation der Elementfunktionen	39
3.53	ComObjects.MsgNumberRequest Klassenreferenz	39
3.54	ComObjects.MsgSelection Klassenreferenz	39
3.54.1	Ausführliche Beschreibung	39
3.54.2	Beschreibung der Konstruktoren und Destruktoren	39
3.54.3	Dokumentation der Elementfunktionen	39
3.55	ComObjects.MsgSelectionRequest Klassenreferenz	40
3.55.1	Ausführliche Beschreibung	40
3.56	ComObjects.MsgUser Klassenreferenz	40
3.56.1	Ausführliche Beschreibung	40

3.56.2	Beschreibung der Konstruktoren und Destruktoren	40
3.56.3	Dokumentation der Elementfunktionen	40
3.57	ComObjects.RulesetMessage Klassenreferenz	40
3.57.1	Ausführliche Beschreibung	40
3.58	Ruleset.Card Schnittstellenreferenz	41
3.58.1	Dokumentation der Elementfunktionen	41
3.59	Ruleset.CardDeck Klassenreferenz	41
3.60	Ruleset.CardDeckBuilder Klassenreferenz	41
3.61	Ruleset.ClientHearts Klassenreferenz	41
3.61.1	Dokumentation der Elementfunktionen	41
3.62	Ruleset.ClientRuleset Klassenreferenz	42
3.62.1	Ausführliche Beschreibung	42
3.62.2	Dokumentation der Elementfunktionen	42
3.63	Ruleset.ClientWizard Klassenreferenz	43
3.63.1	Dokumentation der Elementfunktionen	43
3.64	Ruleset.Colour Enum-Referenz	44
3.65	Ruleset.GameClientUpdate Klassenreferenz	44
3.65.1	Ausführliche Beschreibung	44
3.65.2	Dokumentation der Elementfunktionen	44
3.66	Ruleset.GamePhase Enum-Referenz	45
3.67	Ruleset.GameState Klassenreferenz	45
3.67.1	Ausführliche Beschreibung	45
3.67.2	Beschreibung der Konstruktoren und Destruktoren	45
3.67.3	Dokumentation der Elementfunktionen	45
3.68	Ruleset.HeartsDeck Klassenreferenz	48
3.69	Ruleset.HeartsCard Enum-Referenz	48
3.69.1	Dokumentation der Elementfunktionen	48
3.70	Ruleset.HeartsData Klassenreferenz	48
3.70.1	Dokumentation der Elementfunktionen	48
3.71	Ruleset.isValidMoveWizard Klassenreferenz	49
3.72	Ruleset.OtherData Klassenreferenz	49
3.72.1	Beschreibung der Konstruktoren und Destruktoren	49
3.72.2	Dokumentation der Elementfunktionen	49
3.73	Ruleset.PlayerState Klassenreferenz	49
3.73.1	Ausführliche Beschreibung	50
3.73.2	Beschreibung der Konstruktoren und Destruktoren	50
3.73.3	Dokumentation der Elementfunktionen	50
3.74	Ruleset.RulesetType Enum-Referenz	51
3.75	Ruleset.ServerHearts Klassenreferenz	51
3.75.1	Ausführliche Beschreibung	51

3.75.2 Dokumentation der Elementfunktionen	51
3.76 Ruleset.ServerRuleset Klassenreferenz	51
3.76.1 Ausführliche Beschreibung	52
3.76.2 Beschreibung der Konstruktoren und Destrukturen	52
3.76.3 Dokumentation der Elementfunktionen	52
3.77 Ruleset.ServerWizard Klassenreferenz	55
3.77.1 Ausführliche Beschreibung	55
3.77.2 Dokumentation der Elementfunktionen	55
3.78 Ruleset.WizardCard Enum-Referenz	56
3.78.1 Dokumentation der Elementfunktionen	56
3.79 Ruleset.WizardDeck Klassenreferenz	56
3.80 Ruleset.WizData Klassenreferenz	56
3.80.1 Dokumentation der Elementfunktionen	57
3.81 Server.ClientListenerThread Klassenreferenz	57
3.82 Server.GameServer Klassenreferenz	57
3.82.1 Ausführliche Beschreibung	58
3.82.2 Beschreibung der Konstruktoren und Destrukturen	58
3.82.3 Dokumentation der Elementfunktionen	58
3.83 Server.GameServerRepresentation Klassenreferenz	60
3.83.1 Ausführliche Beschreibung	60
3.83.2 Beschreibung der Konstruktoren und Destrukturen	60
3.84 Server.LobbyServer Klassenreferenz	61
3.84.1 Ausführliche Beschreibung	61
3.84.2 Dokumentation der Elementfunktionen	61
3.85 Server.LobbyServer.ClientListenerThread Klassenreferenz	62
3.85.1 Ausführliche Beschreibung	63
3.86 Server.Player Klassenreferenz	63
3.86.1 Ausführliche Beschreibung	63
3.86.2 Beschreibung der Konstruktoren und Destrukturen	63
3.86.3 Dokumentation der Elementfunktionen	63
3.87 Server.Server Klassenreferenz	64
3.87.1 Ausführliche Beschreibung	64
3.87.2 Dokumentation der Elementfunktionen	64
3.88 Server.ServerMain Klassenreferenz	65
3.88.1 Ausführliche Beschreibung	65
3.88.2 Dokumentation der Elementfunktionen	66

1 Hierarchie-Verzeichnis

1.1 Klassenhierarchie

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

Client.CardID	9
Client.ClientController	9
Client.ClientMain	9
Client.ClientState	12
Client.MessageListenerThread	12
Client.MVMessages	12
Client.View.DiscardPile	15
Client.View.DrawDeck	15
Client.View.Language	19
Client.View.OtherPlayer	21
Client.View.OwnHand	21
Client.ViewNotification	24
ComObjects.ComBeenKicked	24
ComObjects.MsgCardRequest	35
Ruleset.Card	41
Ruleset.HeartsCard	48
Ruleset.WizardCard	56
Ruleset.CardDeck	41
Ruleset.CardDeckBuilder	41
Ruleset.ClientRuleset	42
Ruleset.ClientHearts	41
Ruleset.ClientWizard	43
Ruleset.Colour	44
Ruleset.GameClientUpdate	44
Ruleset.GamePhase	45
Ruleset.GameState	45
Ruleset.HearthsDeck	48
Ruleset.isValidMoveWizard	49

Ruleset.OtherData	49
Ruleset.HeartsData	48
Ruleset.WizData	56
Ruleset.PlayerState	49
Ruleset.RulesetType	51
Ruleset.ServerRuleset	51
Ruleset.ServerHearts	51
Ruleset.ServerWizard	55
Ruleset.WizardDeck	56
Runnable	
Server.ClientListenerThread	57
Server.LobbyServer.ClientListenerThread	62
Server.Player	63
Server.GameServerRepresentation	60
Server.Server	64
Server.GameServer	57
Server.LobbyServer	61
Server.ServerMain	65
Serializable	
ComObjects.ComObject	32
ComObjects.ComChatMessage	24
ComObjects.ComClientLeave	25
ComObjects.ComClientQuit	25
ComObjects.ComCreateGameRequest	25
ComObjects.ComInitGameLobby	27
ComObjects.ComInitLobby	27
ComObjects.ComJoinRequest	28
ComObjects.ComKickPlayerRequest	29
ComObjects.ComLobbyUpdateGamelist	29
ComObjects.ComLoginRequest	30
ComObjects.ComRuleset	32
ComObjects.ComServerAcknowledgement	33
ComObjects.ComStartGame	33

ComObjects.ComUpdatePlayerlist	33
ComObjects.ComWarning	34
ComObjects.RulesetMessage	40
ComObjects.MsgCard	34
ComObjects.MsgGameEnd	35
ComObjects.MsgMultiCards	35
ComObjects.MsgMultiCardsRequest	37
ComObjects.MsgMultipleCardsRequest	37
ComObjects.MsgNumber	37
ComObjects.MsgNumberRequest	39
ComObjects.MsgSelection	39
ComObjects.MsgSelectionRequest	40
ComObjects.MsgUser	40
Observable	
Client.ClientModel	10
Observer	
Client.View.ChooseCards	13
Client.View.ChooseItem	13
Client.View.CreateGame	14
Client.View.Game	15
Client.View.GameLobby	17
Client.View.InputNumber	18
Client.View.Lobby	19
Client.View.Login	20
Client.View.Password	22
Client.View.ScoreWindow	22
Client.View.Warning	23
JFrame	
Client.View.CreateGame	14
Client.View.Game	15
Client.View.GameLobby	17
Client.View.Lobby	19
Client.View.Login	20

Client.View.Password	22
JLabel	
Client.View.Card	12
Client.View.HeartsCard	18
Client.View.WizCard	23
JPanel	
Client.View.GamePanel	17

2 Klassen-Verzeichnis

2.1 Auflistung der Klassen

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

Client.CardID	9
Client.ClientController	9
Client.ClientMain	
Die ClientMain Klasse startet den Spielclient und initialisiert dessen Komponenten	9
Client.ClientModel	
Implementiert das Client Model	10
Client.ClientState	
Dieser Enumerator enthält alle Zustände in denen sich der Client befinden kann	12
Client.MessageListenerThread	12
Client.MVMessages	12
Client.View.Card	
Card ist die View-seitige Repräsentation einer Karte	12
Client.View.ChooseCards	13
Client.View.ChooseItem	
Dieses Fenster ermöglicht es dem Spieler aus einer Liste von Items eines auszuwählen	13
Client.View.CreateGame	
Das Fenster CreateGame dient dem Benutzer zur Erstellung eines neuen Spieles	14
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	15
Client.View.DrawDeck	
Stellt einen Aufnahmestapel dar	15
Client.View.Game	
Im Game Fenster läuft das Spiel ab.Es enthält den Spielchat und ein GamePanel	15
Client.View.GameLobby	
Die GameLobby modelliert das Wartefenster, in dem beigetretene Spieler auf den Start des Spieles durch den Spielleiter warten	17

Client.View.GamePanel	
Das Panel ist die Komponente des Game-Fensters, welche das eigentliche Spiel darstellt	17
Client.View.HeartsCard	
HeartsCard ist die View-seitige Repräsentation einer Hearts-Karte	18
Client.View.InputNumber	
In diesem Fenster, kann der Benutzer eine Zahl eingeben	18
Client.View.Language	
Language stellt Repräsentationen verschiedener Sprachen dar, die von der GUI verwendet werden, um festzustellen welche Anzeigesprache verwendet werden soll	19
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	19
Client.View.Login	
Das Login-Fenster repräsentiert den initialen Dialog zwischen Benutzer und Client	20
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	21
Client.View.OwnHand	
Stellt die Karten dar, die der Spieler auf der Hand hat	21
Client.View.Password	
Dieses Fenster ermöglicht die Eingabe eines Passwortes um einem Passwortgeschütztem Spiel beizutreten oder per 'Leave' wieder in die Lobby zurückzukehren	22
Client.View.ScoreWindow	
Dieses Fenster zeigt den momentanen Punktestand nach jeder Runde und den Gesamtpunktestand am Ende des Spieles an	22
Client.View.Warning	
Das Warning-Fenster zeigt dem Benutzer Fehlermeldungen bzw	23
Client.View.WizCard	
WizCard ist die View-seitige Repräsentation einer Wizard-Karte	23
Client.ViewNotification	24
ComObjects.ComBeenKicked	
Diese Klasse ist ein spezielles Kommunikations-Objekt	24
ComObjects.ComChatMessage	
Diese Klasse ist ein spezielles Kommunikations-Objekt	24
ComObjects.ComClientLeave	
Diese Klasse ist ein spezielles Kommunikations-Objekt	25
ComObjects.ComClientQuit	
Diese Klasse ist ein spezielles Kommunikations-Objekt	25
ComObjects.ComCreateGameRequest	
Diese Klasse ist ein spezielles Kommunikations-Objekt	25
ComObjects.ComInitGameLobby	
Diese Klasse ist ein spezielles Kommunikations-Objekt	27

ComObjects.ComInitLobby	
Diese Klasse ist ein spezielles Kommunikations-Objekt	27
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	29
ComObjects.ComLoginRequest	
Diese Klasse ist ein spezielles Kommunikations-Objekt	30
ComObjects.ComObject	32
ComObjects.ComRuleset	
Diese Klasse ist ein spezielles Kommunikations-Objekt	32
ComObjects.ComServerAcknowledgement	
Diese Klasse ist ein spezielles Kommunikations-Objekt	33
ComObjects.ComStartGame	
Diese Klasse ist ein spezielles Kommunikations-Objekt	33
ComObjects.ComUpdatePlayerlist	
Diese Klasse ist ein spezielles Kommunikations-Objekt	33
ComObjects.ComWarning	
Diese Klasse ist ein spezielles Kommunikations-Objekt	34
ComObjects.MsgCard	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	34
ComObjects.MsgCardRequest	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	35
ComObjects.MsgGameEnd	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	35
ComObjects.MsgMultiCards	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	35
ComObjects.MsgMultiCardsRequest	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	37
ComObjects.MsgMultipleCardsRequest	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	37
ComObjects.MsgNumber	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	37
ComObjects.MsgNumberRequest	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	39
ComObjects.MsgSelection	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	39
ComObjects.MsgSelectionRequest	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	40

ComObjects.MsgUser	
Diese Klasse ist eine Verfeinerung der RulesetMessage-Klasse	40
ComObjects.RulesetMessage	
Diese Klasse ist eine Verfeinerung der ComRuleset-Klasse	40
Ruleset.Card	
Modelliert eine Spielkarte	41
Ruleset.CardDeck	41
Ruleset.CardDeckBuilder	41
Ruleset.ClientHearts	
Diese Klasse bildet das Regelwerk für den Client bei einer Partie Hearts	41
Ruleset.ClientRuleset	
ClientRuleset ist eine abstrakte Klasse und wird zur Regelvorauswertung im Client verwendet	42
Ruleset.ClientWizard	
Diese Klasse bildet das Regelwerk für den Client bei einer Partie Wizard	43
Ruleset.Colour	
Repräsentiert die Farbe einer Karte	44
Ruleset.GameClientUpdate	
Das GameClientUpdate wird vom RuleSet über den GameServer an den Client geschickt und enthält alle Änderungen des GameState , die für den Client relevant sind	44
Ruleset.GamePhase	
Die GamePhase modelliert die verschiedenen Zustände des Spiels im GameState	45
Ruleset.GameState	
Das GameState modelliert einen aktuellen Spielzustand, es wird vom GameServer instanziiert und vom RuleSet bearbeitet	45
Ruleset.HearthsDeck	48
Ruleset.HeartsCard	
Modelliert eine Heartskarte	48
Ruleset.HeartsData	
Die zusätzlichen Informationen eines Spielers zum Spiel Hearts	48
Ruleset.isValidMoveWizard	49
Ruleset.OtherData	
OtherData ist abstract und speichert die zusätzlichen Informationen eines Spielers	49
Ruleset.PlayerState	
Repräsentiert den Spielzustand eines Spielers, und wird unter anderem im GameState gespeichert	49
Ruleset.RulesetType	
Die verschiedenen Regelwerke	51
Ruleset.ServerHearts	
Diese Klasse erstellt das Regelwerk zum Spiel Hearts	51
Ruleset.ServerRuleset	
Das ServerRuleset ist eine akstrakte Klasse und für den Ablauf und die Einhaltung der Regeln eines Spiels zuständig (/L280/)	51

Ruleset.ServerWizard	
Diese Klasse erstellt das Regelwerk zum Spiel Wizard	55
Ruleset.WizardCard	
Modelliert eine Heartskarte	56
Ruleset.WizardDeck	56
Ruleset.WizData	
Die zusätzlichen Informationen eines Spielers zum Spiel Wizard	56
Server.ClientListenerThread	57
Server.GameServer	
Diese Klasse ist für die Spielverwaltung zuständig	57
Server.GameServerRepresentation	
Dies eine Klasse, die Informationen über den Zustand eines Spielervers bereithält	60
Server.LobbyServer	
Diese Klasse ist für die Verwaltung der Spiellobby auf dem Server verantwortlich	61
Server.LobbyServer.ClientListenerThread	
Diese Klasse ist für das Zustandekommen von Clientverbindungen zuständig	62
Server.Player	
Die Player-Klasse wird zum Versenden von Java Serializable Objects verwendet	63
Server.Server	
Ist ein abstrakte Klasse, von der die Klassen LobbyServer und GameServer erben	64
Server.ServerMain	
Diese Klasse startet den Server und ist für die Konfigurationund Wartung des Servers verantwort- lich	65

3 Klassen-Dokumentation

3.1 Client.CardID Enum-Referenz

3.2 Client.ClientController Klassenreferenz

3.3 Client.ClientMain Klassenreferenz

Öffentliche, statische Methoden

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

3.3.1 Dokumentation der Elementfunktionen

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

Parameter

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

3.4 Client.ClientModel Klassenreferenz

Abgeleitet von Observable.

Klassen

- class **MessageListenerThread**

Öffentliche Methoden

- List< String > [getGameLobbyPlayerlist](#) ()
- Set< [GameServerRepresentation](#) > [getFullServerLobbyGamelist](#) ()
- [GameServerRepresentation](#) [getServerLobbyGamelistUpdate](#) ()
- List< String > [getPlayerlistUpdate](#) ()
- String [getChatMessage](#) ()
- [CardID](#) [getPlayedCard](#) ()
- void [setLanguage](#) (final [Language](#) language)
- [Language](#) [getLanguage](#) ()
- void [kickPlayer](#) (final String name)
- void [hostGame](#) (String gameName, String password)
- void [joinGame](#) (final String name, final String password)
- void [makeMove](#) ([CardID](#) id)
- void [createConnection](#) (final String username, final String serverAdress, final int port)

3.4.1 Ausführliche Beschreibung

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

3.4.2 Dokumentation der Elementfunktionen

3.4.2.1 List<String> Client.ClientModel.getGameLobbyPlayerlist ()

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

Rückgabe

List Eine Liste der Spieler in der Spiellobby.

3.4.2.2 Set<GameServerRepresentation> Client.ClientModel.getFullServerLobbyGamelist ()

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

Rückgabe

Set Enthält alle Spiele in der ServerLobby.

3.4.2.3 GameServerRepresentation Client.ClientModel.getServerLobbyGamelistUpdate ()

Diese Methode wird von der View aufgerufen und aktualisiert einzelne Spiele.

Rückgabe

GameServerRepresentation Daten eines Spieles.

3.4.2.4 List<String> Client.ClientModel.getPlayerlistUpdate ()

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

Rückgabe

List Update für die aktuelle Spielerliste.

3.4.2.5 String Client.ClientModel.getChatMessage ()

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

Rückgabe

String die Chatnachricht.

3.4.2.6 CardID Client.ClientModel.getPlayedCard ()

Gibt der View die gespielte Karte eines anderen Spielers zurück.

Rückgabe

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

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

Setzt die Sprache der GUI.

Parameter

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

3.4.2.8 Language Client.ClientModel.getLanguage ()

Liefert die Sprache der GUI.

Rückgabe

language Enumerator der die Spielsprache anzeigt.

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

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

Parameter

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

3.4.2.10 void Client.ClientModel.hostGame (String *gameName*, String *password*)

Wird vom [ClientController](#) aufgerufen und erstellt ein neues Spiel auf dem Server.

Parameter

<i>gameName</i>	String Name des Spieles.
<i>password</i>	String Passwort zum sichern des Spieles.

3.4.2.11 void Client.ClientModel.joinGame (final String *name*, final String *password*)

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

Parameter

<i>name</i>	String Der Name des Spiels.
<i>password</i>	String Passwort eines Spieles.

3.4.2.12 void Client.ClientModel.makeMove (CardID *id*)

Wird vom ClientController aufgerufen um eine Karte auszuspielen.

Parameter

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

3.4.2.13 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.

3.5 Client.ClientState Enum-Referenz

3.6 Client.MessageListenerThread Klassenreferenz

3.7 Client.MVMessages Schnittstellenreferenz

3.8 Client.View.Card Klassenreferenz

Abgeleitet von JLabel.

Basisklasse für [Client.View.HeartsCard](#) und [Client.View.WizCard](#).

Öffentliche Methoden

- [Card](#) (String s)

3.8.1 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.

3.8.2 Beschreibung der Konstruktoren und Destruktoren

3.8.2.1 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
---	------------------------------------

3.9 Client.View.ChooseCards Klassenreferenz

Abgeleitet von Observer.

Öffentliche Methoden

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

3.9.1 Dokumentation der Elementfunktionen

3.9.1.1 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

3.10 Client.View.ChooseItem Klassenreferenz

Abgeleitet von Observer.

Öffentliche Methoden

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

3.10.1 Ausführliche Beschreibung

Autor

m4nkey

3.10.2 Dokumentation der Elementfunktionen

3.10.2.1 void Client.View.ChooseItem.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: openChooseItem

3.11 Client.View.CreateGame Klassenreferenz

Abgeleitet von JFrame und Observer.

Öffentliche Methoden

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

3.11.1 Ausführliche Beschreibung

Es bietet alle Komponenten, um ein Regelwerk zu wählen, einen Spielnamen festzulegen und das Spiel durch ein Passwort zu schützen. 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

3.11.2 Beschreibung der Konstruktoren und Destruktoren

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

Erstellt das [CreateGame](#) Fenster.

Ausnahmebehandlung

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

3.11.3 Dokumentation der Elementfunktionen

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

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

Je nach dem in *arg* übergebenen ViewNotification-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

3.12 Client.View.DiscardPile Klassenreferenz

3.12.1 Ausführliche Beschreibung

Autor

m4nkey

3.13 Client.View.DrawDeck Klassenreferenz

3.13.1 Ausführliche Beschreibung

Autor

m4nkey

3.14 Client.View.Game Klassenreferenz

Abgeleitet von JFrame und Observer.

Öffentliche Methoden

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

3.14.1 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

3.14.2 Beschreibung der Konstruktoren und Destruktoren

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

Erstellt das [Game](#) Fenster.

Ausnahmebehandlung

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

3.14.3 Dokumentation der Elementfunktionen

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

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

Je nach dem in `arg` übergebenen `ViewNotification`-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

3.14.3.2 void Client.View.Game.update (Observable o, String arg)

Wird durch notify() im [ClientModel](#) aufgerufen, wenn eine Chatnachricht übergeben wird.

Parameter

<i>o</i>	erwartet ein Objekt von der Klasse ClientModel
<i>arg</i>	erwartet eine Chatnachricht in String-Form

3.15 Client.View.GameLobby Klassenreferenz

Abgeleitet von JFrame und Observer.

Öffentliche Methoden

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

3.15.1 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

3.15.2 Dokumentation der Elementfunktionen

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

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

Je nach dem in arg übergebenen ViewNotification-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

3.16 Client.View.GamePanel Klassenreferenz

Abgeleitet von JPanel.

3.16.1 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

3.17 Client.View.HeartsCard Klassenreferenz

Abgeleitet von [Client.View.Card](#).

Öffentliche Methoden

- [HeartsCard](#) (HeartsID id)
- HeartsID [getCardID](#) ()

3.17.1 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

3.17.2 Beschreibung der Konstruktoren und Destruktoren

3.17.2.1 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
-----------	--------------------

3.17.3 Dokumentation der Elementfunktionen

3.17.3.1 HeartsID Client.View.HeartsCard.getCardID ()

Gibt die HeartsID der Karte zurück.

Rückgabe

HeartsID der Karte

3.18 Client.View.InputNumber Klassenreferenz

Abgeleitet von Observer.

Öffentliche Methoden

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

3.18.1 Ausführliche Beschreibung

Autor

m4nkey

3.18.2 Dokumentation der Elementfunktionen

3.18.2.1 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

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

3.19 Client.View.Language Enum-Referenz

3.19.1 Ausführliche Beschreibung

Autor

m4nkey

3.20 Client.View.Lobby Klassenreferenz

Abgeleitet von JFrame und Observer.

Öffentliche Methoden

- void [addJoinButtonListener](#) (ActionListener a)
- void [addHostButtonListener](#) (ActionListener a)
- void [addLeaveButtonListener](#) (ActionListener a)
- void [addChatMessageListener](#) (KeyListener k)
- void [setLanguage](#) ([Language](#) l)
- void [update](#) (Observable o, Object arg)

3.20.1 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

3.20.2 Dokumentation der Elementfunktionen

3.20.2.1 void Client.View.Lobby.addJoinButtonListener (ActionListener a)

Fügt einen ActionListener für den 'Join' Button hinzu.

Parameter

<i>a</i>	ein ActionListener
----------	--------------------

3.20.2.2 void Client.View.Lobby.addHostButtonListener (ActionListener *a*)

Fügt einen ActionListener für den 'Host' Button hinzu.

Parameter

<i>a</i>	ein ActionListener
----------	--------------------

3.20.2.3 void Client.View.Lobby.addLeaveButtonListener (ActionListener *a*)

Fügt einen ActionListener für den 'Leave' Button hinzu.

Parameter

<i>a</i>	ein ActionListener
----------	--------------------

3.20.2.4 void Client.View.Lobby.addChatMessageListener (KeyListener *k*)

Fügt einen KeyListener für das Nachricht-Senden-Feld der [Lobby](#) hinzu.

Parameter

<i>k</i>	
----------	--

3.20.2.5 void Client.View.Lobby.setLanguage (Language *l*)

Ändert die Sprache des Fensters.

Parameter

<i>l</i>	Sprache in Form des Language-Enums
----------	------------------------------------

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

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

Je nach dem in *arg* übergebenen ViewNotification-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

3.21 Client.View.Login Klassenreferenz

Abgeleitet von JFrame und Observer.

Öffentliche Methoden

- void [addConnectButtonListener](#) (ActionListener *a*)
- void [addLanguageSelectionListener](#) (ItemListener *i*)
- void [setLanguage](#) (Language *l*)
- void [update](#) (Observable *o*, Object *arg*)

3.21.1 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

3.21.2 Dokumentation der Elementfunktionen

3.21.2.1 void Client.View.Login.addConnectButtonListener (ActionListener *a*)

Fügt einen Listener für den 'Connect' Button des [Login](#) Fensters hinzu.

Parameter

<i>a</i>	ein ActionListener
----------	--------------------

3.21.2.2 void Client.View.Login.addLanguageSelectionListener (ItemListener *i*)

Fügt einen Listener für die Sprachauswahl des [Login](#) Fensters hinzu.

Parameter

<i>i</i>	ein ItemListener
----------	------------------

3.21.2.3 void Client.View.Login.setLanguage (Language *l*)

Ändert die Sprache des Fensters.

Parameter

<i>l</i>	Sprache in Form des Language-Enums
----------	------------------------------------

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

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

Je nach dem in *arg* übergebenen ViewNotification-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

3.22 Client.View.OtherPlayer Klassenreferenz

3.22.1 Ausführliche Beschreibung

Autor

m4nkey

3.23 Client.View.OwnHand Klassenreferenz

3.23.1 Ausführliche Beschreibung

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

Autor

m4nkey

3.24 Client.View.Password Klassenreferenz

Abgeleitet von JFrame und Observer.

Öffentliche Methoden

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

3.24.1 Ausführliche Beschreibung

Autor

M4nkey

3.24.2 Dokumentation der Elementfunktionen

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

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

Je nach dem in arg übergebenen ViewNotification-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

3.25 Client.View.ScoreWindow Klassenreferenz

Abgeleitet von Observer.

Öffentliche Methoden

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

3.25.1 Ausführliche Beschreibung

Autor

m4nkey

3.25.2 Dokumentation der Elementfunktionen

3.25.2.1 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

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

3.26 Client.View.Warning Klassenreferenz

Abgeleitet von [Observer](#).

Öffentliche Methoden

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

3.26.1 Ausführliche Beschreibung

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

Autor

m4nkey

3.26.2 Dokumentation der Elementfunktionen

3.26.2.1 `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

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

3.27 Client.View.WizCard Klassenreferenz

Abgeleitet von [Client.View.Card](#).

Öffentliche Methoden

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

3.27.1 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

3.27.2 Beschreibung der Konstruktoren und Destruktoren

3.27.2.1 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
-----------	-----------------

3.27.3 Dokumentation der Elementfunktionen

3.27.3.1 WizID Client.View.WizCard.getCardID ()

Gibt die WizID der Karte zurück.

Rückgabe

WizID der Karte

3.28 Client.ViewNotification Enum-Referenz

3.29 ComObjects.ComBeenKicked Klassenreferenz

Öffentliche Methoden

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

3.29.1 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.

3.29.2 Beschreibung der Konstruktoren und Destruktoren

3.29.2.1 ComObjects.ComBeenKicked.ComBeenKicked (String *message*)

Dies ist der Kontruktor für eine neue ComBeenKicked-Nachricht.

Parameter

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

3.29.3 Dokumentation der Elementfunktionen

3.29.3.1 String ComObjects.ComBeenKicked.getMessage ()

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

Rückgabe

die Nachricht.

3.30 ComObjects.ComChatMessage Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

Öffentliche Methoden

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

3.30.1 Ausführliche Beschreibung

Sie enthält eine Chatnachricht in Form eines Strings.

3.30.2 Beschreibung der Konstruktoren und Destrukturen

3.30.2.1 ComObjects.ComChatMessage.ComChatMessage (String message)

Dies ist der Konstruktor für eine neue ComChatMessage-Nachricht.

Parameter

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

3.30.3 Dokumentation der Elementfunktionen

3.30.3.1 String ComObjects.ComChatMessage.getChatMessage ()

Hier kann die versendete Nachricht von anderen Klassen ausgelesen werden.

Rückgabe

die Chatnachricht, die versendet wurde.

3.31 ComObjects.ComClientLeave Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

3.31.1 Ausführliche Beschreibung

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

3.32 ComObjects.ComClientQuit Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

3.32.1 Ausführliche Beschreibung

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

3.33 ComObjects.ComCreateGameRequest Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

Öffentliche Methoden

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

3.33.1 Ausführliche Beschreibung

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

3.33.2 Beschreibung der Konstruktoren und Destruktoren

3.33.2.1 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().

3.33.3 Dokumentation der Elementfunktionen

3.33.3.1 String ComObjects.ComCreateGameRequest.getGameName ()

Diese Methode gibt den Namen des Spiels zurück.

Rückgabe

den Spielnamen.

3.33.3.2 Enum ComObjects.ComCreateGameRequest.getRuleset ()

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

Rückgabe

das Regelwerk, welches benutzt wird.

3.33.3.3 boolean ComObjects.ComCreateGameRequest.hasPassword ()

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

Rückgabe

ob es ein Passwort gibt.

Wird benutzt von ComObjects.ComCreateGameRequest.ComCreateGameRequest().

3.33.3.4 String ComObjects.ComCreateGameRequest.getPassword ()

Gibt das Passwort zurück.

Sollte keines gesetzt sein, wird null zurück gegeben.

Rückgabe

das Passwort.

3.34 ComObjects.ComInitGameLobby Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

Öffentliche Methoden

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

3.34.1 Ausführliche Beschreibung

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

3.34.2 Beschreibung der Konstruktoren und Destruktoren

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

Dies ist der Konstruktor für eine neue ComInitGameLobby-Nachricht.

Parameter

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

3.34.3 Dokumentation der Elementfunktionen

3.34.3.1 Object ComObjects.ComInitGameLobby.getPlayerList ()

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

Rückgabe

die Liste der Spieler.

3.35 ComObjects.ComInitLobby Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

Öffentliche Methoden

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

3.35.1 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*.

3.35.2 Beschreibung der Konstruktoren und Destruktoren

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

Dies ist der Konstruktor 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.

3.35.3 Dokumentation der Elementfunktionen

3.35.3.1 List ComObjects.ComInitLobby.getPlayerList ()

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

Rückgabe

die Liste der Spieler.

3.35.3.2 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.

3.36 ComObjects.ComJoinRequest Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

Öffentliche Methoden

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

3.36.1 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.

3.36.2 Beschreibung der Konstruktoren und Destruktoren

3.36.2.1 ComObjects.ComJoinRequest.ComJoinRequest (String *gameMasterName*)

Dies ist der Konstruktor für eine neue ComJoinRequest-Nachricht.

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

Parameter

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

3.36.3 Dokumentation der Elementfunktionen

3.36.3.1 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.

3.37 ComObjects.ComKickPlayerRequest Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

Öffentliche Methoden

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

3.37.1 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.

3.37.2 Beschreibung der Konstruktoren und Destruktoren

3.37.2.1 ComObjects.ComKickPlayerRequest.ComKickPlayerRequest (String playerName)

Dies ist der Konstruktor für eine neue ComKickPlayerRequest-Nachricht.

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

Parameter

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

3.37.3 Dokumentation der Elementfunktionen

3.37.3.1 String ComObjects.ComKickPlayerRequest.getPlayerName ()

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

Rückgabe

den Spielernamen.

3.38 ComObjects.ComLobbyUpdateGamelist Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

Öffentliche Methoden

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

3.38.1 Ausführliche Beschreibung

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

3.38.2 Beschreibung der Konstruktoren und Destruktoren

3.38.2.1 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.

3.38.3 Dokumentation der Elementfunktionen

3.38.3.1 boolean ComObjects.ComLobbyUpdateGamelist.isRemoveFlag ()

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

Rückgabe

ob das Spiel gelöscht wird.

3.38.3.2 GameServerRepresentation ComObjects.ComLobbyUpdateGamelist.getGameServer ()

Diese Methode liefert das Spiel, das geupdated werden soll.

Rückgabe

das Spiel.

3.39 ComObjects.ComLoginRequest Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

Öffentliche Methoden

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

3.39.1 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.

3.39.2 Beschreibung der Konstruktoren und Destrukturen**3.39.2.1 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.
-------------	---

3.39.3 Dokumentation der Elementfunktionen**3.39.3.1 String ComObjects.ComLoginRequest.getPlayerName ()**

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

Dieser muss auf Eindeutigkeit geprüft werden.

Rückgabe

den Spielernamen.

3.40 ComObjects.ComObject Klassenreferenz

Abgeleitet von Serializable.

Basisklasse für [ComObjects.ComChatMessage](#), [ComObjects.ComClientLeave](#), [ComObjects.ComClientQuit](#), [ComObjects.ComCreateGameRequest](#), [ComObjects.ComInitGameLobby](#), [ComObjects.ComInitLobby](#), [ComObjects.ComJoinRequest](#), [ComObjects.ComKickPlayerRequest](#), [ComObjects.ComLobbyUpdateGamelist](#), [ComObjects.ComLoginRequest](#), [ComObjects.ComRuleset](#), [ComObjects.ComServerAcknowledgement](#), [ComObjects.ComStartGame](#), [ComObjects.ComUpdatePlayerlist](#) und [ComObjects.ComWarning](#).

3.41 ComObjects.ComRuleset Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

Öffentliche Methoden

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

3.41.1 Ausführliche Beschreibung

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

3.41.2 Beschreibung der Konstruktoren und Destruktoren**3.41.2.1 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.
-----------------------	---

3.41.3 Dokumentation der Elementfunktionen**3.41.3.1 RulesetMessage ComObjects.ComRuleset.getRulesetMessage ()**

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

Rückgabe

die Nachricht.

3.42 ComObjects.ComServerAcknowledgement Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

3.43 ComObjects.ComStartGame Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

3.43.1 Ausführliche Beschreibung

Sie wird versendet, wenn ein Spiel gestartet werden soll.

3.44 ComObjects.ComUpdatePlayerlist Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

Öffentliche Methoden

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

3.44.1 Ausführliche Beschreibung

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

3.44.2 Beschreibung der Konstruktoren und Destruktoren**3.44.2.1 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.

3.44.3 Dokumentation der Elementfunktionen**3.44.3.1 String ComObjects.ComUpdatePlayerlist.getPlayerName ()**

Diese Methode gibt den Namen des Spielers zurück.

Rückgabe

den Spielernamen.

3.44.3.2 boolean ComObjects.ComUpdatePlayerlist.isRemoveFlag ()

Diese Methode gibt zur ck, ob der Spieler aus der Liste gel scht werden soll oder nicht.

R ckgabe

ob der Spieler gel scht werden soll.

3.45 ComObjects.ComWarning Klassenreferenz

Abgeleitet von [ComObjects.ComObject](#).

 ffentliche Methoden

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

3.45.1 Ausf hrliche Beschreibung

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

3.45.2 Beschreibung der Konstruktoren und Destruktoren

3.45.2.1 ComObjects.ComWarning.ComWarning (String warning)

Dies ist der Konstruktor einer neuen ComWarning-Nachricht.

Er enth lt eine Warnung an den Spieler, wenn ein Fehler passiert.

Parameter

<i>warning</i>	ist die Warnung, die der Spieler erh�lt.
----------------	--

3.45.3 Dokumentation der Elementfunktionen

3.45.3.1 String ComObjects.ComWarning.getWarning ()

Diese Methode gibt die Nachricht zur ck, die dem Spieler den Fehler mitteilt.

R ckgabe

die Warnnachricht.

3.46 ComObjects.MsgCard Klassenreferenz

Abgeleitet von [ComObjects.RulesetMessage](#).

 ffentliche Methoden

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

3.46.1 Ausf hrliche Beschreibung

Sie beinhaltet die ausgespielte Karte eines Spielers.

3.46.2 Beschreibung der Konstruktoren und Destruktoren

3.46.2.1 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.
-------------	----------------

3.46.3 Dokumentation der Elementfunktionen

3.46.3.1 Card ComObjects.MsgCard.getCard ()

Diese Methode gibt die ausgespielte Karte des Spielers zurück.

Rückgabe

die Karte.

3.47 ComObjects.MsgCardRequest Klassenreferenz

3.47.1 Ausführliche Beschreibung

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

3.48 ComObjects.MsgGameEnd Klassenreferenz

Abgeleitet von [ComObjects.RulesetMessage](#).

3.48.1 Ausführliche Beschreibung

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

3.49 ComObjects.MsgMultiCards Klassenreferenz

Abgeleitet von [ComObjects.RulesetMessage](#).

Öffentliche Methoden

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

3.49.1 Ausführliche Beschreibung

Sie liefert mehrere Karten zum Tausch für das Regelwerk Hearts.

3.49.2 Beschreibung der Konstruktoren und Destruktoren

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

Dies ist der Kontruktor für eine neue MsgMultiCards-Nachricht.

Parameter

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

3.49.3 Dokumentation der Elementfunktionen

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

Gibt die Liste der gewählten Karten zurück.

Rückgabe

die Liste der Karten.

3.50 ComObjects.MsgMultiCardsRequest Klassenreferenz

Abgeleitet von [ComObjects.RulesetMessage](#).

Öffentliche Methoden

- int [getCount](#) ()

3.50.1 Dokumentation der Elementfunktionen

3.50.1.1 int ComObjects.MsgMultiCardsRequest.getCount ()

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

Rückgabe

die Anzahl der Karten.

3.51 ComObjects.MsgMultipleCardsRequest Klassenreferenz

Abgeleitet von [ComObjects.RulesetMessage](#).

3.52 ComObjects.MsgNumber Klassenreferenz

Abgeleitet von [ComObjects.RulesetMessage](#).

Öffentliche Methoden

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

3.52.1 Ausführliche Beschreibung

Sie enthält eine Zahl.

3.52.2 Beschreibung der Konstruktoren und Destrukturen

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

Dies ist der Kontruktor für eine neue MsgNumber-Nachricht.

Parameter

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

3.52.3 Dokumentation der Elementfunktionen

3.52.3.1 `int ComObjects.MsgNumber.getNumber ()`

Diese Methode liefert die Eingabe eines Spielers.

Rückgabe

eine Zahl, die Eingabe des Spielers.

3.53 ComObjects.MsgNumberRequest Klassenreferenz

Abgeleitet von [ComObjects.RulesetMessage](#).

3.54 ComObjects.MsgSelection Klassenreferenz

Abgeleitet von [ComObjects.RulesetMessage](#).

Öffentliche Methoden

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

3.54.1 Ausführliche Beschreibung

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

3.54.2 Beschreibung der Konstruktoren und Destruktoren

3.54.2.1 `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.
------------------	--

3.54.3 Dokumentation der Elementfunktionen

3.54.3.1 `int ComObjects.MsgSelection.getSelection ()`

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

Rückgabe

die Auswahl.

3.55 ComObjects.MsgSelectionRequest Klassenreferenz

Abgeleitet von [ComObjects.RulesetMessage](#).

3.55.1 Ausführliche Beschreibung

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

3.56 ComObjects.MsgUser Klassenreferenz

Abgeleitet von [ComObjects.RulesetMessage](#).

Öffentliche Methoden

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

3.56.1 Ausführliche Beschreibung

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

3.56.2 Beschreibung der Konstruktoren und Destruktoren

3.56.2.1 ComObjects.MsgUser.MsgUser ([GameClientUpdate](#) gameClientUpdate)

Dies ist der Konstruktor einer neuen MsgUser-Nachricht.

Parameter

<i>gameClientUpdate</i>	ist der aktuelle Spielstand.
-------------------------	------------------------------

3.56.3 Dokumentation der Elementfunktionen

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

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

Rückgabe

den aktuellen Spielzustand.

3.57 ComObjects.RulesetMessage Klassenreferenz

Abgeleitet von [Serializable](#).

Basisklasse für [ComObjects.MsgCard](#), [ComObjects.MsgGameEnd](#), [ComObjects.MsgMultiCards](#), [ComObjects.MsgMultiCardsRequest](#), [ComObjects.MsgMultipleCardsRequest](#), [ComObjects.MsgNumber](#), [ComObjects.MsgNumberRequest](#), [ComObjects.MsgSelection](#), [ComObjects.MsgSelectionRequest](#) und [ComObjects.MsgUser](#).

3.57.1 Ausführliche Beschreibung

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

3.58 Ruleset.Card Schnittstellenreferenz

Basisklasse für [Ruleset.HeartsCard](#) und [Ruleset.WizardCard](#).

Öffentliche Methoden

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

3.58.1 Dokumentation der Elementfunktionen

3.58.1.1 int Ruleset.Card.getValue ()

Gibt den Wert der Karte zurück.

Rückgabe

Der Wert der Karte

Implementiert in [Ruleset.WizardCard](#) und [Ruleset.HeartsCard](#).

3.58.1.2 Colour Ruleset.Card.getColour ()

Gibt die Farbe der Karte zurück.

Rückgabe

Die Farbe der Karte

Implementiert in [Ruleset.WizardCard](#) und [Ruleset.HeartsCard](#).

3.59 Ruleset.CardDeck Klassenreferenz

3.60 Ruleset.CardDeckBuilder Klassenreferenz

3.61 Ruleset.ClientHearts Klassenreferenz

Abgeleitet von [Ruleset.ClientRuleset](#).

Öffentliche Methoden

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

Weitere Geerbte Elemente

3.61.1 Dokumentation der Elementfunktionen

3.61.1.1 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](#).

3.62 Ruleset.ClientRuleset Klassenreferenz

Basisklasse für [Ruleset.ClientHearts](#) und [Ruleset.ClientWizard](#).

Öffentliche Methoden

- `List< OtherData > getOtherPlayerData ()`
- `PlayerState getCurrentPlayer ()`
- `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)`

3.62.1 Ausführliche Beschreibung

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

3.62.2 Dokumentation der Elementfunktionen

3.62.2.1 `List<OtherData> Ruleset.ClientRuleset.getOtherPlayerData ()`

Holt die Spieldaten der anderen Spieler.

Rückgabe

`otherPlayerData` Die Spieldaten der anderen Spieler

3.62.2.2 `PlayerState Ruleset.ClientRuleset.getCurrentPlayer ()`

Gibt den Spieler der momentan am Zug ist zurück.

Rückgabe

Der momentane Spieler

3.62.2.3 `void Ruleset.ClientRuleset.resolveMessage (MsgUser clientUpdate)`

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

Parameter

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

3.62.2.4 `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
------------------------	--------------------------

3.62.2.5 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
------------------------------	--------------------------

3.62.2.6 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
------------------	--------------------------

3.62.2.7 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
---------------------	--------------------------

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

Schickt eine Nachricht übers Model an den Server.

Parameter

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

3.63 Ruleset.ClientWizard Klassenreferenz

Abgeleitet von [Ruleset.ClientRuleset](#).

Öffentliche Methoden

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

Weitere Geerbte Elemente

3.63.1 Dokumentation der Elementfunktionen

3.63.1.1 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](#).

3.64 Ruleset.Colour Enum-Referenz

3.65 Ruleset.GameClientUpdate Klassenreferenz

Öffentliche Methoden

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

3.65.1 Ausführliche Beschreibung

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

3.65.2 Dokumentation der Elementfunktionen

3.65.2.1 List<Card> Ruleset.GameClientUpdate.getOwnHand ()

Holt die Karten die der Client auf der Hand hat.

Rückgabe

ownHand Die Hand des Clients

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

Holt die gespielten Karten auf dem Ablagestapel.

Rückgabe

discardPile Die gespielten Karten

3.65.2.3 OtherData Ruleset.GameClientUpdate.getOwnData ()

Holt die zusätzlichen Spieldaten des Client.

Rückgabe

ownData Die Spieldaten des Clients

3.65.2.4 List<OtherData> Ruleset.GameClientUpdate.getOtherPlayerData ()

Holt die Spieldaten der anderen Spieler.

Rückgabe

otherPlayerData Die Spieldaten der anderen Spieler

3.65.2.5 PlayerState Ruleset.GameClientUpdate.getCurrentPlayer ()

Gibt den Spieler der momentan am Zug ist zurück.

Rückgabe

Der momentane Spieler

3.66 Ruleset.GamePhase Enum-Referenz

3.67 Ruleset.GameState Klassenreferenz

Öffentliche Methoden

- [GameState](#) ([RulesetType](#) ruleset, List< [Card](#) > deck)
- boolean [addPlayerToGame](#) (String name)
- boolean [setFirstPlayer](#) ([PlayerState](#) player)
- [PlayerState](#) [getFirstPlayer](#) ()
- boolean [setCurrentPlayer](#) ([PlayerState](#) player)
- [PlayerState](#) [getCurrentPlayer](#) ()
- List< [Card](#) > [getCardsLeftInDeck](#) ()
- Map< String, [Card](#) > [getPlayedCards](#) ()
- [PlayerState](#) [getPlayerState](#) (String name)
- void [setTrumpCard](#) ([Card](#) trumpCard)
- [Card](#) [getTrumpCard](#) ()
- int [getNumberOfPlayedCards](#) ()
- List< [Card](#) > [getPlayerCards](#) (String name)
- boolean [dealCards](#) (String name, int number)
- boolean [giveACard](#) (String name, [Card](#) card)
- boolean [playCard](#) ([Card](#) card)

3.67.1 Ausführliche Beschreibung

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

3.67.2 Beschreibung der Konstruktoren und Destrukturen

3.67.2.1 Ruleset.GameState.GameState (RulesetType ruleset, List< Card > deck)

Erstellt eine GameStateklasse.

Parameter

<i>ruleset</i>	Der Regelwerktyp des Spiels
<i>deck</i>	Das Kartendeck im Spiel

Benutzt Ruleset.GamePhase.Start.

3.67.3 Dokumentation der Elementfunktionen

3.67.3.1 boolean Ruleset.GameState.addPlayerToGame (String name)

Fügt den Spieler ins Spiel hinein, falls er nicht schon im Spiel ist.

Parameter

<i>name</i>	
-------------	--

3.67.3.2 boolean Ruleset.GameState.setFirstPlayer (PlayerState player)

Setzt einen neuen Spieler als firstPlayer.

Parameter

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

3.67.3.3 PlayerState Ruleset.GameState.getFirstPlayer ()

Holt den Spieler als erster am Zug war.

Rückgabe

firstPlayer Der Spielzustand des Spielers der als erster am Zug war

3.67.3.4 boolean Ruleset.GameState.setCurrentPlayer (PlayerState player)

Setzt einen neuen Spieler als currentPlayer.

Parameter

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

3.67.3.5 PlayerState Ruleset.GameState.getCurrentPlayer ()

Holt den Spieler der momentan am Zug ist.

Rückgabe

currentPlayer Der Spielzustand des Spielers der grad am Zug ist

3.67.3.6 List<Card> Ruleset.GameState.getCardsLeftInDeck ()

Holt die Karten die noch im Aufnahmestapel sind.

Rückgabe

deck Holt die Karten die noch im Aufnahmestapel sind

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

Holt die gespielten Karten im Ablagestapel.

Rückgabe

discardPile Die gespielten Karten

3.67.3.8 PlayerState Ruleset.GameState.getPlayerState (String name)

Holt einen bestimmten Spieler.

Parameter

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

Rückgabe

player Der Spielzustand des Spielers

3.67.3.9 void Ruleset.GameState.setTrumpCard (Card trumpCard)

Setzt die Trumpfkarte.

Parameter

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

3.67.3.10 Card Ruleset.GameState.getTrumpCard ()

Holt die momentane Trumpfkarte im Spiel.

Rückgabe

trumpCard Die momentane Trumpfkarte

3.67.3.11 int Ruleset.GameState.getNumberOfPlayedCards ()

Holt die Anzahl der gespielten Karten.

Rückgabe

Die Anzahl der gespielten Karten

3.67.3.12 List<Card> Ruleset.GameState.getPlayerCards (String name)

Holt die Karten eines Spielers.

Parameter

<i>name</i>	Der Name vom Spieler
-------------	----------------------

Rückgabe

Karten

3.67.3.13 boolean Ruleset.GameState.dealCards (String name, int number)

Deals a number of cards from the top of the deck.

Parameter

<i>name</i>	Name of the Player who gets the cards
<i>number</i>	The number of cards

Rückgabe

True if a player has no cards, false if he does

3.67.3.14 boolean Ruleset.GameState.giveACard (String name, Card card)

Gives one specific card of the deck to a Player.

Parameter

<i>name</i>	The name of the Player
-------------	------------------------

Rückgabe

true if the card is in the deck

3.67.3.15 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

3.68 Ruleset.HearthsDeck Klassenreferenz**3.69 Ruleset.HeartsCard Enum-Referenz**

Abgeleitet von [Ruleset.Card](#).

Öffentliche Methoden

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

3.69.1 Dokumentation der Elementfunktionen**3.69.1.1 int Ruleset.HeartsCard.getValue ()**

Gibt den Wert der Karte zurück.

Rückgabe

Der Wert der Karte

Implementiert [Ruleset.Card](#).

3.69.1.2 Colour Ruleset.HeartsCard.getColour ()

Gibt die Farbe der Karte zurück.

Rückgabe

Die Farbe der Karte

Implementiert [Ruleset.Card](#).

3.70 Ruleset.HeartsData Klassenreferenz

Abgeleitet von [Ruleset.OtherData](#).

Öffentliche Methoden

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

3.70.1 Dokumentation der Elementfunktionen**3.70.1.1 int Ruleset.HeartsData.getCompletePoints ()**

Holt den gesamten Punktestand eines Spielers.

Rückgabe

Der Gesamtpunktstand eines Spielers

3.70.1.2 int Ruleset.HeartsData.getCurrentPoints ()

Holt den momentanen Punktestand eines Spielers.

Rückgabe

Der momentane Punktestand eines Spielers

3.71 Ruleset.isValidMoveWizard Klassenreferenz**3.72 Ruleset.OtherData Klassenreferenz**

Basisklasse für [Ruleset.HeartsData](#) und [Ruleset.WizData](#).

Öffentliche Methoden

- [OtherData](#) (String name)
- String [getname](#) ()

3.72.1 Beschreibung der Konstruktoren und Destruktoren**3.72.1.1 Ruleset.OtherData.OtherData (String name)**

Erzeugt die zusätzlichen Daten eines Spielers.

Parameter

<i>name</i>	Der Name des Spielers dem die Daten gehören
-------------	---

3.72.2 Dokumentation der Elementfunktionen**3.72.2.1 String Ruleset.OtherData.getname ()**

Holt den Namen des Spielers.

Rückgabe

name Der Name des Spielers

3.73 Ruleset.PlayerState Klassenreferenz**Öffentliche Methoden**

- [PlayerState](#) (String name, [RulesetType](#) ruleset)
- String [getName](#) ()
- List< [Card](#) > [getHand](#) ()
- [OtherData](#) [getOtherData](#) ()
- void [addCard](#) ([Card](#) card)
- boolean [removeCard](#) ([Card](#) card)

3.73.1 Ausführliche Beschreibung

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

3.73.2 Beschreibung der Konstruktoren und Destruktoren

3.73.2.1 Ruleset.PlayerState.PlayerState (String *name*, RulesetType *ruleset*)

Erstellt einen [PlayerState](#).

Parameter

<i>name</i>	Der Name des Spielers
<i>ruleset</i>	Der Typ des Spiels

3.73.3 Dokumentation der Elementfunktionen

3.73.3.1 String Ruleset.PlayerState.getName ()

Holt den namen eines Spielers.

Rückgabe

name Der Name des Spielers

3.73.3.2 List<Card> Ruleset.PlayerState.getHand ()

Holt die Kartenhand des Spielers.

Rückgabe

ownHand Die Kartenhand des Spielers

3.73.3.3 OtherData Ruleset.PlayerState.getOtherData ()

Holt die zusätzlichen Informationen des Spielers.

Rückgabe

ownHand Die zusätzlichen Informationen des Spielers

3.73.3.4 void Ruleset.PlayerState.addCard (Card *card*)

Gibt dem Spieler eine Karte.

Parameter

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

3.73.3.5 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

3.74 Ruleset.RulesetType Enum-Referenz

3.75 Ruleset.ServerHearts Klassenreferenz

Abgeleitet von [Ruleset.ServerRuleset](#).

Öffentliche Methoden

- void [resolveMessage](#) ([MsgMultiCards](#) msgMultiCard, String name)

Geschützte Methoden

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

3.75.1 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.

3.75.2 Dokumentation der Elementfunktionen

3.75.2.1 void Ruleset.ServerHearts.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

3.75.2.2 boolean Ruleset.ServerHearts.isValidMove ([Card](#) card) [protected],[virtual]

Prüft ob ein gemachter Zug in einem Spiel gültig war, wenn nicht wird an den Spieler erneut eine MsgCardRequest.

Parameter

<i>card</i>	Die Karte die gespielt wurde
<i>name</i>	Der Name des Spielers

Rückgabe

true falls Zug gültig und false wenn nicht

Implementiert [Ruleset.ServerRuleset](#).

3.76 Ruleset.ServerRuleset Klassenreferenz

Basisklasse für [Ruleset.ServerHearts](#) und [Ruleset.ServerWizard](#).

Öffentliche Methoden

- [ServerRuleset](#) ([RulesetType](#) ruleset, int min, int max)
- [RulesetType](#) [getRulesetType](#) ()
- int [getMinPlayers](#) ()
- int [getMaxPlayers](#) ()
- void [addPlayerToGame](#) (String name)
- void [resolveMessage](#) ([MsgCard](#) msgCard, String name)

Geschützte Methoden

- boolean [setFirstPlayer](#) ([PlayerState](#) player)
- [PlayerState](#) [getFirstPlayer](#) ()
- boolean [setCurrentPlayer](#) ([PlayerState](#) player)
- [PlayerState](#) [getCurrentPlayer](#) ()
- [PlayerState](#) [getPlayerState](#) (String name)
- List< [Card](#) > [getPlayerCards](#) (String name)
- void [send](#) ([RulesetMessage](#) message, String name)
- void [broadcast](#) ([RulesetMessage](#) message)
- boolean [dealCards](#) (String name, int number)
- boolean [giveACard](#) (String name, [Card](#) card)
- abstract boolean [isValidMove](#) ([Card](#) card)

3.76.1 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.

3.76.2 Beschreibung der Konstruktoren und Destruktoren

3.76.2.1 [Ruleset.ServerRuleset.ServerRuleset](#) ([RulesetType](#) ruleset, int min, int max)

Erstellt ein [ServerRuleset](#).

Parameter

<i>ruleset</i>	Der Spieltyp
----------------	--------------

3.76.3 Dokumentation der Elementfunktionen

3.76.3.1 [RulesetType](#) [Ruleset.ServerRuleset.getRulesetType](#) ()

Gibt den Typ des Regelwerks zurück.

Rückgabe

Der Typ vom Regelwerk

3.76.3.2 int [Ruleset.ServerRuleset.getMinPlayers](#) ()

Gibt die Mindestanzahl an Spielern zurück für dieses Spiel.

Rückgabe

Die Mindestanzahl an Spielern

3.76.3.3 `int Ruleset.ServerRuleset.getMaxPlayers ()`

Gibt die Maximale anzahl an Spielern zurück.

Rückgabe

Die maximale Anzahl an Spielern

3.76.3.4 `boolean Ruleset.ServerRuleset.setFirstPlayer (PlayerState player)` `[protected]`

Setzt den Spieler der als Erster am Zug ist, im Gamestate.

Rückgabe

false wenn der selbe Spieler nochmal als firstPlayer gesetzt wird

3.76.3.5 `PlayerState Ruleset.ServerRuleset.getFirstPlayer ()` `[protected]`

Holt den Spieler der als erster am Zug war.

Rückgabe

firstPlayer Der Spielzustand des Spielers der als erster am Zug war

3.76.3.6 `boolean Ruleset.ServerRuleset.setCurrentPlayer (PlayerState player)` `[protected]`

Setzt den Spieler der am Nächsten am Zug ist, im Gamestate.

Rückgabe

false wenn der selbe Spieler nochmal als currentPlayer gesetzt wird

3.76.3.7 `PlayerState Ruleset.ServerRuleset.getCurrentPlayer ()` `[protected]`

Holt den Spieler der gerade am Zug ist.

Rückgabe

currentPlayer Der Spielzustand des Spielers der grad am Zug ist

3.76.3.8 `void Ruleset.ServerRuleset.addPlayerToGame (String name)`

Fügt einen Spieler ins Spiel ein.

Parameter

<i>name</i>	Der name vom Spieler
-------------	----------------------

3.76.3.9 `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

3.76.3.10 List<Card> Ruleset.ServerRuleset.getPlayerCards (String *name*) [protected]

Holt die Spielkarten eines Spielers.

Parameter

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

Rückgabe

Die Spielkarten des Spielers

3.76.3.11 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

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

Schickt eine Nachricht an alle Spieler.

Parameter

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

3.76.3.13 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

3.76.3.14 boolean Ruleset.ServerRuleset.dealCards (String *name*, int *number*) [protected]

Deals a number of cards from the top of the deck.

Parameter

<i>name</i>	Name of the Player who gets the cards
<i>number</i>	The number of cards

Rückgabe

True if a player has no cards, false if he does

3.76.3.15 boolean Ruleset.ServerRuleset.giveACard (String *name*, Card *card*) [protected]

Gives one specific card of the deck to a Player.

Parameter

<i>name</i>	The name of the Player
-------------	------------------------

Rückgabe

true if the card is in the deck

3.76.3.16 abstract boolean Ruleset.ServerRuleset.isValidMove (Card card) [protected],[pure virtual]

Prüft ob ein gemachter Zug in einem Spiel gültig war, wenn nicht wird an den Spieler erneut eine MsgCardRequest.

Parameter

<i>card</i>	Die Karte die gespielt wurde
<i>name</i>	Der Name des Spielers

Rückgabe

true falls Zug gültig und false wenn nicht

Implementiert in [Ruleset.ServerWizard](#) und [Ruleset.ServerHearts](#).

3.77 Ruleset.ServerWizard Klassenreferenz

Abgeleitet von [Ruleset.ServerRuleset](#).

Öffentliche Methoden

- void [resolveMessage](#) (MsgNumber msgNumber, String name)
- void [resolveMessage](#) (MsgSelection msgSelection, String name)

Geschützte Methoden

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

3.77.1 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.

3.77.2 Dokumentation der Elementfunktionen

3.77.2.1 void Ruleset.ServerWizard.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

3.77.2.2 void Ruleset.ServerWizard.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

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

Prüft ob ein gemachter Zug in einem Spiel gültig war, wenn nicht wird an den Spieler erneut eine `MsgCardRequest`.

Parameter

<i>card</i>	Die Karte die gespielt wurde
<i>name</i>	Der Name des Spielers

Rückgabe

true falls Zug gültig und false wenn nicht

Implementiert [Ruleset.ServerRuleset](#).

3.78 `Ruleset.WizardCard` Enum-Referenz

Abgeleitet von [Ruleset.Card](#).

Öffentliche Methoden

- `int getValue ()`
- `Colour getColour ()`

3.78.1 Dokumentation der Elementfunktionen

3.78.1.1 `int Ruleset.WizardCard.getValue ()`

Gibt den Wert der Karte zurück.

Rückgabe

Der Wert der Karte

Implementiert [Ruleset.Card](#).

3.78.1.2 `Colour Ruleset.WizardCard.getColour ()`

Gibt die Farbe der Karte zurück.

Rückgabe

Die Farbe der Karte

Implementiert [Ruleset.Card](#).

3.79 `Ruleset.WizardDeck` Klassenreferenz3.80 `Ruleset.WizData` Klassenreferenz

Abgeleitet von [Ruleset.OtherData](#).

Öffentliche Methoden

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

3.80.1 Dokumentation der Elementfunktionen

3.80.1.1 int Ruleset.WizData.getAnnouncedTricks ()

Holt die angesagten Stiche des Spielers.

Rückgabe

announcedTricks Die angesagten Stiche

3.80.1.2 int Ruleset.WizData.getAchievedTricks ()

Holt die erreichten Stiche des Spielers.

Rückgabe

achievedTricks Die gemachten Stiche eines Spielers

3.80.1.3 int Ruleset.WizData.getPoints ()

Holt den Punktestand des Spielers.

Rückgabe

points Der Punktestand des Spielers

3.80.1.4 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
----------------------	--

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

Setzt den Punktestand eines Spielers.

Parameter

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

3.81 Server.ClientListenerThread Klassenreferenz

Abgeleitet von Runnable.

3.82 Server.GameServer Klassenreferenz

Abgeleitet von [Server.Server](#).

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** ()

3.82.1 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

3.82.2 Beschreibung der Konstruktoren und Destruktoren

3.82.2.1 **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

3.82.3 Dokumentation der Elementfunktionen

3.82.3.1 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
---------------	---

3.82.3.2 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
---------------	--

3.82.3.3 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

3.82.3.4 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

3.82.3.5 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

3.82.3.6 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

3.82.3.7 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

3.82.3.8 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

3.82.3.9 ComInitGameLobby Server.GameServer.initLobby ()

Baut ein neues ComInitGameLobby Objekt und gibt es zurück.

Rückgabe

Gibt das ComInitGameLobby Objekt zurück

3.83 Server.GameServerRepresentation Klassenreferenz

Öffentliche Methoden

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

3.83.1 Ausführliche Beschreibung

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

Autor

Viktoria

3.83.2 Beschreibung der Konstruktoren und Destruktoren

3.83.2.1 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

3.84 Server.LobbyServer Klassenreferenz

Abgeleitet von [Server.Server](#).

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](#) ()

3.84.1 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

3.84.2 Dokumentation der Elementfunktionen

3.84.2.1 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

3.84.2.2 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

3.84.2.3 void Server.LobbyServer.receiveMessage (Player player, ComCreateGameRequest create)

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

Durch broadcast wird sowohl im [LobbyServer](#) als auch im [GameServer](#) ein ComUpdatePlayerlist verschickt. Zustzlich 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

3.84.2.4 void Server.LobbyServer.receiveMessage (Player player, ComJoinRequest join)

Diese Methode fgt einen [Player](#) dem entsprechenden [GameServer](#) hinzu.

Falls das Passwort nicht leer ist wird geprft, ob es mit dem Passwort des Spieles bereinstimmt, wenn nicht, wird ein ComWarning an den Client geschickt. Ansonsten wird und der [Player](#) dem, durch Namen des Spielleiters identifizierten, durch Aufruf von changeServer Gameserver bergeben. Durch broadcast wird sowohl im [LobbyServer](#) als auch im [GameServer](#) ein ComUpdatePlayerlist verschickt. Zustzlich wird dem joinendenClient mit sendToPlayer 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

3.84.2.5 void Server.LobbyServer.receiveMessage (Player player, ComLoginRequest login)

Diese Methode berprft, 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 eingefgt. Der Name wird in das Set names eingefgt. 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 enthlt

3.84.2.6 ComInitLobby Server.LobbyServer.initLobby ()

Baut ein neues ComInitLobby Objekt und gibt es zurck.

Rckgabe

Gibt das ComInitLobby Objekt zurck

3.85 Server.LobbyServer.ClientListenerThread Klassenreferenz

Abgeleitet von Runnable.

3.85.1 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

3.86 Server.Player Klassenreferenz

Abgeleitet von Runnable.

Öffentliche Methoden

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

3.86.1 Ausführliche Beschreibung

Sie verwaltet für die Dauer einer Serververbindung die Verbindung zum Client

Autor

Viktoria

3.86.2 Beschreibung der Konstruktoren und Destruktoren

3.86.2.1 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

3.86.3 Dokumentation der Elementfunktionen

3.86.3.1 void Server.Player.send ([ComObject](#) com) throws IOException

Diese Methode schickt ein ComObjekt an den Client.

Parameter

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

Ausnahmebehandlung

<i>IOException</i>	wenn der Output nicht funktioniert
--------------------	------------------------------------

3.86.3.2 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
------------------	-------------------------------------

3.86.3.3 String Server.Player.getName ()

Getter-Methode für den Benutzernamen.

Rückgabe

gibt den Benutzernamen des Spielers zurück

3.86.3.4 void Server.Player.setName (String newName)

Setter-Methode für den Benutzernamen.

Parameter

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

3.87 Server.Server Klassenreferenz

Basisklasse für [Server.GameServer](#) und [Server.LobbyServer](#).

Öffentliche Methoden

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

3.87.1 Ausführliche Beschreibung

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

Autor

Viktoria

3.87.2 Dokumentation der Elementfunktionen

3.87.2.1 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 ComObject vom Client verschickt wurde

3.87.2.2 synchronized void Server.Server.sendToPlayer (String *name*, ComObject *com*) throws IOException

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. Ist der Name oder das ComObject leer wird nichts verschickt.

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

Ausnahmebehandlung

<i>IOException</i>	wenn der Output nicht funktioniert
--------------------	------------------------------------

Benutzt Server.Server.playerSet.

3.87.2.3 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
---------------	---

3.87.2.4 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
---------------	--

3.87.2.5 synchronized void Server.Server.broadcast (ComObject *com*) throws IOException

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

Ist das ComObject leer, passiert nichts.

Parameter

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

Ausnahmebehandlung

<i>IOException</i>	wenn der Output nicht funktioniert
--------------------	------------------------------------

Benutzt Server.Server.playerSet.

3.88 Server.ServerMain Klassenreferenz

Öffentliche, statische Methoden

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

3.88.1 Ausführliche Beschreibung

Autor

Viktoria

3.88.2 Dokumentation der Elementfunktionen

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

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

Parameter

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

Index

- addCard
 - Ruleset::PlayerState, 50
- addChatMessageListener
 - Client::View::Lobby, 20
- addConnectButtonListener
 - Client::View::Login, 21
- addHostButtonListener
 - Client::View::Lobby, 20
- addJoinButtonListener
 - Client::View::Lobby, 19
- addLanguageSelectionListener
 - Client::View::Login, 21
- addLeaveButtonListener
 - Client::View::Lobby, 20
- addPlayer
 - Server::GameServer, 58
 - Server::Server, 65
- addPlayerToGame
 - Ruleset::GameState, 45
 - Ruleset::ServerRuleset, 53
- announceTricks
 - Ruleset::WizData, 57
- broadcast
 - Ruleset::ServerRuleset, 54
 - Server::Server, 65
- Card
 - Client::View::Card, 13
- changeServer
 - Server::Player, 64
- Client.CardID, 9
- Client.ClientController, 9
- Client.ClientMain, 9
- Client.ClientModel, 10
- Client.ClientState, 12
- Client.MVMessages, 12
- Client.MessageListenerThread, 12
- Client.View.Card, 12
- Client.View.ChooseCards, 13
- Client.View.ChooseItem, 13
- Client.View.CreateGame, 14
- Client.View.DiscardPile, 15
- Client.View.DrawDeck, 15
- Client.View.Game, 15
- Client.View.GameLobby, 17
- Client.View.GamePanel, 17
- Client.View.HeartsCard, 18
- Client.View.InputNumber, 18
- Client.View.Language, 19
- Client.View.Lobby, 19
- Client.View.Login, 20
- Client.View.OtherPlayer, 21
- Client.View.OwnHand, 21
- Client.View.Password, 22
- Client.View.ScoreWindow, 22
- Client.View.Warning, 23
- Client.View.WizCard, 23
- Client.ViewNotification, 24
- Client::ClientMain
 - main, 9
- Client::ClientModel
 - createConnection, 12
 - getChatMessage, 11
 - getFullServerLobbyGamelist, 10
 - getGameLobbyPlayerlist, 10
 - getLanguage, 11
 - getPlayedCard, 11
 - getPlayerlistUpdate, 11
 - getServerLobbyGamelistUpdate, 10
 - hostGame, 12
 - joinGame, 12
 - kickPlayer, 11
 - makeMove, 12
 - setLanguage, 11
- Client::View::Card
 - Card, 13
- Client::View::ChooseCards
 - update, 13
- Client::View::ChooseItem
 - update, 14
- Client::View::CreateGame
 - CreateGame, 14
 - update, 14
- Client::View::Game
 - Game, 15
 - update, 15, 17
- Client::View::GameLobby
 - update, 17
- Client::View::HeartsCard
 - getCardID, 18
 - HeartsCard, 18
- Client::View::InputNumber
 - update, 19
- Client::View::Lobby
 - addChatMessageListener, 20
 - addHostButtonListener, 20
 - addJoinButtonListener, 19
 - addLeaveButtonListener, 20
 - setLanguage, 20
 - update, 20
- Client::View::Login
 - addConnectButtonListener, 21
 - addLanguageSelectionListener, 21
 - setLanguage, 21
 - update, 21
- Client::View::Password
 - update, 22
- Client::View::ScoreWindow
 - update, 22
- Client::View::Warning

- update, 23
- Client::View::WizCard
 - getCardID, 24
 - WizCard, 24
- ComBeenKicked
 - ComObjects::ComBeenKicked, 24
- ComChatMessage
 - ComObjects::ComChatMessage, 25
- ComCreateGameRequest
 - ComObjects::ComCreateGameRequest, 26
- ComInitGameLobby
 - ComObjects::ComInitGameLobby, 27
- ComInitLobby
 - ComObjects::ComInitLobby, 28
- ComJoinRequest
 - ComObjects::ComJoinRequest, 28
- ComKickPlayerRequest
 - ComObjects::ComKickPlayerRequest, 29
- ComLobbyUpdateGamelist
 - ComObjects::ComLobbyUpdateGamelist, 30
- ComLoginRequest
 - ComObjects::ComLoginRequest, 31
- ComObjects.ComBeenKicked, 24
- ComObjects.ComChatMessage, 24
- ComObjects.ComClientLeave, 25
- ComObjects.ComClientQuit, 25
- ComObjects.ComCreateGameRequest, 25
- ComObjects.ComInitGameLobby, 27
- ComObjects.ComInitLobby, 27
- ComObjects.ComJoinRequest, 28
- ComObjects.ComKickPlayerRequest, 29
- ComObjects.ComLobbyUpdateGamelist, 29
- ComObjects.ComLoginRequest, 30
- ComObjects.ComObject, 32
- ComObjects.ComRuleset, 32
- ComObjects.ComServerAcknowledgement, 33
- ComObjects.ComStartGame, 33
- ComObjects.ComUpdatePlayerlist, 33
- ComObjects.ComWarning, 34
- ComObjects.MsgCard, 34
- ComObjects.MsgCardRequest, 35
- ComObjects.MsgGameEnd, 35
- ComObjects.MsgMultiCards, 35
- ComObjects.MsgMultiCardsRequest, 37
- ComObjects.MsgMultipleCardsRequest, 37
- ComObjects.MsgNumber, 37
- ComObjects.MsgNumberRequest, 39
- ComObjects.MsgSelection, 39
- ComObjects.MsgSelectionRequest, 40
- ComObjects.MsgUser, 40
- ComObjects.RulesetMessage, 40
- ComObjects::ComBeenKicked
 - ComBeenKicked, 24
 - getMessage, 24
- ComObjects::ComChatMessage
 - ComChatMessage, 25
 - getChatMessage, 25
- ComObjects::ComCreateGameRequest
 - ComCreateGameRequest, 26
 - getGameName, 26
 - getPassword, 26
 - getRuleset, 26
 - hasPassword, 26
- ComObjects::ComInitGameLobby
 - ComInitGameLobby, 27
 - getPlayerList, 27
- ComObjects::ComInitLobby
 - ComInitLobby, 28
 - getGameList, 28
 - getPlayerList, 28
- ComObjects::ComJoinRequest
 - ComJoinRequest, 28
 - getGameMasterName, 29
- ComObjects::ComKickPlayerRequest
 - ComKickPlayerRequest, 29
 - getPlayerName, 29
- ComObjects::ComLobbyUpdateGamelist
 - ComLobbyUpdateGamelist, 30
 - getGameServer, 30
 - isRemoveFlag, 30
- ComObjects::ComLoginRequest
 - ComLoginRequest, 31
 - getPlayerName, 32
- ComObjects::ComRuleset
 - ComRuleset, 32
 - getRulesetMessage, 32
- ComObjects::ComUpdatePlayerlist
 - ComUpdatePlayerlist, 33
 - getPlayerName, 33
 - isRemoveFlag, 33
- ComObjects::ComWarning
 - ComWarning, 34
 - getWarning, 34
- ComObjects::MsgCard
 - getCard, 35
 - MsgCard, 35
- ComObjects::MsgMultiCards
 - getCardList, 37
 - MsgMultiCards, 36
- ComObjects::MsgMultiCardsRequest
 - getCount, 37
- ComObjects::MsgNumber
 - getNumber, 39
 - MsgNumber, 38
- ComObjects::MsgSelection
 - getSelection, 39
 - MsgSelection, 39
- ComObjects::MsgUser
 - getGameClientUpdate, 40
 - MsgUser, 40
- ComRuleset
 - ComObjects::ComRuleset, 32
- ComUpdatePlayerlist
 - ComObjects::ComUpdatePlayerlist, 33
- ComWarning
 - ComObjects::ComWarning, 34

- createConnection
 - Client::ClientModel, 12
- CreateGame
 - Client::View::CreateGame, 14
- dealCards
 - Ruleset::GameState, 47
 - Ruleset::ServerRuleset, 54
- Game
 - Client::View::Game, 15
- GameServer
 - Server::GameServer, 58
- GameServerRepresentation
 - Server::GameServerRepresentation, 60
- GameState
 - Ruleset::GameState, 45
- getAchievedTricks
 - Ruleset::WizData, 57
- getAnnouncedTricks
 - Ruleset::WizData, 57
- getCard
 - ComObjects::MsgCard, 35
- getCardID
 - Client::View::HeartsCard, 18
 - Client::View::WizCard, 24
- getCardList
 - ComObjects::MsgMultiCards, 37
- getCardsLeftInDeck
 - Ruleset::GameState, 46
- getChatMessage
 - Client::ClientModel, 11
 - ComObjects::ComChatMessage, 25
- getColour
 - Ruleset::Card, 41
 - Ruleset::HeartsCard, 48
 - Ruleset::WizardCard, 56
- getCompletePoints
 - Ruleset::HeartsData, 48
- getCount
 - ComObjects::MsgMultiCardsRequest, 37
- getCurrentPlayer
 - Ruleset::ClientRuleset, 42
 - Ruleset::GameClientUpdate, 44
 - Ruleset::GameState, 46
 - Ruleset::ServerRuleset, 53
- getCurrentPoints
 - Ruleset::HeartsData, 49
- getFirstPlayer
 - Ruleset::GameState, 46
 - Ruleset::ServerRuleset, 53
- getFullServerLobbyGamelist
 - Client::ClientModel, 10
- getGameClientUpdate
 - ComObjects::MsgUser, 40
- getGameList
 - ComObjects::ComInitLobby, 28
- getGameLobbyPlayerlist
 - Client::ClientModel, 10
- getGameMasterName
 - ComObjects::ComJoinRequest, 29
- getGameName
 - ComObjects::ComCreateGameRequest, 26
- getGameServer
 - ComObjects::ComLobbyUpdateGamelist, 30
- getHand
 - Ruleset::PlayerState, 50
- getLanguage
 - Client::ClientModel, 11
- getMaxPlayers
 - Ruleset::ServerRuleset, 52
- getMessage
 - ComObjects::ComBeenKicked, 24
- getMinPlayers
 - Ruleset::ServerRuleset, 52
- getName
 - Ruleset::PlayerState, 50
 - Server::Player, 64
- getNumber
 - ComObjects::MsgNumber, 39
- getNumberOfPlayedCards
 - Ruleset::GameState, 47
- getOtherData
 - Ruleset::PlayerState, 50
- getOtherPlayerData
 - Ruleset::ClientRuleset, 42
 - Ruleset::GameClientUpdate, 44
- getOwnData
 - Ruleset::GameClientUpdate, 44
- getOwnHand
 - Ruleset::GameClientUpdate, 44
- getPassword
 - ComObjects::ComCreateGameRequest, 26
- getPlayedCard
 - Client::ClientModel, 11
- getPlayedCards
 - Ruleset::GameClientUpdate, 44
 - Ruleset::GameState, 46
- getPlayerCards
 - Ruleset::GameState, 47
 - Ruleset::ServerRuleset, 54
- getPlayerList
 - ComObjects::ComInitGameLobby, 27
 - ComObjects::ComInitLobby, 28
- getPlayerName
 - ComObjects::ComKickPlayerRequest, 29
 - ComObjects::ComLoginRequest, 32
 - ComObjects::ComUpdatePlayerlist, 33
- getPlayerState
 - Ruleset::GameState, 46
 - Ruleset::ServerRuleset, 53
- getPlayerlistUpdate
 - Client::ClientModel, 11
- getPoints
 - Ruleset::WizData, 57
- getRuleset
 - ComObjects::ComCreateGameRequest, 26

- getRulesetMessage
 - ComObjects::ComRuleset, 32
- getRulesetType
 - Ruleset::ServerRuleset, 52
- getSelection
 - ComObjects::MsgSelection, 39
- getServerLobbyGamelistUpdate
 - Client::ClientModel, 10
- getTrumpCard
 - Ruleset::GameState, 47
- getValue
 - Ruleset::Card, 41
 - Ruleset::HeartsCard, 48
 - Ruleset::WizardCard, 56
- getWarning
 - ComObjects::ComWarning, 34
- getname
 - Ruleset::OtherData, 49
- giveACard
 - Ruleset::GameState, 47
 - Ruleset::ServerRuleset, 54
- hasPassword
 - ComObjects::ComCreateGameRequest, 26
- HeartsCard
 - Client::View::HeartsCard, 18
- hostGame
 - Client::ClientModel, 12
- initLobby
 - Server::GameServer, 60
 - Server::LobbyServer, 62
- isRemoveFlag
 - ComObjects::ComLobbyUpdateGamelist, 30
 - ComObjects::ComUpdatePlayerlist, 33
- isValidMove
 - Ruleset::ClientHearts, 41
 - Ruleset::ClientWizard, 43
 - Ruleset::ServerHearts, 51
 - Ruleset::ServerRuleset, 55
 - Ruleset::ServerWizard, 56
- joinGame
 - Client::ClientModel, 12
- kickPlayer
 - Client::ClientModel, 11
- main
 - Client::ClientMain, 9
 - Server::ServerMain, 66
- makeMove
 - Client::ClientModel, 12
- MsgCard
 - ComObjects::MsgCard, 35
- MsgMultiCards
 - ComObjects::MsgMultiCards, 36
- MsgNumber
 - ComObjects::MsgNumber, 38
- MsgSelection
 - ComObjects::MsgSelection, 39
- MsgUser
 - ComObjects::MsgUser, 40
- OtherData
 - Ruleset::OtherData, 49
- playCard
 - Ruleset::GameState, 47
- Player
 - Server::Player, 63
- PlayerState
 - Ruleset::PlayerState, 50
- receiveMessage
 - Server::GameServer, 59, 60
 - Server::LobbyServer, 61, 62
 - Server::Server, 64
- removeCard
 - Ruleset::PlayerState, 50
- removePlayer
 - Server::GameServer, 59
 - Server::Server, 65
- resolveMessage
 - Ruleset::ClientRuleset, 42, 43
 - Ruleset::ServerHearts, 51
 - Ruleset::ServerRuleset, 54
 - Ruleset::ServerWizard, 55
- Ruleset.Card, 41
- Ruleset.CardDeck, 41
- Ruleset.CardDeckBuilder, 41
- Ruleset.ClientHearts, 41
- Ruleset.ClientRuleset, 42
- Ruleset.ClientWizard, 43
- Ruleset.Colour, 44
- Ruleset.GameClientUpdate, 44
- Ruleset.GamePhase, 45
- Ruleset.GameState, 45
- Ruleset.HearthsDeck, 48
- Ruleset.HeartsCard, 48
- Ruleset.HeartsData, 48
- Ruleset.isValidMoveWizard, 49
- Ruleset.OtherData, 49
- Ruleset.PlayerState, 49
- Ruleset.RulesetType, 51
- Ruleset.ServerHearts, 51
- Ruleset.ServerRuleset, 51
- Ruleset.ServerWizard, 55
- Ruleset.WizData, 56
- Ruleset.WizardCard, 56
- Ruleset.WizardDeck, 56
- Ruleset::Card
 - getColour, 41
 - getValue, 41
- Ruleset::ClientHearts
 - isValidMove, 41
- Ruleset::ClientRuleset
 - getCurrentPlayer, 42

- getOtherPlayerData, 42
- resolveMessage, 42, 43
- send, 43
- Ruleset::ClientWizard
 - isValidMove, 43
- Ruleset::GameClientUpdate
 - getCurrentPlayer, 44
 - getOtherPlayerData, 44
 - getOwnData, 44
 - getOwnHand, 44
 - getPlayedCards, 44
- Ruleset::GameState
 - addPlayerToGame, 45
 - dealCards, 47
 - GameState, 45
 - getCardsLeftInDeck, 46
 - getCurrentPlayer, 46
 - getFirstPlayer, 46
 - getNumberOfPlayedCards, 47
 - getPlayedCards, 46
 - getPlayerCards, 47
 - getPlayerState, 46
 - getTrumpCard, 47
 - giveACard, 47
 - playCard, 47
 - setCurrentPlayer, 46
 - setFirstPlayer, 45
 - setTrumpCard, 46
- Ruleset::HeartsCard
 - getColour, 48
 - getValue, 48
- Ruleset::HeartsData
 - getCompletePoints, 48
 - getCurrentPoints, 49
- Ruleset::OtherData
 - getname, 49
 - OtherData, 49
- Ruleset::PlayerState
 - addCard, 50
 - getHand, 50
 - getName, 50
 - getOtherData, 50
 - PlayerState, 50
 - removeCard, 50
- Ruleset::ServerHearts
 - isValidMove, 51
 - resolveMessage, 51
- Ruleset::ServerRuleset
 - addPlayerToGame, 53
 - broadcast, 54
 - dealCards, 54
 - getCurrentPlayer, 53
 - getFirstPlayer, 53
 - getMaxPlayers, 52
 - getMinPlayers, 52
 - getPlayerCards, 54
 - getPlayerState, 53
 - getRulesetType, 52
 - giveACard, 54
 - isValidMove, 55
 - resolveMessage, 54
 - send, 54
 - ServerRuleset, 52
 - setCurrentPlayer, 53
 - setFirstPlayer, 53
- Ruleset::ServerWizard
 - isValidMove, 56
 - resolveMessage, 55
- Ruleset::WizData
 - announceTricks, 57
 - getAchievedTricks, 57
 - getAnnouncedTricks, 57
 - getPoints, 57
 - setPoints, 57
- Ruleset::WizardCard
 - getColour, 56
 - getValue, 56
- send
 - Ruleset::ClientRuleset, 43
 - Ruleset::ServerRuleset, 54
 - Server::Player, 63
- sendToPlayer
 - Server::Server, 65
- Server.ClientListenerThread, 57
- Server.GameServer, 57
- Server.GameServerRepresentation, 60
- Server.LobbyServer, 61
- Server.LobbyServer.ClientListenerThread, 62
- Server.Player, 63
- Server.Server, 64
- Server.ServerMain, 65
- Server::GameServer
 - addPlayer, 58
 - GameServer, 58
 - initLobby, 60
 - receiveMessage, 59, 60
 - removePlayer, 59
- Server::GameServerRepresentation
 - GameServerRepresentation, 60
- Server::LobbyServer
 - initLobby, 62
 - receiveMessage, 61, 62
- Server::Player
 - changeServer, 64
 - getName, 64
 - Player, 63
 - send, 63
 - setName, 64
- Server::Server
 - addPlayer, 65
 - broadcast, 65
 - receiveMessage, 64
 - removePlayer, 65
 - sendToPlayer, 65
- Server::ServerMain
 - main, 66

- ServerRuleset
 - Ruleset::ServerRuleset, [52](#)
- setCurrentPlayer
 - Ruleset::GameState, [46](#)
 - Ruleset::ServerRuleset, [53](#)
- setFirstPlayer
 - Ruleset::GameState, [45](#)
 - Ruleset::ServerRuleset, [53](#)
- setLanguage
 - Client::ClientModel, [11](#)
 - Client::View::Lobby, [20](#)
 - Client::View::Login, [21](#)
- setName
 - Server::Player, [64](#)
- setPoints
 - Ruleset::WizData, [57](#)
- setTrumpCard
 - Ruleset::GameState, [46](#)
- update
 - Client::View::ChooseCards, [13](#)
 - Client::View::ChooseItem, [14](#)
 - Client::View::CreateGame, [14](#)
 - Client::View::Game, [15](#), [17](#)
 - Client::View::GameLobby, [17](#)
 - Client::View::InputNumber, [19](#)
 - Client::View::Lobby, [20](#)
 - Client::View::Login, [21](#)
 - Client::View::Password, [22](#)
 - Client::View::ScoreWindow, [22](#)
 - Client::View::Warning, [23](#)
- WizCard
 - Client::View::WizCard, [24](#)