Trend Hearts SDK

Table of Contents

Releases

Date	Version	Notes	Package
2018-8-20	v1.0.0-alpha.4	ZIP file contains: • Game node (program) to host a table for 4 players. A player can be a dummy bot, a human or your AI agent. • Sample program to demo how to connect your AI agent to the game node and play a complete game.	Download SDK

Overview

Goal

The client and the server use websocket for communication. This page describes the protocol of hearts between the client and the server.

Terminology

Term	Meaning
Cards	• A card is composed of 2 characters. • The first character indicates the rank: 2~9, T(10), J(J), Q(Q), K(K), A(Ace). • The second character indicates the suit: H(eart), S(pade), C(lub), D(iamond). • Example: 3H -> 3♥ TS -> 10♠ QC -> Q♣ AD -> A♦
Match	• One match has the same four players play 4 games at rotating positions. The clients do not need to handle the match.
Game	• One game has 4 deals. In each deal, the target for passing cards is different.
Deal	• One deal has 3 phases, and the playing card phase of the deal has 13 rounds.
Round	• One round has each of the 4 players pick a card to play in clockwise order. The round ends with the player who played the highest-value card of the suit that was led, winning the round and any penalty points. * Note: Our SDK uses the term "rounds" to refer to the term "tricks" you may commonly read across materials related to "Hearts" and other trick-tacking playing card games. We apologize for any confusion this may cause.

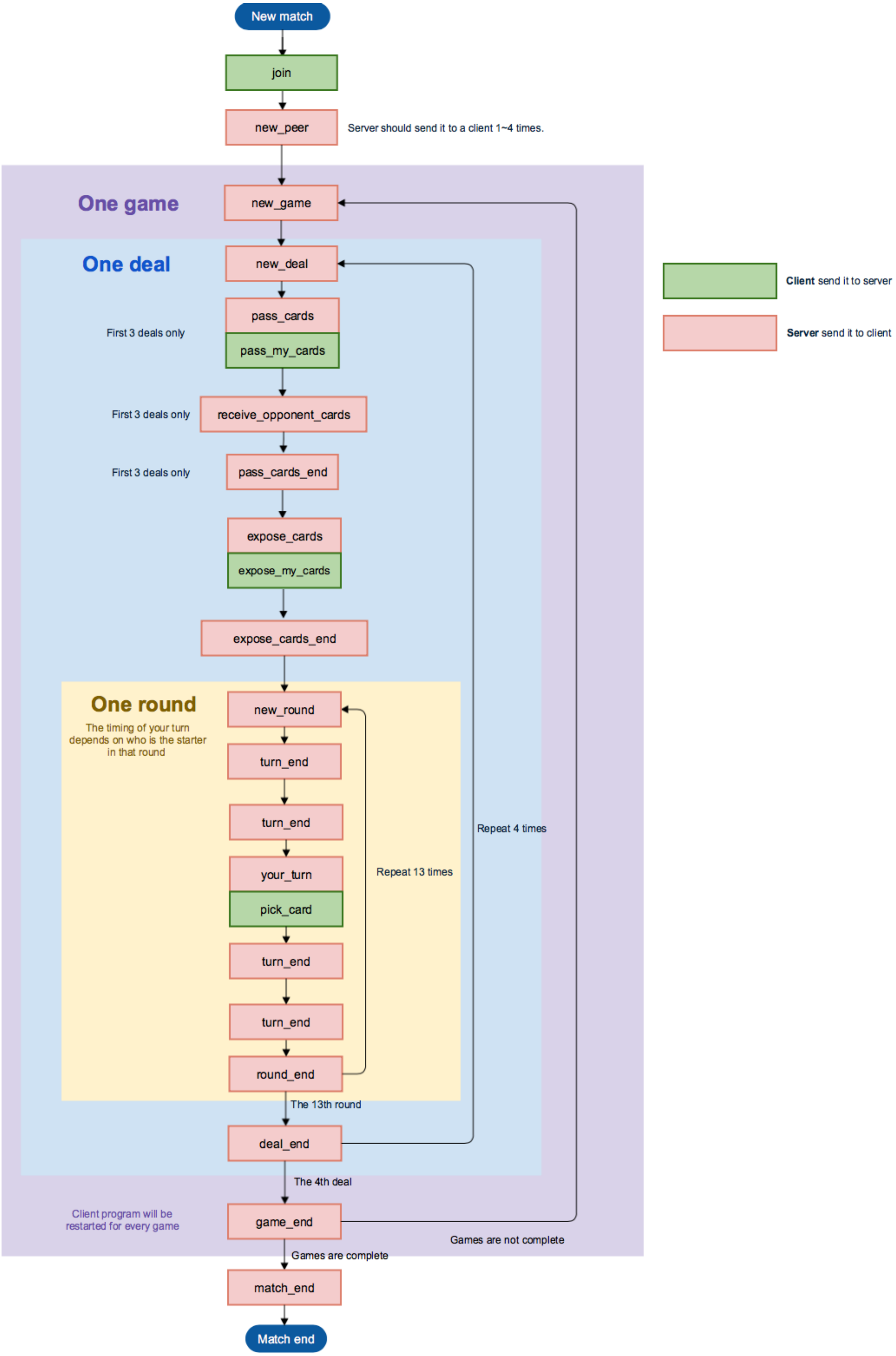
Event Introduction

Event Types

- Server
  - server_broadcast
    - Sent to all clients by the server, these types of events do not contain the private information of any clients.
  - server_request
    - Sent to a specific client by the server, these types of events contain the private information of that client. The server expects the client to reply with a proper event within a set timeout period.
  - server_inform
    - Sent to a specific client by the server, these types of events contain the private information of that client. The server does not expect the client to reply with anything.
- Client
  - client_event
    - Sent to the server by a client, these types of events must use correct values. Wrong values will cause the server to drop the event, and the server will perform a predefined action for the client instead.

Event Flow Chart

The server and client should send and receive events like shown in the figure below.


New match
join
new_peer
Server should send it to a client 1~4 times.
One game
new_game
One deal
new_deal
pass_cards
pass_my_cards
First 3 deals only
receive_opponent_cards
First 3 deals only
pass_cards_end
First 3 deals only
expose_cards
expose_my_cards
expose_cards_end
One round
The timing of your turn depends on who is the starter in that round
new_round
turn_end
turn_end
your_turn
pick_card
turn_end
turn_end
round_end
Repeat 13 times
Repeat 4 times
The 13th round
deal_end
The 4th deal
Client program will be restarted for every game
game_end
Games are not complete
Games are complete
match_end
Match end
Client send it to server
Server send it to client

Event List

Event Name	Sender	Description	Event Type
join	Client	The client could send this event to the server to join a game.	client_event
new_peer	Server	The server will send this event to all clients if there is a new client joining this game.	server_broadcast
new_game	Server	The server will send this event to all clients at the start of a game.	server_broadcast
new_deal	Server	The server will send this event to all clients at the start of a deal.	server_broadcast
pass_cards	Server	The server will send this event to the client to notify that they must pass cards. This event only happens in the first 3 deals. The client must reply with the ”pass_my_cards” event.	server_request
pass_my_cards	Client	The client must use this event to reply to the server's ”pass_cards” event.	client_event
receive_opponent_cards	Server	The server will send this event to the client to inform them what cards they received. This event only happens in the first 3 deals.	server_inform
pass_cards_end	Server	The server will send this event to all clients after it has received the ”pass_my_cards” event from all 4 clients.	server_broadcast
expose_cards	Server	The server will send this event to a specific client to ask if they want to expose their cards. This event is only sent to the client who has the ”AH” card. The client must reply with the ”expose_my_cards” event.	server_broadcast
expose_my_cards	Client	The client must use this event to reply to the server's ”expose_cards” event.	client_event
expose_cards_end	Server	The server will send this event to all clients after receiving the ”expose_my_cards” event from the client who has the ”AH” card.	server_broadcast
new_round	Server	The server will send this event to all clients at the start of a round.	server_broadcast
turn_end	Server	The server will send this event to all clients after receiving a ”pick_card” event from a client.	server_broadcast
your_turn	Server	The server will send this event to your client when it's your turn. The client must reply with the ”pick_card” event.	server_request
pick_card	Client	The client must use this event to reply to the server's ”your_turn” event.	client_event
round_end	Server	The server will send this event to all clients at the end of a round.	server_broadcast
deal_end	Server	The server will send this event to all clients at the end of a deal.	server_broadcast
game_end	Server	The server will send this event to all clients at the end of a game.	server_broadcast

Event Details

Client Events

To play hearts with the server, a client has to send the events below at the right time.

join

The client must send this event to the server to join a game.

Property name	Description	Type	Value
eventName	The name of this event.	String	”join”
playerNumber	The number of this player.	int	1
playerName	The name of this player.	String	As description
token	The token of this player, for authentication use.	String	As description

Example

```
{
  "eventName": "join",
  "data": {
   "playerNumber":1,
   "playerName":"player1",
    "token":"12345678"
  }
}
```

pass_my_cards

The client must use this event to reply to the server’s ”pass_cards” event.

Property name	Description	Type	Value
eventName	The name of this event.	string	”pass_my_cards”
cards	The cards you want to pass. You must pick these cards from the ”candidateCards” of the ”pass_cards” event.	A list containing 3 strings	As description
dealNumber	The number of this deal (starts from 1). The client must use the ”deal_number” of the ”pass_cards” event. A wrong value will cause the server to drop this reply.	int	As description

Example

```
{
  "eventName": "pass_my_cards",
  "data": {
   "dealNumber":1,
   "cards": ["3H","4H", "5H"]
  }
}
```

expose_my_cards

The client must use this event to reply to the server’s ”expose_cards” event.

Property name	Description	Type	Value
eventName	The name of this event.	string	”expose_my_cards”
cards	The cards you want to expose. You must pick these cards from the ”candidateCards” of the ”expose_cards” event. Reply with an empty list if you don’t want to expose any cards.	A list containing 0-1 strings	As description
dealNumber	The number of this deal (starts from 1). The client must use the ”deal_number” of the ”expose_cards” event. A wrong value will cause the server to drop this reply.	int	As description

Example

```
{
  "eventName": "expose_my_cards",
  "data": {
   "dealNumber":1,
   "cards": ["AH"]
  }
}
```

pick_card

The client must use this event to reply to the server's ”your_turn” event.

Property name	Description	Type	Value
eventName	The name of this event.	string	”pick_card”
turnCard	The card you picked. You must pick a card from the ”candidateCards” of the ”your_turn” event.	string	As descriptior
dealNumber	The number of this deal (starts from 1). The client must use the ”deal_number” of the ”your_turn” event. A wrong value will cause the server to drop this reply.	int	As descriptior
roundNumber	The number of this round (starts from 1). The client must use the ”round_number” of the ”your_turn” event. A wrong value will cause the server to drop this reply.	int	As descriptior

Example

```
{
  "eventName": "pick_card",
  "data": {
   "dealNumber":1,
   "roundNumber":1,
   "turnCard": "3H"
  }
}
```

Server Events

The server will send these events to the client. The client is only required to reply to the ”pass_cards”, ”expose_cards” and ”your_turn” events.

Some events contain the detailed information of all players, and some events contain only a subset of the information. If denoted as ”private information”, the client only receives their own information. Players are listed from north to east to south to west.

The following is a reference:

Property name	Description	Type
playerNumber	The number of this player. This is unique to all players.	int
playerName	The name of this player.	string

gameScore	The score of this player in this game. A sum of 4 deals. It will be updated when one deal ends.	int
dealScore	The score of this player in this deal. One deal score is the sum of score cards.	int
cards	The cards of this player. This is private information.	a list of strings
cardsCount	The count of this player's cards.	int
scoreCards	The score cards of this player in this deal.	a list of strings
pickedCards	The cards this player picked. This is private information.	a list of strings
receivedCards	The cards this player received. This is private information.	a list of strings
receivedFrom	The player who passed cards to this player.	a list of strings
exposedCards	The exposed cards of this player.	a list of strings
initialCards	The initial cards of player (before pass cards).	a list of string
roundCard	The card this player picked in this round. This key is not included if the player has not picked a card yet. This value will be cleared at the start of a round.	string
serverRandom	The random card picked by the server to play because the client timed out or replied with wrong values.	boolean
timeoutCount	The count of replies that timed out.	int
errorCount	The count of replies with wrong values.	int
status	The status of this player. 0 - online, 1 - offline , 2 - timed out	int

new_peer

The server will send this event to all clients if there is a new client joining this game.

Property name	Description	Type	Value
eventName	The name of this event.	string	"new_peer"
players	The information of all players (including the new player, north-east-south-west).	A list containing 1~4 json	As description

Example

```
{
  "eventName": "new_peer",
  "data": {
   "players": [
        {
            "playerNumber": 1,
            "playerName": "player1",
            "status": 0
        },
        {
            "playerNumber": 2,
            "playerName": "player2",
            "status": 0
        },
        {
            "playerNumber": 3,
            "playerName": "player3",
            "status": 0
        },
        {
            "playerNumber": 4,
            "playerName": "player4",
            "status": 0
        }
    ]
  }
}
```

new_game

The server will send this event to all clients at the start of a game.

Property name	Description	Type	Value
eventName	The name of this event.	string	”new_game”
players	The information of all players (north-east-south-west).	A list containing 4 strings	As description

Example

```
{
  "eventName": "new_game",
  "data": {
   "players": [
        {
            "playerNumber": 1,
            "playerName": "player1",
            "status": 0
        },
        {
            "playerNumber": 2,
            "playerName": "player2",
            "status": 0
        },
        {
            "playerNumber": 3,
            "playerName": "player3",
            "status": 0
        },
        {
            "playerNumber": 4,
            "playerName": "player4",
            "status": 0
        }
    ]
  }
}
```

new_deal

The server will send this event to all client at the start of a deal.

Property name	Description	Type	Value
eventName	The name of this event.	string	"new_deal"
dealNumber	The number of this deal (starts from 1).	int	As description
self	Your player information.	json	As description
players	The information of all players (north-east-south-west).	A list containing 4 json	As description

Example

```
{
  "eventName": "new_deal",
  "data": {
   "dealNumber" 2,
   "self": {
        "playerNumber": 1,
        "playerName" : "player1",
        "gameScore" : 1,
        "dealScore" : 0,
        "cards":
["2H","3H","4H","5H","6H","7H","8H","9H","TH","JH","QH","KH","AH"],
        "cardsCount": 13
   },
   "players": [
  {
   "playerNumber": 1,
            "playerName" : "player1",
         "gameScore" : 5,
            "dealScore" : 0,
            "cards":
["2H","3H","4H","5H","6H","7H","8H","9H","TH","JH","QH","KH","AH"],
         "cardsCount": 13
        },
  {
   "playerNumber": 2,
            "playerName" : "player2",
            "gameScore" : 4,
            "dealScore" : 0,
            "cards":
["2S","3S","4S","5S","6S","7S","8S","9S","TS","JS","QS","KS","AS"],
         "cardsCount": 13
        },
  {
   "playerNumber": 3,
            "playerName" : "player3",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards":
["2C","3C","4C","5C","6C","7C","8C","9C","TC","JC","QC","KC","AC"],
         "cardsCount": 13
        },
  {
   "playerNumber": 4,
            "playerName" : "player4",
            "gameScore" : 10,
            "dealScore" : 0,
            "cards":
["2D","3D","4D","5D","6D","7D","8D","9D","TD","JD","QD","KD","AD"],
         "cardsCount": 13
        }
```

```
 ]
  }
}
```

pass_cards

The server will send this event to the client to notify that they must pass cards. This event only happens in the first 3 deals. The client must reply with the ”pass_my_cards” event.

Property name	Description	Type	Value
eventName	The name of this event.	string	”pass_cards”
dealNumber	The number of this deal (starts from 1).	int	As description
self	Your player information.	json	As description
players	The information of all players (north-east-south-west).	A list containing 4 json	As description
receiver	The player who will receive the cards you pass.	string	As description

Example

```
{
  "eventName": "pass_cards",
  "data": {
   "dealNumber" 2,
   "self": {
  "playerNumber": 1,
        "playerName" : "player1",
        "gameScore" : 10,
        "dealScore" : 0,
        "cards":
["2H","3H","4H","5H","6H","7H","8H","9H","TH","JH","QH","KH","AH"],
        "cardsCount": 13
   },
   "players": [
  {
   "playerNumber": 1,
            "playerName" : "player1",
            "gameScore" : 10,
            "dealScore" : 0,
            "cards":
["2H","3H","4H","5H","6H","7H","8H","9H","TH","JH","QH","KH","AH"],
         "cardsCount": 13
        },
  {
   "playerNumber": 2,
            "playerName" : "player2",
            "gameScore" : 3,
            "dealScore" : 0,
            "cards":
["2S","3S","4S","5S","6S","7S","8S","9S","TS","JS","QS","KS","AS"],
         "cardsCount": 13
        },
  {
   "playerNumber": 3,
            "playerName" : "player3",
            "gameScore" : 6,
            "dealScore" : 0,
            "cards":
["2C","3C","4C","5C","6C","7C","8C","9C","TC","JC","QC","KC","AC"],
         "cardsCount": 13
        },
  {
   "playerNumber": 4,
            "playerName" : "player4",
            "gameScore" : 8,
            "dealScore" : 0,
            "cards":
["2D","3D","4D","5D","6D","7D","8D","9D","TD","JD","QD","KD","AD"],
         "cardsCount": 13
        }
 ],
   "receiver":"player2"
  }
}
```

receive_opponent_cards

The server will send this event to the client to inform them what cards they received. This event only happens in the first 3 deals.

Property name	Description	Type	Value
eventName	The name of this event.	string	”receive_opponent_cards”
dealNumber	The number of this deal (starts from 1).	int	As description
self	Your player information.	json	As description
players	The information of all players (north-east-south-west).	A list containing 4 json	As description
receivedCards	The cards you received from the opponent.	A list of 3 strings	As description
sender	The player who passed the cards to you.	string	As description

Example

```
{
  "eventName": "receive_opponent_cards",
  "data": {
   "dealNumber" 1,
   "self": {
  "playerNumber": 1,
        "playerName" : "player1",
        "gameScore" : 10,
        "dealScore" : 0,
        "cards":
["2H","3H","4H","5H","6H","7H","8H","9H","TH","JH","QH","KH","AH"],
        "cardsCount": 13,
        "receivedCards": ["3S","4S", "5S"],
        "receivedFrom": "player2"
   },
   "players": [
  {
   "playerNumber": 1,
            "playerName" : "player1",
            "gameScore" : 10,
            "dealScore" : 0,
            "cards":
["2H","3H","4H","5H","6H","7H","8H","9H","TH","JH","QH","KH","AH"],
         "cardsCount": 13,
            "receivedCards": ["3S","4S", "5S"],
            "receivedFrom": "player2
        },
  {
   "playerNumber": 2,
            "playerName" : "player2",
            "gameScore" : 3,
            "dealScore" : 0,
            "cards":
["2S","3S","4S","5S","6S","7S","8S","9S","TS","JS","QS","KS","AS"],
         "cardsCount": 13
        },
  {
   "playerNumber": 3,
            "playerName" : "player3",
            "gameScore" : 6,
            "dealScore" : 0,
            "cards":
["2C","3C","4C","5C","6C","7C","8C","9C","TC","JC","QC","KC","AC"],
         "cardsCount": 13
        },
  {
   "playerNumber": 4,
            "playerName" : "player4",
            "gameScore" : 8,
            "dealScore" : 0,
            "cards":
["2D","3D","4D","5D","6D","7D","8D","9D","TD","JD","QD","KD","AD"],
         "cardsCount": 13
        }
```

```
 ]
  }
}
```

pass_cards_end

The server will send this event to all clients after it has received the ”pass_my_cards” event from all 4 clients.

Property name	Description	Type	Value
eventName	The name of this event.	string	”pass_cards_end”
dealNumber	The number of this deal (starts from 1).	int	As description
players	The information of all players (north-east-south-west).	A list containing 4 json	As description

Example

```
{
  "eventName": "pass_cards_end"
  "data": {
   "dealNumber":1,
   "players": [
  {
   "playerNumber": 1,
            "playerName" : "player1",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards":
["2H","3H","4H","5H","6H","7H","8H","9H","TH","JH","QH","KH","AH"],
         "cardsCount": 13,
   "pickedCards": ["3H","4H","5H"],
            "receivedCards": ["3S","4S","5S"],
     "receivedFrom": "player2"
        },
  {
   "playerNumber": 2,
            "playerName" : "player2",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards":
["2S","3S","4S","5S","6S","7S","8S","9S","TS","JS","QS","KS","AS"],
         "cardsCount": 13,
   "pickedCards": ["3H","4H","5H"],
            "receivedCards": ["3S","4S","5S"],
     "receivedFrom": "player2"
        },
  {
   "playerNumber": 3,
            "playerName" : "player3",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards":
["2C","3C","4C","5C","6C","7C","8C","9C","TC","JC","QC","KC","AC"],
         "cardsCount": 13,
   "pickedCards": ["3H","4H","5H"],
            "receivedCards": ["3S","4S","5S"],
     "receivedFrom": "player2"
        },
  {
   "playerNumber": 4,
            "playerName" : "player4",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards":
["2D","3D","4D","5D","6D","7D","8D","9D","TD","JD","QD","KD","AD"],
         "cardsCount": 13,
   "pickedCards": ["3H","4H","5H"],
            "receivedCards": ["3S","4S","5S"],
     "receivedFrom": "player2"
        }
```

```
 ]
  }
}
```

expose_cards

The server will send this event to a specific client to ask if they want to expose their cards. This event is only sent to the client who has the ”AH” card. The client must reply with the ”expose_my_cards” event.

Property name	Description	Type	Value
eventName	The name of this event.	string	”expose_cards”
dealNumber	The number of this deal (starts from 1).	int	As description
self	Your player information.	json	As description
players	The information of all players (north-east-south-west).	A list containing 4 json	As description

Example

```
{
  "eventName": "expose_cards",
  "data": {
   "dealNumber" 2,
   "self": {
  "playerNumber": 1,
        "playerName" : "player1",
        "gameScore" : 10,
        "dealScore" : 0,
        "cards":
["QS","TC","4H","5H","6H","7H","8H","9H","TH","JH","QH","KH","AH"],
        "candidateCards": ["AH"],
        "cardsCount": 13
   },
   "players": [
  {
   "playerNumber": 1,
            "playerName" : "player1",
            "gameScore" : 10,
            "dealScore" : 0,
         "cards":
["QS","TC","4H","5H","6H","7H","8H","9H","TH","JH","QH","KH","AH"],
         "cardsCount": 13
        },
  {
   "playerNumber": 2,
            "playerName" : "player2",
            "gameScore" : 3,
            "dealScore" : 0,
            "cards":
["2S","3S","4S","5S","6S","7S","8S","9S","TS","JS","QS","KS","AS"],
         "cardsCount": 13
        },
  {
   "playerNumber": 3,
            "playerName" : "player3",
            "gameScore" : 6,
            "dealScore" : 0,
            "cards":
["2C","3C","4C","5C","6C","7C","8C","9C","TC","JC","QC","KC","AC"],
         "cardsCount": 13
        },
  {
   "playerNumber": 4,
            "playerName" : "player4",
            "gameScore" : 8,
            "dealScore" : 0,
            "cards":
["2D","3D","4D","5D","6D","7D","8D","9D","TD","JD","QD","KD","AD"],
         "cardsCount": 13
        }
```

```
 ]
  }
}
```

expose_cards_end

The server will send this event to all clients after receiving the ”expose_my_cards” event from the client who has the ”AH” card.

Property name	Description	Type	Value
eventName	The name of this event.	string	”expose_cards_end”
dealNumber	The number of this deal (starts from 1).	int	As description
players	The information of all players (north-east-south-west).	A list containing 4 json	As description

Example

```
{
  "eventName": "expose_cards_end"
  "data": {
   "dealNumber":1,
   "players": [
  {
   "playerNumber": 1,
            "playerName" : "player1",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards":
["QS","TC","2H","4H","5H","6H","7H","8H","9H","TH","JH","QH","KH"],
         "cardsCount": 13,
   "pickedCards": ["3H","4H", "5H"],
            "receivedCards": ["3S","4S", "5S"],
     "receivedFrom": "player2",
   "exposedCards": ["AH"]
        },
  {
   "playerNumber": 2,
            "playerName" : "player2",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards":
["2S","3S","4S","5S","6S","7S","8S","9S","TS","JS","QS","KS","AH"],
         "cardsCount": 13,
   "pickedCards": ["3H","4H", "5H"],
            "receivedCards": ["3S","4S", "5S"],
     "receivedFrom": "player2"
        },
  {
   "playerNumber": 3,
            "playerName" : "player3",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards":
["2C","3C","4C","5C","6C","7C","8C","9C","TC","JC","QC","KC","AC"],
         "cardsCount": 13,
   "pickedCards": ["3H","4H", "5H"],
            "receivedCards": ["3S","4S", "5S"],
     "receivedFrom": "player2",
   "exposedCards": []
        },
  {
   "playerNumber": 4,
            "playerName" : "player4",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards":
["2D","3D","4D","5D","6D","7D","8D","9D","TD","JD","QD","KD","AD"],
         "cardsCount": 13,
   "pickedCards": ["3H","4H", "5H"],
            "receivedCards": ["3S","4S", "5S"],
     "receivedFrom": "player2",
   "exposedCards": []
        }
```

```
 ]
  }
}
```

new_round

The server will send this event to all clients at the start of a round.

Property name	Description	Type	Value
eventName	The name of this event.	string	”new_round”
dealNumber	The number of this deal (starts from 1).	int	As description
roundNumber	The number of this round (starts from 1).	int	As description
players	The information of all players (north-east-south-west).	A list containing 4 json	As description
roundPlayers	The order for picking cards in this round.	A list containing 4 string	As description

Example

```
{
  "eventName": "new_round"
  "data": {
   "dealNumber":1,
   "roundNumber":4,
   "players": [
  {
   "playerNumber": 1,
            "playerName" : "player1",
            "gameScore" : 0,
            "dealScore" : 10,
            "cards":
["QS","TC","6H","7H","8H","9H","TH","JH","KH","AH"],
         "cardsCount": 10,
   "scoreCards":["4H","5H"],
   "exposedCards": ["AH"]
        },
  {
   "playerNumber": 2,
            "playerName" : "player2",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards":
["2S","3S","4S","5S","6S","7S","8S","9S","TS","JS"],
         "cardsCount": 10,
   "scoreCards":[],
   "exposedCards": []
        },
  {
   "playerNumber": 3,
            "playerName" : "player3",
            "gameScore" : 0,
            "dealScore" : 13,
            "cards":
["2C","3C","4C","5C","6C","7C","8C","9C","TC","JC"],
         "cardsCount": 10,
   "scoreCards":['QH'],
   "exposedCards": []
        },
  {
   "playerNumber": 4,
            "playerName" : "player4",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards": ["2D","3D","4D","5D","6D","7D","8D","9D","TD"],
         "cardsCount": 10,
   "scoreCards":[],
   "exposedCards": []
        }
    ],
   "roundPlayers": ["player4","player1","player2","player3"]
  }
}
```

turn_end

The server will send this event to all clients after receiving a ”pick_card” event from a client.

Property name	Description	Type	Value
eventName	The name of this event.	string	”turn_end”
dealNumber	The number of this deal (starts from 1).	int	As description
roundNumber	The number of this round (starts from 1).	int	As description
players	The information of all players (north-east-south-west).	A list containing 4 json	As description
roundPlayers	The order for picking cards in this round.	A list containing 4 string	As description
turnPlayer	The player whose turn it was.	string	As description
turnCard	The card picked by the player in this turn.	string	As description
serverRandom	If the turn card is random picked by server, may caused by response timeout or pick invalid card.	boolean	As description

Example

```
{
  "eventName": "turn_end",
  "data": {
   "dealNumber":1,
   "roundNumber":4,
   "players": [
  {
   "playerNumber": 1,
            "playerName" : "player1",
            "gameScore" : 0,
            "dealScore" : 2,
            "cards": ["QS","TC","6H","7H","8H","9H","TH","JH","KH"],
         "cardsCount": 9,
   "scoreCards":["4H","5H"],
            "roundCard":"2C",
   "serverRandom": false,
   "exposedCards": ["AH"]
        },
  {
   "playerNumber": 2,
            "playerName" : "player2",
            "gameScore" : 0,
            "dealScore" : 13,
            "cards":
["2S","3S","4S","5S","6S","7S","8S","9S","TS","JS"],
         "cardsCount": 10,
   "scoreCards":["QH"],
   "exposedCards": []
        },
  {
   "playerNumber": 3,
            "playerName" : "player3",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards": ["4C","5C","6C","7C","8C","9C","TC","JC"],
         "cardsCount": 10,
   "scoreCards":[],
   "exposedCards": []
        },
  {
   "playerNumber": 4,
            "playerName" : "player4",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards":
["2D","3D","4D","5D","6D","7D","8D","9D","TD","JD"],
         "cardsCount": 10,
   "scoreCards":[],
            "roundCard":"3C",
   "exposedCards": []
        }
 ],
   "roundPlayers": ["player4","player1","player2","player3"],
   "turnPlayer":"player1",
   "turnCard":"2C",
 "serverRandom": false
  }
}
```

your_turn

The server will send this event to your client when it's your turn. The client must reply with the ”pick_card” event.

Property name	Description	Type	Value
eventName	The name of this event.	string	”your_turn”
dealNumber	The number of this deal (starts from 1).	int	As description
roundNumber	The number of this round (starts from 1).	int	As description
self	Your player information. candidateCards: The cards you can pick in this round. You must pick cards from this list.	json	As description
players	The information of all players (north-east-south-west).	A list containing 4 json	As description
roundPlayers	The order for picking cards in this round.	A list containing 4 string	As description

Example

```
  {
    "eventName": "your_turn",
    "data": {
     "dealNumber" 1,
     "roundNumber":5,
     "self": {
    "playerNumber": 1,
          "playerName" : "player3",
          "gameScore" : 0,
          "dealScore" : 0,
          "cards": ["QS","TC","4C","5C","6C","7C","8C","9C","TC"],
          "cardsCount": 9,
          "candidateCards": ["7C","8C","9C","TC"],
    "exposedCards": ["AH"]
     },
     "players": [
    {
     "playerNumber": 1,
              "playerName" : "player1",
              "gameScore" : 0,
              "dealScore" : 2,
              "cards": ["QS","TC","4H","5H","6H","7H","9H","TH"],
           "cardsCount": 8,
     "scoreCards":["4H","5H"],
              "roundCard":"2C",
     "serverRandom": false,
     "exposedCards": ["AH"]
          },
    {
     "playerNumber": 2,
              "playerName" : "player2",
              "gameScore" : 0,
              "dealScore" : 1,
              "cards": ["3S","4S","5S","6S","7S","8S","9S","10S"],
           "cardsCount": 8,
     "scoreCards":["8H"],
              "roundCard":"2S",
     "serverRandom": false,
     "exposedCards": [],
          },
    {
     "playerNumber": 3,
              "playerName" : "player3",
              "gameScore" : 0,
              "dealScore" : 0,
              "cards": ["2C","3C","4C","5C","6C","7C","8C","9C"],
           "cardsCount": 8,
     "scoreCards":[],
     "exposedCards": []
          },
    {
     "playerNumber": 4,
              "playerName" : "player4",
              "gameScore" : 0,
              "dealScore" : 0,
              "cards": ["2D","3D","4D","5D","6D","7D","8D","9D"],
           "cardsCount": 8,
     "scoreCards":[],
     "exposedCards": []
          }
   ],
     "roundPlayers": ["player4","player1","player2","player3"]
    }
  }
```

round_end

The server will send this event to all clients at the end of a round.

Property name	Description	Type	Value
eventName	The name of this event.	string	”round_end”
dealNumber	The number of this deal (starts from 1).	int	As description
roundNumber	The number of this round (starts from 1).	int	As description
players	The information of all players (north-east-south-west).	A list containing 4 json	As description
roundPlayers	The order for picking cards in this round.	A list containing 4 string	As description
roundPlayer	The player who took all cards in this round.	string	As description

Example

```
{
  "eventName": "round_end"
  "data": {
   "dealNumber":1,
   "roundNumber":5,
   "players": [
  {
   "playerNumber": 1,
            "playerName" : "player1",
            "gameScore" : 0,
            "dealScore" : 2,
            "cards": ["QS","TC","6H","7H","8H","9H","TH","JH"],
         "cardsCount": 8,
   "scoreCards":["4H","5H"],
            "roundCard":"2C",
   "serverRandom": false,
   "exposedCards": ["AH"]
        },
  {
   "playerNumber": 2,
            "playerName" : "player2",
            "gameScore" : 0,
            "dealScore" : 0,
            "cards": ["2S","3S","4S","5S","6S","7S","8S","9S"],
         "cardsCount": 8,
   "scoreCards":[],
            "roundCard":"TS",
   "serverRandom": true,
   "exposedCards": []
        },
  {
```

```
  "playerNumber": 3,
           "playerName" : "player3",
           "gameScore" : 0,
           "dealScore" : 0,
           "cards": ["2C","3C","4C","5C","6C","7C","8C","9C"],
        "cardsCount": 8,
  "scoreCards":[],
           "roundCard":"TC",
  "serverRandom": false,
  "exposedCards": []
       },
 {
  "playerNumber": 4,
           "playerName" : "player4",
           "gameScore" : 0,
           "dealScore" : 0,
           "cards": ["2D","3D","4D","5D","6D","7D","8D","9D"],
        "cardsCount": 8,
  "scoreCards":[],
           "roundCard":"TD",
  "serverRandom": false,
  "exposedCards": []
       }
]
  "roundPlayers": ["player4","Player1","player2","player3"],
   "roundPlayer":"player2"
  }
}
```

deal_end

The server will send this event to all clients at the end of a deal.

Property name	Description	Type	Value
eventName	The name of this event.	string	”deal_end”
dealNumber	The number of this deal (starts from 1).	int	As description
players	The information of all players (north-east-south-west).	A list containing 4 json	As description

Example

```
{
   "eventName":"deal_end",
   "data":{
      "dealNumber":3,
      "players":[
         {
            "playerNumber":1,
            "playerName":"player1",
            "gameScore":-86,
            "errorCount":0,
            "timeoutCount":0,
            "dealScore":-17,
            "scoreCards":[
      "QS",
      "JH",
               "8H",
               "6H",
               "5H"
            ],
            "pickedCards":[
               "7S",
               "3S",
               "AH"
            ],
            "receivedCards":[
               "KS",
               "JS",
               "9S"
            ],
            "receivedFrom":"player3",
            "exposedCards":[

            ],
            "shootingTheMoon":false,
         "initialCards":[
            "7S",
            "3S",
            "AH",
            "QH",
            "TH",
            "8H",
            "2H",
            "QC",
            "3C",
            "2C",
            "QD",
            "8D",
            "5D"
         ],
         "status":0
      },
      {
         "playerNumber":2,
         "playerName":"player2",
         "gameScore":-22,
         "errorCount":0,
         "timeoutCount":0,
         "dealScore":-4,
         "scoreCards":[
            "9H",
            "4H",
            "3H",
            "2H"
         ],
         "pickedCards":[
            "QS",
            "TS",
            "8S"
         ],
         "receivedCards":[
            "6H",
            "5S",
            "6C"
         ],
         "receivedFrom":"player4",
         "exposedCards":[

         ],
         "shootingTheMoon":false,
         "initialCards":[
            "QS",
            "TS",
            "8S",
            "4S",
            "2S",
            "KH",
            "9H",
            "JC",
            "7C",
            "AD",
            "KD",
            "JD",
            "7D"
         ],
         "status":0
      },
      {
         "playerNumber":3,
         "playerName":"player3",
         "gameScore":-10,
         "errorCount":0,
         "timeoutCount":0,
         "dealScore":-10,
         "scoreCards":[
            "AH",
            "KH",
            "QH",
            "TH",
            "7H",
            "TC"
         ],
         "pickedCards":[
            "KS",
            "JS",
            "9S"
         ],
         "receivedCards":[
            "7S",
            "3S",
            "AH"
         ],
         "receivedFrom":"player1",
         "exposedCards":[

         ],
         "shootingTheMoon":false,
         "initialCards":[
            "KS",
            "JS",
            "9S",
            "JH",
            "5H",
            "4H",
            "AC",
            "KC",
            "TC",
            "9C",
            "5C",
            "3D",
            "2D"
         ],
         "status":0
      },
      {
         "playerNumber":4,
         "playerName":"player4",
         "gameScore":-2,
         "errorCount":0,
         "timeoutCount":0,
         "dealScore":0,
         "scoreCards":[

         ],
         "pickedCards":[
            "6H",
            "5S",
            "6C"
         ],
         "receivedCards":[
            "QS",
            "TS",
            "8S"
         ],
         "receivedFrom":"player2",
         "exposedCards":[

         ],
         "shootingTheMoon":false,
         "initialCards":[
            "AS",
            "6S",
            "5S",
            "7H",
            "6H",
            "3H",
            "8C",
            "6C",
            "4C",
            "TD",
            "9D",
            "6D",
            "4D"
         ],
         "status":1
      }
   ]
}
```

```
}
```

game_end

The server will send this event to all clients at the end of a game.

Property name	Description	Type	Value
eventName	The name of this event.	string	"game_end"
players	The information of all players (north-east-south-west).	A list containing 4 json	As description

Example

```
{
  "eventName": "game_end",
  "data": {
   "players": [
  {
   "playerNumber": 1,
            "playerName": "player1",
            "gameScore" : -9,
            "timeoutCount": 0,
            "errorCount": 0,
            "rank": 1,
   "deals": [
                {
     "dealNumber":1,
                    "score": -1,
                    "exposedCards": ["AH"]
                },
                {
     "dealNumber":2,
                    "score": 0,
                    "exposedCards": [],
                    "shootingTheMoon": true
                },
                {
      "dealNumber":3,
                    "score": -5,
                    "exposedCards": []
                },
                {
        "dealNumber":4,
                    "score": -3,
                    "exposedCards": []
                }
            ]
        },
  {
   "playerNumber": 2,
            "playerName": "player2",
            "gameScore" : -50,
            "timeoutCount": 0,
            "errorCount": 0,
            "rank": 2,
   "deals": [
                {
     "dealNumber":1,
                    "score": -10,
                    "exposedCards": []
                },
                {
     "dealNumber":2,
                    "score": -20,
                    "exposedCards": ["AH"]
                },
                {
      "dealNumber":3,
                    "score": -5,
                    "exposedCards": []
                },
                {
        "dealNumber":4,
                    "score": -15,
                    "exposedCards": []
                }
            ]
        },
  {
   "playerNumber": 3,
            "playerName": "player3",
            "gameScore" : -50,
            "timeoutCount": 0,
            "errorCount": 0,
            "rank": 2,
   "deals": [
                {
     "dealNumber":1,
                    "score": -10,
                    "exposedCards": []
                },
                {
     "dealNumber":2,
                    "score": -20,
                    "exposedCards": []
                },
                {
      "dealNumber":3,
                    "score": 0,
                    "exposedCards": []
                },
                {
        "dealNumber":4,
                    "score": -20,
                    "exposedCards": []
                }
            ]
        },
  {
   "playerNumber": 4,
            "playerName": "player4",
            "gameScore" : -60,
            "timeoutCount": 0,
            "errorCount": 0,
            "rank": 4,
   "deals": [
                {
     "dealNumber":1,
                    "score": -10,
                    "exposedCards": []
                },
                {
     "dealNumber":2,
                    "score": 0,
                    "exposedCards": []
                },
                {
      "dealNumber":3,
                    "score": -50,
                    "exposedCards": []
                },
                {
        "dealNumber":4.
                    "score": 0,
                    "exposedCards": []
                }
            ]
        }
   ]
    }
  }
```

How to run the game node

1. Run the executable file (click or run in terminal, such as ./hearts_game_node-linux). If you see "Trend Hearts Poker Node is running, listening on port 8080" in the terminal, it means that the game node start successfully.
2. After starting the game node successfully, it will start a web service which is listen on a port (default 8080). Open http://localhost:8080/ in your web browser and you will see the index page.
3. One game node can only host one table (4 seats). On the page, you can select who will play on the table with 4 seats. You can select "Dummy player" (a simple bot), "Human player" (play yourself), "Empty seat" (play using your AI). If you are using your AI, you should use the specified seat number as your playerNumber (1-4), and avoid using "player1", "player2", "player3", and "player4" as your playerName as they are used by the Dummy). After selecting each seat, click "Start Game". Once all 4 players have joined, the game will start automatically.
4. Advanced configuration using command line arguments :

```
--version: Show version number [boolean]
--port: Port to bind on [number] [default: 8080]
--pass-cards-timeout: Timeout of pass_cards event in millisecond [number] [default: 3000]
--expose-cards-timeout: Timeout of expose_cards event in millisecond [number] [default: 3000]
--pick-card-timeout: Timeout of pick_card event in millisecond [number] [default: 1000]
--round-interval: Time interval between rounds in millisecond [number] [default: 2000]
--deal-interval: Time interval between deals in millisecond [number] [default: 2000]
--command-interval: Time interval between commands in millisecond [number] [default: 500]
--auto-cycle: Auto restart games if value >0 [number] [default: 0]
--help: Show help

Example: hearts_game_node-linux --port 8000 --round-interval 5000
```