Player Avenue, Inc.

Real-time CometD API

Player Avenue, Inc. 1

Real-time CometD API 1

Considerations 4

Player subscribes to game feed 4

Event: Player subscribes to game status channel 4

Event: Dealing new hand 4

Event: Waiting for small blind 5

Event: Small blind posted 5

Event: Small blind did not post 6

Event: Waiting for big blind 6

Event: Big blind posted 6

Event: Big blind did not post 7

Event: Dealing pocket cards 7

Event: Dealing flop 7

Event: Dealing turn 8

Event: Dealing river 8

Event: Showdown (normal) 9

Event: End game (last man standing) 10

Event: Waiting for current player to make a move 11

Event: Player called 11

Event: Player checked 12

Event: Player raised 12

Event: Player folded 13

Event: Player left game 13

Player joins game 14

Pre-conditions 14

Action: Player queues up for the game 14

New hand is dealt 15

Event: New hand is dealt but player didn’t make it to the hand 15

Event: New hand is dealt and player made it to the hand 15

Small blind 16

Event: Player needs to post small blind 16

Action: Player posts small blind 16

Action: Player does not post small blind 17

Big blind 17

Event: Player needs to post big blind 17

Action: Player posts big blind 18

Action: Player does not post big blind 18

Pocket cards 19

Event: Players are dealt pocket cards 19

Bet 20

Event: Player needs to bet 20

Action: Player calls 20

Action: Player checks 21

Action: Player raises 21

Action: Player folds 22

Leave game 23

Action: Player leaves game 23

Considerations

All values need to be passed over cometd using Strings. However, when the fields are defined under Values, they are defined as the primitive it exists as in the DB and what it can be converted to in Javascript if it makes sense.

Player subscribes to game feed

Event: Player subscribes to game status channel

Channel

1. /service/subscribe/game

Values

1. pokergameId:String – Poker game natural key
2. playerId:String - Player natural key

Example

{

pokergameId: “1”,

playerId: “1”

}

Event: Dealing new hand

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. players:Object – Player list
4. player:Object – Player information
5. nickname:String – Player nickname
6. seatNumber:Integer – Player seat number
7. amount:BigDecimal – Player amount on table
8. avatar:String – Player avatar image

Example

{

event: “DEALING\_HAND\_EVENT”,

handId: “1”,

players:

{

player:

{

nickname: “DeathFromAbove”,

seatNumber: “1”,

amount: “100.00”,

avatar: “images/avatar/deathfromabove.jpg”,

}

player:

{

nickname: “Undertaker”,

seatNumber: “2”,

amount: “100.00”,

avatar: “images/avatar/undertaker.jpg”,

}

}

}

Event: Waiting for small blind

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. nickname:String – Player nickname

Example

{

event: “WAITING\_FOR\_SMALL\_BLIND\_EVENT”,

nickname: “DeathFromAbove”,

handId: “1”

}

Event: Small blind posted

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. nickname:String – Player nickname

Example

{

event: “SMALL\_BLIND\_POSTED\_EVENT”,

nickname: “DeathFromAbove”,

handId: “1”

}

Event: Small blind did not post

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. nickname:String – Player nickname

Example

{

event: “SMALL\_BLIND\_NOT\_POSTED\_EVENT”,

nickname: “DeathFromAbove”,

handId: “1”

}

Event: Waiting for big blind

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. nickname:String – Player nickname

Example

{

event: “WAITING\_FOR\_BIG\_BLIND\_EVENT”,

nickname: “Undertaker”,

handId: “1”

}

Event: Big blind posted

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. nickname:String – Player nickname

Example

{

event: “BIG\_BLIND\_ POSTED\_EVENT”,

nickname: “Undertaker”,

handId: “1”

}

Event: Big blind did not post

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. nickname:String – Player nickname

Example

{

event: “BIG\_BLIND\_NOT\_POSTED\_EVENT”,

nickname: “Undertaker”,

handId: “1”

}

Event: Dealing pocket cards

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier

Example

{

event: “DEALING\_POCKET\_CARDS\_EVENT”,

handId: “1”,

}

Event: Dealing flop

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. cards:String – Card and Image url

Example

{

event: “DEALING\_FLOP\_EVENT”,

handId: “1”,

cards:

{

card1: “4h”,

card2: “10s”,

card3: “2h”

}

}

Event: Dealing turn

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. cards:String – Card and Image url

Example

{

event: “DEALING\_TURN\_EVENT”,

handId: “1”,

cards:

{

card1: “4h”

}

}

Event: Dealing river

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. cards:String – Card and Image url

Example

{

event: “DEALING\_RIVER\_EVENT”,

handId: “1”,

cards:

{

card1: “4h”

}

}

Event: Showdown (normal)

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. winner:Object – Winning player
4. nickname:String – Player nickname
5. amountWon:BigDecimal – Amount won
6. handName:String – Poker hand name
7. cards:Object - Cards

Example

{

event: “SHOWDOWN\_EVENT”,

handId: “1”,

winners

{

winner:

{

nickname: “DeathFromAbove”,

amountWon: “200”,

handName: “Four of a kind”,

cards:

{

card1: “4h”,

card2: “4s”,

card3: “4d”,

card4: “4c”,

card5: “5h”

}

},

}

losers:

{

loser:

{

nickname: “Undertaker”,

handName: “Three of a kind”,

cards:

{

card1: “3h”,

card2: “3s”,

card3: “3d”,

card4: “2c”,

card5: “5c”

}

}

}

}

Event: End game (last man standing)

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. winner:Object – Winning player
4. nickname:String – Player nickname
5. amountWon:BigDecimal – Amount won
6. handName:String – Poker hand name
7. cards:Object - Cards

Example

{

event: “ENDGAME\_EVENT”,

handId: “1”,

winner:

{

nickname: “DeathFromAbove”,

amountWon: “200”,

}

}

Event: Waiting for current player to make a move

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. nickname:String – Player nickname

Example

{

event: “WAITING\_FOR\_PLAYER\_RESPONSE\_EVENT”,

nickname: “Undertaker”,

handId: “1”

}

Event: Player called

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. pokergameId – Pokergame Id
4. player:String – Player nickname
5. amount:BigDecimal – Bet amount
6. pot:BigDecimal – New pot amount
7. seatNumber:Integer – Player seat number
8. nextPlayer:String – Next player nickname
9. nextSeatNumber:Integer – Next player seat number

Example

{

event: “PLAYER\_CALLED\_EVENT”,

handId: “1”,

pokergameId: “1”,

player: “DeathFromAbove”,

amount: “10.0”,

pot: “50”,

seatNumber: “1”,

nextPlayer: “Undertaker”,

nextSeatNumber: “2”,

}

Event: Player checked

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. player:String – Player nickname
4. seatNumber:Integer – Player seat number
5. nextPlayer:String – Next player nickname
6. nextSeatNumber:Integer – Next player seat number

Example

{

event: “PLAYER\_CHECKED\_EVENT”,

handId: “1”,

player: “DeathFromAbove”,

seatNumber: “1”,

nextPlayer: “Undertaker”,

nextSeatNumber: “2”,

}

Event: Player raised

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. player:String – Player nickname
4. amount:BigDecimal – Bet amount
5. pot:BigDecimal – New pot amount
6. seatNumber:Integer – Player seat number
7. nextPlayer:String – Next player nickname
8. nextSeatNumber:Integer – Next player seat number

Example

{

event: “PLAYER\_RAISED\_EVENT”,

handId: “1”,

player: “DeathFromAbove”,

amount: “10.0”,

pot: “50”,

seatNumber: “1”,

nextPlayer: “Undertaker”,

nextSeatNumber: “2”,

}

Event: Player folded

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. player:String – Player nickname
4. seatNumber:Integer – Player seat number
5. nextPlayer:String – Next player nickname
6. nextSeatNumber:Integer – Next player seat number

Example

{

event: “PLAYER\_FOLDED\_EVENT”,

handId: “1”,

player: “DeathFromAbove”,

seatNumber: “1”,

nextPlayer: “Undertaker”,

nextSeatNumber: “2”,

}

Event: Player left game

Channel

1. /service/game/1

Values

1. event:String – Server game event
2. handId:Long – Hand identifier
3. player:String – Player nickname
4. seatNumber:Integer – Player seat number
5. nextPlayer:String – Next player nickname
6. nextSeatNumber:Integer – Next player seat number

Example

{

event: “PLAYER\_LEFT\_EVENT”,

handId: “1”,

player: “DeathFromAbove”,

seatNumber: “1”,

nextPlayer: “Undertaker”,

nextSeatNumber: “2”,

}

Player joins game

Pre-conditions

1. Player is watching the game

Action: Player queues up for the game

Channel

1. /service/user/<systemUserId>

Values

1. action:String – Player action
2. pokergameId:Long – Poker game identifier
3. playerId:Long – Player identifier
4. buyin:BigDecimal – Player buyin amount
5. mustHaveSeat:Boolean – Indicates whether player MUST have desired seat
6. seatNumber:Integer – Desired seat number
7. success:Integer – 1 = true, 0 = false

Example

{

action: “QUEUE\_ACTION”,

pokergameId: “1”,

playerId: “1”,

buyin: “10.5”,

mustHaveSeat: “false”,

seatNumber: “1”

}

Success

{

success: “1”,

message: “You are number 11 in the queue.”

}

Failure

{

success: “0”,

message: “You do not have sufficient funds in your account.”

}

New hand is dealt

Event: New hand is dealt but player didn’t make it to the hand

Channel

1. /service/user/<systemUserId>

Values

1. event:String – Server game event
2. message:String – Server message

Example

{

event: “STILL\_QUEUED\_EVENT”,

message: “You are number 1 in the queue”

}

Event: New hand is dealt and player made it to the hand

Channel

1. /service/user/<systemUserId>

Values

1. event:String – Server game event
2. seatNumber:Integer – Player seat number
3. gamblerId:String – Gambler natural key for one hand
4. isDealer:Boolean – Specifies if player is dealer
5. isSmallBlind:Boolean – Specifies if player is small blind
6. isBigBlind:Boolean – Specifies if player is big blind
7. handId:Long – Hand identifier

Example

{

event: “DEALING\_NEW\_HAND\_ EVENT”,

seatNumber: “1”,

gamblerId: “1”,

isDealer: “false”,

isSmallBlind: “true”,

isBigBlind: “false”,

handId: “1”

}

Small blind

Event: Player needs to post small blind

Channel

1. /service/user/<systemUserId>

Values

1. event:String – Server game event
2. time:Integer – Seconds to complete task
3. amount:BigDecimal – Small blind amount
4. handId:String – Hand natural key
5. pokergameId:String – Poker game natural key

Example

{

event: “POST\_SMALL\_BLIND\_EVENT”,

time: “60”,

amount: “5.00”,

handId: “djflkjsf”,

pokergameId: “ueoru”

}

Action: Player posts small blind

Channel

1. /service/user/<systemUserId>

Values

1. action:String – Player action
2. gamblerId:Long – Gambler identifier
3. success:Integer – 1 = true, 0 = false
4. message:String – Server message

Example

{

action: “POST\_SMALL\_BLIND\_ACTION”,

gamblerId: “1”

}

Success

{

success: “1”,

message: “Small blind posted successfully”,

}

Failure

{

success: “0”,

message: “Unable to post small blind”

}

Action: Player does not post small blind

Channel

1. /service/user/<systemUserId>

Values

1. action:String – Player action
2. gamblerId:Long – Gambler identifier
3. success:Integer – 1 = true, 0 = false
4. message:String – Server message

Example

{

action: “NOT\_POST\_SMALL\_BLIND\_ACTION”,

gamblerId: “1”

}

Success

{

success: “1”,

message: “You chose to sit out this game”,

}

Failure

{

success: “0”,

message: “Unable to not post small blind”

}

Big blind

Event: Player needs to post big blind

Channel

1. /service/user/<systemUserId>

Values

1. event:String – Server game event
2. time:Integer – Seconds to complete task
3. amount:BigDecimal – Big blind amount
4. handId:String – Hand natural key
5. pokergameId:String – Poker game natural key

Example

{

event: “POST\_BIG\_BLIND\_EVENT”,

time: “60”,

amount: “10.00” ,

handId: “djflkjsf”,

pokergameId: “ueoru”

}

Action: Player posts big blind

Channel

1. /service/user/<systemUserId>

Values

1. action:String – Player action
2. gamblerId:Long – Gambler identifier
3. success:Integer – 1 = true, 0 = false
4. message:String – Server message

Example

{

action: “POST\_BIG\_BLIND\_ACTION”,

gamblerId: “1”

}

Success

{

success: “1”,

message: “Big blind posted successfully”,

}

Failure

{

success: “0”,

message: “Unable to post big blind”

}

Action: Player does not post big blind

Channel

1. /service/user/<systemUserId>

Values

1. action:String – Player action
2. gamblerId:Long – Gambler identifier
3. success:Integer – 1 = true, 0 = false
4. message:String – Server message

Example

{

action: “NOT\_POST\_BIG\_BLIND\_ACTION”,

gamblerId: “1”

}

Success

{

success: “1”,

message: “You chose to sit out this game”,

}

Failure

{

success: “0”,

message: “Unable to not post big blind”

}

Pocket cards

Event: Players are dealt pocket cards

Channel

1. /service/user/<systemUserId>

Values

1. event:String – Server game event
2. pokergameId:String – Game natural key
3. handId:String – Hand natural key
4. cards:String[] - Cards

Example

{

event: “DEALING\_POCKET\_CARDS\_EVENT”,

pokergameId: “hkhad”,

handId: “jadljlkjd”,

cards:

{

card1: “4h”,

card2: “10s”,

}

}

Flop

Event: Players are dealt flop cards

Channel

1. /service/user/<systemUserId>

Values

1. event:String – Server game event
2. pokergameId:String – Game natural key
3. handId:String – Hand natural key
4. cards:String[] - Cards

Example

{

event: “DEALING\_POCKET\_CARDS\_EVENT”,

pokergameId: “hkhad”,

handId: “jadljlkjd”,

cards:

{

card1: “4h”,

card2: “10s”,

card3: “5s”,

card4: “3s”,

card5: “7c”,

}

}

Turn

Event: Players are dealt turn cards

Channel

1. /service/user/<systemUserId>

Values

1. event:String – Server game event
2. pokergameId:String – Game natural key
3. handId:String – Hand natural key
4. cards:String[] - Cards

Example

{

event: “DEALING\_POCKET\_CARDS\_EVENT”,

pokergameId: “hkhad”,

handId: “jadljlkjd”,

cards:

{

card1: “4h”,

card2: “10s”,

card3: “5s”,

card4: “3s”,

card5: “7c”,

}

}

Bet

Event: Player needs to bet

Channel

1. /service/user/<systemUserId>

Values

1. event:String – Server game event
2. time:Integer – Seconds to complete task
3. canCall:Boolean – Can player call
4. canCheck:Boolean – Can player check
5. canRaise:Boolean – Can player raise
6. minAmount:BigDecimal – Minimum bet amount
7. maxAmount:BigDecimal – Maximum bet amount
8. handId – Hand natural key
9. pokergameId – Poker game natural key

Example

{

handId: “adad”,

pokergameId: “asasas”,

event: “BET\_EVENT”,

time: “60”,

canCall: “true”,

canCheck: “false”,

canRaise: “true”,

minAmount: “10.00”,

maxAmount: “40”,

}

Action: Player calls

Channel

1. /service/user/<systemUserId>

Values

1. action:String – Player action
2. gamblerId:Long – Gambler identifier
3. success:Integer – 1 = true, 0 = false
4. message:String – Server message

Example

{

action: “CALL\_ACTION”,

gamblerId: “1”

}

Success

{

success: “1”,

message: “You called successfully”,

}

Failure

{

success: “0”,

message: “Bet unsuccessful”

}

Action: Player checks

Channel

1. /service/user/<systemUserId>

Values

1. action:String – Player action
2. gamblerId:Long – Gambler identifier
3. success:Integer – 1 = true, 0 = false
4. message:String – Server message

Example

{

action: “CHECK\_ACTION”,

gamblerId: “1”

}

Success

{

success: “1”,

message: “You checked successfully”,

}

Failure

{

success: “0”,

message: “Bet unsuccessful”

}

Action: Player raises

Channel

1. /service/user/<systemUserId>

Values

1. action:String – Player action
2. gamblerId:Long – Gambler identifier
3. amount:BigDecimal – Raise amount (excluding call)
4. success:Integer – 1 = true, 0 = false
5. message:String – Server message

Example

{

action: “RAISE\_ACTION”,

gamblerId: “1”,

amount: 40

}

Success

{

success: “1”,

message: “You raised successfully”,

}

Failure

{

success: “0”,

message: “Bet unsuccessful”

}

Action: Player folds

Channel

1. /service/user/<systemUserId>

Values

1. action:String – Player action
2. gamblerId:Long – Gambler identifier
3. success:Integer – 1 = true, 0 = false
4. message:String – Server message

Example

{

action: “FOLD\_ACTION”,

gamblerId: “1”

}

Success

{

success: “1”,

message: “You folded successfully”,

}

Failure

{

success: “0”,

message: “Fold unsuccessful”

}

Leave game

Action: Player leaves game

Channel

1. /service/user/<systemUserId>

Values

1. action:String – Player action
2. playerId:Long – Player identifier
3. pokergameId:Long – Poker game identifier
4. success:Integer – 1 = true, 0 = false
5. message:String – Server message

Example

{

action: “LEAVE\_ACTION”,

playerId: “1”,

pokergameId: “1”

}

Success

{

success: “1”,

message: “You left the game”,

}

Failure

{

success: “0”,

message: “Couldn’t leave the game”

}