

MaxQuest

Betsoft Gaming

Version 1.5

1. Lobby Protocol

This document describes the communication protocol between client and the server for the multiplayer game “HeroQuest”.

Interaction is carried out via websockets in JSON format.

The lobbyUrl and session id of the logged-in user are passed to the client through the http request parameters: lobbyUrl (wss://host:port/websocket/mplobby), sessionId.

Please note, that we recommend to use only secure websocket connections (wss://).

Rooms can be created and deleted by the server as needed, depending on the current number of players.

Type of all identifiers (id, rid, roomid, ..) is long.

1.1 Error handling

In response to any client request may come an error message like this:

```
{  
  "code": 1,  
  "msg": "Internal error",  
  "date": 1496748898812,  
  "class": "Error",  
  "rid": 1  
}
```

```
{  
  "code": 2,  
  "msg": "Server shutdown",  
  "date": 1496748898812,  
  "class": "Error",  
  "rid": -1  
}
```

Request parameters:

code

error code

msg

error description (may be localized)

date

datetime in msec. from 01.01.1970 (standart unix timestamp)

class

message class

rid

request Id, type is int. Optional parameter, if not -1 this parameter contains id of original request.

Possible error ranges

1-999

FATAL ERROR

1000-4999

ERROR

5000-9999

WARNING

Handling 'FATAL ERROR':

Display message to user and close lobby.

Handling Lobby Websocket errors:

In case of websocket error, you must re-establishing websocket connection to lobbyUrl and send 'Enter Lobby' request.

1.2 Enter Lobby

This request is a first request which should be send by the client after establishing websocket connection to lobby. After that request, the client will be subscribed to receive asynchronous events.

```
{  
    "sid": "4_eb0bf169cbb477b6f8430000015fbe83_fwRFDR4FWFZTV142PiQUCQ4H",  
    "date": 1496750162302,  
    "class": "EnterLobby",  
    "rid": 1,  
    "serverId": 1,  
    "nickname": "Taras"  
    "avatarId": 0  
}
```

Request parameters:

sid

session identifier (send as request parameter on start lobby client).

rid

unique request id

serverId

serverId for connect (send as request parameter on start lobby client).

nickname

preferred nickname, displayed in game. Real assigned nickname may be differ, if selected nickname already used (Taras70 for example). See 4.1 section in game specification.

avatarId

preffered avatar identifier. Optional, if not specified, the default avatar will be used (avatarId=0)

If session not found, invalid or expired server return

```
{  
  "code": 3,  
  "msg": "Invalid session",  
  "date": 1496750162288,  
  "class": "Error",  
  "rid": 1  
}
```

If selected nickname contains obscene word, server return:

```
{  
  "code": 1000,  
  "msg": "Illegal nickname",  
  "date": 1496750162288,  
  "class": "Error",  
  "rid": 1  
}
```

If session exist and enter to lobby is success, server return:

```
{  
  "players": 600,  
  "nickname": "Taras70",  
  "balance": 20000,  
  "currency": "USD",  
  "date": 1496908464148,  
  "class": "EnterLobbyResponse",  
  "rankPonts": 200,  
  "rid": 2,  
  "alreadySeatRoomId": 20  
}
```

Response parameters:

players

total players logged in lobby

nickname

player nickname

balance

current player balance in cents (this is casino side balance, ALL available money)

currency

player currency. Further in this document, all money in player currency

rankPoints

total number of points received by the player, see 4.3 section is spec.

rid

unique request id

alreadySeatRoomId

roomId if player already seat in room; -1 if not. If a player occupies a place in a room, the lobby client must prompt the player to continue the game. If the player agrees, the game is started for the given room (send request “GetStartGameUrl” and open game client window)

1.3 GetRooms

After entering the lobby, the client requests a list of game rooms using a request type:

```
{  
    "type": "UNDISCOVERED_EGYPT",  
    "mode": "REAL",  
    "date": 1496909199002,  
    "class": "GetRooms",  
    "rid": 3  
}
```

Request parameters:

type

room type, at this moment always ‘UNDISCOVERED_EGYPT’. In the future, other games can be added.

mode

money type, possible values: FREE, REAL

rid

unique request id

Response:

```
{  
    "rid": 4,  
    "rooms": [  
        {  
            "id": 1,  
            "name": "VIP#1",  
            "seats": 3,  
            "maxSeats": 8,  
            "minBuyIn": 10000,  
            "stake": 10,  
            "state": "WAIT"  
        },  
        {  
            "id": 2,  
            "name": "Baby#1",  
            "seats": 8,  
            "maxSeats": 8,  
            "minBuyIn": 20000,  
            "stake": 20,  
            "state": "PLAY"  
        }  
    ],  
    "date": 1496919661664,  
    "class": "GetRoomsResponse"  
}
```

Where:

rid

request id from “GetRooms”

rooms

list game rooms

room:id

unique room id (type is long)

room:name

room name

room:seats

current seats count

room:maxSeats

max seats in room

room:minBuyIn

minimal buyIn in cents for occupy seat

room:stake

cost per shot in cents (type – integer)

room:state

current room state. Possible values: WAIT, PLAY, QUALIFY, CLOSED Player can take a place only in a room with state WAIT. Rooms with states PLAY and QUALIFY could be entered only as observer. CLOSED state indicates that room is not available and must be removed. Also see messages “CreateRoom”, “RemoveRoom” which can be sent asynchronously after creation or removal of rooms.

1.4 GetRoomInfo

When the user selects a particular game room, the client requests the server for detailed information about the selected room by a request like:

```
{  
    "roomId": 1,  
    "rid": 5,  
    "date": 1496921495447,  
    "class": "GetRoomInfo"  
}
```

Request parameters:

roomId

unique room id (see 1.3 GetRooms room:id)

rid

unique request id

Response:

```
{  
    "roomId": 1,  
    "rid": 5,  
    "name": "VIP#1",  
    "maxSeats": 8,  
    "minBuyIn": 10000,  
    "stake": 10,  
    "state": "WAIT",  
    "ttnx": 7,  
    "mapId": 1,  
    "width": 800,  
    "height": 600,  
    "alreadySitInNumber": -1,  
    "alreadySitInAmmoAmount": 0,  
    "alreadySitInBalance": 0,  
    "date": 1497004331658,  
    "class": "GetRoomInfoResponse",  
}
```

```
"seats": [
  {
    "id": 1,
    "nickname": "Taras",
    "avatarId": 0,
    "enterDate": 1497004331679,
    "totalScore": 700,
    "currentScore": 10,
    "rankPonts": 120
  },
  {
    "id": 2,
    "nickname": "Bulba",
    "avatarId": 1,
    "enterDate": 1497004331679,
    "totalScore": 230,
    "currentScore": 15,
    "rankPonts": 180
  },
  {
    "id": 3,
    "nickname": "Mikola",
    "avatarId": 1,
    "enterDate": 1497004331679,
    "totalScore": 10,
    "currentScore": 5,
    "rankPonts": 220
  }
],
"enemies": [
  {
    "id": 1,
    "name": "MummyA",
    "width": 10,
    "height": 30,
    "speed": 10,
    "prizes": 3,
    "sumAward": 6,
    "energy": 0,
    "skins": 1,
    "boss": false
  },
  {
    "id": 2,
    "name": "Boss",
    "width": 15,
    "height": 40,
    "speed": 70,
    "prizes": 4,
    "sumAward": 0,
    "energy": 50,
    "boss": true
  }
]
```

```

        "skins": 2,
        "boss":true
    }
],
"roomEnemies": [
    {
        "id": 18684,
        "typeId": 1,
        "speed": 10.0,
        "awardedPrizes": "",
        "awardedSum": 0.0,
        "energy": 0,
        "skin": 1,
        "trajectory": [
            { "x": 10, "y": 50, "time": 120 },
            { "x": 20, "y": 30, "time": 150 }
        ]
    },
    {
        "id": 18693,
        "typeId": 8,
        "speed": 10.0,
        "awardedPrizes": "",
        "awardedSum": 0.0,
        "energy": 0,
        "skin": 1,
        "trajectory": [
            { "x": 15, "y": 40, "time": 120 },
            { "x": 25, "y": 70, "time": 190 }
        ]
    }
],
"weapons": [
    {
        "id": 1,
        "name": "Gun",
        "shots": 3
    },
    {
        "id": 2,
        "name": "Shotgun",
        "shots": 5
    }
]
}

```

Where:

rid

request id from “GetRoomInfo”

roomId

unique room id

name

room name

maxSeats

max seats in room

minBuyIn

minimal buyIn in cents for occupy seat

stake

stake (shot cost) in cents (type – integer)

state

current room state. Possible values: WAIT, PLAY, QUALIFY, CLOSED

ttnx

rough time to next state in seconds, -1 for unknown (ttnx -Time To NeXt).

mapId

current map identifier

width

virtual width of the playing area

height

virtual height of the playing area

alreadySitInNumber

seat number; -1 if player not seat in room;

alreadySitInAmmoAmount

current ammo amount if player seat in room; 0 if player not seat

alreadySitInBalance

current player balance if player seat in room; 0 if player not seat

seats

list of players who have taken their seat and ready to play (or already playing)

enemies

list of enemy types that can appear in this room.

roomEnemies

list of alive enemies in this room. Please do not jumble it, “roomEnemies” are specific instances, but “enemies” are the types of the enemies.

weapons

list of weapons available in the room

seat:id

seat number, may be 0...(maxSeats-1)

seat:nickname

seat nickname

seat:avatarId

seat avatar

seat:enterDate

date when the player started the game

seat:totalScore

total number of points awarded player in this room (total win = totalScore*stake), type=integer

seat:currentScore

number of points awarded player for current round (currentWin = currentScore*stake), type=integer

seat:rankPoints

total number of points received by the player, see 4.3 section is spec.

enemy:id

unique enemy type identifier

enemy:name

enemy name

enemy:width,height

rectangle in which should be placed enemy

enemy:speed

max speed in points/sec.

enemy:prizes

number of prizes available for this type of enemy

enemy:sumAward

total awarded points available for this type of enemy (real win = sumAward*stake)

enemy:energy

Boss only property (can be used to identify the boss). Total points available for this boss. Each energy point represents 1 credit.

enemy:skins

skins count available for this enemy.

enemy:boss

flag indicating that this enemy is the boss; when this enemy type appears, you need remove the special weapon and use regular weapon.

roomEnemy:id

unique enemy identifier

roomEnemy:typeid

enemy type identifier (reference to enemy.id)

roomEnemy:awardedPrizes

prizes (list!) received from hitting the enemy. May be empty if no hits.

roomEnemy:awardedSum

amount of points received from hits to the enemy

roomEnemy:energy

remaining energy, if energy<0 enemy is dead, remember that energy=0 used for identify regular enemies

roomEnemy:skin

skin number (randomly generated, from 1 to enemy.skins)

roomEnemy:trajectory

Trajectory (set of points) along which enemies move

roomEnemy:trajectory:x, y

current x/y coordinate (x=0, y=0 for bottom left corner)

roomEnemy:trajectory:time

at this time the enemy must be at a given point

Weapons

is special weapons list. Special weapon may be used only in base game. When a special weapon is received, it must be used automatically. Once the weapon is received, the bet can not be changed until the weapon is fully consumed. Unused weapons are lost when going to the next round. Shots using special weapons are not paid.

weapon:id

unique weapon type identifier

weapon:name

weapon display name

weapon:shots

number of shots

1.5 SubscribeRoomInfo

This message should be sent to the server if you need to subscribe to the changes in the selected room.

```
{  
    "roomId": 1,  
    "rid": 5,  
    "date": 1497151414840,  
    "class": "SubscribeRoomInfo"  
}
```

Request parameters:

roomId

unique room id (see 1.3 GetRooms room:id)

rid

unique request id

Response:

```
{  
    "roomId": 5,  
    "state": "PLAY",  
    "seats": [  
        {  
            "id": 0,  
            "nickname": "John",  
            "avatarId": 0,  
            "enterDate": 1497152526897,  
            "totalScore": 700,  
            "currentScore": 30,  
            "rankPonts": 220  
        },  
        {  
            "id": 1,  
            "nickname": "Mike",  
            "avatarId": 1,  
            "enterDate": 1497152526897,  
            "totalScore": 230,  
            "currentScore": 100,  
            "rankPonts": 120  
        },  
        {  
            "id": 3,  
            "nickname": "Fred",  
            "avatarId": 0,  
            "enterDate": 1497152526897,  
            "totalScore": 10,  
            "currentScore": 10,  
            "rankPonts": 20  
        },  
        {  
            "id": 4,  
            "nickname": "Andrey",  
            "avatarId": 2,  
            "enterDate": 1497152526897,  
            "totalScore": 0,  
            "currentScore": 10,  
            "rankPonts": 1500  
        }  
    "ttnx": 600,  
    "date": 1497152526865,  
    "rid": 1,  
    "mapId": 1,  
    "class": "SubscribeRoomInfoResponse"  
}
```

response parameters same as “GetRoomInfoResponse”

1.6 UnsubscribeRoomInfo

This message should be send to the server for unsubscribe to the changes in specified room.

```
{  
    "roomId": 1,  
    "date": 1497154444100,  
    "rid": 5,  
    "class": "UnsubscribeRoomInfo"  
}
```

Request parameters:

roomId

unique room id (see 1.3 GetRooms room:id)

rid

unique request id

Response:

```
{  
    "date": 1497154639465,  
    "rid": 1,  
    "class": "Ok"  
}
```

1.7 GetStartGameUrl

After player double click on any room, lobby client send request to server for obtain url for start game.

```
{  
    "roomId": 1,  
    "date": 1497154790288,  
    "rid": 5,  
    "class": "GetStartGameUrl"  
}
```

Request parameters:

roomId

unique room id

rid

```
unique      request      id      Response:      {      "roomId":      5,      "startGameUrl":  
"https://host/mpgameloader.jsp?sid=deb0bf169cbb477b6f8430000015cbe83&serverId=1&roomId  
=5", "date": 1497155412200, "rid": 1, "class": "GetStartGameUrlResponse" }
```

Where:

rid

request id from “GetRoomInfo”

roomId

unique room id

startGameUrl

url for start game.

After receiving this response, the client should open a new window with startGameUrl. You can add any parameters (roomMode for example) that you need to startGameUrl, except already added.

1.8 GetLeaderBoard

Leaderboard should be displayed in lobby. See 4.2 section in specification for details.

```
{  
  "period": "DAILY",  
  "date": 1497154790288,  
  "rid": 5,  
  "class": "GetLeaderBoard"  
}
```

Request parameters:

period

leader board period, possible values: DAILY, WEEKLY, MONTHLY, ALL

rid

unique request id

Response:

```
{  
    "timeToEnd": 18000,  
    "leaders": [  
        {  
            "nickname": "John",  
            "scores": 5000,  
            "country": "US"  
        },  
        {  
            "nickname": "Mikola",  
            "scores": 4500,  
            "country": "UA"  
        }  
    "date": 1507708328757,  
    "rid": 10,  
    "class": "GetLeaderBoardResponse"  
}
```

Where:

rid

request id from “GetLeaderBoard”

timeToEnd

countdown to end of competition (in seconds). For period=ALL always 0;

leaders:nickname

player nickname

leaders:scores

player total score

leaders:country

player country, must be ISO 3166 two letter country code

1.9 CreateRoom

This message is sent by the server after creating a new room

```
{
    "roomId": 10,
    "date": 1497154444100,
    "rid": -1,
    "name": "VIP#10",
    "seats": 0,
    "maxSeats": 6,
    "minBuyIn": 10000,
    "stake": 10,
    "state": "WAIT",
    "ttnx": -1,
    "width": 800,
    "height": 600,
    "mapId": 1,
    "class": "CreateRoom"
}
```

Where:

roomId

unique room id (type is long)

date

datetime in msec. from 01.01.1970 (standart unix timestamp)

rid

request id, always -1

name

room name

seats

current seats count

maxSeats

max seats in room

minBuyIn

minimal buyIn in cents for occupy seat

stake

cost per shot in cents (type – integer)

state

current room state. Possible values: WAIT, PLAY, QUALIFY, CLOSED

ttnx

rough time to next state in seconds, -1 for unknown (ttnx -Time To NeXt).

width

virtual width of the playing area

height

virtual height of the playing area

mapId

current map identifier

class

message class

1.10 RemoveRoom

This message is sent by the server after deleting the room.

```
{  
  "roomId": 10,  
  "date": 1497154444100,  
  "rid": -1,  
  "class": "RemoveRoom"  
}
```

Where:

roomId

unique room id (type is long)

date

datetime in msec. from 01.01.1970 (standart unix timestamp)

rid

request id, always -1

class

message class

2. Game protocol

After the game client is loaded, it should open websocket connection to url passed in request parameter gameServletUrl=wss://host:port/websocket/mpgame and then send “OpenRoom” request.

2.1 Error handling

In response to any client request may come an error message like this:

```
{  
  "code": 1,  
  "msg": "Internal error",  
  "date": 1496748898812,  
  "class": "Error",  
  "rid": 1  
}
```

```
{  
  "code": 2,  
  "msg": "Server shutdown",  
  "date": 1496748898812,  
  "class": "Error",  
  "rid": -1  
}
```

```
{  
  "code": 1005,  
  "msg": "Room moved to another server",  
  "date": 1496748898812,  
  "class": "Error",  
  "rid": 9  
}
```

```
{  
  "code": 1003,  
  "msg": "Room not found",  
  "date": 1496748898812,  
  "class": "Error",  
  "rid": 10  
}
```

Request parameters:

code

error code

msg

error description (may be localized)

date

datetime in msec. from 01.01.1970 (standart unix timestamp)

class

message class

rid

request Id, type is int. Optional parameter, if not -1 this parameter contains id of original request.

Possible error ranges

1-999

FATAL ERROR

1000-4999

ERROR

5000-9999

WARNING

Handling 'FATAL ERROR':

Just display message to user and close game.

Handling Game Websocket errors:

In case of websocket error, you must re-establishing websocket connection to gameServletUrl and resend 'OpenRoom' request.

Handling 1003, 1005 error:

1. Close current websocket connection to game server
2. Call lobby method 'GetStartGameUrl'
3. Re-establish websocket connection to new startGameUrl
4. Send 'OpenRoom' request

2.2 OpenRoom

After this request is completed, the client goes into observer mode and begins to receive all necessary information about the game.

```
{  
    "roomId": 1,  
    "serverId": 1,  
    "lang": "en",  
    "sid": "eb0bf169cbb477b6f8430000015cbe83",  
    "date": 1497157072066,  
    "rid": 5,  
    "class": "OpenRoom"  
}
```

Request parameters:

roomId

unique room id

serverId

serverId for connect via websocket

lang

preferred language

rid

unique request id

sid

session identifier send as request parameter on start game client.

Response:

GetRoomInfoResponse. See paragraph 1.4

If game state is “PLAY” client immediate receive message:

```
{
    "date": 1497170115111,
    "rid": 5,
    "class": "FullGameInfo",
    "mapId": 1,
    "roomEnemies": [
        {
            "id": 1,
            "typeId": 1,
            "awardedPrizes": "1,2",
            "awardedSum": 3.0,
            "energy": 0,
            "skin": 1,
            "trajectory": [
                { "x": 10, "y": 50, "time": 120 },
                { "x": 20, "y": 30, "time": 150 }
            ]
        },
        {
            "id": 10,
            "typeId": 2,
            "awardedPrizes": "1,3",
            "awardedSum": 4.0,
            "energy": 45,
            "skin": 1,
            "trajectory": [
                { "x": 10, "y": 50, "time": 120 },
                { "x": 20, "y": 30, "time": 150 }
            ]
        }
    ]
}
```

Where

roomEnemy:id

unique enemy instance id

roomEnemy:typeId

enemy type id. See GetRoomInfoResponse:enemy:id

roomEnemy:awardedPrizes

prizes (list!) received from hitting the enemy. May be empty if no hits.

roomEnemy:awardedSum

amount of points received from hits to the enemy

roomEnemy:energy

remaining energy, if energy<0 enemy is dead, remember that energy=0 used for identify regular

enemies

roomEnemy:skin

enemy skin

2.3 CloseRoom

This request must be send to server before close client. This required for release all server resources allocated to client.

```
{  
    "roomId": 1,  
    "date": 1497159892576,  
    "rid": 5,  
    "class": "CloseRoom"  
}
```

Request parameters:

roomId

unique room id

rid

unique request id

Response:

```
{  
    "date": 1497154639465,  
    "rid": 1,  
    "class": "Ok"  
}
```

2.4 SitIn

The request is sent by the client to seat in the room. After the successful execution of this request, the user moves from the observer state to the player's state and additional actions available to him. After this request is executed, an amount equal to `ammoAmount*stake` will be debited from casino player balance.

```
{  
    "date": 1497160498696,  
    "rid": 1,  
    "ammoAmount": 100,  
    "class": "SitIn"  
}
```

Request parameters:

ammoAmount

amount of ammunition purchased

Response:

```
{  
    "id": 4,  
    "nickname": "Andrey",  
    "ammoAmount": 120,  
    "balance": 15000,  
    "avatarId": 0,  
    "enterDate": 1497160789104,  
    "date": 1497160789056,  
    "specialWeaponId": 1,  
    "remainingSWSHots": 0,  
    "rid": 1,  
    "class": "SitInResponse"  
}
```

Where:

rid

request id from “SitIn”

id

seat number

nickname

seat nickname

ammoAmount

summary ammo amount

balance

current player balance

avatarId

seat avatar

specialWeaponId

Player may already have a special weapon (SitIn request after reconnect to room). -1 if no special weapon.

remainingSWSHots

number of remaining shots for current special weapon. For regular weapon, always=0

Other clients also receive this response, but with rid=-1.

2.5 SitOut

The request is sent by the client in order to seat out from the room. After that the user goes into observer mode.

```
{  
    "date": 1497161230259,  
    "rid": 1,  
    "class": "SitOut"  
}
```

Response:

```
{  
    "id": 4,  
    "nickname": "Andrey",  
    "outDate": 1497161341687,  
    "date": 1497161341653,  
    "rid": 1,  
    "class": "SitOutResponse"  
}
```

Where:

rid

request id from “SitOut”

id

seat number

nickname

seat nickname

Other clients also receive this response, but with rid=-1.

2.6 GameStateChanged

This message sent from server to all clients on each game state change.

```
{  
    "state": "PLAY",  
    "ttnx": 600,  
    "date": 1497171419692,  
    "rid": 1,  
    "class": "GameStateChanged"  
}
```

Where:

state

current room state. Possible values: WAIT, PLAY, QUALIFY, CLOSED

ttx

rough time to next state in seconds, -1 for unknown

rid

request id, always -1

2.7 NewEnemy

This message sent from server to all clients when there is a new enemy

```
{  
    "newEnemy": {  
        "id": 1,  
        "typeId": 1,  
        "awardedPrizes": "",  
        "awardedSum": 0,  
        "energy": 0,  
        "skin": 2,  
        "trajectory": [  
            { "x": 10, "y": 50, "time": 120 },  
            { "x": 20, "y": 30, "time": 150 }  
        ]  
    },  
    "date": 1497173799426,  
    "rid": -1,  
    "class": "NewEnemy"  
}
```

newEnemy is same as FullGameInfo:roomEnemies in paragraph 2.2

2.8 BuyIn

The request is sent by the client if need buy ammo.

```
{  
    "ammoAmount": 100,  
    "date": 1497174203319,  
    "rid": 10,  
    "class": "BuyIn"  

```

ammoAmount

amount of ammunition purchased

Response:

```
{  
    "ammoAmount": 120,  
    "balance": 15000,  
    "date": 1497174620105,  
    "rid": 10,  
    "class": "BuyInResponse"  
}
```

Where:

rid

request id from “BuyIn” request

ammoAmount

summary ammo amount

balance

current balance in cents

2.9 ChangeStake

Deprecated. Stake cannot be changed, it is fixed for the room.

2.10 Shot

This request is sent by the client when the shot is fired. Response for shot request returned to all players. If a special weapon was used, then more than one answer can be sent / miss

```
{  
    "enemyId": 5,  
    "x": 200,  
    "y": 300,  
    "date": 1497177502148,  
    "rid": 34,  
    "class": "Shot"  
}
```

Where:

enemyId

unique enemy id of who made the shot

x,y

shoot coordinates

If player miss, server return

```
{  
    "date": 1497177502148,  
    "rid": -1,  
    "seatId": 2,  
    "killedMiss": true,  
    "awardedWeaponId": -1,  
    "enemyId": 2,  
    "usedSpecialWeapon": 1,  
    "remainingSWSHots": 3,  
    "score": 5,  
    "class": "Miss"  
}
```

Where:

seatId

seat id that made shot

killedMiss

true if a shot is fired on an already killed enemy. The money for this shot is returned to the player.

awardedWeaponId

if special weapon awarded, awardWeaponId contains weapon id, -1 if no award. Any shot has a chance to get special weapons

usedSpecialWeapon

special weapon id used for this shot. For regular weapon, always=-1

remainingSWSHots

number of remaining shots for current special weapon. For regular weapon, always=0

enemyId

unique enemy id

If player hit, server return:

```
{
  "seatId": 1,
  "damage": 1,
  "win": 50,
  "awardedWeaponId": -1,
  "usedSpecialWeapon": 1,
  "remainingSWSHots": 3,
  "enemy": {
    "id": 1,
    "typeId": 1,
    "x": 10,
    "y": 50,
    "angle": 270,
    "awardedPrizes": "1,3",
    "awardedSum": 4,
    "energy": 45,
    "score": 1,
    "skin": 1,
  },
  "date": 1497177502148,
  "rid": -1,
  "class": "Hit"
}
```

Where:

seatId

seat id that made a shot

damage

caused damage to the enemy

win

prize in cents

awardedWeaponId

if special weapon awarded, awardWeaponId contains weapon id, -1 if no award

usedSpecialWeapon

special weapon id used for this shot. For regular weapon, always=-1

remainingSWSHots

number of remaining shots for current special weapon. For regular weapon, always=0

enemy

state of the affected enemy, same as RoomEnemy in paragraph 2.2 enemy.id and all other attributes may be different from the one for whom the shot was fired. This is possible if the enemy is already killed by another player. The client must correctly process the animation of the rebound and the defeat of another enemy. Also keep in mind that the Hit.message does not

contain a field "trajectory", as it is not necessary.

rid

request id for the player who made the shot, -1 for all other players.

2.11 GetFullGameInfo

This request can be sent by the client if connection lost and need refresh all room info.

```
{  
    "date": 1497178236042,  
    "rid": 1,  
    "class": "GetFullGameInfo"  
}
```

Response: same as FullGameInfo in paragraph 2.2

2.12 RoundResult

After round finished all clients receive message “GameStateChanged” described in paragraph 2.6 with state=QUALIFY. After qualification completed all clients receive round result in message:

```
{
  "winAmount": 500,
  "balance": 12000,
  "currentScore": 30,
  "totalScore": 350,
  "hitCount": 5,
  "missCount": 25,
  "qualifyWin": 10,
  "specialWeaponCompensation": 10,
  "seats": [
    {
      "id": 0,
      "nickname": "John",
      "avatarId": 0,
      "enterDate": 1497179064797,
      "totalScore": 700.0,
      "currentScore": 7,
      "rankPonts": 200
    },
    {
      "id": 1,
      "nickname": "Mike",
      "avatarId": 1,
      "enterDate": 1497179064797,
      "totalScore": 230.0,
      "currentScore": 10,
      "rankPonts": 50
    },
    {
      "id": 3,
      "nickname": "Fred",
      "avatarId": 1,
      "enterDate": 1497179064797,
      "totalScore": 0,
      "currentScore": 0,
      "rankPonts": 2000
    }
  ],
  "date": 1497179064748,
  "rid": -1,
  "class": "RoundResult"
}
```

Where:

winAmount

amount won in this round

balance

current player balance

totalScore

total number of points awarded player in this room (total win = totalScore*stake), type=integer

currentScore

number of points awarded player for current round (currentWin = currentScore*stake), type=integer

hitCount

hit count for this round

missCount

miss count for this round

qualifyWin

hoard boss win amount (in cents)

specialWeaponCompensation

compensation amount for unused special weapon (in cents)

seat:rankPoints

total number of points received by the player, see 4.3 section is spec. seats is same as GetRoomInfoResponse:seats in paragraph 1.4

2.13 ChangeMap

The server can send a message about changing the current map (for example, when a boss appears)

```
{  
  "class": "ChangeMap",  
  "mapId": 2,  
  "reason": "BASE",  
  "rid": -1,  
  "date": 34534534  
}
```

Where:

mapId

new map identifier (now only one map with id=1 available)

reason

reason for changeMap (class of enemies available for this map, also may be classified as subround), possible values: BASE, BOSS, HOARD, HOARD_BOSS

2.14 GetRoomHistory

In progress, will be provided later

2.15 EnemyDestroyed

After the enemy was destroyed (for example, as a result of a shot), the server sends this message

```
{  
    "enemyId": 100,  
    "reason": 0,  
    "date": 1521872381930,  
    "rid": -1,  
    "class": "EnemyDestroyed"  
}
```

Where:

enemyId

destroyed enemy identifier

reason

identifier of the event as a result of which the enemy was destroyed. The client can use this field for the different death animations. Existing types: 0 - death as a result of a shot from regular weapons

2.16 Award

When enemy is killed, player can receive additional non-money award, which consist of leader-board points and collectible items.

When such award is granted, player will receive message:

```
{  
    "date": 1521872381930,  
    "rid": -1,  
    "seatId": 1,  
    "enemyId": 100,  
    "score": 250,  
    "treasures": [5, 10],  
    "class": "Award"  
}
```

Where:

seatId

seatId of player that received achievement, -1 if prize is awarded to each player

enemyId

id of enemy which was dropped this items after death

points

amount of awarded leader-board points, could be zero

treasures

list of ids of dropped treasures

For more info about available treasures, please see achievements specification.

2.17 Achievement

When player receives an achievement, he will receive message:

```
{  
    "date": 1521872381930,  
    "rid": -1,  
    "seatId": 1,  
    "type": 1,  
    "id": 10,  
    "level": 3,  
    "score": 100,  
    "class": "Achievement"  
}
```

Where:

seatId

seatId of player that received achievement

type

- 1 - Achievement for killing N enemies with the same skin
- 2 - Achievement for reaching certain level of killed enemies on a page
- 3 - Achievement for collecting N treasures of the same type
- 4 - Achievement for collecting certain amount of treasures on a page

id

- EnemyId*100+SkinId for type=1
- Enemy group id for type=2
- TreasureId for type=3
- Treasure group id for type=3

level

Achievement level

score

Amount of leaderboard points awarded for completing achievement

2.18 Reload

Converts current win back into the ammo

Request:

```
{  
    "date": 1521872381930,  
    "rid": 100,  
    "class": "Reload"  
}
```

Response:

```
{  
    "date": 1521872381930,  
    "rid": 100,  
    "ammoAmount": 150  
    "class": "ReloadResponse"  
}
```

Where:

ammoAmount

Total amount of ammo after reload

3. Enemy Skins

Table 1. Enemy Skins

Enemy Type	Skin Id	Description
0	1	Scarab
0	2	Black Scarab
1	1	Gold Scarab
2	1	Diamond Scarab
2	2	Ruby Scarab
3	1	Small Mummy
4	1	Small Black Mummy
5	1	Small White Mummy
6	1	Walking Mummy Warrior
6	2	Running Mummy Warrior
7	1	Mummy God Red
8	1	Mummy God Green
16	1	Anubis

Enemy Type	Skin Id	Description
16	2	Osiris
16	3	Thoth