WebSocket API

- Connection **{}://{ip}:{port}?roomName={roomName}&userMail={userMail}**

Notification

```
{
        method: string
        data: Object
}
```

Notifications

When a Notification source is **Client**, then the **Server** broadcasts the same notification.

userJoined

Fired by establishment of WebSocket connection.

- Source: **Server**
- Method: ‘userJoined’
- Data: {

```
                user: string,
                producers: [
                        {
                                type: MediaStreamTypes
                                id: ProducerID
                        }
                        ,...
                ]
        }
```

userLeft

Fired by WebSocket disconnection.

- Source: **Server**
- Method: ‘userLeft’
- Data: {

```
                user: string
        }
```

producerAdded

- Source: **Server**
- Method: ‘producerAdded’
- Data: {
  user: string,
  type: MediaStreamType,
  id: ProducerID
  }

producerRemoved

- Source: **Client/Server**
- Method: ‘producerRemoved’
- Data: {
  user: string,
  type: MediaStreamType,
  }

producerPaused

- Source: **Client/Server**
- Method: ‘producerPaused’
- Data: {
  user: string,
  type: MediaStreamType
  }

producerResumed

- Source: **Client/Server**
- Method: ‘producerResumed’
- Data: {
  user: string,
  type: MediaStreamType
  }

HTTP API

- API Path **/api/media**

Response

{
    “status”: ‘success’ or ‘failed’,
    “result”: null or object
}

Endpoints

exists

- Endpoint **/rooms/exists**
- Method **GET**
- Parameters

name	optional	type
roomName	no	string

- “result”: {
    “exists”: bool
  }

createRoom

- Endpoint **/rooms/createRoom**
- Method **POST**
- Parameters

name	optional	type
roomName	no	string
userMail	no	string

- “result”: {
    “created”: bool
  }

attendees

- Endpoint **/rooms/attendees**
- Method **GET**
- Parameters

name	optional	type
roomName	no	string

- “result”: [

```
                {
                        id: “attendee1@gmail.com”
                },
                ...
]
```

createAttendee

- Endpoint **/rooms/createAttendee**
- Method **POST**
- Parameters

name	optional	type
roomName	no	string
userMail	no	string

- “result”: {

```
        “rtpCapabilities”: RTPCapabilities,
        “sendParams”: {
                “id”: Transport ID,
                “iceParameters”: ICE Parameters,
                “iceCandidates”: ICE Candidates,
                “dtlsParameters”: DTLS Parameters
        },
        “recvParams”: {
                “id”: Transport ID,
                “iceParameters”: ICE Parameters,
                “iceCandidates”: ICE Candidates,
                “dtlsParameters”: DTLS Parameters
        }
 }
```

connectSender

- Endpoint **/rooms/connectSender**
- Method **POST**
- Parameters

name	optional	type
roomName	no	string
userMail	no	string
dtlsParameters	no	DTLS Parameters

createProducer

Tell the server to create a Producer, MediaKind is obtained from the MediaStreamType sent. Server emits *producerAdded* event only if *userMail* had already joined the room.

- Endpoint **/rooms/createProducer**
- Method **POST**
- Parameters

name	optional	type
roomName	no	string
userMail	no	string
type	no	MediaStreamTypes
rtpParameters	no	RTP Parameters

- “result” : {
        “id”: ProducerID
  }

producers

If an attendee is given in *producerMail*, filters only his producers.

- Endpoint **/rooms/producers**
- Method **GET**
- Parameters

name	optional	type
roomName	no	string
producerMail	yes	string

- “result”:

```
{
        “user1@gmail.com”: [
                {
                        “type”: MediaStreamTypes
                        “id”: ProducerID
                }, ...
        ], ...
}
```

connectReceiver

- Endpoint **/rooms/connectReceiver**
- Method **POST**
- Parameters

name	optional	type
roomName	no	string
userMail	no	string
dtlsParameters	no	DTLS Parameters

createConsumer

- Endpoint **/rooms/createConsumer**
- Method **POST**
- Parameters

name	optional	type
roomName	no	string
userMail	no	string
id	no	ProducerID
rtpCapabilities	no	RTP Capabilities
producerMail	no	string

- “result”:

```
{
        “id”: ConsumerID,
        “rtpParameters”: RTP Parameters
        “producerPaused”: bool
}
```