WebSocket Server API Documentation

WebSocket server for inbound call handling. Processes call-related events and handles binary audio data streaming. You need to implement your own WebSocket server and handle events and actions during WebSocket communication process.

Table of Contents

Server Information

- **Port**: 4143 (Customizable - define your own port and inform development team)
- **Protocol**: `ws://` (Can be either `ws://` or `wss://`)

Message Formats

The server handles two types of messages:

1. **JSON Text**: For signaling and control events
2. **Binary**: For audio data streaming

API Events

Client → Server Events

`incoming_call`

Sent when a new call is received by server.

Payload:

```
{
  "event": "incoming_call",
  "callerId": "string",
  "didNumber": "string",
  "sessionId": "string"
}
```

Parameters:

- `callerId`: Caller phone number
- `didNumber`: Destination phone number

- `sessionId`: Unique session identifier for the call

dtmf

Sent when a DTMF digit is pressed by the caller.

Payload:

```
{
  "event": "dtmf",
  "digit": "string"
}
```

Valid Digits: 0-9, *, #, A-D

hangup

Sent when client terminates the call.

Payload:

```
{
  "event": "hangup"
}
```

cdr

Sent after the call ends with call detail record information.

Payload:

```
{
  "event": "cdr",
  "sessionId": "string",
  "source": "string",
  "destination": "string",
  "startTime": "string",
  "answerTime": "string",
  "endTime": "string",
  "duration": number,
  "billableSeconds": number,
  "disposition": "string",
  "hangupBy": "string",
  "hangupCauseCode": string,
  "hangupCauseText": "string"
}
```

Parameters:

- `sessionId`: Unique session identifier for the call
- `source`: Source phone number
- `destination`: Destination phone number
- `startTime`: Call start time (ISO format)
- `answerTime`: Call answer time (ISO format)
- `endTime`: Call end time (ISO format)
- `duration`: Total call duration in seconds
- `billableSeconds`: Billable duration in seconds
- `disposition`: Call disposition status
- `hangupBy`: Who initiated hangup
- `hangupCauseCode`: Hangup cause code
- `hangupCauseText`: Hangup cause description

Server → Client Events

answer

Sent to answer the incoming call.

Payload:

```
{
    "event": "answer"
}
```

dtmf

Sent to initiate DTMF tone to the caller.

Payload:

```
{
    "event": "dtmf",
    "digit": "string",
    "duration": number
}
```

Parameters:

- `digit`: DTMF digit to send (`0-9`, `*`, `#`)
- `duration`: Tone duration in milliseconds (max 1000ms)

interrupt

Sent to interrupt audio streaming.
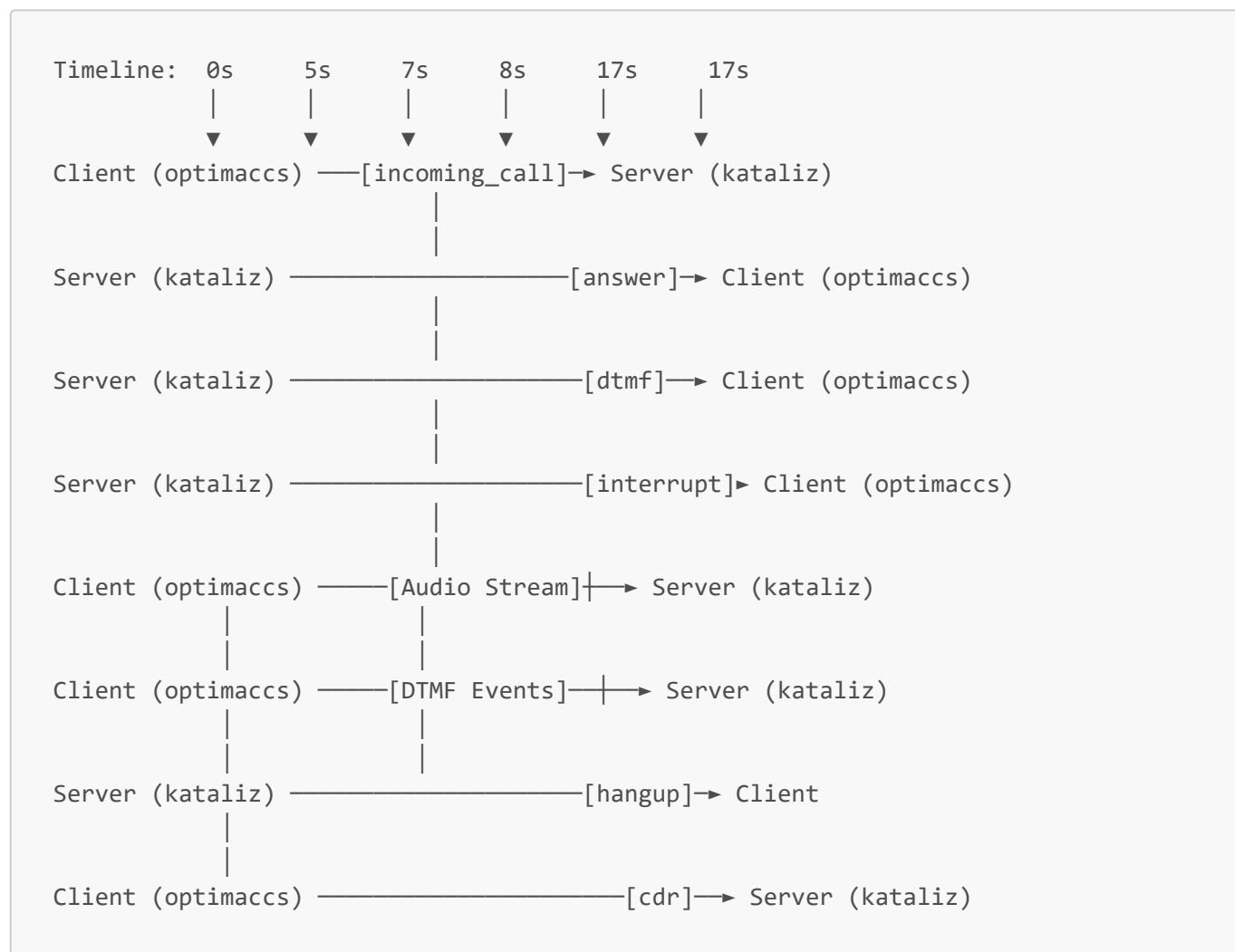
Payload:

```
{
  "event": "interrupt"
}
```

hangup

Sent to terminate the call.

Payload:

```
{
  "event": "hangup"
}
```

Call Flow

```
Timeline:  0s       5s       7s       8s       17s       17s
           |        |        |        |        |         |
           ▼        ▼        ▼        ▼        ▼         ▼
Client (optimaccs) ──[incoming_call]─► Server (kataliz)
                          |
                          |
Server (kataliz) ──────────────────[answer]─► Client (optimaccs)
                          |
                          |
Server (kataliz) ──────────────────[dtmf]─► Client (optimaccs)
                          |
                          |
Server (kataliz) ──────────────────[interrupt]► Client (optimaccs)
                          |
                          |
Client (optimaccs) ─────[Audio Stream]┼─► Server (kataliz)
                  |        |
                  |        |
Client (optimaccs) ─────[DTMF Events]─┼─► Server (kataliz)
                  |        |
                  |        |
Server (kataliz) ──────────────────[hangup]─► Client
                  |
                  |
Client (optimaccs) ──────────────────[cdr]─► Server (kataliz)
```

Detailed Flow:

1. **Connection** (0s)

   - Client connects to `ws://localhost:4143`

2. **Incoming Call** (5s)

   - Client sends `{ "event": "incoming_call", "callerId": "...", "didNumber": "...", "sessionId": "..." }`

3. **Answer** (5s)

   - Server sends `{ "event": "answer" }`

4. **Send DTMF** (7s)

   - Server sends `{ "event": "dtmf", "digit": "1", "duration": 200 }`

5. **Interrupt Audio** (8s)

   - Server sends `{ "event": "interrupt" }`

6. **Audio Stream & DTMF** (8s - 17s)

   - Client streams binary audio data
   - Server echoes audio data back
   - Client sends DTMF events: `{ "event": "dtmf", "digit": "1" }`

7. **Hangup** (17s)

   - Server sends `{ "event": "hangup" }`

8. **CDR** (17s)

   - Client sends call detail record: `{ "event": "cdr", "sessionId": "...", "source": "...", "destination": "...", "duration": ..., "billableSeconds": ..., "disposition": "...", "hangupBy": "...", "hangupCauseCode": "...", "hangupCauseText": "..." }`
   - Connection closes

Example Source Code

Complete example source code available at: https://github.com/sunvone/websocket-inbound-call