

config.js

- Analyzing the source code
 - 1. Get the room configuration
 - 2. Get the video streams list with access to local media devices
 - 3. Get the audio streams list with access to local media devices
 - 4. Get local media devices access and add the stream to list
 - 5. Get media devices access according to constraints
 - 5.1. Audio constraint set up
 - 5.2. Video constraints set up
 - 5.3. Screen capture constraints set up
 - 5.4. Get local media devices access

The functions to configure the room connection and local media access are moved to `config.js` module. The configuration file example follows:

config.json

```
{
  "room": {
    "url": "ws://127.0.0.1:8080",
    "name": "ROOM1",
    "pin": "1234",
    "nickName": "User1",
    "failedProbesThreshold": 5,
    "pingInterval": 5000
  },
  "media": {
    "audio": {
      "tracks": [{
        "source": "mic",
        "channels": 2,
        "type": "mic1"
      }]
    },
    "video": {
      "tracks": [
        {
          "source": "camera",
          "width": 1280,
          "height": 720,
          "codec": "H264",
          "constraints": {
            "frameRate": 25
          },
          "encodings": [
            { "rid": "180p", "active": true, "maxBitrate": 200000, "scaleResolutionDownBy": 4 },
            { "rid": "360p", "active": true, "maxBitrate": 500000, "scaleResolutionDownBy": 2 },
            { "rid": "720p", "active": true, "maxBitrate": 900000 }
          ],
          "type": "cam1"
        }
      ]
    }
  }
}
```

Analyzing the source code

To analyze `config.js` source code take the version available [here](#)

1. Get the room configuration

`getRoomConfig()` [code](#)

```
const getRoomConfig = function(config) {
  let roomConfig = {
    url: config.room.url || "ws://127.0.0.1:8080",
    roomName: config.room.name || "ROOM1",
    pin: config.room.pin || "1234",
    nickname: config.room.nickName || "User1"
  };
  if (config.room.failedProbesThreshold !== undefined) {
    roomConfig.failedProbesThreshold = config.room.failedProbesThreshold;
  }
  if (config.room.pingInterval !== undefined) {
    roomConfig.pingInterval = config.room.pingInterval;
  }
  return roomConfig;
}
```

2. Get the video streams list with access to local media devices

getVideoStreams() [code](#)

```
const getVideoStreams = async function(config) {
  let streams = [];
  if (config.media && config.media.video && config.media.video.tracks) {
    streams = await getStreams(config.media.video.tracks);
  }
  return streams;
}
```

3. Get the audio streams list with access to local media devices

getAudioStreams() [code](#)

```
const getAudioStreams = async function(config) {
  let streams = [];
  if (config.media && config.media.audio && config.media.audio.tracks) {
    streams = await getStreams(config.media.audio.tracks);
  }
  return streams;
}
```

4. Get local media devices access and add the stream to list

getStreams() [code](#)

```
const getStreams = async function(tracks) {
  let streams = [];
  for (let track of tracks) {
    let stream = await getMedia(track);
    if (stream) {
      streams.push({
        stream: stream,
        encodings: track.encodings,
        source: track.source,
        type: track.type
      });
    }
  }
  return streams;
}
```

5. Get media devices access according to constraints

5.1. Audio constraint set up

getMedia() [code](#)

```
const getMedia = async function(track) {
  //convert to constraints
  let screen = false;
  const constraints= {};
  if (track.source === "mic") {
    //audio
    constraints.audio = {};
    if (track.constraints) {
      constraints.audio = track.constraints;
    }
    constraints.audio.stereo = track.channels !== 1
    if (track.channels && track.channels === 2) {
      constraints.audio.echoCancellation = false;
      constraints.audio.googEchoCancellation = false;
    }
  } else if (track.source === "camera") {
    ...
  } else if (track.source === "screen") {
    ...
  }
  ...
  return stream;
}
```

5.2. Video constraints set up

getMedia() [code](#)

```
const getMedia = async function(track) {
  //convert to constraints
  let screen = false;
  const constraints= {};
  if (track.source === "mic") {
    ...
  } else if (track.source === "camera") {
    constraints.video = {};
    if (track.constraints) {
      constraints.video = track.constraints;
    }
    constraints.video.width = track.width;
    constraints.video.height = track.height;
  } else if (track.source === "screen") {
    ...
  }
  ...
  return stream;
}
```

5.3. Screen capture constraints set up

getMedia() [code](#)

```

const getMedia = async function(track) {
  //convert to constraints
  let screen = false;
  const constraints= {};
  if (track.source === "mic") {
    ...
  } else if (track.source === "camera") {
    ...
  } else if (track.source === "screen") {
    constraints.video = {};
    if (track.constraints) {
      constraints.video = track.constraints;
    }
    constraints.video.width = track.width;
    constraints.video.height = track.height;
    screen = true;
  }
  ...
  return stream;
}

```

5.4. Get local media devices access

getMedia() [code](#)

```

const getMedia = async function(track) {
  //convert to constraints
  let screen = false;
  const constraints= {};
  if (track.source === "mic") {
    ...
  } else if (track.source === "camera") {
    ...
  } else if (track.source === "screen") {
    ...
  }

  //get access to a/v
  let stream;
  if (screen) {
    stream = await navigator.mediaDevices.getDisplayMedia(constraints);
  } else {
    stream = await navigator.mediaDevices.getUserMedia(constraints);
  }
  return stream;
}

```