

## Video Wheels-Off Screenshots - Labs/Quickstart/ReactApp/Android

Credentials used in Lab:

Account SID - AC3daf41d39c3368308f467d37c077d33a

Auth Token - 9786b148fe6332fb30f5f54109e5b9e5

API Key - SKe55ee6121b76ebcb1ab3b56fcdf755a0

TWILIO\_API\_KEY\_SECRET - hdHrMFdExdQ29H20YhkynYVGjz9aYtR9

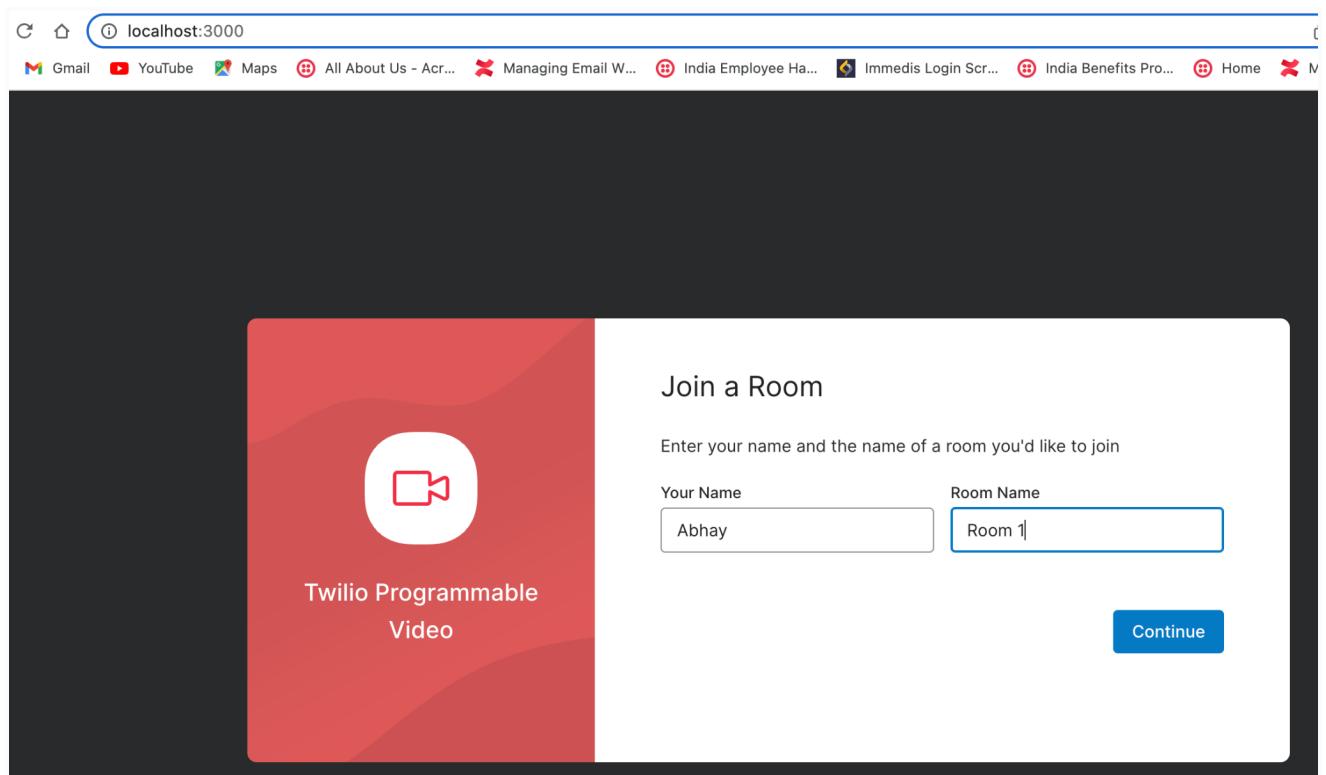
TWILIO\_CONVERSATIONS\_SERVICE\_SID - ISf9b037ceebfc41ae8a84d6c370285428

### Ahoy! Demo Apps- REACT DEMO APP

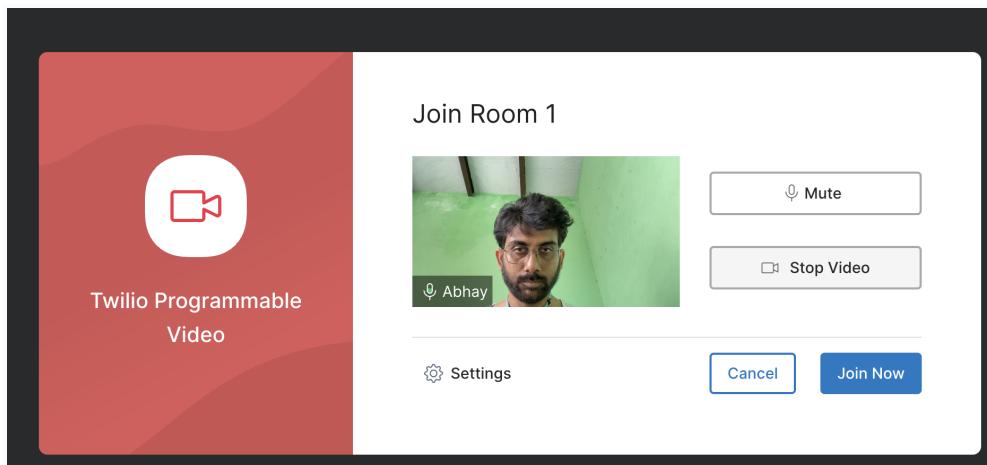
<https://github.com/twilio/twilio-video-app-react> JS SDK Ahoy! Demo App

Screen-shots of React Demo App:

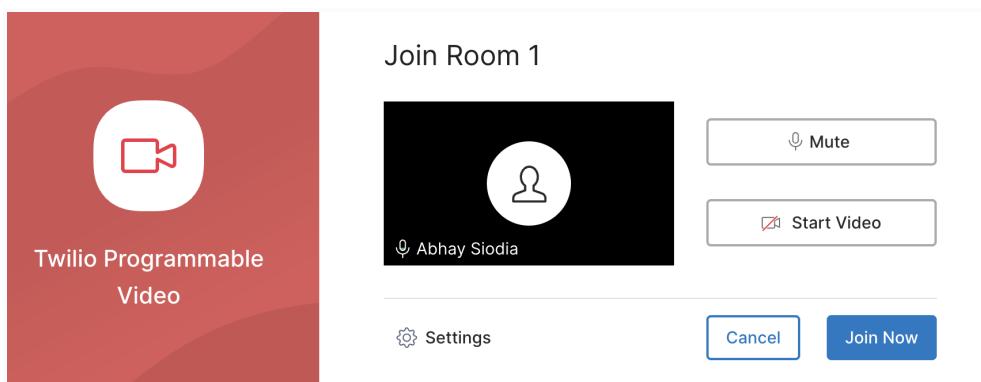
First Screenshot of App while adding entering in Room:



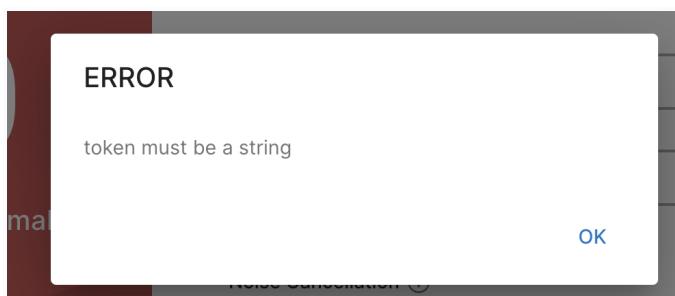
Screenshot #2 Participants joined Room 1.



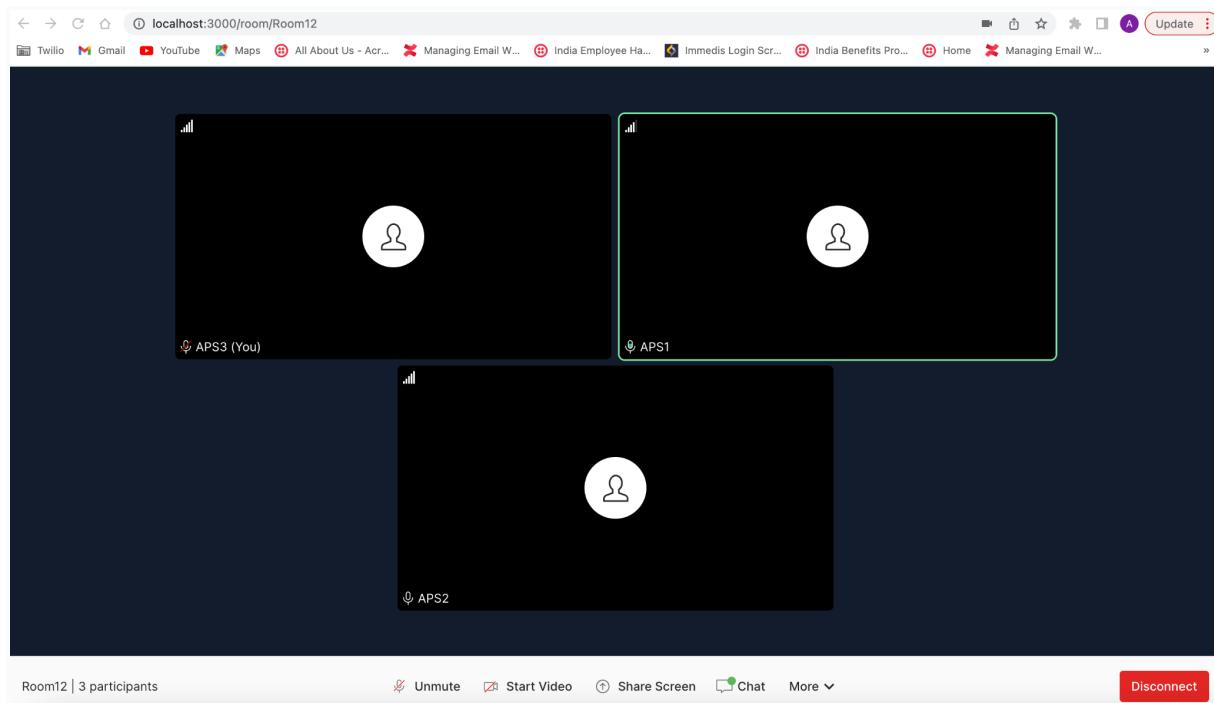
Screenshot #3 Video turned off.



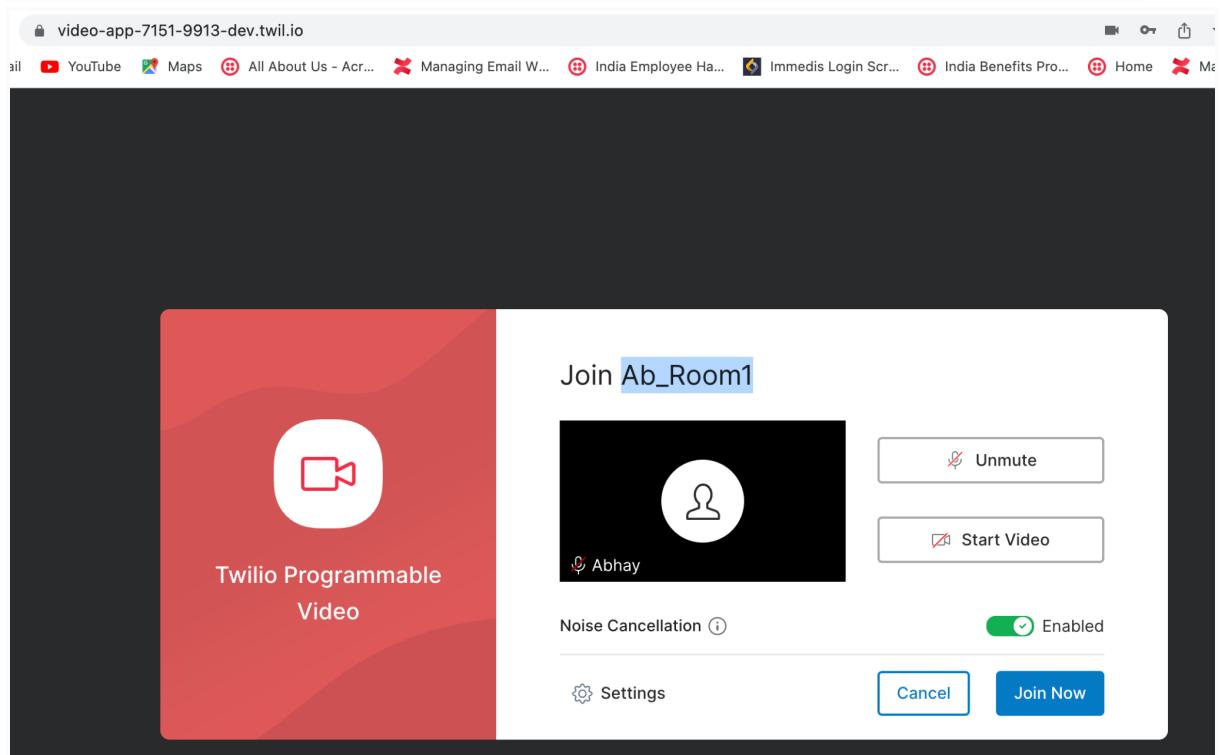
When same user in same room:



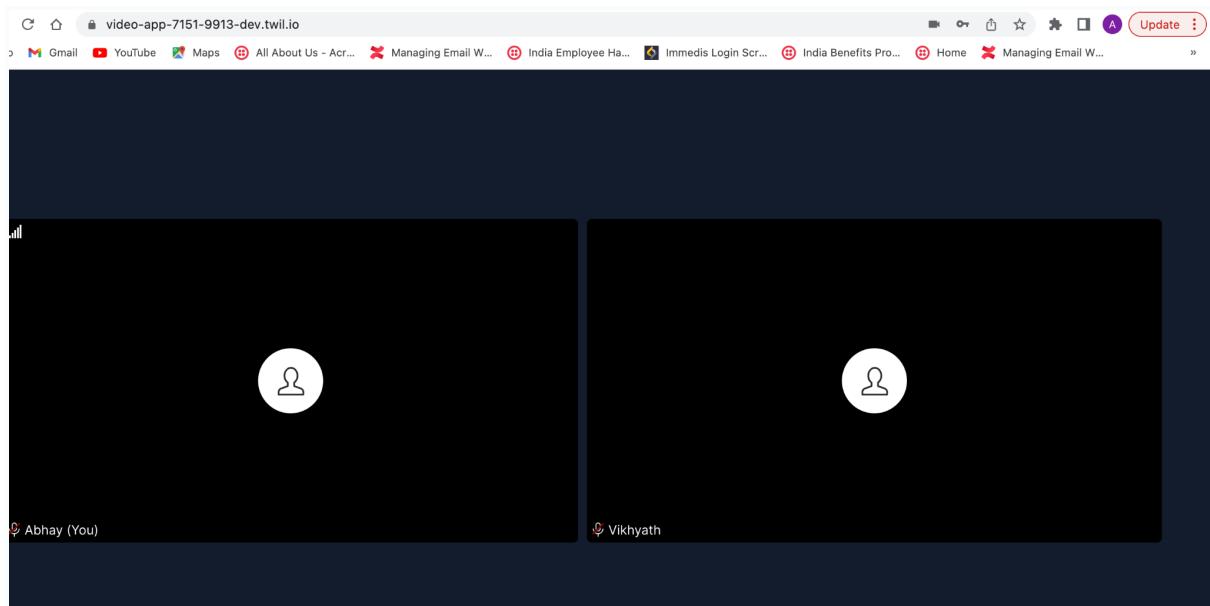
Screenshot #4 Multiple Participants joined Room:



Screenshot #5 Test done in different machine, Joined VS through Public URL:



Screenshot #6 Me and VS, joined same Room:



Screenshot #7, Terminal screenshot:

```
✓ TERMINAL

File sizes after gzip:
882.83 kB  build/static/js/main.1e69779e.js
3.41 kB    build/static/css/main.32f81bc4.css

The bundle size is significantly larger than recommended.
Consider reducing it with code splitting: https://goo.gl/9VhYWB
You can also analyze the project dependencies: https://goo.gl/LeUzfb

serve -s build

Find out more about deployment here:
https://cra.link/deployment

deploying app... done
Web App URL: https://video-app-7151-9913-dev.twil.io?passcode=03579371519913
Passcode: 035 793 7151 9913
Expires: Thu Jul 20 2023 15:23:30 GMT+0530
Room Type: group
Edit your token server at: https://www.twilio.com/console/functions/editor/ZS5e0a83fc10ec52b8364t/ZE1cb0ca62da31d9e980e2b9168af04907/function/ZHdfc5e0f0da67cb92ece49268bb754fba
○ asisodia@LRY7WF91V twilio-video-app-react %
```

Screenshot #8

```
> ▾ TERMINAL zsh + ~
```

Why you should do it regularly: <https://github.com/browserslist/update-db#readme>  
Browserslist: caniuse-lite is outdated. Please run:  
npx update-browserslist-db@latest  
Why you should do it regularly: <https://github.com/browserslist/update-db#readme>  
**Compiled successfully.**

File sizes after gzip:

882.83 kB build/static/js/main.1e69779e.js  
3.41 kB build/static/css/main.32f81bc4.css

The bundle size is significantly larger than recommended.  
Consider reducing it with code splitting: <https://goo.gl/9VhYWB>  
You can also analyze the project dependencies: <https://goo.gl/LeUzfb>

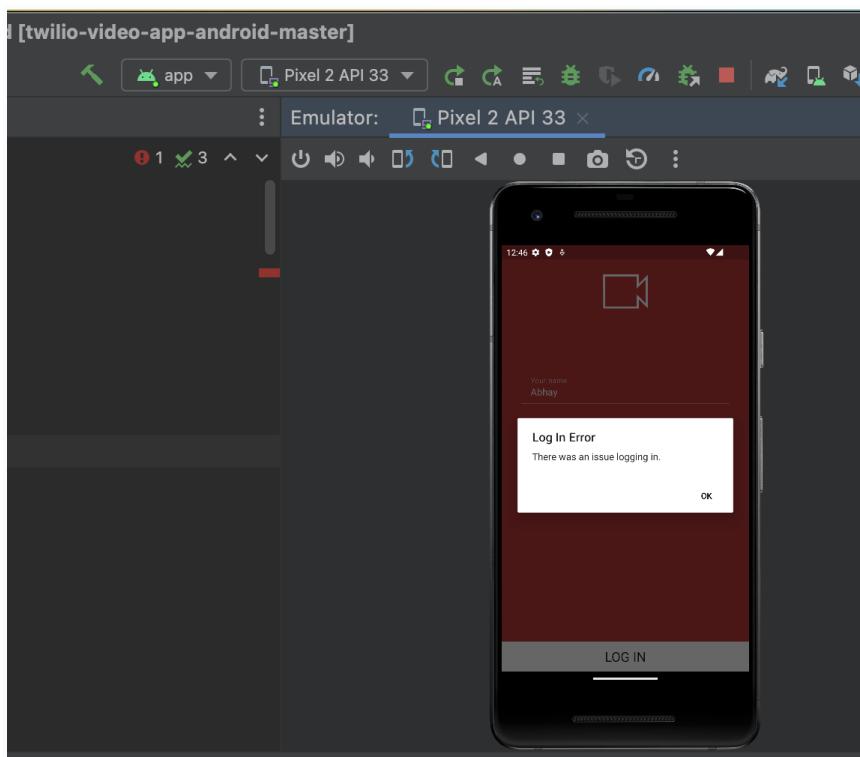
The project was built assuming it is hosted at /.  
A Video app is already deployed. Use the --override flag to override the existing deployment.  
Web App URL: <https://video-app-7680-4221-dev.twilio.io?passcode=76804221>  
Passcode: 76804221  
Expires: Thu Jan 01 1970 05:30:00 GMT+0530  
Edit your token server at: <https://www.twilio.com/console/functions/editor/Z5d80f8d3b36c12fda5709dcffe9b87c7/environment/ZEcc8e93f876ab661d3beed61c2252522e/function/ZH09c2b030d5b232c6ff6b709490b06ade>

asisodia@LRY7WF91V twilio-video-app-react %

## Android SDK Ahoy! Demo App

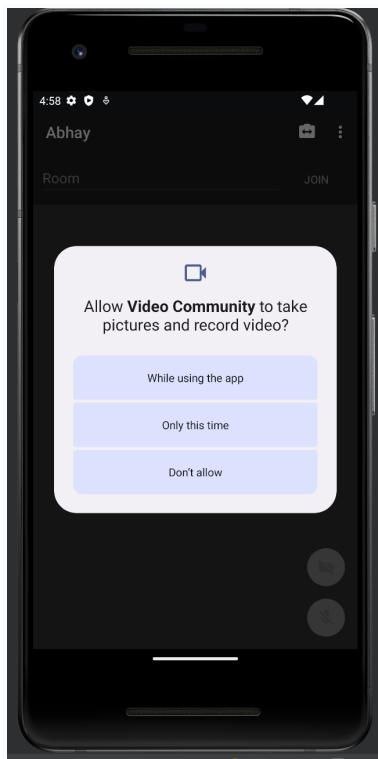
### Below are the screenshots of Android app

Screenshot #1 Lab started using Android studio:

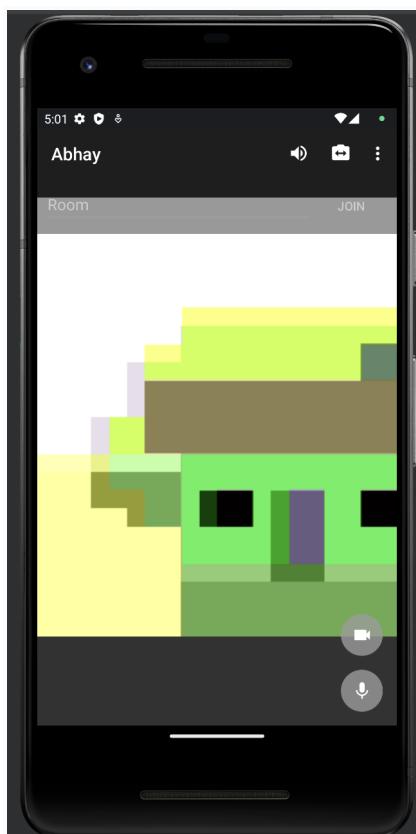


After generating passcode, i am able to login:

Screenshot #2 After credentials, asked for few permissions::



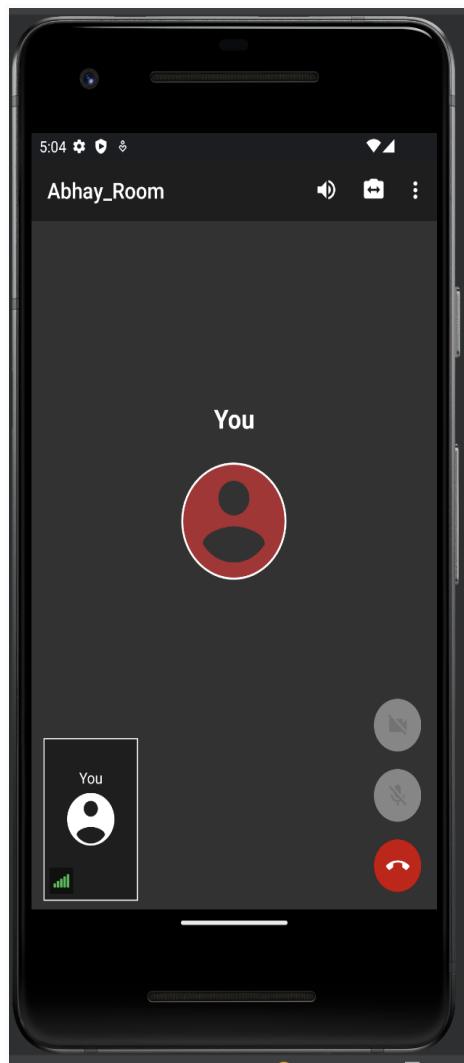
Screenshot #3 Abhay joined as first participant in Room:



Screenshot #5 Two participant joined same Room(Abhay\_Room):



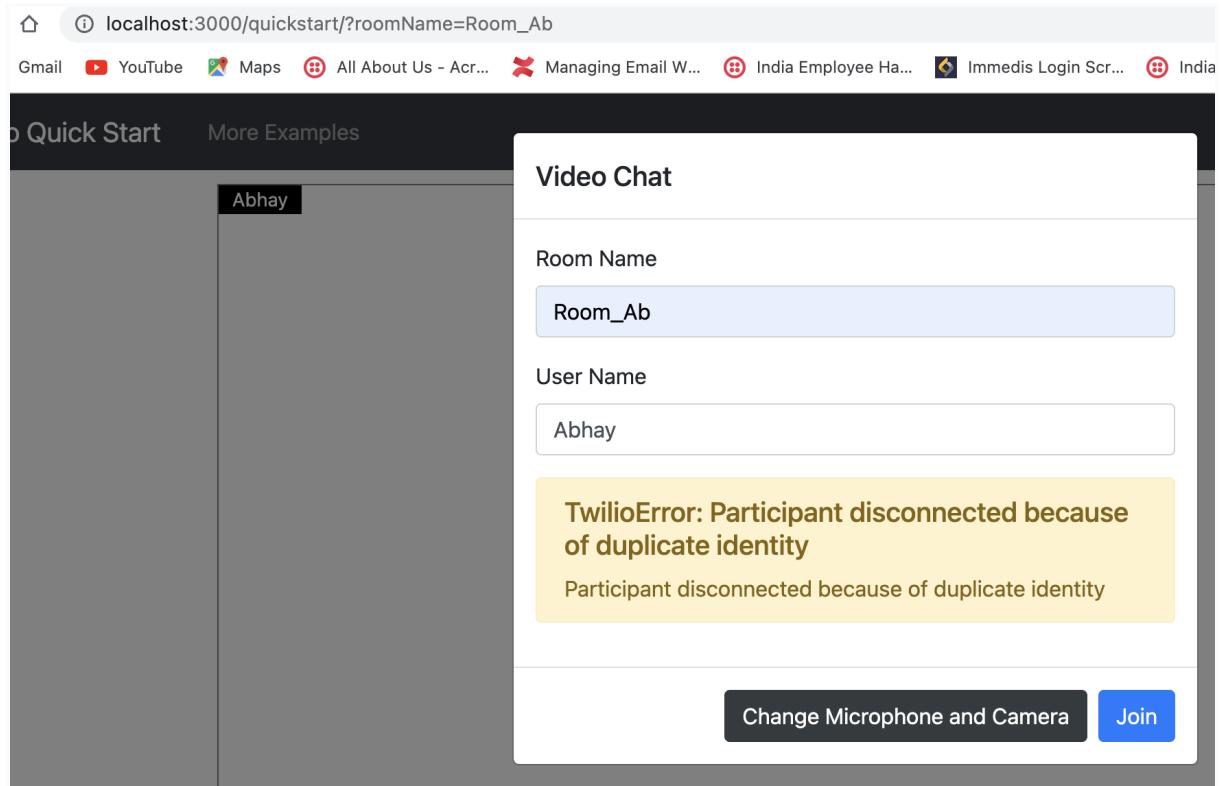
Screenshot #6 Two participant joined same Room(Abhay\_Room):



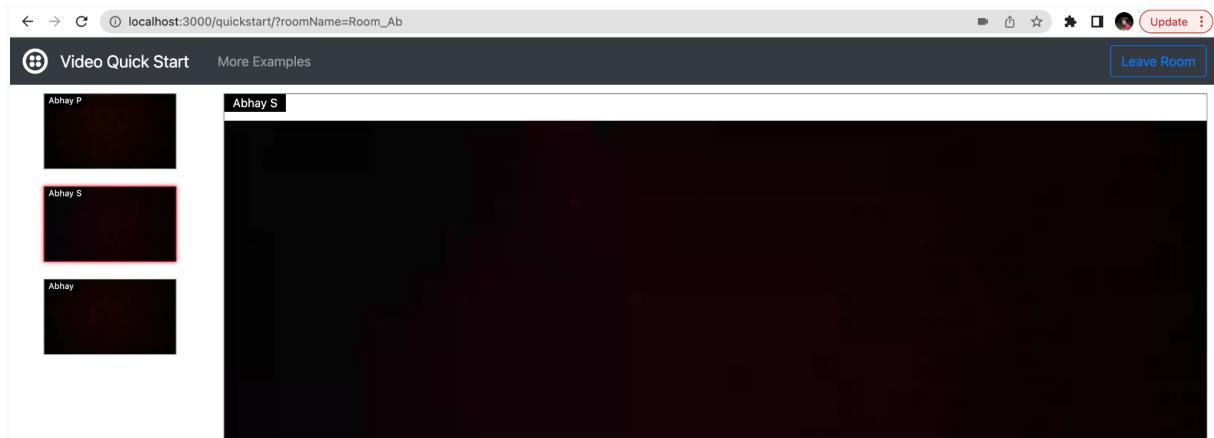
## JS SDK Quickstart

Below are the screenshots of JS Quickstart Lab:

Screenshot # Duplicate participant joined same room:



Screenshot # Three Participant joined same Room:



# JS QuickStart Examples:

## Example Screenshots: DataTracks

### Screenshot #1

**Data Tracks**

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to the given Room with a LocalDataTrack.
 * @param {string} token - AccessToken for joining the Room
 * @returns {CancelablePromise<Room>}
 */
async function connectToRoomWithDataTrack(token, roomName) {
  const localDataTrack = new Video.LocalDataTrack({
    name: 'chat',
  });

  const room = await Video.connect(token, {
    name: roomName,
    tracks: [localDataTrack],
  });

  return room;
}

/**
 * Send a chat message using the given LocalDataTrack.
 * @param {LocalDataTrack} dataTrack - The {@link LocalDataTrack}
 * @param {string} message - The message to be sent
 */
function sendChatMessage(dataTrack, message) {
  dataTrack.send(message);
}
```

**P1**

**P2**

**Disconnect from Room**



Message

**Submit**

**Connect to Room**



Message

**Submit**

Console Output:

```
2023-07-04T03:47:20.153Z info [connect #1] index.js:26063
Connecting to a Room
2023-07-04T03:47:20.153Z debug [connect #1] Options: index.js:26063
  <eventObserver: EventObserver, wsServer: 'wss://global.vss.twilio.com/signaling', automaticSubscription: true, dominantSpeaker: false, enableDscp: false, ...>
2023-07-04T03:47:20.153Z info [connect #1] Getting index.js:26063
LocalTracks
2023-07-04T03:47:20.153Z debug [connect #1] Options: index.js:26063
  <eventObserver: EventObserver, wsServer: 'wss://global.vss.twilio.com/signaling', automaticSubscription: true, dominantSpeaker: false, enableDscp: false, ...>
2023-07-04T03:47:20.153Z info [createLocalTrack #1] index.js:26063
Add a user-provided LocalTracks
2023-07-04T03:47:20.153Z debug [createLocalTracks index.js:26063
#1] LocalTracks: > [LocalDataTrack]
2023-07-04T03:47:20.154Z debug [connect #1] Creating index.js:26063
a new LocalParticipantV2 {events: {}, _eventsCount: 1, _maxListeners: undefined, ...}
2023-07-04T03:47:20.154Z info [LocalParticipant #1] index.js:26063
Created a new Participant
2023-07-04T03:47:20.154Z info [LocalParticipant #1] index.js:26063
Added a new LocalDataTrack: 29bc875c-3bfe-4fe9-92a9-0ae0a16658ce
2023-07-04T03:47:20.154Z debug [LocalParticipant #1] index.js:26063
LocalDataTrack:
  LocalDataTrack {kind: 'data', name: 'chat', id: '29bc875c-3bfe-4fe9-92a9-0ae0a16658ce', maxPacketLifeTime: null, maxRetransmits: null, ...}
```

### Screenshot #2

**Data Tracks**

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to the given Room with a LocalDataTrack.
 * @param {string} token - AccessToken for joining the Room
 * @returns {CancelablePromise<Room>}
 */
async function connectToRoomWithDataTrack(token, roomName) {
  const localDataTrack = new Video.LocalDataTrack({
    name: 'chat',
  });

  const room = await Video.connect(token, {
    name: roomName,
    tracks: [localDataTrack],
  });

  return room;
}

/**
 * Send a chat message using the given LocalDataTrack.
 * @param {LocalDataTrack} dataTrack - The {@link LocalDataTrack}
 * @param {string} message - The message to be sent
 */
function sendChatMessage(dataTrack, message) {
  dataTrack.send(message);
}

/**
 * Receive chat messages from RemoteParticipants in the Room
 * @param {Room} room - The Room you are currently in
 * @param {Function} onMessageReceived - Updates UI when a message is received
 */
function onMessageReceived(room, onMessageReceived) {
  room.on('message', (message) => {
    onMessageReceived(message);
  });
}
```

**P1**

**P2**

**Disconnect from Room**



Message

**Submit**

**Connect to Room**



Message

**Submit**

Console Output:

```
2023-07-04T03:47:21.833Z debug [Room #1: RM459f946dc150b91a808af364ae3be8e] Setting up RemoteParticipant creation for all subsequent ParticipantSignalings that connect to the RoomSignalings
2023-07-04T03:47:21.833Z info [Room #1: index.js:26063] RM459f946dc150b91a808af364ae3be8e] Created a new Room: Chat1
2023-07-04T03:47:21.833Z debug [Room #1: index.js:26063] RM459f946dc150b91a808af364ae3be8e] Initial RemoteParticipants: > []
2023-07-04T03:47:21.833Z debug [connect #1] Creating index.js:26063
a new LocalParticipantV2 {events: {}, _eventsCount: 1, _maxListeners: undefined, ...}
  > Room {localParticipant: LocalParticipant, name: 'Chat1', participants: Map(0), ...}
2023-07-04T03:47:21.834Z info [connect #1] Connected index.js:26063
Room: [Room #1: RM459f946dc150b91a808af364ae3be8e]
2023-07-04T03:47:21.834Z info [connect #1] Room: index.js:26063
  > Room {localParticipant: LocalParticipant, name: 'Chat1', participants: Map(0), ...}
2023-07-04T03:47:21.838Z debug [TwilioConnection #1: index.js:26063] RM459f946dc150b91a808af364ae3be8e"version":2,"type":"update","id":"RM459f946dc150b91a808af364ae3be8e","name":"Chat1","participants":[]}, "tracks":[]}, "revision":1,"state":"connected","media_warnings":[]}, "participants":[], "recording":{}}, "enabled":false, "revision":1, "is_recording":false}, "subscribe":{}}, "revision":1,"participants": [{"type": "include", "all": true}], "published": {"revision":1, "tracks": []}, "type": "msg"}]
2023-07-04T03:47:21.840Z debug [PeerConnectionV2 #1: index.js:26063] 3f97043b-05bf-46a2-ad2c-b97c545d968f] ICE gathering state is "gathering"
2023-07-04T03:47:21.840Z debug [PeerConnectionV2 #1: index.js:26063] 3f97043b-05bf-46a2-ad2c-b97c545d968f] Starting ICE gathering timeout: 15000
2023-07-04T03:47:21.841Z debug [PeerConnectionV2 #1: index.js:26063] 3f97043b-05bf-46a2-ad2c-b97c545d968f] ICE connection state is "closed"
2023-07-04T03:47:21.843Z debug [PeerConnectionV2 #1: index.js:26063] 3f97043b-05bf-46a2-ad2c-b97c545d968f] ICE connection state is "closed"
2023-07-04T03:47:21.844Z debug [PeerConnectionV2 #1: index.js:26063] 3f97043b-05bf-46a2-ad2c-b97c545d968f] Clearing ICE gathering timeout
2023-07-04T03:47:21.844Z debug [PeerConnectionV2 #1: index.js:26063] 3f97043b-05bf-46a2-ad2c-b97c545d968f] Clearing ICE gathering timeout
```

### Screenshot #3

## Data Tracks

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to the given Room with a LocalDataTrack.
 * @param {string} token - AccessToken for joining the Room
 * @returns {CancelablePromise<Room>}
 */
async function connectToRoomWithDataTrack(token, roomName) {
  const localDataTrack = new Video.LocalDataTrack({
    name: 'chat',
  });

  const room = await Video.connect(token, {
    name: roomName,
    tracks: [localDataTrack],
  });

  return room;
}

/**
 * Send a chat message using the given LocalDataTrack.
 * @param {LocalDataTrack} dataTrack - The @link LocalDataTrack
 * @param {string} message - The message to be sent
 */
function sendChatMessage(dataTrack, message) {
  dataTrack.send(message);
}

/**
 * Receive chat messages from RemoteParticipants in the Room
 * @param {Room} room - The Room you are currently in
 * @param {Function} onMessageReceived - Updates UI when a message is received
 */
function onMessageReceived(room, onMessageReceived) {
  room.on('message', (message) => {
    onMessageReceived(message);
  });
}
```

P1

[Disconnect from Room](#)

P2: has connected

Message

[Submit](#)

P2

[Disconnect from Room](#)

Message

[Submit](#)

```
2023-07-04T03:51:48.251Z debug [PeerConnectionV2 #2: index.js:26063] 52d70660-e080-4abf-a35b-e57a5348b42d] Clearing ICE gathering timeout
2023-07-04T03:51:48.353Z debug [PeerConnectionV2 #2: index.js:26063] 52d70660-e080-4abf-a35b-e57a5348b42d] ICE connection state is "connected"
2023-07-04T03:51:48.354Z debug [PeerConnectionV2 #2: index.js:26063] 52d70660-e080-4abf-a35b-e57a5348b42d] ICE gathering state is "complete"
2023-07-04T03:51:48.355Z debug [TwilioConnection #2: index.js:26063] wss://global.vss.twilio.com/signaling] Outgoing: {"body": {"session": "3da41439c3368304f67d37c07d33a459f946dbc150b91a8b0af364e3be", "name": "P1", "participants": [{"id": "3b4c8a26-9d94-468b-9d7c-5ba60216297d", "kind": "data", "name": "chat", "priority": "standard"}, {"id": "PA82760c3d11460b56c9b0208cd0aee647", "identity": "Oaken Lando Davenport", "tracks": [], "revision": 2, "state": "connected", "media_warnings": [], "participants": [{"sid": "PA1e786fe39e2f0de99af482e4dfdf88c8", "identity": "Mushy Anna Hanover", "tracks": [{"kind": "data", "priority": "standard", "id": "a4980cd7-de90-478e-b681-d023bf240be", "enabled": true, "sid": "MTead89c72ced43a8cbe4ba67dfbd2f10", "name": "chat", "state": "ready"}], "revision": 2, "state": "connected", "recording": {"enabled": false, "revision": 1, "is_recording": false}, "subscribed": {"revision": 0, "tracks": []}, "published": {"revision": 1, "tracks": [{"kind": "data", "priority": "standard", "id": "3b4c8a26-9d94-468b-9d7c-5ba60216297d", "enabled": true, "name": "chat", "state": "created"}]}, "type": "msg"}]
2023-07-04T03:51:48.628Z debug [TwilioConnection #2: index.js:26063] wss://global.vss.twilio.com/signaling] Incoming: {"body": {"version": 2, "type": "update", "sid": "RM459f946dc150b91a8b8af364ae3be", "name": "Chat1", "participant": {"sid": "PA82760c3d11460b56c9b0208cd0aee647", "identity": "Oaken Lando Davenport", "tracks": [], "revision": 2, "state": "connected", "media_warnings": [], "participants": [{"sid": "PA1e786fe39e2f0de99af482e4dfdf88c8", "identity": "Mushy Anna Hanover", "tracks": [{"kind": "data", "priority": "standard", "id": "a4980cd7-de90-478e-b681-d023bf240be", "enabled": true, "sid": "MTead89c72ced43a8cbe4ba67dfbd2f10", "name": "chat", "state": "ready"}], "revision": 2, "state": "connected", "recording": {"enabled": false, "revision": 1, "is_recording": false}, "subscribed": {"revision": 0, "tracks": []}, "published": {"revision": 1, "tracks": [{"kind": "data", "priority": "standard", "id": "3b4c8a26-9d94-468b-9d7c-5ba60216297d", "enabled": true, "name": "chat", "state": "created"}]}, "type": "msg"}}
2023-07-04T03:51:49.526Z debug [TwilioConnection #2: index.js:26063] wss://global.vss.twilio.com/signaling] Incoming: {"body": {"version": 2, "type": "update", "sid": "RM459f946dc150b91a8b8af364ae3be", "name": "Chat1", "participant": {"sid": "PA82760c3d11460b56c9b0208cd0aee647", "identity": "Oaken Lando Davenport", "tracks": [], "revision": 2, "state": "connected", "media_warnings": [], "participants": [{"sid": "PA1e786fe39e2f0de99af482e4dfdf88c8", "identity": "Mushy Anna Hanover", "tracks": [{"kind": "data", "priority": "standard", "id": "a4980cd7-de90-478e-b681-d023bf240be", "enabled": true, "sid": "MTead89c72ced43a8cbe4ba67dfbd2f10", "name": "chat", "state": "ready"}], "revision": 2, "state": "connected", "recording": {"enabled": false, "revision": 1, "is_recording": false}, "subscribed": {"revision": 0, "tracks": []}, "published": {"revision": 1, "tracks": [{"kind": "data", "priority": "standard", "id": "3b4c8a26-9d94-468b-9d7c-5ba60216297d", "enabled": true, "name": "chat", "state": "created"}]}, "type": "msg"}}
2023-07-04T03:51:49.551Z info [RemoteParticipant #4: index.js:26063] PA82760c3d11460b56c9b0208cd0aee647] Added a new DataTrack: 204a0698-b76d-48ed-b459-a232d0ff9d2
2023-07-04T03:51:49.551Z debug [RemoteParticipant #4: index.js:26063] #4: PA82760c3d11460b56c9b0208cd0aee647] DataTrack: RemoteDataTrack {data: "chat", isEnabled: true, maxPacketLifeTime: null, ...}
2023-07-04T03:51:49.551Z debug [TwilioConnection #1: index.js:26063] wss://global.vss.twilio.com/signaling] Incoming: {"body": {"version": 2, "type": "update", "sid": "RM459f946dc150b91a8b8af364ae3be", "name": "RM459f946dc150b91a8b8af364ae3be", "participants": [{"sid": "PA1e786fe39e2f0de99af482e4dfdf88c8", "identity": "Mushy Anna Hanover", "tracks": [{"kind": "data", "priority": "standard", "id": "a4980cd7-de90-478e-b681-d023bf240be", "enabled": true, "sid": "MTead89c72ced43a8cbe4ba67dfbd2f10", "name": "chat", "state": "ready"}], "revision": 2, "state": "connected", "media_warnings": [], "participants": [{"sid": "PA82760c3d11460b56c9b0208cd0aee647", "identity": "Oaken Lando Davenport", "tracks": [], "revision": 2, "state": "connected", "media_warnings": [], "participants": [{"sid": "PA1e786fe39e2f0de99af482e4dfdf88c8", "identity": "Mushy Anna Hanover", "tracks": [{"kind": "data", "priority": "standard", "id": "a4980cd7-de90-478e-b681-d023bf240be", "enabled": true, "sid": "MTead89c72ced43a8cbe4ba67dfbd2f10", "name": "chat", "state": "ready"}], "revision": 2, "state": "connected", "recording": {"enabled": false, "revision": 1, "is_recording": false}, "subscribed": {"revision": 0, "tracks": []}, "published": {"revision": 1, "tracks": [{"kind": "data", "priority": "standard", "id": "3b4c8a26-9d94-468b-9d7c-5ba60216297d", "enabled": true, "name": "chat", "state": "ready"}]}, "type": "msg"}]}
2023-07-04T03:51:50.236Z debug [TwilioConnection #1: index.js:26063] wss://global.vss.twilio.com/signaling]
```

## Screenshot #4

### Data Tracks

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to the given Room with a LocalDataTrack.
 * @param {string} token - AccessToken for joining the Room
 * @returns {CancelablePromise<Room>}
 */
async function connectToRoomWithDataTrack(token, roomName) {
  const localDataTrack = new Video.LocalDataTrack({
    name: 'chat',
  });

  const room = await Video.connect(token, {
    name: roomName,
    tracks: [localDataTrack],
  });

  return room;
}

/**
 * Send a chat message using the given LocalDataTrack.
 * @param {LocalDataTrack} dataTrack - The @link LocalDataTrack
 * @param {string} message - The message to be sent
 */
function sendChatMessage(dataTrack, message) {
  dataTrack.send(message);
}

/**
 * Receive chat messages from RemoteParticipants in the Room
 * @param {Room} room - The Room you are currently in
 * @param {Function} onMessageReceived - Updates UI when a message is received
 */
function onMessageReceived(room, onMessageReceived) {
  room.on('message', (message) => {
    onMessageReceived(message);
  });
}
```

P1

[Disconnect from Room](#)

P2: has connected

P1: Message from P1

Message

[Submit](#)

P2

[Disconnect from Room](#)

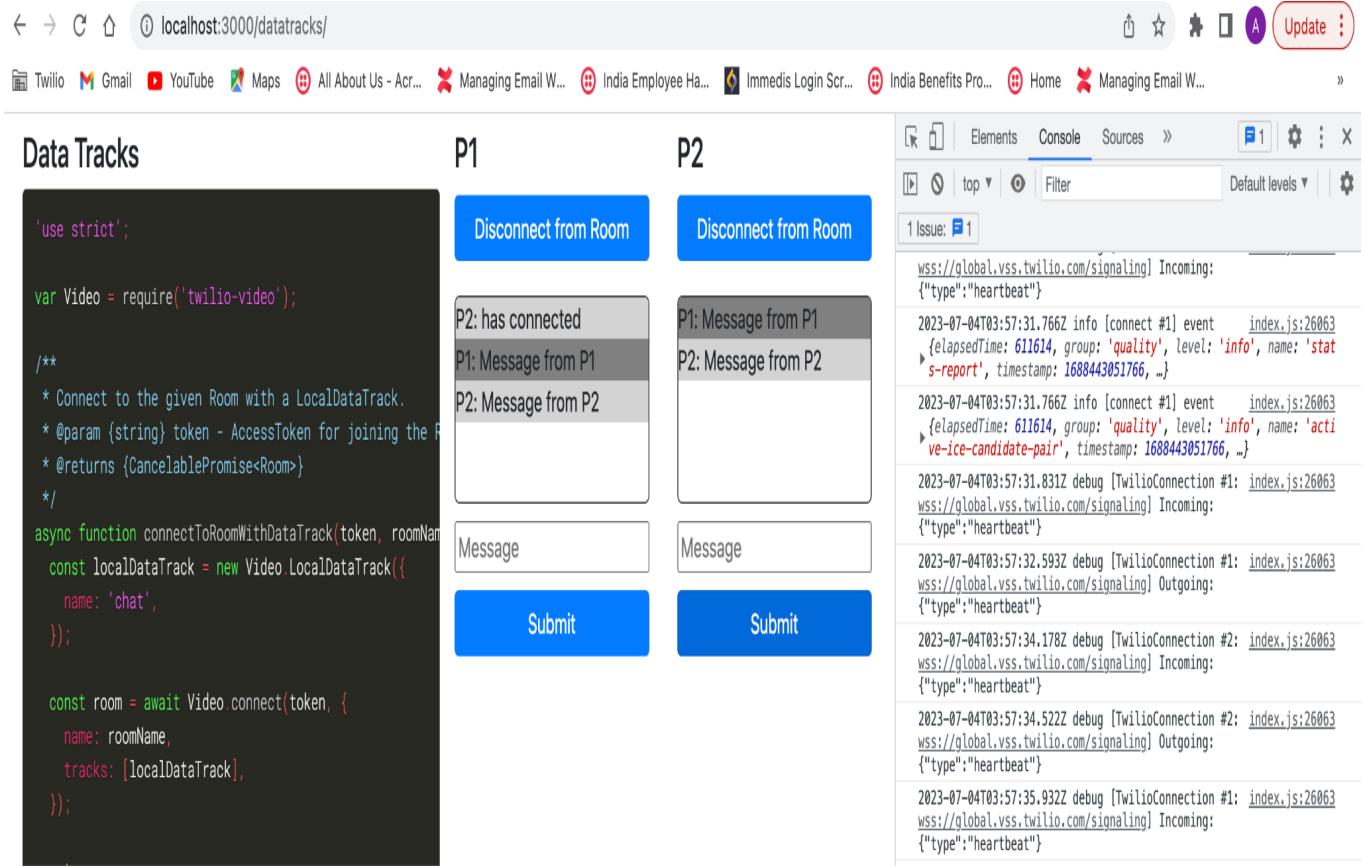
P1: Message from P1

Message

[Submit](#)

```
get ISTRACKENABLED: f()
get publishPriority: f()
get track: f()
[[Prototype]]: RemoteTrackPublication
2023-07-04T03:51:49.551Z info [RemoteParticipant #4: index.js:26063] PA82760c3d11460b56c9b0208cd0aee647] Added a new DataTrack: 204a0698-b76d-48ed-b459-a232d0ff9d2
2023-07-04T03:51:49.551Z debug [RemoteParticipant #4: index.js:26063] #4: PA82760c3d11460b56c9b0208cd0aee647] DataTrack: RemoteDataTrack {data: "chat", isEnabled: true, maxPacketLifeTime: null, ...}
2023-07-04T03:51:49.551Z debug [TwilioConnection #1: index.js:26063] wss://global.vss.twilio.com/signaling] Incoming: {"body": {"version": 2, "type": "update", "sid": "RM459f946dc150b91a8b8af364ae3be", "name": "RM459f946dc150b91a8b8af364ae3be", "participants": [{"sid": "PA1e786fe39e2f0de99af482e4dfdf88c8", "identity": "Mushy Anna Hanover", "tracks": [{"kind": "data", "priority": "standard", "id": "a4980cd7-de90-478e-b681-d023bf240be", "enabled": true, "sid": "MTead89c72ced43a8cbe4ba67dfbd2f10", "name": "chat", "state": "ready"}], "revision": 2, "state": "connected", "media_warnings": [], "participants": [{"sid": "PA82760c3d11460b56c9b0208cd0aee647", "identity": "Oaken Lando Davenport", "tracks": [], "revision": 2, "state": "connected", "media_warnings": [], "participants": [{"sid": "PA1e786fe39e2f0de99af482e4dfdf88c8", "identity": "Mushy Anna Hanover", "tracks": [{"kind": "data", "priority": "standard", "id": "a4980cd7-de90-478e-b681-d023bf240be", "enabled": true, "sid": "MTead89c72ced43a8cbe4ba67dfbd2f10", "name": "chat", "state": "ready"}], "revision": 2, "state": "connected", "recording": {"enabled": false, "revision": 1, "is_recording": false}, "subscribed": {"revision": 0, "tracks": []}, "published": {"revision": 1, "tracks": [{"kind": "data", "priority": "standard", "id": "3b4c8a26-9d94-468b-9d7c-5ba60216297d", "enabled": true, "name": "chat", "state": "ready"}]}}, "type": "msg"}]}
2023-07-04T03:51:49.551Z debug [TwilioConnection #1: index.js:26063] wss://global.vss.twilio.com/signaling]
```

## Screenshot #5



# Example- Bandwidth Constraints

## Screenshot #1

### Bandwidth Constraints

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to a Room with the given bandwidth constraints.
 * @param {string} token - Token for joining the Room
 * @param {string} roomName - Room name
 * @param {?number} maxAudioBitrate - Max audio bitrate (bps)
 * @param {?number} maxVideoBitrate - Max video bitrate (bps)
 * @returns {CancelablePromise<Room>}
 */
function connectWithBandwidthConstraints(token, roomName, maxAudioBitrate, maxVideoBitrate) {
    return Video.connect(token, {
        maxAudioBitrate: maxAudioBitrate,
        maxVideoBitrate: maxVideoBitrate,
        name: roomName
    });
}
```

### Participant's Media



#### Select Bandwidth Constraints

Max Audio Bitrate (bps)

16000

Max Video Bitrate (bps)

64000

Connect to Room

## Screenshot #2

### Bandwidth Constraints

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to a Room with the given bandwidth constraints.
 * @param {string} token - Token for joining the Room
 * @param {string} roomName - Room name
 * @param {?number} maxAudioBitrate - Max audio bitrate (bps)
 * @param {?number} maxVideoBitrate - Max video bitrate (bps)
 * @returns {CancelablePromise<Room>}
 */
function connectWithBandwidthConstraints(token, roomName, maxAudioBitrate, maxVideoBitrate) {
    return Video.connect(token, {
        maxAudioBitrate: maxAudioBitrate,
        maxVideoBitrate: maxVideoBitrate,
        name: roomName
    });
}

/**
 * Update the bandwidth constraints of a Room.
 * @param {Room} room - The Room whose bandwidth constraints have to be updated
 * @param {?number} maxAudioBitrate - Max audio bitrate (bps)
 * @param {?number} maxVideoBitrate - Max video bitrate (bps)
 */
function updateBandwidthConstraints(room, maxAudioBitrate, maxVideoBitrate) {
    room.localParticipant.setParameters({
        maxAudioBitrate: maxAudioBitrate,
        maxVideoBitrate: maxVideoBitrate
    });
}

exports.connectWithBandwidthConstraints = connectWithBandwidthConstraints;
exports.updateBandwidthConstraints = updateBandwidthConstraints;
```

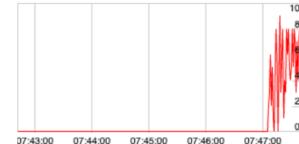
### Participant's Media



#### Select Bandwidth Constraints

Max Audio Bitrate (bps)

16000



Max Video Bitrate (bps)

64000



Disconnect from Room

## Screenshot #3

```

    * @param {string} roomName - Room name
    * @param {?number} maxAudioBitrate - Max audio bitrate (bps)
    * @param {?number} maxVideoBitrate - Max video bitrate (bps)
    * @returns {CancelablePromise<Room>}
    */
    function connectWithBandwidthConstraints(token, roomName, maxAudioBitrate, maxVideoBitrate) {
        return Video.connect(token, {
            maxAudioBitrate,
            maxVideoBitrate,
            name: roomName
        });
    }

    /**
     * Update the bandwidth constraints of a Room.
     * @param {Room} room - The Room whose bandwidth constraints have to be updated
     * @param {?number} maxAudioBitrate - Max audio bitrate (bps)
     * @param {?number} maxVideoBitrate - Max video bitrate (bps)
     */
    function updateBandwidthConstraints(room, maxAudioBitrate, maxVideoBitrate) {
        room.localParticipant.setParameters({
            maxAudioBitrate,
            maxVideoBitrate
        });
    }

```



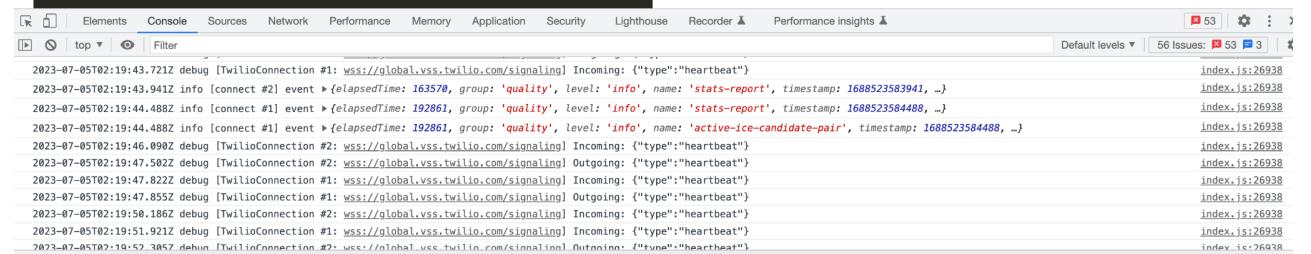
## Screenshot #4

```

    * @param {?number} maxVideoBitrate - Max video bitrate (bps)
    * @returns {CancelablePromise<Room>}
    */
    function connectWithBandwidthConstraints(token, roomName, maxAudioBitrate, maxVideoBitrate) {
        return Video.connect(token, {
            maxAudioBitrate,
            maxVideoBitrate,
            name: roomName
        });
    }

    /**
     * Update the bandwidth constraints of a Room.
     * @param {Room} room - The Room whose bandwidth constraints have to be updated
     * @param {?number} maxAudioBitrate - Max audio bitrate (bps)
     * @param {?number} maxVideoBitrate - Max video bitrate (bps)
     */
    function updateBandwidthConstraints(room, maxAudioBitrate, maxVideoBitrate) {
        room.localParticipant.setParameters({
            maxAudioBitrate,
            maxVideoBitrate
        });
    }

```



# Example- Media Device Selection

## Screenshot #1

```
function getDevicesOfKind(deviceInfos, kind) {
  return deviceInfos.filter(function(deviceInfo) {
    return deviceInfo.kind === kind;
  });
}

/**
 * Apply the selected audio output device.
 * @param {string} deviceId
 * @param {HTMLAudioElement} audio
 * @returns {Promise<void>}
 */
function applyAudioOutputDeviceSelection(deviceId, audio) {
  return typeof audio.setSinkId === 'function'
    ? audio.setSinkId(deviceId)
    : Promise.reject('This browser does not support setting the audio output device');
}

/**
 * Apply the selected input device.
 * @param {string} deviceId
 * @param {?LocalTrack} localTrack - LocalAudioTrack or LocalVideoTrack
 * @param {'audio' | 'video'} kind
 * @returns {Promise<LocalTrack>} - The created or restored track
 */
function applyInputDeviceSelection(deviceId, localTrack) {
  var constraints = { deviceId: { exact: deviceId } };
  if (localTrack) {
    return localTrack.restart(constraints).then(function() {
      return localTrack;
    });
  }
  return kind === 'audio'
    ? Video.createLocalAudioTrack(constraints)
    : Video.createLocalVideoTrack(constraints);
}

/**
 * Ensure that media permissions are obtained.
 * @returns {Promise<void>}

```



### Select Media Devices

Audio Input

Default - MacBook Pro Microphone (Built-in)

Apply

Video Input

FaceTime HD Camera

Apply

Audio Output

Default - MacBook Pro Speakers (Built-in)

Apply

Disconnect

### Join room:

RMccb5d07a9a9d4da32cab79a89f6d1642

Please join the above Room from another device/browser using [QuickStart App](#) to see media input device changes taking effect.

```
2023-07-05T02:44:48.849Z info [createLocalTracks #1] index.js:26292
Call to getUserMedia successful; got tracks: >Array(1)
2023-07-05T02:44:48.850Z debug [LocalAudioTrack #1: index.js:26292
undefined] Initializing
2023-07-05T02:44:48.850Z debug [LocalAudioTrack #1: index.js:26292
d99d6c8a-514f-4b26-8c24-d3fb3d4cc62c] defaultDeviceCaptureMode: auto
2023-07-05T02:44:48.850Z info [LocalAudioTrack #1: index.js:26292
d99d6c8a-514f-4b26-8c24-d3fb3d4cc62c] LocalAudioTrack will be
restarted if the default device changes
2023-07-05T02:44:48.851Z debug [LocalAudioTrack #1: index.js:26292
d99d6c8a-514f-4b26-8c24-d3fb3d4cc62c] Started
2023-07-05T02:44:52.760Z info [createLocalTracks #2] index.js:26292
Call to getUserMedia successful; got tracks: >Array(1)
2023-07-05T02:44:52.761Z debug [LocalVideoTrack #2: index.js:26292
undefined] Initializing
2023-07-05T02:44:52.761Z debug [LocalVideoTrack #2: index.js:26292
8cef7c3f-dc15-4882-af66-9be9df743091] Attempting to attach to
element:
<video id="videoinputpreview" autoplay playsinline></video>
2023-07-05T02:44:53.019Z debug [LocalVideoTrack #2: index.js:26292
8cef7c3f-dc15-4882-af66-9be9df743091] Dimensions: >Object
2023-07-05T02:44:53.019Z debug [LocalVideoTrack #2: index.js:26292
8cef7c3f-dc15-4882-af66-9be9df743091] Started
2023-07-05T02:45:06.278Z info [connect #1] index.js:26292
Connecting to a Room
2023-07-05T02:45:06.279Z debug [connect #1] Options: index.js:26292
> Object
2023-07-05T02:45:06.279Z info [connect #1] Getting index.js:26292
LocalTracks
2023-07-05T02:45:06.279Z debug [connect #1] Options: index.js:26292
> Object
2023-07-05T02:45:06.279Z info [createLocalTracks #3] index.js:26292
Adding user-provided LocalTracks
2023-07-05T02:45:06.279Z debug [createLocalTracks index.js:26292
#3] LocalTracks: >Array(2)
2023-07-05T02:45:06.279Z debug [connect #1] Creating index.js:26292
a new LocalParticipant; > LocalParticipantV2
2023-07-05T02:45:06.279Z info [LocalParticipant #1] index.js:26292
Created a new participant
2023-07-05T02:45:06.280Z info [LocalParticipant #1] index.js:26292
Added a new LocalVideoTrack: 8cef7c3f-dc15-4882-af66-9be9df743091
2023-07-05T02:45:06.280Z debug [LocalVideoTrack #2: index.js:26292
8cef7c3f-dc15-4882-af66-9be9df743091]
```

## Screenshot #2

```
/***
 * Get the list of available media devices of the given
 * @param {Array<MediaDeviceInfo>} deviceInfos
 * @param {string} kind - One of 'audioinput', 'audiooutput'
 * @returns {Array<MediaDeviceInfo>} - Only those media
 * devices which have the specified kind
 */
function getDevicesOfKind(deviceInfos, kind) {
  return deviceInfos.filter(function(deviceInfo) {
    return deviceInfo.kind === kind;
  });
}

/**
 * Apply the selected audio output device.
 * @param {string} deviceId
 * @param {HTMLAudioElement} audio
 * @returns {Promise<void>}
 */
function applyAudioOutputDeviceSelection(deviceId, audio) {
  return typeof audio.setSinkId === 'function'
    ? audio.setSinkId(deviceId)
    : Promise.reject('This browser does not support setting the audio output device');
}

/**
 * Apply the selected input device.
 * @param {string} deviceId
 * @param {?LocalTrack} localTrack - LocalAudioTrack or
 * @param {'audio' | 'video'} kind
 * @returns {Promise<LocalTrack>} - The created or restored
 * track
 */
function applyInputDeviceSelection(deviceId, localTrack) {
  var constraints = { deviceId: { exact: deviceId } };
  if (localTrack) {
    return localTrack.restart(constraints).then(function() {
      return localTrack;
    });
  }
  return kind === 'audio'
    ? Video.createLocalAudioTrack(constraints)
    : Video.createLocalVideoTrack(constraints);
}

/**
 * Ensure that media permissions are obtained.
 * @returns {Promise<void>}

```



### Select Media Devices

Audio Input

MacBook Pro Microphone (Built-in)

Apply

Video Input

FaceTime HD Camera

Apply

Audio Output

MacBook Pro Speakers (Built-in)

Apply

Disconnect

### Join room:

RMccb5d07a9a9d4da32cab79a89f6d1642

Please join the above Room from another device/browser using [QuickStart App](#) to see media input device changes taking effect.

```
enabled: true
id: "52ced964-2af9-4997-b8c8-95488e952866"
kind: "audio"
label: "MacBook Pro Microphone (Built-in)"
muted: false
oncapturechange: null
onended: null
onmute: null
onunmute: null
readyState: "live"
> [[Prototype]: MediaStreamTrack
2023-07-05T02:54:57.000Z debug [LocalAudioTrack #1: index.js:26292
d99d6c8a-514f-4b26-8c24-d3fb3d4cc62c] Initializing
2023-07-05T02:54:57.000Z debug [LocalAudioTrack #1: index.js:26292
d99d6c8a-514f-4b26-8c24-d3fb3d4cc62c] Started
2023-07-05T02:54:58.000Z info [connect #1] event index.js:26292
> {elapsedTime: 592530, group: 'quality', level: 'info', name: 'status-report', timestamp: 1688525698800, ...}
2023-07-05T02:54:58.000Z info [connect #1] event index.js:26292
> {elapsedTime: 592531, group: 'quality', level: 'info', name: 'active-ice-candidate-pair', timestamp: 1688525698809, ...}
2023-07-05T02:55:00.050Z debug [TwilioConnection #1: index.js:26292
wss://global.vss.twilio.com/signaling] Incoming:
{"type": "heartbeat"}
2023-07-05T02:55:00.245Z debug [TwilioConnection #1: index.js:26292
wss://global.vss.twilio.com/signaling] Outgoing:
{"type": "heartbeat"}
2023-07-05T02:55:04.150Z debug [TwilioConnection #1: index.js:26292
wss://global.vss.twilio.com/signaling] Incoming:
{"type": "heartbeat"}
2023-07-05T02:55:05.051Z debug [TwilioConnection #1: index.js:26292
wss://global.vss.twilio.com/signaling] Outgoing:
{"type": "heartbeat"}
2023-07-05T02:55:08.259Z debug [TwilioConnection #1: index.js:26292
wss://global.vss.twilio.com/signaling] Incoming:
{"type": "heartbeat"}
2023-07-05T02:55:08.611Z info [connect #1] event index.js:26292
> {elapsedTime: 602533, group: 'quality', level: 'info', name: 'status-report', timestamp: 1688525708811, ...}
2023-07-05T02:55:09.853Z debug [TwilioConnection #1: index.js:26292
wss://global.vss.twilio.com/signaling] Outgoing:
{"type": "heartbeat"}
```

## Screenshot #3

```
/**
 * Get the list of available media devices of the given
 * @param {Array<MediaDeviceInfo>} deviceInfos
 * @param {string} kind - One of 'audioinput', 'audiooutput'
 * @returns {Array<MediaDeviceInfo>} - Only those media
 */
function getDevicesOfKind(deviceInfos, kind) {
  return deviceInfos.filter(function(deviceInfo) {
    return deviceInfo.kind === kind;
  });
}

/**
 * Apply the selected audio output device.
 * @param {string} deviceId
 * @param {HTMLAudioElement} audio
 * @returns {Promise<void>}
 */
function applyAudioOutputDeviceSelection(deviceId, audio) {
  return typeof audio.setSinkId === 'function'
    ? audio.setSinkId(deviceId)
    : Promise.reject('This browser does not support setting the audio output device');
}

/**
 * Apply the selected input device.
 * @param {string} deviceId
 * @param {LocalTrack} localTrack - LocalAudioTrack or LocalVideoTrack
 * @param {'audio' | 'video'} kind
 * @returns {Promise<LocalTrack>} - The created or restored track
 */
function applyInputDeviceSelection(deviceId, localTrack) {
  var constraints = { deviceId: { exact: deviceId } };
  if (localTrack) {
    return localTrack.restart(constraints).then(function() {
      return localTrack;
    });
  }
  return kind === 'audio'
    ? localTrack.createAudioTrack()
    : localTrack.createVideoTrack();
}

function getDevicesOfKind(deviceInfos, kind) {
  return deviceInfos.filter(function(deviceInfo) {
    return deviceInfo.kind === kind;
  });
}

function applyDeviceSelection(deviceId, deviceType) {
  if (deviceType === 'audio') {
    applyAudioOutputDeviceSelection(deviceId);
  } else {
    applyInputDeviceSelection(deviceId);
  }
}

function handleDeviceChange() {
  var deviceType = document.querySelector('#device-type').value;
  applyDeviceSelection(deviceId, deviceType);
}

document.addEventListener('DOMContentLoaded', function() {
  var deviceType = document.querySelector('#device-type');
  deviceType.value = 'audio';
  deviceType.addEventListener('change', handleDeviceChange);
});
```



**Select Media Devices**

Audio Input

ZoomAudioDevice (Virtual)

Apply

Video Input

FaceTime HD Camera

Apply

Audio Output

ZoomAudioDevice (Virtual)

Apply

Disconnect

Join room:

RMccb5d07a9a9d4da32cab79a89f6d1642

Please join the above Room from another device/browser using [QuickStart App](#) to see media input device changes taking effect.

Elements Console Sources >

Default levels

39 Issues: 35 4

```
2023-07-05T02:55:18.592Z info [LocalAudioTrack #1: index.js:26292 d996c0a-514f-4b26-8c24-d3fb3d4cc62c] Re-acquiring the MediaStreamTrack
2023-07-05T02:55:18.592Z debug [LocalAudioTrack #1: index.js:26292 d996c0a-514f-4b26-8c24-d3fb3d4cc62c] Constraints: >{deviceId: {}}
2023-07-05T02:55:18.592Z info [LocalAudioTrack #1: index.js:26292 d996c0a-514f-4b26-8c24-d3fb3d4cc62c] Re-acquired the MediaStreamTrack
2023-07-05T02:55:18.648Z debug [LocalAudioTrack #1: index.js:26292 d996c0a-514f-4b26-8c24-d3fb3d4cc62c] MediaStreamTrack:
  MediaStreamTrack {kind: 'audio', id: '2adb50c0-c2de-4b55-8611-b6db51acf518', label: 'ZoomAudioDevice (Virtual)', enabled: true, mute: false, ...}
  contentHint: ''
  enabled: true
  id: '2adb50c0-c2de-4b55-8611-b6db51acf518'
  kind: 'audio'
  label: 'ZoomAudioDevice (Virtual)'
  muted: false
  oncapturechange: null
  onended: null
  onmute: null
  onunmute: null
  readyState: 'live'
  > [[Prototype]]: MediaStreamTrack
2023-07-05T02:55:18.648Z debug [LocalAudioTrack #1: index.js:26292 d996c0a-514f-4b26-8c24-d3fb3d4cc62c] Initializing
2023-07-05T02:55:18.649Z debug [LocalAudioTrack #1: index.js:26292 d996c0a-514f-4b26-8c24-d3fb3d4cc62c] Started
2023-07-05T02:55:18.810Z info [connect #1] event index.js:26292
  ▶ {elapsedTime: 612532, group: 'quality', level: 'info', name: 'stat s-report', timestamp: 1688525718810, ...}
2023-07-05T02:55:18.811Z info [connect #1] event index.js:26292
  ▶ {elapsedTime: 612533, group: 'quality', level: 'info', name: 'active-ice-candidate-pair', timestamp: 1688525718811, ...}
2023-07-05T02:55:19.466Z debug [TwilioConnection #1: index.js:26292 ws://global.vss.twilio.com/signaling] Outgoing:
  {"type": "heartbeat"}
2023-07-05T02:55:20.558Z debug [TwilioConnection #1: index.js:26292 ws://global.vss.twilio.com/signaling] Incoming:
  {"type": "heartbeat"}
2023-07-05T02:55:24.267Z debug [TwilioConnection #1: index.js:26292 ws://global.vss.twilio.com/signaling] Outgoing:
  {"type": "heartbeat"}
```

## Example - Codec Preferences

### Screenshot #1

#### Codec Preferences

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to a Room with the given codec preferences.
 * @param {string} token - Token for joining the Room
 * @param {string} roomName - Room name
 * @param {Array<AudioCodecName>} preferredAudioCodecs - Preferred audio codecs
 * @param {Array<VideoCodecName>} preferredVideoCodecs - Preferred video codecs
 * @returns {CancelablePromise<Room>}
 */
function connectWithPreferredCodecs(token, roomName, preferredAudioCodecs, preferredVideoCodecs) {
  return Video.connect(token, {
    preferredAudioCodecs: preferredAudioCodecs,
    preferredVideoCodecs: preferredVideoCodecs,
    name: roomName
  });
}
```

#### Participant's Media



#### Select Codec Preferences

Preferred Audio Codec

ISAC

Preferred Video Codec

H264

Connect to Room

## Screenshot #2

## Codec Preferences

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to a Room with the given codec preferences.
 * @param {string} token - Token for joining the Room
 * @param {string} roomName - Room name
 * @param {Array<AudioCodecName>} preferredAudioCodecs
 * @param {Array<VideoCodecName>} preferredVideoCodecs
 * @returns {CancelablePromise<Room>}
 */
function connectWithPreferredCodecs(token, roomName, preferredAudioCodecs, preferredVideoCodecs) {
    return Video.connect(token, {
        preferredAudioCodecs: preferredAudioCodecs,
        preferredVideoCodecs: preferredVideoCodecs,
        name: roomName
    });
}

exports.connectWithPreferredCodecs = connectWithPreferredCodecs;
```

## Participant's Media



### Select Codec Preferences

Preferred Audio Codec

ISAC

Selected Audio Codec: opus

Preferred Video Codec

H264

Selected Video Codec: H264

Disconnect from Room

## Console Output

38 Issues: 35 3

```
2023-07-05T02:56:42.249Z debug [connect #1] Creating index.js:26182
new RoomSignaling
2023-07-05T02:56:42.250Z info [connect #1] event index.js:26182
▶ Object
2023-07-05T02:56:42.251Z debug [TwilioConnection #1: index.js:26182
ws://global.vss.twilio.com/signaling] Created a new WebSocket:
▶ WebSocket
2023-07-05T02:56:42.251Z info [connect #1] event index.js:26182
▶ Object
2023-07-05T02:56:42.396Z debug [TwilioConnection #1: index.js:26182
ws://global.vss.twilio.com/signaling] WebSocket opened: ▶ WebSocket
2023-07-05T02:56:42.396Z info [connect #1] event index.js:26182
▶ Object
2023-07-05T02:56:42.396Z debug [TwilioConnection #1: index.js:26182
ws://global.vss.twilio.com/signaling] Outgoing: {"id": "d744148-8fe1-4aa0-871f-7ff092b73f86", "immediate": true, "type": "hello", "version": 2, "body": "edge", "roaming": false, "token": "eyJhbGciOiJIUzI1NiIsInRcClxKpVCIiS0NeSI6InR3awXpbymGE7Dj0xIn0.yeqJqGk10jTS2UWV1WmV1ZwXlWzYzIj22aNKzJ1NWEwLTe20dgMjU4MDf1LjCjnFudhM1OnsiJwRhRpdh1k10JwAxZpCBK2W5vdpmExUvhbnR0p281LCj2awhLbyf6e1YQ00jPj20Eg0MjU4MDisVm4cC16MTY40D0uMD1iMwiaXNzIjoiUlNTVLTzXWfMjFnz2Ylm1WFm2I1NmzJZGy3NTVHMcIiN1Y161fKDM2R2BzJ0xZDM57y2NjgjMDhNDY3ZDM3y2AFz2Q2mEif0.YrK_Uj945XvYrLztBjbghfG6-JF39srFvSwS3jk", "type": "ice", "version": 1}
2023-07-05T02:56:42.437Z debug [TwilioConnection #1: index.js:26182
ws://global.vss.twilio.com/signaling] Incoming: {"negotiatedTimeout": 5000, "type": "welcome"}
2023-07-05T02:56:42.437Z info [connect #1] event index.js:26182
▶ Object
2023-07-05T02:56:43.606Z debug [TwilioConnection #1: index.js:26182
ws://global.vss.twilio.com/signaling] Incoming: {"body": {}, "version": 1, "type": "iced", "participants": [], "ice_servers": [{"url": "turn:turn.balabit.turn.twilio.com:34787", "transport": "udp", "username": "601aeeb067d755588984924a44204e246f38687707c1f14492d67d012", "credential": "1BGX6cNaYgCFavWhr1PM9GQ19tNUExyDPSNg6f4fq="}, {"url": "turns:turns.balabit.turn.twilio.com:443", "transport": "tcp", "username": "601aeeb067d755588984924a44204e246f38687707c1f14492d67d012", "credential": "+1BGX6cNaYgCFavWhr1PM9GQ19tNUExyDPSNg6f4fq="}], "type": "msg"}
2023-07-05T02:56:43.606Z debug [connect #1] Got ICE index.js:26182
servers: ▶ Array(2)
2023-07-05T02:56:43.620Z debug [connect #1] createAndOfferer() succeeded.
2023-07-05T02:56:43.620Z debug [connect #1] Got ICE index.js:26182
servers: ▶ Array(2)
2023-07-05T02:56:43.620Z debug [TwilioConnection #1: index.js:26182
ws://global.vss.twilio.com/signaling] Got ICE index.js:26182
servers: ▶ Array(2)
```

## Screenshot #3

## Codec Preferences

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to a Room with the given codec preferences.
 * @param {string} token - Token for joining the Room
 * @param {string} roomName - Room name
 * @param {Array<AudioCodecName>} preferredAudioCodecs
 * @param {Array<VideoCodecName>} preferredVideoCodecs
 * @returns {CancelablePromise<Room>}
 */
function connectWithPreferredCodecs(token, roomName, preferredCodecs) {
    return Video.connect(token, {
        preferredAudioCodecs: preferredAudioCodecs,
        preferredVideoCodecs: preferredVideoCodecs,
        name: roomName
    });
}

exports.connectWithPreferredCodecs = connectWithPreferredCodecs;
```

## Participant's Media



## Select Codec Preferences

Preferred Audio Codec

Opus

Selected Audio Codec: opus

Preferred Video Codec

VP9

Selected Video Codec: H264

[Disconnect from Room](#)

Elements Console Sources > A 1 35 Default levels

38 Issues: 35 3

```
've-ice-candidate-pair', timestamp: 1688525985503, ...}
2023-07-05T02:59:46.519Z debug [TwilioConnection #1: index.js:26182
wss://global.vss.twilio.com/signaling] Incoming:
{"type": "heartbeat"}}

2023-07-05T02:59:48.060Z debug [TwilioConnection #2: index.js:26182
wss://global.vss.twilio.com/signaling] Incoming:
{"type": "heartbeat"}}

2023-07-05T02:59:50.001Z debug [TwilioConnection #1: index.js:26182
wss://global.vss.twilio.com/signaling] Outgoing:
{"type": "heartbeat"}}

2023-07-05T02:59:50.223Z debug [TwilioConnection #2: index.js:26182
wss://global.vss.twilio.com/signaling] Outgoing:
{"type": "heartbeat"}}

2023-07-05T02:59:50.619Z debug [TwilioConnection #1: index.js:26182
wss://global.vss.twilio.com/signaling] Incoming:
{"type": "heartbeat"}}

2023-07-05T02:59:52.161Z debug [TwilioConnection #2: index.js:26182
wss://global.vss.twilio.com/signaling] Incoming:
{"type": "heartbeat"}}

2023-07-05T02:59:54.719Z debug [TwilioConnection #1: index.js:26182
wss://global.vss.twilio.com/signaling] Incoming:
{"type": "heartbeat"}}

2023-07-05T02:59:54.801Z debug [TwilioConnection #1: index.js:26182
wss://global.vss.twilio.com/signaling] Outgoing:
{"type": "heartbeat"}}

2023-07-05T02:59:55.002Z debug [TwilioConnection #2: index.js:26182
wss://global.vss.twilio.com/signaling] Outgoing:
{"type": "heartbeat"}}

2023-07-05T02:59:55.199Z info [connect #1 event index.js:26182
elapsedTime: 192951, group: 'quality', level: 'info', name: 'stat
s-report', timestamp: 1688525995199, ...}

2023-07-05T02:59:55.199Z info [connect #1 event index.js:26182
elapsedTime: 192951, group: 'quality', level: 'info', name: 'acti
ve-ice-candidate-pair', timestamp: 1688525995199, ...}

2023-07-05T02:59:55.502Z info [connect #2 event index.js:26182
elapsedTime: 144636, group: 'quality', level: 'info', name: 'stat
s-report', timestamp: 1688525995502, ...}

2023-07-05T02:59:56.260Z debug [TwilioConnection #2: index.js:26182
wss://global.vss.twilio.com/signaling] Incoming:
{"type": "heartbeat"}}

2023-07-05T02:59:58.819Z debug [TwilioConnection #1: index.js:26182
wss://global.vss.twilio.com/signaling] Incoming:
{"type": "heartbeat"}]
```

## Screenshot #4

The screenshot shows a browser window with three main sections:

- Codec Preferences:** A code editor displaying JavaScript code for connecting to a room with preferred codecs.
- Participant's Media:** A video preview of a participant and a "Select Codec Preferences" section with dropdown menus for Preferred Audio Codec (PCMU) and Preferred Video Codec (VP9), both set to their selected values.
- Developer Console:** Shows a log of Twilio connection events, including debug logs for heartbeats and quality reports.

## Example - Share Your Screen

### Screenshot #1

#### Share Your Screen

```
'use strict';

const Video = require('twilio-video');

/**
 * Create a LocalVideoTrack for your screen. You can then share it
 * with other Participants in the Room.
 * @param {number} height - Desired vertical resolution in pixels
 * @param {number} width - Desired horizontal resolution in pixels
 * @returns {Promise<LocalVideoTrack>}
 */
function createScreenTrack(height, width) {
  if (typeof navigator === 'undefined'
    || !navigator.mediaDevices
    || !navigator.mediaDevices.getDisplayMedia) {
    return Promise.reject(new Error('getDisplayMedia is not supported'));
  }
  return navigator.mediaDevices.getDisplayMedia({
    video: {
      height: height,
      width: width
    }
  }).then(function(stream) {
    return new Video.LocalVideoTrack(stream.getVideoTracks()[0]);
  });
}

exports.createScreenTrack = createScreenTrack;
```

#### Local Screen



#### Remote Screen



## Screenshot #2

The screenshot shows a browser window sharing the local screen. At the top, there's a sharing status bar with "Sharing https://www.twilio.com to http://localhost:3000" and "Stop sharing". To the right is a link to "View tab: www.twilio.com". Below this, the main content area has two sections: "Share Your Screen" on the left and "Local Screen" on the right.

**Share Your Screen:**

```
'use strict';

const Video = require('twilio-video');

/**
 * Create a LocalVideoTrack for your screen. You can then share it
 * with other Participants in the Room.
 * @param {number} height - Desired vertical resolution in pixels
 * @param {number} width - Desired horizontal resolution in pixels
 * @returns {Promise<LocalVideoTrack>}
 */
function createScreenTrack(height, width) {
  if (typeof navigator === 'undefined'
    || !navigator.mediaDevices
    || !navigator.mediaDevices.getDisplayMedia) {
    return Promise.reject(new Error('getDisplayMedia is not supported'));
  }
  return navigator.mediaDevices.getDisplayMedia({
    video: {
      height,
      width
    }
  }).then(function(stream) {
    return new Video.LocalVideoTrack(stream.getVideoTracks()[0]);
  });
}

exports.createScreenTrack = createScreenTrack;
```

**Local Screen:**

This section displays the local video feed from the browser. It includes a "Stop Screen Capture" button at the bottom.

## Screenshot #3

The screenshot shows a browser window sharing the local screen, similar to Screenshot #2. It includes the "Share Your Screen" and "Local Screen" sections. Additionally, it features several control panels and developer tools.

**Share Your Screen:**

```
'use strict';

const Video = require('twilio-video');

/**
 * Create a LocalVideoTrack for your screen. You can then share it
 * with other Participants in the Room.
 * @param {number} height - Desired vertical resolution in pixels
 * @param {number} width - Desired horizontal resolution in pixels
 * @returns {Promise<LocalVideoTrack>}
 */
function createScreenTrack(height, width) {
  if (typeof navigator === 'undefined'
    || !navigator.mediaDevices
    || !navigator.mediaDevices.getDisplayMedia) {
    return Promise.reject(new Error('getDisplayMedia is not supported'));
  }
  return navigator.mediaDevices.getDisplayMedia({
    video: {
      height,
      width
    }
  }).then(function(stream) {
    return new Video.LocalVideoTrack(stream.getVideoTracks()[0]);
  });
}

exports.createScreenTrack = createScreenTrack;
```

**Local Screen:**

This section displays the local video feed. It includes a "Stop Screen Capture" button at the bottom. To the right, there are several control panels:

- Bandwidth Constraints:** Describes the Bandwidth Constraints API.
- Local Video Filter:** Describes the Local Video Filter API.
- Media Device Selection:** Describes Media Device Selection.
- Local Video Snapshot:** Describes Local Video Snapshot.
- Codec Preferences:** Describes Codec Preferences.
- Dominant Speaker:** Describes Dominant Speaker.
- Reconnection States and Events:** Describes Reconnection States and Events.
- Network Quality:** Describes Network Quality.
- Enabling and Disabling Tracks:** Describes Enabling and Disabling Tracks.

**Remote Screen:**

This section displays the remote video feed. It includes a "Stop Screen Capture" button at the bottom. To the right, there are several control panels:

- Bandwidth Constraints:** Describes the Bandwidth Constraints API.
- Local Video Filter:** Describes the Local Video Filter API.
- Media Device Selection:** Describes Media Device Selection.
- Local Video Snapshot:** Describes Local Video Snapshot.
- Codec Preferences:** Describes Codec Preferences.
- Share Your Screen:** Describes Share Your Screen.
- Reconnection States and Events:** Describes Reconnection States and Events.
- Network Quality:** Describes Network Quality.
- Enabling and Disabling Tracks:** Describes Enabling and Disabling Tracks.

**Developer Tools:**

To the far right, the browser's developer tools are visible, showing 36 issues. The tabs include Elements, Console, Sources, and a large JavaScript preview pane.

## Screenshot #4

Sharing http://localhost:3000 to http://localhost:3000 [Stop sharing](#) [View tab: localhost:3000](#)

### Share Your Screen

```
'use strict';

const Video = require('twilio-video');

/**
 * Create a LocalVideoTrack for your screen. You can do this
 * with other Participants in the Room.
 * @param {number} height - Desired vertical resolution
 * @param {number} width - Desired horizontal resolution
 * @returns {Promise<LocalVideoTrack>}
 */
function createScreenTrack(height, width) {
  if (typeof navigator === 'undefined')
    || !navigator.mediaDevices
    || !navigator.mediaDevices.getDisplayMedia)
    return Promise.reject(new Error('getDisplayMedia is not supported'));
  return navigator.mediaDevices.getDisplayMedia({
    video: {
      height: height,
      width: width
    }
  }).then(function(stream) {
    return new Video.LocalVideoTrack(stream.getVideoTracks());
  });
}

exports.createScreenTrack = createScreenTrack;
```

### Local Screen

- [Bandwidth Constraints](#)
- [Local Video Filter](#)
- [Media Device Selection](#)
- [Share Your Screen](#)
- [Reconnection States and Events](#)
- [Enabling and Disabling Tracks](#)

### Stop Screen Capture

### Remote Screen

- [Bandwidth Constraints](#)
- [Local Video Filter](#)
- [Media Device Selection](#)
- [Share Your Screen](#)
- [Reconnection States and Events](#)
- [Enabling and Disabling Tracks](#)

36 Issues: [35](#) [1](#)

## Example - Dominant Speaker Detection Screenshot #1

### Dominant Speaker Detection

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to a Room with the Dominant Speaker API enabled
 * This API is available only in Small Group or Group Rooms
 * @param {string} token - Token for joining the Room
 * @returns {CancelablePromise<Room>}
 */
function connectToRoomWithDominantSpeaker(token) {
  return Video.connect(token, {
    dominantSpeaker: true
  });
}

/**
 * Listen to changes in the dominant speaker and update
 * @param {Room} room - The Room you just joined
 * @param {function} updateDominantSpeaker - Updates the room
 * @returns {void}
 */
function setupDominantSpeakerUpdates(room, updateDominantSpeaker) {
  room.on('dominantSpeakerChanged', function(participant) {
    console.log('A new RemoteParticipant is now the dominant speaker');
    updateDominantSpeaker(participant);
  });
}

exports.connectToRoomWithDominantSpeaker = connectToRoomWithDominantSpeaker;
exports.setupDominantSpeakerUpdates = setupDominantSpeakerUpdates;
```

### Join room:

RM5eff778472ba5a10e9d6cf7e56b4421d9

Please join the above Room using the [QuickStart App](#) or connect the below Participants to the Room and unmute only one of them.

Alice	<a href="#">Disconnect</a>	<a href="#">Mute</a>
Bob	<a href="#">Disconnect</a>	<a href="#">Mute</a>
Charlie	<a href="#">Disconnect</a>	<a href="#">Mute</a>
Mak	<a href="#">Disconnect</a>	<a href="#">Mute</a>

Alice

[Intervention] Slow network is detected. See <https://www.chromestatus.com/feature/5636954674692896> for more details. Fallback font will be used while loading: [https://fonts.gstatic.com/s/robotomono/v22/L0xuDFxLMF-BfRBDXInJHh45mvgwGEPt0\\_gPa\\_R0WA4J18SJ0t.woff2](https://fonts.gstatic.com/s/robotomono/v22/L0xuDFxLMF-BfRBDXInJHh45mvgwGEPt0_gPa_R0WA4J18SJ0t.woff2)

A new RemoteParticipant is now the dominant speaker:  
▶ `RemoteParticipant {audioTracks: Map(1), dataTracks: Map(0), ...}`

## Screenshot #2

### Dominant Speaker Detection

```
'use strict';

var Video = require('twilio-video');

/** 
 * Connect to a Room with the Dominant Speaker API enabled
 * This API is available only in Small Group or Group Rooms
 * @param {string} token - Token for joining the Room
 * @returns {CancelablePromise<Room>}
 */
function connectToRoomWithDominantSpeaker(token) {
  return Video.connect(token, {
    dominantSpeaker: true
  });
}

/** 
 * Listen to changes in the dominant speaker and update
 * @param {Room} room - The Room you just joined
 * @param {function} updateDominantSpeaker - Updates the Room
 * @returns {void}
 */
function setupDominantSpeakerUpdates(room, updateDominantSpeaker) {
  room.on('dominantSpeakerChanged', function(participant) {
    console.log('A new RemoteParticipant is now the dominant speaker');
    updateDominantSpeaker(participant);
  });
}

exports.connectToRoomWithDominantSpeaker = connectToRoomWithDominantSpeaker;
exports.setupDominantSpeakerUpdates = setupDominantSpeakerUpdates;
```

### Join room:

RM663b89d0b22a5307597f4083b80c3800

Please join the above Room using the [QuickStart App](#) or connect the below Participants to the Room and unmute only one of them.



```
[Intervention] Slow network is detected. See https://www.chrome.com/status/com/feature/5636954674692096 for more details.Fallback font will be used while loading: https://fonts.gstatic.com/s/robotomono/v2/1/0xuF4xVMP-BTRQdAM1nJhg5mwGFr10_P0_R0W4A185j0Lw0ff2
2023-07-15T07:02:35.126Z info [connect #1] Connecting to a index.js:25995
Room
2023-07-15T07:02:35.126Z debug [connect #1] Options: > Object index.js:25995
2023-07-15T07:02:35.126Z info [connect #1] localTracks were index.js:25995 not provided, so they will be acquired automatically before connecting to the Room. LocalTracks will be released if connecting to the Room fails or if the Room is disconnected
2023-07-15T07:02:37.052Z info [createLocalTracks #1] Call to index.js:25995 getUserMedia successful; got tracks: > Array(2)
2023-07-15T07:02:37.052Z debug [LocalAudioTrack #1: undefined] Initializing
2023-07-15T07:02:37.052Z debug [LocalVideoTrack #1: 9a49758b- index.js:25995 d5d1-4c13-8569-f978d8f400c9] defaultDeviceCaptureMode: auto
2023-07-15T07:02:37.052Z info [LocalAudioTrack #1: 9a49758b- index.js:25995 d5d1-4c13-8569-f978d8f400c9] LocalAudioTrack will be restarted if the default device changes
2023-07-15T07:02:37.052Z debug [LocalVideoTrack #2: index.js:25995 undefined] Initializing
2023-07-15T07:02:37.052Z debug [connect #1] Creating a new index.js:25995 LocalParticipant: > LocalParticipantV2
2023-07-15T07:02:37.054Z info [LocalParticipant #1] Created a index.js:25995 new Participant
2023-07-15T07:02:37.054Z info [LocalParticipant #1] Added a index.js:25995 new LocalAudioTrack: 9a49758b-d5d1-4c13-8569-f978d8f400c9
2023-07-15T07:02:37.054Z debug [LocalParticipant #1] LocalAudioTrack: > LocalAudioTrack
2023-07-15T07:02:37.054Z info [LocalParticipant #1] Added a index.js:25995 new LocalVideoTrack: c93eef04-ce72-4556-abf3-fed22722ebbc
2023-07-15T07:02:37.054Z debug [LocalParticipant #1] LocalVideoTrack
2023-07-15T07:02:37.054Z debug [connect #1] Creating a new index.js:25995 RoomSignalaling
2023-07-15T07:02:37.054Z info [connect #1] mount & Object index.js:25995
2023-07-15T07:02:37.054Z info [connect #1] mount & Object index.js:25995
```

## Screenshot #3

### Dominant Speaker Detection

```
'use strict';

var Video = require('twilio-video');

/** 
 * Connect to a Room with the Dominant Speaker API enabled
 * This API is available only in Small Group or Group Rooms
 * @param {string} token - Token for joining the Room
 * @returns {CancelablePromise<Room>}
 */
function connectToRoomWithDominantSpeaker(token) {
  return Video.connect(token, {
    dominantSpeaker: true
  });
}

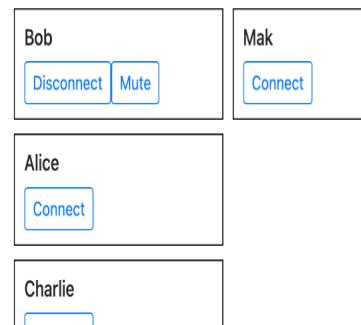
/** 
 * Listen to changes in the dominant speaker and update
 * @param {Room} room - The Room you just joined
 * @param {function} updateDominantSpeaker - Updates the Room
 * @returns {void}
 */
function setupDominantSpeakerUpdates(room, updateDominantSpeaker) {
  room.on('dominantSpeakerChanged', function(participant) {
    console.log('A new RemoteParticipant is now the dominant speaker');
    updateDominantSpeaker(participant);
  });
}

exports.connectToRoomWithDominantSpeaker = connectToRoomWithDominantSpeaker;
exports.setupDominantSpeakerUpdates = setupDominantSpeakerUpdates;
```

### Join room:

RM663b89d0b22a5307597f4083b80c3800

Please join the above Room using the [QuickStart App](#) or connect the below Participants to the Room and unmute only one of them.



```
obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"} index.js:26
A new RemoteParticipant is now the dominant speaker: index.js:26
> RemoteParticipant {audioTracks: Map(1), dataTracks: Map(0), ...}
2023-07-15T07:03:16.937Z debug [TwilioConnection #2: ws://gl index.js:25995 obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
2023-07-15T07:03:18.040Z debug [TwilioConnection #1: ws://gl index.js:25995 obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
2023-07-15T07:03:18.221Z debug [TwilioConnection #2: ws://gl index.js:25995 obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
2023-07-15T07:03:18.729Z info [connect #1] event index.js:25995
  > {elapsedTime: 43604, group: 'quality', level: 'info', name: 'stats-report', timestamp: 1689404598729, ...}
2023-07-15T07:03:19.236Z debug [TwilioConnection #1: ws://gl index.js:25995 obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
2023-07-15T07:03:21.036Z debug [TwilioConnection #2: ws://gl index.js:25995 obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
2023-07-15T07:03:22.846Z debug [TwilioConnection #1: ws://gl index.js:25995 obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
2023-07-15T07:03:23.024Z debug [TwilioConnection #2: ws://gl index.js:25995 obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
2023-07-15T07:03:23.203Z info [connect #2] event index.js:25995
  > {elapsedTime: 11299, group: 'quality', level: 'info', name: 'stats-report', timestamp: 1689404683203, ...}
2023-07-15T07:03:23.204Z info [connect #2] event index.js:25995
  > {elapsedTime: 11299, group: 'quality', level: 'info', name: 'active-ice-canidate-pair', timestamp: 1689404683203, ...}
2023-07-15T07:03:23.336Z debug [TwilioConnection #1: ws://gl index.js:25995 obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
2023-07-15T07:03:25.135Z debug [TwilioConnection #2: ws://gl index.js:25995 obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
2023-07-15T07:03:27.437Z debug [TwilioConnection #1: ws://gl index.js:25995 obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
2023-07-15T07:03:27.656Z debug [TwilioConnection #1: ws://gl index.js:25995 obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
2023-07-15T07:03:27.827Z debug [TwilioConnection #2: ws://gl index.js:25995 obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
2023-07-15T07:03:28.728Z info [connect #1] event index.js:25995
  > {elapsedTime: 53603, group: 'quality', level: 'info', name: 'stats-report', timestamp: 1689404608728, ...}
```

# Example - Reconnection States and Events

## Screenshot #1

The screenshot shows a web browser window with the URL `localhost:3001/reconnection/`. The page title is "Reconnection States and Events". On the left, there is a code editor containing JavaScript code for handling room reconnection events. On the right, there is a "Room state is:" section with three buttons: "Connected" (white background), "Disconnected" (red background), and "Reconnecting" (light gray background). Below these buttons is a "Create Room" button. A note below the buttons says: "After you have created the Room, please either turn off your internet network for a little while and then turn it back on, or switch between networks. The Twilio Video SDK will try to re-establish connection to the Room, which will transition to the 'connecting' state. Once reconnection is complete, the Room will transition back to the 'connected' state. You can join other user to the room using button below." At the bottom is a "Join Room" button. The browser's developer tools are open at the top right, showing a warning about a slow network connection.

```
'use strict';

/**
 * Listen to Room reconnection events and update the UI
 * @param {Room} room - The Room you have joined
 * @param {function} updateRoomState - Updates the app
 * @returns {void}
 */
function setupReconnectionUpdates(room, updateRoomState) {
  room.on('disconnected', (room, error) => {
    if (error.code === 20104) {
      console.log('Signaling reconnection failed due to');
    } else if (error.code === 53000) {
      console.log('Signaling reconnection attempts exhausted');
    } else if (error.code === 53204) {
      console.log('Signaling reconnection took too long');
    }
    updateRoomState(room.state);
  });

  room.on('reconnected', function() {
    console.log('Reconnected to the Room!');
    updateRoomState(room.state);
  });

  room.on('reconnecting', function(error) {
    if (error.code === 53001) {
      console.log('Reconnecting your signaling connection');
    } else if (error.code === 53405) {
      console.log('Reconnecting your media connection!');
    }
    updateRoomState(room.state);
  });
}

exports.setupReconnectionUpdates = setupReconnectionUpdates;
```

## Screenshot #2

The screenshot shows a web browser window with the URL `localhost:3001/reconnection/`. The page title is "Reconnection States and Events". On the left, there is a code editor containing the same JavaScript code as in Screenshot #1. On the right, there is a "Room state is:" section with three buttons: "Connected" (green background), "Disconnected" (white background), and "Reconnecting" (light gray background). Below these buttons is a "Create Room" button. A note below the buttons says: "After you have created the Room, please either turn off your internet network for a little while and then turn it back on, or switch between networks. The Twilio Video SDK will try to re-establish connection to the Room, which will transition to the 'connecting' state. Once reconnection is complete, the Room will transition back to the 'connected' state. You can join other user to the room using button below." At the bottom is a "Join Room" button. The browser's developer tools are open at the top right, showing a log of console messages related to the connection process.

```
'use strict';

/**
 * Listen to Room reconnection events and update the UI
 * @param {Room} room - The Room you have joined
 * @param {function} updateRoomState - Updates the app
 * @returns {void}
 */
function setupReconnectionUpdates(room, updateRoomState) {
  room.on('disconnected', (room, error) => {
    if (error.code === 20104) {
      console.log('Signaling reconnection failed due to');
    } else if (error.code === 53000) {
      console.log('Signaling reconnection attempts exhausted');
    } else if (error.code === 53204) {
      console.log('Signaling reconnection took too long');
    }
    updateRoomState(room.state);
  });

  room.on('reconnected', function() {
    console.log('Reconnected to the Room!');
    updateRoomState(room.state);
  });

  room.on('reconnecting', function(error) {
    if (error.code === 53001) {
      console.log('Reconnecting your signaling connection');
    } else if (error.code === 53405) {
      console.log('Reconnecting your media connection!');
    }
    updateRoomState(room.state);
  });
}

exports.setupReconnectionUpdates = setupReconnectionUpdates;
```

Console Log (partial):

```
2023-07-15T07:11:52.577Z info [connect #1] Connecting to a index.js:25948 Room
2023-07-15T07:11:52.577Z debug [connect #1] Options: index.js:25948
  {eventObserver: EventObserver, wsServer: 'wss://global.vss.twilio.com/signaling', automaticSubscription: true, dominantSpeaker: false, enableDscp: false, ...}
2023-07-15T07:11:52.577Z info [connect #1] Getting index.js:25948 LocalTracks
2023-07-15T07:11:52.577Z debug [connect #1] Options: index.js:25948
  {eventObserver: EventObserver, wsServer: 'wss://global.vss.twilio.com/signaling', automaticSubscription: true, dominantSpeaker: false, enableDscp: false, ...}
2023-07-15T07:11:52.578Z info [createLocalTracks #1] Adding index.js:25948 user-provided LocalTracks
2023-07-15T07:11:52.578Z debug [createLocalTracks #1] index.js:25948 LocalTracks: []
2023-07-15T07:11:52.578Z debug [connect #1] Creating a new index.js:25948 LocalParticipant:
  LocalParticipantV2 {_events: {}, _eventsCount: 1, _maxListeners: undefined, ...}
2023-07-15T07:11:52.579Z info [LocalParticipant #1] Created a index.js:25948 new Participant
2023-07-15T07:11:52.579Z debug [connect #1] Creating a new index.js:25948 RoomSignaling
2023-07-15T07:11:52.580Z info [connect #1] event index.js:25948
  {elapsedTime: 4, group: 'signaling', level: 'info', name: 'early', timestamp: 1689405112500}
2023-07-15T07:11:52.580Z debug [TwilioConnection #1: wss://gl index.js:25948 global.vss.twilio.com/signaling] index.js:25948 created a new WebSocket:
  WebSocket {url: 'wss://global.vss.twilio.com/signaling', readyState: 0, bufferedAmount: 0, onopen: null, onerror: null, ...}
2023-07-15T07:11:52.581Z info [connect #1] event index.js:25948
  {elapsedTime: 5, group: 'room', level: 'info', name: 'connect', timestamp: 1689405112581, ...}
2023-07-15T07:11:53.591Z debug [TwilioConnection #1: wss://gl index.js:25948 global.vss.twilio.com/signaling] index.js:25948 WebSocket opened:
  WebSocket {url: 'wss://global.vss.twilio.com/signaling', readyState: 1, bufferedAmount: 0, onopen: null, onerror: null, ...}
2023-07-15T07:11:53.592Z info [connect #1] event index.js:25948
  {elapsedTime: 1016, group: 'signaling', level: 'info', name: 'connecting', timestamp: 1689405113592}
2023-07-15T07:11:53.593Z debug [TwilioConnection #1: wss://gl index.js:25948 global.vss.twilio.com/signaling] Outgoing: {"id": "b45ba963-9cd8-4060-9d64-"}
```

## Screenshot #3

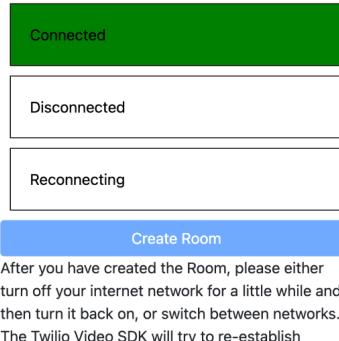
```
'use strict';

/**
 * Listen to Room reconnection events and update the UI
 * @param {Room} room - The Room you have joined
 * @param {function} updateRoomState - Updates the app
 * @returns {void}
 */
function setupReconnectionUpdates(room, updateRoomState) {
    room.on('disconnected', (room, error) => {
        if (error.code === 20104) {
            console.log('Signaling reconnection failed due to');
        } else if (error.code === 53000) {
            console.log('Signaling reconnection attempts exha');
        } else if (error.code === 53204) {
            console.log('Signaling reconnection took too long');
        }
        updateRoomState(room.state);
    });

    room.on('reconnected', function() {
        console.log('Reconnected to the Room!');
        updateRoomState(room.state);
    });
}

room.on('reconnecting', function(error) {
    if (error.code === 53001) {
        console.log('Reconnecting your signaling connecti');
    } else if (error.code === 53405) {
        console.log('Reconnecting your media connection!');
    }
    updateRoomState(room.state);
});

exports.setupReconnectionUpdates = setupReconnectionUpd
```



After you have created the Room, please either turn off your internet network for a little while and then turn it back on, or switch between networks. The Twilio Video SDK will try to re-establish connection to the Room, which will transition to the "connecting" state. Once reconnection is complete, the Room will transition back to the "connected" state. You can join other user to the room using button below.

Leave Room



# Example - Network Quality

## Screenshot #1

### Network Quality

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to a Room with the Network Quality API enabled
 * This API is available only in Small Group or Group Room
 * @param {string} token - Token for joining the Room
 * @param {number} localVerbosity - Verbosity level of
 *   for the LocalParticipant [1 - 3]
 * @param {number} remoteVerbosity - Verbosity level of
 *   for the RemoteParticipants [0 - 3]
 * @returns {CancelablePromise<Room>}
 */
function connectToRoomWithNetworkQuality(token, localVerbosity) {
  networkQuality: {
    local: localVerbosity,
    remote: remoteVerbosity
  }
}

/**
 * Listen to changes in the Network Quality report of a
 * your application.
 * @param {Participant} participant - The Participant who
 * @param {function} updateNetworkQualityReport - Updates
 *   Network Quality report of the Participant.
 * @returns {void}
 */
function setupNetworkQualityUpdatesForParticipant(participant) {
  participant.on('networkQualityLevelChanged', function() {
    updateNetworkQualityReport(participant);
  });
}

/**
 * Listen to changes in the Network Quality reports and
 * @param {Room} room - The Room you just joined
 */

```

### Set Network Quality Verbosity Levels

LocalParticipant

1

RemoteParticipant(s)

1

Leave Room

**Join Room:**

RM63f60513b20daea793e710a77b65bed5

After creating the Room with the desired verbosity levels, please join the above Room using the [QuickStart App](#) or connect the below Participants to the Room. You can also change the verbosity levels and observe the change in the verbosity of the reported Network Quality stats.

Charlie Connect

Alice Connect

Mak Connect

Bob Connect

NQ Level (You): 1

```
2023-07-15T07:19:26.687Z info [MediaSignaling #1:network_quality] Tearing down index.js:26091
2023-07-15T07:19:33.364Z info [connect #2] Connecting to a Room index.js:26091
2023-07-15T07:19:33.364Z debug [connect #2] Options: {eventObserver: EventObserver, wsServer: 'ws://global.vss.twilio.com/signaling', automaticSubscription: true, dominantSpeaker: false, enableDscp: false} index.js:26091
2023-07-15T07:19:33.364Z info [connect #2] LocalTracks were not provided, so they will be acquired automatically before connecting to the Room. LocalTracks will be released if connecting to the Room fails or if the Room is disconnected index.js:26091
2023-07-15T07:19:35.286Z info [createLocalTracks #2] Call to undefined Initializing index.js:26091
2023-07-15T07:19:35.286Z debug [LocalAudioTrack #3: d56b0d1a-302a-492e-9dbb-bd73ab716f4] defaultDeviceCaptureMode: auto index.js:26091
2023-07-15T07:19:35.287Z info [LocalAudioTrack #3: d56b0d1a-302a-492e-9dbb-bd73ab716f4] LocalAudioTrack will be restarted if the default device changes index.js:26091
2023-07-15T07:19:35.287Z debug [LocalVideoTrack #4: undefined] Initializing index.js:26091
2023-07-15T07:19:35.287Z debug [connect #2] Creating a new LocalParticipant: > LocalParticipantV2 {_events: {}, _eventsCount: 1, _maxListeners: undefined, _d: {}} index.js:26091
2023-07-15T07:19:35.287Z info [LocalParticipant #2] Created a new Participant index.js:26091
2023-07-15T07:19:35.287Z info [LocalParticipant #2] Added a new LocalAudioTrack: d56b0d1a-302a-492e-9dbb-bd73ab716f4 index.js:26091
2023-07-15T07:19:35.287Z debug [LocalParticipant #2] LocalAudioTrack: LocalAudioTrack {kind: 'audio', name: 'd56b0d1a-302a-492e-9dbb-bd73ab716f4', processedTrack: null, _d: {}} index.js:26091
2023-07-15T07:19:35.287Z info [LocalParticipant #2] Added a new LocalVideoTrack: 5d140f01-4030-4eb6-85d4-6ee2d527ae62 index.js:26091
2023-07-15T07:19:35.288Z debug [LocalParticipant #2] LocalVideoTrack: LocalVideoTrack {kind: 'video', name: '5d140f01-4030-4eb6-85d4-6ee2d527ae62', processedTrack: null, _d: {}} index.js:26091
2023-07-15T07:19:35.288Z debug [connect #2] Creating a new RoomSignalizing index.js:26091
2023-07-15T07:19:35.288Z info [connect #2] event {elapsdTime: 1924, group: 'signaling', level: 'info', name: 'early', times: 1, timestamp: 1689405375288} index.js:26091
2023-07-15T07:19:35.288Z info [connect #2] event {elapsdTime: 1924, group: 'signaling', level: 'info', name: 'early', times: 1, timestamp: 1689405375288} index.js:26091
```

## Screenshot #2

localhost:3001/networkquality/

### Network Quality

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to a Room with the Network Quality API enabled
 * This API is available only in Small Group or Group Room
 * @param {string} token - Token for joining the Room
 * @param {number} localVerbosity - Verbosity level of
 *   for the LocalParticipant [1 - 3]
 * @param {number} remoteVerbosity - Verbosity level of
 *   for the RemoteParticipant(s) [0 - 3]
 * @returns {CancelablePromise<Room>}
 */
function connectToRoomWithNetworkQuality(token, localVerbosity) {
  networkQuality: {
    local: localVerbosity,
    remote: remoteVerbosity
  }
}

/**
 * Listen to changes in the Network Quality report of a
 * your application.
 * @param {Participant} participant - The Participant who
 * @param {function} updateNetworkQualityReport - Updates
 *   Network Quality report of the Participant.
 * @returns {void}
 */
function setupNetworkQualityUpdatesForParticipant(participant) {
  participant.on('networkQualityLevelChanged', function() {
    updateNetworkQualityReport(participant);
  });
}

/**
 * Listen to changes in the Network Quality reports and
 * @param {Room} room - The Room you just joined
 */

```

### Set Network Quality Verbosity Levels

LocalParticipant

1

RemoteParticipant(s)

1

Leave Room

**Join Room:**

RM63f60513b20daea793e710a77b65bed5

After creating the Room with the desired verbosity levels, please join the above Room using the [QuickStart App](#) or connect the below Participants to the Room. You can also change the verbosity levels and observe the change in the verbosity of the reported Network Quality stats.

Charlie Disconnect

Alice Disconnect

Mak Connect

Bob Connect

```
2023-07-15T07:20:33.132Z debug [TwilioConnection #2: wss://gl index.js:26091
obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"} index.js:26091
2023-07-15T07:20:33.339Z debug [TwilioConnection #4: wss://gl index.js:26091
obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"} index.js:26091
2023-07-15T07:20:33.481Z debug [TwilioConnection #2: wss://gl index.js:26091
obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"} index.js:26091
2023-07-15T07:20:33.493Z info [connect #4] event {elapsdTime: 11382, group: 'quality', level: 'info', name: 'stats-report', timestamp: 168940533493, _d: {}} index.js:26091
2023-07-15T07:20:33.493Z info [connect #4] event {elapsdTime: 11382, group: 'quality', level: 'info', name: 'active-ice-can-didate-pair', timestamp: 168940533493, _d: {}} index.js:26091
2023-07-15T07:20:33.918Z debug [MediaSignaling #7:network_quality] Incoming: > {local: {}, remotes: Array(2), type: 'network_quality'} index.js:26091
2023-07-15T07:20:35.152Z debug [MediaSignaling #7:network_quality] Incoming: > {local: {}, remotes: Array(2), type: 'network_quality'} index.js:26091
2023-07-15T07:20:35.171Z info [connect #3] event {elapsdTime: 21622, group: 'quality', level: 'info', name: 'stats-report', timestamp: 168940535171, _d: {}} index.js:26091
2023-07-15T07:20:35.199Z debug [TwilioConnection #2: wss://gl index.js:26091
obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"} index.js:26091
2023-07-15T07:20:35.468Z debug [TwilioConnection #4: wss://gl index.js:26091
obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"} index.js:26091
2023-07-15T07:20:36.398Z debug [MediaSignaling #7:network_quality] Incoming: > {local: {}, remotes: Array(2), type: 'network_quality'} index.js:26091
2023-07-15T07:20:36.692Z info [connect #2] event {elapsdTime: 63228, group: 'quality', level: 'info', name: 'stats-report', timestamp: 168940536692, _d: {}} index.js:26091
2023-07-15T07:20:37.583Z debug [TwilioConnection #2: wss://gl index.js:26091
obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"} index.js:26091
2023-07-15T07:20:37.935Z debug [TwilioConnection #2: wss://gl index.js:26091
obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"} index.js:26091
2023-07-15T07:20:37.935Z debug [TwilioConnection #2: wss://gl index.js:26091
obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"} index.js:26091
2023-07-15T07:20:38.144Z debug [TwilioConnection #4: wss://gl index.js:26091
obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"} index.js:26091
2023-07-15T07:20:38.879Z debug [MediaSignaling #7:network_quality] Incoming: > {local: {}, remotes: Array(2), type: 'network_quality'} index.js:26091
```

## Screenshot #3

The screenshot shows a browser window with two main sections: a code editor on the left and a user interface on the right.

**Code Editor (Left):**

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to a Room with the Network Quality API enabled
 * This API is available only in Small Group or Group Rooms
 * @param {string} token - Token for joining the Room
 * @param {number} localVerbosity - Verbosity level of
 *   for the LocalParticipant [1 - 3]
 * @param {number} remoteVerbosity - Verbosity level of
 *   for the RemoteParticipant(s) [0 - 3]
 * @returns {CancelablePromise<Room>}
 */
function connectToRoomWithNetworkQuality(token, localVerbosity, remoteVerbosity) {
    return Video.connect(token, {
        networkQuality: {
            local: localVerbosity,
            remote: remoteVerbosity
        }
    });
}

/**
 * Listen to changes in the Network Quality report of a
 * your application.
 * @param {Participant} participant - The Participant who
 *   @param {function} updateNetworkQualityReport - Update
 *     Network Quality report of the Participant.
 *   @returns {void}
 */
function setupNetworkQualityUpdatesForParticipant(participant) {
    updateNetworkQualityReport(participant);
    participant.on('networkQualityLevelChanged', function() {
        updateNetworkQualityReport(participant);
    });
}

/**
 * Listen to changes in the Network Quality reports and

```

**User Interface (Right):**

**Set Network Quality Verbosity Levels**

LocalParticipant:

RemoteParticipant(s):

**Leave Room**

**Join Room:**

Room ID: RM63f60513b20daea793e710a77b65bed5

After creating the Room with the desired verbosity levels, please join the above Room using the [QuickStart App](#) or connect the below Participants to the Room. You can also change the verbosity levels and observe the change in the verbosity of the reported Network Quality stats.

Participants:

- Charlie [Disconnect](#)
- Alice [Disconnect](#)
- Mak [Disconnect](#)
- Bob [Disconnect](#)

**Console (Top Right):**

Default levels ▾ 9 Issues: 9

```
2023-07-15T07:21:02.339Z debug [TwilioConnection #4: wss://ql_index.js:26091
obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}  
2023-07-15T07:21:00.182Z debug [TwilioConnection #6: wss://ql_index.js:26091
obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}  
2023-07-15T07:21:01.103Z debug [MediaSignaling
#7:network_quality] Incoming: index.js:26091  
  > {local: {}, remotes: Array(4), type: 'network_quality'}  
2023-07-15T07:21:02.156Z info [connect #5] event index.js:26091
  > {elapsedTime: 21392, group: 'quality', level: 'info', name: 'stats-report',
  timestamp: 1689405662156, ...}  
2023-07-15T07:21:02.184Z debug [TwilioConnection #2: wss://ql_index.js:26091
obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}  
2023-07-15T07:21:02.258Z debug [TwilioConnection #5: wss://ql_index.js:26091
obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}  
2023-07-15T07:21:02.341Z debug [MediaSignaling
#7:network_quality] Incoming: index.js:26091  
  > {local: {}, remotes: Array(4), type: 'network_quality'}  
2023-07-15T07:21:03.326Z debug [TwilioConnection #2: wss://ql_index.js:26091
obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}  
2023-07-15T07:21:03.384Z debug [TwilioConnection #4: wss://ql_index.js:26091
obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}  
2023-07-15T07:21:03.384Z debug [TwilioConnection #3: wss://ql_index.js:26091
obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}  
2023-07-15T07:21:03.384Z debug [TwilioConnection #5: wss://ql_index.js:26091
obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}  
2023-07-15T07:21:03.490Z info [connect #4] event index.js:26091
  > {elapsedTime: 41380, group: 'quality', level: 'info', name: 'stats-report',
  timestamp: 1689405663490, ...}  
2023-07-15T07:21:03.545Z debug [TwilioConnection #6: wss://ql_index.js:26091
obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}  
2023-07-15T07:21:03.576Z debug [MediaSignaling
#7:network_quality] Incoming: index.js:26091  
  > {local: {}, remotes: Array(4), type: 'network_quality'}  
2023-07-15T07:21:03.682Z debug [TwilioConnection #3: wss://ql_index.js:26091
obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}  
2023-07-15T07:21:04.054Z debug [TwilioConnection #4: wss://ql_index.js:26091
obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}  
2023-07-15T07:21:04.099Z info [connect #6] event index.js:26091
  > {elapsedTime: 21315, group: 'quality', level: 'info', name: 'stats-report',
  timestamp: 1689405664099, ...}  
2023-07-15T07:21:04.282Z debug [TwilioConnection #6: wss://ql_index.js:26091
obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}  
2023-07-15T07:21:04.815Z debug [MediaSignaling
#7:network_quality] Incoming: index.js:26091  
  > {local: {}, remotes: Array(4), type: 'network_quality'}
```

# Example - Enabling and Disabling Tracks

## Screenshot #1

localhost:3001/localmediacontrols/

### Enabling and Disabling Tracks

```
'use strict';

/**
 * Mute/unmute your media in a Room.
 * @param {Room} room - The Room you have joined
 * @param {'audio'|'video'} kind - The type of media you want to mute/unmute
 * @param {'mute'|'unmute'} action - Whether you want to mute or unmute
 */

function muteOrUnmuteYourMedia(room, kind, action) {
  const publications = kind === 'audio'
    ? room.localParticipant.audioTracks
    : room.localParticipant.videoTracks;

  publications.forEach(function(publication) {
    if (action === 'mute') {
      publication.track.disable();
    } else {
      publication.track.enable();
    }
  });
}

/**
 * Mute your audio in a Room.
 * @param {Room} room - The Room you have joined
 * @returns {void}
 */
function muteYourAudio(room) {
  muteOrUnmuteYourMedia(room, 'audio', 'mute');
}

/**
 * Mute your video in a Room.
 * @param {Room} room - The Room you have joined
 * @returns {void}
 */
function muteYourVideo(room) {
  muteOrUnmuteYourMedia(room, 'video', 'mute');
}
```

### Local Media Controls

Disable Audio
Disable Video

### RemoteParticipant View

Elements
Console
Sources
Network
Default levels
1 Issue
Update

```
2023-07-15T07:23:02.690Z info [connect #1] Connecting to a Room
2023-07-15T07:23:02.691Z debug [connect #1] Options: index.js:25972
{eventObserver: EventObserver, wsServer: 'wss://global.vss.twilio.com/signaling', automaticSubscription: true, dominantSpeaker: false, enableDscp: false, ...}
2023-07-15T07:23:02.691Z info [connect #1] LocalTracks were index.js:25972 not provided, so they will be acquired automatically before connecting to the Room. LocalTracks will be released if connecting to the Room fails or if the Room is disconnected
2023-07-15T07:23:02.788Z info [createLocalTracks #1] Call to index.js:25972 getUserMedia successful; got tracks:
  ▶ (2) [MediaStreamTrack, MediaStreamTrack]
2023-07-15T07:23:02.789Z debug [LocalAudioTrack #1: undefined] Initializing
2023-07-15T07:23:02.789Z debug [LocalAudioTrack #1: aa942a8c-7f99-4c56-9687-c4f972da4f9e] defaultDeviceCaptureMode: auto
2023-07-15T07:23:02.789Z info [LocalAudioTrack #1: aa942a8c- index.js:25972 7f99-4c56-9687-c4f972da4f9e] LocalAudioTrack will be restarted if the default device changes
2023-07-15T07:23:02.789Z debug [LocalVideoTrack #2: undefined] Initializing
2023-07-15T07:23:02.790Z debug [connect #1] Creating a new LocalParticipant:
  ▶ LocalParticipantV2 {_events: {}, _eventsCount: 1, _maxListeners: undefined, ...}
2023-07-15T07:23:02.790Z info [LocalParticipant #1] Created a index.js:25972 new Participant
2023-07-15T07:23:02.791Z info [LocalParticipant #1] Added a index.js:25972 new LocalAudioTrack: aa942a8c-7f99-4c56-9687-c4f972da4f9e
2023-07-15T07:23:02.791Z debug [LocalParticipant #1] LocalAudioTrack:
  ▶ LocalAudioTrack {kind: 'audio', name: 'aa942a8c-7f99-4c56-9687-c4f972da4f9e', processedTrack: null, ...}
2023-07-15T07:23:02.791Z info [LocalParticipant #1] Added a index.js:25972 new LocalVideoTrack: 35874920-1302-4d07-b555-5ela33d09b5e
2023-07-15T07:23:02.791Z debug [LocalParticipant #1] LocalVideoTrack:
  ▶ LocalVideoTrack {kind: 'video', name: '35874920-1302-4d07-b555-5ela33d09b5b', processedTrack: null, ...}
2023-07-15T07:23:02.791Z debug [connect #1] Creating a new index.js:25972 RoomSignaling
2023-07-15T07:23:02.792Z info [connect #1] event
  ▶ {elapsedTime: 102, group: 'signaling', level: 'info', name: 'early', timestamp: 1690405302702}

```

## Screenshot #2

localhost:3001/localmediacontrols/

### Enabling and Disabling Tracks

```
'use strict';

/**
 * Mute/unmute your media in a Room.
 * @param {Room} room - The Room you have joined
 * @param {'audio'|'video'} kind - The type of media you want to mute/unmute
 * @param {'mute'|'unmute'} action - Whether you want to mute or unmute
 */

function muteOrUnmuteYourMedia(room, kind, action) {
  const publications = kind === 'audio'
    ? room.localParticipant.audioTracks
    : room.localParticipant.videoTracks;

  publications.forEach(function(publication) {
    if (action === 'mute') {
      publication.track.disable();
    } else {
      publication.track.enable();
    }
  });
}

/**
 * Mute your audio in a Room.
 * @param {Room} room - The Room you have joined
 * @returns {void}
 */
function muteYourAudio(room) {
  muteOrUnmuteYourMedia(room, 'audio', 'mute');
}

/**
 * Mute your video in a Room.
 * @param {Room} room - The Room you have joined
 * @returns {void}
 */
function muteYourVideo(room) {
  muteOrUnmuteYourMedia(room, 'video', 'mute');
}
```

### Local Media Controls

Disable Audio
Enable Video

### RemoteParticipant View

Elements
Console
Sources
Network
Default levels
1 Issue
Update

```
2023-07-15T07:23:32.450Z debug [TwilioConnection #1: wss://gl.global.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
2023-07-15T07:23:33.857Z debug [TwilioConnection #2: wss://gl.global.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
2023-07-15T07:23:34.206Z info [connect #1] event
  ▶ {elapsedTime: 31516, group: 'quality', level: 'info', name: 'stats-report', timestamp: 1699405814206, ...}
2023-07-15T07:23:34.206Z info [connect #1] event
  ▶ {elapsedTime: 31516, group: 'quality', level: 'info', name: 'active-ice-can-didate-pair', timestamp: 1699405814206, ...}
2023-07-15T07:23:34.383Z debug [TwilioConnection #1: wss://gl.global.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
2023-07-15T07:23:34.550Z debug [TwilioConnection #2: wss://gl.global.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
2023-07-15T07:23:35.486Z info [connect #2] event
  ▶ {elapsedTime: 31283, group: 'quality', level: 'info', name: 'stats-report', timestamp: 1699405814206, ...}
2023-07-15T07:23:35.486Z info [connect #2] event
  ▶ {elapsedTime: 31283, group: 'quality', level: 'info', name: 'active-ice-can-didate-pair', timestamp: 1699405814206, ...}
2023-07-15T07:23:36.453Z debug [TwilioConnection #1: wss://gl.global.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
2023-07-15T07:23:37.950Z debug [TwilioConnection #2: wss://gl.global.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
2023-07-15T07:23:39.191Z debug [TwilioConnection #1: wss://gl.global.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
2023-07-15T07:23:39.358Z debug [TwilioConnection #2: wss://gl.global.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
2023-07-15T07:23:40.553Z debug [TwilioConnection #1: wss://gl.global.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
2023-07-15T07:23:42.068Z debug [TwilioConnection #2: wss://gl.global.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
2023-07-15T07:23:43.997Z debug [TwilioConnection #1: wss://gl.global.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
```

## Screenshot #3

← → ⏪ i localhost:3001/localmediacontrols/

## Enabling and Disabling Tracks

```
'use strict';

/** 
 * Mute/unmute your media in a Room.
 * @param {Room} room - The Room you have joined
 * @param {'audio'|'video'} kind - The type of media you want to mute/unmute
 * @param {'mute'|'unmute'} action - Whether you want to mute or unmute
 */
function muteOrUnmuteYourMedia(room, kind, action) {
  const publications = kind === 'audio'
    ? room.localParticipant.audioTracks
    : room.localParticipant.videoTracks;

  publications.forEach(function(publication) {
    if (action === 'mute') {
      publication.track.disable();
    } else {
      publication.track.enable();
    }
  });
}

/** 
 * Mute your audio in a Room.
 * @param {Room} room - The Room you have joined
 * @returns {void}
 */
function muteYourAudio(room) {
  muteOrUnmuteYourMedia(room, 'audio', 'mute');
}

/** 
 * Mute your video in a Room.
 * @param {Room} room - The Room you have joined
 * @returns {void}
 */
```

## Local Media Controls

Enable Audio

Enable Video

## RemoteParticipant View



2023-07-15T07:23:47.596Z debug [RemoteAudioTrack #4: index.js:25972 MT9ab3417285f76e954ccb951eb8d1] Attempting to detach from elements: > [audio]

2023-07-15T07:23:47.597Z debug [TwilioConnection #1: wss://gl\_index.js:25972 obal.vss.twilio.com/signaling] Incoming: {"body": "msg"}

{"version":2,"type":"update","sid":"RM3977eb89d7d8d35cdebfdf1ad9259aff9","name":"RM3977eb89d7d8d35cdebfdf1ad9259aff9","participant": {"sid":"PA27fe99dca4e4493d321df1d4fb99","identity":"Dapper Tammy Gunights","tracks": [{"kind": "video", "priority": "standard", "id": "0d5fe574-8dca-444f-9e24-6735e061e27", "enabled": false}, {"kind": "audio", "priority": "standard", "id": "64457de0-83b5-4313-d2b8-d0ef5fa93e2e", "enabled": false}, {"kind": "audio", "priority": "standard", "id": "MT9ab3417285f76e954ccb951eb8d1", "name": "aa942ac8-7f99-4c56-9867-c4972d49fe", "state": "ready"}], "revision": 5, "state": "connected", "media\_warnings": []}, "participants": [{"sid": "PA41964bb11bf3604e9ff2cdcc54aa", "identity": "Yawning Penny Raleigh", "tracks": [], "revision": 1, "state": "connected"}, {"revision": 0, "tracks": []}, {"published": "revision": 3, "tracks": [{"kind": "video", "priority": "standard", "id": "0d5fe574-8dca-444f-9e24-6735e061e27", "enabled": false}, {"kind": "audio", "priority": "standard", "id": "64457de0-83b5-4313-d2b8-d0ef5fa93e2e", "enabled": false}, {"kind": "audio", "priority": "standard", "id": "MT9ab3417285f76e954ccb951eb8d1", "name": "aa942ac8-7f99-4c56-9867-c4972d49fe", "state": "ready"}]}, {"type": "msg"}

2023-07-15T07:23:48.752Z debug [TwilioConnection #1: wss://gl\_index.js:25972 obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}

2023-07-15T07:23:48.971Z debug [TwilioConnection #2: wss://gl\_index.js:25972 obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}

2023-07-15T07:23:50.251Z debug [TwilioConnection #2: wss://gl\_index.js:25972 obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}

2023-07-15T07:23:52.134Z debug [TwilioConnection #1: wss://gl\_index.js:25972 obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}

2023-07-15T07:23:52.850Z debug [TwilioConnection #1: wss://gl\_index.js:25972 obal.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}

2023-07-15T07:23:53.785Z debug [TwilioConnection #2: wss://gl\_index.js:25972 obal.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}

2023-07-15T07:23:54.212Z info [connect #1] event index.js:25972 > [elapsedTime: 51522, group: "quality", level: "info", name: "stats-report", timestamp: 1689405834212, -]

# Example - Remote Participant Reconnection States

## Screenshot #1

The screenshot shows a browser window with the URL `localhost:3001/remotereconnection/`. The page has two main sections: "Remote Participant Reconnection States" and "Local View".

**Remote Participant Reconnection States:** This section contains a code editor with the following JavaScript code:

```
'use strict';

/**
 * Listen to RemoteParticipant reconnection events and
 * @param {Room} room - The Room you have joined
 * @param {function} updateRoomState - Updates the app
 * @returns {void}
 */
function handleRemoteParticipantReconnectionUpdates(room) {
    room.on('participantReconnecting', function(participant) {
        updateParticipantState(participant.state);
    });

    room.on('participantReconnected', function(participant) {
        updateParticipantState(participant.state);
    });
}

/**
 * Listen to LocalParticipant reconnection events and
 * @param {Room} room - The Room you have joined
 * @param {function} updateRoomState - Updates the app
 * @returns {void}
 */
function handleLocalParticipantReconnectionUpdates(room) {
    const localParticipant = room.localParticipant;

    localParticipant.on('reconnecting', function() {
        updateParticipantState(localParticipant.state);
    });

    localParticipant.on('reconnected', function() {
        updateParticipantState(localParticipant.state);
    });
}

exports.handleLocalParticipantReconnectionUpdates = handleLocalParticipantReconnectionUpdates;
exports.handleRemoteParticipantReconnectionUpdates = handleRemoteParticipantReconnectionUpdates;
```

**Local View:** This section shows a video feed of a person with glasses and a beard. Below the video is a blue button labeled "Simulate Reconnection".

**State:** A green box labeled "Connected".

**Remote View:** This section shows the same video feed from a different perspective.

**Console:** The right side of the browser window shows a log of Twilio connection events. Some key entries include:

- 2023-07-15T10:41:49.630Z info [LocalParticipant #1: index.js:25921 PAB6c89bc0fe7a14ec074835143c553c9a] reconnected
- 2023-07-15T10:41:49.630Z info [Room #1: index.js:25921 RM6148b019231da0fa689478fa12229b1] Transitioned to state: connected
- 2023-07-15T10:41:49.846Z debug [TwilioConnection #2: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
- 2023-07-15T10:41:50.952Z debug [TwilioConnection #2: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
- 2023-07-15T10:41:51.665Z debug [TwilioConnection #3: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
- 2023-07-15T10:41:52.418Z debug [TwilioConnection #3: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
- 2023-07-15T10:41:53.451Z info [connect #2] event index.js:25921 {elapsedTime: 74871, group: 'quality', level: 'info', name: 'stats-report', timestamp: 1689417713451, ...}
- 2023-07-15T10:41:53.451Z info [connect #2] event index.js:25921 {elapsedTime: 74871, group: 'quality', level: 'info', name: 'active-ice-can-didate-r', timestamp: 1689417713451, ...}
- 2023-07-15T10:41:53.947Z debug [TwilioConnection #2: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
- 2023-07-15T10:41:55.759Z debug [TwilioConnection #2: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
- 2023-07-15T10:41:55.765Z debug [TwilioConnection #3: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
- 2023-07-15T10:41:57.226Z debug [TwilioConnection #3: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
- 2023-07-15T10:41:57.939Z debug [TwilioConnection #2: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
- 2023-07-15T10:41:58.586Z info [connect #1] event index.js:25921 {elapsedTime: 84619, group: 'quality', level: 'info', name: 'stats-report', timestamp: 1689417718586, ...}
- 2023-07-15T10:41:59.771Z debug [TwilioConnection #3: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
- 2023-07-15T10:42:00.568Z debug [TwilioConnection #2: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
- 2023-07-15T10:42:02.031Z debug [TwilioConnection #3: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Outgoing: {"type": "heartbeat"}
- 2023-07-15T10:42:02.047Z debug [TwilioConnection #2: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}
- 2023-07-15T10:42:02.451Z info [connect #2] event index.js:25921 {elapsedTime: 84871, group: 'quality', level: 'info', name: 'stats-report', timestamp: 1689417723451, ...}
- 2023-07-15T10:42:03.865Z debug [TwilioConnection #3: wss://au\_index.js:25921 l.vss.twilio.com/signaling] Incoming: {"type": "heartbeat"}

# Example - Video Track Manual Controls

## Screenshot #1

The screenshot shows a browser window with the URL `localhost:3001/manualrenderhint/`. The page has three main sections: "Video Track Manual Controls", "Remote Video Controls", and "Remote Video Track".

**Video Track Manual Controls:** This section contains a code editor with the following JavaScript code:

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to a Room with 'manual' mode. The default mode is 'auto'.
 * @param {string} token - AccessToken for joining the room
 * @returns {Room}
 */
function joinRoom(token) {
    return Video.connect(token, {
        name: 'my-cool-room',
        bandwidthProfile: {
            video: {
                contentPreferencesMode: 'manual',
                clientTrackSwitchOffControl: 'manual'
            }
        }
    });
}

/**
 * Switch on the RemoteVideoTrack.
 * @param {RemoteVideoTrack} track - The RemoteVideoTrack
 * @returns {RemoteVideoTrack}
 */
function switchOn(track) {
    return track.switchOn();
}

/**
 * Switch off the RemoteVideoTrack.
 * @param {RemoteVideoTrack} track - The RemoteVideoTrack
 * @returns {RemoteVideoTrack}
 */
function switchOff(track) {
    return track.switchOff();
}
```

**Remote Video Controls:** This section includes a "Switch Off Control" button (disabled), a "Switch On" button, and a "Switch Off" button. It also shows a dropdown for "Content Preferences" set to "640x480".

**Remote Video Track:** This section shows a video feed of a person with glasses and a beard. A "On" button is visible above the video.

**Video Bitrate:** This section shows a graph of video bitrate over time. The x-axis represents time from 16:09:00 to 16:13:00. The y-axis represents bitrate in kbps, with markers at 0.0K, 0.5K, and 1.0K. The graph shows a sharp peak at approximately 1.0K around 16:12:30.

**Console:** The right side of the browser window shows a log of intervention errors. Some key entries include:

- [Intervention] Slow network is detected. See <https://www.chromestatus.com/feature/5636954674692086> for more details. Fallback font will be used while loading: <https://fonts.gstatic.com/s/robotomono/v2/1.0xuDF4xLVMFBFRBdMM1hHq45mwGxFI0.gPq.R0W4AJ185J0t.woff2>
- Uncaught TypeError: Cannot read properties of null (reading 'switchOn')  
at switchOn ([index.js:29:16](#))  
at switchOnBtn.onclick ([index.js:147:5](#))
- Uncaught TypeError: Cannot read properties of null (reading 'switchOff')  
at switchOff ([index.js:38:16](#))  
at switchOffBtn.onclick ([index.js:152:5](#))
- Uncaught TypeError: Cannot read properties of null (reading 'setContentPreferences')  
at setContentPreferences ([index.js:48:16](#))  
at [HTMLSelectElement.<anonymous>](#) ([index.js:164:5](#))

# Example - Video Track Manual Controls

## Screenshot #1

### Video Track Automatic Controls

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to a Room with 'auto' mode. This is the default mode.
 * @param {string} token - AccessToken for joining the Room
 * @returns {Room}
 */
function joinRoom(token) {
  return Video.connect(token, {
    name: 'my-cool-room',
    bandwidthProfile: {
      video: {
        contentPreferencesMode: 'auto',
        clientTrackSwitchOffControl: 'auto'
      }
    }
  });
}

module.exports.joinRoom = joinRoom;
```

### Remote Video Controls

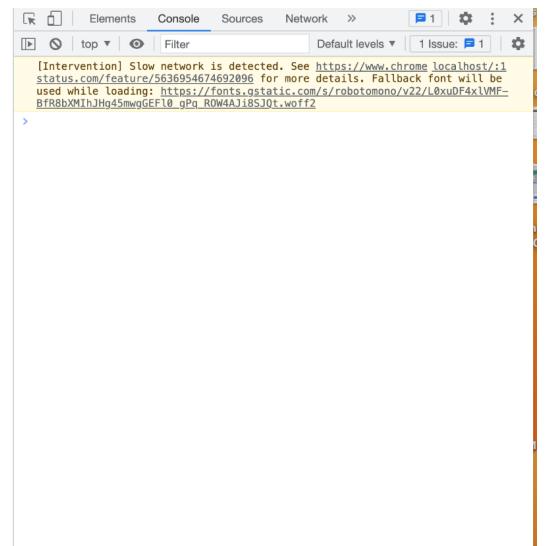
Toggle Visibility:

Video Size:

### Remote Video Track



### Video Bitrate



## Screenshot #2

localhost:3001/autorenderhint/

### Video Track Automatic Controls

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to a Room with 'auto' mode. This is the default mode.
 * @param {string} token - AccessToken for joining the Room
 * @returns {Room}
 */
function joinRoom(token) {
  return Video.connect(token, {
    name: 'my-cool-room',
    bandwidthProfile: {
      video: {
        contentPreferencesMode: 'auto',
        clientTrackSwitchOffControl: 'auto'
      }
    }
  });
}

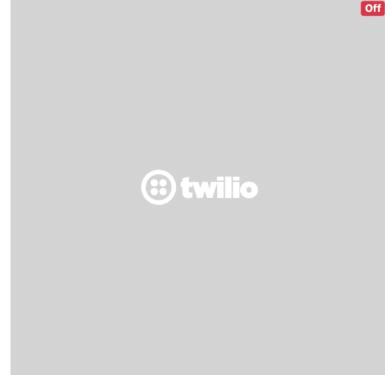
module.exports.joinRoom = joinRoom;
```

### Remote Video Controls

Toggle Visibility:

Video Size:

### Remote Video Track



### Video Bitrate



## Screenshot #3

localhost:3001/autorenderhint/

### Video Track Automatic Controls

```
'use strict';

var Video = require('twilio-video');

/**
 * Connect to a Room with 'auto' mode. This is the default mode.
 * @param {string} token - AccessToken for joining the Room
 * @returns {Room}
 */
function joinRoom(token) {
  return Video.connect(token, {
    name: 'my-cool-room',
    bandwidthProfile: {
      video: {
        contentPreferencesMode: 'auto',
        clientTrackSwitchOffControl: 'auto'
      }
    }
  });
}

module.exports.joinRoom = joinRoom;
```

### Remote Video Controls

Toggle Visibility: [Show](#) [Hide](#)

Video Size: [Render Dimensions](#)

### Remote Video Track



The video track is currently active ('On').

### Video Bitrate



Time	Bitrate (approx.)
1:00 - 16:13:00	0
16:14:00 - 16:14:50	350-400
16:15:00 - Present	0

