

## Lab - Using Cisco Webex for Developers List Rooms API

### Objectives

- Send an API request using Cisco Webex Teams for Developers to list Webex Teams rooms

### Background / Scenario

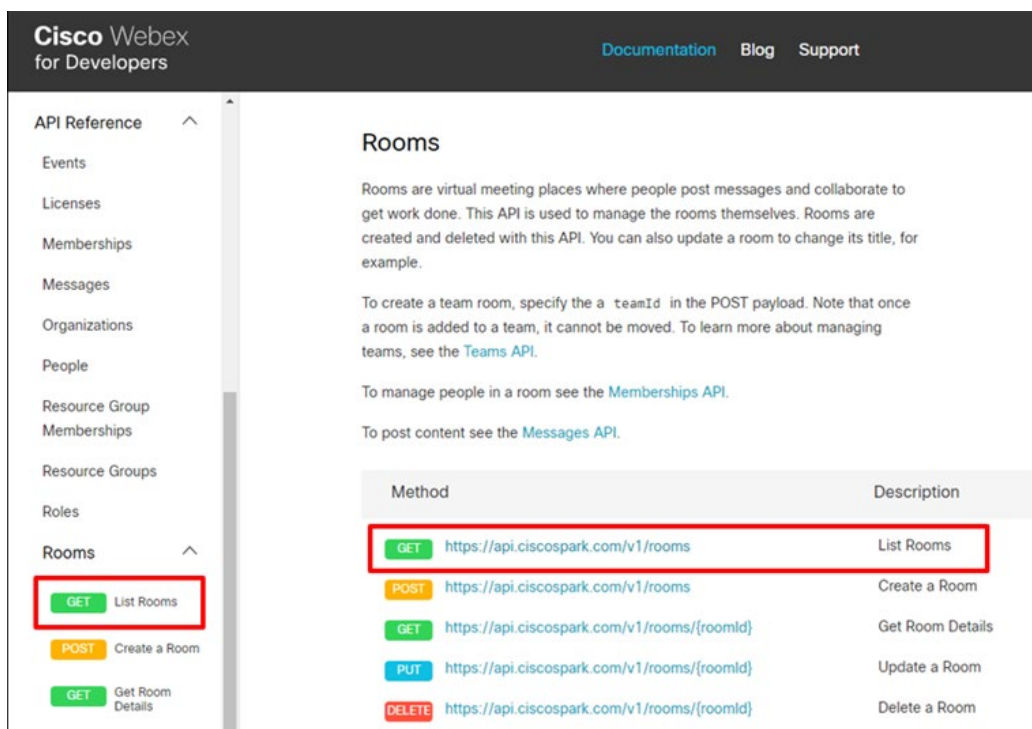
In this lab, you will use your Access Token and Cisco Webex for Developers to make an API request to list your Webex Teams rooms. You will then use the id for one of your Webex Teams rooms to make another API request to list all the messages for that room.

### Required Resources

- Webex Teams account
- Webex Teams desktop application

### Step 1: Investigate the documentation for the Rooms API.

The **Rooms** API is used to manage the rooms themselves, including listing, creating, updating, and deleting rooms. Under **API Reference**, click **Rooms**. Note the HTTP methods that are available here. We are going to use the GET API method to retrieve a list of rooms associated with an account:



**Cisco Webex for Developers** Documentation Blog Support

**API Reference**

- Events
- Licenses
- Memberships
- Messages
- Organizations
- People
- Resource Group Memberships
- Resource Groups
- Roles
- Rooms**
  - GET** List Rooms
  - POST** Create a Room
  - GET** Get Room Details

### Rooms

Rooms are virtual meeting places where people post messages and collaborate to get work done. This API is used to manage the rooms themselves. Rooms are created and deleted with this API. You can also update a room to change its title, for example.

To create a team room, specify the `teamId` in the POST payload. Note that once a room is added to a team, it cannot be moved. To learn more about managing teams, see the [Teams API](#).

To manage people in a room see the [Memberships API](#).

To post content see the [Messages API](#).

Method	Description
<b>GET</b> <a href="https://api.ciscospark.com/v1/rooms">https://api.ciscospark.com/v1/rooms</a>	List Rooms
<b>POST</b> <a href="https://api.ciscospark.com/v1/rooms">https://api.ciscospark.com/v1/rooms</a>	Create a Room
<b>GET</b> <a href="https://api.ciscospark.com/v1/rooms/{roomId}">https://api.ciscospark.com/v1/rooms/{roomId}</a>	Get Room Details
<b>PUT</b> <a href="https://api.ciscospark.com/v1/rooms/{roomId}">https://api.ciscospark.com/v1/rooms/{roomId}</a>	Update a Room
<b>DELETE</b> <a href="https://api.ciscospark.com/v1/rooms/{roomId}">https://api.ciscospark.com/v1/rooms/{roomId}</a>	Delete a Room

### Step 2: Investigate the List Rooms method.

Select **List Rooms** in the **API Reference** column. You should see something similar to the following:

**List Rooms**  
List rooms.  
The `title` of the room for 1-to-1 rooms will be the display name of the other person.  
By default, lists rooms to which the authenticated user belongs.  
Long result sets will be split into [pages](#).

**GET** `/v1/rooms`

**Query Parameters**

Name	Description
<code>teamId</code> string	List rooms associated with a team, by ID.
<code>type</code> string	List rooms by type.
<code>sortBy</code> string	Sort results.
<code>max</code> number	Limit the maximum number of rooms in the response.

**Response Properties**

Name	Description
<code>id</code> string	A unique identifier for the room.
<code>title</code> string	A user-friendly name for the room.

**Try It** **Example**

**Header**

Authorization ☒ Use personal access token

**Bearer** `*****`

*This limited-duration personal access token is hidden for your security.*

**Parameters**

`teamId` `Y2IzY29zcGFyYXZovL3VzL1JPT00vNjRINDVh.`

`type` `direct,group`

`sortBy` `id,lastactivity,created`

`max` `1000`

**Run**

The left side of the **List Rooms** method provides details for the **Query Parameters**, **Response Properties**, and **Response Codes** (scroll down). The right side of the **List Rooms** method provides a **Try It** interface for quickly making the request.

In the **Try It** interface, the **Header** section of the page contains the **Authorization** for the request. It has the value **Bearer** followed by your access token, which is automatically included.

Your access token is prefaced by the word “Bearer” followed by a space because the value is known as a bearer token. The IETF defines the use of a bearer token as; “Any party in possession of a bearer token (a “bearer”) can use it to get access to the associated resources (without demonstrating possession of a cryptographic key). To prevent misuse, bearer tokens need to be protected from disclosure in storage and in transport.”

In the **Parameters** section, you see the query parameters for the **List Rooms** method. These parameters are used when we want to query information regarding a particular value. If the parameter is mandatory, then the word **Required** will be under the parameter name. For the **List Rooms** method, there are no required parameters.

On the right side, scroll down to see a listing of the **Response Codes** that are used to respond to successful and unsuccessful requests.

### Step 3: Send a request for the List Rooms API and investigate the JSON response.

- To send a request, make sure **Try It** is selected and **Authorization** is turned on, as shown below.
- Click **Run** to send the API call and retrieve a list of your rooms.

## Lab - Using Cisco Webex for Developers List Rooms API

- c. To see the full width of the output in the response section, click the expand button in the top right corner.

The screenshot shows the Cisco Webex API Explorer interface. On the left, the 'List Rooms' endpoint is documented with a description and query parameters. On the right, the configuration panel is shown with several numbered callouts:

- 1**: Points to the 'List Rooms' title in the documentation.
- 2**: Points to the 'Authorization' section where 'Use personal access token' is selected.
- 3**: Points to the 'Run' button at the bottom right of the configuration panel.
- 4**: Points to the expand button (two arrows) in the top right corner of the configuration panel.

**Documentation (Left):**

**List Rooms**  
List rooms.  
The `title` of the room for 1-to-1 rooms will be the display name of the other person.  
By default, lists rooms to which the authenticated user belongs.  
Long result sets will be split into [pages](#).

**GET** `/v1/rooms`

**Query Parameters**

Name	Description
<code>teamId</code> string	List rooms associated with a team, by ID.
<code>type</code> string	List rooms by type.
<code>sortBy</code> string	Sort results.
<code>max</code> number	Limit the maximum number of rooms in the response.

**Response Properties**

Name	Description
<code>id</code> string	A unique identifier for the room.
<code>title</code> string	A user-friendly name for the room.

**Configuration (Right):**

**Header**

**Authorization** ☒ Use personal access token

**Bearer** `*****`

*This limited-duration personal access token is hidden for your security.*

**Parameters**

`teamId` `Y2lzY29zcGFyazovL3VzL1JPT00vNjRINDVh,`

`type` `direct,group`

`sortBy` `id,lastactivity,created`

`max` `1000`

- d. The response code should be **200**, as shown below. If it is something other than this, check the Response Codes to discover what the issue is.

The screenshot shows the 'Response' section of the API Explorer. The status code is **200**. The response is a JSON array of room objects. The first object is expanded, showing its details. The `id` value for the first room is highlighted with a red box.

```
{
  "items": [
    {
      "id": "Y2lzY29zcGFyazovL3VzL1JPT00vNjRINDVh",
      "title": "My new room",
      "type": "group",
      "islocked": false,
      "lastActivity": "2018-12-14T14:23:03.135Z",
      "creatorId": "Y2lzY29zcGFyazovL3VzL1BFT1BMR591ZTYwODQxOj00MjZLTQ0YmMtOTBhZi04Yjd1MTAzOGQ5MzQ",
      "created": "2018-01-10T15:35:38.997Z"
    },
    {
      "id": "Y2lzY29zcGFyazovL3VzL1JPT00vNjRINDVh",
      "title": "Test Space",
      "type": "group",
      "islocked": false,
      "lastActivity": "2018-01-03T15:38:35.121Z",
      "creatorId": "Y2lzY29zcGFyazovL3VzL1BFT1BMR591ZTYwODQxOj00MjZLTQ0YmMtOTBhZi04Yjd1MTAzOGQ5MzQ",
      "created": "2018-01-02T15:50:54.835Z"
    },
    {
      "id": "Y2lzY29zcGFyazovL3VzL1JPT00vMTkyNmU3ZDItNGQxNC0zNmEyLWZlZmIwZjZjYzg4ZjNm",
      "title": "Sheyda Tomkins",
      "type": "direct"
    }
  ]
}
```

- e. Notice that the list of rooms is displayed in the JSON format. The `id` value is used as a key to uniquely identify each room. Other information in the JSON output includes the name of the room (`title`) and the ID of the person who created the room (`creatorId`).
- f. Choose a room that you know has messages and copy the `id` value (the long alphanumeric string of text) following the `id` key. Do not include the quotes in your selection. The `id` value for Test Space is selected

above. The ID information uniquely identifies this room and will be used in the next step to obtain list all the messages for that room.

### Step 4: List messages for a specific room using the roomId query parameter.

- Select **Messages** in the API Reference column and then select **List Messages** in one of the two places shown below.

The screenshot shows the Cisco Webex for Developers API Reference page. The left sidebar lists API categories: API Reference, Events, Licenses, Memberships, Messages, Organizations, and People. The 'Messages' category is expanded, showing four endpoints: GET List Messages, POST Create a Message, GET Get Message Details, and DELETE Delete a Message. The 'List Messages' endpoint is highlighted with a red box. The main content area is titled 'Messages' and provides an overview of the Messages API, including a description of messages and a note about room membership. Below the text is a table listing the API endpoints with their methods, URLs, and descriptions. The first row, 'GET https://api.ciscospark.com/v1/messages List Messages', is highlighted with a red box.

Method	Description
GET	https://api.ciscospark.com/v1/messages List Messages
POST	https://api.ciscospark.com/v1/messages Create a Message
GET	https://api.ciscospark.com/v1/messages/{messageId} Get Message Details
DELETE	https://api.ciscospark.com/v1/messages/{messageId} Delete a Message

- b. Notice that the **roomId** parameter is required. Paste the ID value you copied in the previous step.

The screenshot shows the Cisco Webex API client interface. At the top, there are tabs for 'Try it' and 'Example'. Below this is a text input field for the URL: `GET /v1/messages{?,mentionedPeople,before,beforeMessage,max}`. The value `Y2lzY29zcGFyYXovL3VzL1JPT00vZThlNTIyNTAtZjYxYi0xMWU3LWUwZjktYmY3MWRlNjE5MmUw` is pasted into the field. Below the URL field is the 'Header' section, which includes an 'Authorization' toggle set to 'Use personal access token' and a 'Bearer' token field containing a masked token. Below the header is the 'Parameters' section, which includes several input fields: 'roomId' (required), 'mentionedPeople', 'before', 'beforeMessage', and 'max'. The 'roomId' field is highlighted with a red box and contains the same long alphanumeric string as the URL. The 'mentionedPeople' field contains a long alphanumeric string. The 'before' field contains a date and time string. The 'beforeMessage' field contains a long alphanumeric string. The 'max' field contains the number '1000'. At the bottom right of the interface is a yellow 'Run' button, which is also highlighted with a red box.

Try it Example

GET /v1/messages{?,mentionedPeople,before,beforeMessage,max} Y2lzY29zcGFyYXovL3VzL1JPT00vZThlNTIyNTAtZjYxYi0xMWU3LWUwZjktYmY3MWRlNjE5MmUw

Header

Authorization ☒ Use personal access token

Bearer \*\*\*\*\*

This limited-duration personal access token is hidden for your security.

Parameters

roomId **Required** Y2lzY29zcGFyYXovL3VzL1JPT00vZThlNTIyNTAtZjYxYi0xMWU3LWUwZjktYmY3MWRlNjE5MmUw  
List messages in a room, by ID.

mentionedPeople Y2lzY29zcGFyYXovL3VzL1BFT1BMRS8yNDlmNzRkOS1kYjhLTQzY2EiODk2Yi04NzllZDI0MGFjNTM  
List messages with these people mentioned, by ID. Use 'me' as a shorthand for the current API user.

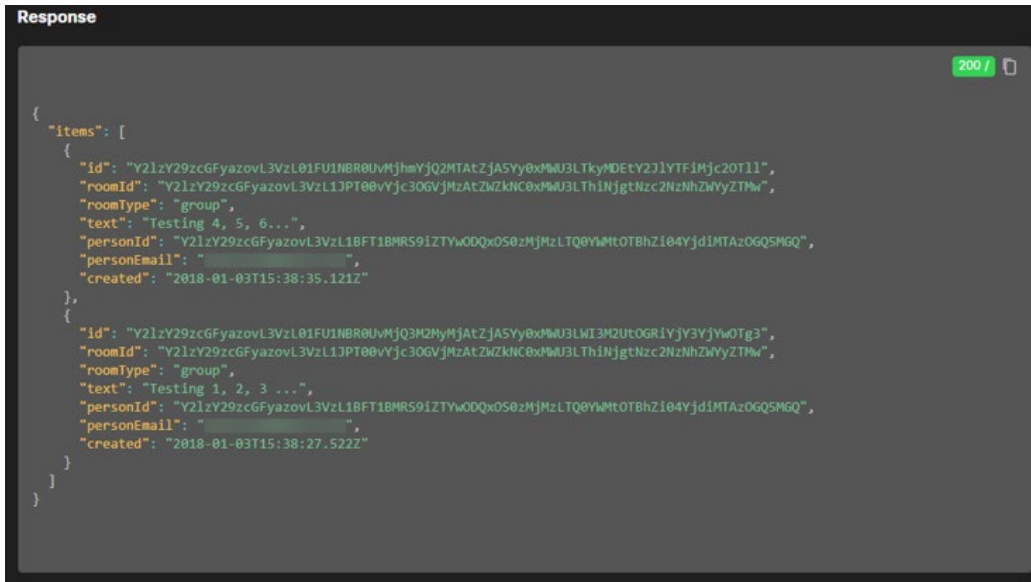
before 2016-04-21T19:01:55.966Z  
List messages sent before a date and time.

beforeMessage Y2lzY29zcGFyYXovL3VzL01FU1NBR0UvOTJkYjNiZTAiNDNiZC0xMWU2LThhZTkzZGQ1YjNkZmM1NjVh  
List messages sent before a message, by ID.

max 1000  
Limit the maximum number of messages in the response.

Run

- c. Click **Run**. Select a message and examine the **text** key of the JSON output. Compare the response JSON to the messages for the room that are displayed in the Webex Teams application. They should be the same.



```
{
  "items": [
    {
      "id": "Y2l2Y29zcGFyYXovL3VzL01FU1NBR0UvMjhmYjQ2MTAtZjA5Yy0xMmU3LTkyMDYyZjJlYTFIMjc2OT1l",
      "roomId": "Y2l2Y29zcGFyYXovL3VzL1JPT08vYjc3OGVjMzAtZWZkNC0xMmU3LTlhInJgtNzc2NzNhZWYyZTRw",
      "roomType": "group",
      "text": "Testing 4, 5, 6...",
      "personId": "Y2l2Y29zcGFyYXovL3VzL1BFT1BMRS9lZTYwODQxODUzMjMzLTQ0YVhtOTBhZi04YjdIMTAzOGQ5SMGQ",
      "personEmail": "",
      "created": "2018-01-03T15:38:35.121Z"
    },
    {
      "id": "Y2l2Y29zcGFyYXovL3VzL01FU1NBR0UvMjhmYjQ2MTAtZjA5Yy0xMmU3LTkyMDYyZjJlYTFIMjc2OT1l",
      "roomId": "Y2l2Y29zcGFyYXovL3VzL1JPT08vYjc3OGVjMzAtZWZkNC0xMmU3LTlhInJgtNzc2NzNhZWYyZTRw",
      "roomType": "group",
      "text": "Testing 1, 2, 3 ...",
      "personId": "Y2l2Y29zcGFyYXovL3VzL1BFT1BMRS9lZTYwODQxODUzMjMzLTQ0YVhtOTBhZi04YjdIMTAzOGQ5SMGQ",
      "personEmail": "",
      "created": "2018-01-03T15:38:27.522Z"
    }
  ]
}
```