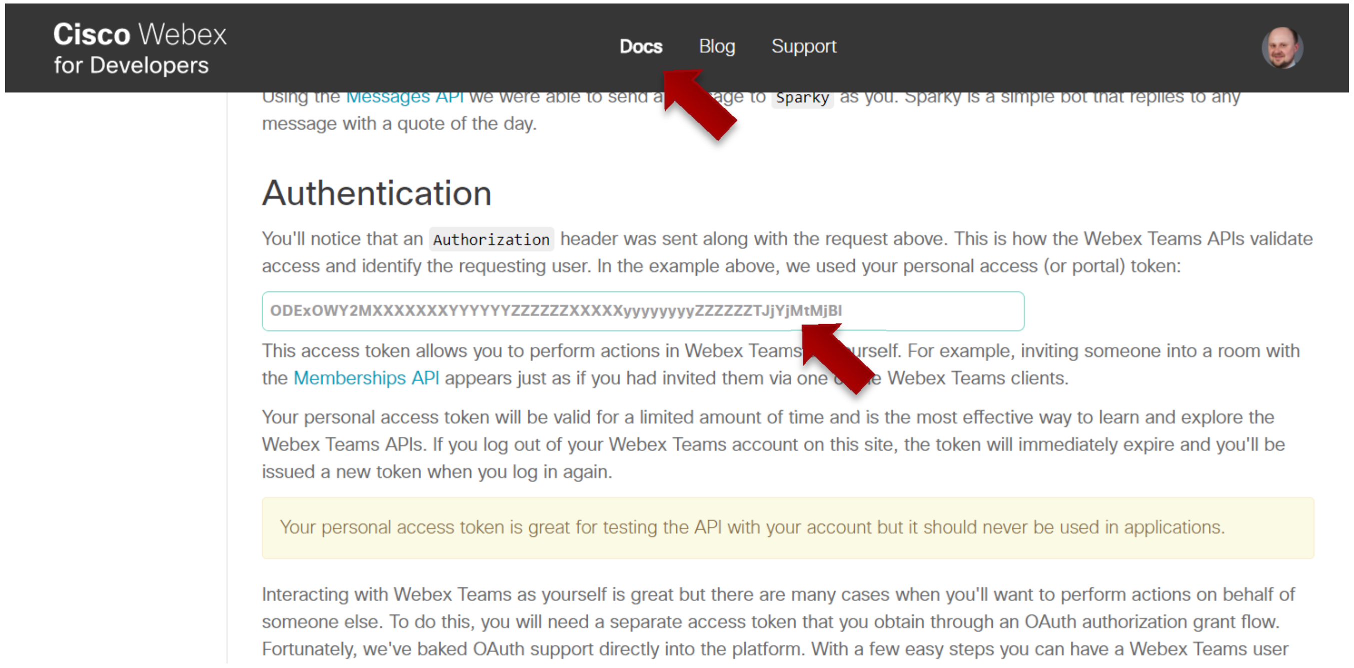
Interact with Spark from PT smart device

Getting your access token

1. Go to https://developer.webex.com/
2. Log in with your Spark account
3. Go to **Docs** page
4. Scroll down the **Getting Started** page down to **Authentication** section
5. Copy your personal **access token**



Using the Messages API we were able to send a message to `Sparky` as you. Sparky is a simple bot that replies to any message with a quote of the day.

Authentication

You'll notice that an `Authorization` header was sent along with the request above. This is how the Webex Teams APIs validate access and identify the requesting user. In the example above, we used your personal access (or portal) token:

```
ODExOWY2MXXXXXXXYYYYYYZZZZZZXXXXXyyyyyyyyZZZZZZTJjYjMtMjBl
```

This access token allows you to perform actions in Webex Teams as yourself. For example, inviting someone into a room with the Memberships API appears just as if you had invited them via one of the Webex Teams clients.

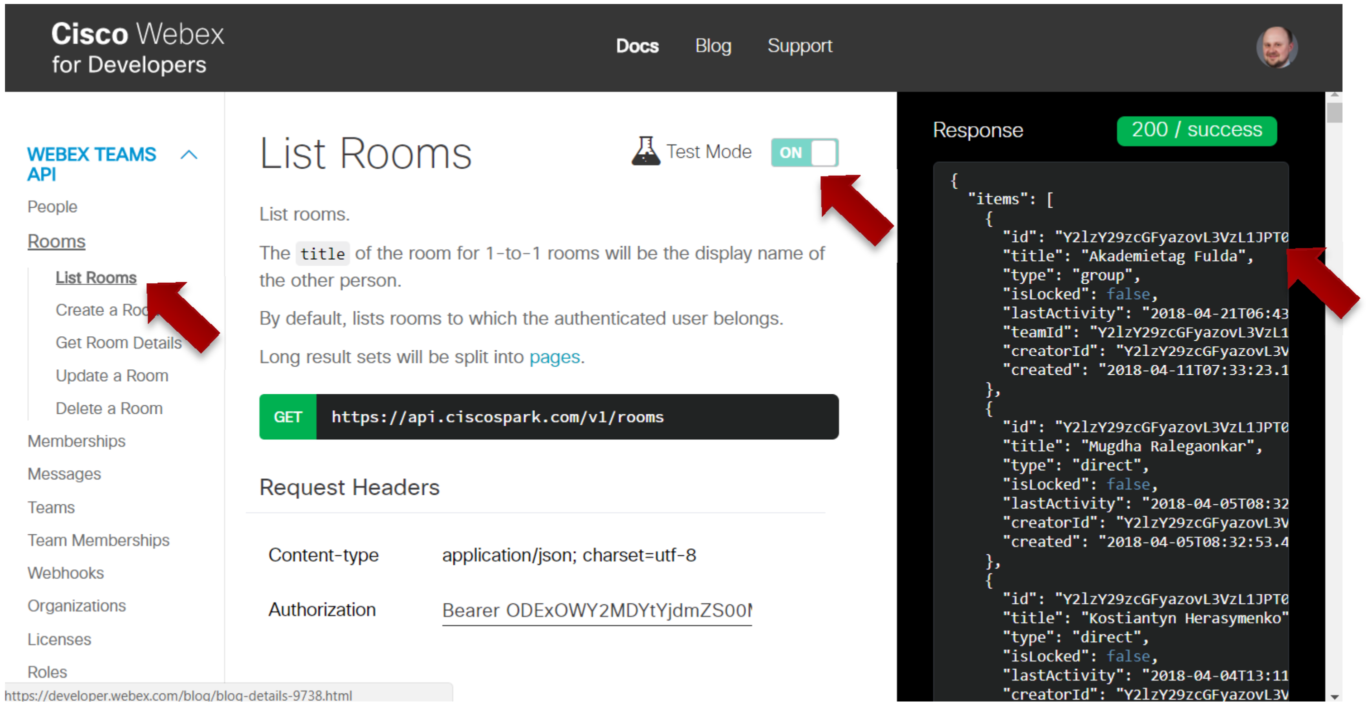
Your personal access token will be valid for a limited amount of time and is the most effective way to learn and explore the Webex Teams APIs. If you log out of your Webex Teams account on this site, the token will immediately expire and you'll be issued a new token when you log in again.

Your personal access token is great for testing the API with your account but it should never be used in applications.

Interacting with Webex Teams as yourself is great but there are many cases when you'll want to perform actions on behalf of someone else. To do this, you will need a separate access token that you obtain through an OAuth authorization grant flow. Fortunately, we've baked OAuth support directly into the platform. With a few easy steps you can have a Webex Teams user

Getting the room id

1. From the **Docs** page,
2. Under **API Reference**, navigate to **Rooms – List Rooms**
3. Enable **Test Mode**
4. Click **Run**
5. Copy room **id** from the server Response



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  https://api.ciscospark.com/v1/rooms
```

Request Headers

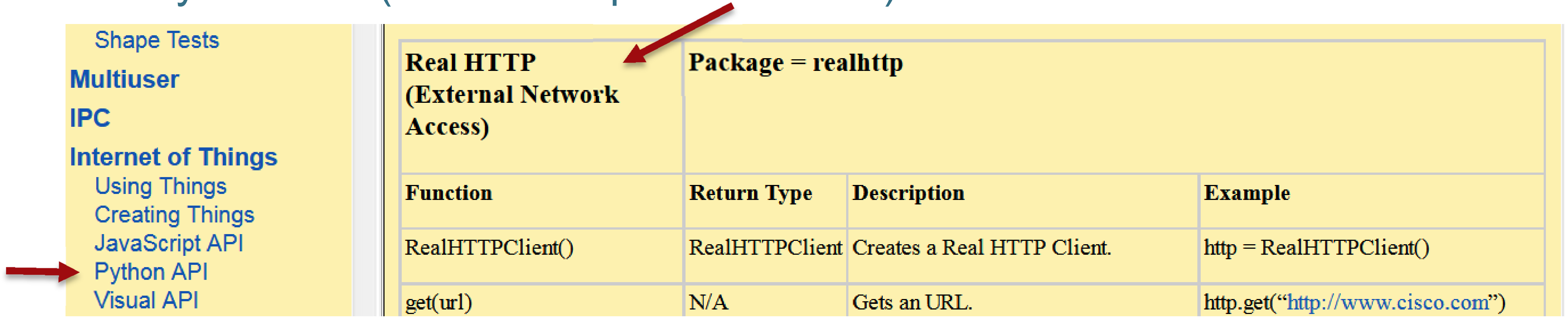
Content-type	application/json; charset=utf-8
Authorization	Bearer ODExOWY2MDYtYjdmZS00I

```
{
  "items": [
    {
      "id": "Y2lzY29zcGFyazovL3VzL1JPT0
      "title": "Akademietag Fulda",
      "type": "group",
      "isLocked": false,
      "lastActivity": "2018-04-21T06:43
      "teamId": "Y2lzY29zcGFyazovL3VzL1
      "creatorId": "Y2lzY29zcGFyazovL3V
      "created": "2018-04-11T07:33:23.1
    },
    {
      "id": "Y2lzY29zcGFyazovL3VzL1JPT0
      "title": "Mugdha Ralegaonkar",
      "type": "direct",
      "isLocked": false,
      "lastActivity": "2018-04-05T08:32
      "creatorId": "Y2lzY29zcGFyazovL3V
      "created": "2018-04-05T08:32:53.4
    },
    {
      "id": "Y2lzY29zcGFyazovL3VzL1JPT0
      "title": "Kostiantyn Herasymenko"
      "type": "direct",
      "isLocked": false,
      "lastActivity": "2018-04-04T13:11
      "creatorId": "Y2lzY29zcGFyazovL3V
```

https://developer.webex.com/blog/blog-details-9738.html

“Real” API

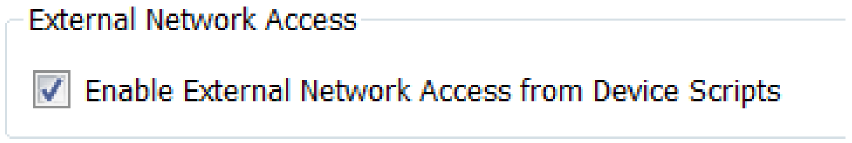
- In PT7, your smart device can communicate with real world using TCP, UDP, and HTTP protocols. Functions that help to do that described in Python API (PT7 → Help → Contents):



Shape Tests
Multiuser
IPC
Internet of Things
Using Things
Creating Things
JavaScript API
Python API
Visual API

Real HTTP (External Network Access)	**Package = realhttp**		
Function	**Return Type**	**Description**	**Example**
RealHTTPClient()	RealHTTPClient	Creates a Real HTTP Client.	http = RealHTTPClient()
get(url)	N/A	Gets an URL.	http.get(“http://www.cisco.com”)

- By default, external network communication is disabled. Enable it in Options → Preferences → Miscellaneous → External Network Access

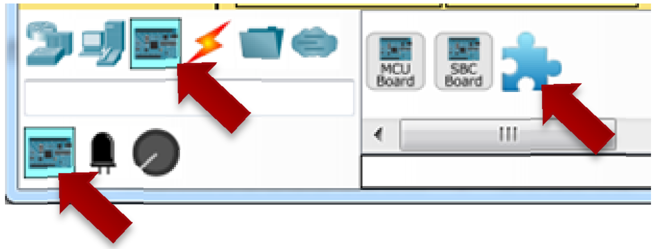


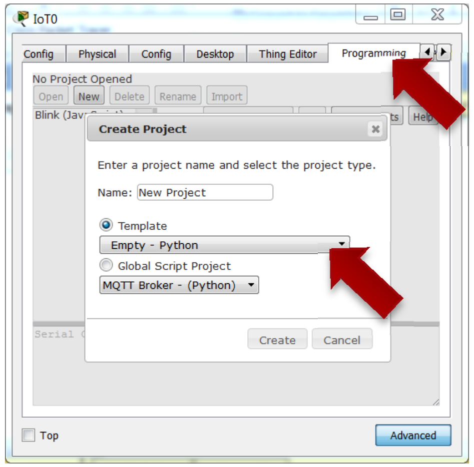
External Network Access

[✓] Enable External Network Access from Device Scripts

Coding for Spark messaging

8. Create new device
9. Go to the **Programming** tab
10. Create a new empty Python project





No Project Opened

Enter a project name and select the project type.

Name: New Project

[✓ Template] Empty - Python

Global Script Project: MQTT Broker - (Python)

11. Use this code sample:

```
from realhttp import *

roomId = "Y2lzY29zcZZZzzzYOURroomIDzzzZZZzzzZZZ"
accessToken = "YmRkYZZZzzzzZZZzzzZZZzzz..."
message = "1 Test message to Spark from Packet Tracer 7.1!"

http = RealHTTPClient()

http.postWithHeader(
    "https://api.ciscospark.com/v1/messages",
    {"roomId":roomId,"text":message},
    {"Authorization":"Bearer "+accessToken, "Content-Type":"application/json"}
    )
delay(1000)
```

Import additional Python module

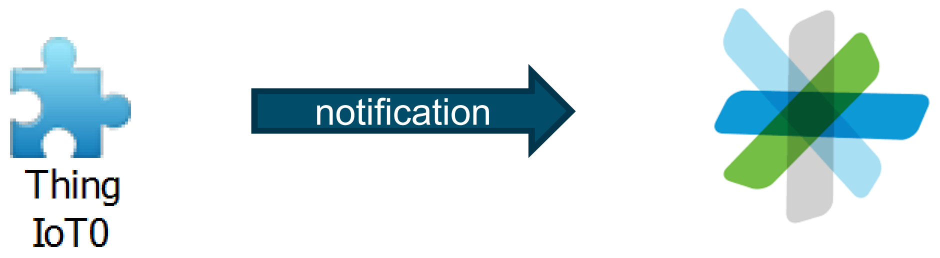
Define variables to store room ID, access token and message you want to send

This performs the HTTP POST request

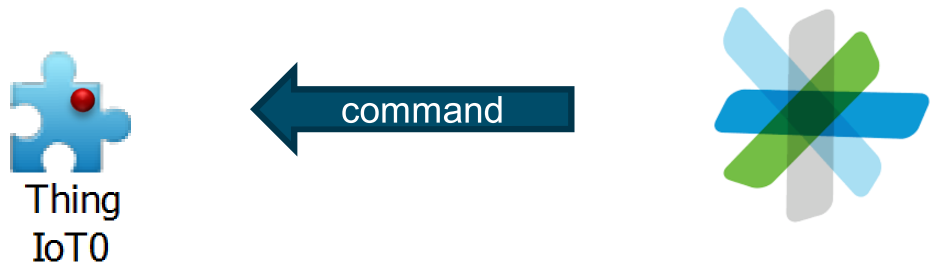
A small delay to let request to be processed

Challenges

Challenge 1:
Pick any smart device and modify its Python code to **send notification to Spark** when device state changes



Challenge 2:
Pick any smart device and modify its Python code to **react on control commands** from the Spark room


Thing
IoT0
command