API BotConnect

Table of Contents

Administration Part

Access

To access administration of the server, you must connect to the ip address where the server is deployed (will name IP in the document) follow by /admin/ tag.

Example:

IP = 172.24.1.1:8000

The URL is so http://172.24.1.1:8000/admin/.

Possibility

An administrator can add and delete different objects of the database server.

These things are especially users, commands and message. These concepts are quite abstract for now but it will be explain in the next part of the document.

The interface is intuitive, it won’t be explain in the document.

Restrictions

For the API works, the database must at least contain one user.

Add of command and message mustn’t be done with an administration account.

API Part

Information

The API is done with the Framework Django and Django Rest.

The interaction between API and the others clients are Http requests.

All these requests have their content compose of JSON. So please inform you on the JSON format before to read this documentation.

To have a good understanding of the server source, you must know python, Django, Django Rest, Json and Pygame (for camera).

Run

To run the server, you must deploy it.

To deploy it, you use the next command line on the root of ConnectBot/API/b2b_server :

```
$ python manage.py runserver IP
```

If during, the reception of image you have the error “Permission Denied” the command must be launch in admin.

```
$ sudo python  manage.py runserver IP
```

Package

The server is composed of three parts:

- The root of ConnectBot/API/b2b_server
- The folder ConnectBot/API/b2b_server/b2b_control_s
- The folder ConnectBot/API/b2b_server/b2b_server

These split is quite classic in a Django Project.

ConnectBot/API/b2b_server contained:

- Database name “db.sqlite3”
- Script to deploy server “manage.py”
- Https useful stuffs as ssl.crt, ssl.key, etc

ConnectBot/API/b2b_server/b2b_server contained the general settings of the application as the routes of the URL in setting.py.

ConnectBot/API/b2b_server/b2b_control_s contained all models, views (controller in Django), migrations, etc of the API.

It the most important part of the source. If a modification must be done, it is in this folder.

Login

Before to exchange data with the server you must login with an account create by the administrator.

In the deliverable database, we have the user Geoffrey which have the password Geoffrey.

We will use it has example.

Example:

Client Side:

The client must do a **POST** request to the next URL: http://IP/api/users/login/.

These request must contain in his content a JSON composed of field “username” and password.

In our example the JSON will be:

{ { “username” : “Geoffrey” , “password” : “Geoffrey”} }

The client will receive as response a **token** with the name of the user. These token must save somewhere because it will be ask for all actions with the server.

For our request we can have as answer:

{ {“token” : “SJHIORUDK0°SPOIKJSOIPKSDOP.S44s7d5s4sqsd8q4s8dsqqssqqsikoudhsoiqsq447” ,”username” : Geoffrey }}

Important

In all the next sections, we must use this token as a header of the request.

This header is { “Authorization” : “JWT”+token}

All these request are also JSON so they must also contain the next header :

{ “Content-type”, “application/json” }

Command

The next section treat about the URL: http://IP/api/command/ .

Post

The post of a command must be done by the **client** side.

A command correspond to the action we want to execute on the robot.

A command can have value, in the current server a value is the number of time we repeat the character.

For example:

If the character “o” is use to make the robot forward.

We can post a request with the content:

{ { “characters” : “o”} }

If we want to forward two time faster:

{ { “characters” : “o”, “value” : “2”}}

Get

The get of a command must be done by **the script link with the robot**.

The script must send these command to the command by different ways as serial connection.

For the example of the Post we will receive:

We need to convert the receive JSON to understandable characters for the robot.

So must transform to “o” and the second to “oo”.

Image

The next section treat about the URL: http://IP/api/image/get/ .

As the title indicate it, the url is use to get the image from the camera.

It is no use to precise the method of the request because it will have the same result.

The result of the request is the data of the snapshot of the robot.

Video

The server does not have for now a streaming of video. To replace it, the client can request a lot of photos in real time. It simulate the effect of video (a true video have the same principia) but with latency. If the reader of this document is motivate, he can it in the server.

Message

The next section treat about the URL: http://IP/api/message/ .

A message correspond to all types of data which can be send by the robot. For our robot, we use to have the information of sensors.

Post

In the think logic, it is **the script of the robot** to send the message.

But it is not shocking to have the client to send message too.

An example of request will be:

{ { “message” : “30 45.117 7888”}}

Get

The get request will receive all the messages send by the script.

So the script can send data even it is not receive by the client.