comfort click

Hello!

Here we have a short tutorial on how to integrate bOS and IFTTT using the new bOS RPC service.

bOS offers a JSON RPC Service for integration with IFTTT, Tasker and other services.

This tutorial will show bOS RPC service interacting with IFTTT in a simple task, but the important thing here is to understand how to set-up the RPC service so you can do more complex integrations.

RPC service offers HTTP basic authentication with username, and password.

In this example, we'll be using IFTTT to turn a KNX light ON based on your location, so a geofencing scenario that could be changed and used to open a door, turn the music on, etc.

The possibilities are endless.

NOTE: Webhooks is now only available for **IFTTT PRO+** users.

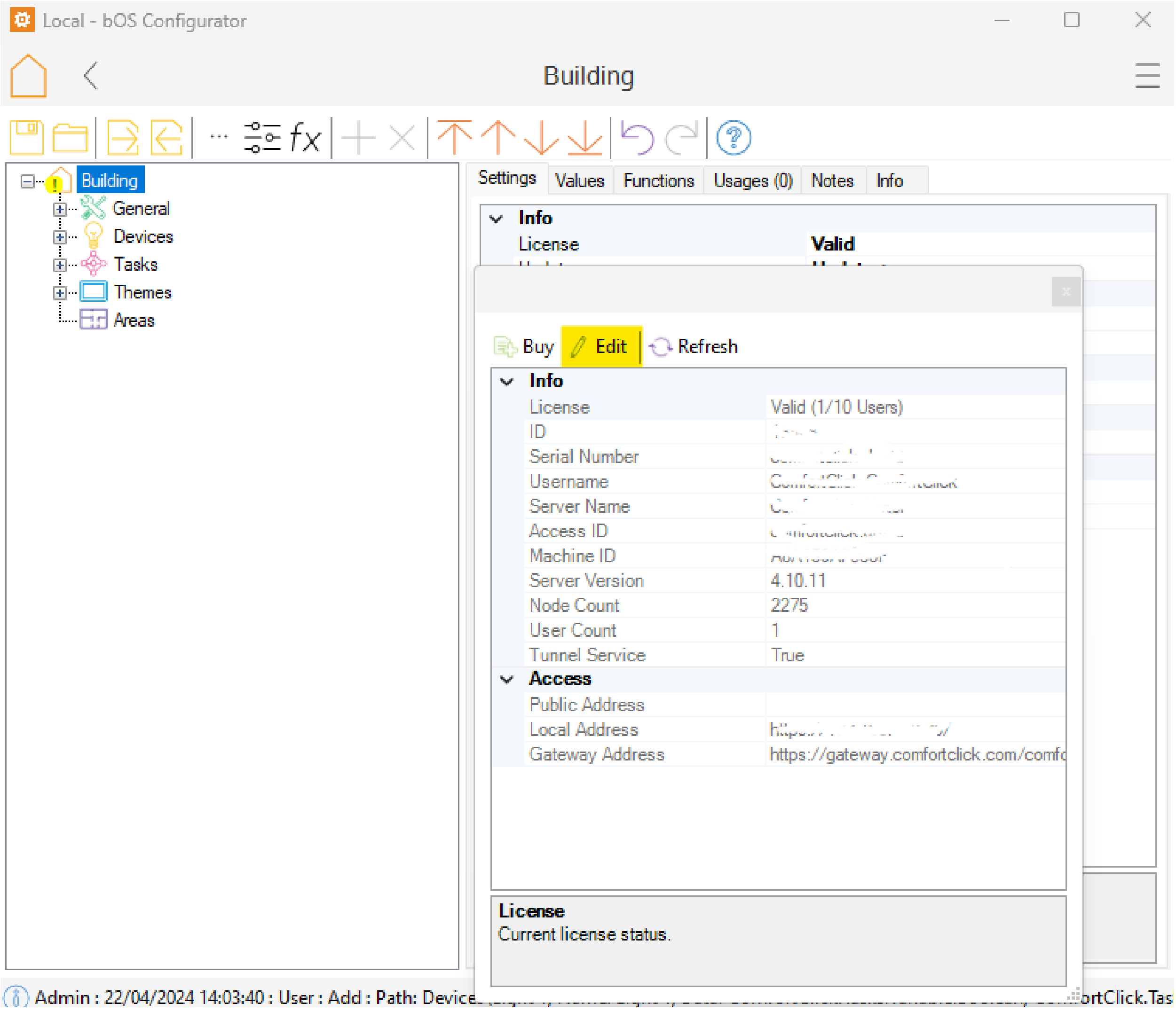
Step 1: Port forwarding

- Since the command will be coming from IFTTT cloud, your server must be set to receive commands from external networks by having port forwarding enabled on your router. You can check your port forwarding on this site: https://www.yougetsignal.com/tools/open-ports/ If you have port forwarding already enabled, you can skip to Step 2

- A static WAN IP address is needed to access your network and should be obtained from your internet provider. If you don’t have a static WAN IP address, use a dynamic DNS service. On your router setup page enable port forwarding (NAT) to the local IP ddress of your controller. It is recommended to only forward the port 443 for secure https connection.

- Now open bOS Configurator, under Building, open the menu from the License section:

comfort click



Info

License	Valid

Info

License	Valid (1/10 Users)
ID	[redacted]
Serial Number	[redacted]
Username	[redacted]
Server Name	[redacted]
Access ID	[redacted]
Machine ID	[redacted]
Server Version	4.10.11
Node Count	2275
User Count	1
Tunnel Service	True

Access

Public Address	
Local Address	[redacted]
Gateway Address	https://gateway.comfortclick.com/comfo

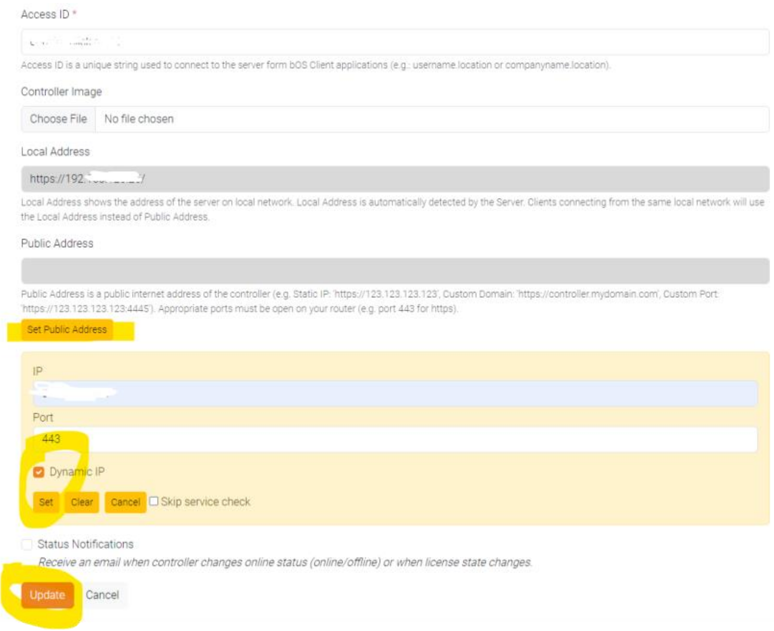
License
Current license status.

Admin : 22/04/2024 14:03:40 : User : Add : Path: Device ... ortClick.Tas

Click “Edit” and comfortclick page will open. Login into your correct account and fill out the network settings.

Press the “Set Public Address” button, fill out the public IP and forwarded port, for Dynamic Ip tick the Dynamic IP option and press set button.

Once everything is set, press the Update button at the bottom of the page.



Access ID *

[redacted]

Access ID is a unique string used to connect to the server form bOS Client applications (e.g.: username.location or companyname.location).

Controller Image

Choose File | No file chosen

Local Address

https://192.[redacted]/

Local Address shows the address of the server on local network. Local Address is automatically detected by the Server. Clients connecting from the same local network will use the Local Address instead of Public Address.

Public Address

Public Address is a public internet address of the controller (e.g. Static IP: 'https://123.123.123.123', Custom Domain: 'https://controller.mydomain.com', Custom Port: 'https://123.123.123.123:4445'). Appropriate ports must be open on your router (e.g. port 443 for https).

Set Public Address

IP

[redacted]

Port

443

☑ Dynamic IP

Set | Clear | Cancel | □ Skip service check

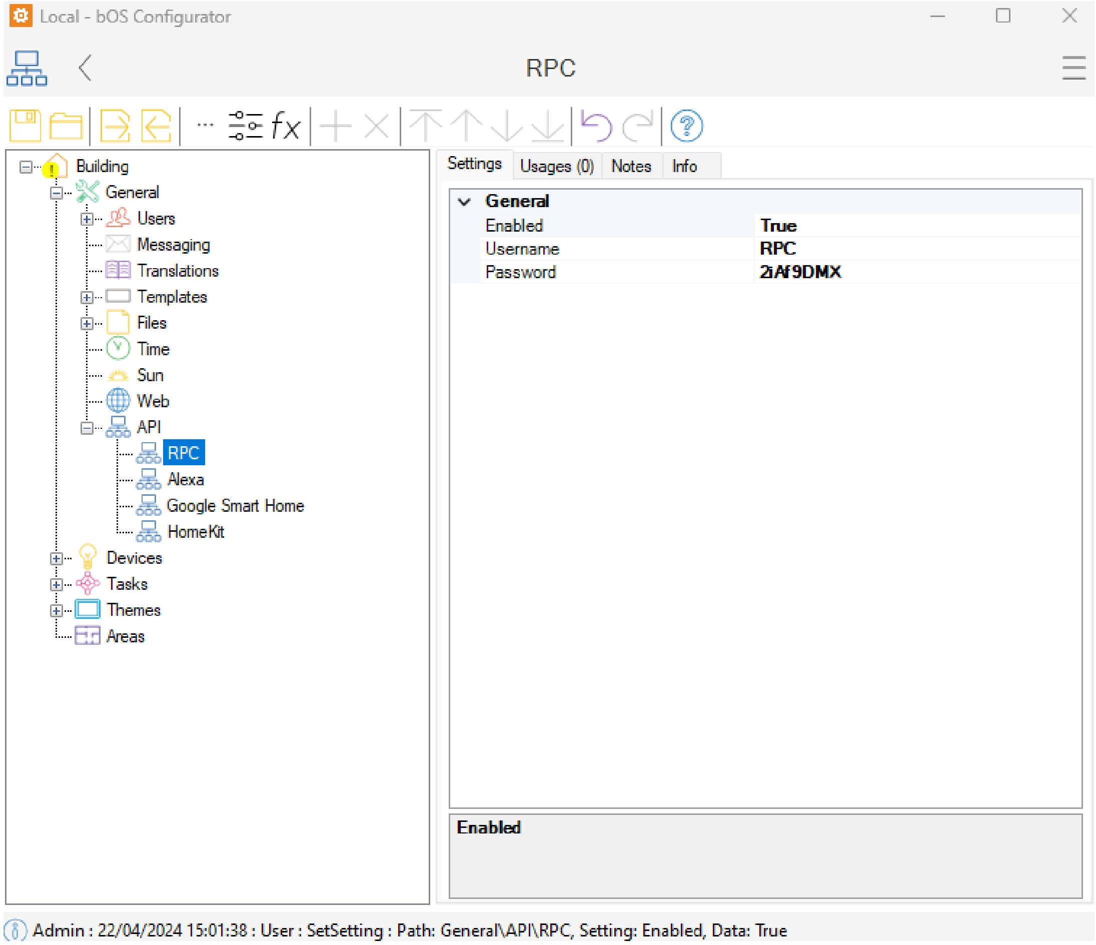
□ Status Notifications

Receive an email when controller changes online status (online/offline) or when license state changes.

Update | Cancel

Step 2: Configure your RPC service

- In bOS Configurator, under API, enable RPC and define your Username and Password:



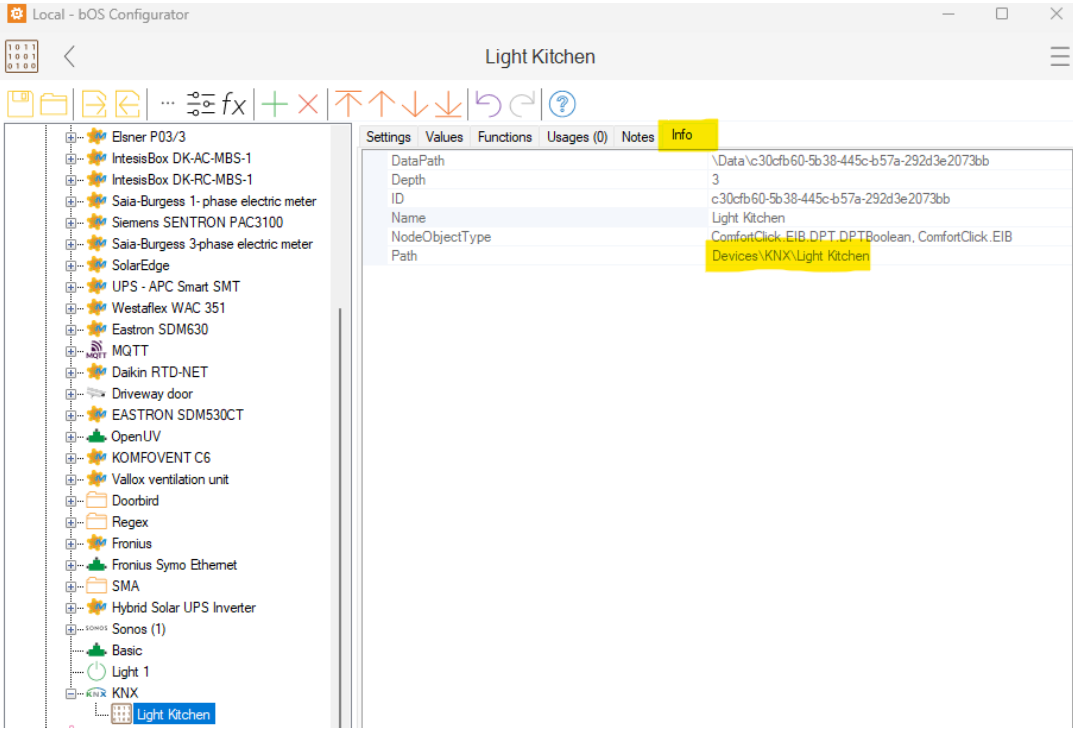
General	
Enabled	True
Username	RPC
Password	2iAf9DMX

Enabled

Admin : 22/04/2024 15:01:38 : User : SetSetting : Path: General\API\RPC, Setting: Enabled, Data: True

Step 3: Find your Device Path

- In this example, the Info tab from the KNX light will show us the path: Devices\KNX\Light Kitchen



DataPath	\Data\c30cfb60-5b38-445c-b57a-292d3e2073bb
Depth	3
ID	c30cfb60-5b38-445c-b57a-292d3e2073bb
Name	Light Kitchen
NodeObjectType	ComfortClick.EIB.DPT.DPTBoolean, ComfortClick.EIB
Path	Devices\KNX\Light Kitchen

Now we need to form the JSON body to POST on IFTTT. The Path we have is: **Devices\KNX\Light Kitchen**

We need to double the backslashes on the Path to form the JSON body to POST on IFTTT service, so now we have: **Devices\KNX\Light Kitchen**

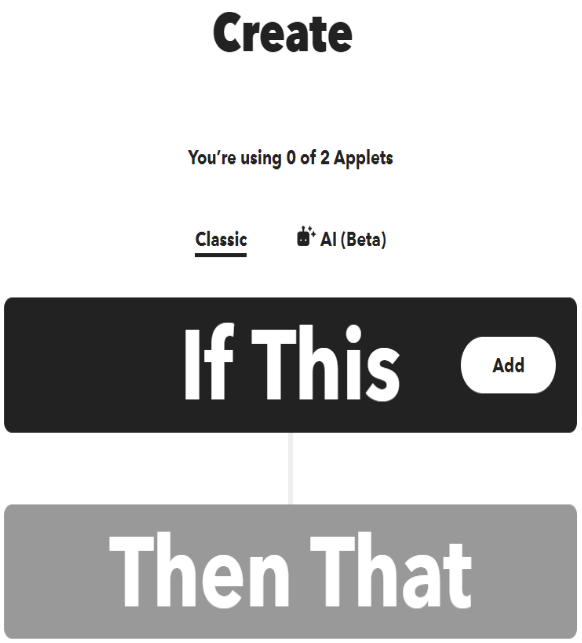
The simple SetValue body post looks something like this: **{"objectName":"INSERT PATH HERE","valueName":"Value","value":"true"} where "value":"true"** is to turn the value to true or 1. You can change to false or 0 or look for other RPC examples at the end of this tutorial.

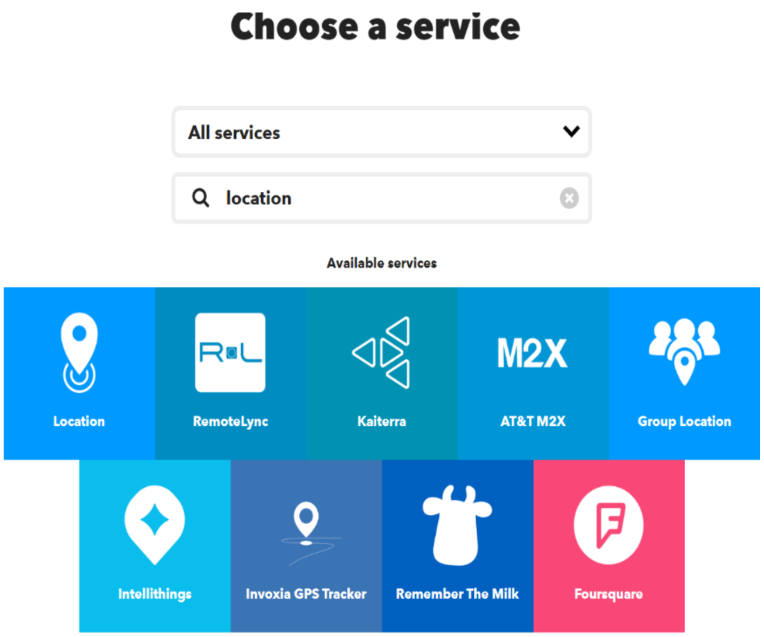
To turn the Light ON, our body post is: **{"objectName":"Devices\\KNX\\Light Kitchen","valueName":"Value","value":"true"}**

*the value, in this case "true", must be written in lowercase.

Step 4: IFTTT

Create a new Applets from scratch, click on “This”, select the “location” trigger and set your address area.





Click on “That”, select the “**Webhooks**” trigger and click on “Make a web request”:

Web request form:

URL

https://RPCUserName:RPCPassword@your_ip_address_or_dynamic_dns_address/API/RPC /SetValue

in our example:

https://RPC:RPCCOLIBRI@ip_address_or_dynamic_dns_address/API/RPC/SetValue

Method

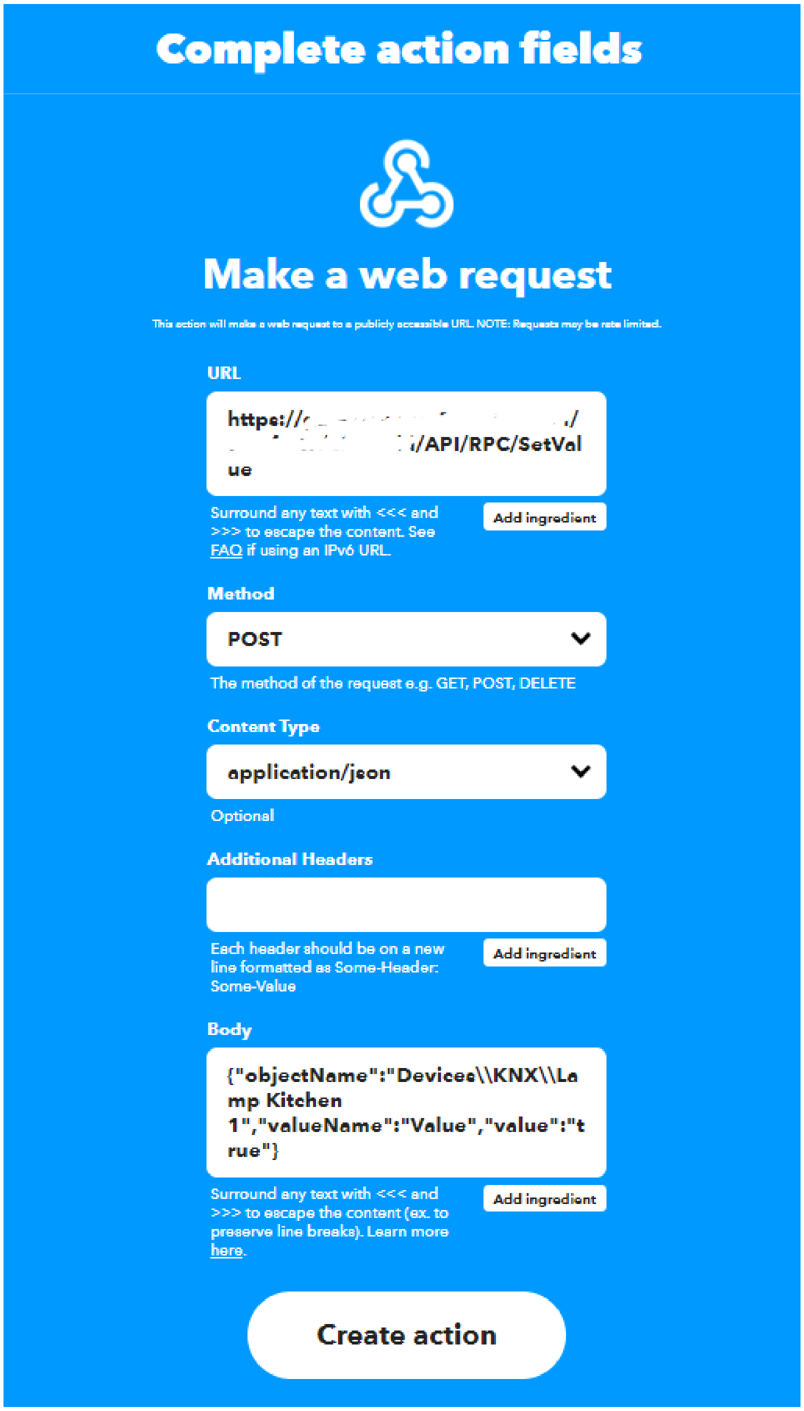
POST

Content Type

application/json

Body

{"objectName":"Devices\\KNX\\Lamp Kitchen","valueName":"Value","value":"true"}



Complete action fields

Make a web request

This action will make a web request to a publicly accessible URL. NOTE: Requests may be rate limited.

URL

https://[redacted]/API/RPC/SetValue

Surround any text with <<< and >>> to escape the content. See FAQ if using an IPv6 URL.

Method

POST

The method of the request e.g. GET, POST, DELETE

Content Type

application/json

Optional

Additional Headers

Each header should be on a new line formatted as Some-Header: Some-Value

Body

{"objectName":"Devices\\KNX\\Lamp Kitchen 1","valueName":"Value","value":"true"}

Surround any text with <<< and >>> to escape the content (ex. to preserve line breaks). Learn more here.

Save and finish your Applet.

Done! Now every time we enter the specified area the KNX light will turn on.

This is a simple example for demo purposes, but with more complex commands you can do whatever you want… call scenes, set other values, etc.

To call an RPC Service an HTTP POST call must be performed with data content-type = "application/json"; Service offers the following commands:

SetValue

POST data example (Turn ON a light): {"objectName":"Devices\\KNX\\Lamp Kitchen","valueName":"Value","value":"true"}

GetValue

POST data example (Get light status): {"objectName":"Devices\\KNX\\ Lamp Kitchen ","valueName":"Value"}

CallFunction

POST data example (Send alert to user): {"objectName":"Building\\General\\Users\\User","functionName":"SendAlert","value":["'Hello'"]}

I hope you've enjoyed the tutorial and please post here the examples and ideas you have using the new bOS' RPC API.