

- 1. 概要
- 2. 簡易 http サーバー
- 3. 標準の Http ブロック
- 4. カスタムブロック
 - 4.1. プログラム作成手順
 - 4.2. カスタムブロックの作り方

atoom lite + Env 2

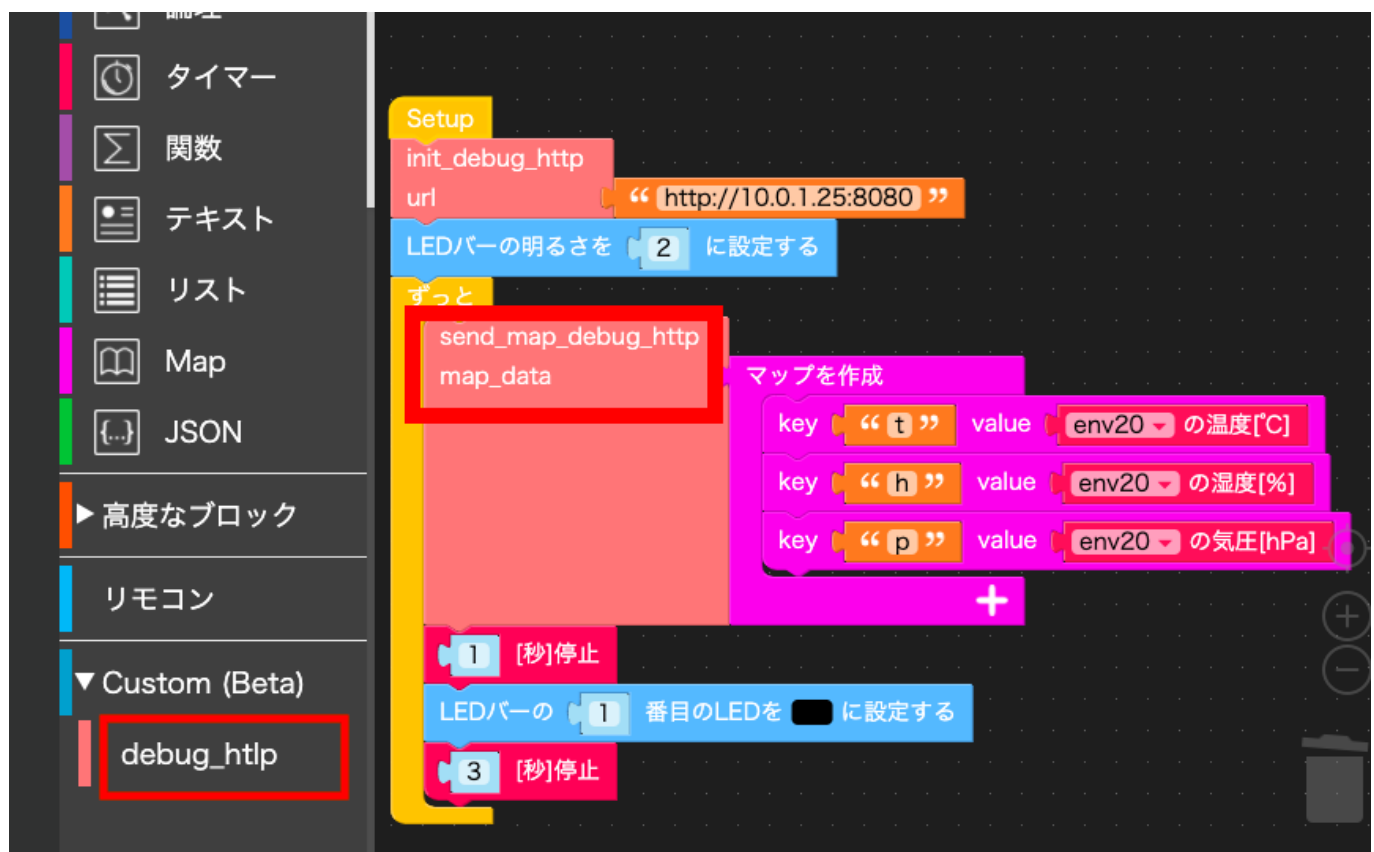
1. 概要

atom lite に Env 2 unit を接続し、uiflow でその値を取得するプログラムを書きました。

atom lite には LED が1つ付いているだけで、数値などを表示することはできません。

そこで数値を http サーバーに値を送るようにします。http サーバー側のログなので、Env 2 で取得した値を確認しようということです。

http 送信のブロックは標準で用意されていますが、custom block をつくって http 送信をしました。



```

ruby http_server.rb
[2020-08-09 09:36:37] INFO WEBrick 1.6.0
[2020-08-09 09:36:37] INFO ruby 2.7.1 (2020-03-31) [x86_64-darwin19]
[2020-08-09 09:36:37] INFO WEBrick::HTTPServer#start: pid=8551 port=8080
{"host"=>["10.0.1.25"], "content-type"=>["application/json", "application/json"], "content-length"=>["38"]}
POST / HTTP/1.0
Host: 10.0.1.25
Content-Type: application/json; charset=UTF-8
Content-Type: application/json
Content-Length: 38
{"t": 29.51, "p": 1001.61, "h": 66.78}
10.0.1.40 - - [09/Aug/2020:09:36:38 JSI] "POST / HTTP/1.0" 200 0
- -> /
{"host"=>["10.0.1.25"], "content-type"=>["application/json; charset=UTF-8", "application/json"], "content-length"=>["38"]}
POST / HTTP/1.0
Host: 10.0.1.25
Content-Type: application/json; charset=UTF-8
Content-Type: application/json
Content-Length: 38
{"t": 29.48, "p": 1001.63, "h": 66.76}
10.0.1.40 - - [09/Aug/2020:09:36:43 JSI] "POST / HTTP/1.0" 200 0
- -> /

```

環境：

- Uiflow 1.6.2
- macos catalina
- ruby 2.7 (on mac)
- [atom lite](#)
- [Env 2](#)

2. 簡易 http サーバー

atom lite からの http 送信先の サーバーを用意します。

言語はなんでもよいのですが、ここでは ruby で書いてみました。http GET と POST の要求があったら requer 内容をコンソールに表示するだけのものです。

[./src/http_server.rb](#)

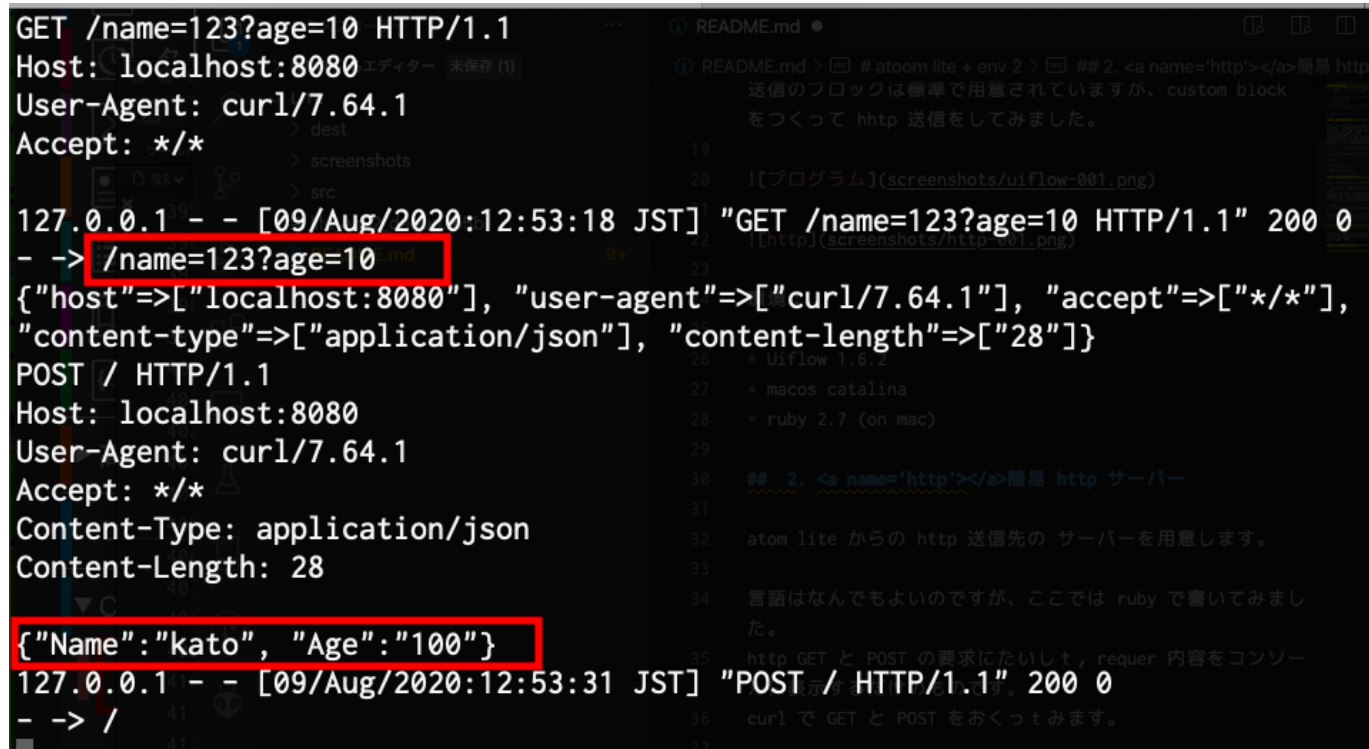
curl で GET と POST を送ってみます。

```

$curl http://localhost:8080/name=123?age=10

$curl -X POST -H "Content-Type: application/json" -d '{"Name":"kato",
"Age":"100"}' localhost:8080

```



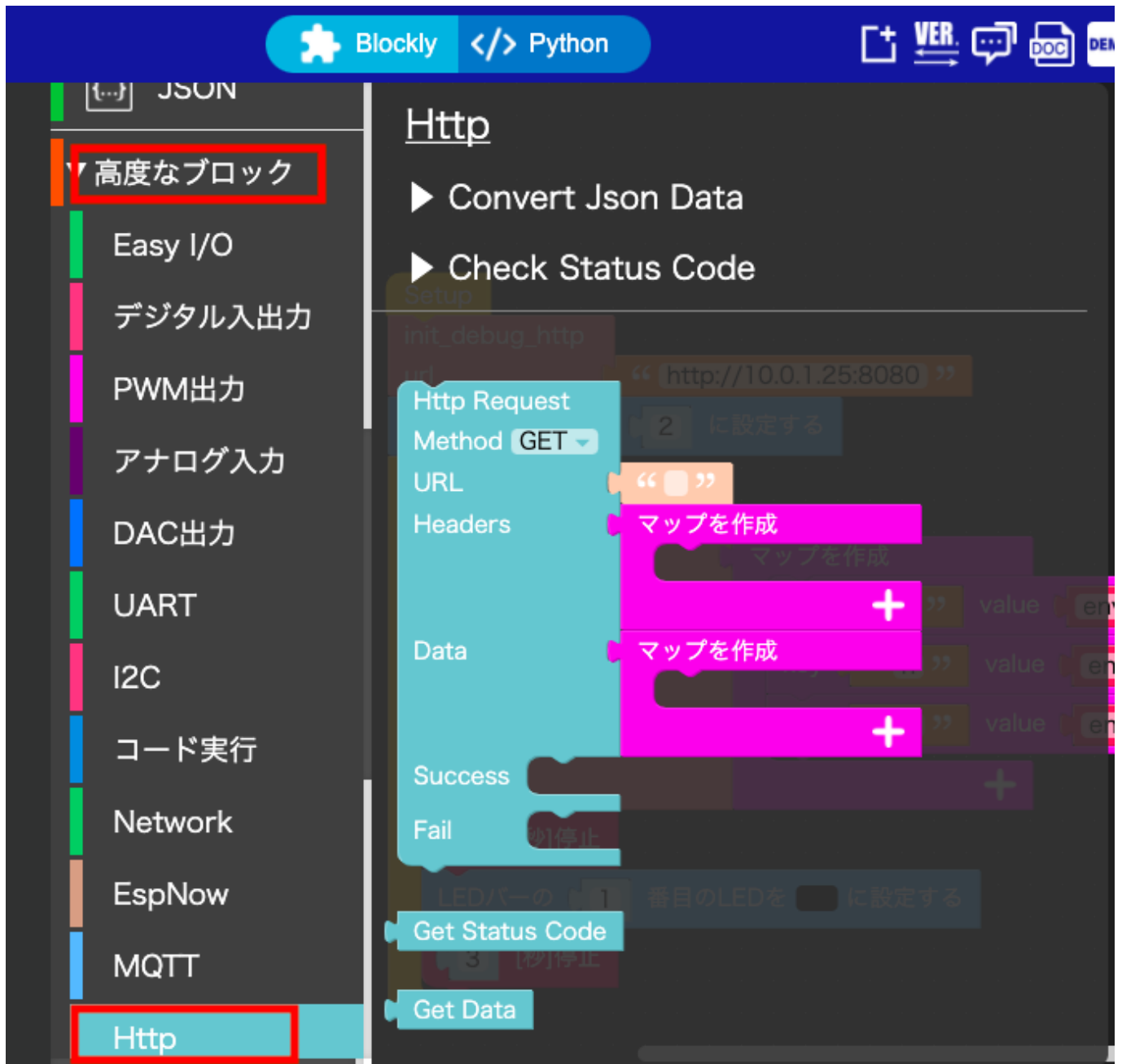
```
GET /name=123?age=10 HTTP/1.1
Host: localhost:8080
User-Agent: curl/7.64.1
Accept: */*

127.0.0.1 - - [09/Aug/2020:12:53:18 JST] "GET /name=123?age=10 HTTP/1.1" 200 0
- -> /name=123?age=10
{"host"=>["localhost:8080"], "user-agent"=>["curl/7.64.1"], "accept"=>["*/*"],
"content-type"=>["application/json"], "content-length"=>["28"]}
POST / HTTP/1.1
Host: localhost:8080
User-Agent: curl/7.64.1
Accept: */*
Content-Type: application/json
Content-Length: 28
{"Name": "kato", "Age": "100"}
127.0.0.1 - - [09/Aug/2020:12:53:31 JST] "POST / HTTP/1.1" 200 0
- -> /
```

実際には、[NODE-Red](#) や [Ambient](#) にデータを送ったりするということになるでしょう。

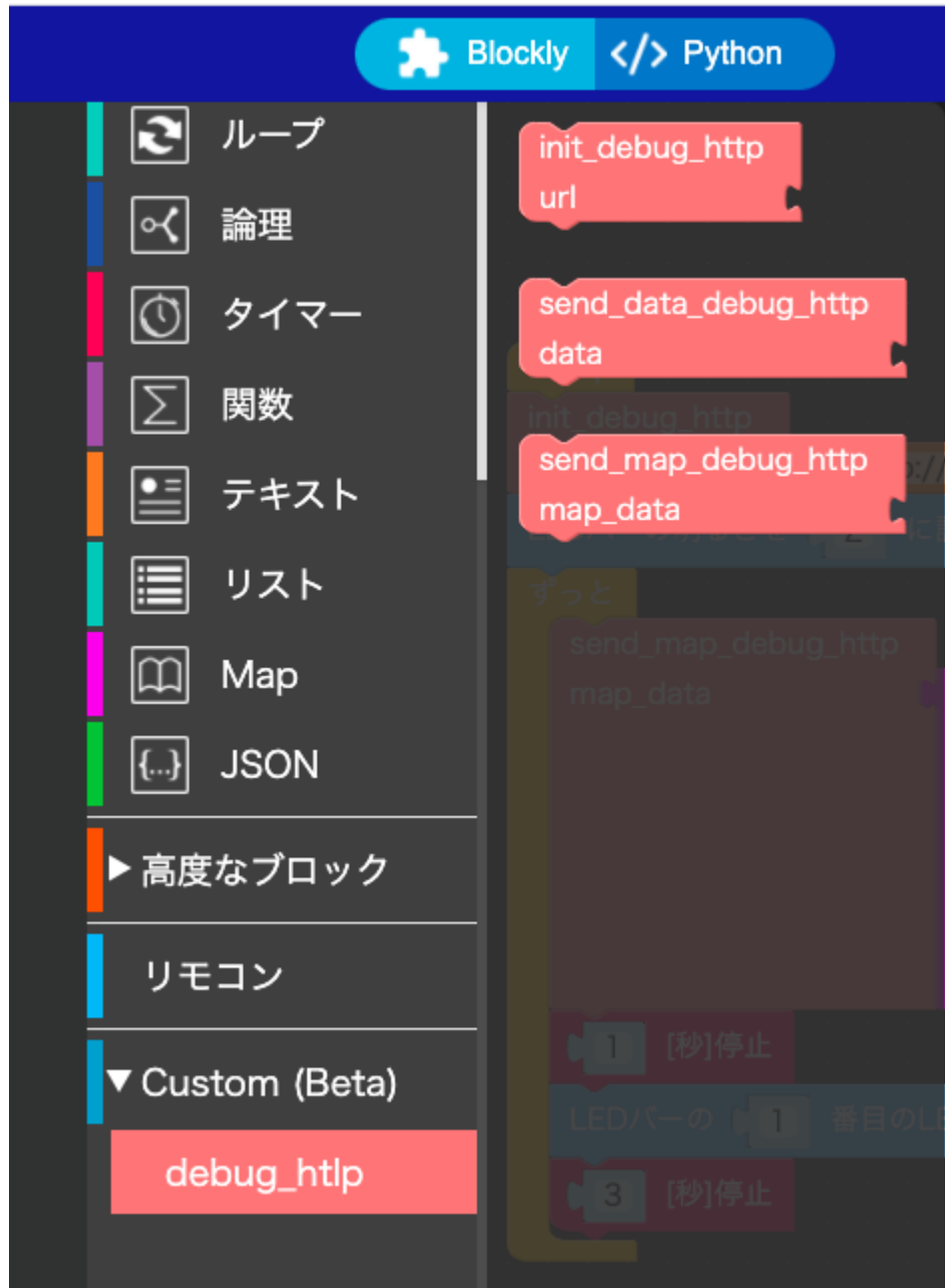
3. 標準の Http ブロック

UiFlow の 高度なブロック の生木 Http のカテゴリがあります。



4. カスタムブロック

atim lite の UiFlow でのプログラムで変数の値などをデバッグのために出力させるのに使いやすいようにカスタムブロックをつくってみました。（シリアル出力の代わりに使うことを想定しています）



各ブロックの機能を述べます。

- `init_debug_http` http サーバー呼び出しの初期化をします。
http サーバーの url をテキストで指定します。(例: "http://10.0.1.1:8080")
- `send_data_debug_http` テキストや数値などを Get で http サーバーに送ります。
送信が成功すれば、LED が青になります。
失敗すれば、LED が赤になります。
data に送信したい情報を指定します。内部で `str(data)` されて GET で送信されます。
- `send_map_debug_http` Map を Post で http サーバーに送ります。
送信が成功すれば、LED が青になります。
失敗すれば、LED が赤になります。
map_data に Map を指定します。

このカスタムブロックをつかったプログラム例を示します。



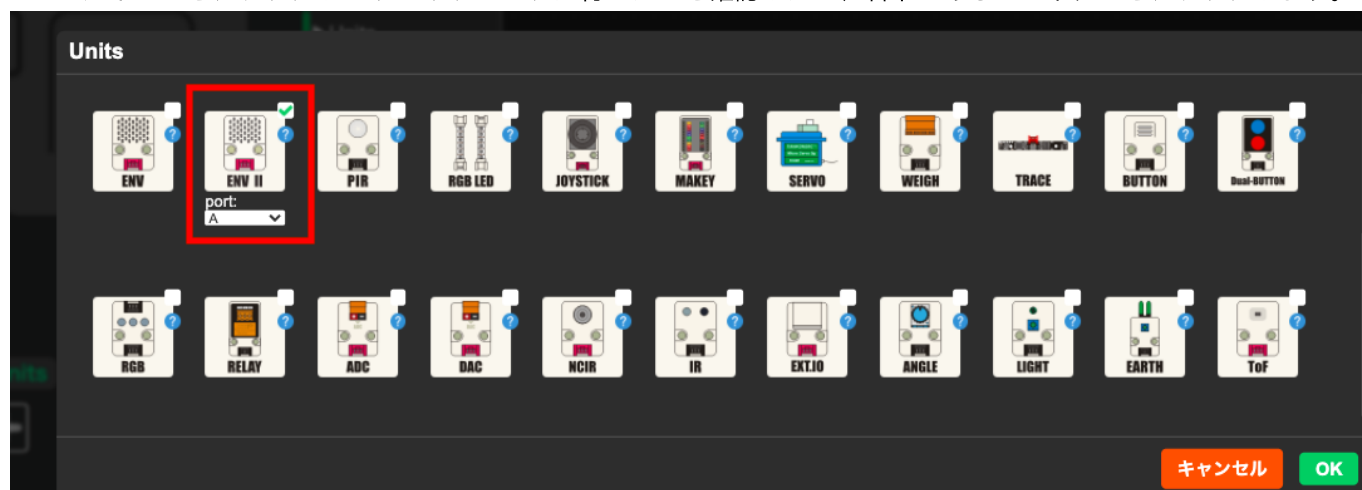
4.1. プログラム作成手順

atom lite に Env 2 をケーブルでつなげます。ケーブルで接続しただけでは、uiflow から Env 2 は利用できません。Env 2 を uiflow に登録する必要があります。

画面左下の + アイコンをクリックします。



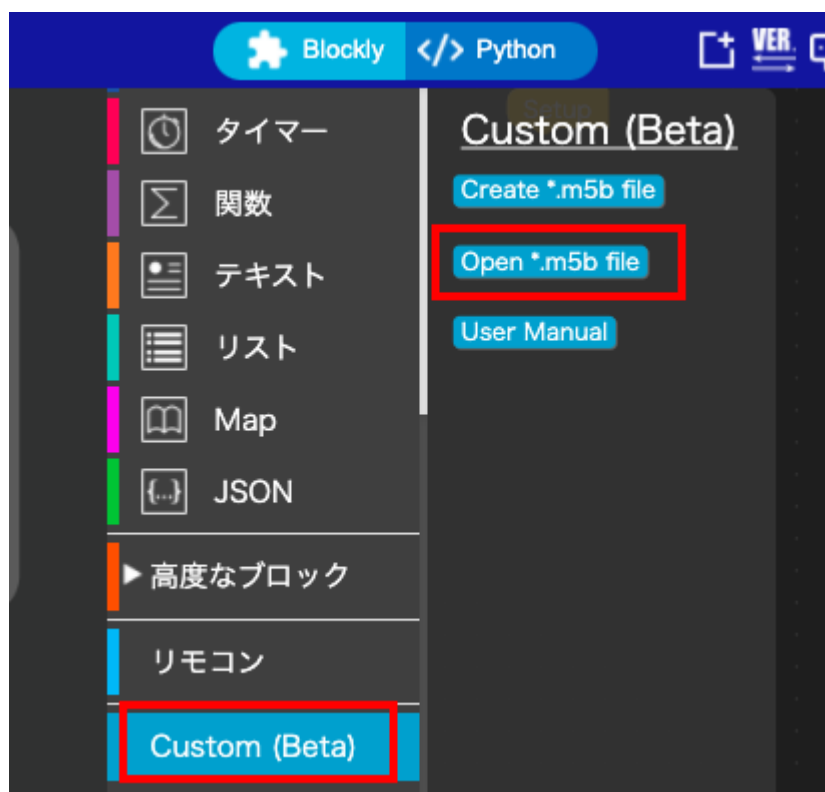
Env 1 アイコンをクリックして チェックマークが付いたのを確認したら、右下にある OK ボタンをクリックします。



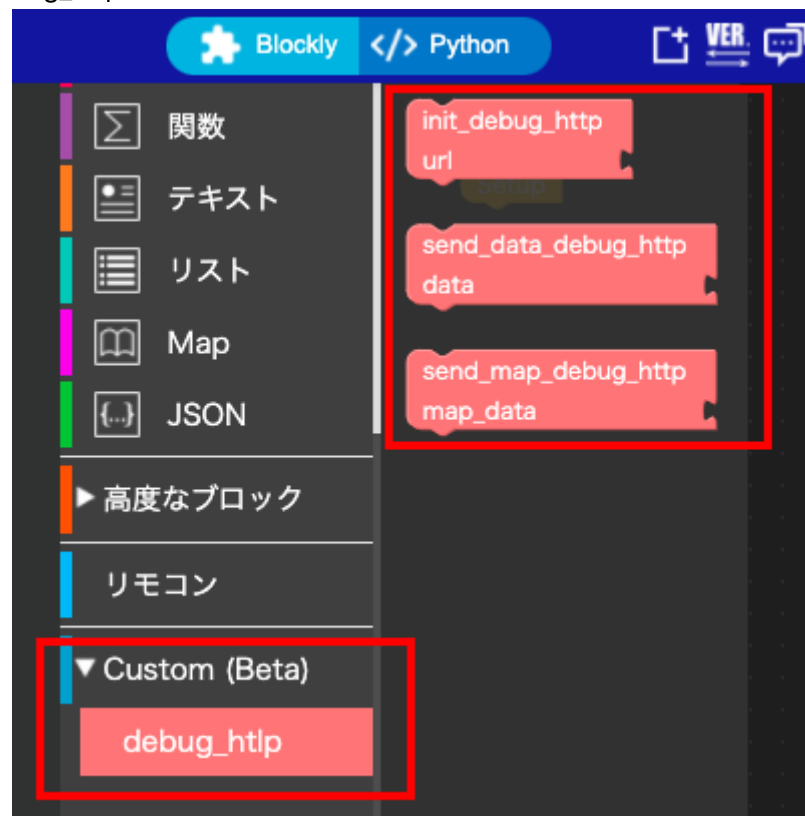
Unit カテゴリーに環境ブロックが追加されます。



debug_http ブロックを追加するには次のようにします。 [debug_http.mb5](#) を予め PC の任意の場所に download して起きます。 Custom カテゴリーを開き、 open *.m5b file をクリックします。



ファイル指定ダイアログが開かれるので、debug_http.m5b を指定します。load が終了すれば、Custom カテゴリ



ーに debug_http ブロックが追加されます。

このddbug_http ブロックにあるブロックは、他のブロックと同等に利用できます。

これで、Env 2 からの情報取得と、カスタムブロック debug_http を利用できるようになりました。

init_debug_http で 送信先 http サーバーの URL を設定し、 ループ中で数秒感覚で Env 2 から得た気温、湿度、気圧の値を Map にして send_map_debug_http で送るようにブロックを組み上げていってプログラムを作成させます。(あるいは、 作成済みのプログラムフィアル [dest/main_0.m5f](#) を load してください)

プログラムを実行させれば、http サーバー起動をした PC のコンソールに 4 秒ごとに 気温, 湿度, 気圧が表示されます。

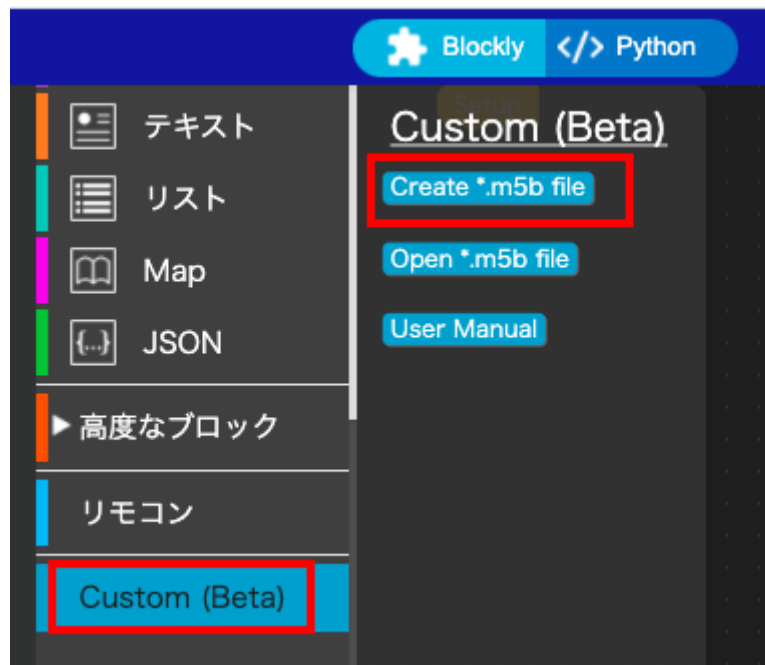
atom lite の LED は 4 秒音に 青くなりあす。(送信が失敗している場合は LED は赤くなります。)

```
Content-Length: 38
{"t": 31.05, "p": 1002.88, "h": 66.65}
10.0.1.40 - - [09/Aug/2020:20:40:32 JST] "POST / HTTP/1.0" 200 0
- -> /
{"host"=>["10.0.1.25"], "content-type"=>["application/json; charset=UTF-8", "application/json"], "content-length"=>["38"]}
POST / HTTP/1.0
Host: 10.0.1.25
Content-Type: application/json; charset=UTF-8
Content-Type: application/json
Content-Length: 38
{"t": 31.08, "p": 1002.83, "h": 66.54}
10.0.1.40 - - [09/Aug/2020:20:40:36 JST] "POST / HTTP/1.0" 200 0
- -> /
{"host"=>["10.0.1.25"], "content-type"=>["application/json; charset=UTF-8", "application/json"], "content-length"=>["38"]}
POST / HTTP/1.0
Host: 10.0.1.25
Content-Type: application/json; charset=UTF-8
Content-Type: application/json
```

4.2. カスタムブロックの作り方

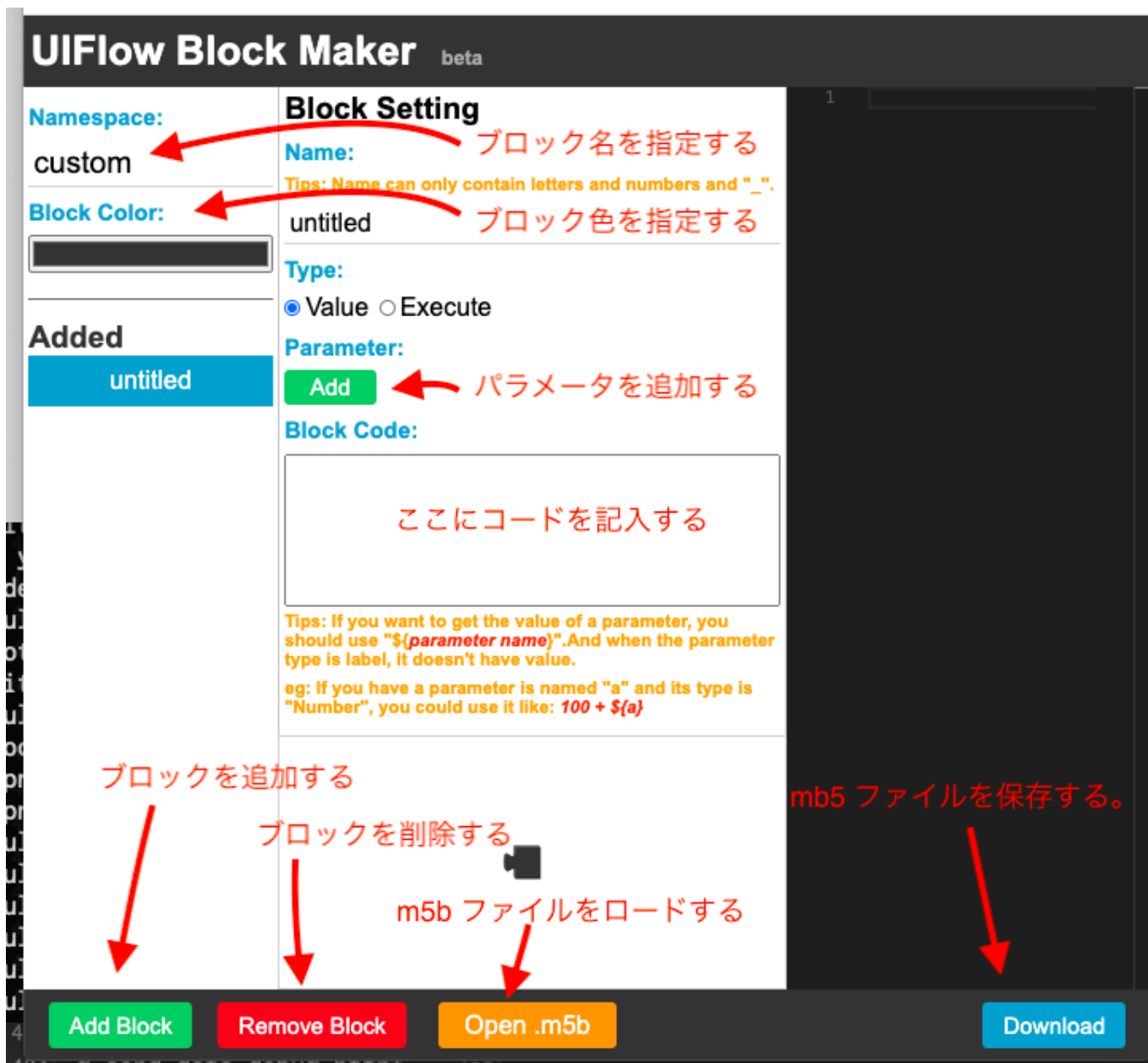
このカスタムブロックの作り方を述べます。

カスタムブロックの作り方を覚えると、uiflow でのプログラムの幅が広がります。



create *.m4b file ボタンをクリックします。

ブロック作成のための Block Maker 画面が開きます。



init_debug_http ブロックを編集します。

このブロックは Excute タイプと指定します。パラメータの一番目はブロックの名前を Label として指定します。これはブロックのアイコンにブロック名として表示されます。2 番目のパラメータに variable タイプで url を指定します。

コード部は以下のものを指定します。

```
import urequests
_g_url_debug_http=${url}
_g_success_color_debug_http=0x000099
_g_fail_color_debug_http=0x990000
def _g_send_data_debug_http(data):
    try:

urequests.request(method='GET',url=_g_url_debug_http+"/data="+str(data),headers={})
    rgb.setColorAll(_g_success_color_debug_http)
except:
```

```

    rgb.setColorAll(_g_fail_color_debug_http)
def _g_send_map_data_debug_http(map_data):
    try:

urequests.request(method='POST',url=_g_url_debug_http,json=map_data,header
s={'Content-Type':'application/json;charset=UTF-8'})
    rgb.setColorAll(_g_success_color_debug_http)
except:
    rgb.setColorAll(_g_fail_color_debug_http)

```

UIFlow Block Maker beta

Namespace: debug_http

Block Color: [Color Selector]

Added

- init_debug_http
- send_data_debug_http
- send_map_debug_http

Block Setting

Name: init_debug_http

Tip: Name can only contain letters and numbers and "_".

Type: ☐ Value ☒ Execute

Parameter:

name: init_debug_http type: Label Remove

url: type: Variable Remove

Block Code:

```

import urequests
_g_url_debug_http=${url}
_g_success_color_debug_http=0x000099
_g_fail_color_debug_http=0x990000
def _g_send_data_debug_http(data):
    try:
        urequests.request(method='POST',url=_g_url_debug_http,json=data,header
s={'Content-Type':'application/json;charset=UTF-8'})
        rgb.setColorAll(_g_success_color_debug_http)
    except:
        rgb.setColorAll(_g_fail_color_debug_http)

```

Tip: If you want to get the value of a parameter, you should use "\${parameter name}". And when the parameter type is label, it doesn't have value.

eg: If you have a parameter named "a" and its type is "Number", you could use it like: 100 + \${a}

Block Preview: init_debug_http, url

Buttons: Add Block, Remove Block, Open .m5b, Download

send_data_debug_http ブロックを編集します。

2 番目のパラメータに variable タイプで data を指定します。

コード部は以下のものを指定します。

```
_g_send_data_debug_http({data})
```

UIFlow Block Maker beta

Namespace:
debug_http

Block Color:

Added

- init_debug_http
- send_data_debug_http**
- send_map_debug_http

Block Setting

Name:
send_data_debug_http
Tips: Name can only contain letters and numbers and "_".

Type:
☐ Value ☒ Execute

Parameter:

name: send_data	type: Label	Remove
name: data	type: Variable	Remove

Block Code.

```
_g_send_data_debug_http({data})
```

Tips: If you want to get the value of a parameter, you should use "\${parameter name}". And when the parameter type is label, it doesn't have value.
eg: If you have a parameter is named "a" and its type is "Number", you could use it like: 100 + \${a}

Block Code.

```

1 // Block __debug_http_init
2 var __debug_http_init_debug
3   "previousStatement": null,
4   "nextStatement": null,
5   "message0": "%1",
6   "args0": [
7     {
8       "type": "field",
9       "text": "init_c
10    }
11  ],
12  "message1": "%1 %2",
13  "args1": [
14    {
15      "type": "field",
16      "text": "url"
17    },
18    {
19      "type": "input_
20      "name": "url"
21    }
22  ],
23  "colour": "#ff8080"
24 };
25
26 window['Blockly'].Blocks['_
27   init: function() {
28     this.jsonInit(__det
29   }
30 };
31
32 window['Blockly'].Python['_
33   var url = Blockly.Pytho
34   return `import urequest
35   _g_url_debug_http=${url}
36   _g_success_color_debug_http
37   _g_fail_color_debug_http=0>
38   def _g_send_data_debug_http
39     try:
40       urequests.request(metho
41       rgb.setColorAll(_g_succ
42     except:
43       rgb.setColorAll(_g_fail
44   def _g_send_map_data_debug

```

Add Block Remove Block Open .m5b Download

send_map_debug_http ブロックを編集します。


2 番目のパラメータに variable タイプで map_data を指定します。

コード部は以下のものを指定します。

```
_g_send_map_data_debug_http({map_data})
```

UIFlow Block Maker beta

Namespace:
debug_http

Block Color:


Added

- init_debug_http
- send_data_debug_http
- send_map_debug_http**

Block Setting

Name:
Tips: Name can only contain letters and numbers and "_".
send_map_debug_http

Type:
☐ Value ☒ Execute


Parameter:

name: send_map_	type: Label	Remove
name: map_data	type: Variable	Remove

Block Code:

```
_g_send_map_data_debug_http(${map_data})
```

Tips: If you want to get the value of a parameter, you should use "\${parameter name}". And when the parameter type is label, it doesn't have value.
eg: If you have a parameter is named "a" and its type is "Number", you could use it like: **100 + \${a}**



```
1 // Block __debug_http_init
2 var __debug_http_init_debug
3   "previousStatement": null,
4   "nextStatement": null,
5   "message0": "%1",
6   "args0": [
7     {
8       "type": "field",
9       "text": "init_c"
10    }
11  ],
12  "message1": "%1 %2",
13  "args1": [
14    {
15      "type": "field",
16      "text": "url"
17    },
18    {
19      "type": "input",
20      "name": "url"
21    }
22  ],
23  "colour": "#ff8080"
24 };
25
26 window['Blockly'].Blocks['_
27   init: function() {
28     this.jsonInit(__det
29   }
30 };
31
32 window['Blockly'].Python['_
33   var url = Blockly.Pytho
34   return `import urequest
35   _g_url_debug_http=${url}
36   _g_success_color_debug_http
37   _g_fail_color_debug_http=0x
38   def _g_send_data_debug_http
39   try:
40     urequests.request(metho
41     rgb.setColorAll(_g_succ
42   except:
43     rgb.setColorAll(_g_fail
44   def _g_send_map_data_debug
```

Add Block Remove Block Open .m5b Download

download ボタンをクリックして、m5b ファイルを作成します。

Tips: If you want to get the value of a parameter, you should use "\${parameter name}". And when the parameter type is label, it doesn't have value.
eg: If you have a parameter is named "a" and its type is "Number", you could use it like: **100 + \${a}**




```
32 window['Blockly'].Python['__debug_http
33   var url = Blockly.Python.valueToCol
34   return `import urequests
35   _g_url_debug_http=${url}
36   _g_success_color_debug_http=0x000099
37   _g_fail_color_debug_http=0x990000
38   def _g_send_data_debug_http(data):
39   try:
40     urequests.request(method='GET', url=
41     rgb.setColorAll(_g_success_color_d
42   except:
43     rgb.setColorAll(_g_fail_color_debu
44   def _g_send_map_data_debug_http(map da
```

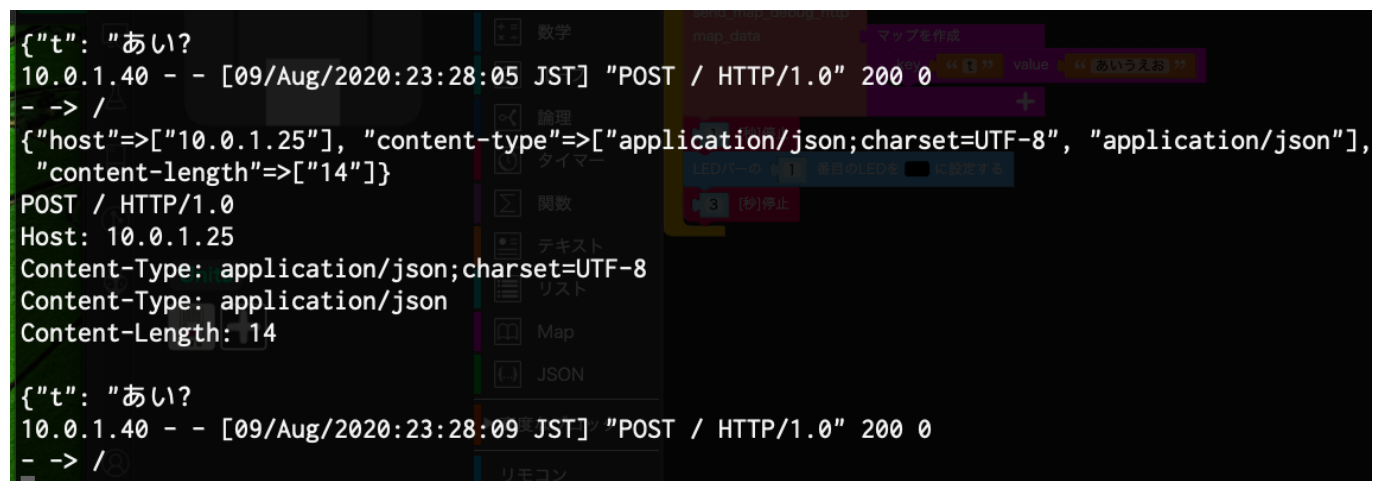
Add Block Remove Block Open .m5b **Download**

課題

現状のコードでは、map データに日本語を指定すると、http サーバー側では、データが欠けて表示されてしまいます。

修正方法は調査中です...

 プログラム3



```
{
  "t": "あい?"
}
10.0.1.40 - - [09/Aug/2020:23:28:05 JST] "POST / HTTP/1.0" 200 0
- -> /
{"host"=>["10.0.1.25"], "content-type"=>["application/json;charset=UTF-8", "application/json"],
"content-length"=>["14"]}
POST / HTTP/1.0
Host: 10.0.1.25
Content-Type: application/json;charset=UTF-8
Content-Type: application/json
Content-Length: 14

{"t": "あい?"
10.0.1.40 - - [09/Aug/2020:23:28:09 JST] "POST / HTTP/1.0" 200 0
- -> /
```