

```
1 #!/usr/bin/env python
2 # -*- coding: utf-8 -*-
3
4
5 import sys
6 import json
7 import datetime
8 from web3 import Web3
9
10 DEBUG = False #True
11
12
13 # Ethereum にアクセスするライブラリ
14
15 class Ethereum:
16     # コンストラクタ
17     def __init__(self, url, account, password, private_key):
18         print("Constructor of Ethereum")
19         self.url = url
20         self.account = account
21         self.password = password
22         self.private_key = private_key
23         self.contracts = {}
24
25         # Ethereum へ接続するクライアント作成
26         self.w3 = Web3(Web3.HTTPProvider(url))
27
28
29
30 # デストラクタ
31 def __del__(self):
32     print("Destructor of Ethereum")
33
34
35 # 実行するコントラクトの情報
36 def set_contract(self, addr, abi):
37     print("set_contract")
38     self.cnt_addr = addr
39     self.cnt_abi = abi
40     self.cnt = self.w3.eth.contract(address=self.w3.toChecksumAddress(addr), abi=abi)
```

```

41
42
43 # データ数取り出し
44 def get_data_number(self, channel_id, measurement):
45     print("get_data_number")
46
47     res = self.cnt.functions.get_data_num(channel_id, measurement).call()
48     return res
49
50 # データ取り出し
51 def get_data(self, channel_id, measurement, start=0, stop=10, source=None):
52     print("get_data")
53
54     res = self.cnt.functions.get_data(channel_id, measurement, start, stop).call()
55     if DEBUG : print("RES = ",res)
56     return res
57
58 # データ保存
59 def put_data(self, channel_id, measurement, data):
60     print("put_data")
61     """
62     data を timestamp とデータに分ける
63     """
64     d = json.loads(data)
65     if DEBUG:
66         print(d)
67
68     timestamp = d['timestamp']
69     if timestamp == None:
70         timestamp = datetime.datetime.utcnow().strftime('%Y%m%d%H%M%S')
71
72     ch_ac = self.w3.toChecksumAddress(self.account)
73     tx = self.cnt.functions.put_data(channel_id, measurement, timestamp, data).buildTransaction({'nonce':
self.w3.eth.getTransactionCount(ch_ac)})
74     s_tx = self.w3.eth.account.signTransaction(tx, self.private_key)
75     tx_hash = self.w3.eth.sendRawTransaction(s_tx.rawTransaction)
76     tx_receipt = self.w3.eth.wait_for_transaction_receipt(tx_hash)
77     if DEBUG:
78         print(tx_hash)
79         print(self.w3.eth.get_transaction(tx_hash))

```

```

80
81     # トランザクション情報を残しておく
82     self.tx = tx
83     self.s_tx = s_tx
84     self.tx_hash = tx_hash
85     self.tx_receipt = tx_receipt
86
87
88     # イベント処理コントラクトの登録
89     def set_event_contract(self, event, addr, abi):
90         if DEBUG : print("set_contract", event, addr,abi)
91
92         self.contracts[event] = {"address":addr, "abi":abi, "contract":self.w3.eth.contract(address=self.w3.toChecksumAddress(addr), abi=abi)}
93
94
95     # イベント処理コントラクトの実行
96     def put_event_contract(self, event, account, params):
97         if DEBUG : print("put_event_contract", event, account, params)
98         if event not in self.contracts:
99             print("no event:", event)
100            return None
101        if account == None:
102            print("use self.account:", self.account)
103            account = self.account
104        cnt = self.contracts[event]['contract']
105        ch_ac = self.w3.toChecksumAddress(self.account)
106        keys = []
107        values = []
108        for k in params:
109            keys.append(k)
110            if k == 'account':
111                #values.append(self.w3.toChecksumAddress(params[k]))
112                values.append(params[k].lower())
113                account = params[k].lower()
114            else:
115                values.append(params[k])
116        tx = cnt.functions.put(self.w3.toChecksumAddress(account), keys, values).buildTransaction({'nonce':
self.w3.eth.getTransactionCount(ch_ac)})
117        s_tx = self.w3.eth.account.signTransaction(tx, self.private_key)
118        tx_hash = self.w3.eth.sendRawTransaction(s_tx.rawTransaction)

```

```
119         tx_receipt = self.w3.eth.wait_for_transaction_receipt(tx_hash)
120
121     # イベント処理コントラクトの情報取得
122     def get_event_contract(self, event, params):
123         if DEBUG : print("get_event_contract", event, params)
124         if event not in self.contracts:
125             print("no event:", event)
126             return None
127         cnt = self.contracts[event]['contract']
128         keys = []
129         values = []
130         for k in params:
131             keys.append(k)
132             if k == 'account':
133                 #values.append(self.w3.toChecksumAddress(params[k]))
134                 values.append(params[k].lower())
135             else:
136                 values.append(params[k])
137         res = cnt.functions.get(keys, values).call()
138         if DEBUG : print("get_event_contract:res:",res)
139         return res
```