



10時より開始します

Cisco Community Expert Series Community Live

ネットワーク機器管理の 自動化について

飯山 克志 (Katsushi Iiyama)
Top Out Human Capital 株式会社
September 29th, 2021



ご参加ありがとうございます。

本日の資料はこちらからダウンロードいただけます。

<https://community.cisco.com/t5/-/-/ec-p/4462945>



今すぐ登録

資料のダウンロード

[エキスパートスピーカー紹介]



飯山 克志 (Katsushi Iiyama)
Top Out Human Capital 株式会社 CTO

CCIE #2023 / CCDP / CCSI
DevNet 500 / DevNet Specialist
Cisco Champion 2019, 2020, 2021

<https://www.topout.co.jp/>



音声ブロードキャストについて

[音声ブロードキャスト (Audio Broadcast)] ウィンドウが自動的に表示され、コンピュータのスピーカーから音声がかかります。

[音声ブロードキャスト (Audio Broadcast)] ウィンドウが表示されない場合は、[通話(Communicate)] メニューから [音声ブロードキャスト (Audio Broadcast)] を選択します。

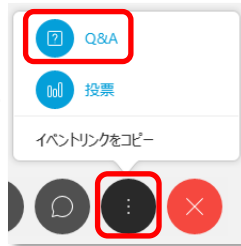
イベントが開始されると自動的に音声が流れ始めます。

音声接続に関する詳細はこちらをご参照ください。

解決しない場合は、Q&A ウィンドウより

[すべてのパネリスト (All Panelists)] 宛にお知らせください。

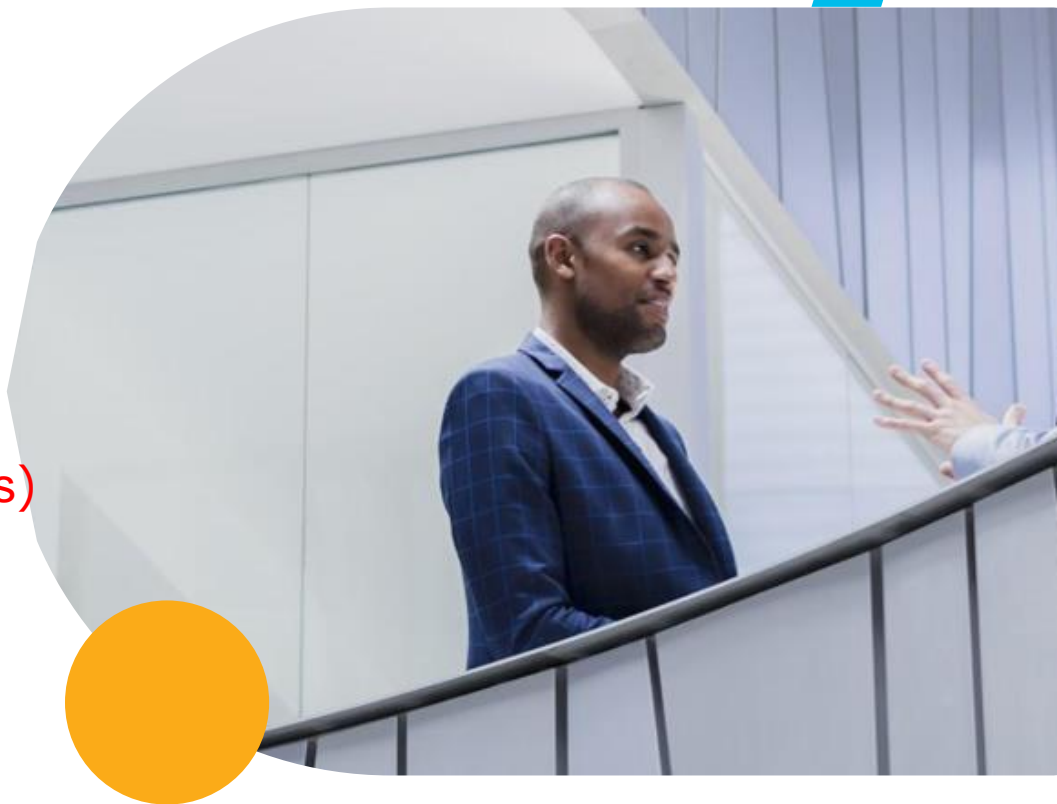
<https://community.cisco.com/t5/-/-/ta-p/3129991>



※ Q&A ウィンドウが画面右側に見つからない場合はここから表示

ご質問方法

Community Live中のご質問は、
画面右側の Q&A ウィンドウから
すべてのパネリスト (All Panelists)
宛に送信してください。



エキスパートスピーカー



飯山 克志 (Katsushi Iiyama)

Top Out Human Capital株式会社 CTO
CCIE #2023 (1996~) / CCDP / CCSI
DevNet 500 / DevNet 2020 / DevNet Specialist
Cisco Champion 2019, 2020, 2021





Top Out Human Capital 株式会社

会社紹介とご案内

日本で唯一がここにある!

- Cisco 認定ラーニングパートナー (日本で唯一、Collaboration, Security関連コースを実施)
- NetApp 認定ラーニングパートナー (日本で唯一の認定ラーニングパートナー)
- EC-Council 認定トレーニングセンター (Cybersecurity、全てのEC-Council認定コースを唯一実施)
- CompTIAトレーニングパートナー (日本で唯一、Cybersecurity資格に対応する全コースを実施)
- Pythonエンジニア育成推進協会認定スクール (Pythonによる運用&監視の自動化コースを提供)
- Citrix認定ラーニングセンター (日本で唯一、Citrix SD-WANコースを実施)
- HPE Aruba認定トレーニングセンター (国内唯一、Wireless LAN、Switch、Clear Pass)
- DevOps 認定教育パートナー (独占的販売代理店契約)
- F5 認定トレーニングセンター (BIG-IPシリーズ)
- Acronis #CyberFit エデュケーションパートナー (国内唯一、サイバープロテクション)
- Huawei認定ラーニングパートナー (国内唯一、5G、Wireless LAN 他)
- IoT関連トレーニング (IoTハッキング・セキュリティ、IoT実践ハンズオン)
- Alibaba Cloud ハンズオントレーニング (1日で学べるハンズオン)
- Gigamon 認定トレーニング
- RedHat 認定トレーニング
- ビジネストレーニング (ITILなど)



AUTHORIZED TRAINING CENTER  Alibaba Cloud



AUTHORIZED TRAINING CENTER





【Top Out Human Capital 株式会社共催】

Cisco Community Expert Series Community Live

ネットワーク機器管理の自動化について

Katsushi Iiyama
Top Out Human Capital 株式会社
September 29th, 2021



本日の目次

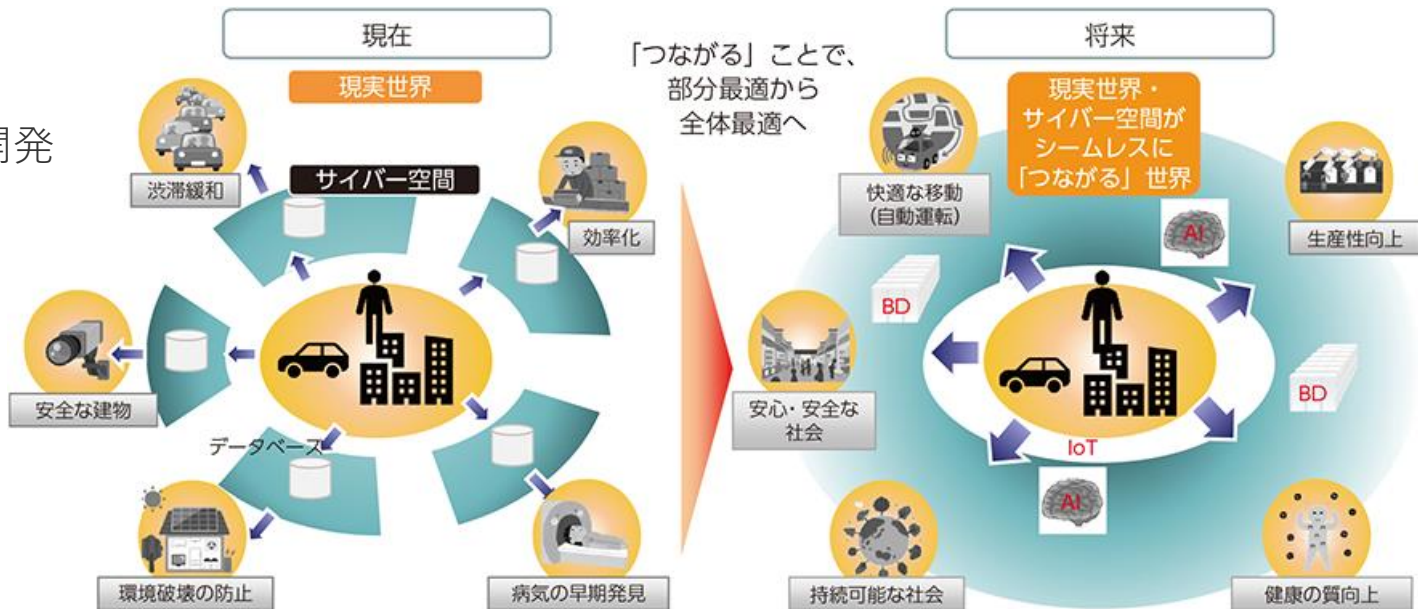
- ◆ はじめに
- ◆ まずは、簡単な監視をPythonで実施してみる
- ◆ 受け取ったパラメータシートをPythonで処理してみる
- ◆ ネットワーク機器管理・監視をPythonで実施してみる
- ◆ サーバ管理・監視をPythonで実施してみる
- ◆ 担当者へPythonで通知してみる
- ◆ さいごに

はじめに



時代の背景

- ◆ DX (Digital Transformation)
- ◆ IoT
- ◆ 自動化
- ◆ アジャイル開発
- ◆ AI



出典：総務省ホームページ

(<https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/h30/html/nd102200.html>)

投票質問 1

管理における自動化について、いまどのような状況か教えてください

- A) ほとんどの管理が自動化されている
- B) 一部で管理が自動化されている
- C) 今、自動化を取り入れだしたところ
- D) これから自動化を取り入れないといけない
- E) 自動化の予定はない。興味がない

これまでとこれからの比較

◆ これまで

- Teratermマクロ
- コマンドプロンプト
- ターミナル
- 処理が途切れる(excel、teraterm、pdf作成、メール)
- NWとサーバ監視が別々

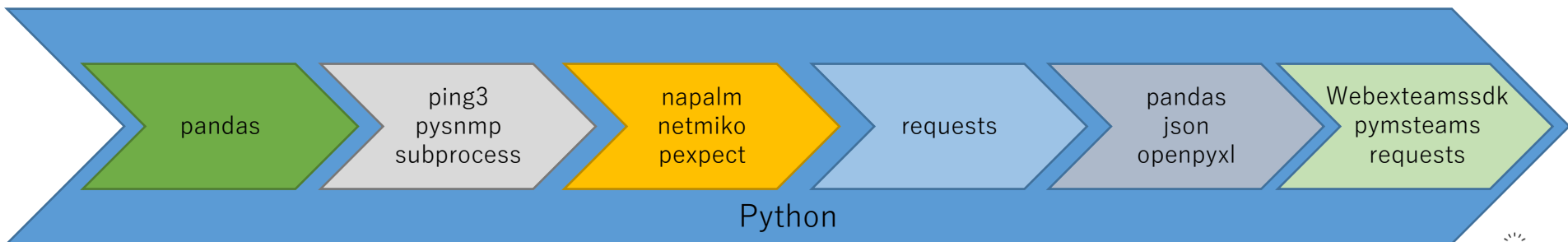


◆ これから

- 全ての処理を一元的に実施
- オペレーターを介さずに自動で処理を実施
- 問題が起きた内容を管理者がリモートですぐに把握できる
- NW機器だけでなくサーバ機器についても一緒に確認することが可能



図にすると・・・



※表示しているライブラリは一例、ベストなものを使用する事



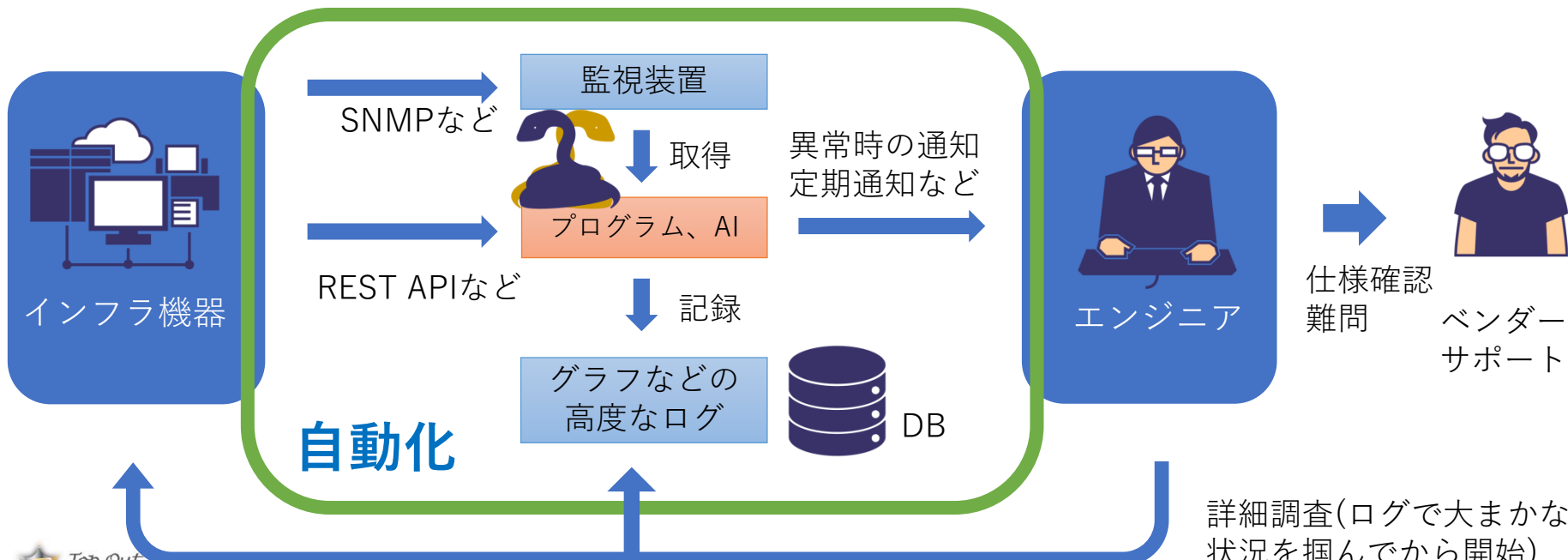
投票質問 2

ネットワーク機器管理の自動化においては Pythonが非常に有用です。Pythonの利用状況を教えてください。

- A) 自分でプログラムを作って、よく使っている
- B) 自分でプログラムを作ったことがある
- C) これから使わなければいけない
- D) 初めて知った
- E) 自分は使わないだろう

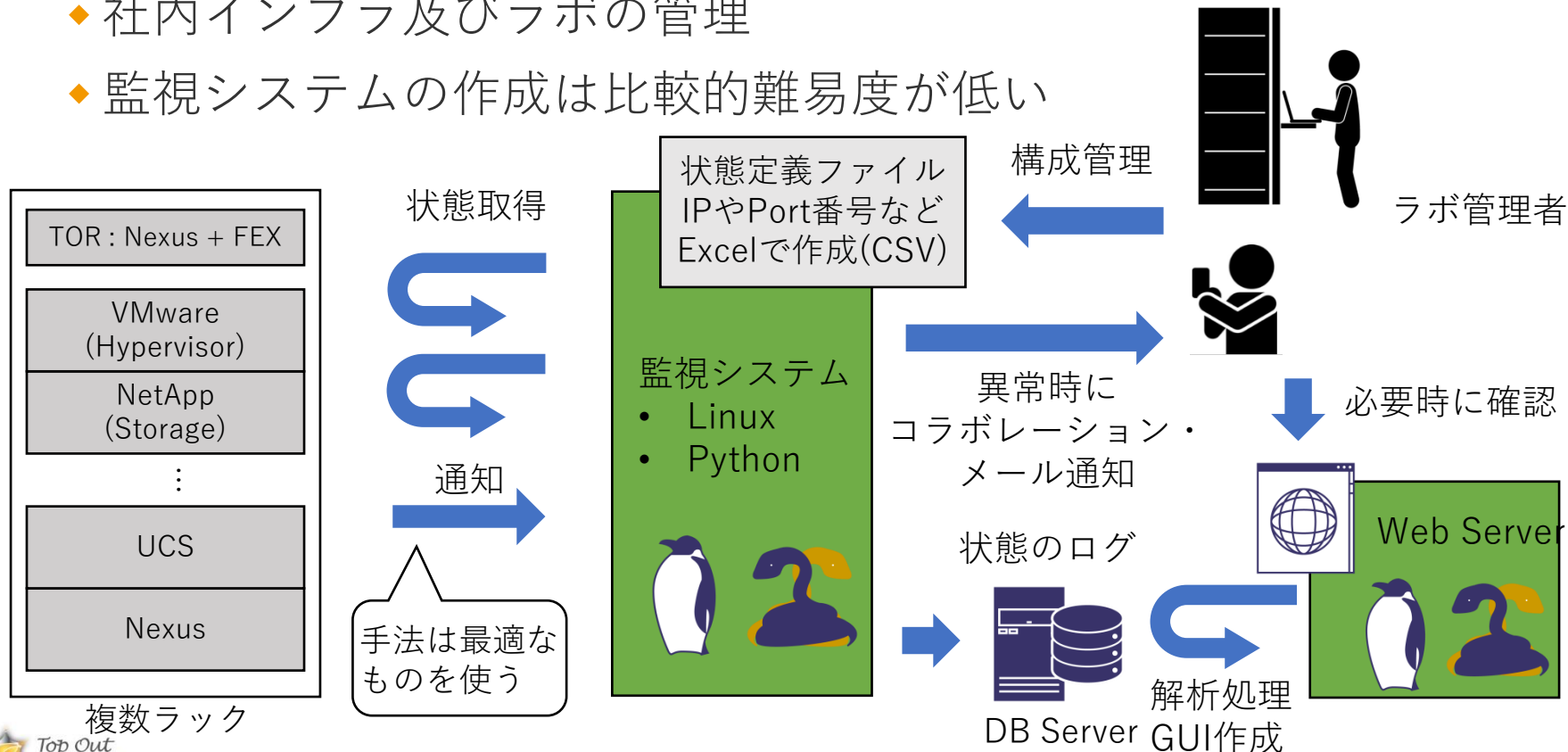
新しい運用と管理

- ◆ 単純作業は自動化する。オペレーターを置き換え
- ◆ 欲しい情報(ログ)をプログラムで作成する



実環境での管理の利用例

- ◆ 社内インフラ及びラボの管理
- ◆ 監視システムの作成は比較的難易度が低い



実行するコマンド/プログラム/モジュールを選ぶときに注意する事

- ◆ どのコマンド/プログラム/モジュールを使うか？
 - OSの種類 / Version
 - 移植性
 - 戻り値の扱い (文字の固まり or 変数で細かく扱える)
 - 連続性 (複数の作業を一つのコマンドで実行できるか)
 - 多種多様なファイルを扱える
 - 学習が簡単か？
 - 誰でも使えるか？



戻り値の例

show running-configの結果

```
interface Loopback99
 ip address 99.99.99.99 255.255.255.255
!
interface VirtualPortGroup0
 ip unnumbered GigabitEthernet1
!
interface GigabitEthernet1
 description Internal
 ip address 192.168.0.1 255.255.255.0
 ip nat inside
 negotiation auto
!
interface GigabitEthernet2
 ip address 1.0.0.1 255.0.0.0
 negotiation auto
!
```



ipconfigの結果

```
Windows IP 構成

ホスト名 . . . . . : w2k12
プライマリ DNS サフィックス . . . . . : demo.netapp.com
ノード タイプ . . . . . : ハイブリッド
IP ルーティング有効 . . . . . : いいえ
WINS プロキシ有効 . . . . . : いいえ
DNS サフィックス検索一覧 . . . . . : demo.netapp.com

イーサネット アダプター Lab Internal:

接続固有の DNS サフィックス . . . . . :
説明 . . . . . : Intel(R) 82574L Gigabit Network Connection #2
物理アドレス . . . . . : 00-50-56-A8-73-DD
DHCP 有効 . . . . . : いいえ
自動構成有効 . . . . . : (はい)
IPv4 アドレス . . . . . : 192.168.0.71(優先)
サブネット マスク . . . . . : 255.255.255.0
デフォルト ゲートウェイ . . . . . :
NetBIOS over TCP/IP . . . . . : 有効

イーサネット アダプター External:

接続固有の DNS サフィックス . . . . . :
説明 . . . . . : Intel(R) 82574L Gigabit Network Connection
物理アドレス . . . . . : 00-0C-29-B0-0B-24
DHCP 有効 . . . . . : いいえ
自動構成有効 . . . . . : (はい)
IPv4 アドレス . . . . . : 172.16.100.190(優先)
サブネット マスク . . . . . : 255.255.255.0
デフォルト ゲートウェイ . . . . . : 172.16.100.250
```

snmpwalkの結果

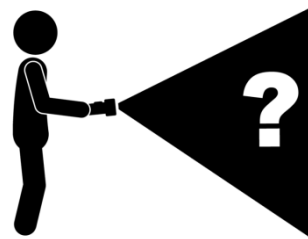
```
SNMPv2-SMI::mib-2.47.1.1.1.1.7.1 = STRING: "Chassis"
SNMPv2-SMI::mib-2.47.1.1.1.1.7.10 = STRING: "module R0"
SNMPv2-SMI::mib-2.47.1.1.1.1.7.11 = STRING: "cpu R0/0"
SNMPv2-SMI::mib-2.47.1.1.1.1.7.12 = STRING: "NME R0"
SNMPv2-SMI::mib-2.47.1.1.1.1.7.30 = STRING: "module F0"
SNMPv2-SMI::mib-2.47.1.1.1.1.7.51 = STRING: "GigabitEthernet1"
SNMPv2-SMI::mib-2.47.1.1.1.1.7.52 = STRING: "GigabitEthernet2"
SNMPv2-SMI::mib-2.47.1.1.1.1.7.53 = STRING: "GigabitEthernet3"
```

戻り値の例 (napalmでget_interfacesを使用した例)

```
{'GigabitEthernet1':  
  {'description': 'Internal',  
    'is_enabled': True,  
    'is_up': True,  
    'last_flapped': -1.0,  
    'mac_address': '00:50:56:BD:0A:1C',  
    'mtu': 1500,  
    'speed': 1000},  
'GigabitEthernet2':  
  {'description': 'External',  
    'is_enabled': True,  
    'is_up': True,  
    'last_flapped': -1.0,  
    'mac_address': '00:50:56:BD:FC:D8',  
    'mtu': 1500,  
    'speed': 1000},  
'loopback99':  
  {'description': 'Loopback',  
    'is_enabled': True,  
    'is_up': True,  
    'last_flapped': -1.0,  
    'mac_address': '',  
    'mtu': 1514,  
    'speed': 8000}}
```

```
ROUTER#sh int  
GigabitEthernet1 is up, line protocol is up  
  Hardware is CSR vNIC, address is 0050.56bd.0a1c (bia 0050.56bd.0a1c)  
  Description: Internal  
  Internet address is 192.168.0.1/24  
  MTU 1500 bytes, BW 1000000 Kbit/sec, DLY 10 usec,  
    reliability 255/255, txload 1/255, rxload 1/255  
  Encapsulation ARPA, loopback not set  
  Keepalive set (10 sec)  
  Auto Duplex, Auto Speed, link type is auto, media type is RJ45  
  output flow-control is unsupported, input flow-control is unsupported  
  ARP type: ARPA, ARP Timeout 04:00:00  
  Last input 00:00:21, output 00:00:21, output hang never  
  Last clearing of "show interface" counters never  
  Input queue: 0/375/0/0 (size/max/drops/flushes); Total output drops: 0  
  Queueing strategy: fifo  
  Output queue: 0/40 (size/max)  
  5 minute input rate 3000 bits/sec, 4 packets/sec  
  5 minute output rate 1000 bits/sec, 3 packets/sec  
    17238 packets input, 1647950 bytes, 0 no buffer  
    Received 0 broadcasts (0 IP multicasts)  
    0 runts, 0 giants, 0 throttles  
    0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored
```

まずは、簡単な監視を
Pythonで実施してみる



Top Out
Human Capital

デバイスの情報を確認する

- ◆ ホスト名、IPアドレス、MACアドレスの確認
 - これまでは、ipconfig / ifconfig
- ◆ 疎通確認
 - これまでは、ping、tracert



IPアドレス確認 (ホスト名取得、ローカルIPアドレス取得)

```
import socket  
# ホスト名を取得  
print(socket.gethostname())  
# Win10-N12  
  
# ローカルIPアドレスを取得  
print(socket.gethostbyname(socket.gethostname()))  
# 192.168.1.1
```



IPアドレス確認 (デフォルトゲートウェイ取得)

```
import netifaces
#Default Gatewayの表示
print(netifaces.gateways())
#{2: [(
    '192.168.1.1',
    '{0D61DA8C-6081-4A86-9CAB-E53126A2404D}',
    True)],
'default': { 2: (
    '192.168.1.1',
    '{0D61DA8C-6081-4A86-9CAB-E53126A2404D}')}}
print(netifaces.gateways()['default'][netifaces.AF_INET][0])
```


全てのインターフェイスのMAC/IPV4/IPV6

```
import netifaces

for iface_name in netifaces.interfaces():
    iface_data = netifaces.ifaddresses(iface_name)
    print ('Interface: %s' % (iface_name))

    if iface_data.get(netifaces.AF_LINK) is not None :
        print ('MAC Address',iface_data.get(netifaces.AF_LINK))

    if iface_data.get(netifaces.AF_INET) is not None :
        print ('IPv4 Address',iface_data.get(netifaces.AF_INET))

    if iface_data.get(netifaces.AF_INET6) is not None :
        print ('IPv6 Address',iface_data.get(netifaces.AF_INET6))
```

リモートホスト名からIPアドレス取得

```
# -*- coding:utf-8 -*-  
import socket  
  
# ポート番号とプロトコル名からサービス名を取得  
def get_remote_ip(remote_host):  
    print('Remote host name:', remote_host)  
    print('Remote IP:', socket.gethostbyname(remote_host))  
  
get_remote_ip(remote_host='www.yahoo.co.jp')  
''''''  
Remote host name: www.yahoo.co.jp  
Remote IP: 182.22.24.252  
''''''
```



Pythonでpingを実行する方法

ライブラリ名	subprocess	pings / pyping	ping3
特徴/メリット	OSが持っている機能を利用	シンプルに利用することが可能	DEBUGや多数の引数を扱えるなど、非常に多彩
デメリット	OS依存のため、OS毎に引数が異なる	必要最低限のパラメータしか使えない	
	戻り値を解釈しないといけない		
管理者権限	不要	必要	必要

競技種目¹⁶ ping疎通確認

ISNT IT SET TO REJECT ICMP?



pingsを使用したpython pingの例

```
import pings

host = "www.google.com"

p = pings.Ping()

res = p.ping(host, times=3) # googleを監視, 3回pingを飛ばす
print(res)

if not res.is_reached():
    print("届きませんでした")
else:
    res.print_messages()
```

pingsを使用したpython pingの結果

```
PS > python .\mypings.py
```

```
{'max_rtt': 3.9098189999999981, 'min_rtt': 3.79433099999999286, 'avg_rtt':  
3.83854299999999716, 'packet_lost': None, 'ret_code': 0, 'packet_size': 47,  
'timeout': 1000, 'dest': 'google.com', 'dest_ip': '172.217.24.142'}
```

```
PING www.google.com (172.217.24.142): 55 data bytes
```

```
47 bytes from 172.217.24.142: icmp_seq=0 ttl=116 time=3.811 ms
```

```
47 bytes from 172.217.24.142: icmp_seq=1 ttl=116 time=3.910 ms
```

```
47 bytes from 172.217.24.142: icmp_seq=2 ttl=116 time=3.794 ms
```

```
--- google.com ping statistics ---
```

```
3 packets transmitted, 3 packets received, 0.0% packet loss
```

```
round-trip min/avg/max = 3.794/3.839/3.910 ms
```

```
{'max_rtt': 0.0, 'min_rtt': 0.0, 'avg_rtt': 0.0, 'packet_lost': None, 'ret_code': 1,  
'packet_size': 0, 'timeout': 1000, 'dest': '192.168.0.3', 'dest_ip': '192.168.0.3'}
```

pypingを使用したpython pingの例 / 結果

```
import pyping
```

```
host = "www.google.com" #確認したいアドレス
```

```
RPing = pyping.ping(host)
```

```
print("Dest:"+str(RPing.destination)) #送り先
```

```
print("DestIP:"+str(RPing.destination_ip)) #送り先IP
```

```
print("max:"+str(RPing.max_rtt)+"ms") #最大往復秒
```

```
print("min:"+str(RPing.min_rtt)+"ms") #最小往復秒
```

```
print("avg:"+str(RPing.avg_rtt)+"ms") #平均往復秒
```

```
print("code:"+str(RPing.ret_code)) #戻り値
```

結果

```
Dest:www.google.com
```

```
DestIP:172.217.175.36
```

```
max:59.223ms
```

```
min:31.438ms
```

```
avg:41.376ms
```

```
code:0 # 0:成功
```

ping3を使用したpython pingの例

```
import ping3
```

```
ping3.DEBUG = True
```

```
ret = ping3.ping("www.google.com")
```

```
print(ret)
```

```
ret = ping3.verbose_ping("www.google.com") # Ping 4 times in a row.
```

```
ret = ping3.verbose_ping("www.google.com", count=3, src_addr='172.16.100.190', ttl=5)
```



ping3を使用したpython pingの結果 [DEBUG有]

```
[DEBUG] Ping3 Version: 3.0.2
[DEBUG] LOGGER: <Logger ping3 (DEBUG)>
[DEBUG] Function called: ping('www.google.com')
[DEBUG] Function called: send_one_ping({'sock': <socket.socket fd=432, family=AddressFamily.AF_INET, type=SocketKind.SOCK_RAW, proto=1>, 'dest_addr': 'www.google.com', 'icmp_id': 49146, 'seq': 0, 'size': 56})
[DEBUG] Destination address: 'www.google.com'
[DEBUG] Destination IP address: 142.250.199.100
[DEBUG] Sent ICMP header: {'type': 8, 'code': 0, 'checksum': 37396, 'id': 49146, 'seq': 0}
[DEBUG] Sent ICMP payload: b'A\xd8T\x1c\x9a\xf6\xd5eQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ'
[DEBUG] Function returned: send_one_ping -> None
[DEBUG] Function called: receive_one_ping({'sock': <socket.socket fd=432, family=AddressFamily.AF_INET, type=SocketKind.SOCK_RAW, proto=1, laddr=('0.0.0.0', 0)>, 'icmp_id': 49146, 'seq': 0, 'timeout': 4})
[DEBUG] Timeout time: Sun Sep 26 22:15:27 2021 (1632662127.8723981)
[DEBUG] Timeout left: 4.00s
[DEBUG] Received time: Sun Sep 26 22:15:23 2021 (1632662123.8723981)
[DEBUG] Received IP header: {'version': 69, 'tos': 96, 'len': 84, 'id': 0, 'flags': 0, 'ttl': 116, 'protocol': 1, 'checksum': 57115, 'src_addr': '142.250.199.100', 'dest_addr': '172.16.100.190'}
[DEBUG] Received ICMP header: {'type': 0, 'code': 0, 'checksum': 39444, 'id': 49146, 'seq': 0}
[DEBUG] Received ICMP payload: b'A\xd8T\x1c\x9a\xf6\xd5eQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ'
[DEBUG] Received sent time: Sun Sep 26 22:15:23 2021 (1632662123.8567746)
[DEBUG] Function returned: receive_one_ping -> 0.01562356948852539
[DEBUG] Function returned: ping -> 0.01562356948852539
```


受け取ったパラメータシートを Pythonで処理してみる



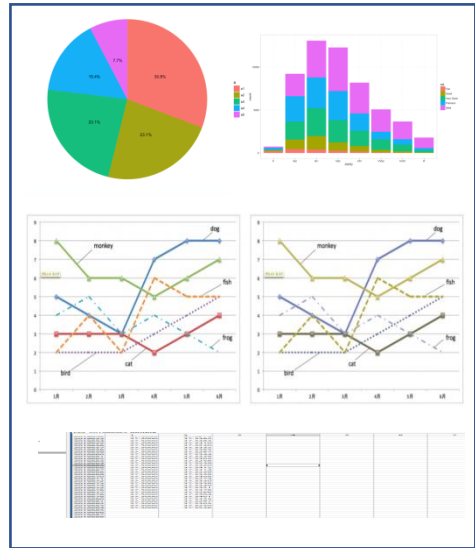
Pythonでのデータ処理 (入力 / 出力)

Operation	Module	Settlement	Account	Security	Template	Asset	Type	Side	Interest	Account	Interest	Quantity	Price	Commission	Fee	Net	Settlement	Settlement	
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout01	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout02	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout03	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout04	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout05	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout06	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout07	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout08	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout09	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout10	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout11	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout12	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout13	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout14	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout15	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout16	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout17	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout18	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout19	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout20	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout21	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout22	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout23	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout24	18077	21400.2	21.34	1	1	18077	2017
1/2/2019	1/2/2019	1/2/2019	30000	TEST	0000	DUMMY	DUMMY	Short	21.34	Settle	Callout	Callout25	18077	21400.2	21.34	1	1	18077	2017



監視システム

- Linux/Win
- Python



パラメータシート(例)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI																																																														
1	顧客名																ハードウェア構成										作成日:																																																																						
2																											作成者:																																																																						
3																											更新日:																																																																						
4																											更新者:																																																																						
5	ハードウェア情報																																																																																																
6	■サーバー本体情報 ■ ご記入いただきたい項目																																																																																																
7	<table border="1"> <thead> <tr> <th>No</th> <th>項目</th> <th>値</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>製品名</td> <td></td> </tr> <tr> <td>2</td> <td>サーバスタグ</td> <td></td> </tr> </tbody> </table>																																No	項目	値	1	製品名		2	サーバスタグ																																																									
No	項目	値																																																																																															
1	製品名																																																																																																
2	サーバスタグ																																																																																																
8																																																																																																	
9																																																																																																	
10																																																																																																	
11	■RAID構成情報																																																																																																
12	<table border="1"> <thead> <tr> <th>No</th> <th>設定項目</th> <th>設定値</th> <th>備考</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>RAIDレベル</td> <td></td> <td></td> </tr> <tr> <td>2</td> <td>アレイメンバードライブ</td> <td></td> <td></td> </tr> <tr> <td>3</td> <td>実効容量</td> <td></td> <td></td> </tr> <tr> <td>4</td> <td>ホットスワップ</td> <td></td> <td></td> </tr> </tbody> </table>																																No	設定項目	設定値	備考	1	RAIDレベル			2	アレイメンバードライブ			3	実効容量			4	ホットスワップ																																															
No	設定項目	設定値	備考																																																																																														
1	RAIDレベル																																																																																																
2	アレイメンバードライブ																																																																																																
3	実効容量																																																																																																
4	ホットスワップ																																																																																																
13																																																																																																	
14																																																																																																	
15																																																																																																	
16																																																																																																	
17																																																																																																	
18																																																																																																	
19	iLo (管理ポート) 設定																																																																																																
20	■iDRAC Configuration																																																																																																
21	<table border="1"> <thead> <tr> <th>No</th> <th>設定項目</th> <th>設定値</th> <th>デフォルト値</th> <th>備考</th> </tr> </thead> <tbody> <tr> <td>1</td> <td colspan="4">Ethernet Configuration</td> </tr> <tr> <td>2</td> <td>iLoHostname</td> <td></td> <td></td> <td></td> </tr> <tr> <td>3</td> <td>管理者ログインユーザー名</td> <td></td> <td>Administrator</td> <td></td> </tr> <tr> <td>4</td> <td>パスワード</td> <td></td> <td></td> <td></td> </tr> <tr> <td>5</td> <td colspan="4">IPv4</td> </tr> <tr> <td>6</td> <td>Enable DHCP</td> <td></td> <td>Enabled</td> <td></td> </tr> <tr> <td>7</td> <td>Static IPv4 address</td> <td></td> <td>N/A</td> <td></td> </tr> <tr> <td>8</td> <td>Default Gateway</td> <td></td> <td>N/A</td> <td></td> </tr> <tr> <td>9</td> <td>Network mask</td> <td></td> <td>N/A</td> <td></td> </tr> <tr> <td>10</td> <td colspan="4">DNS</td> </tr> <tr> <td>11</td> <td>Preferred DNS Server</td> <td></td> <td>N/A</td> <td></td> </tr> <tr> <td>12</td> <td>Alternate DNS Server</td> <td></td> <td>N/A</td> <td></td> </tr> </tbody> </table>																																No	設定項目	設定値	デフォルト値	備考	1	Ethernet Configuration				2	iLoHostname				3	管理者ログインユーザー名		Administrator		4	パスワード				5	IPv4				6	Enable DHCP		Enabled		7	Static IPv4 address		N/A		8	Default Gateway		N/A		9	Network mask		N/A		10	DNS				11	Preferred DNS Server		N/A		12	Alternate DNS Server		N/A	
No	設定項目	設定値	デフォルト値	備考																																																																																													
1	Ethernet Configuration																																																																																																
2	iLoHostname																																																																																																
3	管理者ログインユーザー名		Administrator																																																																																														
4	パスワード																																																																																																
5	IPv4																																																																																																
6	Enable DHCP		Enabled																																																																																														
7	Static IPv4 address		N/A																																																																																														
8	Default Gateway		N/A																																																																																														
9	Network mask		N/A																																																																																														
10	DNS																																																																																																
11	Preferred DNS Server		N/A																																																																																														
12	Alternate DNS Server		N/A																																																																																														
22																																																																																																	
23																																																																																																	
24																																																																																																	
25																																																																																																	
26																																																																																																	
27																																																																																																	
28																																																																																																	
29																																																																																																	
30																																																																																																	
31																																																																																																	
32																																																																																																	
33																																																																																																	

論理ディスクを複数構成する場合は、お手数ですが項目部分を複製いただけますでしょうか



pandasで各種パラメータファイルの読み込み

	読み込み	
CSV	<code>pandas.read_csv</code>	<code>pandas.DataFrame.to_csv</code>
TSV	<code>pandas.read_table</code>	<code>pandas.DataFrame.to_csv</code> # (区切り文字 <code>sep='/t'</code>)
Excel	<code>pandas.read_excel</code>	<code>pandas.DataFrame.to_excel()</code>
JSON	<code>pandas.read_json</code>	<code>pandas.DataFrame.json()</code>

基本的な読み込みと書出し

```
import pandas
```

```
df = pandas .read_csv('{{ 読み込むCSVファイルのパス }}')
```

```
df.to_csv('{{ 保存先のCSVファイルのパス }}')
```

各種形式に合わせて関数を変更する

データへのアクセス (インデックス有)

行 \ 列	A(index)	B(0)	C(1)
1(column)		Age	
2(0)	Alice	24	
3(1)	Bob	60	
4(2)	Charie	18	

```
import pandas
```

```
df = pandas.read_csv('{{ 読み込むCSVファイルのパス }}')
```

```
print( df.iat[1,0]) # 行, 列  
# 60
```

```
df.iat[1,0] = 12
```

```
print( df.at["Bob", "Age"])  
# 60
```

```
df.at["Bob", "Age"] = 12
```

```
print(df.iloc[1,0])  
# 60
```

```
df.iloc[1,0] = 12
```

データへのアクセス (インデックス無)

行 \ 列	A(0)	B(1)	C(2)
1(0)	24		
2(1)	60		
3(2)	18		
4(3)	33		

```
import pandas
```

```
df = pandas .read_csv('{{ 読み込むCSVファイルのパス }}')
```

```
print( df.iat[1,0]) # 行, 列
```

```
# 60
```

```
df.iat[1,0] = 12
```

```
print(df.iloc[1,0])
```

```
# 60
```

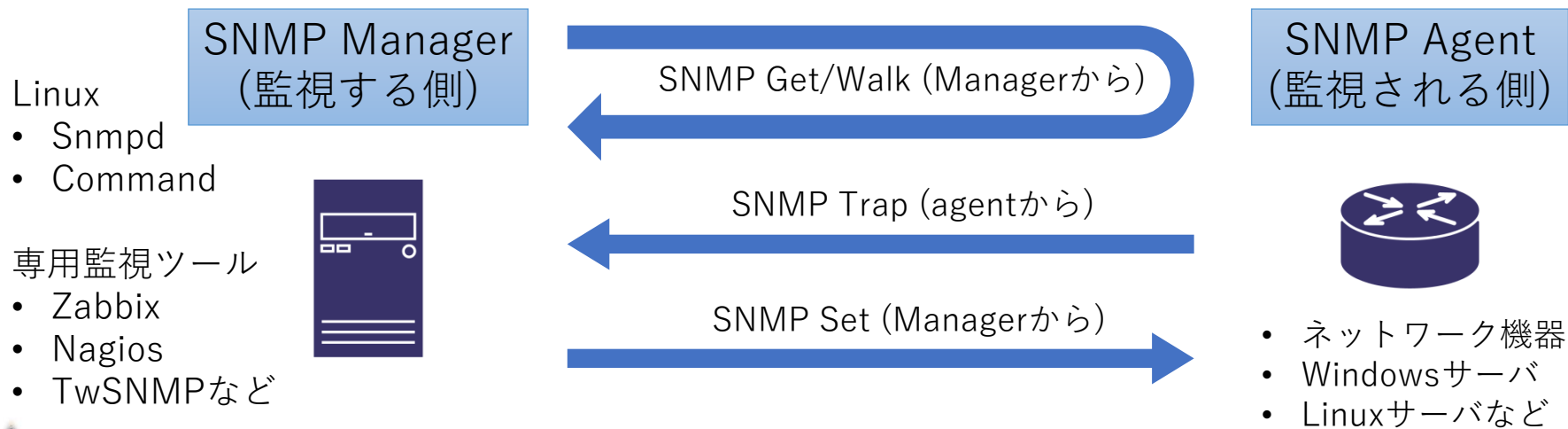
```
df.iloc[1,0] = 12
```

ネットワーク機器管理・監視を Pythonで実施してみる



SNMP

- ◆ 機器の状態を取得(通知)したり設定するためのプロトコル
- ◆ 長所: 普及している
- ◆ 短所: 情報の取得(SNMP Get/Trap)以外はほとんど使われない



snmpwalk

- ◆ WindowsやLinuxではsnmpwalkコマンドを使用して、状態を入手

```
% snmpwalk -v2c -c public 192.168.0.1 1.3.6.1.2.1.2.2.1.10.1
```

```
!--- SNMP 問い合わせ
```

```
IF-MIB::ifInOctets.1 = Counter32: 139137
```

```
!--- レスポンス
```



snmpwalkをPythonで実行する

- ◆ Cisco CSR1000vの1分間のCPU使用率の平均を確認

```
import subprocess
command = 'snmpwalk -v2c -c public 192.168.0.1
1.3.6.1.2.1.2.2.1.10.1'
output = subprocess.check_output(command.split(' ')).decode()
print(output)
```

```
PS:\> python lab2-5.py
```

```
SNMPv2-SMI::enterprises.9.9.109.1.1.1.1.7.2 = Gauge32: 3
```

各インタフェースの受信バイト総数をpysnmpで取得

```
from pysnmp.hlapi import *  
  
def printGetRequest(ip, community, mib):  
    errorIndication, errorStatus, errorIndex, varBinds = next(  
        getCmd(SnmpEngine(),  
              CommunityData(community),  
              UdpTransportTarget((ip, 161)),  
              ContextData(),  
              ObjectType(ObjectIdentity(mib))))
```

```
for varBind in varBinds:
```

```
    print(' = '.join([x.prettyPrint() for x in varBind]))
```

```
printGetRequest(ip = "192.168.0.1", community = "public", mib = "1.3.6.1.2.1.2.2.1.10.1")
```

```
printGetRequest(ip = "192.168.0.1", community = "public", mib = "1.3.6.1.2.1.2.2.1.10.2")
```

結果(例)

SNMPv2-SMI::mib-2.2.2.1.10.1 = 1398137

SNMPv2-SMI::mib-2.2.2.1.10.2 = 349699

ネットワーク機器向けのライブラリ比較

	Expect Pexpect	Paramiko	Netmiko	Napalm	Requests
コンソール	✓				
ssh	✓	✓			
telnet	✓				
ネットワーク機器			✓	✓	
Webサーバ					✓
REST API					✓

Pexpect の利用方法(1)

- ◆ Spawnしてセッションのインスタンスを作る
- ◆ インスタンスのexpectメソッドで待ちに入り、期待する入力 cameたら解除
- ◆ インスタンスのsendlineメソッドで相手に文字列(キー入力相当)を送る

Pythonの
プログラム

```
import pexpect
ssh = 'ssh admin@192.168.0.1'
p = pexpect.spawn(ssh)

p.expect('Password:')
p.sendline('python123')
p.expect(r'.+[>#]')

p.sendline('enable')
p.expect('Password:')
p.sendline('python123')
p.expect(r'.+[>#]')
```

\$ ssh admin@192.168.0.1

Password:
MY-ROUTER-X>

MY-ROUTER-X>enable
Password:
MY-ROUTER-X#

対応する
コンソール

Pexpect の利用方法(2)

- ◆ 同時出力行数を無制限にしておく (terminal length 0)
- ◆ リモートからの入力(コンソール出力)は、インスタンスのafter変数で取得

```
p.sendline('terminal length 0')  
p.expect(r'.+[>#]')
```

```
p.sendline('show clock')  
p.expect(r'.+[>#]')  
output = p.after  
print(output.decode('utf-8'))
```

```
MY-ROUTER-X#terminal length 0
```

```
MY-ROUTER-X#show clock  
*00:32:10.464 UTC Tue Aug 29 2017  
MY-ROUTER-X#
```

```
show clock  
*00:32:10.464 UTC Tue Aug 29 2017  
MY-ROUTER-X#
```

プログラムで取得したコンソール出力

Paramiko

- ◆ PythonのSSHライブラリ
- ◆ SSHセッションを使ったりリモートマシン(鍵なしOK)の操作をしやすい

```
import paramiko
client = paramiko.SSHClient()
client.load_system_host_keys()
client.set_missing_host_key_policy(paramiko.AutoAddPolicy())

client.connect('192.168.0.61',username='root',password='python123',
look_for_keys=False,allow_agent=False)
command = 'cat /proc/meminfo | egrep -i "memtotal|memfree"'
(stdin,stdout,stderr) = client.exec_command(command)
for line in stdout:
    print(line, end='')
```

MemTotal:	8193312 kB
MemFree:	6675736 kB

Netmiko(1)

- ◆ Paramikoをネットワーク機器(Cisco IOSなど)に対応させたライブラリ
- ◆ コマンドを通じて情報の取得や設定の変更を行う
- ◆ 長所：学習コストが低いため、簡単な自動化には便利
- ◆ 短所：出力をパースする必要があり、複雑なシステムでは開発効率が低い

```
import netmiko
session = netmiko.ConnectHandler(device_type='cisco_ios',
host='192.168.0.1', username='admin',
password='python123', secret='python123')

session.enable()
output = session.send_command('show ip int bri')
print(output)
session.disconnect()
```

設定確認の例

Interface	IP-Address	OK?	Method	Status	
Protocol					
GigabitEthernet1	192.168.0.1	YES	manual	up	up
GigabitEthernet2	10.0.0.1	YES	manual	up	up
VirtualPortGroup0	192.168.0.2	YES	unset	up	up

Netmiko(2)

- ◆ 設定確認メソッド : `send_command`(コマンドの文字列)
- ◆ 設定変更メソッド : `send_config_set`(複数の命令(文字列)のリスト)

```
import netmiko
session = netmiko.ConnectHandler(device_type='cisco_ios', host='192.168.0.1',
username='admin', password='python123', secret='python123')

session.enable()
print(session.send_command('show ip int bri') + '\n')

create_loopback99 = ['int loopback99',
'ip address 99.99.99.99 255.255.255.255',
'no shut']
session.send_config_set(create_loopback99)

print(session.send_command('show ip int bri'))
session.disconnect()
```

設定変更の例

napalm

- ◆ 機器の操作(インタフェースの取得など)を抽象化したライブラリ
- ◆ コマンドに対応するAPIが存在する
- ◆ 長所: プログラムに組み込みやすい
- ◆ 短所: 対応するコマンドが少なく、設定変更方法に癖がある

```
import napalm
from pprint import pprint

driver = napalm.get_network_driver('ios')
device = driver(hostname='192.168.0.1',
username='admin', password='python123',
optional_args={'secret':'python123'})

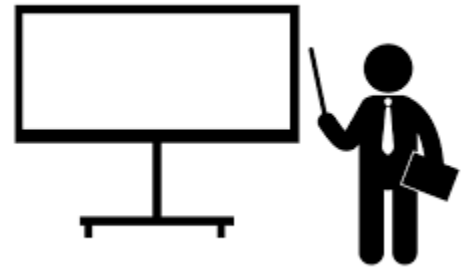
device.open()
output = device.get_interfaces()
pprint(output)
device.close()
```

```
{'GigabitEthernet1': {'description': '',
'is_enabled': True,
'is_up': True,
'last_flapped': -1.0,
'mac_address': '00:50:56:83:1A:EB',
'speed': 1000},
'GigabitEthernet2': {'description': '',
'is_enabled': True,
'is_up': True,
'last_flapped': -1.0,
'mac_address': '00:50:56:83:0D:0D',
'speed': 1000}}
```



他にも . . .

- ◆ REST(requests)
- ◆ NETCONF (ncclient)
- ◆ Ansible



Cisco CSR 1000v のAPIの仕様

- ◆ REST APIの仕様は製品ごとに異なるためドキュメントを参照
- ◆ CSR 1000v のREST APIの仕様も公開(CSR1000V REST APIなどで検索)

Home / ... / Cisco Cloud Services Router 1000V Series / Configuration Guides /
Cisco IOS XE REST API Management Reference Guide
Book Contents Find Matches in This Book Download Print
Chapter: Global Configuration Requirements Updated: December 4, 2016

Resource Summary for Global Configuration

Resource	URL (BaseURL)	HTTP Method			
		GET	POST	PUT	DELETE
Banner	/api/v1/global/banner	Y	N	Y	N
Host name	/api/v1/global/host-name	Y	N	Y	N
Domain name	/api/v1/global/domain-name	Y	N	Y	N

APIとメソッドの表(Host名はGET, PUT が使える)



Retrieve Device Hostname
Resource URI

Verb	URI
GET	/api/v1/global/host-name

Example
JSON Request

```
GET /api/v1/global/host-name
Accept: application/json
```

JSON Response

```
200 Ok
Content-Type: application/json

{
  "kind" : "object#host-name",
  "host-name" : "{string}"
}
```

ホスト名の取得の詳細

Modify Device Hostname
Resource URI

Verb	URI
PUT	/api/v1/global/host-name

Example
JSON Request

```
PUT /api/v1/global/host-name
Content-Type: application/json
Accept: application/json
```

```
{
  "host-name": "eng-router"
}
```

JSON Response ← 200 or 204

```
200 Ok
Content-Type: application/json

{
  "host-name": "eng-router"
}
```

JSON Response with no Response Body

```
204 No Content
```

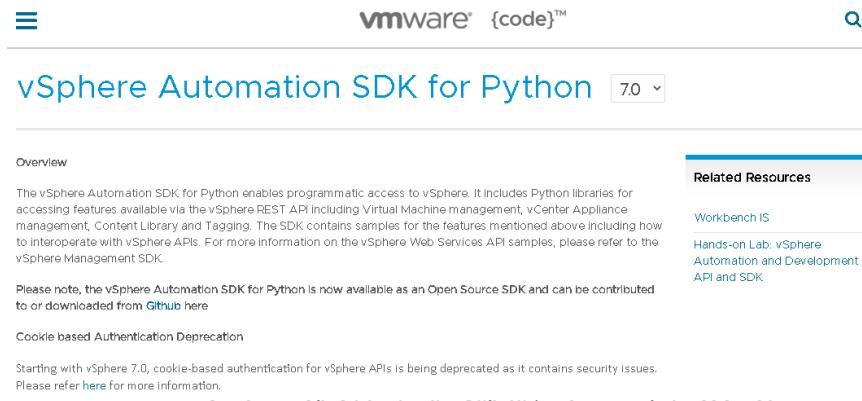
ホスト名の変更の詳細

サーバ管理・監視をPythonで 実施してみる



vSphere Automation SDK for Python

- ◆ VMware vSphereでは様々な開発用ツールキットが提供されている
- ◆ vSphere Automation SDK for PythonによりPythonプログラムによる自動化が可能
- ◆ vSphere Automation SDK for PythonはVMware Webサイト又はGitHubから入手可能
 - ✓ <https://github.com/vmware/pyvmomi-community-samples>
 - ✓ <https://code.vmware.com/web/sdk/7.0/vsphere-automation-python>
- ◆ 多くのサンプルプログラムが提供されているためプログラム開発時のテンプレートとして利用可能



The screenshot shows the VMware vSphere Automation SDK for Python page on the code.vmware.com website. The page title is "vSphere Automation SDK for Python" with a version dropdown set to "7.0". The main content area includes an "Overview" section with the following text: "The vSphere Automation SDK for Python enables programmatic access to vSphere. It includes Python libraries for accessing features available via the vSphere REST API including Virtual Machine management, vCenter Appliance management, Content Library and Tagging. The SDK contains samples for the features mentioned above including how to interoperate with vSphere APIs. For more information on the vSphere Web Services API samples, please refer to the vSphere Management SDK." Below this is a "Please note" section stating: "Please note, the vSphere Automation SDK for Python is now available as an Open Source SDK and can be contributed to or downloaded from [GitHub](#) here". There is also a "Cookie based Authentication Deprecation" section with the text: "Starting with vSphere 7.0, cookie-based authentication for vSphere APIs is being deprecated as it contains security issues. Please refer [here](#) for more information." On the right side, there is a "Related Resources" section with two links: "Workbench IS" and "Hands-on Lab: vSphere Automation and Development API and SDK".

サンプルプログラム

<https://github.com/vmware/pyvmomi-community-samples/samples>

- ◆ サンプルプログラムを元にカスタマイズすれば新規作成と比較して短時間でプログラムを作成することができる



A screenshot of the GitHub repository page for 'vmware/pyvmomi-community-samples'. The page shows the repository name, public status, and various navigation options like Code, Issues (304), Pull requests (35), Actions, Projects, Wiki, Security, and Insights. It also displays notification, star (831), and fork (817) counts.

master | pyvmomi-community-samples / samples | Go to file

Commit	Message	Time
...
ca9ec45	Use disableSslCertValidation=True to match vmware/pyvmomi@4f3294d	6 months ago
...
...	Fix PEP 8 issues	6 months ago
...	Use disableSslCertValidation=True to match vmware/pyvmomi@4f3294d	6 months ago
...	Samples Documentation	7 years ago
...	make testing actually possible	7 years ago
...	Fix pylint issues	6 months ago
...	Fix pylint issues	6 months ago
...	Fix pylint issues	6 months ago
...	Fix pylint issues	6 months ago
...	Fix pylint issues	6 months ago
...	Fix pylint issues	6 months ago
...	Fix pylint issues	6 months ago
...	Fix pylint issues	6 months ago
...	Fix pylint issues	6 months ago

仮想マシン詳細情報の取得

- ◆ サンプルプログラム `getallvms.py`を使用すると仮想マシン詳細情報の取得ができる

```
#python getallvms.py -s 172.31.0.254 -o 443 -u administrator@python.local -p Python123! -S
```

```
Name      · 01-TinyOS
Template   : False
Path       : [datastore1] 01-TinyOS/01-TinyOS.vmx
Guest      : Other 2.4.x Linux (32-bit)
Instance  UUID : 502adb91-4e8a-3955-4c4c-5ecdff874402
Bios      UUID  : 422a72b3-5d2b-d512-26e5-0a16dfe7bf07
Annotation : Created by Mike Brown
           2 February 2012
```

<途中省略>

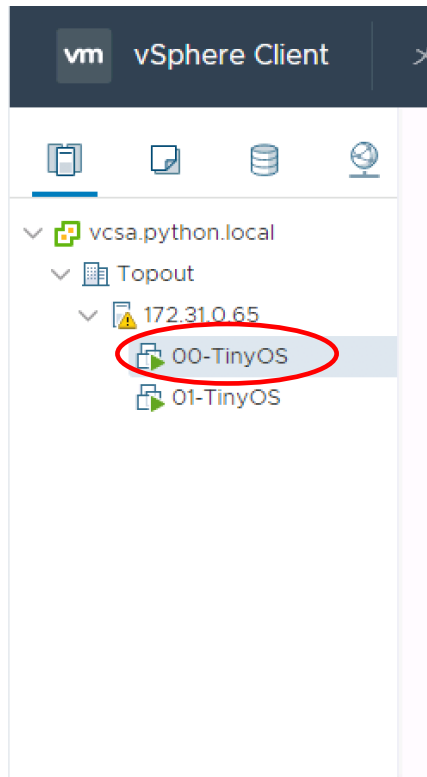
128 MB .vmdk

64 MB RAM

1 vCPU

VMware Tools Installed: No

<途中省略>



仮想マシン電源状態の操作 - シャットダウン

- ◆ Pythonプログラムでの仮想マシンシャットダウン
- ◆ `SerachIndex.FindByDnsName`によりDNS名から仮想マシンのオブジェクトIDを取得

```
vmname = "00-TinyOS"
```

```
si = SmartConnect(host=host,user=username,pwd=password)  
atexit.register(Disconnect, si)
```

```
content = si.content
```

```
vm = content.searchIndex.FindByDnsName(None,vmname,vmSearch=True)
```

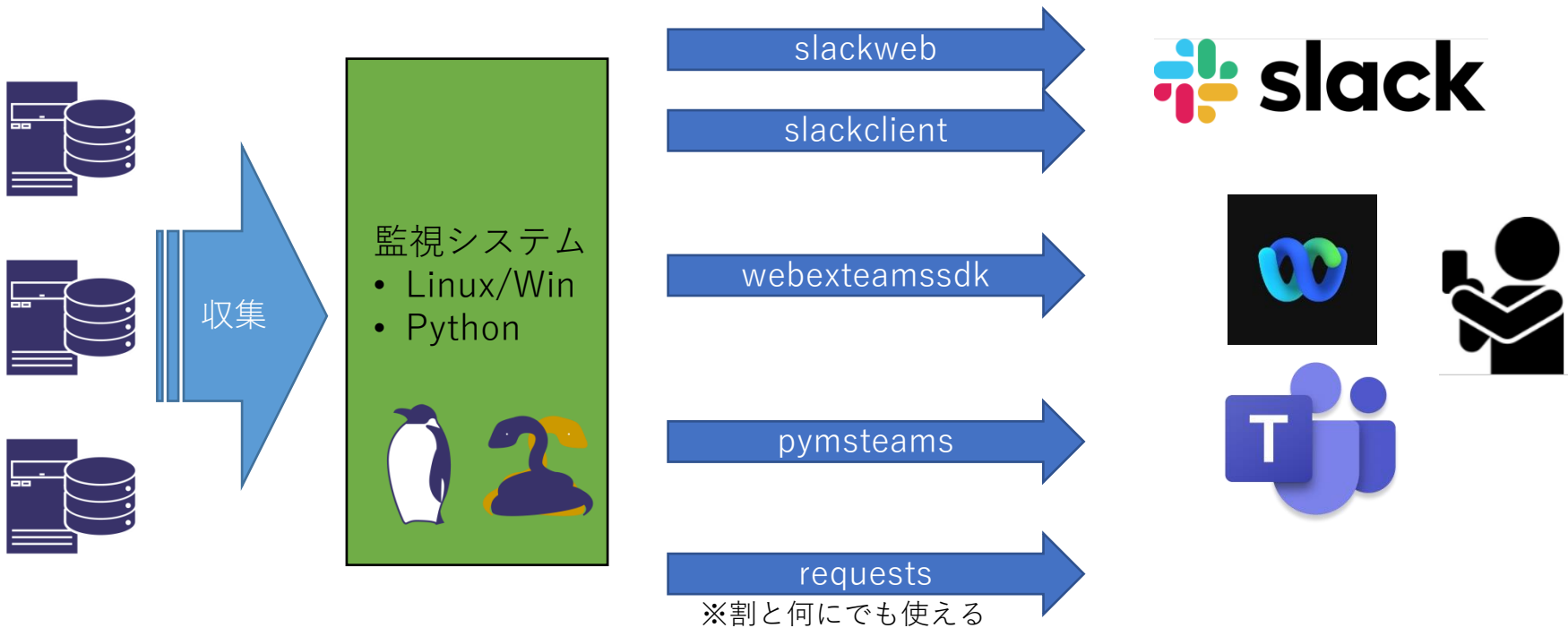
```
TASK = vm.ShutdownGuest()
```

※仮想マシンDNS名でオブジェクトIDを検索してシャットダウンを実行

担当者へPythonで通知してみる



コラボレーションツールへの通知



こんな用途に使えます

- ◆ 管理対象のデバイスから収集した情報を元に、しきい値を超えた場合に、プログラムが判断し、状態を管理者に自動的に通知
- ◆ 同時に、configやshowコマンドの結果を送信することで、発生から短時間で状態を把握することが可能
- ◆ 様々なコラボレーションツールに加えて、メール、SMS、ログサーバやファイルへの保存、ポータルサーバの起動など



Python ライブラリ (webex teams sdk) を使って Webex へ書込み

```
from webex teams sdk import WebexTeamsAPI
```

```
BOT_ACCESS_TOKEN = <BOT のアクセストークン>
```

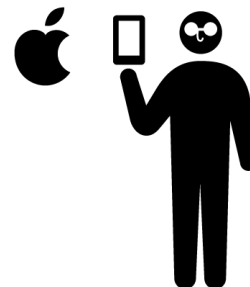
```
ROOM_ID = <ROOM ID>
```

```
markdown = "送信したい内容"
```

```
api = WebexTeamsAPI(access_token=BOT_ACCESS_TOKEN)
```

```
messages = api.messages.create(roomId=ROOM_ID, markdown=markdown)
```

```
print(messages)
```





Python ライブラリ (requests) を使ってWebexへ書込み

```
import requests
```

```
BOT_ACCESS_TOKEN = <BOT のアクセストークン>  
ROOM_ID = <ROOM ID>  
url = "https://webexapis.com/v1/messages"  
markdown = "送信したい内容"
```

```
payload = {  
    "roomId": ROOM_ID,  
    "markdown": markdown  
}
```

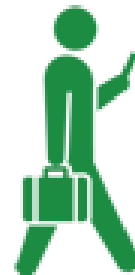
```
headers = {  
    "Authorization": f"Bearer {BOT_ACCESS_TOKEN}",  
    "Content-Type": "application/json"  
}
```

```
response = requests.post(url, headers=headers, json=payload)  
print(response.text)
```

slackの利用



- ◆ 情報を共有する方法として、メンバーが連携と取りながら仕事を進める新しい手段
- ◆ slackをpythonから利用するライブラリとモジュール（一例）
 - slackweb
 - slackclientライブラリ (slackモジュール)
 - requests
- ◆ 認証
 - Incoming WebHooks
 - Token



slackwebの利用(メッセージの送信)



- ◆ WebHookで簡単に投稿を行うことが可能

```
import slackweb
```

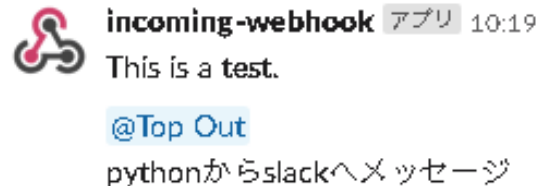
```
WEBHOOKURL = "https://hooks.slack.com/services/TR-----/BRH-----/9B-----"
```

```
SLACKUSERID = "<@URH----->¥n"
```

```
slack = slackweb.Slack(url = WEBHOOKURL )
```

```
slack.notify(text="This is a *test*.")
```

```
slack.notify(text= SLACKUSERID + "pythonからslackへメッセージ")
```



slackclientの利用(メッセージの送信/ファイルの共有)



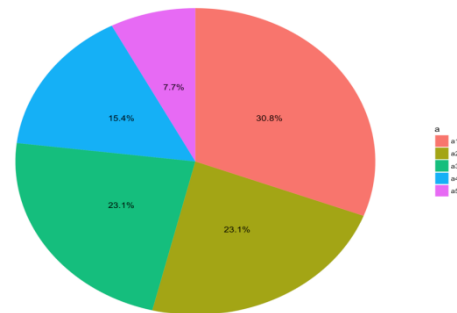
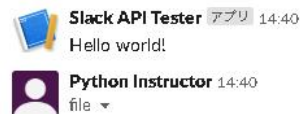
- ◆ slackが用意したdevelopment Kitを利用するため、簡単に利用することが可能

```
import slack
TOKEN =
"xoyp-85xxxxx-867xxxxx-867xxxxx-79cxxxxxxxx"
CHANNEL = '#顧客A-NW報告'
FILEPATH = r'c:¥python¥graph.jpg'
```

```
client = slack.WebClient(token=TOKEN)
```

```
res = client.chat_postMessage( channel=CHANNEL, text="Hello world!")
```

```
res = client.files_upload( channels=CHANNEL, file=FILEPATH)
```



requestsの利用(ファイル共有)



◆ 汎用的なrequestsモジュールを利用

```
import requests
```

```
TOKEN =
```

```
" xoxp-85xxxxx-867xxxxx-867xxxxx-79cxxxxxxxx "
```

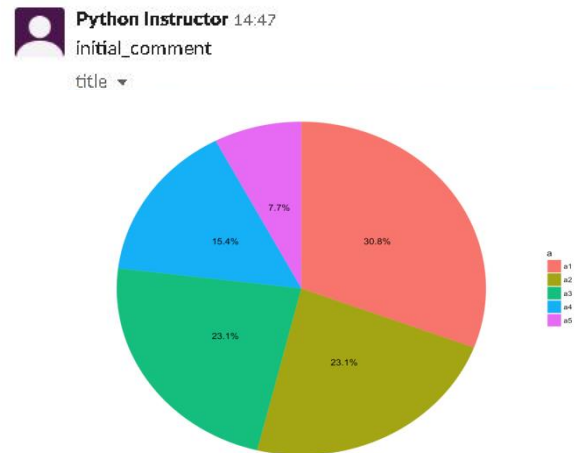
```
CHANNEL = '#python自動化編'
```

```
FILEPATH = r'c:¥python¥graph.jpg'
```

```
files = {'file': open(FILEPATH, 'rb')}
```

```
param = {'token':TOKEN, 'channels':CHANNEL, 'filename':"file",  
         'initial_comment': "initial_comment",'title': "title"}
```

```
requests.post(url="https://slack.com/api/files.upload"  
             ,params=param, files=files)
```



その他の通知方法

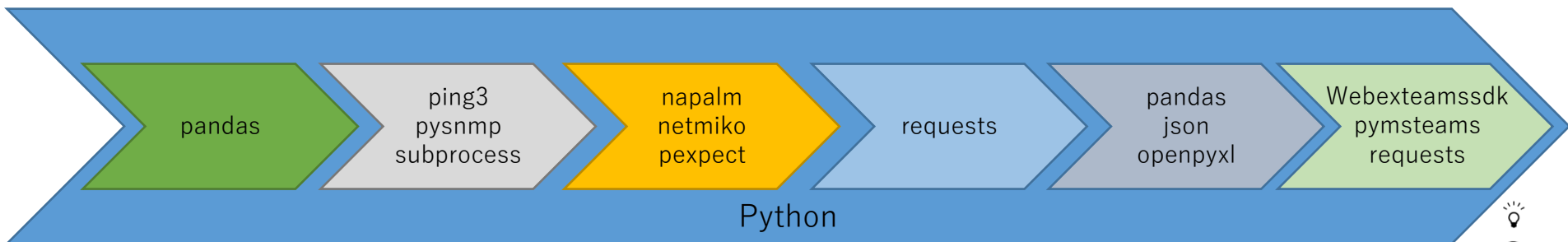
- ◆ メール
- ◆ ログ
- ◆ データベースへ保存
- ◆ ファイルへ保存
- ◆ ポータルサーバ(web)



さいごに



ふりかえりに



※表示しているライブラリは一例、ベストなものを使用する事



Pythonによる自動化を進めるために

- ◆ Pythonを使って自動化を進めていく方法を今回学習しました
- ◆ 同じことを3回実施するのであれば、この後も行う可能性が非常に高いと言われます。まずはそこから、Pythonに変更できないか考えてみてください
- ◆ 全ての作業を一度に変更することはなかなか難しいため、少しずつPythonのプログラムに変更していくのが簡単です



関連コースの紹介

- ◆ DevNet関連コース <https://www.topout.co.jp/cisco>
 - DevNet Associate対応コース
 - DEVASC (Developing Applications and Automating Workflows using Cisco Core Platforms)
 - DevNet Specialist / Professional対応コース
 - DEVCOR (Developing Applications Using Cisco Platforms and APIs)
- ◆ 前提条件のPythonの習得に <https://www.topout.co.jp/python>
 - Python 初級編・中級編・ネットワーク編・自動化編・サーバ編



ご清聴ありがとうございました



Thank You

Q&A

画面右側の Q&A ウィンドウから、
すべてのパネリスト (All Panelists) 宛
に送信してください。



次回の オンラインセミナー予定



Wireless TAC Time

– 今すぐ現場に効く Tips 紹介 –

2021 年 11 月 17 日 (水) 10:00 - 11:30

大崎 秀行 (Hideyuki Osaki)

シスコシステムズ

グローバル カスタマー エクスペリエンス センター
テクニカル コンサルティング エンジニア

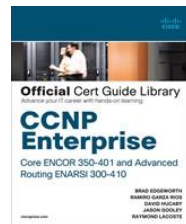


シスコ コミュニティ **100** 万人登録達成記念 スカベンジャー ハント開催中！

スペシャル バナーを 3 つ探せ！

コミュニティ内のところどころに、今回のゲーム専用のスペシャルバナーが設置されています。そのうちの3つを探し出し、バナーにあるミッションを遂行するとサプライズプレゼントが！

詳細はこちら



<https://community.cisco.com/t5/-/-/ta-p/4021064>

お役立てください！

シスココミュニティの歩き方



ついに公開！

シスココミュニティの歩き方

2020年2月版

利用方法や小技集など
All in One 役立ちガイド

詳しくはこちら

CISCO

The image shows a blue rectangular graphic with rounded corners. At the top left, it says 'ついに公開！' (Finally published!). Below that is the main title 'シスココミュニティの歩き方' (Cisco Community Guide) in large white characters. Underneath the title is '2020年2月版' (February 2020 edition). Further down, it lists '利用方法や小技集など All in One 役立ちガイド' (Usage methods, tips, etc. All in One handy guide). At the bottom left, there is a dark blue button with white text that says '詳しくはこちら' (Click here for details). In the top right corner, the Cisco logo is displayed. On the right side of the graphic, there is a circular icon containing stylized human figures in various colors (red, blue, orange, green) connected by dotted lines, representing a community.



<https://community.cisco.com/t5/-/-/ta-p/4021064>



ご参加ありがとうございました。

Community Liveと Cisco Communityの
各アンケートにも ぜひ ご協力ください。

