

IBM i で Python やってみた. 1



+



python™

=



ティアンドトラスト株式会社 北原 征夫

アジェンダ

- python 概説
- インストール
- データベース・アクセス
- itoolkit
- Web サーバー連携
- PHP と比較

Pytho 概説

■ 歴史

- 1991年 Guido van Rossum (ガイド=ヴァンロッサム:オランダ) v0.9 公開
- 1994年 Python 1.0 リリース
- 2000年 Python 2.0 リリース
- 2008年 Python 3.0 リリース 最新の安定版 3.9 (2020/10)

■ ライセンス形態

- PSFL : Python Software Foundation License
 - GPL互換
 - 改変したプログラムの配布時に改変部分の公開は不要
 - PSFL 以外のライセンスの配布物と一緒に配布可能

Java : 1995年公開
PHP : 1995年公開
RPG : 1959年開発

Python 概説

■ 特徴

- インデント
 - 波括弧{ } の代わりに コロン : とインデント でブロックを表現
 - 行の終端にセミコロン ; は不要 (改行で良い)
- インタープリター、マルチプラットフォーム
 - コンパイル不要
- pypi : Python Package Index
 - 豊富なモジュール群を利用できる : 268,148 project
 - Pip (Python Package Installer) によりインストール
- PEP : Python Enhancement Proposales
 - Python の決め事や標準化などが記されている
 - PEP8 : Python コーディング規約

Python

```
if night:
    print("Hello Night World")
else:
    print("hello Morning Wrold")
```

PHP

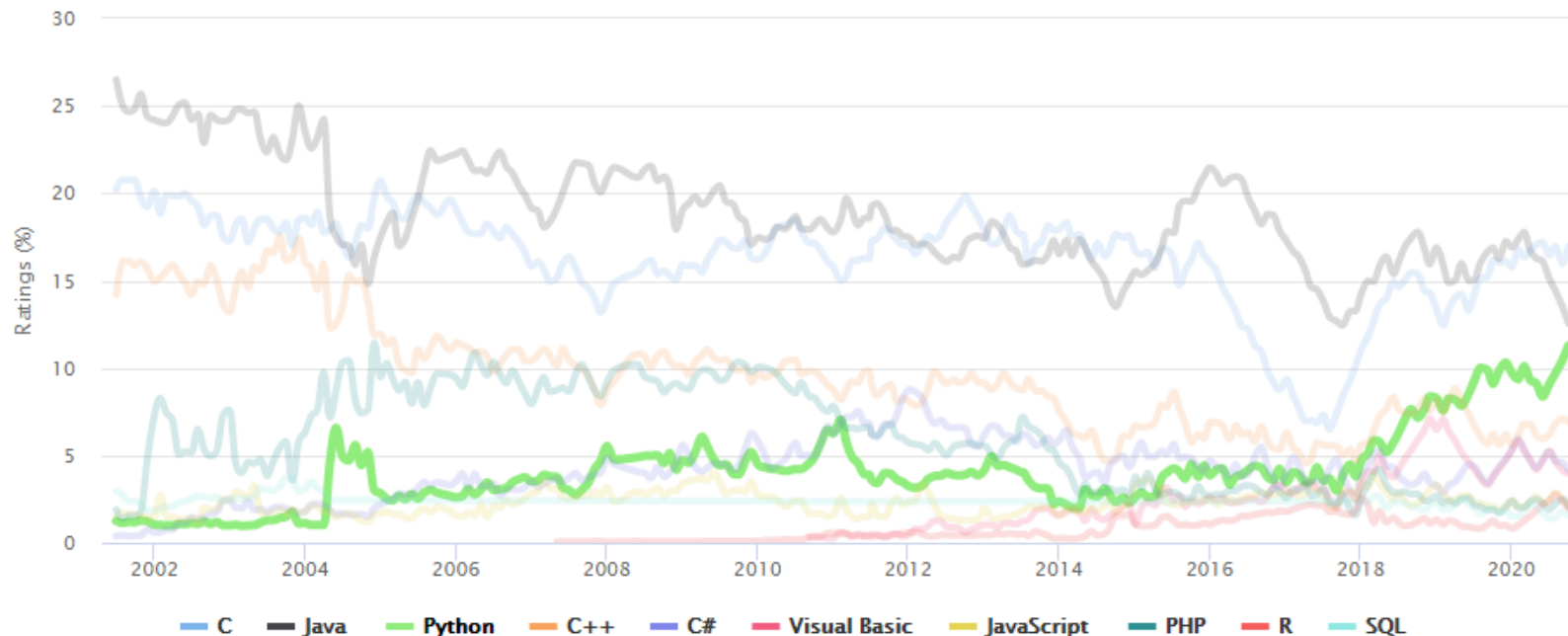
```
if(night){
    print("Hello Night World");
} else {
    print("Hello Morning World");
}
```

Python 概説

■ 2020年10月 人気度：TIOBE

TIOBEプログラミングコミュニティインデックス

出典：www.tiobe.com



Oct 2020	Oct 2019	Change	Programming Language	Ratings	Change
1	2	▲	C	16.95%	+0.77%
2	1	▼	Java	12.56%	-4.32%
3	3		Python	11.28%	+2.19%
4	4		C++	6.94%	+0.71%

AI (機械学習, 統計, 分析)

での利用が起因?

- 第三次 AIブーム (2006年?~)
- 非プログラマー



- ・ 習得の容易性
- ・ 豊富なライブラリー
- ・ 活用範囲の広さ

2018年ごろから急激な増加
→ Java に変わる言語として注目が

※1) TIOBE のソース
条件を満たした 25の検索エンジンのヒット数
(詳細は以下のリンクを参照)

<https://www.tiobe.com/tiobe-index/>

<https://www.tiobe.com/tiobe-index/programming-languages-definition/>



Python のインストール

インストール

■ 流れ

1. sshd の開始
2. yum を導入
3. Python の導入

インストール : 1. sshd の開始

■ 5250 を利用

- 5250エミュレーターで以下の CL コマンドを実行
 - STRTCPSVR SERVER(*SSHD)
- 開始は以下の CL コマンドで確認
 - NETSTAT OPTION(*CNN)

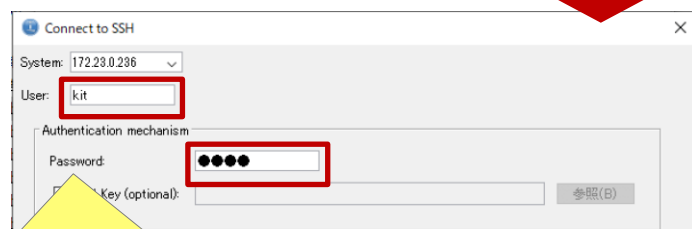
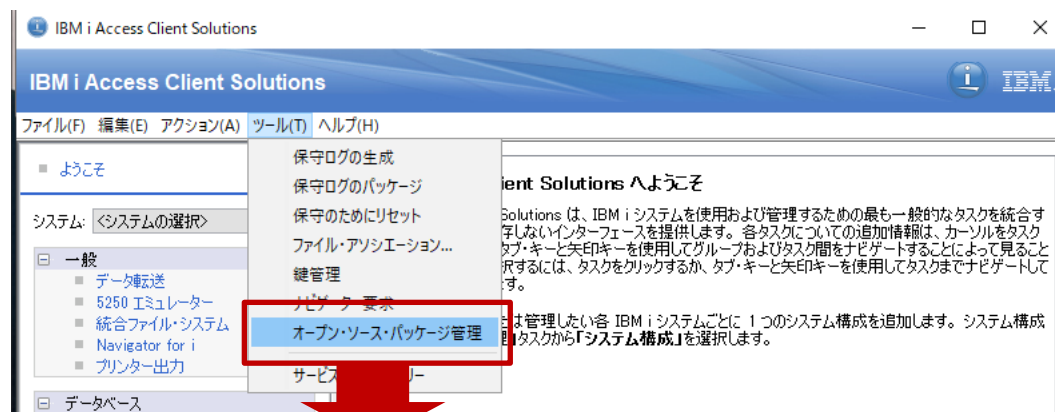
```
【PV4 接続状況の処理】
システム :
オプションを入力して、実行キーを押してください。
3= デバッグ使用可能   4= 終了   5= 詳細の表示   6= デバッグ使用不可
8= ジョブの表示

OPT  y-n      y-n      w-fモ     abn'モ
      an' 又   t'-n     t'-n      時間     状態
  |  *       *       ftp-con >
  |  *       *       ssh
  |  *       *       telnet
```

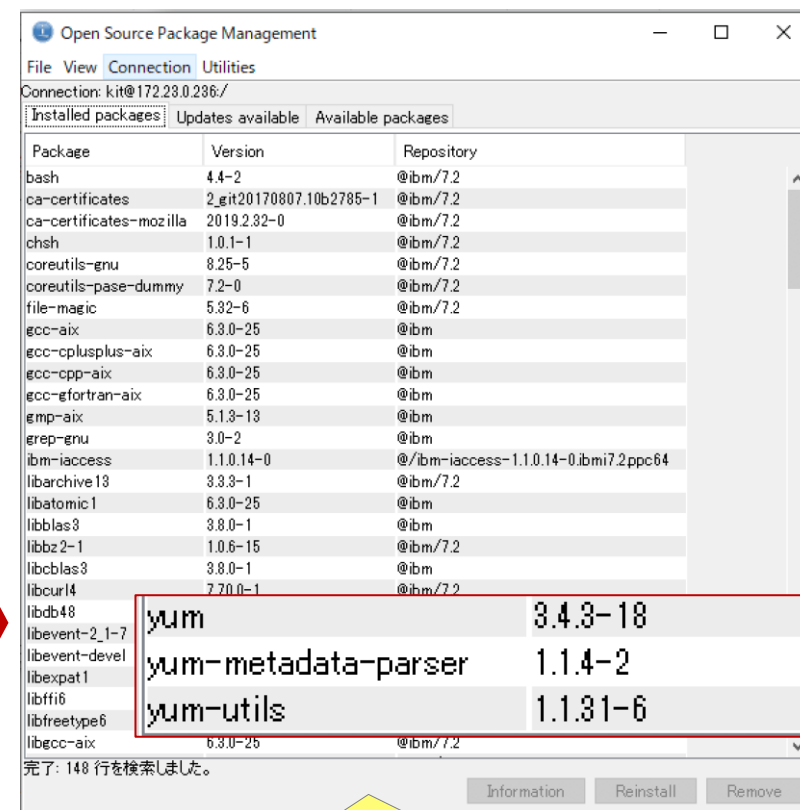
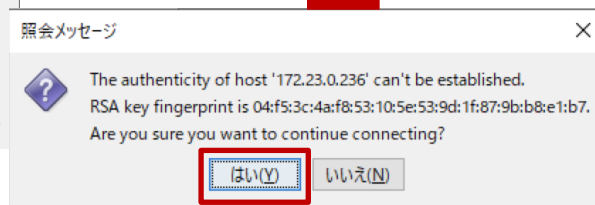
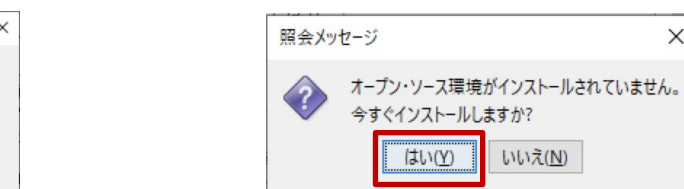
ssh が表示されればOK

インストール : 2. yum の導入

■ ACS オープンソース管理 から導入



IBM i の
ユーザーとパスワード



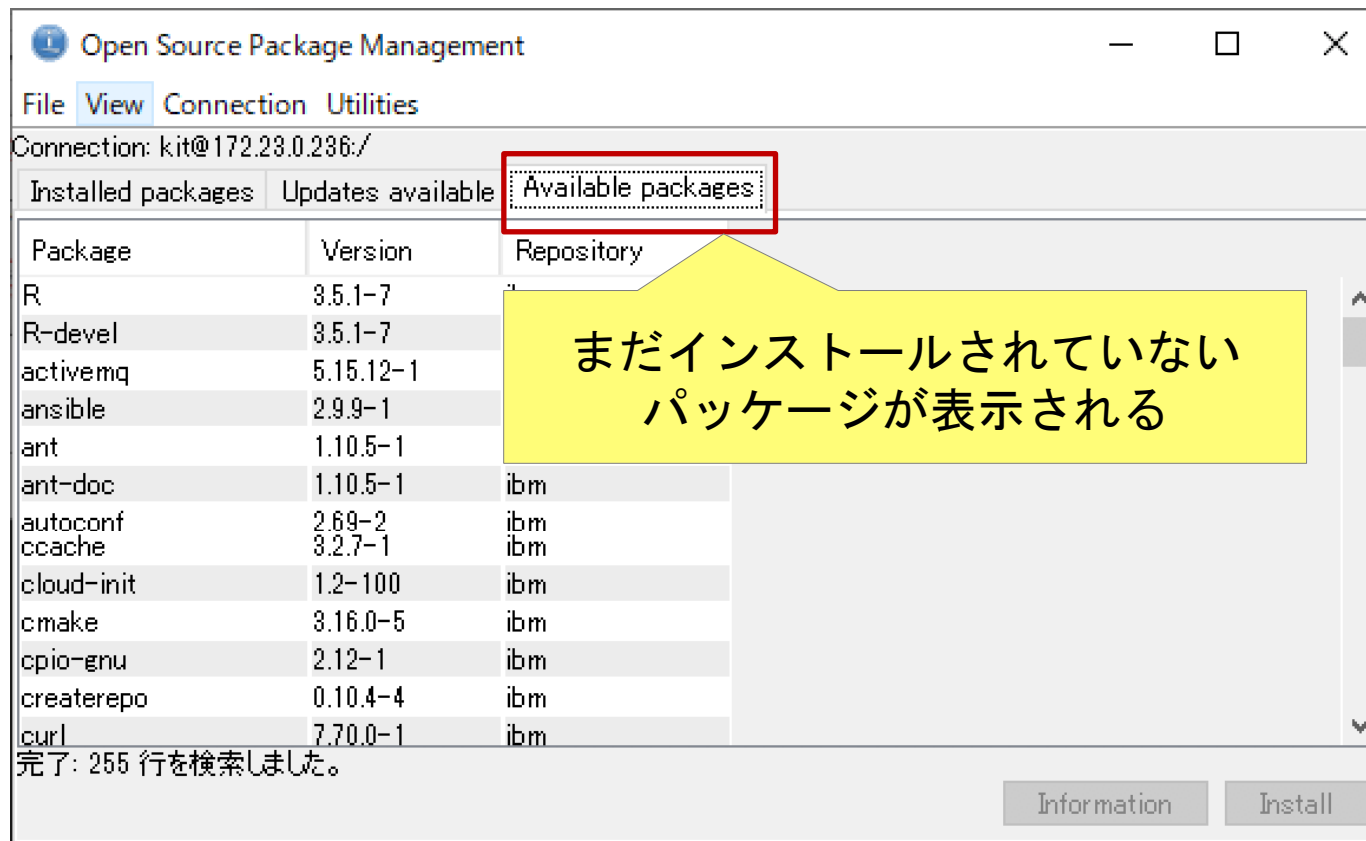
yum 3.4.3-18
yum-metadata-parser 1.1.4-2
yum-utils 1.1.31-6

基本的なパッケージと共に
Yum がインストールされる

※ 詳細は yum 分科会の発表資料をご参照ください。 https://i5php.jp/wp-content/uploads/2019/09/yum_guide.pdf

インストール : 3. Python の導入 1

- ACS オープンソース管理 から導入
 1. Open Source Package Management の「Available Packages」タブを開く



インストール : 3. Python の導入 2

■ ACS オープンソース管理 から導入

2. Python3 で始まるパッケージを全て選択し「Install」

Package	Version	Repository
python3	3.6.11-2	@ibm
python3-Pillow	5.0.0-5	@ibm
python3-asn1crypto	0.24.0-1	@ibm
python3-bcrypt	3.1.4-6	@ibm
python3-cffi	1.11.5-3	@ibm
python3-cryptography	2.8-0	@ibm
python3-dateutil	2.8.0-1	@ibm
python3-devel	3.6.11-2	@ibm
python3-ibm_db	2.0.5.12-0	@ibm
python3-idna	2.8-1	@ibm
python3-itoolkit	1.6.1-1	@ibm
python3-jinja2	2.11.2-1	@ibm
python3-lxml	4.2.1-4	@ibm
python3-markupsafe	1.1.1-1	@ibm
python3-numpy	1.15.4-0	@ibm
python3-pandas	0.22.0-5	@ibm
python3-paramiko	2.6.0-1	@ibm
python3-pip	9.0.1-3	@ibm
python3-psutil	5.5.1-0	@ibm
python3-psycopg2	2.8.5-1	@ibm

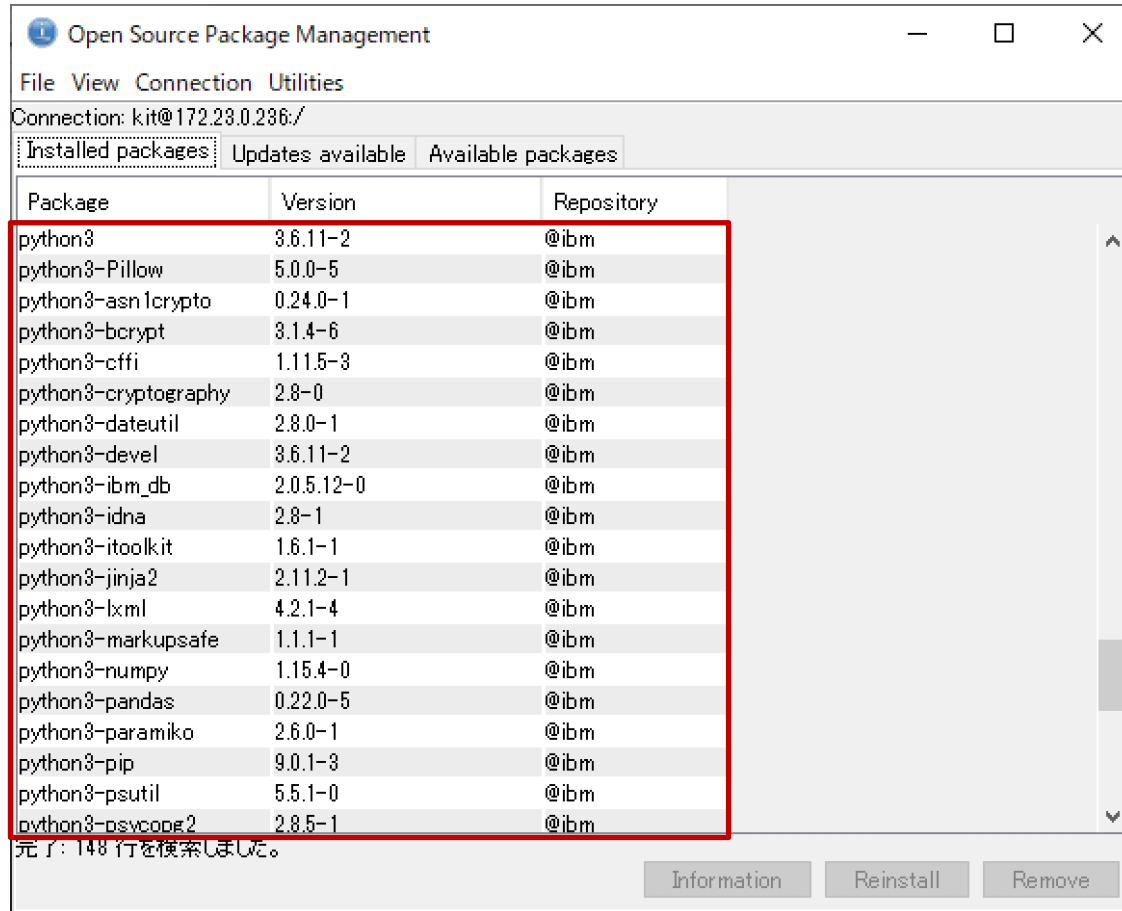
完了: 255 行を検索しました。

Information **Install**

Shift + ↓

インストール : 3. Python の導入 2

- ACS オープンソース管理 から導入
- ## 3. Install package タブに表示されればOK



Open Source Package Management

File View Connection Utilities

Connection: kit@172.28.0.236:/

Installed packages Updates available Available packages

Package	Version	Repository
python3	3.6.11-2	@ibm
python3-Pillow	5.0.0-5	@ibm
python3-asn1crypto	0.24.0-1	@ibm
python3-bcrypt	3.1.4-6	@ibm
python3-cffi	1.11.5-3	@ibm
python3-cryptography	2.8-0	@ibm
python3-dateutil	2.8.0-1	@ibm
python3-devel	3.6.11-2	@ibm
python3-ibm_db	2.0.5.12-0	@ibm
python3-idna	2.8-1	@ibm
python3-itookit	1.6.1-1	@ibm
python3-jinja2	2.11.2-1	@ibm
python3-lxml	4.2.1-4	@ibm
python3-markupsafe	1.1.1-1	@ibm
python3-numpy	1.15.4-0	@ibm
python3-pandas	0.22.0-5	@ibm
python3-paramiko	2.6.0-1	@ibm
python3-pip	9.0.1-3	@ibm
python3-psutil	5.5.1-0	@ibm
python3-psycopg2	2.8.5-1	@ibm

完了: 148 行を検索しました。

Information Reinstall Remove

yum コマンド の場合
\$ yum install python3*



データベース・アクセス

■ Db2 for i へのアクセス方法は 2種類

- **ibm_db** ← 今回はこちら
- **pyODBC**

データベース・アクセス : ibm_db

■ インストール

- ACS の場合

- ACS で python3-ibm_db を選択し Install

- yum コマンドの場合

- yum install python3-ibm_db

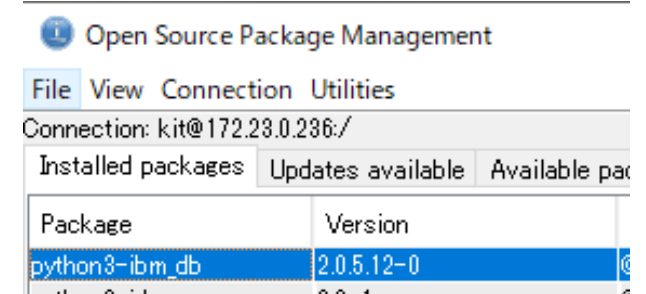
■ 2種類のモジュールが含まれる

- **ibm_db** : ベース・モジュール

- SQL 照会の発行、ストアド・プロシージャの呼び出し等

- **ibm_db_dbi** : Python データアクセス仕様に準拠

- PEP249 : Python Database API Specification v2.0



The screenshot shows a terminal window titled "Open Source Package Management". It has a menu bar with "File", "View", "Connection", and "Utilities". Below the menu bar, it says "Connection: kit@172.23.0.236:/". There are three tabs: "Installed packages", "Updates available", and "Available packages". The "Installed packages" tab is active, showing a table with two columns: "Package" and "Version". The first row in the table is "python3-ibm_db" with version "2.0.5.12-0".

Package	Version
python3-ibm_db	2.0.5.12-0

← 今回はこちら

データベース・アクセス : shell で実行してみた

■ ibm_db_dbi の利用

exdb01_app_cli.py

```
1 import ibm_db_dbi as db
2 import io,sys
3 sys.stdout = io.TextIOWrapper(sys.stdout.buffer, encoding='utf-8')
4
5 conn = db.connect("DATABASE=*LOCAL;", "KIT", "<password>")
6 cursor = conn.cursor()
7 cursor.execute("SELECT * FROM QEOL.TOKMSP")
8 for r in cursor.fetchall():
9     print(r[0], end=' ')
10    print(r[1], end=' ')
11    print(r[2])
12
```

\$ python exdb01_app_cli.py



01010	アイ リヨカ	阿井旅館
01020	アイ コウキョウ	阿井工業
01030	アイカワ コウキョウ	相川工業
01040	アイ リヨウシャ	阿井旅行社
01050	アイ ショウトウK.K	阿井食品K. K
01060	アイ シットウシャ	阿井自動車
01070	アイカワ カメラ	相川カメラ
01080	アイカワ コウケンK.K	相川広告K. K
01090	アイカワ テンキK.K	相川電機K. K
01100	アイカワ ガツケン	相川楽器店
01110	アイカワ セツケイジムコ	相川設計事務所
01120	アイカワ ショウジ	相川商事

データベース・アクセス: shell で実行してみた

■ 解説

ibm_db_dbi
モジュールの読み込み

標準出力を utf-8 に設定しないと
UnicodeEncodeError になる

```
1 import ibm_db_dbi as db
2 import io,sys
3 sys.stdout = io.TextIOWrapper(sys.stdout.buffer, encoding='utf-8')
4
5 conn = db.connect("DATABASE=*LOCAL;", "KIT", "<password>")
6 cursor = conn.cursor()
7 cursor.execute("SELECT * FROM QEOL.TOKMSP")
8 for r in cursor.fetchall():
9     print(r[0], end=' ')
10    print(r[1], end=' ')
11    print(r[2])
12
```

IBM I との接続
リレーショナル
データベース名を
指定

フェッチし、ループ
レコード1件毎に
出力

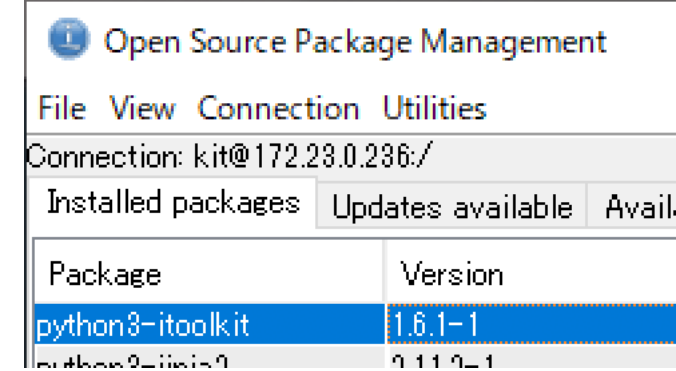
カーソル作成と
SQLの実行



itoolkit

■ インストール

- ACS の場合
 - ACS で python3-itoolkit を選択し Install
- yum コマンドの場合
 - yum install python3-itoolkit



Open Source Package Management

File View Connection Utilities

Connection: kit@172.29.0.236:/

Installed packages Updates available Avail.

Package	Version
python3-itoolkit	1.6.1-1
python3-itsio-0	0.11.0-1

itoolkit : shell で実行してみた

■ itoolkit の利用

exxt01_app_cli.py

```
1 import ibm_db_dbi
2 from itoolkit import *
3 from itoolkit.transport import DatabaseTransport
4
5 conn = ibm_db_dbi.connect("DATABASE=*LOCAL","KIT","<password>")
6 itransport = DatabaseTransport(conn, ctl="*pase(1208/5035)")
7
8 itool = iToolKit()
9 itool.add(iCmd5250('dspsyssts', 'DSPSYSSTS'))
10 itool.call(itransport)
11 result = itool.dict_out('dspsyssts')
12 print(result['dspsyssts'])
13
```

\$ export LANG=JA_JP.UTF-8
\$ python exxt01_app_cli.py



システム状況情報										ページ 1	
5770SS1 V7R3M0 160422										SPIKE73	20/10/16 19:29:24 JST
CPU使用%									.9	システムASP	95.45 G
上限なしCPU容量使用%									.1	システムASP使用%	45.5479
経過時間									00:00:01	合計補助記憶域	95.45 G
システム内のジョブ									1383	現行使用一時	4901 M
永久アドレス%									.007	ピーク使用一時	4928 M
一時アドレス%									.010		
s^-											
ニヲmw	ホ-モ	予約済	MAX	-----DB-----	-----非DB-----	ACT->	WAIT->	ACT->			送り
ホ-モ	カヲ M	カヲ M	ACT	不在	s^-	不在	s^-	ホ-モ	カヲニヲmw		eヲニ]
1	679.21	342.26	+++++	.0	.0	.0	.0	54.1	.0	.0	*MACHINE *FIXED
2	4447.20	6.00	121	.0	.0	.0	.0	4007.2	.0	.0	*BASE *FIXED
3	575.97	.00	144	.0	.0	.0	.0	.0	.0	.0	*INTERACT *FIXED
4	57.59	.00	5	.0	.0	.0	.0	.0	.0	.0	*SPOOL *FIXED
***** リ ス ト の 終 わ り *****											

itoolkit : shell で実行してみた

■ 解説

```
1 import ibm_db_dbi
2 from itoolkit import *
3 from itoolkit.transport import DatabaseTransport
4
5 conn = ibm_db_dbi.connect("DATABASE=*LOCAL","KIT","<password>")
6 itransport = DatabaseTransport(conn, ctl="*pase(1208/5035)")
7
8 itool = iToolKit()
9 itool.add(iCmd5250('dspsyssts', 'DSPSYSSTS'))
10 itool.call(itransport)
11 result = itool.dict_out('dspsyssts')
12 print(result['dspsyssts'])
13
```

ibm_db_dbi, itoolkit
モジュールの読み込み

IBM i との接続

コマンド実行時の設定

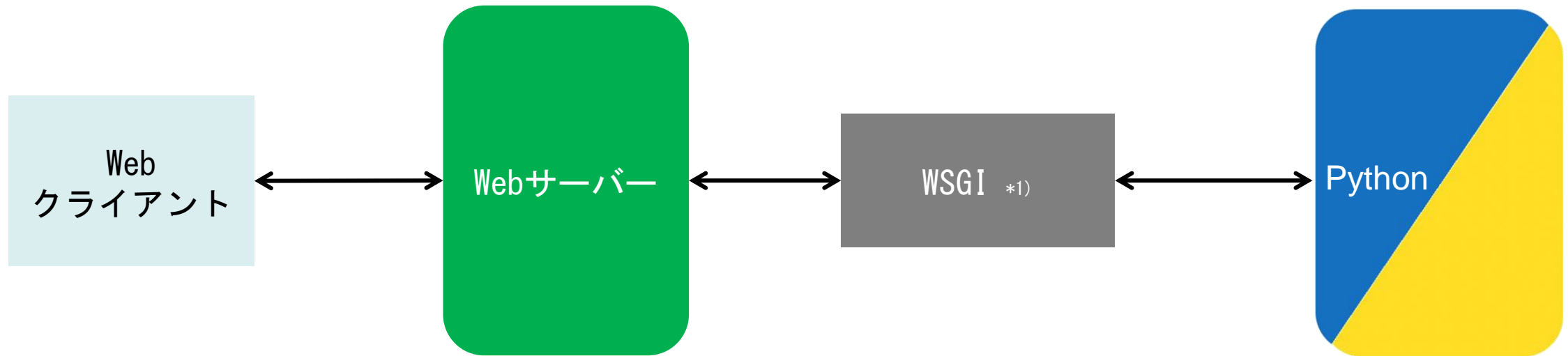
コマンドの指定と実行

結果の出力



Web サーバー連携

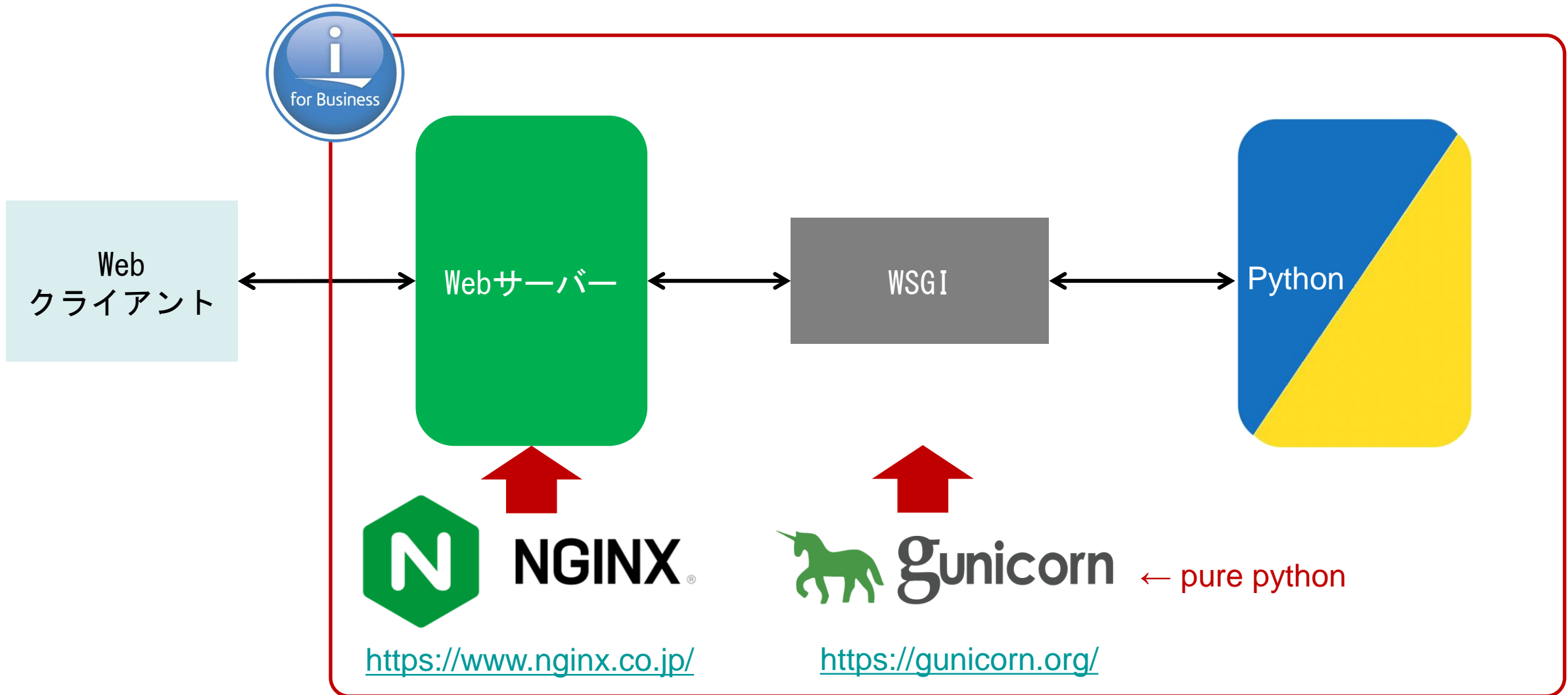
■ Python Web アプリケーション構成



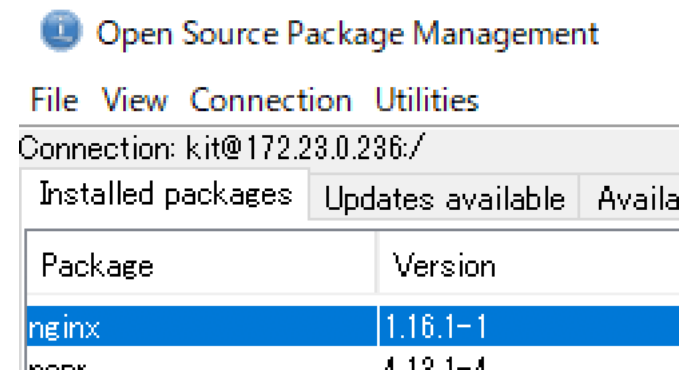
*1) Web Server Gateway Interface

Web サーバー連携

■ 今回試した構成



- Nginx インストール
 - ACS の場合
 - ACS で nginx を選択し Install
 - yum コマンドの場合
 - yum install nginx
- Gunicorn インストール
 - pip コマンド *2)
 - pip install gunicorn



Open Source Package Management

File View Connection Utilities

Connection: kit@172.28.0.236:/

Package	Version
nginx	1.16.1-1

*2) Pip Install Package : Python のパッケージ管理システム

```
bash-4.4$ pip install gunicorn
Collecting gunicorn
  Downloading https://files.pythonhosted.org/packages/69/ca/926f7cd3a2014b16870086b2d0fdc84a9e49473c68a8dff8b57f7c156f43/gunicorn-20.0.4-py2.py3-none-any.whl (77kB)
    100% |#####| 81kB 1.2MB/s
Requirement already satisfied: setuptools>=3.0 in /QOpenSys/pkgs/lib/python3.6/site-packages (from gunicorn)
Installing collected packages: gunicorn
Successfully installed gunicorn-20.0.4
```

■ nginx.conf

```
bash-4.4$ cat nginx.conf
worker_processes 1;

events {
    worker_connections 512;
}

http {
    server {
        listen 9123;
        server_name INFRA-PRACTICE-NGINX;
        charset UTF-8;

        proxy_set_header    Host    $host;

        location / {
            proxy_pass http://127.0.0.1:9876;
        }
    }
}
```

■ gunicorn_settings.py

```
bash-4.4$ cat gunicorn_settings.py
import os

bind = '127.0.0.1:' + str(os.getenv('PORT', 9876))
proc_name = 'Infrastructure-Practice'
workers = 1
```

配置

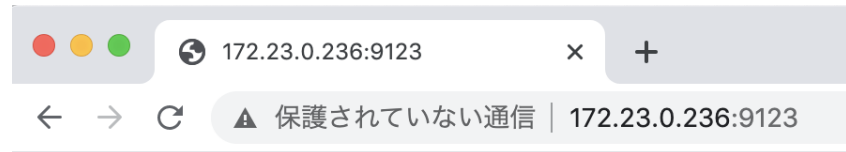
```
bash-4.4$ tree -L 2 /Q0penSys/usr/share/nginx_gp
/Q0penSys/usr/share/nginx_gp
├── config
│   ├── __pycache__
│   ├── gunicorn_settings.py
│   └── nginx.conf
```

Web サーバー連携 : データベース・アクセス

exdb01_app.py

```
1 import ibm_db_dbi as db
2
3 def application(env, start_response):
4     start_response('200 OK', [('Content-Type', 'text/html')])
5     conn = db.connect("DATABASE=*LOCAL", "KIT", "KITA")
6     cursor = conn.cursor()
7     cursor.execute("SELECT * FROM QEOL.TOKMSP")
8     ret = ""
9     for r in cursor.fetchall():
10         ret += r[0] + ' ' + r[1] + ' ' + r[2] + '</br>'
11     return [bytes(ret, 'utf-8')]
12
```

```
bash-4.4$ tree -L 2 /QopenSys/usr/share/nginx_gp
/QopenSys/usr/share/nginx_gp
├── config
│   ├── __pycache__
│   ├── gunicorn_settings.py
│   └── nginx.conf
└── exdb01_app.py
```



Nginx, Gunicorn 起動

```
bash-4.4$ nginx -c /QopenSys/usr/share/nginx_gp/config/nginx.conf
bash-4.4$ cd /QopenSys/usr/share/nginx_gp
bash-4.4$ gunicorn exdb01_app:application -c config/gunicorn_settings.py --reload
[2020-10-16 21:25:42 JST] [4848] [INFO] Starting gunicorn 20.0.4
[2020-10-16 21:25:42 JST] [4848] [INFO] Listening at: http://127.0.0.1:9876 (4848)
[2020-10-16 21:25:42 JST] [4848] [INFO] Using worker: sync
[2020-10-16 21:25:42 JST] [4850] [INFO] Booting worker with pid: 4850
```



- 01010 アイリョク 阿井旅館
- 01020 アイコギヨ 阿井工業
- 01030 アイカコギヨ 相川工業
- 01040 アイリョクヤ 阿井旅行社
- 01050 アイソウトウK.K 阿井食品K. K
- 01060 アイジドウヤ 阿井自動車
- 01070 アイカカメラ 相川カメラ
- 01080 アイカコウケンK.K 相川広告K. K
- 01090 アイカデンキK.K 相川電機K. K
- 01100 アイカガキヤ 相川製菓店

Web サーバー連携 : itoolkit

exxt01_app.py

```
1 from itoolkit import *
2 from itoolkit.transport import DatabaseTransport
3 import ibm_db_dbi
4
5 def application(env, start_response):
6     start_response('200 OK', [('Content-Type', 'text/plain')])
7     conn = ibm_db_dbi.connect("DATABASE=*LOCAL","KIT","KITA")
8     itransport = DatabaseTransport(conn,ctl="*pase(1208/5035)")
9
10    itool = iToolKit()
11    itool.add(iCmd5250('dspsyssts', 'DSPSYSSTS'))
12    itool.call(itransport)
13    result = itool.dict_out('dspsyssts')
14    return [bytes(str(result['dspsyssts']), 'utf-8')]
15
```

```
bash-4.4$ tree -L 2 /Q0penSys/usr/share/nginx_gp
/Q0penSys/usr/share/nginx_gp
├── config
│   ├── __pycache__
│   ├── gunicorn_settings.py
│   └── nginx.conf
└── exxt01_app.py
```

Nginx, Gunicorn 起動

```
bash-4.4$ nginx -c /Q0penSys/usr/share/nginx_gp/config/nginx.conf
bash-4.4$ cd /Q0penSys/usr/share/nginx_gp
bash-4.4$ gunicorn exxt01_app:application -c config/gunicorn_settings.py --reload
[2020-10-16 22:01:22 JST] [4914] [INFO] Starting gunicorn 20.0.4
[2020-10-16 22:01:22 JST] [4914] [INFO] Listening at: http://127.0.0.1:9876 (4914)
[2020-10-16 22:01:22 JST] [4914] [INFO] Using worker: sync
[2020-10-16 22:01:22 JST] [4916] [INFO] Booting worker with pid: 4916
```



172.23.0.236:9123 x +

保護されていない通信 | 172.23.0.236:9123

システム状況情報

システム状況情報		ページ 1	
5770SS1 V7R3M0 160422		SPIKE73 20/10/16 21:50:59 JST	
CPU使用%	2.3	システムASP	95.45 G
上限なしCPU容量使用%4	システムASP使用%	45.5669
経過時間	00:00:01	合計補助記憶域	95.45 G
システム内のジョブ	1382	現行使用一時	4904 M
永久アドレス%007	ピーク使用一時	5113 M
一時アドレス%010		

送

スレッド	予約済	MAX	DB	非DB	ACT->	WAIT->	ACT->	送り
1	673.41	342.25	+++++	.0 .0 .0 .0	55.4	.0 .0	*MACHINE	*FIXED
2	4453.01	6.01	121	.0 .0 .0 .0	4432.1	.0 .0	*BASE	*FIXED
3	575.97	.00	144	.0 .0 .0 .0	.0 .0	.0 .0	*INTERACT	*FIXED
4	57.59	.00	5	.0 .0 .0 .0	.0 .0	.0 .0	*SPOOL	*FIXED

***** リストの終わり *****



ZendPHP7 (ZendServer)

PHP と比較

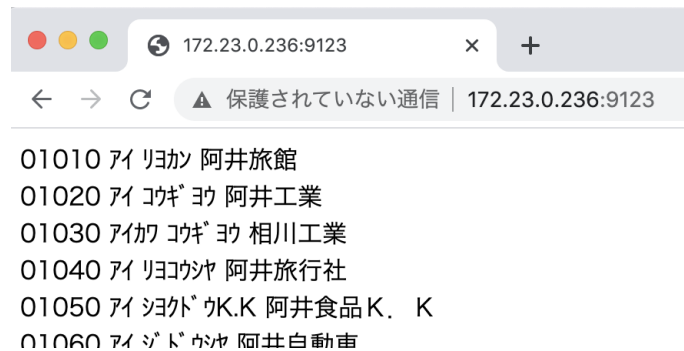
PHP との比較 : データベース・アクセス

■ Python

```
1 import ibm_db_dbi as db
2
3 def application(env, start_response):
4     start_response('200 OK', [('Content-Type', 'text/html')])
5     conn = db.connect("DATABASE=*LOCAL","KIT","KITA")
6     cursor = conn.cursor()
7     cursor.execute("SELECT * FROM QEOL.TOKMSP")
8     ret = ""
9     for r in cursor.fetchall():
10         ret += r[0] + ' ' + r[1] + ' ' + r[2] + '<br>'
11     return [bytes(ret,'utf-8')]
12
```

■ PHP

```
1 <?php
2
3 header("Content-type: text/html; charset=utf-8");
4 $conn = db2_connect('*LOCAL', 'KIT', 'KITA');
5 $result = db2_exec($conn, "SELECT * FROM QEOL.TOKMSP");
6 $ret = "";
7 while($r = db2_fetch_array($result)) {
8     $ret .= $r[0]. " " . $r[1]. " " . $r[2]. "<br>";
9 }
10 echo $ret;
11
12 ?>
```



PHP との比較 : TOOLKIT

Python

```
1 from itoolkit import *
2 from itoolkit.transport import DatabaseTransport
3 import ibm_db_dbi
4
5 def application(env, start_response):
6     start_response('200 OK', [('Content-Type', 'text/plain')])
7     conn = ibm_db_dbi.connect("DATABASE=*LOCAL", "KIT", "KITA")
8     itransport = DatabaseTransport(conn,ctl="*pase(1208/5035)")
9
10    itool = iToolkit()
11    itool.add(iCmd5250('dspsyssts', 'DSPSYSSTS'))
12    itool.call(itransport)
13    result = itool.dict_out('dspsyssts')
14    return [bytes(str(result['dspsyssts']), 'utf-8')]
15
```

PHP

```
1 <?php
2 include_once 'ToolkitService.php';
3
4 header("Content-type: text/plain; charset=utf-8");
5 $conn = db2_connect('*LOCAL', 'KIT', 'KITA');
6 $itool = ToolkitService::getInstance($conn);
7
8 $result = $itool->CLInteractiveCommand("DSPSYSSTS");
9 $ret = "";
10 foreach($result as $r)
11     $ret .= $r."\n";
12 echo $ret;
13 ?>
```



システム状況情報

5770SS1 V7R3M0 160422	SPIKE73 20/10/16 21:50:59 JST
CPU使用% 2.3	システムASP 95.45 G
上限なしCPU容量使用% 4	システムASP使用% 45.5669
経過時間 00:00:01	合計補助記憶域 95.45 G
システム内のジョブ 1382	現行使用一時 4904 M
永久アドレス% 0.07	ピーク使用一時 5113 M
一時アドレス% 0.10	

システム内ジョブ詳細:

ジョブ名	状態	優先度	待ち時間	実行時間	待ち時間	実行時間	待ち時間	実行時間	待ち時間	実行時間	待ち時間	実行時間
1	673.41	342.25	+++++	.0	.0	.0	.0	55.4	.0	.0	*MACHINE	*FIXED
2	4453.01	6.01	121	.0	.0	.0	.0	4432.1	.0	.0	*BASE	*FIXED
3	575.97	.00	144	.0	.0	.0	.0	.0	.0	.0	*INTERACT	*FIXED
4	57.59	.00	5	.0	.0	.0	.0	.0	.0	.0	*SPOOL	*FIXED



今後の Python 分科会活動

Python 分科会の活動候補

■ 今後の Python 検証

- 今回の続き
 - pyODBC, ストアドプロシージャ (RPG連携), iToolkit, PHPとの比較
- Web アプリケーション
 - フレームワーク(Django/Flask), パフォーマンス, Rest API
- 解析
 - IBM i だけで「画像解析」がどこまでできるのか
 - RFE に OpenCV をリクエスト中です。投票 お願いします。
 - https://www.ibm.com/developerworks/rfe/execute?use_case=viewRfe&CR_ID=140124
- 機械学習
 - IBM i だけで何が、どこまでできるのか

※その他、検証して欲しい事があればアンケートにご記入ください



python™



IBM i の pase 環境を利用するにあたり、今回は chrome secure shell を利用しました

準備 : Chrome secure shell

準備 : Chrome Secure Shell

■ Chrome Secure Shell の利用

1. Chrome ブラウザで「chrome ウェブストア」を表示
 - <https://chrome.google.com/webstore>
2. 「ストアを検索」に “secure shell app” と入力し Enter
3. 一覧から “Secure Shell” (以下のアイコン)をクリック
4. 「Chrome に追加」をクリック
5. 確認画面で「拡張機能を追加」をクリック
6. Chrome の右上にある「拡張機能アイコン」をクリック
7. 表示された「接続ダイアログ」をクリック
8. ユーザー、IBM I の IPアドレス、ポート(22)を入力し [ENTER]接続
9. 接続後、パスワードを入力 (初回のみ接続確認があるので yes で回答)

準備 : Chrome Secure Shell

KIT@172.23.0.236:22

KIT 172.23.0.236 22

SSH 中継サーバーのオプション

ID: [default] インポート...

SSH 引数: その他のコマンドライン引数

現在のプロファイル: デフォルト

[DEL] 削除 オプション フィードバックを送信 SFTP [ENTER] 接続



```
KIT@172.23.0.236:22 - Secure Shell 0.35
Secure Shell バージョン 0.35 へようこそ。
よくある質問とその回答: https://goo.gl/muppJj (Ctrl キーを押しながらリンクをクリックして開きます)
変更ログ、リリースノート: /html/changelog.html

ヒント #14: Ctrl キー (Mac OS の場合は Cmd キー) を押しながら右クリックすると、コンテキスト メニ

NaCl プラグインを読み込んでいます... 完了しました。
KIT@172.23.0.236 に接続しています...
KIT@172.23.0.236's password:
bash-4.4$
```



事前に以下を対応しておくとう使いやすいです。
※ 5250 から実施

1. /home/<ユーザー>を作成
2. .profile 作成
3. .bashrc 作成

```
bash-4.4$ pwd
/home/kit
bash-4.4$ cat .profile
bash
bash-4.4$ cat .bashrc
alias python=python3
alias pip=pip3
export PATH=/Q0penSys/pkgs/bin:/Q0penSys/pkgs/sbin:$PATH
```