Pymee2019 第 1 回勉強会サンプルプログラム

コードについて

このプログラムはPymee2019 第 1 回勉強会のサンプルプログラムです。
2018年の勉強会で学んだこと+αで作成しています。
一部変な書き方なところがあるかもしれないですが、動いているので今回はよしとしてます。
また、一部説明にコードを書いていないばいいもあります。

※実行する際に以下のコマンドを実行すると開発者と同じ環境で実行出来ます。

```
source ./venv/bin/activate
pip install -r requirements.txt
```

各種ライブラリのインポート

```
import csv
import os
import socket
import string
import sys
from datetime import datetime

import paramiko
```

コマンドライン引数とCSVファイルのチェック

```
# コマンドライン引数の数のチェック
if len(sys.argv) <= 1:
    print("Usage: {} <csvfile>".format(sys.argv[0]), file=sys.stderr)
    sys.exit(1)

# csvファイルのパスを変数に格納
csv_file_path = sys.argv[1]

# コマンドライン引数で指定したcsvファイルがなければ終了させる
if not os.path.exists(csv_file_path):
    print("ERROR: CSVファイルが見つかりません", file=sys.stderr)
    sys.exit(1)

# CSVファイルが空だったら終了させる
if os.path.getsize(csv_file_path) == 0:
    print("ERROR: ファイルが空の可能性があります。", file=sys.stderr)
    sys.exit(1)
```

空のファイルの判定は、`elif os.path.getsize(csv_file_path) == 0:` で行っています。

Paramikoのインスタンス化

```
# Paramikoをインスタンス化して初期設定とか済ませておく
ssh = paramiko.SSHClient()
ssh.load_system_host_keys()
ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
```

Paramikoモジュールをインスタンス化してから各種設定をしています。

出力用テンプレートの作成

```
# 出力用のテンプレートを生成
templates_text = string.Template("""\
==============================
IPアドレス: $ipaddr
ログインユーザ: $username
実行時刻: $nowtime
==============================
$command
$command_result

""")
```

stringライブラリを使って出力用のテンプレートを定義しています。
`$ipaddr`とか`$username`の部分を簡単に置換出来ます。
結構便利なので覚えておくとそのうち役立つときがあるかもしれません。

サーバに接続してデータを取得(コード)

```
# CSVファイルを開いてIPアドレス、ユーザ名、パスワード、コマンドを取り出したあとにコマンドを実行して出力テキストを生成
with open(csv_file_path, 'r') as csv_file:
    csv_reader = csv.reader(csv_file)

                # CSVファイルの行数分繰り返す
                # SSHに必要な情報を取得し、変数に代入
    for row in csv_reader:
        if not (row[0] == "" and row[1] == "" or row[0] == "" and row[1] == "" and row[2] == ""):
            hostname = row[0]
            username = row[1]
            password = row[2]
        command = row[3]

        # サーバに接続
        try:
            ssh.connect(hostname=hostname, username=username, password=password,
                        port=22, timeout=10.0, look_for_keys=False)
        except paramiko.ssh_exception.AuthenticationException as e:
            print("ERROR: SSH認証に失敗しました。", file=sys.stderr)
            sys.exit(1)
        except paramiko.ssh_exception.NoValidConnectionsError:
            print("ERROR: 接続エラー。", file=sys.stderr)
            sys.exit(1)
        except socket.timeout:
            print("ERROR: タイムアウトしました。", file=sys.stderr)
            sys.exit(1)

        # ホスト名を取得 Linuxの $HOSTNAME 環境変数でホスト名を取得
        stdin, stdout, stderr = ssh.exec_command("echo $HOSTNAME")
        server_hostname = stdout.readline().strip("\n")

        # プロンプトの構築
        prompt = "[{user}@{hostname}]".format(user=username, hostname=server_hostname)

        # 三項演算子を使用して一行でif-elseを書く
        prompt += "# " if username == "root" else "$ "

        # コマンド発行
        stdin, stdout, stderr = ssh.exec_command(command)

        command_result = ""

                                # stdoutからコマンド実行結果を取り出し、出力ファイル用に整形
        for i in stdout:
            command_result += i.strip('\n') + '\n'

        contents += templates_text.substitute(ipaddr=hostname, username=username,
                                              nowtime=datetime.now().strftime("%H:%M"),
                                              command=prompt+command, command_result=command_result)
```

サーバに接続してデータを取得(解説)

今回のプログラムで一番重要なところ(?)です。主に以下の順番で処理しています。

1. csvファイルを開いて`csv.reader`を使ってカンマ区切りで2次元のリストを作成
2. for文を使ってリストから一行ずつ取り出して、hostnameとかを代入する。
3. インスタンス化済みのparamikoでサーバにSSH接続する。※例外処理については、のちほど記載しています
4. ホスト名の取得してPROMPTを構築
5. コマンドを実行
6. 実行結果を取り出して`contents`変数に格納
7. 念の為接続をクローズ

例外処理について

1. paramiko.ssh_exception.AuthenticationException
    - この部分は認証エラーの際に例外をキャッチします
2. paramiko.ssh_exception.NoValidConnectionsError
    - この部分は接続エラーの際に例外をキャッチします
3. socket.timeout
    - この部分はタイムアウトの際に例外をキャッチします。
    - SSH接続をする際の引数にtimeoutを指定することにより使えます

保存するファイル名を生成

```
# 保存ファイル名の定義（全角にしたのは定数として認識しやすくするためで、Pythonでは特に意味は持たない）
SAVE_FILE_NAME = "out_{}.txt".format(datetime.now().strftime("%Y%m%d%H%M"))
```

保存処理

```
# 保存処理
try:
    with open(SAVE_FILE_NAME, 'x', encoding='UTF-8') as f:
        f.write(contents)

    print("保存完了！")
except FileExistsError:
    print("ファイル名 [{}] は重複しています。".format(SAVE_FILE_NAME), file=sys.stderr)
    sys.exit(1)
```