Reading and Writing Database Files

In this lab, you will:

- Read device information from a JSON file
- Create device objects for the network devices in your lab environment
- Connect to each device
- Get interface information
- Write the device information into a database file

You will then read the information from the database file you have just created, in order to insure that the data has been written correctly.

Step 1

Read device information from a JSON file, and create a list of network device objects holding connectivity information for each device.

```
#=============================================================================
def read_devices_info(devices_file):

    devices_list = []

    # Open the device file with JSON data and read into string
    json_file = open(devices_file,'r')    # open the JSON file
    json_device_data = json_file.read()  # read in the JSON data from file

    # Convert JSON string into Python data structure
    devices_info_list = json.loads(json_device_data)

    for device_info in devices_info_list:

        # Create a device object with this data
        if device_info['os'] == 'ios':

            device = NetworkDeviceIOS(device_info['name'],device_info['ip'],
                                      device_info['user'],device_info['password']

        elif device_info['os'] == 'ios-xr':

            device = NetworkDeviceXR(device_info['name'],device_info['ip'],
                                      device_info['user'],device_info['password']

        else:
            device = NetworkDevice(device_info['name'],device_info['ip'],
                                       device_info['user'],device_info['password

        devices_list.append(device) # Append this device object to list

    return devices_list
```

Step 2

Connect to each device and get interface information, printing device information at each iteration.

```
#---- Class to hold information about an IOS network device --------
class NetworkDeviceIOS(NetworkDevice):

    def __init__(self, name, ip, user='cisco', pw='cisco'):
        NetworkDevice.__init__(self, name, ip, user, pw)
        self.os_type = 'ios'

    def connect(self):
        print '--- connecting IOS: telnet '+self.ip_address

        self.session = pexpect.spawn('telnet '+self.ip_address, timeout=20)
        result = self.session.expect(['Username:', pexpect.TIMEOUT])

        self.session.sendline(self.username)
        result = self.session.expect('Password:')

        # Successfully got password prompt, logging in with password
        self.session.sendline(self.password)
        self.session.expect('>')

    def get_interfaces(self):

        self.session.sendline('show interfaces summary')
        result = self.session.expect('>')

        self.interfaces = self.session.before
```

```
#=====================================================================
def write_devices_db(devices_db_file, devices_list):

    print '---- writing devices to db ------------------------------'

    # Connect to the database and get a connection
    db_connection = sqlite3.connect(devices_db_file) # DB connection
    db_cursor = db_connection.cursor()               # DB cursor

    # Create our devices table in the database we just opened
    db_cursor.execute('''CREATE TABLE IF NOT EXISTS devices
                                (name VARCHAR(16) PRIMARY KEY,
                                 ip   VARCHAR(16),
                                 os   VARCHAR(8),
                                 user VARCHAR(16),
                                 pw   VARCHAR(16))''')

    # Iterate through devices, printing one per line in devices table
    for device in devices_list:

        # Insert or replace the device information into devices table
        sql_cmd = 'REPLACE INTO devices (name,ip,os,user,pw) VALUES(?,?,?,?,
        db_cursor.execute(sql_cmd, (device.name,
                                    device.ip_address,
                                    device.os_type,
                                    device.username,
                                    device.password))

    db_connection.commit() # Must commit changes or they are not saved!

    db_cursor.close()
    db_connection.close()
```

Step 4

Use the sqlite3 library to read information from the database file.

Answer

```
#=====================================================================
def read_devices_db(devices_db_file):

    print '---- reading devices from db ---------------------------'

    # Connect to the database and get a connection
    db_connection = sqlite3.connect(devices_db_file) # DB connection
    db_cursor = db_connection.cursor()               # DB cursor

    db_cursor.execute('SELECT * FROM devices')
    devices_from_db = db_cursor.fetchall()

    pprint(devices_from_db)
    print '---- end devices from db -------------------------------'

    # Done reading devices from the table, close it down.
    db_cursor.close()
    db_connection.close()

    return devices_from_db
```

Step 5

Print the device information that has just been read from the database.

Answer

```
pprint(devices_from_db)
```

Step 6

Run your application and verify that the data was read from the database.

Answer

Your complete application should look similar to:

file: util.py

```
import sqlite3
import json

from pprint import pprint

from devclass import NetworkDevice
from devclass import NetworkDeviceIOS
from devclass import NetworkDeviceXR

#=============================================================================
def read_devices_info(devices_file):

    devices_list = []

    # Open the device file with JSON data and read into string
    json_file = open(devices_file,'r')    # open the JSON file
    json_device_data = json_file.read()  # read in the JSON data from file

    # Convert JSAON string into Python data structure
    devices_info_list = json.loads(json_device_data)

    for device_info in devices_info_list:

        # Create a device object with this data
        if device_info['os'] == 'ios':

            device = NetworkDeviceIOS(device_info['name'],device_info['ip'],
                                      device_info['user'],device_info['passw
```

```
        elif device_info['os'] == 'ios-xr':

            device = NetworkDeviceXR(device_info['name'],device_info['ip'],
                                     device_info['user'],device_info['passwo

        else:
            device = NetworkDevice(device_info['name'],device_info['ip'],
                                   device_info['user'],device_info['password

        devices_list.append(device) # Append this device object to list

    return devices_list



#=====================================================================
def print_device_info(device):

    print '-------------------------------------------------------'
    print '    Device Name:      ',device.name
    print '    Device IP:        ',device.ip_address
    print '    Device username:  ',device.username,
    print '    Device password:  ',device.password
    print '-------------------------------------------------------'
```

```
#=============================================================================
def write_devices_db(devices_db_file, devices_list):

    print '---- writing devices to db ------------------------------'

    # Connect to the database and get a connection
    db_connection = sqlite3.connect(devices_db_file) # DB connection
    db_cursor = db_connection.cursor()               # DB cursor

    # Create our devices table in the database we just opened
    db_cursor.execute('''CREATE TABLE IF NOT EXISTS devices
                                (name VARCHAR(16) PRIMARY KEY,
                                 ip   VARCHAR(16),
                                 os   VARCHAR(8),
                                 user VARCHAR(16),
                                 pw   VARCHAR(16))''')

    # Iterate through devices, printing one per line in devices table
    for device in devices_list:

        # Insert or replace the device information into devices table
        sql_cmd = 'REPLACE INTO devices (name, ip, os, user, pw) VALUES(?,?,
        db_cursor.execute(sql_cmd, (device.name,
                                    device.ip_address,
                                    device.os_type,
                                    device.username,
                                    device.password))

    db_connection.commit() # Must commit changes or they are not saved!

    db_cursor.close()
    db_connection.close()
```

```
#=====================================================================
def read_devices_db(devices_db_file):

    print '---- reading devices from db ----------------------------'

    # Connect to the database and get a connection
    db_connection = sqlite3.connect(devices_db_file) # DB connection
    db_cursor = db_connection.cursor()               # DB cursor

    db_cursor.execute('SELECT * FROM devices')
    devices_from_db = db_cursor.fetchall()

    pprint(devices_from_db)
    print '---- end devices from db --------------------------------'

    # Done reading devices from the table, close it down.
    db_cursor.close()
    db_connection.close()

    return devices_from_db
```

file: devclass.py

```
import pexpect

#---- Class to hold information about a generic network device --------
class NetworkDevice():

    def __init__(self, name, ip, user='cisco', pw='cisco'):
        self.name = name
        self.ip_address = ip
        self.username = user
        self.password = pw
        self.os_type = None

    def connect(self):
        self.session = None

    def get_interfaces(self):
        self.interfaces = '--- Base Device, does not know how to get interfa
```

```
#---- Class to hold information about an IOS network device --------
class NetworkDeviceIOS(NetworkDevice):

    def __init__(self, name, ip, user='cisco', pw='cisco'):
        NetworkDevice.__init__(self, name, ip, user, pw)
        self.os_type = 'ios'

    def connect(self):
        print '--- connecting IOS: telnet '+self.ip_address

        self.session = pexpect.spawn('telnet '+self.ip_address, timeout=20)
        result = self.session.expect(['Username:', pexpect.TIMEOUT])

        self.session.sendline(self.username)
        result = self.session.expect('Password:')

        # Successfully got password prompt, logging in with password
        self.session.sendline(self.password)
        self.session.expect('>')

    def get_interfaces(self):

        self.session.sendline('show interfaces summary')
        result = self.session.expect('>')

        self.interfaces = self.session.before
```

```
#---- Class to hold information about an IOS-XR network device --------
class NetworkDeviceXR(NetworkDevice):

    #---- Initialize -------------------------------------------------
    def __init__(self, name, ip, user='cisco', pw='cisco'):
        NetworkDevice.__init__(self, name, ip, user, pw)
        self.os_type = 'ios-xr'

    #---- Connect to device ------------------------------------------
    def connect(self):

        print '--- connecting XR: ssh '+self.username+'@'+self.ip_address

        self.session = pexpect.spawn('ssh '+self.username+
                                     '@'+self.ip_address, timeout=20)
        result = self.session.expect(['password:', pexpect.TIMEOUT])

        # Check for failure
        if result != 0:
            print '--- Timout or unexpected reply from device'
            return 0

        # Successfully got password prompt, logging in with password
        print '--- password:',self.password
        self.session.sendline(self.password)
        self.session.expect('#')

    #---- Get interfaces from device ---------------------------------
    def get_interfaces(self):

        self.session.sendline('show interface brief')
        result = self.session.expect('#')

        self.interfaces = self.session.before
```

file: main.py

```
from util import read_devices_info
from util import print_device_info
from util import write_devices_db
from util import read_devices_db

from pprint import pprint

#=====================================================================
# Main program: connect to device, show interface, display

devices_list = read_devices_info('json-devices')  # read JSON info for all d

for device in devices_list:

    print '==== Device ========================================================

    device.connect()          # connect to this specific device
    device.get_interfaces()   # get interface info for this specific device

    print_device_info(device)   # print device details for this device

devices_db_file = 'devices.db'
write_devices_db(devices_db_file, devices_list)   # write device data to dat

# Now read in the device information we just wrote
devices_from_db = read_devices_db(devices_db_file)

print '==== Device info from database ======================================
pprint(devices_from_db)
```