

Python

Login and list your domains

```
#!/usr/bin/env python
import requests

dmapiURL = 'https://dmapi.ote.joker.com'
dmapiUser = 'username'
dmapiPassword = 'password'

def main():
    loginResponse = login(dmapiUser, dmapiPassword)
    print "Login: Status-Code:", loginResponse.header['Status-Code']
    if loginResponse.header['Status-Code'] != '0':
        print loginResponse.header['Status-Text']
        return

    sessionId = loginResponse.header['Auth-Sid'];
    print ""
    domainResponse = domainList(sessionId, 1, 5)
    print "Domain List: Status-Code:", domainResponse.header['Status-Code']
    domains = domainResponse.resultListWithNames()
    for domain in domains:
        for key, value in domain.iteritems():
            print " %s: %s" % (key, value)
        print ""
    logoutResponse = logout(sessionId)
    print "Logout: Status-Code:", logoutResponse.header['Status-Code']

# implement dmapi commands as functions
def login(username, password):
    parameters = { 'username': username, 'password': password }
    message = sendCommand('login', parameters)
    return message;

def logout(sessionId):
    parameters = { 'auth-sid': sessionId }
    message = sendCommand('logout', parameters)
    return message;

def domainList(sessionId, list_from=1, list_to=""):
    parameters = { 'auth-sid': sessionId, 'from': list_from, 'to': list_to }
    message = sendCommand('query-domain-list', parameters)
    return message;

# general dmapi command call
def sendCommand(command, parameter={}):
    try:
```

Python

```
url = dmapiURL+'/request/'+command
    print "Request-URL: ", url
    response = requests.get(url, params=parameter)
    # print URL with parameters for debugging purposes
    # print "Request-URL: ", response.url
    if response.status_code != requests.codes.ok:
        raise CommandError("Command Failed! HTTP Status Code: %s" % r
response.status_code)
    return DmapiResponse(response.text)
except requests.ConnectionError as e:
    raise CommandError("Connection Error: %s" % str(e))
except requests.HTTPError as e:
    raise CommandError("Http Error: %s" % str(e))
except CommandError as e:
raise
except Exception as e:
    raise CommandError("Unexpected Error: %s" % str(e))

class DmapiResponse():
    def __init__(self,responseBody):
        parts = responseBody.split("\n\n",1)
        if len(parts)>0:
            self.header = self.__parseKeyValueList(parts[0])
        if len(parts)>1:
            self.body = parts[1]

    def __parseKeyValueList(self,text):
        lines = text.split("\n")
        keyValueList = {}
        for line in lines:
            keyValue = line.split(' ',1)
            key = keyValue[0].rstrip(':')
            value = keyValue[1]
            keyValueList[key] = value
        return keyValueList

    def __getSeparator(self):
        if self.header.has_key('Separator') and self.header['Separator
'] == 'TAB':
            return "\t"
        else:
            return " "

    def resultList(self):
        lines = self.body.split("\n")
        resultList = []
        separator = self.__getSeparator()
        for line in lines:
            values = line.split(separator)
```

Python

```
        resultList.append(line.split(separator))
    return resultList

def resultListWithNames(self):
    columnNames = self.resultListColumns()
    resultList = []
    if len(columnNames) > 0:
        rawList = self.resultList()
        resultList = []
        for row in rawList:
            columns = {}
            for idx, column in enumerate(row):
                columns[columnNames[idx]] = column
            resultList.append(columns)
    return resultList

def resultListColumns(self):
    if self.header.has_key('Columns'):
        columnsText = self.header['Columns']
        columns = columnsText.split(',')
        return columns
    else:
        return []

def resultValues(self):
    return self.__parseKeyValueList(self.body)

class CommandError(Exception):
    def __init__(self, value):
        self.value = value
    def __str__(self):
        return repr(self.value)

# call main function
try:
    main()
except CommandError as e:
    print "Error:", str(e).strip("'")
```

Unique solution ID: #2470

Author: Joker.com

Last update: 2017-10-20 11:39