During my current assignment I was asked if I could provide the data table parameters I used in all my HP Quick Test Pro test cases. Normally this would mean opening QTP, export the data table to an Excel file and open the next test case. Since I had to do this for more than 200+ test cases and they were all on the HP Quality Centre server (which means the test cases would open even slower than usual) I tried to find a faster solution.

I used Python 3.4 and the Quality Center OTA API to access the data directly on the QC server and download it to a local drive.

First the Python class to connect to QC OTA:This class returns the connection object so we can now use it to retrieve all the testcase data in our script.

```
'''
@author: Michel Lalmohamed
'''
from win32com.client import Dispatch
class QC_ConnectorClass(object):
    '''
    Connector Class which returns as the connection object for further QC API use
    '''
    def __init__(self):
        print("class init")

    # This function creates a connection to the HP QC server and logs in to the server by
using the
    # user credentials
    def ConnectToQC(self, qcServer, qcUser, qcPassword, qcDomain, qcProject):
        #HP QC OTA methods
        self.TD = Dispatch("TDApiOle80.TDConnection.1")
        self.TD.InitConnectionEx(qcServer)
        self.TD.Login(qcUser,qcPassword)
        self.TD.Connect(qcDomain,qcProject)

        print("Logged in")

        #Return the connection object
        return self.TD
```

This class returns the connection object so we can now use it to retrieve all the testcase data in our script.

```
'''
Created on 17 feb. 2015
@author: Michel Lalmohamed
'''
import time
from QC_Connector import QC_ConnectorClass

#QC login user credentials
qcServer = "[http://qc11.*******.**/qcbin"]http://qc11.*******.**/qcbin";
qcUser = "milalmohamed"
qcPassword = ********
qcDomain = *******
qcProject = *******

#Connection is being handled by the ConnectorClass & ConnectToQC method.
#Returns the QC connection object
QC_ConnectedObj = QC_ConnectorClass().ConnectToQC(qcServer, qcUser, qcPassword, qcDomain,
qcProject)

#Grab the entire folder tree structure from the QC project as an object. This gives the user
all the power to search the entire project structure
TreeObject = QC_ConnectedObj.TreeManager

#Specify which folder and sub folders you want to search in but because we grabbed the entire
tree a high level folder is already sufficient!
folder = TreeObject.NodeByPath("Subject<the main folder which contains all the tests>")


#Name prefix of the tests you want to find. We don't have to now the entire name of the test.
#A partial name will already return the correct result. In this situation a test case prefix
is enough to get ALL the cases. Just like we want, a complete list with test cases.
testList = folder.FindTests(<search prefix>")

if (len(testList)) == 0:
    print("No Tests found")
else:
    #Loop through all the test objects in the list
    for test in range(len(testList)):
        print (str(testList[test].ID) + "; " + testList[test].name)
        #grab all test data from the test including the data sheet en script files
        TestStorage = testList[test].ExtendedStorage
        #set path to download files
        TestStorage.ClientPath = 'c:\\QC_Extract\\' + testList[test].name
        #download all the test case data
        TestStorage.Load("", True)
        #progress tracker
        print(len(testList) - test )
        #pause to not overload the server
        time.sleep(5)

QC_ConnectedObj.Logout()

print("Logged out")
```

All the QTP test case data is now stored in the specified folder. Which in my case meant writing a simple script to extract the default.xls, which contained the parameter data, from every folder and rename it.

Please take caution when extracting lots of data from the QC server. Especially the ExtendedStorage can cause the server to crash. Hence my little wait function to keep IT support of my back.