```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# movoirVOT2FITSconverter

import os
import sampy
import urllib
import pdb

downloadFolder="/Users/jdsant/Downloads/"

vot2fitsMD = {
    'samp.name': 'movoirV2FITS',
    'samp.description.text': 'Conversor de VOTable a FITS',
    'movoir.version': '0.1',
    'samp.icon.url': 'http://www.iaa.es/~jdsant/thesis/v2fitsIcon.png'
}

def convertVotable2FITS(votableUrl, pFileName):
    print "Entering convertVotable2FITS", votableUrl, pFileName
    # pdb.set_trace()
    # Download url 2 temp folder
    extension = '.fits'
    fileName = ""
    if pFileName=="":
        fileName = votableUrl.split('/')[-1]
    else:
        fileName = urllib.quote(pFileName)

    print "Filename:", fileName
    curlCommand = "/usr/bin/curl -s -o "+downloadFolder+fileName+" '"+votableUrl+"'"
    print "Executing command:\n"+curlCommand+"\n"
    osCode = os.system(curlCommand)
    if osCode != 0:
        print "Download failed"
    else:
        # We succeeded
        # Lets convert!
        stiltsCommand = "./stilts tcopy "+downloadFolder+fileName
        stiltsCommand = stiltsCommand+" "+downloadFolder+fileName+extension
        stiltsCommand = stiltsCommand+" ifmt=votable ofmt=fits"
        print "Executing command:\n"+stiltsCommand+"\n"
        os.system(stiltsCommand)
        os.system("open "+downloadFolder)

    # Call STILTS to convert file
    # Open folder


def notificationHandler(private_key, sender_id, mtype, params, extra):
    #pdb.set_trace()
    print ("notifHandler", private_key, sender_id, mtype, params, extra)
    if mtype == "table.load.votable":
        if params.has_key('url'):
            tableName=""
            if params.has_key('name'):
                tableName=params['name']
            print "In the good branch",params['url'], tableName
            convertVotable2FITS(params['url'], tableName)



def callHandler(private_key, sender_id, msg_id, mtype, params, extra):
    # As this is a call, we should return an empty samp.result
    # after having processed it
    print ("callHandler: ", private_key, sender_id, msg_id, mtype, params, extra)
    if mtype == "table.load.votable":
        if params.has_key('url'):
            tableName=""
            if params.has_key('name'):
                tableName=params['name']
            print "In the good branch",params['url'], tableName
            convertVotable2FITS(params['url'], tableName)
            replyMsg = {'samp.status': sampy.SAMP_STATUS_OK,
                        'samp.result': {}}
            vot2fitsClient.reply(msg_id, replyMsg)
        else:
            print "In the error branch"
            replyMsg = {'samp.status': sampy.SAMP_STATUS_ERROR,
                        'samp.result': {},
                        'samp.error': {'samp.errortxt': 'No url parameter provided.'}}
            vot2fitsClient.reply(msg_id, replyMsg)



def responseHandler(private_key, sender_id, msg_id, response):
    # Response received, typically an OK response,
    # we should not be receiving none of these
    print ("Receiving response", private_key, sender_id, msg_id, response)



# Start binding behaviours
vot2fitsClient = sampy.SAMPIntegratedClient(vot2fitsMD)
vot2fitsClient.connect()
vot2fitsClient.bindReceiveNotification("table.load.votable", notificationHandler)
vot2fitsClient.bindReceiveCall("table.load.votable", callHandler)
#movoirVOT2FITSconverter.vot2fitsClient.disconnect()
```