Service Now App Sync with Agile Central

Contents

[APP\_SYNC\_MANAGER.bat 2](#_Toc96423731)

[APP\_SYNC\_DOWNLOAD\_P.bat 3](#_Toc96423732)

[Python Scripts 7](#_Toc96423733)

[Download\_app\_list.py 7](#_Toc96423734)

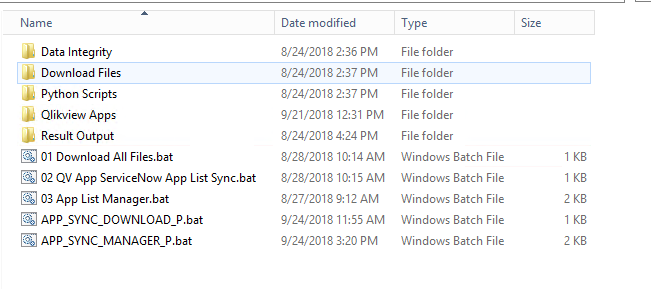
[app\_list\_manager.py 10](#_Toc96423735)

[File Download Scripts 18](#_Toc96423736)

[Agile\_central\_app\_list\_download.py 18](#_Toc96423737)

[Service\_now\_app\_sync\_sn\_download.py 20](#_Toc96423738)

[config.json 23](#_Toc96423739)



# APP\_SYNC\_MANAGER.bat

@echo off

REM Start iexplore.exe

REM echo Internet Explorer Started!

REM cd C:\Program Files (x86)\Python37-32\

REM Feature List

python.exe "D:/Service Now App Sync/Python Scripts/Drop Down Multi Value Manager/app\_list\_manager.py" "D:/Service Now App Sync/Python Scripts/Drop Down Multi Value manager/config\_feature/config.json" "D:/Service Now App Sync/Python Scripts/Drop Down Multi Value manager/config\_feature/ac\_desired\_values.txt" "D:/Service Now App Sync/Python Scripts/Drop Down Multi Value Manager/config\_feature/logging.json" >"D:\Service Now App Sync\Result Output\Feature\_output.txt"

echo Feature List Done!

REM User Story List

python.exe "D:/Service Now App Sync/Python Scripts/Drop Down Multi Value Manager/app\_list\_manager.py" "D:/Service Now App Sync/Python Scripts/Drop Down Multi Value manager/config\_user\_story/config.json" "D:/Service Now App Sync/Python Scripts/Drop Down Multi Value manager/config\_feature/ac\_desired\_values.txt" "D:/Service Now App Sync/Python Scripts/Drop Down Multi Value Manager/config\_feature/logging.json" >"D:\Service Now App Sync\Result Output\User\_Story\_output.txt"

echo User Story List Done!

REM cd C:\Program Files (x86)\Python36\

REM python.exe "D:/Service Now App Sync/Python Scripts/File Download Scripts/agile\_central\_app\_list\_download.py"

REM echo Application list from Agile Central Downloaded !

REM cd C:\Program Files (x86)\Python36\

REM python.exe "D:/Service Now App Sync/Python Scripts/File Download Scripts/service\_now\_app\_sync\_sn\_download.py"

REM echo Script Downloaded ServiceNow Files Successfully!

# APP\_SYNC\_DOWNLOAD\_P.bat

@echo off

Taskkill /IM iexplore.exe /F

Start iexplore.exe

echo Internet Explorer Started!

REM cd C:\Program Files (x86)\Python36\

python.exe "D:/Service Now App Sync/Python Scripts/File Download Scripts/agile\_central\_app\_list\_download.py" "D:/Service Now App Sync/Python Scripts/File Download Scripts/config/config.json"

echo Application list from Agile Central Downloaded !

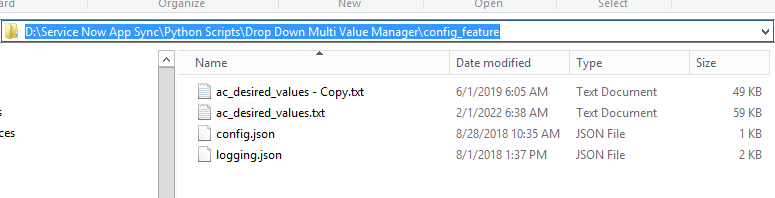
REM cd C:\Program Files (x86)\Python36\

python.exe "D:/Service Now App Sync/Python Scripts/File Download Scripts/service\_now\_app\_sync\_sn\_download.py"

echo Scripts Ran Successfully

REM pause

D:\Service Now App Sync\Python Scripts\Drop Down Multi Value Manager\config\_feature



Config.json

{

"preview": false,

"server": "rally1.rallydev.com",

"workspace":"Unum",

"project":"Unum Group",

"element\_name":"PortfolioItem",

"attribute\_name": "App\_ID",

"values\_have\_key": true,

"threshold":1450,

"proxy":"10.215.109.16",

"apikey":"\_zdCoOTvjQKGUcFoE9KkYXvnxrE0iTk3UqQftTGnst9A",

"output\_result\_path": "D:/Service Now App Sync/Result Output/Feature\_output.txt"

}

Logging.json

{

"version": 1,

"disable\_existing\_loggers": false,

"formatters": {

"brief": {

"class": "logging.Formatter",

"datefmt": "%I:%M:%S",

"format": "%(levelname)-8s; %(name)-15s; %(message)s"

},

"single-line": {

"class": "logging.Formatter",

"datefmt": "%I:%M:%S",

"format": "%(levelname)-8s; %(asctime)s; %(name)-15s; %(module)s:%(funcName)s;%(lineno)d: %(message)s"

},

"verbose": {

"class": "logging.Formatter",

"datefmt": "%I:%M:%S",

"format": "%(levelname)-8s; [%(process)d]; %(threadName)s; %(name)-15s; %(module)s:%(funcName)s;%(lineno)d: %(message)s"

},

"multiline": {

"class": "logging.Formatter",

"format": "Level: %(levelname)s\nTime: %(asctime)s\nProcess: %(process)d\nThread: %(threadName)s\nLogger: %(name)s\nPath: %(module)s:%(lineno)d\nFunction :%(funcName)s\nMessage: %(message)s\n"

}

},

"handlers": {

"console": {

"level": "DEBUG",

"class": "logging.StreamHandler",

"formatter": "brief",

"stream": "ext://sys.stdout"

},

"file": {

"level": "DEBUG",

"class": "logging.handlers.RotatingFileHandler",

"formatter": "single-line",

"filename": "D:/Service Now App Sync/Python Scripts/Drop Down Multi Value Manager/log/mv\_manager.log",

"mode": "a",

"encoding": "utf-8",

"maxBytes": 10485760,

"backupCount": 5

}

},

"loggers": {

"ValueManager": {

"level": "DEBUG",

"handlers": ["console", "file"],

"propagate": false

}

},

"root": {

"handlers": ["console", "file"],

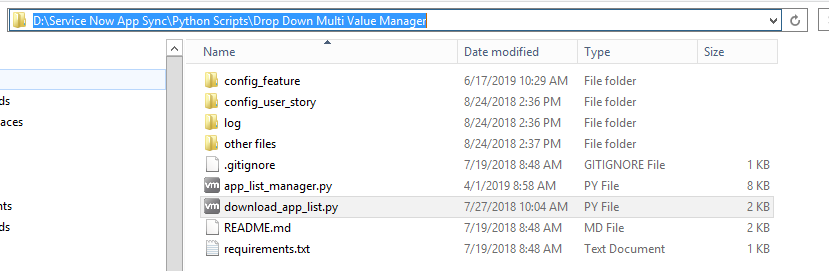
"level": "ERROR"

}

}

# Python Scripts

D:\Service Now App Sync\Python Scripts\Drop Down Multi Value Manager



# Download\_app\_list.py

#written by Nabin

#07/20/2018

import os

import sys

import json

from array import \*

from pyral import Rally, rallyWorkset, rallySettings

class DownloadMulitValuesList:

def \_\_init\_\_(self, rally):

self.rally = rally

return

def application\_list(self, file\_name, element\_name, attribute\_name):

response = self.rally.get('TypeDefinition', fetch=True, query='ElementName = "{0}"'.format(element\_name))

#response = self.rally.getAllowedValues(element\_name,attribute\_name)

if response.errors:

print(response.errors)

element\_typedef = response.next()

attributes = element\_typedef.Attributes

attribute = [attribute for attribute in attributes if attribute.Name == attribute\_name][0]

attribute\_definition\_objectid = attribute.ObjectID # Post this to / create to add a value

#Get the AllowedValue objects

allowed\_values = attribute.AllowedValues

file\_path = r'D:\ClarityDataAutomation\Download Files\\' + file\_name +'.csv'

with open(file\_path, 'w', newline='', encoding='utf-8') as fp:

for value in allowed\_values:

print(value.StringValue)

app\_name = value.StringValue + '\n'

fp.write(app\_name)

fp.close

print('Downloaded '+ file\_name + '!')

if \_\_name\_\_ == '\_\_main\_\_':

#main()

config\_filename = sys.argv[1]

with open(config\_filename,"r") as config\_file:

config = json.load(config\_file)

os.environ["HTTPS\_PROXY"]=config['proxy']

options =[opt for opt in sys.argv[1:] if opt.startswith ('--')]

server, username, pwd, workspace, project, element\_name = rallyWorkset(options)

rally = Rally(config['server'], config['username'], config['pwd'], workspace=config['workspace'], project=config['project'])

app\_list = DownloadMulitValuesList(rally)

#app\_list.application\_list(config['file\_name'], config['element\_name'], config['attribute\_name'])

app\_list.application\_list(config['file\_name'], config['element\_name'], config['attribute\_name'])

# app\_list\_manager.py

import os

import sys

import json

from pyral import Rally, rallyWorkset

import logging, logging.config

import smtplib

from email.message import EmailMessage

from email.mime.text import MIMEText

class ListManager:

preview = True

def \_\_init\_\_(self, rally, preview=True, values\_has\_key=False):

self.rally = rally

self.preview = preview

self.values\_has\_key= values\_has\_key

if (self.preview):

logger.info("PREVIEW MODE")

return

def load(self, element\_name, attribute\_name, desired\_value\_filename):

desired\_value\_names = self.load\_desired\_values(desired\_value\_filename)

#print(desired\_value\_names)

response =self.rally.get('TypeDefinition', fetch=True, query='ElementName = "{0}"'.format(element\_name))

element\_typedef = response.next()

attributes = element\_typedef.Attributes

attribute = [attribute for attribute in attributes if attribute.Name == attribute\_name][0]

attribute\_definition\_objectid = attribute.ObjectID

allowed\_values = attribute.AllowedValues

values\_to\_modify = self.values\_to\_modify(allowed\_values, desired\_value\_names)

old\_new\_values = [(value.LocalizedStringValue, desired\_value) for (value, desired\_value) in values\_to\_modify]

logger.info("Values to modify: %s", len(values\_to\_modify))

for (value, desired\_name) in values\_to\_modify:

self.modify\_value(value, desired\_name)

allowed\_names\_to\_modify =[old for (old,new) in old\_new\_values]

new\_names\_to\_modify = [new for (old,new) in old\_new\_values]

value\_objects\_to\_remove = self.value\_objects\_to\_remove(allowed\_values, desired\_value\_names)

value\_objects\_to\_remove =[value for value in value\_objects\_to\_remove if value.LocalizedStringValue not in allowed\_names\_to\_modify]

logger.info("Values to remove: %s", len(value\_objects\_to\_remove))

for value in value\_objects\_to\_remove:

self.remove\_value(value)

names\_to\_add = self.value\_names\_to\_add(allowed\_values, desired\_value\_names)

names\_to\_add = [name for name in names\_to\_add if name not in new\_names\_to\_modify]

logger.info("Values to add: %s", len(names\_to\_add))

for name in names\_to\_add:

self.add\_value(attribute\_definition\_objectid, name)

return

def add\_value(self,attribute\_definition\_objectid, name):

logger.info("Adding '%s'", name)

if not self.preview:

self.rally.create('AllowedAttributeValue',{

"StringValue": name,

"AttributeDefinition": '/attributedefinition/{0}'.format(attribute\_definition\_objectid)

})

return

def value\_names\_to\_add(self, allowed\_values ,desired\_value\_names):

result=[]

allowed\_value\_names = [value.LocalizedStringValue for value in allowed\_values]

result = [value for value in desired\_value\_names if value not in allowed\_value\_names]

return result

def remove\_value(self,value):

logger.info("Removing '%s'", value.LocalizedStringValue)

if not self.preview:

self.rally.delete('AllowedAttributeValue',value.ObjectID)

return

def value\_objects\_to\_remove(self,allowed\_values, desired\_value\_names):

result =[]

result = [value for value in allowed\_values if value.LocalizedStringValue not in desired\_value\_names and value.LocalizedStringValue !='']

return result

def modify\_value(self,value, desired\_name):

logger.info("Modifying '%s' to new value '%s'", value.LocalizedStringValue, desired\_name)

if not self.preview:

self.rally.post('AllowedAttributeValue',{

"ObjectID": value.ObjectID,

"StringValue": desired\_name

})

return

def load\_desired\_values(self, filename):

desired\_values = []

with open(filename, "r", encoding='utf-8-sig') as file:

for line in file:

line = line.strip()

desired\_values.append(line)

return desired\_values

def values\_to\_modify(self, allowed\_values, desired\_value\_names):

result =[]

desired\_value\_keys ={}

if self.values\_has\_key:

for name in desired\_value\_names:

key = self.get\_key(name)

if key:

desired\_value\_keys[key] = name

for allowed\_value in allowed\_values:

allowed\_name = allowed\_value.LocalizedStringValue

key = self.get\_key(allowed\_name)

if key and key in desired\_value\_keys:

desired\_name = desired\_value\_keys[key]

if desired\_name != allowed\_name:

result.append((allowed\_value, desired\_name))

return result

def get\_key(self,name):

split = name.split(' ')

if len(split) > 1:

return split[0]

else:

return None

def send\_email\_notification(txt\_file):

#with open(txt\_msg) as msgFile1:

# Create a text/plain message

#msg = MIMEText(msgFile1.read())

#msg = MIMEText(txt\_msg)

with open(txt\_file) as fp:

msg = EmailMessage()

msg.set\_content(fp.read())

sender = 'donotreply@unum.com'

receivers= ['nshrestha@unum.com','nshrestha@unum.com']

msg['Subject']= 'PROD:Agile Central App Synced Result Output!'

msg['From'] = sender

msg['To'] =",".join(receivers)

try:

# Send the message via our own SMTP servers

# s = smtplib.SMTP('localhost')

s = smtplib.SMTP('mail.unum.com',25)

s.sendmail(sender, receivers, msg.as\_string())

s.quit()

except Exception:

print("Error: unable to send email")

if \_\_name\_\_ == '\_\_main\_\_':

config\_filename = sys.argv[1]

desired\_value\_filename = sys.argv[2]

config\_logging\_file\_name = sys.argv[3]

#config\_logging\_file\_name ='D:/Service Now App Sync/Python Scripts/Drop Down Multi Value Manager/config\_feature/logging.json'

with open(config\_logging\_file\_name,"r", encoding="utf-8") as fd:

logging.config.dictConfig(json.load(fd))

logger = logging.getLogger("ValueManager")

logger.info("Starting...")

with open(config\_filename, "r") as config\_file:

config = json.load(config\_file)

count =0

with open(desired\_value\_filename,"r") as count\_line:

#count = sum(1 for line in count\_line)

for line in count\_line:

count+=1

logger.info("count: %s", count)

os.environ["HTTPS\_PROXY"] = config['proxy']

options = [opt for opt in sys.argv[1:] if opt.startswith('--')]

#server, username, pwd, workspace,project, element\_name = rallyWorkset(options)

#rally = Rally(config['server'], config['username'], config['pwd'], workspace=config['workspace'], project=config['project'])

server, apikey, workspace,project, element\_name,attribute\_name = rallyWorkset(options)

rally = Rally(config['server'], apikey=config['apikey'], workspace=config['workspace'], project=config['project'])

app\_manager = ListManager(rally, preview=config['preview'],values\_has\_key=config['values\_have\_key'])

#Email Notification

#threshold = 1550

message = "\nWorkspace: {} \nProject: {} \nAttribute: {}\nElement: {} \n\nBelow list synced successfully!!!\n".format(config['workspace'],config['project'],config['attribute\_name'], config['element\_name'])

message1= "\nWorkspace: {} \nProject: {} \nAttribute: {}\nElement: {} \n\nRecord count is less then {} in Desired Flat File. Please verify the file and rerun the script!!!\n".format(config['workspace'],config['project'],config['attribute\_name'], config['element\_name'],config['threshold'])

#message1="Record count is less then in Desired Flat File. Please check the file and run the app"

if count > config['threshold']:

logger.info(message)

app\_manager.load(config['element\_name'], config['attribute\_name'], desired\_value\_filename)

logger.info("Done")

send\_email\_notification(config['output\_result\_path'])

else:

logger.info(message1)

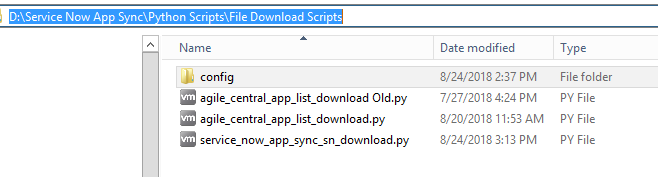
#logger.info("Record count is less then {}. Check the desired file!".format(config['threshold']))

#send\_email\_notification(message1)

send\_email\_notification(config['output\_result\_path'])

# File Download Scripts

D:\Service Now App Sync\Python Scripts\File Download Scripts



# Agile\_central\_app\_list\_download.py

import sys

import os

import json

from array import \*

from pyral import Rally, rallyWorkset, rallySettings

class AppListDownload:

def \_\_init\_\_(self, rally):

self.rally = rally

return

def app\_list(self,element\_name,attribute\_name,file\_name):

file\_path = r'D:\Service Now App Sync\Download Files\\' + file\_name +'.csv'

response = self.rally.getAllowedValues(element\_name,attribute\_name)

with open(file\_path, 'w', newline='', encoding='utf-8') as fp:

for item in response:

app\_name = item + '\n'

fp.write(app\_name)

fp.close

print('Downloaded ' + file\_name+ '!')

if \_\_name\_\_ == '\_\_main\_\_':

config\_filename = sys.argv[1]

with open(config\_filename, "r") as config\_file:

config = json.load(config\_file)

os.environ["HTTPS\_PROXY"] = config['proxy']

options = [opt for opt in sys.argv[1:] if opt.startswith('--')]

server, apikey, workspace,project, element\_name,attribute\_name = rallyWorkset(options)

rally = Rally(config['server'], apikey=config['apikey'], workspace=config['workspace'], project=config['project'])

app\_list\_download = AppListDownload(rally)

app\_list\_download.app\_list(config['element\_name'], config['attribute\_name'], config['file\_name'])

# "element\_name": "hierarchicalrequirement",

# Service\_now\_app\_sync\_sn\_download.py

import os

import datetime

from dateutil.relativedelta import relativedelta

import requests

from requests.auth import HTTPBasicAuth

#from requests.auth import HTTPProxyAuth

import json

import csv

#abs\_path = os.path.abspath('..')

abs\_path = r'D:\Service Now App Sync'

def custom\_date():

today = datetime.date.today()

one\_month\_ago =today - relativedelta(months=2)

month = one\_month\_ago.strftime("%b")

year = one\_month\_ago.year

custom\_date = month + ':FY'+ str(year) #Jul:FY2017

return custom\_date

def download\_file\_excel(url,username,password,file\_name):

print('Downloading ' + file\_name+ '...')

#file\_path = 'C:/Users/AWH14/Desktop/ClarityDataAutomation/Download Files/'+ file\_name +'.xlsx'

file\_path = abs\_path +'\Download Files\\' + file\_name +'.xlsx'

#print(file\_path)

try:

# download excel file

file\_request = requests.get(url, auth=HTTPBasicAuth(username, password))

output = open(file\_path, 'wb')

output.write(file\_request.content)

output.close()

except ValueError as err:

print(err)

def download\_file\_json(url,user,pw,file\_name):

print('Downloading ' + file\_name+ '...')

file\_path = abs\_path +'\Download Files\\' + file\_name +'.csv'

#print(file\_path)

try:

# download excel file

resp = requests.get(url, auth=HTTPBasicAuth(user, pw))

dataset = json.loads(resp.text)

data = dataset['result']

# Open a file for writing

output = open(file\_path, 'w', newline='', encoding='utf-8')

# Create the csv writer object

csvwriter = csv.writer(output)

count = 0

for appinfo in data:

if count == 0:

header = appinfo.keys()

csvwriter.writerow(header)

count +=1

csvwriter.writerow(appinfo.values())

output.close()

except ValueError as err:

print(err)

def main():

#os.environ["HTTPS\_PROXY"]="10.8.109.16"

#username ='datalink@unum.com'

password='UnumApptio202!'

user ='apptio'

pw = 'apptio99'

date = custom\_date()

url\_serverNow ='https://unumprod.service-now.com/api/unugr/apptio/unumapps'

download\_file\_json(url\_serverNow,user,pw,'api\_service\_now\_applications')

print('Download completed for the month of '+ date)

main()

# config.json

D:\Service Now App Sync\Python Scripts\File Download Scripts\config

{

"server": "rally1.rallydev.com",

"workspace": "Unum",

"project": "Unum Group",

"element\_name": "PortfolioItem",

"attribute\_name": "AppID",

"file\_name":"Agile Central Dropdown list",

"proxy":"10.215.109.16",

"apikey":"\_zdCoOTvjQKGUcFoE9KkYXvnxrE0iTk3UqQftTGnst9A"

}