# -\*- coding: utf-8 -\*-

""" Plexus (c) 2015 enen92

This file contains web utilities

Classes:

download\_tools() -> Contains a downloader, a extraction function and a remove function

Functions:

get\_page\_source -> Get a webpage source code through urllib2

mechanize\_browser(url) -> Get a webpage source code through mechanize module. To avoid DDOS protections.

makeRequest(url, headers=None) -> check if a page is up and retrieve its source code

clean(text) -> Remove specific characters from the page source

url\_isup(url, headers=None) -> Check if url is up. Returns True or False.

"""

import xbmc,xbmcplugin,xbmcgui,xbmcaddon,urllib,urllib2,tarfile,os,sys,re,gzip

from StringIO import StringIO

user\_agent = 'Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/28.0.1468.0 Safari/537.36'

class download\_tools():

def Downloader(self,url,dest,description,heading):

dp = xbmcgui.DialogProgress()

dp.create(heading,description,'')

dp.update(0)

urllib.urlretrieve(url,dest,lambda nb, bs, fs, url=url: self.\_pbhook(nb,bs,fs,dp))

def \_pbhook(self,numblocks, blocksize, filesize,dp=None):

try:

percent = int((int(numblocks)\*int(blocksize)\*100)/int(filesize))

dp.update(percent)

except:

percent = 100

dp.update(percent)

if dp.iscanceled():

dp.close()

def extract(self,file\_tar,destination):

dp = xbmcgui.DialogProgress()

dp.create(translate(30000),translate(30023))

tar = tarfile.open(file\_tar)

tar.extractall(destination)

dp.update(100)

tar.close()

dp.close()

def remove(self,file\_):

dp = xbmcgui.DialogProgress()

dp.create(translate(30000),translate(30024))

os.remove(file\_)

dp.update(100)

dp.close()

def get\_page\_source(url):

req = urllib2.Request(url)

req.add\_header('User-Agent', user\_agent)

response = urllib2.urlopen(req)

if response.info().get('Content-Encoding') == 'gzip':

buf = StringIO(response.read())

f = gzip.GzipFile(fileobj=buf)

link = f.read()

else:

link = response.read()

response.close()

return link

def makeRequest(url, headers=None):

try:

if not headers:

headers = {'User-agent' : 'Mozilla/5.0 (Windows NT 6.1; WOW64; rv:19.0) Gecko/20100101 Firefox/19.0'}

req = urllib2.Request(url,None,headers)

response = urllib2.urlopen(req)

data = response.read()

response.close()

return data

except:

sys.exit(0)

def url\_isup(url, headers=None):

try:

if not headers:

headers = {'User-agent' : 'Mozilla/5.0 (Windows NT 6.1; WOW64; rv:19.0) Gecko/20100101 Firefox/19.0'}

req = urllib2.Request(url,None,headers)

response = urllib2.urlopen(req)

data = response.read()

response.close()

return True

except: return False

def clean(text):

command={'\r':'','\n':'','\t':'','&nbsp;':' ','&quot;':'"','&#039;':'','&#39;':"'",'&#227;':'ã','&170;':'ª','&#233;':'é','&#231;':'ç','&#243;':'ó','&#226;':'â','&ntilde;':'ñ','&#225;':'á','&#237;':'í','&#245;':'õ','&#201;':'É','&#250;':'ú','&amp;':'&','&#193;':'Á','&#195;':'Ã','&#202;':'Ê','&#199;':'Ç','&#211;':'Ó','&#213;':'Õ','&#212;':'Ó','&#218;':'Ú'}

regex = re.compile("|".join(map(re.escape, command.keys())))

return regex.sub(lambda mo: command[mo.group(0)], text)