################################################################################

# Copyright (C) 2015 Surfacingx #

# #

# This Program is free software; you can redistribute it and/or modify #

# it under the terms of the GNU General Public License as published by #

# the Free Software Foundation; either version 2, or (at your option) #

# any later version. #

# #

# This Program is distributed in the hope that it will be useful, #

# but WITHOUT ANY WARRANTY; without even the implied warranty of #

# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the #

# GNU General Public License for more details. #

# #

# You should have received a copy of the GNU General Public License #

# along with XBMC; see the file COPYING. If not, write to #

# the Free Software Foundation, 675 Mass Ave, Cambridge, MA 02139, USA. #

# http://www.gnu.org/copyleft/gpl.html #

################################################################################

import xbmc, xbmcaddon, xbmcgui, xbmcplugin, os, sys, xbmcvfs, glob

import shutil

import urllib2,urllib

import re

import uservar

import time

try: from sqlite3 import dbapi2 as database

except: from pysqlite2 import dbapi2 as database

from datetime import date, datetime, timedelta

from resources.libs import wizard as wiz

ADDON\_ID = uservar.ADDON\_ID

ADDONTITLE = uservar.ADDONTITLE

ADDON = wiz.addonId(ADDON\_ID)

DIALOG = xbmcgui.Dialog()

HOME = xbmc.translatePath('special://home/')

ADDONS = os.path.join(HOME, 'addons')

USERDATA = os.path.join(HOME, 'userdata')

PLUGIN = os.path.join(ADDONS, ADDON\_ID)

PACKAGES = os.path.join(ADDONS, 'packages')

ADDONDATA = os.path.join(USERDATA, 'addon\_data', ADDON\_ID)

ADDOND = os.path.join(USERDATA, 'addon\_data')

LOGINFOLD = os.path.join(ADDONDATA, 'Login')

ICON = os.path.join(PLUGIN, 'icon.png')

TODAY = date.today()

TOMORROW = TODAY + timedelta(days=1)

THREEDAYS = TODAY + timedelta(days=3)

KEEPLOGIN = wiz.getS('keeplogin')

LOGINSAVE = wiz.getS('loginlastsave')

COLOR1 = uservar.COLOR1

COLOR2 = uservar.COLOR2

ORDER = ['sportsaccess', 'smoothstreams', 'communityportal', 'tvportal', 'flawless', 'opensubtitles', 'sportsmania', 'sportsnationhdtv', 'ultimatemania', 'otttv', 'ivue', 'ontapptv', 'vpnicity', 'overlordtv', 'xtreamcodes']

LOGINID = {

'sportsaccess': {

'name' : 'Sports Access',

'plugin' : 'plugin.video.sportsaccess',

'saved' : 'loginsportsaccess',

'path' : os.path.join(ADDONS, 'plugin.video.sportsaccess'),

'icon' : os.path.join(ADDONS, 'plugin.video.sportsaccess', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'plugin.video.sportsaccess', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'sportsaccess\_login'),

'settings' : os.path.join(ADDOND, 'plugin.video.sportsaccess', 'settings.xml'),

'default' : 'skyusername',

'data' : ['skyusername', 'skypassword'],

'activate' : 'RunPlugin(plugin://plugin.video.sportsaccess/?mode=259)'},

'smoothstreams': {

'name' : 'Smooth Streams',

'plugin' : 'script.smoothstreams',

'saved' : 'loginsmoothstreams',

'path' : os.path.join(ADDONS, 'script.smoothstreams'),

'icon' : os.path.join(ADDONS, 'script.smoothstreams', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'script.smoothstreams', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'smoothstreams\_login'),

'settings' : os.path.join(ADDOND, 'script.smoothstreams', 'settings.xml'),

'default' : 'username',

'data' : ['service', 'username', 'user\_password'],

'activate' : ''},

'opensubtitles': {

'name' : 'OpenSubtitles',

'plugin' : 'service.subtitles.opensubtitles',

'saved' : 'loginopensub',

'path' : os.path.join(ADDONS, 'service.subtitles.opensubtitles'),

'icon' : os.path.join(ADDONS, 'service.subtitles.opensubtitles', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'service.subtitles.opensubtitles', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'opensub\_login'),

'settings' : os.path.join(ADDOND, 'service.subtitles.opensubtitles', 'settings.xml'),

'default' : 'OSuser',

'data' : ['OSuser', 'OSpass'],

'activate' : ''},

'communityportal': {

'name' : 'Community Portal',

'plugin' : 'plugin.program.totalinstaller',

'saved' : 'logintotalinstaller',

'path' : os.path.join(ADDONS, 'plugin.program.totalinstaller'),

'icon' : os.path.join(ADDONS, 'plugin.program.totalinstaller', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'plugin.program.totalinstaller', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'totalinstaller\_login'),

'settings' : os.path.join(ADDOND, 'plugin.program.totalinstaller', 'settings.xml'),

'default' : 'username',

'data' : ['login', 'username', 'password'],

'activate' : ''},

'tvportal': {

'name' : 'TV Portal',

'plugin' : 'script.tvportal',

'saved' : 'logintvportal',

'path' : os.path.join(ADDONS, 'script.tvportal'),

'icon' : os.path.join(ADDONS, 'script.tvportal', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'script.tvportal', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'tvportal\_login'),

'settings' : os.path.join(ADDOND, 'script.tvportal', 'settings.xml'),

'default' : 'username',

'data' : ['login', 'username', 'password'],

'activate' : ''},

'flawless': {

'name' : 'Flawless IPTV',

'plugin' : 'plugin.video.FlawlessTv',

'saved' : 'loginflawless',

'path' : os.path.join(ADDONS, 'plugin.video.FlawlessTv'),

'icon' : os.path.join(ADDONS, 'plugin.video.FlawlessTv', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'plugin.video.FlawlessTv', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'flawless\_login'),

'settings' : os.path.join(ADDOND, 'plugin.video.FlawlessTv', 'settings.xml'),

'default' : 'Username',

'data' : ['Username', 'Password'],

'activate' : ''},

'sportsmania': {

'name' : 'Sports Mania',

'plugin' : 'plugin.video.sportsmania',

'saved' : 'loginsportsmania',

'path' : os.path.join(ADDONS, 'plugin.video.sportsmania'),

'icon' : os.path.join(ADDONS, 'plugin.video.sportsmania', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'plugin.video.sportsmania', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'sportsmania\_login'),

'settings' : os.path.join(ADDOND, 'plugin.video.sportsmania', 'settings.xml'),

'default' : 'snusername',

'data' : ['snusername', 'snpassword'],

'activate' : 'RunPlugin(plugin://plugin.video.sportsmania/?mode=202)'},

'sportsnationhdtv': {

'name' : 'Sports NationHD',

'plugin' : 'plugin.video.sportsnationhdtv',

'saved' : 'loginsportsnationhd',

'path' : os.path.join(ADDONS, 'plugin.video.sportsnationhdtv'),

'icon' : os.path.join(ADDONS, 'plugin.video.sportsnationhdtv', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'plugin.video.sportsnationhdtv', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'sportsnationhd\_login'),

'settings' : os.path.join(ADDOND, 'plugin.video.sportsnationhdtv', 'settings.xml'),

'default' : 'snusername',

'data' : ['snusername', 'snpassword'],

'activate' : 'RunPlugin(plugin://plugin.video.sportsnationhdtv/?mode=202)'},

'ultimatemania': {

'name' : 'Ultimate Mania',

'plugin' : 'plugin.video.ultimatemania',

'saved' : 'loginultimatemania',

'path' : os.path.join(ADDONS, 'plugin.video.ultimatemania'),

'icon' : os.path.join(ADDONS, 'plugin.video.ultimatemania', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'plugin.video.ultimatemania', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'sportsmania\_login'),

'settings' : os.path.join(ADDOND, 'plugin.video.ultimatemania', 'settings.xml'),

'default' : 'snusername',

'data' : ['snusername', 'snpassword'],

'activate' : 'RunPlugin(plugin://plugin.video.ultimatemania/?mode=202)'},

'otttv': {

'name' : 'Over The Top TV',

'plugin' : 'plugin.video.ottalpha',

'saved' : 'loginotttv',

'path' : os.path.join(ADDONS, 'plugin.video.ottalpha'),

'icon' : os.path.join(ADDONS, 'plugin.video.ottalpha', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'plugin.video.ottalpha', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'otttv\_login'),

'settings' : os.path.join(ADDOND, 'plugin.video.ottalpha', 'settings.xml'),

'default' : 'Username',

'data' : ['Username', 'Password'],

'activate' : ''},

'ivue': {

'name' : 'Ivue TV Guide',

'plugin' : 'script.ivueguide',

'saved' : 'loginivue',

'path' : os.path.join(ADDONS, 'script.ivueguide'),

'icon' : os.path.join(ADDONS, 'script.ivueguide', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'script.ivueguide', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'ivue\_login'),

'settings' : os.path.join(ADDOND, 'script.ivueguide', 'settings.xml'),

'default' : 'username',

'data' : ['username', 'password'],

'activate' : ''},

'ontapptv': {

'name' : 'OnTapp.TV Guide',

'plugin' : 'script.tvguidedixie',

'saved' : 'loginontapptv',

'path' : os.path.join(ADDONS, 'script.tvguidedixie'),

'icon' : os.path.join(ADDONS, 'script.tvguidedixie', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'script.tvguidedixie', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'ontapptv\_login'),

'settings' : os.path.join(ADDOND, 'script.tvguidedixie', 'settings.xml'),

'default' : 'username',

'data' : ['username', 'password'],

'activate' : ''},

'vpnicity': {

'name' : 'VPNicity',

'plugin' : 'plugin.program.vpnicity',

'saved' : 'loginvpnicity',

'path' : os.path.join(ADDONS, 'plugin.program.vpnicity'),

'icon' : os.path.join(ADDONS, 'plugin.program.vpnicity', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'plugin.program.vpnicity', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'vpnicity\_login'),

'settings' : os.path.join(ADDOND, 'plugin.program.vpnicity', 'settings.xml'),

'default' : 'USER',

'data' : ['USER', 'PASS'],

'activate' : ''},

'overlordtv': {

'name' : 'Overlord TV',

'plugin' : 'plugin.video.overlordtv',

'saved' : 'loginoverlord',

'path' : os.path.join(ADDONS, 'plugin.video.overlordtv'),

'icon' : os.path.join(ADDONS, 'plugin.video.overlordtv', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'plugin.video.overlordtv', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'overlordtv\_login'),

'settings' : os.path.join(ADDOND, 'plugin.video.overlordtv', 'settings.xml'),

'default' : 'Username',

'data' : ['Username', 'Password'],

'activate' : ''},

'xtreamcodes': {

'name' : 'Xtream Codes',

'plugin' : 'plugin.video.xtream-codes',

'saved' : 'loginxtreamcodes',

'path' : os.path.join(ADDONS, 'plugin.video.xtream-codes'),

'icon' : os.path.join(ADDONS, 'plugin.video.xtream-codes', 'icon.png'),

'fanart' : os.path.join(ADDONS, 'plugin.video.xtream-codes', 'fanart.jpg'),

'file' : os.path.join(LOGINFOLD, 'xtreamcodes\_login'),

'settings' : os.path.join(ADDOND, 'plugin.video.xtream-codes', 'settings.xml'),

'default' : 'kasutajanimi',

'data' : ['kasutajanimi', 'salasona'],

'activate' : ''}

}

def loginUser(who):

user=None

if LOGINID[who]:

if os.path.exists(LOGINID[who]['path']):

try:

add = wiz.addonId(LOGINID[who]['plugin'])

user = add.getSetting(LOGINID[who]['default'])

except:

pass

return user

def loginIt(do, who):

if not os.path.exists(ADDONDATA): os.makedirs(ADDONDATA)

if not os.path.exists(LOGINFOLD): os.makedirs(LOGINFOLD)

if who == 'all':

for log in ORDER:

if os.path.exists(LOGINID[log]['path']):

try:

addonid = wiz.addonId(LOGINID[log]['plugin'])

default = LOGINID[log]['default']

user = addonid.getSetting(default)

if user == '' and do == 'update': continue

updateLogin(do, log)

except: pass

else: wiz.log('[Login Data] %s(%s) is not installed' % (LOGINID[log]['name'],LOGINID[log]['plugin']), xbmc.LOGERROR)

wiz.setS('loginlastsave', str(THREEDAYS))

else:

if LOGINID[who]:

if os.path.exists(LOGINID[who]['path']):

updateLogin(do, who)

else: wiz.log('[Login Data] Invalid Entry: %s' % who, xbmc.LOGERROR)

def clearSaved(who, over=False):

if who == 'all':

for login in LOGINID:

clearSaved(login, True)

elif LOGINID[who]:

file = LOGINID[who]['file']

if os.path.exists(file):

os.remove(file)

wiz.LogNotify('[COLOR %s]%s[/COLOR]' % (COLOR1, LOGINID[who]['name']), '[COLOR %s]Login Data: Removed![/COLOR]' % COLOR2, 2000, LOGINID[who]['icon'])

wiz.setS(LOGINID[who]['saved'], '')

if over == False: wiz.refresh()

def updateLogin(do, who):

file = LOGINID[who]['file']

settings = LOGINID[who]['settings']

data = LOGINID[who]['data']

addonid = wiz.addonId(LOGINID[who]['plugin'])

saved = LOGINID[who]['saved']

default = LOGINID[who]['default']

user = addonid.getSetting(default)

suser = wiz.getS(saved)

name = LOGINID[who]['name']

icon = LOGINID[who]['icon']

if do == 'update':

if not user == '':

try:

with open(file, 'w') as f:

for login in data:

f.write('<login>\n\t<id>%s</id>\n\t<value>%s</value>\n</login>\n' % (login, addonid.getSetting(login)))

f.close()

user = addonid.getSetting(default)

wiz.setS(saved, user)

wiz.LogNotify("[COLOR %s]%s[/COLOR]" % (COLOR1, name),'[COLOR %s]Login Data: Saved![/COLOR]' % COLOR2, 2000, icon)

except Exception, e:

wiz.log("[Login Data] Unable to Update %s (%s)" % (who, str(e)), xbmc.LOGERROR)

else: wiz.LogNotify("[COLOR %s]%s[/COLOR]" % (COLOR1, name),'[COLOR %s]Login Data: Not Registered![/COLOR]' % COLOR2, 2000, icon)

elif do == 'restore':

if os.path.exists(file):

f = open(file,mode='r'); g = f.read().replace('\n','').replace('\r','').replace('\t',''); f.close();

match = re.compile('<login><id>(.+?)</id><value>(.+?)</value></login>').findall(g)

try:

if len(match) > 0:

for login, value in match:

addonid.setSetting(login, value)

user = addonid.getSetting(default)

wiz.setS(saved, user)

wiz.LogNotify("[COLOR %s]%s[/COLOR]" % (COLOR1, name), '[COLOR %s]Login: Restored![/COLOR]' % COLOR2, 2000, icon)

except Exception, e:

wiz.log("[Login Data] Unable to Restore %s (%s)" % (who, str(e)), xbmc.LOGERROR)

#else: wiz.LogNotify(name,'login Data: [COLOR red]Not Found![/COLOR]', 2000, icon)

elif do == 'clearaddon':

wiz.log('%s SETTINGS: %s' % (name, settings), xbmc.LOGDEBUG)

if os.path.exists(settings):

try:

f = open(settings, "r"); lines = f.readlines(); f.close()

f = open(settings, "w")

for line in lines:

match = wiz.parseDOM(line, 'setting', ret='id')

if len(match) == 0: f.write(line)

else:

if match[0] not in data: f.write(line)

else: wiz.log('Removing Line: %s' % line, xbmc.LOGNOTICE)

f.close()

wiz.LogNotify("[COLOR %s]%s[/COLOR]" % (COLOR1, name),'[COLOR %s]Addon Data: Cleared![/COLOR]' % COLOR2, 2000, icon)

except Exception, e:

wiz.log("[Trakt Data] Unable to Clear Addon %s (%s)" % (who, str(e)), xbmc.LOGERROR)

wiz.refresh()

def autoUpdate(who):

if who == 'all':

for log in LOGINID:

if os.path.exists(LOGINID[log]['path']):

autoUpdate(log)

elif LOGINID[who]:

if os.path.exists(LOGINID[who]['path']):

u = loginUser(who)

su = wiz.getS(LOGINID[who]['saved'])

n = LOGINID[who]['name']

if u == None or u == '': return

elif su == '': loginIt('update', who)

elif not u == su:

if DIALOG.yesno(ADDONTITLE, "[COLOR %s]Would you like to save the [COLOR %s]Login[/COLOR] data for [COLOR %s]%s[/COLOR]?" % (COLOR2, COLOR1, COLOR1, n), "Addon: [COLOR green][B]%s[/B][/COLOR]" % u, "Saved:[/COLOR] [COLOR red][B]%s[/B][/COLOR]" % su if not su == '' else 'Saved:[/COLOR] [COLOR red][B]None[/B][/COLOR]', yeslabel="[B][COLOR green]Save Data[/COLOR][/B]", nolabel="[B][COLOR red]No Cancel[/COLOR][/B]"):

loginIt('update', who)

else: loginIt('update', who)

def importlist(who):

if who == 'all':

for log in LOGINID:

if os.path.exists(LOGINID[log]['file']):

importlist(log)

elif LOGINID[who]:

if os.path.exists(LOGINID[who]['file']):

d = LOGINID[who]['default']

sa = LOGINID[who]['saved']

su = wiz.getS(sa)

n = LOGINID[who]['name']

f = open(LOGINID[who]['file'],mode='r'); g = f.read().replace('\n','').replace('\r','').replace('\t',''); f.close();

m = re.compile('<login><id>%s</id><value>(.+?)</value></login>' % d).findall(g)

if len(m) > 0:

if not m[0] == su:

if DIALOG.yesno(ADDONTITLE, "[COLOR %s]Would you like to import the [COLOR %s]Login[/COLOR] data for [COLOR %s]%s[/COLOR]?" % (COLOR2, COLOR1, COLOR1, n), "File: [COLOR green][B]%s[/B][/COLOR]" % m[0], "Saved:[/COLOR] [COLOR red][B]%s[/B][/COLOR]" % su if not su == '' else 'Saved:[/COLOR] [COLOR red][B]None[/B][/COLOR]', yeslabel="[B][COLOR green]Save Data[/COLOR][/B]", nolabel="[B][COLOR red]No Cancel[/COLOR][/B]"):

wiz.setS(sa, m[0])

wiz.log('[Import Data] %s: %s' % (who, str(m)), xbmc.LOGNOTICE)

else: wiz.log('[Import Data] Declined Import(%s): %s' % (who, str(m)), xbmc.LOGNOTICE)

else: wiz.log('[Import Data] Duplicate Entry(%s): %s' % (who, str(m)), xbmc.LOGNOTICE)

else: wiz.log('[Import Data] No Match(%s): %s' % (who, str(m)), xbmc.LOGNOTICE)

def activateLogin(who):

if LOGINID[who]:

if os.path.exists(LOGINID[who]['path']):

act = LOGINID[who]['activate']

addonid = wiz.addonId(LOGINID[who]['plugin'])

if act == '': addonid.openSettings()

else: url = xbmc.executebuiltin(LOGINID[who]['activate'])

else: DIALOG.ok(ADDONTITLE, '%s is not currently installed.' % LOGINID[who]['name'])

else:

wiz.refresh()

return

check = 0

while loginUser(who) == None or loginUser(who) == "":

if check == 30: break

check += 1

time.sleep(10)

wiz.refresh()