Transmission

Firefox addon to add magnet/torrent links to transmission web server:
https://addons.mozilla.org/en-GB/firefox/addon/transmitter-for-transmission/?src=search
https://addons.mozilla.org/en-GB/firefox/addon/magnet-link-to-transmission/

https://forum.transmissionbt.com/viewtopic.php?p=67122#p67122

Python script for adding magnet links:

```
#!/usr/bin/python3
#
# Simple magnet link uploader
#
# Needs httplib2: http://code.google.com/p/httplib2/
#

import httplib2
from sys import argv, exit
from json import dumps

# is magnet link given in command line?
if len(argv) < 2:
    print('Usage: magnet-handler.py "magnet link"')
    exit(1)

# Replace hostname, transmission-username and transmission-password
url = 'http://hostname:9091/transmission/rpc'
username = 'transmission-username'
password = 'transmission-password'

h = httplib2.Http(".cache")
h.add_credentials(username, password)
resp, content = h.request(url, "GET")

headers = { "X-Transmission-Session-Id": resp['x-transmission-session-id'] }
body = dumps( { "method": "torrent-add",
                "arguments": { "filename": argv[1] } } )

response, content = h.request(url, 'POST', headers=headers, body=body)

if str(content).find("success") == -1:
    print("Magnet Link: " + argv[1])
    print("Answer: " + content)
    print("No 'success' here!")
    print("Press Enter to Exit.")
    input()
```

Python script for adding magnet links with notification (windows):

```
#!/usr/bin/python3
#
# Transmission magnet link uploader
# (https cert check disabled)
#
# Uses requests & PyQt5
# http://docs.python-requests.org/en/latest/
# http://www.riverbankcomputing.co.uk/software/pyqt/download5
#

# Standard
from json import dumps
from sys import argv, exit
from urllib.parse import unquote

# 3rd Party
import requests
from PyQt5.QtCore import QTimer
from PyQt5.QtGui import QIcon
from PyQt5.QtWidgets import (QApplication, QMessageBox, QSystemTrayIcon)

# Settings
url = 'https://address.domain:port/transmission/rpc'
username = 'transmission'
password = 'secret'
icon = 'C:\\Icons\\Magnet-icon.png'

# Functions

# Show Message Box
def show_message(title, message):
    app = QApplication([])
    QMessageBox.information(None, title, message)

# Show Tray Message
def show_tray_message(title, message):
    app = QApplication([])
    trayIcon = QSystemTrayIcon(QIcon(icon), app)
    trayIcon.show()
    trayIcon.showMessage(title, message, QSystemTrayIcon.Information)
    QTimer.singleShot(8000, app.quit)
    app.exec_()

# Get RPC Session ID
def get_session_id():
    sessionid_request = requests.get(url, auth=(username, password),
verify=False)
    return sessionid_request.headers['x-transmission-session-id']

# Post Magnet Link
def post_link(magnetlink):
```

```
    sessionid = get_session_id()
    if sessionid:
        headers = {"X-Transmission-Session-Id": sessionid}
        body = dumps({"method": "torrent-add", "arguments": {"filename":
magnetlink}})
        post_request = requests.post(url, data=body, headers=headers,
auth=(username, password), verify=False)
        if str(post_request.text).find("success") == -1:
            title =  argv[0] + ' - Error'
            message = 'Magnet Link: ' + magnetlink + '\nAnswer: ' +
post_request.text
            show_message(title, message)
        else:
            message = '\n'.join(names) + '\n' + '\n'.join(trackers)
            show_tray_message('Magnet Link Added', message)

# End of Functions

# Main prog
if __name__ == '__main__':

    # Check Argument and Extract Name/Trackers or Display Alert Message
    if len(argv) < 2:
        show_message(argv[0] + ' - Alert', 'Usage: ' + argv[0] + ' [magnet
link]')
        exit(1)
    elif argv[1].startswith('magnet'):
        names = []
        trackers = []
        for item in argv[1].split('&'):
            decoded = unquote(item)
            if decoded.startswith('dn='):
                names.append(decoded.replace('dn=', ''))
            if decoded.startswith('tr='):
                trackers.append(decoded.replace('tr=', ''))
    else:
        show_message(argv[0] + ' - Alert', argv[1] + ' not magnet link!')
        exit(1)

    post_link(argv[1])
```

Re-add failed torrents

Store all current torrents in csv file, then prompt for confirmation for each failed torrent to remove it with data and re-submit it to transmission.

```
import transmissionrpc
import csv
```

```
# Connect to Transmission server with authentication
tc = transmissionrpc.Client('localhost', port=9091, user='USERNAME',
password='PASSWORD')

# Retrieve list of torrents with error messages
torrents = tc.get_torrents(arguments=['id', 'name', 'status', 'progress',
'magnetLink', 'errorString'])
torrents_with_errors = [t for t in torrents if t.errorString != '']

# Iterate over each torrent with errors
for torrent in torrents_with_errors:
    # Output the torrent information
    print(f"Name: {torrent.name}\nError: {torrent.errorString}")

    # Ask for confirmation to delete and re-add the torrent
    confirmation = input("Delete and re-add this torrent? (y/n):
").strip().lower()
    if confirmation == 'y':
        # Delete the torrent and re-add it with the magnet link
        tc.remove_torrent(torrent.id, delete_data=True)
        tc.add_torrent(torrent.magnetLink)
        print(f"{torrent.name} has been deleted and re-added.")
    else:
        print(f"{torrent.name} will not be deleted and re-added.")

# Output the torrent information to a CSV file
with open('torrent_info.csv', 'w', newline='') as csvfile:
    fieldnames = ['id', 'name', 'magnet', 'error']
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
    writer.writeheader()
    for torrent in torrents_with_errors:
        writer.writerow({'id: torrent.id, 'name': torrent.name, 'magnet':
torrent.magnetLink, 'error': torrent.errorString})
```

From:
http://wuff.dyndns.org/ - **Wulf's Various Things**

Permanent link:
http://wuff.dyndns.org/doku.php?id=config:transmission&rev=1680558253

Last update: **2023/05/29 11:53**