Box Upload Documentation

A tool created by Brenden Sweetman

**Introduction:**

This script was initially developed in conjunction with the Teaching Center Video Processing project to automaticity upload videos after they had been processed. The ultimate goal was to automate practically the entire process from receiving videos to uploading the final videos. I developed the box\_uplaod.py script, however, to stand alone so it could be used for any further projects. The script uses the python box SDK and oAuth2 authentication to upload a file as a given user to a specified folder in box. The project is made up of 2 scripts: box\_upload.py to upload the file and auth.py to make the initial configuration file.

**Box\_upload.py**

The box\_uplaod.py script performs a couple different action before uploading the file to box. First, the script parses supplied arguments (see usage). Second, the script will authorize with the box SDK if a new access token is needed the script will reauthorize the user. The script will then use the box SDK to upload the file to the provided folder location.

**Methods in box\_upload.py:**

**main:**

The main method reads in and parses the command line arguments (see usage). Then, reads in the configuration file and checks expiration time on the current access token. If the token has expired the refreshAuth method will be called. Otherwise the access token will be used to grant access to the requested file. Once authorized the script will upload the file to the provided location in box.

**upload:**

The upload method attempts to upload the provided file to the provided location in box using the box SDK. The method will pass an error to the user if the upload fails.

**refreshAuth:**

The refreshAuth method uses the refresh token, client ID and client secret to generate a new universal access token. For more information on the refresh protocol see the box documentation at: <https://developer.box.com/reference#oauth-2-overview>. The script will then immediately down-scope the universal access token to one that only has access to the requested folder. This prevents the script from accessing or modifying unwanted folders in box. For more information about down-scoped access see the box documentation here: <https://developer.box.com/reference#token-exchange>.

**writeCfg:**

The writecfg method overwrites the provided configuration file with the client ID, client secret, refresh token, access token and expiration time of the current access token.

**readCfg:**

The readCfg method reads the provided configuration file.

**System requirements:**

The script runs on python3 using the box python SDK. Thus, the script must be run under a python3 environment with the boxsdk installed. The boxsdk can be installed using pip and found here: <https://github.com/box/box-python-sdk>. The script also handles all http communications using the requests package: <http://docs.python-requests.org/en/master/>. Other python packages used (time, re, sys, ntpath, argsparse) are usually part of core python installations. The script can run under any python environment matching the necessary requirements.

**Usage:**

The script is designed as a command line tool called with 4 possible arguments.

* The –f (--file) argument provides the path to the file to be uploaded and is required.
* The –c (--config) argument provides the path to the configuration file to be used and is required.
* The –i (--ID) argument provides the box folder ID for the destination folder in box. If this argument is not provided the file will be uploaded to the authorized user’s base directory (file ID=0).
* The –v (--verbose) argument activates the script’s verbose mode that can be used for additional output in the case of advanced trouble shooting.

An example of the command line call is given below:

$ ./box\_upload.py –f <file path> -c <config file path>

**Auth.py**

The auth.py is a script to create a new configuration file for the box\_upload.py script. The script is designed to be simple and straight forward for any user to follow. I will leave the explanation here brief, please refer to the “create a new configuration file” document for specific instructions on use. The script can be used to reauthorize a user if the box\_upload.py stops working at any time. It can also be used to change the authorized user.

NOTE: auth.py generates a universal access token. The token in the configuration file will remain universal until the token expires and box\_upload.py reauthorizes and down-scopes the token.

**Usage:**

The script requires two command line arguments:

* The –i (--client\_id) argument provides the client ID for the new user.
* The –f (--file\_name) argument provides the name of the generated configuration file

In addition to the command line arguments the user is asked to provide the client secret. Then paste a URL into a browser and copy a code generated by the box API into the command line. The script will then generate the configuration file.

**System Requirements:**

The auth.py script will run happily in the same python environment as box\_upload.py