DuraCloud Sync Tool - Command Line

Introduction

The Sync Tool is a utility which was created in order to provide a simple way to move files from a local file system to DuraCloud and subsequently keep the files in DuraCloud synchronized with those on the local system.

Download

Download the Sync Tool from the Downloads page.

Getting Started

The Sync Tool can be installed using one of the installers on the downloads page linked above. Once installed, the Sync Tool will default to running in GUI mode. To run in command line mode, open a terminal window (or command prompt) and navigate to the Sync Tool installation directory. Once there, execute the Sync Tool JAR file using: "java -jar duracloudsync.jar --help". This will print the usage information for the tool.

How the Sync Tool Works

- When you run the Sync Tool for the first time, you must include DuraCloud connection information (host, port, username, password) as well as the space where you would like all of your files stored. You must also provide a list of directories which will be synced to DuraCloud and a directory for the Sync Tool to use for its own work.
- When the Sync Tool starts up, it will look through all of the files in each of the local content directories and add them to its internal queue for processing. Each of those files will then be written to your DuraCloud space. As this initial write is happening a listener is set up to watch for any file changes within each of the content directories. As a change occurs (a file is added, updated, or deleted), that change is added to the queue, and the appropriate action is taken to make the DuraCloud space consistent with the local file (i.e. the file is either written to the space or deleted from the space.)
- You can stop the Sync Tool at any time by typing 'x' or 'exit' on the command line where it is running. It will stop all listeners, complete any file transfers that are in progress, and close down.
- When you restart the Sync Tool, if you point it at the same work directory and use the same options, it will pick up where it left off. While the Sync Tool is running, it is constantly writing backups of its internal queue, so it first reads the most current backup and begins processing the files there. It then scans the content directories to see if there are any files which have been added or updated since the last backup, and it also pulls a list of files from the DuraCloud space and scans that list to see if any local files have been deleted. Any changes detected are added to the internal queue, and the Sync Tool continues to run as usual.

Operational notes

- Running
  - To ensure that the command line interface is selected, at least one command line option must be included when executing the Sync Tool on the command line. To see the help text, simply include a "--help" parameter. Running with no parameters will start the Sync Tool in GUI mode.
- Jump Start
  - The Jump Start option available in the SyncTool is designed to streamline the transfer of new file sets to DuraCloud. This is accomplished by removing the checks that the SyncTool traditionally performs before uploading a file. These checks normally try to determine if a file already exists in DuraCloud. With the Jump Start option enabled, the SyncTool assumes that all files are new and need to be moved to DuraCloud. This is option is ideal for the initial data transfer into DuraCloud, when all selected data needs to be transferred. The Jump Start option should be turned off when running the SyncTool over a data set that is already in DuraCloud (in order to discover and transfer any new files), so that unnecessary content transfers can be avoided.
- Restarting
  - You can perform a restart of the Sync Tool by using the -g command line option to point to the Sync Tool configuration file, which is written into the work directory (named synctool.config)
  - If you would like the Sync Tool to perform a clean start rather than a restart (i.e. you would like it to compare all files in the content directories to DuraCloud) you will need to either point it to a new work directory, or clear out the existing work directory.
  - The Sync Tool will perform a clean start (not a restart) if the list of content directories is not the same as the previous run. This is to ensure that all files in all content directories are processed properly.
- Getting a clean start
  - If you specifically do not want to restart from a previous run, and would like to ensure that the sync tool considers every file in all directories specified, you can use the -l (or --clean-start) command line option to indicate this desire.

        - A clean start will also occur by default whenever the host, destination space, destination store, or the list of content directories changes from one run of the tool to the next.
- Collisions
    - The Sync Tool allows you to sync multiple local directories into a single space within DuraCloud. Because of this, there is the possibility of file naming collisions, where two local files resolve to the same DuraCloud ID. If this happens, one file will be overwritten by the other. There are a few ways to ensure that this does not occur:
        - Ensure that the top level files and directories within the set of content directories do not have overlapping names.
        - Sync only a single directory to a space. You can run multiple copies of the Sync Tool, each over a single local directory, syncing to its own DuraCloud space.
- Work Directory default
    - As of DuraCloud version 2.3.0, the work directory parameter is not required. If not specified, the work directory will be named "duracloud-sync-work", and will be placed under the user's home directory
- Work Directory - these files and directories can be found in the work directory (specified using the -w command line parameter)
    - Config Files
        - When the Sync Tool starts up, it writes the list of parameters and values provided by the user on startup to a file called synctool.config in the work directory. This file can be used to restart the Sync Tool, using the -g parameter to point to the file's location. You can also restart the Sync Tool by indicating the same set of options as used originally. The -g parameter is for convenience only and is not required in any circumstance. Note that this file is overwritten each time the Sync Tool is run with a different set of parameters, so you may choose to copy the file elsewhere (or give it a new name) if you would like to keep a copy of a particular configuration set.
        - You may also see a file named synctool.config.bak in the work directory which is used to compare against the current config in order to determine if a restart is possible. In order for a restart to occur, the list of content directories (-c parameter) must be the same as the previous execution of the tool, and there must be at least one changed list backup (see below.)
    - Changed List Directory
        - While the Sync Tool is running it is constantly updating the list of files which have been changed (when starting the first time, this includes all files in the directories that need to be synced). In order to allow the Sync Tool to restart after it has been stopped, this list of files is continually backed up into the *changedList* directory. There is no reason to edit these files, but you may choose to delete the *changedList* directory along with the config files mentioned above to ensure that the Sync Tool does not attempt to perform a restart.
    - Logs Directory
        - There are three logs captured in this directory:
        - (1) history.log - captures a list of files which the SyncTool has transferred, providing a history of transfer events
        - (2) sync.log - captures log output from the SyncTool application. If something goes wrong, you will usually find information about the problem here.
        - (3) complete.log - captures all log output, including logs generated by the SyncTool application as well as all dependent libraries. This is useful for application debugging when the information in the sync-tool.log file is insufficient to understand a problem.
- Time Stamps
    - As of DuraCloud version 2.3.0, the Sync Tool will collect time stamp information for each transferred file from the file system and store this information as properties on the content item in DuraCloud
    - Note that the time stamps collected may vary somewhat based on the operating system and file system on which the content is stored
- Destination Prefix
    - Using the prefix option, the content IDs that are created for the files being moved to DuraCloud by the SyncTool can be made to begin with a consistent text value. There are several reasons this might be useful, such as to include the name of a top-level directory in the path, or to be able to run the Sync from a new sub-directory, but still maintain the full path included on all existing stored content. Suppos e the path to a local file (found within the watch directory) is "dir1/file.txt" and you would like the resulting content stored in DuraCloud to be 'a/b/c/dir1/file.txt. To achieve that result, the destination prefix of "a/b/c/" would need to be set.

    > Adding or changing a prefix for content that has already been transferred to DuraCloud will result in those files being duplicated in DuraCloud storage. Removing the duplicate files can be done by using the "sync deletes" option, but this will cause all content in the destination space which does not include the prefix to be deleted (along with any content that is not found in the local watch directories.) Be cautious when using this feature if you have already uploaded content to your DuraCloud space.

    > If you use a prefix to include a file path (such as a top level directory name), remember to include the "/" character at the end of your prefix. For example, using the prefix "dir1/" with file "file.txt", your final content ID will be "dir1/file.txt". If you were to forget the slash, your prefix would be "dir1", which would lead to a content ID of "dir1file.txt", which is likely not what you want.

- Optimizing Transfer Rate
    - You can potentially increase the transfer rate of your content by increasing the thread count. The thread count can be set using the -t option. To help you determine the optimal thread count in order to maximize throughput, we've added a new diagnostic tool. Please see DuraCloud SyncOptimize Tool for more information.

Large Datasets and Out of Memory Errors

When using the SyncTool to transmit data sets with a large number of files (i.e. hundreds of thousands of files or more) users occasionally run into out of memory errors. Users with sufficient memory resources on their machines can usually remedy this problem by increasing the maximum heap space available to the Java VM. We recommend starting with a setting of at least 1 GB when working with sets over 100,000 files. If the problem persists, try increasing the memory value until the problem ceases to manifest. To increase the heap space use the -Xmx java option. Click for more information on setting the heap space.

An alternative solution is to upload files in smaller sets. The prefix option can be used to ensure that files are added to DuraCloud with the preferred ID values.

To run the SyncTool in UI mode with 1 GB of heap memory space, download the Jar version of the SyncTool and execute the following on the command line:

```
java -Xmx1g -jar duracloudsync-{version}.jar
```

Alternatively, you can set the system environment variable JAVA_TOOL_OPTIONS to a value like "-Xmx1g", which will be picked up by the SyncTool on startup, meaning that you can start up the SyncTool UI as usual.

To run the SyncTool in command-line mode with 1 GB of heap memory space, download the Jar version of the SyncTool and execute the above command followed by the command line parameter values.

Large Files

When the SyncTool encounters a large file (by default, this is 1 GB+, but this can be set up to 5GB via the --max-file-size parameter) it will "chunk" that file prior to transfer to DuraCloud. This means that the file will land in DuraCloud as multiple components with an associated manifest file to indicate the set of component files and the checksum of each. As part of this process, the SyncTool will create a local temporary file for each chunk prior to transfer, as this allows the tool to generate the checksum for that chunk, and also allows retries on failure. These temporary files are stored in the default java temp directory.

The number and size of temp files which may be created depends on the number of threads and the max chunk size settings. Each thread has the potential of creating one temp file at a time and the size of the temp files can be up to the max chunk size. So multiplying the number of threads setting by the max chunk size will tell you the maximum number of GBs that may be consumed on local storage at one time. The SyncTool removes temp files as transfers complete, but if the tool it terminated abruptly, some of those temp files can be orphaned (and may require manual cleanup.)

If you'd like to change the location of where temp files are stored, this can be done with the "java.io.tmpdir" system property. This can be done on the command line, by adding "-Djava.io.tmpdir=/yourpath" after "java" on the command line. Alternatively, you can set the system environment variable JAVA_TOOL_OPTIONS to this value ("-Djava.io.tmpdir=/yourpath") and it will be picked up as the tool starts.

Prerequisites

> As of DuraCloud version 7.0.0, the Sync Tool requires Java 11 to run. The latest version of Java can be downloaded from here.

- You must have Java version 11 or above installed on your local system. If Java is not installed, or if a previous version is installed, you will need to download and install Java. To determine if the correct version of Java is installed, open a terminal or command prompt and enter

  ```
  java -version
  ```

  The version displayed should be 11.0.0 or above. If running this command generates an error, Java is likely not installed.
- You must have downloaded the Sync Tool. It is available as a link near the top of this page.

Using the Sync Tool

- To run the Sync Tool, open a terminal or command prompt and navigate to the directory where the Sync Tool is located
- To display the help for the Sync Tool, run

  ```
   java -jar duracloudsync-{version}.jar --help
  ```

- When running the Sync Tool for the first time, you will need to use these options:

Short Option	Long Option	Argument Expected	Required	Description	Default Value (if optional)
-h	--host	Yes	Yes	The host address of the DuraCloud DuraStore application	
-r	--port	Yes	No	The port of the DuraCloud DuraStore application	443
-i	--store-id	Yes	No	The Store ID for the DuraCloud storage provider	The primary storage provider is used
-s	--space-id	Yes	Yes	The ID of the DuraCloud space where content will be stored	
-u	--username	Yes	Yes	The username necessary to perform writes to DuraStore	

-p	--password	Yes	No	The password necessary to perform writes to DuraStore. If not specified the sync tool will first check to see if an environment variable named "DURACLOUD_PASSWORD" exists, if it does exist the sync tool will use its value as the password, otherwise you will be prompted to enter the password. Please note that when using the environment variable or the -p parameter you must escape your password according the conventions of your commandline shell. If you're using bash for example, any dollar ($) or backslash (\) chars must be escaped with a backslash. So the password p$ssw\rd would need to be entered as p\$ssw\\rd. There are many other special characters that need to be escaped. Here is a list of bash special characters for your reference.	Not set
-c	--content-dirs	Yes	Yes	A list of the directory paths to monitor and sync with DuraCloud. If multiple directories are included in this list, they should be separated by a space.	
-j	--jump-start	No	No	This option indicates that the sync tool should not attempt to check if content to be synchronized is already in DuraCloud, but should instead transfer all content. This option is best used for new data sets.	Not set
-a	--prefix	Yes	No	A prefix to be added to the content IDs of all files in the content directories as they are added to DuraCloud. The same prefix applies to all files in all content directories. For example, if a content directory is C:/users/bob/pictures with one file in it, C:/users/bob/pictures/001.jpg, and the prefix value is "bobs-pictures/", the file would be given a DuraCloud content ID of bobs-pictures/001.jpg. Note that the name of the content directory is not included in the path, so if you would like for it to appear as part of the content ID, you will need to include it in the prefix. Also note that the prefix does not need to be a directory name, it can be any value. If, however, you would like for it to appear as a directory path, do not forget to end the prefix with a "/" character.	Not set
-w	--work-dir	Yes	No	The state of the sync tool is persisted to this directory. If not specified, this value will default to a directory named duracloud-sync-work in the user's home directory.	duracloud-sync-work
-f	--poll-frequency	Yes	No	The time (in ms) to wait between each poll of the sync-dirs	10000 (10 seconds)
-t	--threads	Yes	No	The number of threads in the pool used to manage file transfers	3
-m	--max-file-size	Yes	No	The maximum size of a stored file in GB (value must be between 1 and 5), larger files will be split into pieces	1
-n	--rename-updates <suffix>	No	No	Indicates that when a local file is changed, the original copy of the file in DuraCloud should be renamed prior to the new local version being transferred to DuraCloud. The newest version of the file will retain the original file name while older versions will have a suffix value along with a date appended to the name. For example, a local file named "myfile.txt" is copied to DuraCloud by the SyncTool. The local file is updated, and the SyncTool is run again with this parameter enabled. The result is that DuraCloud will contain "myfile.txt", which is the updated version of the file, and "myfile.txt.orig.<date>" (with <date> replaced by the date on which the file was updated) which is the original version of the file. If "myfile.txt" is updated again, another version file will be created. Specify an optional suffix to override default ("orig"). To prevent updates altogether, see option -o. (Note that this option cannot be used together with either the -o or the -d options.)	orig
-o	--no-update	No	No	Indicate that changed files should not be updated. In order to perform updates without overwriting, see option -n.	
-d	--sync-deletes	No	No	Indicates that deletes performed on files within the content directories should also be performed on those files in DuraCloud; if this option is not included all deletes are ignored	Not set
-x	--exit-on-completion	No	No	Indicates that the sync tool should exit once it has completed a scan of the content directories and synced all files; if this option is included, the sync tool will not continue to monitor the content dirs	Not set
-l	--clean-start	No	No	Indicates that the sync tool should perform a clean start, ensuring that all files in all content directories are checked against DuraCloud, even if those files have not changed locally since the last run of the sync tool	Not set
-e	--exclude	Yes	No	The full path to a file which specifies a set of exclusion rules. The purpose of the exclusion rules is to indicate that certain files or directories should not be transferred to DuraCloud. The rules must be listed one per line in the file. The rules will match only on the name of a file or directory, not an entire path, so path separators should not be included in rules. Rules are not case sensitive (so a rule "test.log" will match a file "test.LOG"). The rules may include wildcard characters ? and *. The ? matches a single character, while * matches 0 or more characters. Examples of valid rules: test.txt : Will match a file named "test.txt" test : Will match a file or directory named "test" test.* : Will match files like "test.jpg", "test.txt", "test.doc", etc *.log : Will match files named "test.log" or "daily-01-01-2050.log" as well as a directory named ".log" backup-19?? : Will match a directory named "backup-1999" but not "backup-190000" or "backup-2000"	Not set

- When the Sync Tool runs, it creates a backup of your configuration in the work directory that you specify. When running the tool again, you can make use of this file to keep from having to re-enter all of the options specified on the initial run. In this case you need only a single option:

Short Option	Long Option	Argument Expected	Required	Description
-g	--config-file	Yes	Yes	Read configuration from this file (a file containing the most recently used configuration can be found in the work-dir, named synctool.config)

Examples of commands to use the command line Sync Tool

- Command to sync the contents of a single local content directory to DuraCloud.

```
java -jar duracloudsync-{version}.jar -c C:\files\important -h test.duracloud.org -s important-dir-backup -u myname -p mypassword
```

- Command to sync the contents of multiple local content directories to DuraCloud.

```
java -jar duracloudsync-{version}.jar -c C:\files\important C:\Users\me\Documents\important -h test.
duracloud.org -s important-dir-backup -u myname -p mypassword
```

Runtime commands

- While the Sync Tool is running, these commands are available. Just type them on the command line where the tool is running. These commands are not available when running in exit-on-completion mode.

Short Command	Long Command	Description
x	exit	Tells the Sync Tool to end its activity and close
c	config	Prints the configuration of the Sync Tool (the same information is printed at startup)
s	status	Prints the current status of the Sync Tool
l <Level>	N/A	Changes the log level to <Level> (may be any of DEBUG, INFO, WARN, ERROR)
h	help	Prints the runtime command help

Running the Sync Tool in a server shell environment

As noted above, the Sync Tool can be run in one of two modes, one which allows it to run continually, and the other which allows it to exit once it completes transferring all current files. The mode you choose will determine the way in which you deploy the Sync Tool on a server. The following examples assume the use of the bash shell.

To start the Sync Tool in continually running mode, you would use a command like this:

```
nohup java -jar duracloudsync-{version}.jar {parameters} > ~/synctool-output.log 2>&1 &
```

In this case, the & at the end of the command instructs the command to run in the background, and the "nohup" at the beginning tells the command to continue running even when the terminal being used is closed or when you disconnect from the server machine. The output of the Sync Tool would be placed in a file called "synctool-output.txt" in the user's home directory.
In order for the Sync Tool to be run on startup when the server machine boots, additional settings will need to be added which depend on the operating system being used. In Ubuntu, for example, an Upstart script could be used for this purpose.
Running the Sync Tool in exit on completion mode works best when the tool is run on a scheduled basis. A popular choice for handling this type of task is the cron utility. To run daily using cron a script should be placed in /etc/cron.daily. The script would look something like:

```
#!/bin/bash

if ps -ef | grep -v grep | grep duracloudsync ; then
  echo 'DuraCloud Sync is Running'
  exit 0
else
  echo 'Starting DuraCloud Sync'
  java -jar duracloudsync-{version}.jar -x [parameters] >> ~/synctool-output.log 2>&1 &
  exit 0
fi
```

The -x parameter is included here to ensure the Sync Tool exists after completing its run. This script also includes a check to ensure that the tool is not already running before starting.