File System Shell Guide

Table of contents

1. Overview

The File System (FS) shell includes various shell-like commands that directly interact with the Hadoop Distributed File System (HDFS) as well as other file systems that Hadoop supports, such as Local FS, HFTP FS, S3 FS, and others. The FS shell is invoked by:

```
bin/hdfs dfs <args>
```

All FS shell commands take path URIs as arguments. The URI format is *scheme://autority/path*. For HDFS the scheme is *hdfs*, and for the Local FS the scheme is *file*. The scheme and authority are optional. If not specified, the default scheme specified in the configuration is used. An HDFS file or directory such as */parent/child* can be specified as *hdfs://namenodehost/parent/child* or simply as */parent/child* (given that your configuration is set to point to *hdfs://namenodehost*).

Most of the commands in FS shell behave like corresponding Unix commands. Differences are described with each of the commands. Error information is sent to *stderr* and the output is sent to *stdout*.

1.1. cat

`Usage: hdfs dfs -cat URI [URI …]`

Copies source paths to *stdout*.

Example:

- `hdfs dfs -cat hdfs://nn1.example.com/file1 hdfs://nn2.example.com/file2`
- `hdfs dfs -cat file:///file3 /user/hadoop/file4`

Exit Code:
`Returns 0 on success and -1 on error.`

1.2. chgrp

`Usage: hdfs dfs -chgrp [-R] GROUP URI [URI …]`

Change group association of files. With `-R`, make the change recursively through the directory structure. The user must be the owner of files, or else a super-user. Additional information is in the Permissions Guide.

1.3. chmod

```
Usage: hdfs dfs -chmod [-R] <MODE[,MODE]... | OCTALMODE> URI
[URI …]
```

Change the permissions of files. With `-R`, make the change recursively through the directory structure. The user must be the owner of the file, or else a super-user. Additional information is in the Permissions Guide.

1.4. chown

```
Usage: hdfs dfs -chown [-R] [OWNER][:[GROUP]] URI [URI ]
```

Change the owner of files. With `-R`, make the change recursively through the directory structure. The user must be a super-user. Additional information is in the Permissions Guide.

1.5. copyFromLocal

```
Usage: hdfs dfs -copyFromLocal <localsrc> URI
```

Similar to **put** command, except that the source is restricted to a local file reference.

1.6. copyToLocal

```
Usage: hdfs dfs -copyToLocal [-ignorecrc] [-crc] URI
<localdst>
```

Similar to **get** command, except that the destination is restricted to a local file reference.

1.7. count

```
Usage: hdfs dfs -count [-q] <paths>
```

Count the number of directories, files and bytes under the paths that match the specified file pattern.
The output columns with `-count` are:
`DIR_COUNT, FILE_COUNT, CONTENT_SIZE FILE_NAME`
The output columns with `-count -q` are:
`QUOTA, REMAINING_QUATA, SPACE_QUOTA, REMAINING_SPACE_QUOTA, DIR_COUNT, FILE_COUNT, CONTENT_SIZE, FILE_NAME`

Example:

- `hdfs dfs -count hdfs://nn1.example.com/file1 hdfs://nn2.example.com/file2`
- `hdfs dfs -count -q hdfs://nn1.example.com/file1`

Exit Code:

```
Returns 0 on success and -1 on error.
```

1.8. cp

```
Usage: hdfs dfs -cp URI [URI …] <dest>
```

Copy files from source to destination. This command allows multiple sources as well in which case the destination must be a directory.
Example:

- `hdfs dfs -cp /user/hadoop/file1 /user/hadoop/file2`
- `hdfs dfs -cp /user/hadoop/file1 /user/hadoop/file2 /user/hadoop/dir`

Exit Code:

```
Returns 0 on success and -1 on error.
```

1.9. du

```
Usage: hdfs dfs -du [-s] [-h] URI [URI …]
```

Displays sizes of files and directories contained in the given directory or the length of a file in case its just a file.

Options:

- The `-s` option will result in an aggregate summary of file lengths being displayed, rather than the individual files.
- The `-h` option will format file sizes in a "human-readable" fashion (e.g 64.0m instead of 67108864)

Example:
`hdfs dfs -du /user/hadoop/dir1 /user/hadoop/file1 hdfs://nn.example.com/user/hadoop/dir1`
Exit Code:
`Returns 0 on success and -1 on error.`

1.10. dus

```
Usage: hdfs dfs -dus <args>
```

Displays a summary of file lengths. This is an alternate form of `hdfs dfs -du -s`.

1.11. expunge

```
Usage: hdfs dfs -expunge
```

Empty the Trash. Refer to the HDFS Architecture Guide for more information on the Trash feature.

1.12. get

```
Usage: hdfs dfs -get [-ignorecrc] [-crc] <src> <localdst>
```

Copy files to the local file system. Files that fail the CRC check may be copied with the `-ignorecrc` option. Files and CRCs may be copied using the `-crc` option.

Example:

- `hdfs dfs -get /user/hadoop/file localfile`
- `hdfs dfs -get hdfs://nn.example.com/user/hadoop/file localfile`

Exit Code:

```
Returns 0 on success and -1 on error.
```

1.13. getmerge

```
Usage: hdfs dfs -getmerge <src> <localdst> [addnl]
```

Takes a source directory and a destination file as input and concatenates files in src into the destination local file. Optionally `addnl` can be set to enable adding a newline character at the end of each file.

1.14. ls

```
Usage: hdfs dfs -ls <args>
```

For a file returns stat on the file with the following format:

```
permissions number_of_replicas userid groupid filesize
modification_date modification_time filename
```

For a directory it returns list of its direct children as in unix.A directory is listed as:

```
permissions userid groupid modification_date modification_time
dirname
```

Example:

```
hdfs dfs -ls /user/hadoop/file1
```

Exit Code:

```
Returns 0 on success and -1 on error.
```

1.15. lsr

```
Usage: hdfs dfs -lsr <args>
```
Recursive version of `ls`. Similar to Unix `ls -R`.

1.16. mkdir

```
Usage: hdfs dfs -mkdir <paths>
```

Takes path uri's as argument and creates directories. The behavior is much like unix mkdir -p creating parent directories along the path.

Example:

- `hdfs dfs -mkdir /user/hadoop/dir1 /user/hadoop/dir2`
- `hdfs dfs -mkdir hdfs://nn1.example.com/user/hadoop/dir hdfs://nn2.example.com/user/hadoop/dir`

Exit Code:

```
Returns 0 on success and -1 on error.
```

1.17. moveFromLocal

```
Usage: dfs -moveFromLocal <localsrc> <dst>
```

Similar to **put** command, except that the source `localsrc` is deleted after it's copied.

1.18. moveToLocal

```
Usage: hdfs dfs -moveToLocal [-crc] <src> <dst>
```

Displays a "Not implemented yet" message.

1.19. mv

```
Usage: hdfs dfs -mv URI [URI …] <dest>
```

Moves files from source to destination. This command allows multiple sources as well in which case the destination needs to be a directory. Moving files across file systems is not permitted.
Example:

- `hdfs dfs -mv /user/hadoop/file1 /user/hadoop/file2`
- `hdfs dfs -mv hdfs://nn.example.com/file1 hdfs://nn.example.com/file2 hdfs://nn.example.com/file3 hdfs://nn.example.com/dir1`

Exit Code:

`Returns 0 on success and -1 on error.`

1.20. put

`Usage: hdfs dfs -put <localsrc> ... <dst>`

Copy single src, or multiple srcs from local file system to the destination file system. Also reads input from stdin and writes to destination file system.

- `hdfs dfs -put localfile /user/hadoop/hadoopfile`
- `hdfs dfs -put localfile1 localfile2 /user/hadoop/hadoopdir`
- `hdfs dfs -put localfile hdfs://nn.example.com/hadoop/hadoopfile`
- `hdfs dfs -put - hdfs://nn.example.com/hadoop/hadoopfile`
  Reads the input from stdin.

Exit Code:

`Returns 0 on success and -1 on error.`

1.21. rm

`Usage: hdfs dfs -rm [-skipTrash] URI [URI …]`

Delete files specified as args. Only deletes files. If the `-skipTrash` option is specified, the trash, if enabled, will be bypassed and the specified file(s) deleted immediately. This can be useful when it is necessary to delete files from an over-quota directory. Refer to rmr for recursive deletes.
Example:

- `hdfs dfs -rm hdfs://nn.example.com/file`

Exit Code:

Returns 0 on success and -1 on error.

1.22. rmr

Usage: hdfs dfs -rmr [-skipTrash] URI [URI …]

Recursive version of delete. The rmr command recursively deletes the directory and any content under it. If the -skipTrash option is specified, the trash, if enabled, will be bypassed and the specified file(s) deleted immediately. This can be useful when it is necessary to delete files from an over-quota directory.
Example:

- hdfs dfs -rmr /user/hadoop/dir
- hdfs dfs -rmr hdfs://nn.example.com/user/hadoop/dir

Exit Code:

Returns 0 on success and -1 on error.

1.23. setrep

Usage: hdfs dfs -setrep [-R] <path>

Changes the replication factor of a file. -R option is for recursively increasing the replication factor of files within a directory.

Example:

- hdfs dfs -setrep -w 3 -R /user/hadoop/dir1

Exit Code:

Returns 0 on success and -1 on error.

1.24. stat

Usage: hdfs dfs -stat URI [URI …]

Returns the stat information on the path.

Example:

- hdfs dfs -stat path

Exit Code:
Returns 0 on success and -1 on error.

1.25. tail

```
Usage: hdfs dfs -tail [-f] URI
```

Displays last kilobyte of the file to stdout. -f option can be used as in Unix.

Example:

- `hdfs dfs -tail pathname`

Exit Code:
`Returns 0 on success and -1 on error.`

1.26. test

```
Usage: hdfs dfs -test -[ezd] URI
```

Options:
-e check to see if the file exists. Return 0 if true.
-z check to see if the file is zero length. Return 0 if true.
-d check to see if the path is directory. Return 0 if true.

Example:

- `hdfs dfs -test -e filename`

1.27. text

```
Usage: hdfs dfs -text <src>
```

Takes a source file and outputs the file in text format. The allowed formats are zip and TextRecordInputStream.

1.28. touchz

```
Usage: hdfs dfs -touchz URI [URI …]
```

Create a file of zero length.

Example:

- `hadoop -touchz pathname`

Exit Code:
`Returns 0 on success and -1 on error.`