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/hadoop fs <args>
```

All FS shell commands take path URIs as arguments. The URI format is `scheme://authority/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.

If HDFS is being used, `hdfs dfs` is a synonym.

See the Commands Manual for generic shell options.

appendToFile

Usage: `hadoop fs -appendToFile <localsrc> ... <dst>`

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

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

Exit Code:

Returns 0 on success and 1 on error.

cat

Usage: `hadoop fs -cat URI [URI ...]`

Copies source paths to stdout.

Example:

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

Exit Code:

Returns 0 on success and -1 on error.

checksum

Usage: `hadoop fs -checksum URI`

Returns the checksum information of a file.

Example:

- `hadoop fs -checksum hdfs://nn1.example.com/file1`
- `hadoop fs -checksum file:///etc/hosts`

chgrp

Usage: `hadoop fs -chgrp [-R] GROUP URI [URI ...]`

Change group association of files. The user must be the owner of files, or else a super-user. Additional information is in the Permissions Guide.

Options

- The -R option will make the change recursively through the directory structure.

chmod

Usage: `hadoop fs -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.

Options

- The -R option will make the change recursively through the directory structure.

chown

Usage: `hadoop fs -chown [-R] [OWNER][:[GROUP]] URI [URI ]`

Change the owner of files. The user must be a super-user. Additional information is in the Permissions Guide.

Options

- The -R option will make the change recursively through the directory structure.

copyFromLocal

Usage: `hadoop fs -copyFromLocal <localsrc> URI`

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

Options:

- The -f option will overwrite the destination if it already exists.

copyToLocal

Usage: `hadoop fs -copyToLocal [-ignorecrc] [-crc] URI <localdst>`

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

count

Usage: `hadoop fs -count [-q] [-h] [-v] <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, PATHNAME

The output columns with -count -q are: QUOTA, REMAINING_QUATA, SPACE_QUOTA, REMAINING_SPACE_QUOTA, DIR_COUNT, FILE_COUNT, CONTENT_SIZE, PATHNAME

The -h option shows sizes in human readable format.

The -v option displays a header line.

Example:

- `hadoop fs -count hdfs://nn1.example.com/file1 hdfs://nn2.example.com/file2`
- `hadoop fs -count -q hdfs://nn1.example.com/file1`
- `hadoop fs -count -q -h hdfs://nn1.example.com/file1`
- `hdfs dfs -count -q -h -v hdfs://nn1.example.com/file1`

Exit Code:

Returns 0 on success and -1 on error.

cp

Usage: `hadoop fs -cp [-f] [-p | -p[topax]] 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.

'raw.*' namespace extended attributes are preserved if (1) the source and destination filesystems support them (HDFS only), and (2) all source and destination pathnames are in the /.reserved/raw hierarchy. Determination of whether raw.* namespace xattrs are preserved is independent of the -p (preserve) flag.

Options:

- The -f option will overwrite the destination if it already exists.
- The -p option will preserve file attributes [topx] (timestamps, ownership, permission, ACL, XAttr). If -p is specified with no *arg*, then preserves timestamps, ownership, permission. If -pa is specified, then preserves permission also because ACL is a super-set of permission. Determination of whether raw namespace extended attributes are preserved is independent of the -p flag.

Example:

- `hadoop fs -cp /user/hadoop/file1 /user/hadoop/file2`
- `hadoop fs -cp /user/hadoop/file1 /user/hadoop/file2 /user/hadoop/dir`

Exit Code:

Returns 0 on success and -1 on error.

createSnapshot

See HDFS Snapshots Guide.

deleteSnapshot

See HDFS Snapshots Guide.

df

Usage: `hadoop fs -df [-h] URI [URI ...]`

Displays free space.

Options:

- The -h option will format file sizes in a "human-readable" fashion (e.g 64.0m instead of 67108864)

Example:

- `hadoop dfs -df /user/hadoop/dir1`

du

Usage: `hadoop fs -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:

- `hadoop fs -du /user/hadoop/dir1 /user/hadoop/file1 hdfs://nn.example.com/user/hadoop/dir1`

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

dus

Usage: `hadoop fs -dus <args>`

Displays a summary of file lengths.

Note: This command is deprecated. Instead use `hadoop fs -du -s`.

expunge

Usage: `hadoop fs -expunge`

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

find

Usage: `hadoop fs -find <path> ... <expression> ...`

Finds all files that match the specified expression and applies selected actions to them. If no *path* is specified then defaults to the current working directory. If no expression is specified then defaults to -print.

The following primary expressions are recognised:

- -name pattern
  -iname pattern

  Evaluates as true if the basename of the file matches the pattern using standard file system globbing. If -iname is used then the match is case insensitive.

- -print
  -print0Always

  evaluates to true. Causes the current pathname to be written to standard output. If the -print0 expression is used then an ASCII NULL character is appended.

The following operators are recognised:

- expression -a expression
  expression -and expression
  expression expression

  Logical AND operator for joining two expressions. Returns true if both child expressions return true. Implied by the juxtaposition of two expressions and so does not need to be explicitly specified. The second expression will not be applied if the first fails.

Example:

```
hadoop fs -find / -name test -print
```

Exit Code:

Returns 0 on success and -1 on error.

get

Usage: `hadoop fs -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:

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

Exit Code:

Returns 0 on success and -1 on error.

getfacl

Usage: `hadoop fs -getfacl [-R] <path>`

Displays the Access Control Lists (ACLs) of files and directories. If a directory has a default ACL, then getfacl also displays the default ACL.

Options:

- -R: List the ACLs of all files and directories recursively.
- *path*: File or directory to list.

Examples:

- `hadoop fs -getfacl /file`
- `hadoop fs -getfacl -R /dir`

Exit Code:

Returns 0 on success and non-zero on error.

getfattr

Usage: `hadoop fs -getfattr [-R] -n name | -d [-e en] <path>`

Displays the extended attribute names and values (if any) for a file or directory.

Options:

- -R: Recursively list the attributes for all files and directories.
- -n name: Dump the named extended attribute value.
- -d: Dump all extended attribute values associated with pathname.
- -e *encoding*: Encode values after retrieving them. Valid encodings are “text”, “hex”, and “base64”. Values encoded as text strings are enclosed in double quotes ("), and values encoded as hexadecimal and base64 are prefixed with 0x and 0s, respectively.
- *path*: The file or directory.

Examples:

- `hadoop fs -getfattr -d /file`
- `hadoop fs -getfattr -R -n user.myAttr /dir`

Exit Code:

Returns 0 on success and non-zero on error.

getmerge

Usage: `hadoop fs -getmerge [-nl] <src> <localdst>`

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

Examples:

- `hadoop fs -getmerge -nl /src /opt/output.txt`
- `hadoop fs -getmerge -nl /src/file1.txt /src/file2.txt /output.txt`

Exit Code:

Returns 0 on success and non-zero on error.

help

Usage: `hadoop fs -help`

Return usage output.

ls

Usage: `hadoop fs -ls [-d] [-h] [-R] <args>`

Options:

- -d: Directories are listed as plain files.
- -h: Format file sizes in a human-readable fashion (eg 64.0m instead of 67108864).
- -R: Recursively list subdirectories encountered.

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

```
permissions number_of_replicas userid groupid filesize modification_date modificatic
```

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
```

Files within a directory are order by filename by default.

Example:

- `hadoop fs -ls /user/hadoop/file1`

Exit Code:

Returns 0 on success and -1 on error.

lsr

Usage: `hadoop fs -lsr <args>`

Recursive version of ls.

Note: This command is deprecated. Instead use `hadoop fs -ls -R`

mkdir

Usage: `hadoop fs -mkdir [-p] <paths>`

Takes path uri's as argument and creates directories.

Options:

- The -p option behavior is much like Unix mkdir -p, creating parent directories along the path.

Example:

- `hadoop fs -mkdir /user/hadoop/dir1 /user/hadoop/dir2`
- `hadoop fs -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.

moveFromLocal

Usage: `hadoop fs -moveFromLocal <localsrc> <dst>`

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

moveToLocal

Usage: `hadoop fs -moveToLocal [-crc] <src> <dst>`

Displays a "Not implemented yet" message.

mv

Usage: `hadoop fs -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:

- `hadoop fs -mv /user/hadoop/file1 /user/hadoop/file2`
- `hadoop fs -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.

put

Usage: `hadoop fs -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.

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

Exit Code:

Returns 0 on success and -1 on error.

renameSnapshot

See HDFS Snapshots Guide.

rm

Usage: `hadoop fs -rm [-f] [-r |-R] [-skipTrash] URI [URI ...]`

Delete files specified as args.

Options:

- The -f option will not display a diagnostic message or modify the exit status to reflect an error if the file does not exist.
- The -R option deletes the directory and any content under it recursively.
- The -r option is equivalent to -R.
- The -skipTrash option will bypass trash, if enabled, and delete the specified file(s) immediately. This can be useful when it is necessary to delete files from an over-quota directory.

Example:

- `hadoop fs -rm hdfs://nn.example.com/file /user/hadoop/emptydir`

Exit Code:

Returns 0 on success and -1 on error.

rmdir

Usage: `hadoop fs -rmdir [--ignore-fail-on-non-empty] URI [URI ...]`

Delete a directory.

Options:

- `--ignore-fail-on-non-empty`: When using wildcards, do not fail if a directory still contains files.

Example:

- `hadoop fs -rmdir /user/hadoop/emptydir`

rmr

Usage: `hadoop fs -rmr [-skipTrash] URI [URI ...]`

Recursive version of delete.

Note: This command is deprecated. Instead use `hadoop fs -rm -r`

setfacl

Usage: `hadoop fs -setfacl [-R] [-b |-k -m |-x <acl_spec> <path>] |[--set <acl_spec> <path>]`

Sets Access Control Lists (ACLs) of files and directories.

Options:

- -b: Remove all but the base ACL entries. The entries for user, group and others are retained for compatibility with permission bits.
- -k: Remove the default ACL.
- -R: Apply operations to all files and directories recursively.
- -m: Modify ACL. New entries are added to the ACL, and existing entries are retained.
- -x: Remove specified ACL entries. Other ACL entries are retained.
- `--set`: Fully replace the ACL, discarding all existing entries. The *acl_spec* must include entries for user, group, and others for compatibility with permission bits.
- *acl_spec*: Comma separated list of ACL entries.
- *path*: File or directory to modify.

Examples:

- `hadoop fs -setfacl -m user:hadoop:rw- /file`
- `hadoop fs -setfacl -x user:hadoop /file`
- `hadoop fs -setfacl -b /file`
- `hadoop fs -setfacl -k /dir`
- `hadoop fs -setfacl --set user::rw-,user:hadoop:rw-,group::r--,other::r-- /file`
- `hadoop fs -setfacl -R -m user:hadoop:r-x /dir`
- `hadoop fs -setfacl -m default:user:hadoop:r-x /dir`

Exit Code:

Returns 0 on success and non-zero on error.

setfattr

Usage: `hadoop fs -setfattr -n name [-v value] | -x name <path>`

Sets an extended attribute name and value for a file or directory.

Options:

- -b: Remove all but the base ACL entries. The entries for user, group and others are retained for compatibility with permission bits.
- -n name: The extended attribute name.
- -v value: The extended attribute value. There are three different encoding methods for the value. If the argument is enclosed in double quotes, then the value is the string inside the quotes. If the argument is prefixed with 0x or 0X, then it is taken as a hexadecimal number. If the argument begins with 0s or 0S, then it is taken as a base64 encoding.
- -x name: Remove the extended attribute.
- *path*: The file or directory.

Examples:

- `hadoop fs -setfattr -n user.myAttr -v myValue /file`
- `hadoop fs -setfattr -n user.noValue /file`
- `hadoop fs -setfattr -x user.myAttr /file`

Exit Code:

Returns 0 on success and non-zero on error.

setrep

Usage: `hadoop fs -setrep [-R] [-w] <numReplicas> <path>`

Changes the replication factor of a file. If *path* is a directory then the command recursively changes the

replication factor of all files under the directory tree rooted at *path*.

Options:

- The -w flag requests that the command wait for the replication to complete. This can potentially take a very long time.
- The -R flag is accepted for backwards compatibility. It has no effect.

Example:

- `hadoop fs -setrep -w 3 /user/hadoop/dir1`

Exit Code:

Returns 0 on success and -1 on error.

stat

Usage: `hadoop fs -stat [format] <path> ...`

Print statistics about the file/directory at <path> in the specified format. Format accepts filesize in blocks (%b), type (%F), group name of owner (%g), name (%n), block size (%o), replication (%r), user name of owner(%u), and modification date (%y, %Y). %y shows UTC date as “yyyy-MM-dd HH:mm:ss” and %Y shows milliseconds since January 1, 1970 UTC. If the format is not specified, %y is used by default.

Example:

- `hadoop fs -stat "%F %u:%g %b %y %n" /file`

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

tail

Usage: `hadoop fs -tail [-f] URI`

Displays last kilobyte of the file to stdout.

Options:

- The -f option will output appended data as the file grows, as in Unix.

Example:

- `hadoop fs -tail pathname`

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

test

Usage: `hadoop fs -test -[defsz] URI`

Options:

- -d: f the path is a directory, return 0.
- -e: if the path exists, return 0.
- -f: if the path is a file, return 0.
- -s: if the path is not empty, return 0.
- -z: if the file is zero length, return 0.

Example:

- `hadoop fs -test -e filename`

text

Usage: `hadoop fs -text <src>`

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

touchz

Usage: `hadoop fs -touchz URI [URI ...]`

Create a file of zero length.

Example:

- `hadoop fs -touchz pathname`

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

truncate

Usage: `hadoop fs -truncate [-w] <length> <paths>`

Truncate all files that match the specified file pattern to the specified length.

Options:

- The `-w` flag requests that the command waits for block recovery to complete, if necessary. Without -w flag the file may remain unclosed for some time while the recovery is in progress. During this time file cannot be reopened for append.

Example:

- `hadoop fs -truncate 55 /user/hadoop/file1 /user/hadoop/file2`
- `hadoop fs -truncate -w 127 hdfs://nn1.example.com/user/hadoop/file1`

usage

Usage: `hadoop fs -usage command`

Return the help for an individual command.