

NAME

datapacker – Tool to pack files into the minimum number of bins

SYNOPSIS

datapacker [**-0**] [**-a ACTION**] [**-b FORMAT**] [**-d**] [**-p**] [**-S SIZE**] **-s SIZE FILE ...**

datapacker -h | --help

DESCRIPTION

datapacker is a tool to group files by size. It is designed to group files such that they fill fixed-size containers (called "bins") using the minimum number of containers. This is useful, for instance, if you want to archive a number of files to CD or DVD, and want to organize them such that you use the minimum possible number of CDs or DVDs.

In many cases, **datapacker** executes almost instantaneously. Of particular note, the hardlink action (see **OPTIONS** below) can be used to effectively copy data into bins without having to actually copy the data at all.

datapacker is a tool in the traditional Unix style; it can be used in pipes and call other tools.

OPTIONS

Here are the command-line options you may set for **datapacker**. Please note that **-s** and at least one file (see **FILE SPECIFICATION** below) is mandatory.

-0

--null When reading a list of files from standard input (see **FILE SPECIFICATION** below), expect the input to be separated by NULL (ASCII 0) characters instead of one per line. Especially useful with **find -print0**.

-a ACTION

--action=ACTION

Defines what action to take with the matches. Please note that, with any action, the output will be sorted by bin, with bin 1 first. Possible actions include:

print Print one human-readable line per file. Each line contains the bin number (in the format given by **-b**), an ASCII tab character, then the filename.

printfull

Print one semi-human-readable line per bin. Each line contains the bin number, then a list of filenames to place in that bin, with an ASCII tab character after the bin number and between each filename.

print0 For each file, output the bin number (according to the format given by **-b**), an ASCII NULL character, the filename, and another ASCII NULL character. Ideal for use with **xargs -0 -L 2**.

exec:COMMAND

For each file, execute the specified **COMMAND** via the shell. The program **COMMAND** will be passed information on its command line as indicated below.

It is an error if the generated command line for a given bin is too large for the system.

A nonzero exit code from any **COMMAND** will cause **datapacker** to terminate. If **COMMAND** contains quotes, don't forget to quote the entire command, as in:

```
datapacker '--action=exec:echo "Bin: $1"; shift; ls "$@"'
```

The arguments to the given command will be:

- `argv[0]` (`$0` in shell) will be the name of the shell used to invoke the command `--SHELL` or `/bin/sh`.
- `argv[1]` (`$1` in shell) will be the bin number, formatted according to `-b`.
- `argv[2]` and on (`$2` and on in shell) will be the files to place in that bin

hardlink

For each file, create a hardlink at `bin/filename` pointing to the original input filename. Creates the directory `bin` as necessary. Alternative locations and formats for `bin` can be specified with `-b`. All bin directories and all input must reside on the same filesystem.

After you are done processing the results of the bin, you may safely delete the bins without deleting original data. Alternatively, you could leave the bins and delete the original data. Either approach will be workable.

It is an error to attempt to make a hard link across filesystems, or to have two input files with the same filename in different paths. **datapacker** will exit on either of these situations.

See also `--deep-links`.

symlink

Like **hardlink**, but create symlinks instead. Symlinks can span filesystems, but you will lose information if you remove the original (pre-bin) data. Like **hardlink**, it is an error to have a single filename occur in multiple input directories with this option.

See also `--deep-links`.

-b *FORMAT*

--binfmt=*FORMAT*

Defines the output format for the bin name. This format is given as a `%d` input to a function that interprets it as `printf(3)` would. This can be useful both to define the name and the location of your bins. When running **datapacker** with certain arguments, the bin format can be taken to be a directory in which files in that bin are linked. The default is `%03d`, which outputs integers with leading zeros to make all bin names at least three characters wide.

Other useful variants could include `destdir/%d` to put the string "destdir/" in front of the bin number, which is rendered without leading zeros.

-d

--debug

Enable debug mode. This is here for future expansion and does not currently have any effect.

-D

--deep-links

When used with the `symlink` or `hardlink` action, instead of making all links in a single flat directory under the bin, mimic the source directory structure under the bin. Makes most sense when used with `-p`, but could also be useful without it if there are files with the same name in different source directories.

--help Display brief usage information and exit.

-p

--preserve-order

Normally, **datapacker** uses an efficient algorithm that tries to rearrange files such that the number of bins required is minimized. Sometimes you may instead wish to preserve the ordering of files

at the expense of potentially using more bins. In these cases, you would want to use this option.

As an example of such a situation: perhaps you have taken one photo a day for several years. You would like to archive these photos to CD, but you want them to be stored in chronological order. You have named the files such that the names indicate order, so you can pass the file list to **datapacker** using **-p** to preserve the ordering in your bins. Thus, bin 1 will contain the oldest files, bin 2 the second-oldest, and so on. If **-p** wasn't used, you might use fewer CDs, but the photos would be spread out across all CDs without preserving your chronological order.

-s *SIZE*

--size=*SIZE*

Gives the size of each bin in bytes. Suffixes such as "k", "m", "g", etc. may be used to indicate kilobytes, megabytes, gigabytes, and so forth. Numbers such as 1.5g are valid, and if needed, will be rounded to the nearest possible integer value.

The size of the first bin may be overridden with **-S**.

Here are the sizes of some commonly-used bins. For each item, I have provided you with both the underlying recording capacity of the disc and a suggested value for **-s**. The suggested value for **-s** is lower than the underlying capacity because there is overhead imposed by the filesystem stored on the disc. You will perhaps find that the suggested value for **-s** is lower than optimal for discs that contain few large files, and higher than desired for discs that contain vast amounts of small files.

- CD-ROM, 74-minute (standard): 650m / 600m
- CD-ROM, 80-minute: 703m / 650m
- CD-ROM, 90-minute: 790m / 740m
- CD-ROM, 99-minute: 870m / 820m
- DVD+-R: 4.377g / 4g
- DVD+R, dual layer: 8.5g / 8g

-S

--size-first

The size of the first bin. If not given, defaults to the value given with **-s**. This may be useful if you will be using a mechanism outside **datapacker** to add additional information to the first bin: perhaps an index of which bin has which file, the information necessary to make a CD bootable, etc. You may use the same suffixes as with **-s** with this option.

--sort Sorts the list of files to process before acting upon them. When combined with **-p**, causes the output to be sorted. This option has no effect save increasing CPU usage when not combined with **-p**.

FILE SPECIFICATION

After the options, you must supply one or more files to consider for packing into bins. Alternatively, instead of listing files on the command line, you may list a single hyphen (-), which tells **datapacker** to read the list of files from standard input (stdin).

datapacker never recurses into subdirectories. If you want a recursive search -- finding all files in a given directory and all its subdirectories -- see the second example in the EXAMPLES section below. **datapacker** is designed to integrate with **find(1)** in this situation to let you take advantage of **find**'s built-in powerful recursion and filtering features.

When reading files from standard input, it is assumed that the list contains one distinct filename per line. Seasoned POSIX veterans will recognize the inherent limitations in this format. For that reason, when given **-0** in conjunction with the single file -, **datapacker** will instead expect, on standard input, a list of files, each one terminated by an ASCII NULL character. Such a list can be easily generated with **find(1)** using its **-print0** option.

EXAMPLES

- Put all JPEG images in ~/Pictures into bins (using hardlinks) under the pre-existing directory ~/bins, no more than 600MB per bin:

```
datapacker -b ~/bins/%03d -s 600m -a hardlink ~/Pictures/*.jpg
```

- Put all files in ~/Pictures or any subdirectory thereof into 600MB bins under ~/bins, using hardlinking. This is a simple example to follow if you simply want a recursive search of all files.

```
find ~/Pictures -type f -print0 | \
datapacker -0 -b ~/bins/%03d -s 600m -a hardlink -
```

- Find all JPEG images in ~/Pictures or any subdirectory thereof, put them into bins (using hardlinks) under the pre-existing directory ~/bins, no more than 600MB per bin:

```
find ~/Pictures -name "*.jpg" -print0 | \
datapacker -0 -b ~/bins/%03d -s 600m -a hardlink -
```

- Find all JPEG images as above, put them in 4GB bins, but instead of putting them anywhere, calculate the size of each bin and display it.

```
find ~/Pictures -name "*.jpg" -print0 | \
datapacker -0 -b ~/bins/%03d -s 4g \
'--action=exec:echo -n "$1: "; shift; du -ch "$@" | grep total' \
-
```

This will display output like so:

```
/home/jgoerzen/bins/001: 4.0G total
/home/jgoerzen/bins/002: 4.0G total
/home/jgoerzen/bins/003: 4.0G total
/home/jgoerzen/bins/004: 992M total
```

Note: the grep pattern in this example is simple, but will cause unexpected results if any matching file contains the word "total".

- Find all JPEG images as above, and generate 600MB ISO images of them in ~/bins. This will generate the ISO images directly without ever hardlinking files into ~/bins.

```
find ~/Pictures -name "*.jpg" -print0 | \
datapacker -0 -b ~/bins/%03d.iso -s 4g \
'--action=exec:BIN="$1"; shift; mkisofs -r -J -o "$BIN" "$@"' \
-
```

You could, if you so desired, pipe this result directly into a DVD-burning application. Or, you could use growisofs to burn a DVD+R in a single step.

ERRORS

It is an error if any specified file exceeds the value given with -s or -S.

It is also an error if any specified files disappear while **datapacker** is running.

BUGS

Reports of bugs should be reported online at the **datapacker** homepage. Debian users are encouraged to instead use the Debian bug-tracking system.

COPYRIGHT

datapacker, and this manual, are Copyright (C) 2008 John Goerzen.

All code, documentation, and build scripts are under the following license unless otherwise noted:

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see <URL:<http://www.gnu.org/licenses/>>.

The GNU General Public License is available in the file COPYING in the source distribution. Debian GNU/Linux users may find this in /usr/share/common-licenses/GPL-3.

If the GPL is unacceptable for your uses, please e-mail me; alternative terms can be negotiated for your project.

AUTHOR

datapacker, its libraries, documentation, and all included files, except where noted, was written by John Goerzen <jgoerzen@complete.org> and copyright is held as stated in the COPYRIGHT section.

datapacker may be downloaded, and information found, from its homepage <URL:<http://software.complete.org/datapacker>>.

SEE ALSO

mkisofs(1), **genisoimage(1)**