NAME

chafa − Character art facsimile generator

SYNOPSIS

chafa [OPTION...] [IMAGE...]

DESCRIPTION

chafa is a command−line utility that converts image data, including animated GIFs, into graphics formats or ANSI/Unicode character art suitable for display in a terminal. It has broad feature support, allowing it to be used on devices ranging from historical teleprinters to modern terminal emulators and everything in between.

You can specify one or more input files, but the default behavior is slightly different with multiple files; for instance, animations will not loop forever when there is more than one input file.

GENERAL OPTIONS

−−files *path*

Read a list of paths to process from *path*, or '−' for stdin. The paths must be separated by newlines.

Can be specified multiple times. Any additional paths on the command line will be processed last.

−−files0 *path*

Read a list of paths to process from *path*, or '−' for stdin. The paths must be separated by NUL bytes. This is more robust if your file system allows newlines in filenames (as POSIX does). Useful in conjunction with other tools that support it, e.g.:

find . −type f −print0 | chafa −−files0 −

Can be specified multiple times. Any additional paths on the command line will be processed last.

−h, −−help

Show a brief help text.

−−probe *arg*

Probe terminal's capabilities and wait for response [auto, on, off]. A positive real number denotes the maximum time to wait for a response, in seconds. Defaults to 5.0.

−−probe−mode *mode*

How to probe the terminal [any, ctty, stdio]. "Any" is the default and will probe by any means available. "Ctty" will probe the controlling tty, usually /dev/tty, which is useful when chafa is part of a pipeline. This may block if done in the background. "Stdio" will only consider standard input/output.

−−version

Show version, feature and copyright information.

OUTPUT ENCODING

−f, −−format *format*

Set output format; one of [iterm, kitty, sixels, symbols]. The default is iterm, kitty or sixels if the connected terminal supports one of these, falling back to symbols ("ANSI art") otherwise.

−O *num*, **−−optimize** *num*

Compress the output by using control sequences intelligently [0−9]. 0 disables, 9 enables every available optimization. Defaults to 5, except for when used with "−c none", where it defaults to 0.

−−relative *bool*

Use relative cursor positioning [on, off]. When on, control sequences will be used to position images relative to the cursor. When off, newlines will be used to separate rows instead for e.g. 'less −R' interop. Defaults to off.

−−passthrough *mode*

Graphics protocol passthrough [auto, none, screen, tmux]. Used to show pixel graphics from within

multiplexers. Defaults to auto, which will enable passthrough if the Kitty terminal is detected along with one of the supported multiplexers. Other combinations must be enabled manually; use with the −f option to select the appropriate graphics protocol.

−−polite *bool*
Polite mode [on, off]. Inhibits escape sequences that on rare occasions may confuse the terminal or other programs. Defaults to off.

SIZE AND LAYOUT

−−align *ALIGN*
Align images in viewport. The following alignments are understood: left, right, top, bottom, hcenter, vcenter, center. Two orthogonal alignments can be separated by a comma, e.g. "center,right". The meaning of "center" depends on context, and defaults to "hcenter" if ambiguous. "center,center" will center along both axes.

Centering vertically makes sense when used together with "−−clear", or possibly as part of a scheme where the cursor is pre−positioned at the top−left corner of the view, or a subview when used with "−−relative on".

−C *bool*, **−−center** *bool*
Center images horizontally in the view [on, off]. Defaults to off. This option is deprecated; use "−−align center" instead.

−−clear
Clear screen before processing each file.

−−exact−size *mode*
Try to match the input's size exactly [auto, on, off]. When on, this will override other sizing options and produce output images at the exact pixel size of the inputs. In auto mode, scaling will be avoided (in exchange for padding) if the output size is equal to or slightly bigger than the input. When off, padding will never be added, and the image is scaled to fit the containing cell extent. Defaults to auto.

−−fit−width
Fit images to the view's width, potentially exceeding its height.

−−font−ratio *width*/*height*
Target font's width/height ratio. Can be specified as a real number or a fraction. Defaults to 1/2. This will only be applied in symbol mode.

−−grid *cols*x*rows*
Lay out images in a grid of *cols* columns and *rows* rows per screenful. Either *cols* or *rows* may be omitted, e.g. −−grid 4 or −−grid x4, in which case cell allocations will be approximately square. If "auto" is specified, dimensions will be picked automatically.

−g
Alias for "−−grid auto".

−−label *bool*
Labeling [on, off]. When this is enabled, each image will be labeled with its filename. Defaults to off.

−l
Alias for "−−label on".

−−margin−bottom *num*
When terminal size is detected, reserve at least this many rows at the bottom as a safety margin. Can be used to prevent images from scrolling out. Defaults to 1.

−−margin−right *num*
When terminal size is detected, reserve at least this many columns on the right−hand side as a safety margin. Defaults to 0.

−−scale *num*
Scale image, respecting terminal's maximum dimensions. 1.0 approximates original pixel dimensions.

Specify "max" to use all available space. Defaults to 1.0 for pixel graphics and 4.0 for symbols.

−s *width***x***height***, −−size** *width***x***height*
Set maximum output image dimensions in columns and rows. By default this will be equal to the view size (see −−view−size).

−−stretch
Stretch image to fit output dimensions; ignore aspect. Implies −−scale max.

−−view−size *width***x***height*
Set the view size in columns and rows. By default this will be the size of your terminal, or 80x25 if size detection fails. If one dimension is omitted (by providing a size of e.g. 80x or x25), it will be set to a reasonable approximation of infinity.

ANIMATION AND TIMING

−−animate *bool*
Whether to allow animation [on, off]. Defaults to on. When off, will show a still frame from each animation.

−d, −−duration *seconds*
Time to show each file, in seconds. Defaults to zero for still images and for animations when multiple files are specified. If a single animation is specified, defaults to infinite, or "inf". Animations will always be played through at least once, even if duration is e.g. zero. See the "Duration" section for more.

−−speed *speed*
Set the speed animations will play at. This can be either a unitless multiplier (fractions are allowed), or a real number followed by "fps" to apply a specific framerate.

−−watch
Watch a single input file, redisplaying it whenever its contents change. Will run until manually interrupted or, if −−duration is set, until it expires.

COLORS AND PROCESSING

−−bg *color*
Background color of display (color name or hex). Partially transparent input will be blended with this color. Color names are based on those provided with X.Org. Defaults to black, or the terminal's default background color when probing is enabled (see −−probe).

−c *mode***, −−colors** *mode*
Set output color mode; one of [none, 2, 8, 16/8 16, 240, 256, full]. The 240−color mode is recommended over the 256−color one, since the lower 16 colors are unreliable and tend to differ between terminals. 16−color mode will use aixterm extensions to produce 16 foreground and background colors. The 16/8 mode allows for 8 colors plus another "bright" 8 colors in the foreground implemented with the "bold" escape sequence. 2−color mode will only emit the ANSI codes for reverse color and attribute reset, while "none" will emit no escape sequences at all.

In sixel mode, "full" will dynamically generate a 256−color palette for each image or animation frame. The other modes refer to built−in palettes. "none" and "2" are interchangeable and will use the specified foreground/background colors (see −−fg and −−bg).

If left unspecified, an optimal default will be chosen based on the current environment.

−−color−extractor *extractor*
Method for extracting color from an area; one of [average, median]. Median normally produces crisper output, while average may perform better on noisy images. Defaults to average.

−−color−space *cs*
Color space used for quantization; one of [rgb, din99d]. Defaults to rgb, which is faster but less accurate.

−−dither *type*

Type of dithering to apply during quantization. One of [none, ordered, diffusion, noise]. "Bayer" is a synonym for "ordered", and "fs" (Floyd−Steinberg) is a synonym for "diffusion". Defaults to "noise" in sixel mode, otherwise "none".

−−dither−grain *width***x***height*
Dimensions of grain used when dithering. Specified as width x height, where each can be one of [1, 2, 4, 8] pixels. One character cell is by definition 8 pixels across in both dimensions. Defaults to 4x4 in symbol mode and 1x1 in sixel mode.

−−dither−intensity *intensity*
Intensity of dithering pattern. Ranges from 0.0 to infinity, with 1.0 considered neutral. Lower values tend to reduce the amount of dithering done, while higher values increase it. In practice, values higher than 10.0 are unlikely to produce useful results.

−−fg *color*
Foreground color of display (color name or hex). Together with the background color specified by −−bg, this specifies the terminal's palette in color modes 2 and none. Color names are based on those provided with X.Org. Defaults to white, or the terminal's default foreground color when probing is enabled (see −−probe).

−−invert
Invert video. For display with bright backgrounds in color modes 2 and none. Swaps −−fg and −−bg.

−p *bool***, −−preprocess** *bool*
Image preprocessing [on, off]. Defaults to on with 16 colors or lower, off otherwise. This enhances colors and contrast prior to conversion, which can be useful in low−color modes.

−t *threshold***, −−threshold** *threshold*
Threshold above which full transparency will be used [0.0 − 1.0]. Setting this to 0.0 will render a blank image, while a value of 1.0 will replace any transparency with the background color (configurable with −−bg).

RESOURCE ALLOCATION

−−threads *num*
Maximum number of CPU threads to use. If left unspecified or negative, this will equal available CPU cores.

−w *num***, −−work** *num*
How hard to work in terms of CPU and memory [1−9]. 1 is the cheapest, 9 is the most accurate. Defaults to 5.

EXTRA OPTIONS FOR SYMBOL ENCODING

−−fg−only
Leave the background color untouched. This produces character−cell output using foreground colors only, and will avoid resetting or inverting the colors.

−−fill *symbols*
Specify character symbols to use for fill/gradients. Defaults to none. Usage is similar to that of −−symbols; see below.

−−glyph−file *file*
Load glyph information from file, which can be any font file supported by FreeType (TTF, PCF, etc). The glyph outlines will replace any existing outlines, including builtins. Useful in symbol mode for custom font support or for improving quality with a specific font. Note that this only makes sense if the output terminal is using a matching font. Can be specified multiple times.

−−symbols *symbols*
Specify character symbols to employ in final output. See below for full usage and a list of symbol classes.

EXIT STATUS

chafa will return 0 on success, 1 on partial failure or 2 on complete failure (including when invoked with no arguments).

Status	*Meaning*
0	Success
1	Some files failed to display
2	All files failed to display

SYMBOLS

Accepted classes for −−symbols and −−fill are [all, none, space, solid, stipple, block, border, diagonal, dot, quad, half, hhalf, vhalf, inverted, braille, technical, geometric, ascii, legacy, sextant, wedge, wide, narrow]. Some symbols belong to multiple classes, e.g. diagonals are also borders.

You can add specific characters with the letter "u" followed by a hexadecimal code point, e.g. "ue080", or a range of code points by separating the first and last index by "..", e.g. "u100..u200".

Symbol sets can also be specified as a string of UTF−8 characters in square brackets, e.g. [abcd]. To include a closing bracket in the set, escape it with a backslash.

You can specify a list of classes separated by commas, or prefix them with + and − to add or remove symbols relative to the existing set. The ordering is significant.

The default symbol set is block+border+space−wide−inverted for all modes except "none", which uses block+border+space−wide (including inverse symbols).

DURATION

In order to accommodate both interactive use and batch processing, an animation's duration is determined according to a few simple rules:

1. If one or more **−−duration** arguments are present, the final instance is respected and applied to every file.
2. Otherwise, if there's a controlling terminal attached (indicating there's an interactive session), and only a single file argument is provided, and that file is an animation, it will have infinite duration.
3. Otherwise (no controlling terminal, multiple files, file is a still image), duration will be zero, causing animations to play once and then stop.

EXAMPLES

chafa in.gif

Show a potentially animated GIF image in the terminal. If this is an animation, it will run until the user generates an interrupt (typically ctrl−c). All parameters will be autodetected based on the current environment.

chafa −c full −s 200 in.gif

Like the above, but force truecolor output that is 200 characters wide and calculate the height preserving the aspect of the original image.

chafa −c 16 −−color−space din99d −−symbols −dot in.jpg

Generate 16−color output with perceptual color picking and avoid using dot symbols.

chafa −c none −−symbols block+border−solid in.png

Generate uncolored output using block and border symbols, but avoid the solid block symbol.

FURTHER READING

See the **Chafa homepage**[1] for more information.

AUTHOR

Written by **Hans Petter Jansson**[2] <hpj@hpjansson.org>.

NOTES

1. Chafa homepage
   https://hpjansson.org/chafa/
2. Hans Petter Jansson
   https://hpjansson.org/