Package: tinyimg (via r-universe)

June 1, 2026

Title Optimize and Compress Images

Version 0.4.5

Description Optimize and compress images using 'Rust' libraries to reduce file sizes while maintaining image quality. Supports PNG palette reduction and dithering via the 'exoquant' crate before lossless PNG optimization via the 'oxipng' crate, and JPEG re-encoding via the 'mozjpeg' crate. The package provides functions to optimize individual image files or entire directories, with configurable compression levels. Use tinyimg() as a convenient entry point for mixed PNG/JPEG workflows.

License MIT + file LICENSE

URL https://github.com/yihui/tinyimg

BugReports https://github.com/yihui/tinyimg/issues

SystemRequirements Cargo (Rust's package manager), rustc (>= 1.82.0)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Suggests testit

Config/rextendr/version 0.3.1

Config/pak/sysreqs libclang-dev

Repository https://yihui.r-universe.dev

Date/Publication 2026-06-01 03:31:42 UTC

RemoteUrl https://github.com/yihui/tinyimg

RemoteRef HEAD

RemoteSha 123c4a653fd4c567ae9419a49eac7b24e20da8b1

Contents

`tinyimg` *Optimize PNG and JPEG images*

Description

`tinyimg()` dispatches PNG files to `tinypng()` and JPEG files to `tinyjpg()`. `tinypng()` optimizes PNG files using optional lossy palette reduction and dithering (via the `exoquant` crate) before lossless compression (via the `oxipng` crate). `tinyjpg()` optimizes JPEG files by re-encoding with mozjpeg's optimized Huffman coding and quantization tables.

Usage

```
tinyimg(
  input,
  output = tiny_output,
  recursive = TRUE,
  verbose = TRUE,
  level = 2L,
  quality = 75,
  lossy = 0,
  ...
)

tinyjpg(
  input,
  output = tiny_output,
  quality = 75,
  recursive = TRUE,
  verbose = TRUE
)

tiny_output(input, lossy = 0, quality = 75)

tinypng(
  input,
  output = tiny_output,
  level = 2L,
  alpha = FALSE,
  preserve = TRUE,
  recursive = TRUE,
  verbose = TRUE,
  lossy = 0
)
```

Arguments

input	Path to an image file, a character vector of image file paths, or a directory. `tinyimg()` accepts `.png`, `.apng`, `.jpg`, and `.jpeg` files; `tinypng()` accepts `.png` and `.apng`; `tinyjpg()` accepts `.jpg` and `.jpeg`.
output	Path to the output file or directory, a function that maps input paths to output paths, or `identity` to optimize in place. Defaults to `tiny_output()`, which adds a suffix encoding the optimization parameters so that the original file is never overwritten by a lossy result.
recursive	When `input` is a directory, also search subdirectories.
verbose	Print file size change info for each file.
level	PNG optimization level (0–6). Higher values give better compression but take longer. Passed to `tinypng()` by `tinyimg()`.
quality	JPEG quality level (0–100). Higher quality means larger files; lower quality means smaller files. Passed to `tinyjpg()` by `tinyimg()`. `tiny_output()` appends `_q<value>` when `quality < 100`.
lossy	Numeric threshold for per-color ΔE_{76} in lossy PNG palette reduction. Values `<= 0` disable lossy optimization. See Details. Passed to `tinypng()` by `tinyimg()` via `...`. When `> 0`, `tiny_output()` appends `_l<value>` to the output filename.
...	Additional arguments passed from `tinyimg()` to `tinypng()` (e.g., `alpha`, `preserve`).
alpha	Optimize transparent pixels in PNG files for better compression. This is technically lossy but visually lossless.
preserve	Preserve file permissions and timestamps when optimizing PNG files. Ignored when `lossy > 0`.

Details

`tiny_output()` generates output file paths by appending a suffix that encodes the optimization parameters, e.g., `foo_l2.3.png` for lossy PNG (`lossy = 2.3`) and `foo_q70.jpg` for JPEG at `quality = 70`. It is the default `output` for all three optimizers so that lossy or quality-reduced results are never silently written over the original files. Pass `output = identity` to optimize in place (lossless PNG optimization is always safe to do in place).

The lossy PNG algorithm uses color difference in the CIE 1976 $L^*a^*b^*$ color space. For a candidate palette size `n`, the image is quantized with `n` colors using nearest-color mapping. Sampled pixels are then **grouped by their original color**, so each distinct color gets one equal vote regardless of how many pixels share it (preventing a large uniform background from masking errors in rarer content colors). The worst-case ΔE_{76} within each group is recorded, and the 95th percentile of those per-color values is taken. Bisection on `n` (1–256) finds the smallest palette whose per-color p95 is `<= lossy`.

$\Delta E_{76} \approx 2.3$ is the just noticeable difference (JND) threshold. Larger values allow more color difference and smaller palettes, with more loss of color fidelity.

Value

`tinyimg()`, `tinypng()`, and `tinyjpg()` invisibly return a character vector of output file paths. `tiny_output()` returns a character vector of output file paths (visibly).

References

https://en.wikipedia.org/wiki/Color_difference

Examples

```
# Create test images
tmp_png = tempfile(fileext = ".png")
png(tmp_png, width = 400, height = 400); plot(1:10); dev.off()

tmp_jpg = tempfile(fileext = ".jpg")
jpeg(tmp_jpg, width = 400, height = 400); plot(1:10); dev.off()

# Optimize both in one call (uses tiny_output() by default)
tinyimg(c(tmp_png, tmp_jpg))

# Optimize in place (lossless PNG is safe; use with care for JPEG)
tinypng(tmp_png, identity)
tinyjpg(tmp_jpg, identity)

# Lossy PNG: output gets a suffix automatically
tinypng(tmp_png, lossy = 2.3)

# JPEG at a specific quality
tinyjpg(tmp_jpg, quality = 60)

# See what output names tiny_output() would generate
tiny_output(c(tmp_png, tmp_jpg), lossy = 2.3, quality = 60)
```

Index