

PhotoCraft

ALPHA

clone of Adobe Photoshop

Layers, masks, adjustment layers, type and real PSD files, in pure Rust. You describe the edit; Claude runs engine commands through the `photocraft` MCP server and shows you a flattened preview of the result.

v0.9.0

Driven from Claude Code via MCP
Richard Young • richard@deepneuro.ai
2026-10-15

Alpha software. PhotoCraft is a pre-1.0 ArtCraft app. Expect missing features, rough edges and the occasional crash; keep originals safe and let Claude work on copies.

25

MCP tools

909

engine commands

2

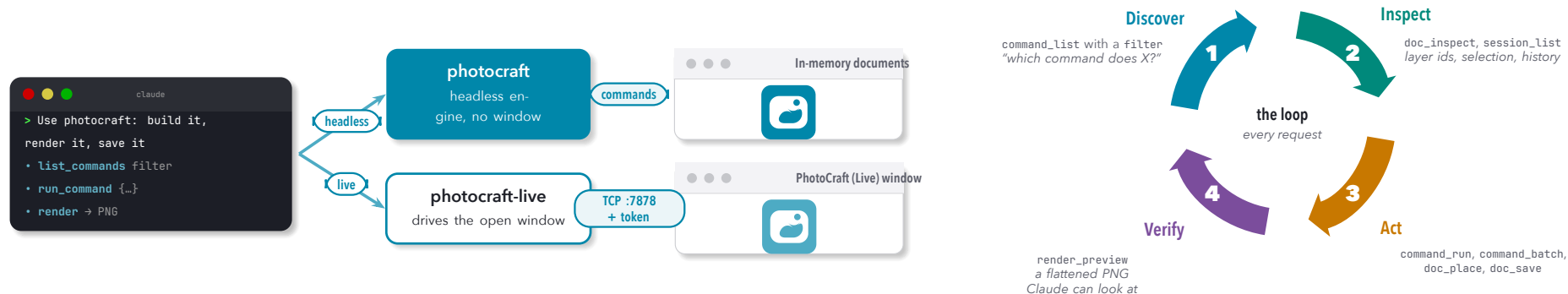
MCP servers

16

adjustment-layer kinds

01 The idea in one picture

The UI, the CLI and MCP all call the same command registry, so most things you can click, Claude can run. You type a goal; Claude finds the command id, runs it, and `render_preview` hands back a PNG so both of you can check the result before anything is saved.



🔍 **Check it is wired up:** `/mcp` in Claude Code should list `photocraft` (headless, 25 tools) as connected; `photocraft-live` connects only while the Live window is open and is the only place the `ui_*` tools work. Files must sit beneath the automation roots, usually your home folder.

02 First five minutes

The image on this page is real output from PhotoCraft, captured through the MCP server while writing this guide. Paste the prompts as they are.

- 1 **Make a blank canvas** “Use photocraft: new 1200 x 800 document, dark blue background, render a preview.” A PNG comes back. That proves the connection works.
- 2 **Add type and a style** Two text layers and a drop shadow (storyboard 1). Each command returns the id of the layer it made.
- 3 **Explore a command family** “List the photocraft commands matching `select.`” shows every selection command with its parameter doc.
- 4 **Look, then fix** Always end with “render a preview”. When a step misfires, Claude reads `doc_inspect`, deletes the layer and tries again.
- 5 **Save** “Save as ~/Pictures/title.pcraft and export a PNG next to it.” `.pcraft` is lossless; other extensions export.

Storyboard 1: a title card from nothing

YOU, IN CLAUDE CODE

"Use photocraft. New 1200 x 800 document, background #1e3a5f. Put "PhotoCraft" in white 140 px type with a soft drop shadow, a yellow subtitle "driven by Claude over MCP" under it, and a blurred yellow glow on the right. Render a preview."

MCP CALLS CLAUDE MADE

- 1 `doc_new` {"width": 1200, "height": 800, "background": "#1e3a5f"}
- 2 `command_batch` type.create({"x": 80, "y": 300, "text": "PhotoCraft", "size": 140, "color": "#ffffff"} → {"layer": 3}); layer.layerStyle.dropShadow {"distance": 12, "size": 20, "opacity": 60}; type.create subtitle → {"layer": 4}
- 3 `command_batch` layer.new.layer "Glow"; select.rect {"x": 800, "y": 150, "width": 360, "height": 360, "ellipse": true}; edit.fill {"color": "#ffd166"}; select.deselect; filter.blur.gaussianBlur {"radius": 45}; layer.arrange.sendToBack; layer.arrange.bringForward
- 4 `render_preview` {"max_side": 1200}

WHAT CAME BACK



- 1 Type layer (id 3) with `layer.layerStyle.dropShadow` applied to it.
- 2 Second type layer (id 4): `type.create` returns the id, never guessed.
- 3 "Glow": `select.rect` with `ellipse` true, `edit.fill`, Gaussian blur 45. There is no `select.ellipse`, and `edit.fill` with no selection fills the whole layer, which is why the first try was removed with `layer.delete`.

A photo-editing conversation

The same loop on a real photograph. This flow (remove the background, place a new one, grade it with adjustment layers) was run end to end on the headless server when the first edition of this guide was written.

YOU, IN CLAUDE CODE

"Open ~/Pictures/portrait.jpg, cut the person out, put ~/Pictures/beach.jpg behind them, add a gentle S-curve and a little vibrance as adjustment layers, show me, then save a PSD and a JPEG."

MCP CALLS CLAUDE MADE

- 1 `doc_open` {"path": "Pictures/portrait.jpg"} → index 0, size, import warnings
- 2 `doc_inspect` the photo is one pixel layer, id 2 (ids are not 0/1-based)
- 3 `command_run` layer.removeBackground {"refine": true} (Select Subject, local, no cloud)
- 4 `doc_place` {"path": "Pictures/beach.jpg", "fit": true} as an embedded Smart Object
- 5 `command_batch` layer.arrange.sendToBack; layer.selectTop; layer.newAdjustmentLayer.curves {"points": [[0,0],[64,48],[192,212],[255,255]]}; layer.newAdjustmentLayer.vibrance {"vibrance": 20}
- 6 `render_preview` {"max_side": 1024} for a look
- 7 `doc_save` {"path": "Pictures/portrait-beach.psd"} then {"path": "Pictures/portrait-beach.jpg", "quality": 90}

CLAUDE

Done. The subject mask came from Select Subject; the beach sits below it as a Smart Object; curves and vibrance are adjustment layers above both, so everything stays editable in the PSD. The JPEG is the flattened result. I read the warnings list `doc_save` returned for the PSD and kept a lossless .pcraft copy as well, in case anything PhotoCraft-only was dropped.

Prompts to explore with

●●● warm-up ●●● real work ●●● stretch

●●●

"Use photocraft: list the commands matching "adjustmentLayer" and show me their parameters."

`command_list` with a `filter`: 16 adjustment-layer kinds, each with ranges and defaults.

●●●

"Open ~/Pictures/any.psd, describe the layer tree and anything that did not import cleanly, and render a preview."

`doc_open` returns warnings; `doc_inspect` and `render_preview` change nothing.

●●●

"Resize every JPEG in ~/Pictures/shoot to 1600px wide at quality 85 into ~/Pictures/web."

Claude loops open, `image.imageSize`, save, close per file.

●●●

"Remove the trash can at about x 900-1050, y 600-800 in ~/Pictures/street.jpg and show me before and after."

`select.rect` then `edit.contentAwareFill`; two previews to compare.



"Export each visible layer of ~/Pictures/mockup.psd as PNG into a fresh folder."

`doc_export_layers`; refuses to overwrite, so use a new folder.



"Make the sky in ~/Pictures/lake.jpg deeper blue without touching the foreground, then prepare a coated-CMYK TIFF for print."

`select.sky` masks an adjustment layer; `edit.convertToProfile` for print.



"One speaker card per row of ~/Pictures/speakers.csv, swapping the Name and Title text layers in card.psd; show me the first three."

Variables, data sets, export, render: the whole loop several times over.

03 Goals → how

Each card is a prompt you can paste, the tools Claude reaches for, and the command ids doing the work. Command ids run through `command_run` (one) or `command_batch` (several, one undo step each). Flows marked *verified* were run headless against the 0.9.0 registry.

Batch work & export

1 Batch-resize a folder

ASK CLAUDE

"Resize every JPEG in ~/Pictures/shoot to 1600 px wide, quality 85, into ~/Pictures/web."

TOOLS `doc_open` → `command_run` → `doc_save`
→ `doc_close`

KEY `image.imageSize` `width`, `resample`; `doc_save` `quality`.

● Big folder? CLI is faster: `photocraft-cli batch` (verified).

2 Convert PSDs to PNG / JPEG / WebP / TIFF

ASK CLAUDE

"Export poster.psd as a lossless WebP and a layered TIFF next to it."

TOOLS `doc_open` → `doc_export`

KEY `path` (extension = format), `quality`, `tiffLayers`, `tgaBits`.

● WebP with a `quality` is lossy; without, lossless.

3 Apply a Photoshop .atn action

ASK CLAUDE

"Play 'Warm Fade' from Film Look.atn on every image in ~/Pictures/raw."

TOOLS `command_run` → `actions.load`
→ `actions.play`

KEY `actions.load` `data` = base64 of the .atn;
`actions.play` `action`.

● Or CLI: `photocraft-cli batch --actions set.atn --action NAME`.

4 Export every layer as a file

ASK CLAUDE

"Export each visible layer of mockup.psd as PNG into ~/Exports/mockup-layers."

TOOLS `doc_open` → `doc_export_layers`

KEY `dir`, `format`, `prefix`, `visible_only` (verified).

● Existing files are an error, never overwritten: use a fresh folder.

5 Templated cards from a CSV

ASK CLAUDE

"One speaker card per row of speakers.csv, swapping the Name and Title text layers."

TOOLS `command_run` → `doc_import_data_sets`
→ `doc_export_data_sets`

KEY `image.variables.define` `defs`
(`textReplacement`, ...); export `naming`.

● Colliding filenames are refused before anything is written.

6 Stitch a panorama / merge HDR

ASK CLAUDE

"Stitch the five photos in ~/Pictures/pano into one panorama PSD."

TOOLS Bash: `photocraft-cli run --new ... --cmd file.automate.photomerge`

KEY `file.automate.photomerge` `paths`, `layout`, `blend`.

● Refused over MCP; the CLI route was verified.

Retouching & selections

7 Remove the background

ASK CLAUDE

"Cut the product out of shoe.jpg and save a transparent PNG."

TOOLS `doc_open` → `command_run` → `doc_save`

KEY `layer.removeBackground` `refine` (default true).
Verified.

- Mask comes from Select Subject; runs locally, no cloud.

8 Replace the background

ASK CLAUDE

"Put the person in portrait.jpg in front of beach.jpg instead of the studio wall."

TOOLS `command_run` → `doc_place` → `command_run`
→ `render_preview`

KEY `layer.removeBackground`, then `doc_place fit`,
then `layer.arrange.sendToBack`.

- Verified end-to-end headless.

9 Remove an object

ASK CLAUDE

"Remove the trash can at about x 900-1050, y 600-800 in street.jpg."

TOOLS `command_batch`

KEY `select.rect` then `edit.contentAwareFill`
`output new`.

- Needs a pixel layer active (`layer.selectBottom`).
Verified.

10 Edit only the sky or subject

ASK CLAUDE

"Make the sky in lake.jpg deeper blue without touching the foreground."

TOOLS `command_batch`

KEY `select.sky` `threshold`, `softness` → any
`layer.newAdjustmentLayer.*`.

- Adjustment layers use the active selection as their mask.

Tone, color & LUTs

11 Quick fix: tone, denoise, sharpen

ASK CLAUDE

"Auto-fix the exposure and color of dim.jpg, reduce noise, then sharpen gently."

TOOLS `command_batch`

KEY `image.autoTone` `image.autoColor`
`filter.noise.reduceNoise`
`filter.sharpen.smartSharpen`

- On a Smart Object, filters stay editable (smart filters).

12 Non-destructive grade

ASK CLAUDE

"Add a gentle S-curve and a little vibrance as adjustment layers; save as PSD."

TOOLS `command_batch` → `doc_save`

KEY `layer.newAdjustmentLayer.curves` `points`
[[in,out], ...]; also ...levels, ...vibrance.

- 16 adjustment-layer kinds in all.

13 Apply a LUT (.cube / .3dl / .look)

ASK CLAUDE

"Apply FilmEmulation.cube as a Color Lookup layer to the three hero shots."

TOOLS `command_run`

KEY `layer.newAdjustmentLayer.colorLookup` `data` =
the .cube text, or `lut tealOrange`.

- file paths are refused over MCP; `data` works (verified).

14 Prepare for print (CMYK / ICC)**ASK CLAUDE**

"Convert cover.psd to coated CMYK, perceptual intent, and save a TIFF."

TOOLS `command_run` → `doc_export`

KEY `edit.convertToProfile` `profile` coated-cmyk, `intent`, `bpc`.

- Or `image.mode.cmyk` with the same options.

Layout, inspection & the live window**15 Crop to a social-media size****ASK CLAUDE**

"Make a 1080 x 1350 portrait crop of group.jpg centred on the people."

TOOLS `command_run` → `command_run`

KEY `select.subject` (gives `bounds`) → `image.crop` `targetWidth`, `targetHeight`.

- Pad instead with `image.canvasSize` anchor.

16 Add a caption, watermark or logo**ASK CLAUDE**

"Add a white 28 pt copyright line bottom-left with a soft shadow; place logo.png at 20%."

TOOLS `command_batch` → `doc_place`

KEY `type.create` `x`, `y`, `text`, `size`, `color`; `layer.layerStyle.dropShadow`.

- `doc_place` `scale` is a percentage. Verified (and used in storyboard 1).

17 Audit a PSD someone sent**ASK CLAUDE**

"Open client.psd, list its layer tree and anything that didn't import cleanly, show a preview."

TOOLS `doc_open` → `doc_inspect` → `render_preview`

KEY `doc_open` returns `warnings`; preview `max_side` ≤ 2048.

- CLI equivalent: `photocraft-cli info client.psd`.

18 Work side-by-side in the live app**ASK CLAUDE**

"(Live) Pick the Brush, fit the view, paint a tapered stroke across the top, then screenshot."

TOOLS `ui_set` → `ui_pointer` → `ui_screenshot`

KEY `ui_set` fields (`tool`, `fit`, `brushSize`); events with `pressure`.

- Menus: `ui_menu_invoke`; tree via `control_call` `ui.menu.list`.

✓ Always add: "render a preview"

`render_preview` returns a flattened PNG (`max_side` up to 2048) that Claude can actually see. Ask for one after each step; it costs image tokens, so `doc_inspect` (cheap JSON) is the better check when only layer structure matters.

04 The command map (909 ids behind command_run)

Every menu item is a dotted command id. Darker tiles are bigger families. Ask Claude to “list photocraft commands matching *word*” to open any tile; each entry carries a *params* doc with ranges, defaults and the result shape.

214 layer. new, delete, arrange, masks, styles, 16 adjustment layers, removeBackground, smart objects	124 filter. blur, sharpen, noise, distort, stylize, render, liquify, camera raw	79 edit. undo, fill, stroke, transform, content-aware fill and scale, convertToProfile	66 type. create, edit, font, size, color, warp, paragraph, OpenType	65 image. imageSize, canvasSize, crop, rotate, mode, autoTone, auto-Color, variables	54 file. new, open, save, export, place, automate, scripts (some re-fused over MCP)
35 select. rect (ellipse), lasso, mag-icWand, subject, sky, colorRange, modify	30 view. zoom, fit, guides, rulers, screen mode	26 paint. brush, eraser, gradient, bucket, clone, heal strokes	22 channel. new, duplicate, split, merge, calculations	21 path. pen, anchor edits, stroke, fill, selection from path	17 actions. load (base64), play, record, stop
14 brush. presets, size, hardness, dynamics, import	13 LayerComp. snapshots of layer visibility and position	12 gradient. presets, define, import; swatches. has 12 more	105 other document, pattern, tools, shape, history, count, slice, timeline		

05 Tool reference (all 25 MCP tools)

MCP tool	What it does
doc_new	New document (default 1920×1080 RGB 8-bit white): <i>width</i> , <i>height</i> , <i>mode</i> , <i>depth</i> , <i>background</i> .
doc_open	Open an image/PSD beneath the read root; returns index, size, import warnings.
doc_place	Place a file as an embedded Smart Object layer (<i>scale %</i> , <i>fit</i> , <i>center</i>).
doc_save	Save; .pcraft is native and lossless, other extensions export.
doc_export	Export to another format (as <i>doc_save</i> with an explicit path/format).
doc_export_layers	Layers to Files into <i>dir</i> ; never overwrites.
doc_import_data_sets	Import a variable-data CSV (<i>path</i> , <i>delimiter</i>).
doc_export_data_sets	One flattened file per data set (<i>dir</i> , <i>format</i> , <i>naming</i>).
doc_inspect	Layer tree, history, selection, active layer as JSON.

MCP tool	What it does (cont.)
<code>doc_select</code>	Make document <code>index</code> active.
<code>doc_close</code>	Close without saving.
<code>session_list</code>	Open documents (<code>index</code> , <code>name</code> , <code>size</code> , <code>dirty</code>) and the active index.
<code>command_list</code>	List engine commands (<code>filter</code> , <code>enabled_only</code>) with a <code>params</code> doc each.
<code>command_run</code>	Run one command: <code>id</code> , <code>params</code> ; <code>wait</code> <code>false</code> = background job.
<code>command_batch</code>	Run <code>steps</code> in order; stops on error unless <code>stop_on_error</code> <code>false</code> . Not atomic.
<code>jobs_list</code>	Running and recently finished background jobs, with progress.
<code>jobs_cancel</code>	Cancel one job (or all); the document is left unchanged.
<code>render_preview</code>	Bounded flattened PNG (<code>max_side</code> $\leq$ 2048); window screenshot in bridge mode.
<code>doc_render_preview</code>	Flattened PNG of a given document; bridge mode returns the window.
<code>ui_inspect</code>	<i>Live only.</i> Full UI state: tool, panels, views, dialogs, windows.
<code>ui_screenshot</code>	<i>Live only.</i> PNG of the app window (<code>max_side</code>).
<code>ui_set</code>	<i>Live only.</i> Change tool, panels, zoom, theme, brush settings via <code>fields</code> .
<code>ui_pointer</code>	<i>Live only.</i> Pointer events in document coordinates (pressure, tilt).
<code>ui_menu_invoke</code>	<i>Live only.</i> Activate a menu item by id.
<code>control_call</code>	<i>Live only.</i> Any control-protocol method, e.g. <code>ui.menu.list</code> .

06 Example tool calls

Storyboard 1: the glow layer in one `command_batch`

```
{
  "steps": [
    {
      "id": "layer.new.layer",
      "params": {
        "name": "Glow"
      }
    },
    {
      "id": "select.rect",
      "params": {
        "x": 800,
        "y": 150,
        "width": 360,
        "height": 360,
        "ellipse": true
      }
    },
    {
      "id": "edit.fill",
      "params": {
        "color": "#ffd166"
      }
    },
    {
      "id": "select.deselect"
    },
    {
      "id": "filter.blur.gaussianBlur",
      "params": {
        "radius": 45
      }
    },
    {
      "id": "layer.arrange.sendToBack"
    },
    {
      "id": "layer.arrange.bringForward"
    }
  ]
}
```

One file of a batch resize (Claude repeats this per file)

```
{
  "tool": "doc_open",
  "arguments": {
    "path": "Pictures/shoot/IMG_0412.jpg"
  }
},
{
  "tool": "command_run",
  "arguments": {
    "id": "image.imageSize",
    "params": {
      "width": 1600,
      "resample": "bicubicSharper"
    }
  }
},
{
  "tool": "doc_save",
  "arguments": {
    "path": "Pictures/web/IMG_0412.jpg",
    "quality": 85
  }
},
{
  "tool": "doc_close",
  "arguments": {}
}
```

A one-shot grade with `command_batch`

```
{
  "steps": [
    {
      "id": "image.autoTone",
    },
    {
      "id": "layer.newAdjustmentLayer.curves",
      "params": {
        "points": [[0,0],[64,48],[192,212],[255,255]]
      }
    },
    {
      "id": "layer.newAdjustmentLayer.colorLookup",
      "params": {
        "lut": "tealOrange"
      }
    },
    {
      "id": "filter.sharpen.smartSharpen",
      "params": {
        "amount": 80,
        "radius": 1
      }
    }
  ]
}
```

07 Limits & tips

! Early alpha

The README is blunt: *“not yet a Photoshop replacement for daily professional work.”* Biggest gaps: AI/generative features, 15 missing tools, typography depth, plug-in compatibility. Every menu item is wired to a command, but “that measures wiring, not behaviour.”

! Work on copies of PSDs

Re-saving renders the same for 307 of 309 psd-tools test files, but the file is not byte-identical, and PhotoCraft-only features (e.g. some smart filters) are dropped with a warning. Use `.pcraft` for lossless intermediates and read the `warnings` every save returns.

! Commands refused over MCP (ambient file paths)

`file.scripts.imageProcessor`, `file.automate.batch`, `file.automate.photomerge`, `file.placeEmbedded`, `layer.exportAs`, `file.export.quickExport`, `actions.load` with `path`, `layer.newAdjustmentLayer.colorLookup` with `file` (all tested). Use the scoped MCP tools (`doc_place`, `doc_export_layers`, ...), pass file contents as `data`, or let Claude run `photocraft-cli` (`run`, `batch`, `convert`, `droplet`) in Bash.

i File roots

Each server only reads and writes beneath its automation roots (`--automation-read-root`, `--automation-write-root`, normally your home folder). Paths may be relative to the root (`Pictures/a.jpg`) or absolute beneath it; copy files from other volumes in first. `doc_save` does not create folders; have Claude `mkdir` them.

✓ Working habits

- Ask Claude to call `command_list` with a `filter` before exotic edits: each command's `params` doc gives ranges, defaults and the result shape (that is how `select.rect`'s `ellipse` flag turned up in storyboard 1).
- Each command is one undo step; `command_batch` does not roll back on failure. `history.newSnapshot` and `history.selectState` are available.
- Layer ids come back from the command that created the layer, or from `doc_inspect`; `layer.selectTop` / `layer.selectBottom` when you do not care which.
- Long filters: `wait` false, then `jobs_list` / `jobs_cancel`.
- The Live port is loopback-only and token-authenticated; keep the token file private.