crane Registry swiss knife

CONCEPTS

> crane talks straight to registries with no daemon: copy keeps the digest and moves only missing layers, so it beats pull, tag and push.

Registry tags, manifests, blobs
↓ pull / export
crane no daemon, no root
↓ push / append
Local tarball or OCI layout directory

Registry commands read and write remote images; only pull, push and export touch local files

Image reference forms

Form	Meaning
`<repo>:<tag>`	Tag: can move
`<repo>@sha256:<hex>`	Digest: immutable
`<registry>/<repo>`	Host-qualified; bare names mean Docker Hub
-	Stdin or stdout in export

> Global flags such as --platform and --insecure apply to every command. Without --platform, commands act on all platforms of an index.

INSPECT

`crane ls <repo>`
List tags in a repository

`crane digest <image>`
Print the image digest

`crane manifest <image>`
Print the manifest (or index)

`crane config <image>`
Print the image config JSON

`crane catalog <registry>`
List repositories in a registry

`crane ls --omit-digest-tags <repo>`
Hide sha256-... tags

`crane digest --full-ref <image>`
Print repo@digest

`crane manifest --platform <os/arch> <image>`
Manifest of one platform

COPY & TAG

`crane copy <src> <dst>`
Copy an image, keeping its digest

`crane copy --all-tags <src-repo> <dst-repo>`
Copy every tag of a repo

`crane copy --no-clobber <src> <dst>`
Skip tags that already exist

`crane tag <image> <new-tag>`
Add a tag without re-uploading

`crane delete <image>`
Delete an image reference

`crane copy --platform <os/arch> <src> <dst>`
Copy a single platform

PULL, PUSH & EXPORT

`crane pull <image> <file.tar>`
Save an image as a tarball

`crane pull --format oci <image> <dir>`
Save as an OCI layout directory

`crane push <file.tar> <image>`
Push a tarball to a registry

`crane push <oci-dir> <image>`
Push an OCI layout directory

`crane export <image> <file.tar>`
Write filesystem to a tarball

`crane export <image> - | tar -tf -`
Stream the filesystem to tar

`crane export - <file.tar>`
Read the image from stdin

MODIFY IMAGES

`crane append -b <base> -f <layer.tar> -t <image>`
Add a layer on a base and push

`crane mutate <image> -t <new-tag> --entrypoint <cmd>`
Change the entrypoint

`crane mutate <image> -t <new-tag> -l <key>=<value>`
Add a label

`crane mutate <image> -t <new-tag> -e <KEY>=<value>`
Add an environment variable

MODIFY IMAGES (cont.)

`crane flatten <image> -t <new-tag>`
Squash all layers into one

`crane rebase <image> --old_base <old> --new_base <new> -t <tag>`
Move an image onto a new base

AUTH & TLS

`crane auth login <registry> -u <user> -p <password>`
Log in and store credentials

`crane auth login <registry> -u <user> --password-stdin`
Read the password from stdin

`crane auth logout <registry>`
Remove stored credentials

`crane --insecure <command> <ref>`
Allow plain HTTP without TLS

`crane --verbose <command> <ref>`
Debug logs for failing requests

`crane --platform <os/arch[/variant]> <command> <ref>`
Pick a platform

RECIPES

Retag a CI image as latest

```
crane auth login
registry.example.com -u ci --
password-stdin <<< "$TOKEN"
crane tag registry.example.com/
app:$SHA latest
crane digest registry.example.com/
app:latest
```

Mirror a repo to another registry

```
crane copy --all-tags \
  docker.io/library/alpine \
  registry.example.com/mirror/
alpine
```

Layer a file onto a base image

```
tar -cf app.tar app/
crane append -b alpine -f app.tar
\
  -t registry.example.com/app:v1
```

Inspect an image without Docker

```
crane manifest alpine | jq .
crane config alpine | jq .config
crane export alpine - | tar -tf -
| head
```