crane Registry swiss knife

START HERE

`crane` **`ls`** *`<repo>`*
List a repo's tags

`crane` **`digest`** *`<image>`*
Resolve a tag to a digest

`crane` **`manifest`** *`<image>`*
Print the manifest

`crane` **`copy`** *`<src>`* *`<dst>`*
Registry to registry

`crane` **`pull`** *`<image>`* *`<file.tar>`*
Save to a tarball

`crane` **`push`** *`<file.tar>`* *`<image>`*
Upload a tarball

`crane` **`export`** *`<image>`* **`-`**
Filesystem as a tar stream

`crane` **`auth login`** *`<registry>`*
Store credentials

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

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.

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

INSPECT (cont.)

`crane` **`manifest`** *`<image>`*
Print the manifest (or index)
For a multi-arch image this is the index listing every platform; add --platform to get one image's manifest.

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

`crane` **`catalog`** *`<registry>`*
List repositories in a registry
Many hosted registries disable or restrict the catalog endpoint.

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

`crane` **`ls --full-ref`** *`<repo>`*
Print full references, not bare tags

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

`crane` **`digest --tarball`** *`<file.tar>`* *`<image>`*
Digest of an image in a tarball

`crane` **`blob`** *`<repo>`***`@sha256:`***`<hex>`*
Print a layer or config blob

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

`crane` **`validate --remote`** *`<image>`*
Check an image is well-formed

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
Needs a registry that allows deletes; deleting a digest removes every tag pointing at it.

`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

PULL, PUSH & EXPORT (cont.)

`crane` **`push`** *`<oci-dir>`* *`<image>`*
Push an OCI layout directory
A directory is read as an OCI layout, anything else as a docker-style tarball.

`crane` **`push --index`** *`<dir>`* *`<image>`*
Push several images as one index

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

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

MODIFY IMAGES

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

`crane` **`append -f`** *`<layer.tar>`* **`-o`** *`<file.tar>`*
Save the result locally instead

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

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

MODIFY IMAGES (cont.)

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

`crane` **`mutate`** *`<image>`* **`-t`** *`<new-tag>`* **`--cmd`** *`<args>`*
Set a new CMD

`crane` **`flatten`** *`<image>`* **`-t`** *`<new-tag>`*
Squash all layers into one
Without -t the result is pushed by digest to the original repository.

`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
Prefer this over -p, which leaves the password in shell history.

`crane` **`auth logout`** *`<registry>`*
Remove stored credentials

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

AUTH & TLS (cont.)

`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
```