# OCI Image Support Comes to Open Source Docker Registry

Thursday, October 11, 2018 by Open Container Initiative

By Phil Estes & Mike Brown

The Open Container Initiative (OCI) was formed in 2015 as a place to collaborate on the definition of a standard container runtime and image format. By that time, Docker had effectively become the de facto standard for container images, and DockerHub and many other public and private registries were filled with tens of thousands of available container images using the Docker image format.

Given this state of the world in late 2015, the OCI image specification work began in earnest with a strong group of collaborating independent and vendor-associated participants, using the Docker v2.2 image format as a starting point. Fast forward eighteen months, and by summer of 2017, both the runtime and image specifications reached their intended 1.0 milestone release.

That release was a great milestone for the OCI community and container ecosystem at large, but the next step beyond declaring victory on any specification is always: adoption! The runtime specification had a head start here. The default

reference implementation within the OCI, runc, was already in use by several implementers including the Docker engine by the time the specification was released. For the image specification, tools, and most notably, container image registries would have to adopt the OCI v1.0 image specification specifically for it to be useful to developers and implementers. Many tools existed to operate on Docker's v2.2 image manifest format already, and these tools would now need to adopt the OCI format.

This work—enabling OCI images in the core registry used on a daily basis for millions of images—started in 2016 on the Docker distribution project long before the specification even reached 1.0. This open source project is probably better known as the registry that backs DockerHub as well as many other public and private registries. Now that this pull request has been merged, we'll take a few minutes to describe a bit more about the OCI image spec and how the open source registry was modified to support OCI in addition to its native Docker image formats.

## Background

Container registries today are usually the combination of an HTTP API backed by some form of content store, with the content itself delineated by various media types. In the Docker image specification, you have metadata types (like the image's Docker runtime configuration) as JSON content, combined with references to image layers, usually tarred and compressed binary blobs which are then stored in the backing filesystem along with their requisite media types. For example, in the Docker v2.2 image world, each container image will have a manifest with the media type application/vnd.docker.distribution.manifest.v2+json. A list of image references (to support multi-platform images) is known as a manifest list with media type:

application/vnd.docker.distribution.manifest.list.v2+json. An image layer commonly has the media type of application/vnd.docker.image.rootfs.diff.tar.gzip.

The OCI specification took this v2.2 image specification from Docker as a starting point, but as definitions were changed in small ways during the specification process, each one of the Docker image metadata or layer media types developed into an OCI counterpart. Given these efforts, the resulting official OCI media types include application/vnd.oci.image.manifest.v1+json for the image manifest. Manifest lists have been renamed to “indexes” in OCI v1, giving us the media type: application/vnd.oci.image.index.v1+json. A layer type becomes application/vnd.oci.image.layer.v1.tar+gzip in OCI parlance, and so on.

## Implementation

To add OCI v1 support into the open source distribution project meant handling all the new media types from the OCI specification, and appropriately handling the HTTP API interactions when a client of the registry wants to “speak” in OCI media-types versus the already supported Docker types. If you think of image manifests being the top-most objects in the registry’s world, then effectively the registry was currently supporting three: the original “schema 1” Docker image format; the “schema 2” single image Docker format, and the manifest list multi-platform Docker image format (part of the schema 2.2 definition, but treated as a separate object type). Adding the OCI v1 types would mean supporting OCI v1 manifests and indexes—two additional formats, which would cascade into all the OCI media type references from these high level manifests and indexes.

The first attempt looked at simply expanding the registry's handlers for the schema 2 and manifest list types to support OCI v1 manifests and OCI v1 indexes. These two types are quite similar between Docker v2.2 and OCI v1, so it seemed like it might be a quick path to getting support for OCI into the registry codebase. However, over time it became clear that this would not be an optimal path for long-term OCI image support.

## Success

Finally, the approach presented in GitHub pull request #2076 –to add specific new handlers to the open source registry for the OCI main image types (v1 manifests and v1 indexes)— was agreed on and in July of 2018 this PR was merged into the docker/distribution project. This PR adds new handlers for the OCI types as well as new tests, validation code, and integration that crosses various core sections of the registry codebase. As with most PRs that add significant capability, it was not an easy task, and Mike Brown from IBM persisted on the project for over a year before working out all the requested issues, reviews, and bugs found during testing and validation of the OCI support!

## Summary

Interested parties can try out the recently available v2.7.0 release candidate with OCI v1 image support. We expect after final release that registries based on the open source distribution, including DockerHub, will update in the near future to adopt these new features. At that point, client tools which today already support the OCI image formats will have interoperability with these registries thanks to the hard work

of many involved from Docker, to the OCI, to participating vendors and contributors.

In addition to the growing adoption of the OCI v1 image formats—used natively in other projects like the CNCF containerd project as well as the Moby project's LinuxKit implementation—we're also excited about the standardization of the registry API itself coming to OCI this year. The proposal to take the Docker HTTP registry API has already been accepted into the OCI. This specification will add industry standardization around the protocol to talk to registries in addition to the existing interoperability brought in with the OCI v1 image specification.

It's great to see growing adoption of the OCI specifications, and with the added support in the open source Docker registry for OCI images, we see this as just the beginning of a whole host of tools, vendor products, and software that will be enabled to utilize the OCI specifications now and in the future.