This is the open-source Docker registry implementation. If you have

issues with the closed-source [hub][], please report them to

`support-index@docker.com` instead of here.

# Docker-Registry for developers

## Bare minimum

System-wide, you need `git` and `pip` installed. You also need `tox` (usually just `pip install tox`).

You optionally need `nose`, `coverage` and `flake8` (usually just `pip install nose coverage flake8`) .

## Architecture

### docker-registry-core

The code for this lives in the main repository, under the folder depends/docker-registry-core.

It provides:

\* exceptions

\* a driver interface describing how the registry interacts with the data storage

\* a compatibility layer (e.g.: abstracting behavior differences between python versions)

\* common useful utility code

\* filesystem storage driver

It doesn't depend on anything, and is published as a standalone pip package. Its version closely reflects the driver interface versioning.

If you are going to change the way data is stored, this is where you should look.

If you are going to hack on the registry itself, this is probably not interesting for you.

If you are going to create a new storage driver, see below.

### docker-registry

This is what lives in the main repository, and contains the bulk of the server code (including the s3 storage driver).

It depends on `docker-registry-core`.

This is likely where you are going to hack if you want to modify the registry behavior.

It is published as a pip package, although the recommended way to use is to use the official docker image.

### docker-registry-driver-X

Storage drivers (like elliptics) are implemented as independent pip packages.

Said packages depend on `docker-registry-core` only (save their own dependencies).

## Namespaces

We use python namespaces.

`docker-registry-core` uses:

\* `docker\_registry`

\* `docker\_registry.core`

\* `docker\_registry.testing`

\* `docker\_registry.drivers`

`docker-registry` uses:

\* `docker-registry`

\* others

Drivers must stay inside `docker\_registry.drivers`.

## Tooling

We use `nose`, `coverage`, `hacking` (for `flake`), `tox` and `travis`.

Wherever you are coding (registry, core, or driver), your friends are thus:

\* run the tests: `python setup.py nosetests`

\* check your style: `flake8`

\* run the tests on all platform: `tox`

If any of these three fail, then your PR will get rejected :-)

If the travis build fails, your PR will need to be updated.

## Acceptance platforms

Any new code must run with python2.7.

Any new code should better run with python2.6 and python3.4.

Existing code shouldn't regress.

## Storage driver developer howto

Have a look at the [elliptics driver](https://github.com/noxiouz/docker-registry-driver-elliptics) and copy its structure.

Explore the files.

Pretty much:

\* you have to use namespaces

\* you have to implement methods from the base.Driver interface

\* you have to use the provided test suite

\* all the tests must pass

### Word of warning

Drivers are expected to receive bytes and to return bytes.

Don't try to decode or encode content.

## Development environment notes

We don't currently run any tests for python3, as we are stuck on gevent not being py3 ready.

On OSX, in order for the dependencies to compile properly (inside tox venv) you might need to have extra include and lib specified. Environment variables are provided for that, namely $TOX\_INCLUDE and $TOX\_LIB.

[hub]: https://registry.hub.docker.com/