Docker and Python

3 juillet 2017, RMLL St-Etienne, Michael Bright
Running Python apps under Docker
The Docker API
Docker-machine from Python
docker-py: Controlling Docker from Python
Other Modules
Ansible Container
Of course we can run any app under Docker - Python or not.

This can be a very useful alternative to virtualenv

- complete filesystem isolation allowing to have complex dependancies beyond Python itself
- launch multiple instances of a Python process in isolation
  - isolated filesystem, process space, ports
- pull existing images of applications, frameworks
- build upon existing images and push the results to share
- can compose complex applications from multiple containers
What "standard" images exist on the Docker Hub? hub.docker.com

Python: "python"
  ○ small, large base images
  ○ based on Debian, Alpine (& even WindowsServerCore !)
  ○ Python 2.x [2.7.13], 3.x [3.3.6, 3.4.6, 3.5.3, 3.6.1, 3.6.2rc1]

Frameworks:
  ○ Flask
  ○ Cherrypy
  ○ Django
  ○ ...

@mjbright
Let's take a look at how the docker command-line client makes use of the docker API to perform requests

Can we make our own command-line requests using curl?
There are many docker-related modules available in Python.

Let's first look at "docker-machine" which provides a simple-* encapsulation around the docker-machine executable.

* - a bit too simple: if docker-machine issues a prompt you will not be informed (problem with Azure login).

Nevertheless it provides a nice wrapper enabling to simply fire up Docker hosts and pass their config to the docker-py client.
Installing the module:
From local repo (modified to work with docker-py v2)
git clone https://github.com/mjbright/python-docker-machine
python setup.py install

Initializing the module:
import machine
import docker
m = machine.Machine(path="/usr/local/bin/docker-machine")
Using the module:

```python
m.create('test-machine', driver='virtualbox', blocking=True)
m.ls()
client = docker.DockerClient(**m.config(machine='test-machine'))
m.rm(machine='test-machine')
```
This module, github.com/docker/docker-py is maintained by the Docker team.

NOTE: Now called "docker", use "pip install docker", not docker-py. Currently at the 2.4.2 release it has 2 apis

- a low-level 'docker.APIClient' api
  - with operations corresponding to the Docker api
    - e.g. "create_container", "inspect_container", ...
- a higher-level 'docker.Client' object-oriented api
  - objects similar to new (> 1.13) "docker" client commands
    - e.g. "docker.containers.create", "docker.images.list"
There are many docker related Python modules, "pip3 search docker" gives 443 entries of which 217 have docker in the title, including:

**build tools**
- e.g. docker-build-tool, whalelinter, grocker

**alternative API clients**
- e.g. aiodockerpy, docker-map, python-dockercloud

**Docker based utilities**
- e.g. docker-cleaner, docker-scripts, dockeranalyser

**Monitoring tools**
- e.g. check_docker, dockermon

**CI tools**
- e.g. travis2docker,

@mjbright
Provides a useful abstraction for building and running containers using the Ansible Configuration Management tool.

Image from: https://tech.napsty.com
Install: `pip install ansible-container`

Initialize a new project: generates yaml template files

`ansible-container init`

Modify yaml files as needed for your project, then build:

`ansible-container --debug build`

Run:

`ansible-container run`

When ready upload container images to your configured registry:

`ansible-container shipit`

@mjbright
<table>
<thead>
<tr>
<th>Resource</th>
<th>Docs</th>
<th>Github</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running Python under Docker</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Docker-py</td>
<td>docker-py.readthedocs.io</td>
<td>docker/docker-py</td>
</tr>
<tr>
<td>Docker Maps</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ansible Container</td>
<td>ansible.com/ansible-container</td>
<td>ansible/ansible-container</td>
</tr>
</tbody>
</table>

@mjbright