Docker Containers

Marko Ambrož,
Žiga Hudolin
Ministrstvo za javno upravo
DIREKTORAT ZA INFORMATIKO
“Vendor lock in” with DOCKER???

http://blog.dennybritz.com/2015/10/01/a-brief-guide-to-the-docker-ecosystem/
## Supported operating systems

<table>
<thead>
<tr>
<th>General purpose operating systems:</th>
<th>Container optimized operating systems:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon EC2</td>
<td>RedHat Project Atomic</td>
</tr>
<tr>
<td>Arch Linux</td>
<td>Boot2Docker</td>
</tr>
<tr>
<td>Azure Microsoft</td>
<td>CoreOS</td>
</tr>
<tr>
<td>CentOS</td>
<td>OSv</td>
</tr>
<tr>
<td>CRUX Linux</td>
<td>VMWare PhotonOS</td>
</tr>
<tr>
<td>Debian</td>
<td>RancherOS</td>
</tr>
<tr>
<td>Fedora</td>
<td>Ubuntu Core 7+</td>
</tr>
<tr>
<td>FrugalWare</td>
<td></td>
</tr>
<tr>
<td>Gentoo</td>
<td></td>
</tr>
<tr>
<td>Google Cloud</td>
<td></td>
</tr>
<tr>
<td>Joyent</td>
<td></td>
</tr>
<tr>
<td>Mac OS X</td>
<td></td>
</tr>
<tr>
<td>Oracle Linux</td>
<td></td>
</tr>
<tr>
<td>Rackspace Cloud</td>
<td></td>
</tr>
<tr>
<td>Red Hat Enterprise Linux</td>
<td></td>
</tr>
<tr>
<td>SoftLayer IBM</td>
<td></td>
</tr>
<tr>
<td>openSUSE and SUSE Linux</td>
<td></td>
</tr>
<tr>
<td>Ubuntu</td>
<td></td>
</tr>
<tr>
<td>Windows</td>
<td></td>
</tr>
<tr>
<td>20+</td>
<td></td>
</tr>
</tbody>
</table>
FROM ubuntu
RUN apt-get update
RUN apt-get -y install git build-essential
RUN apt-get -y install nodejs npm
RUN apt-get install -y git-core zip
RUN apt-get clean
WORKDIR /opt/coder/
RUN git clone github:googlecoder.git .
RUN ls -la && pwd
WORKDIR /opt/coder/coder-apps
RUN ./install_common.sh ../coder-base
WORKDIR /opt/coder/coder-base
RUN npm install
COPY src/config.js /opt/coder/coder-base/

#EXPOSE 80 443
EXPOSE 8180 8181
CMD ["nodejs", "server.js"]
Are Docker Containers cloud neutral?

**IaaS** (Infrastructure as a Service)
e.g.: DigitalOcean droplets: Ubuntu, Fedora, Debian, CoreOS, Centos

**CaaS** (Container as a Service)
e.g.: RackSpace: CARINA

```bash
$ eval "$( carina env mycluster )"
$ echo $DOCKER_HOST
tcp://104.130.0.121:2376
$ docker run nginx
```
If the demo fails you can swim with:

GoogleCoder in a Container at DigitalOcean:

https://188.166.34.228:7443

Password: SInog32016
Demonstration
Example 1

- Goal: Create a container running as a daemon and a command that print hello world on screen every second
- In this example we will use
  - `docker run; runs the container`
  - `-d; flag runs the container in the background`
  - `ubuntu; is the image you would like to run`
  - `docker logs; looks inside the container`
- Finally we specify a command to run:
  - `/bin/sh -c "while true; do echo hello world; sleep 1; done"`
Example 1: docker run

- docker run -d ubuntu /bin/sh -c "while true; do echo hello world; sleep 1; done"

  This long string “72b14b95f96864779a06a2b...” is called a container ID

  docker ps command queries the Docker daemon for information about all the containers.
Example 1: docker logs

- To see what is going on inside the container we use the `docker logs` command.

```
[root@localhost ziggy]# docker logs elated_nobel
hello world
hello world
hello world
hello world
hello world
hello world
hello world
hello world
```

- The `docker stop` command tells Docker to politely stop the running container and returns the name of the container it stopped in our case “elated_nobel”.
Example 2

- Goal: Create a container running as a deamon and a command that print Hello World every second and write it in a specific file
- In this example we will use
  - `docker run;` runs the container
  - `-d;` flag runs the container in the background
  - `ubuntu;` is the image you would like to run
  - `docker exec;` runs a new bash session in a running container
- Finally we specify a command to run an write result in a file
Example 2: docker run

- docker run -d ubuntu /bin/sh -c "while true; do echo Hello World >> /root/test.txt; sleep 1; done"

- This long string “07efe03bbb09a676cb6fba73...” is called a container ID
- docker ps command queries the Docker daemon for information about all the containers.
Example 2: docker exec

- With command `docker exec` we run a new bash session in a running container and we can check the file
  - `-i` -- interactive; keep STDIN open even if not attached
    - `--privileged; Give extended Linux capabilities to the command`
  - `-t` -- `tty`; Allocate a pseudo-TTY
Example 2: export TERM

- We set TERM environment variable in running container
Example 3: docker commit

- **Goal:** update a container created from an image with a specific file and commit the result to an image
- **In this example we will use**
  - `docker commit; create a new image from a container's changes`
Example 4

- **Goal:** Install application Google Coder
- **In this example we will use**
  - docker pull; pull an image or a repository from the registry
  - docker images; list images
- **Finally we check our application in browser**
We will use image from ambrom/ambro-gc repository on dockerhub
Example 4: docker run

- With command docker run we will run our application but we have to specify ports “8180 and 8181”

- We will open the browser and check if application is alive and responsible
Primer 4: test aplikacije

- https://localhost:8181

Getting Started

Check out the quick tips above. For a more detailed tour, go to room!coder. When you're ready to dive in, just click 'Got It.'