

Leverage Docker and Makefile for consistent testing environment

Docker Miami - 03/31/2015
@charme_g

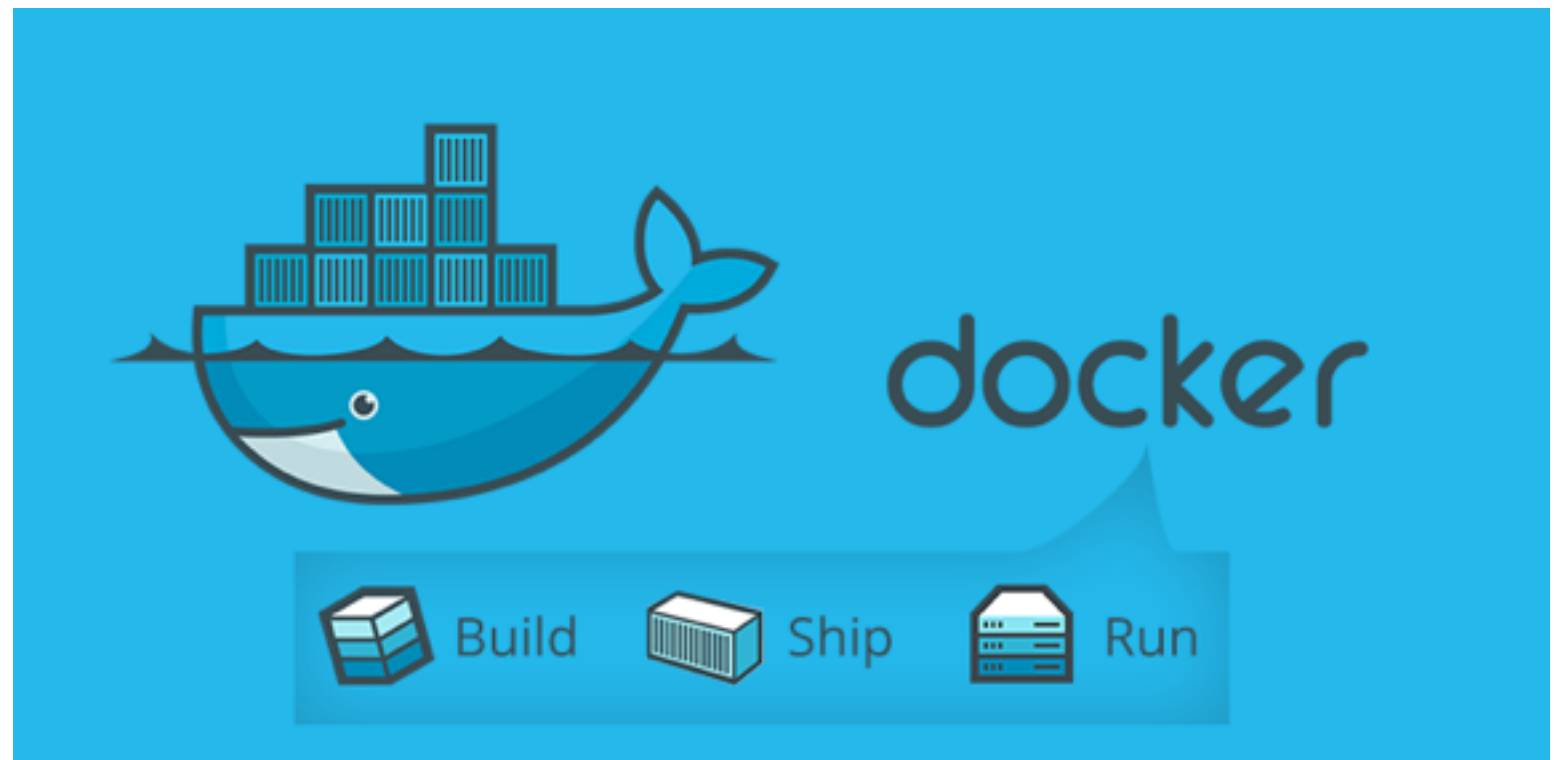
Docker 2nd Birthday Party

About me

- Guillaume J. Charmes
- @charme_g
- github.com/creack
- Lead developer on Docker from 0 to 1.0

Docker

- What is Docker?
- Registry
- Use cases
- Dockerfile



Makefile

- How it works?
- Example
- Add Docker into the mix
 - links

The diagram shows a Makefile with two main sections: 'definition part' and 'creation rules'. The 'definition part' defines variables for the compiler (CC), linker (LNK), and converter (CNV). The 'creation rules' define the targets and their dependencies. Red arrows point from labels to specific parts of the Makefile.

```
definition part {
CC = g++
LNK = -lLightflow
CNV = convert
}

creation rules {
all:      ball.jpg
<<Tab>>
ball.jpg: ball.tga
<<Tab>> $(CNV) ball.tga ball.jpg

ball.tga: simplescene
<<Tab>> simplescene

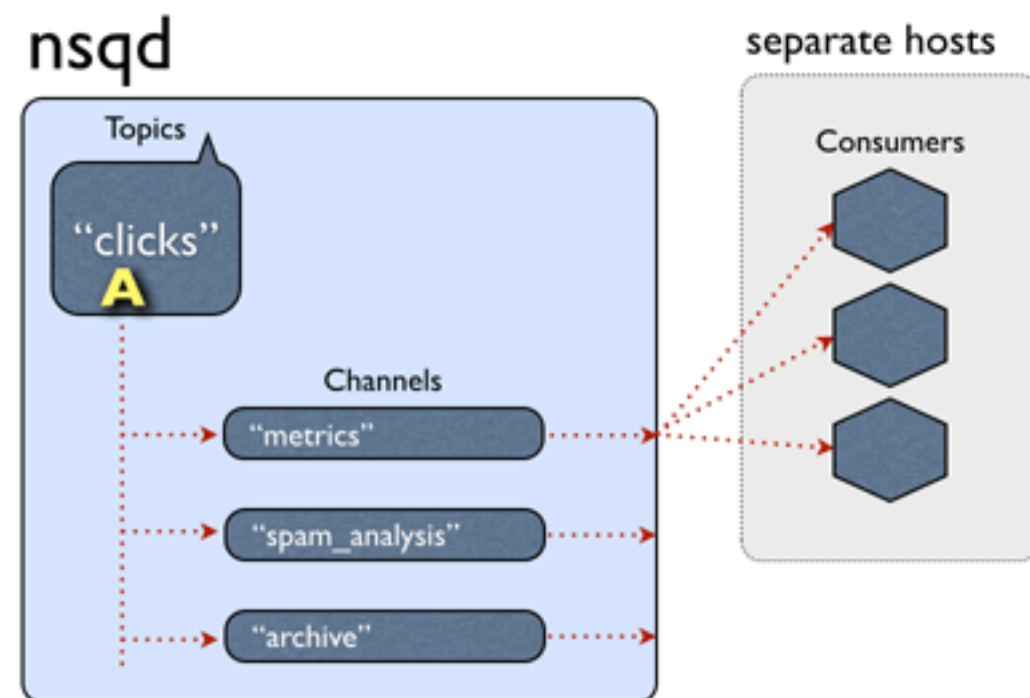
simplescene: main.cpp
<<Tab>> $(CC) $(LNK) main.cpp -o simplescene
}
```

Annotations:

- definition part**: A bracket on the left side of the first three lines.
- creation rules**: A bracket on the left side of the last five lines.
- compiler specification**: Points to `CC = g++`.
- linker flags**: Points to `LNK = -lLightflow`.
- target**: Points to `simplescene` in the rule `ball.tga: simplescene`.
- dependencies**: Points to `simplescene` in the rule `simplescene: main.cpp`.
- creation rule**: Points to the command `$(CC) $(LNK) main.cpp -o simplescene`.

Multi-component Service

- NSQ
 - nsqd
 - nsqlookupd
 - nsqadmin
- Docker
- Makefile + Docker



```
1 all      :      nsq
2
3 nsq       :      nsqd
4
5 nsqlookupd:
6     docker run -d --name nsqlookupd_c nsqio/nsq /nsqlookupd
7
8 nsqd      :      nsqlookupd
9     docker run -d --name nsqd_c --link nsqlookupd_c:lookupd \
10         nsqio/nsq bash -c "/nsqd
11         --lookupd-tcp-address=$$LOOKUPD_PORT_4160_TCP_ADDR"
12
13
14 nsqadmin:      nsqd
15     docker run -d --name nsqadmin_c -p 4171:4171 --link nsqlookupd_c:lookupd nsqio/nsqadmin \
16     nsqio/nsq bash -c "/nsqadmin
17     --lookup-tcp-address=$$LOOKUPD_PORT_4160_TCP_ADDR"
```

Use it for integration test

- Isolate integration test
 - tags
 - subdir
- Write the Makefile

```
1 NAME           =      sql_example
2 SQL_IMAGE      =      mysql
3 DOCKER_IMAGE   =      creack/sql_example
4
5
6 all            :      build
7
8 .build         :      .
9                 docker build -t $(NAME) .
10                 docker inspect -f '{{.Id}}' $(NAME) > .build
11
12 build          :      .build
13
14 mysql          :
15                 docker run -d --name $(SQL_IMAGE)_c $(SQL_IMAGE)
16                 sleep 2
17
18 test           :      build mysql
19                 docker run --link "$(SQL_IMAGE)_c:mysql" $(NAME) \
20                 bash -c "MYSQL_ADDR=$$MYSQL_PORT_3306_TCP_ADDR go test -v"
21
22 clean          :
23                 docker rm -f $(SQL_IMAGE)
```

Real life example

- github.com/gofinance/ib
 - Java server
 - Weird/complex setup
 - Docker

Go further

- Use Makefiles and Docker to release code

```
1 NAME           = example
2 DOCKER_IMAGE    = 127.0.0.1:5000/$(NAME)
3
4 all             : build
5
6 .build          : .
7                 docker build -t $(NAME) .
8                 docker inspect -f '{{.Id}}' $(NAME) > .build
9
10 build           : .build
11
12 release/$(NAME) : build
13                 docker run --rm --entrypoint /bin/sh $(NAME) -c 'tar cf - /$(NAME) /etc/ssl' > $@ || (rm -f $@; false)
14                 docker build --rm -t $(DOCKER_IMAGE) release
15
16 release         : release/$(NAME)
17
18 push            : release
19                 docker push $(DOCKER_IMAGE)
20
21 clean           :
22                 $(RM) .build release/$(NAME)
23
24 .PHONY          : push release build all clean
```