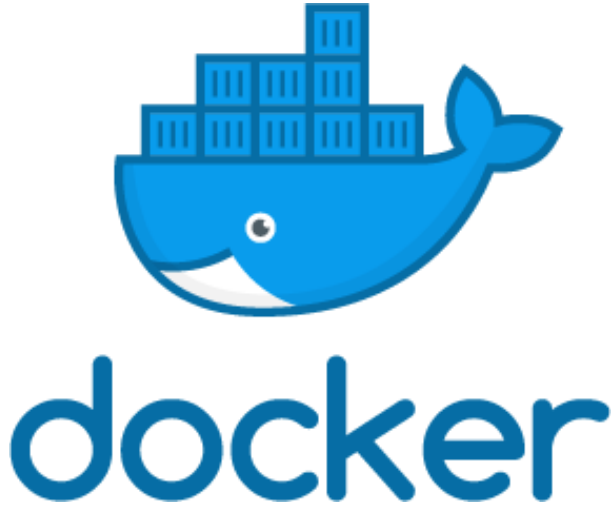
MEETUP TECHNOLOGIES WEB & LOGICIELS LIBRES À MONTRÉAL

MONTRÉAL, MERCREDI 19 JUILLET 2017

CRÉER SON ENVIRONNEMENT DE DEV PERSONNEL AVEC DOCKER



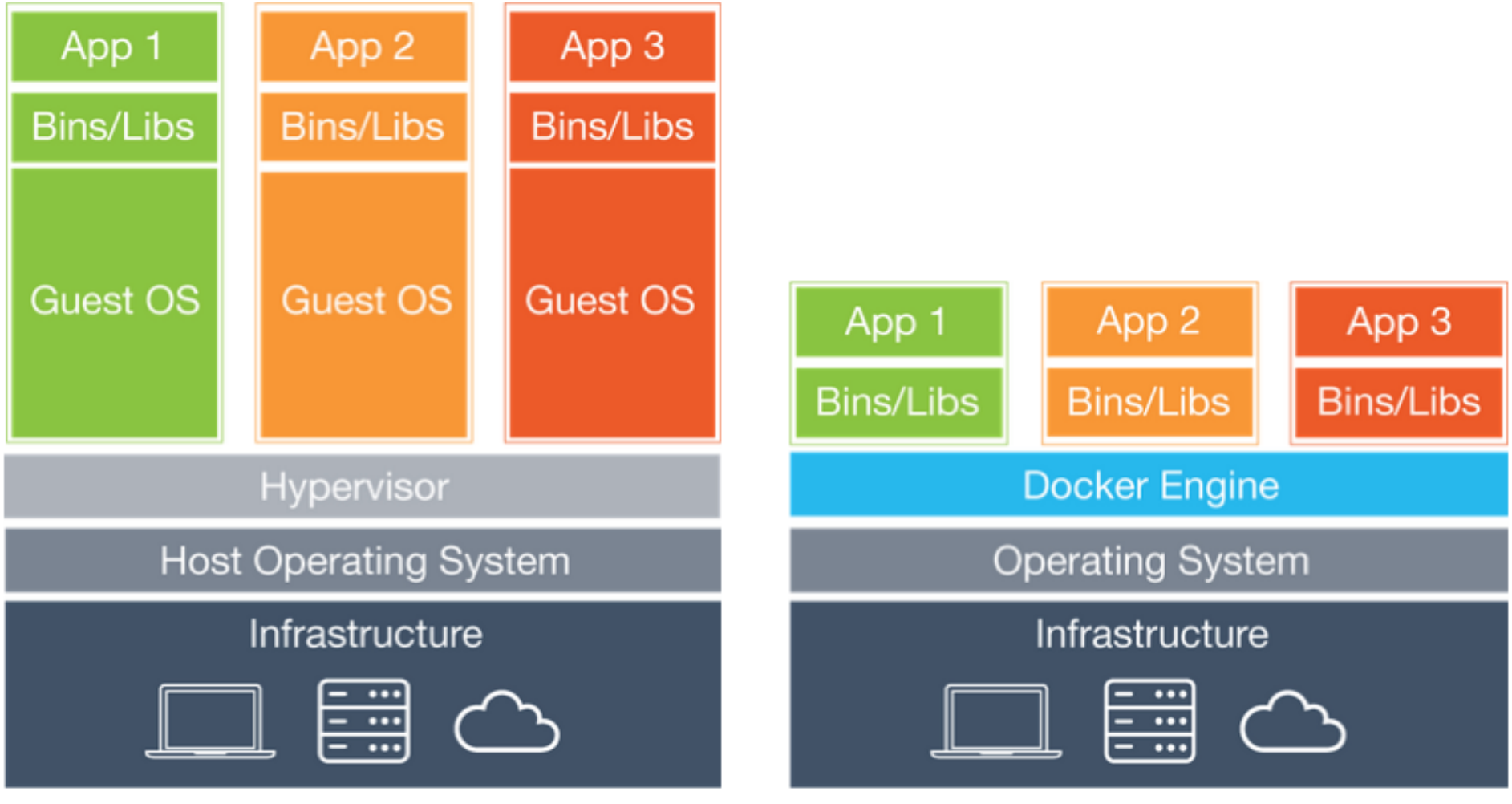
SOMMAIRE

- Qu'est-ce que Docker et comment l'utiliser ?
- Créer son environnement de dev
- Démonstration

QU'EST-CE QUE DOCKER?

- C'est un outil pour créer des conteneurs logiciels simplement
- Permet d'encapsuler une application et toutes ses dépendances
- Quelle est la différence entre un conteneur et une VM?

CONTENEUR VS VM



POURQUOI UTILISER DOCKER POUR SON ENVIRONNEMENT DE DEV?

- Légèreté
- Portabilité
- Reproduction

CONCEPTS IMPORTANTS

DOCKERFILE

Contient les instructions pour créer une image

```
FROM debian:jessie

RUN apt-get update && \
    apt-get install -y python-pip

ADD /dashboard /opt/dashboard
RUN pip install -r /opt/dashboard/requirements.txt

EXPOSE 5000

WORKDIR /opt/dashboard
CMD python dashboard.py
```

IMAGE

Une image est un paquet léger, autonome et exécutable qui comprend tout ce qui est nécessaire pour exécuter un logiciel, y compris le code, un temps d'exécution, des bibliothèques, des variables d'environnement et des fichiers de configuration.

CONTENEUR

Instance d'une image

VOLUME

Dossier qui contourne le UFS pour persister ou partager les données

COMMENT UTILISER DOCKER

ENGINE CLI

```
$ docker

Usage:  docker COMMAND

A self-sufficient runtime for containers
```

Le CLI de Engine permet de contrôler l'ensemble de l'environnement de Docker
(conteneur, images, volumes, etc.)

QUELQUES COMMANDES DE BASE

LANCER UN CONTENEUR

```
$ docker run [OPTIONS] IMAGE [COMMAND] [ARG...]
```

Exemple:

```
$ docker run hello-world
```

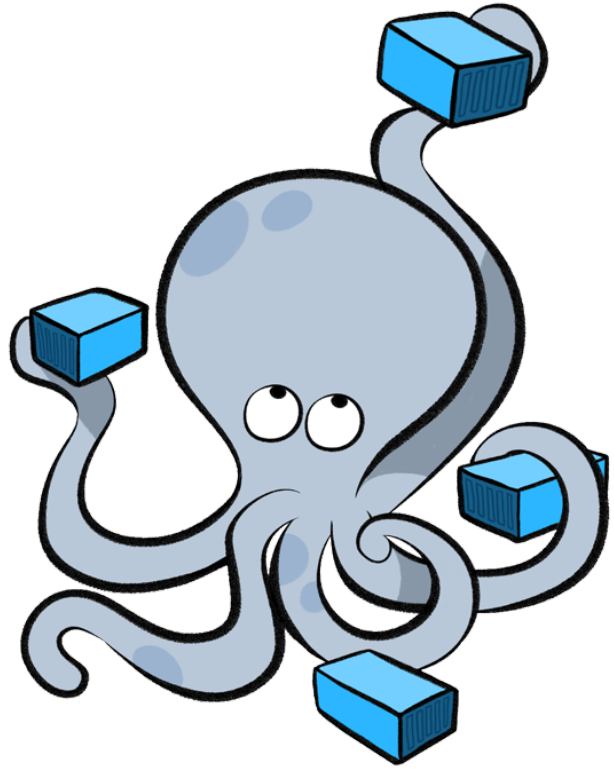
LISTER LES CONTENEURS

```
$ docker ps
```

LISTER LES IMAGES

```
$ docker images
```

DOCKER COMPOSE



Compose est un outil pour définir et lancer des applications Docker multi-conteneurs (ou pas).

DOCKER-COMPOSE.YML

Permet de définir:

- les services
- les volumes
- les réseaux

EXEMPLE SIMPLE

```
version: '3'

services:
  php_apache:
    image: php:5-apache
    ports:
      - "8000:80"
    volumes:
      - ./evolix.ca/:/var/www/html/:ro
      - ./config/php.ini:/etc/php5/apache2/php.ini:ro
      - ./config/000-default.conf:/etc/apache2/sites-available/000-default.conf:ro
```

EXEMPLE MULTI-CONTENEURS

```
  version: '2.2'
  services:
      nginx:
          build: docker/nginx
          image: lanets-website/nginx
          links:
              - uwsgi-websockets
              - uwsgi-django
          depends_on:
              - uwsgi-websockets
              - uwsgi-django
          ports:
              - ${LANETS_WEBSITE_PORT}:80

      uwsgi-websockets:
          build: docker/uwsgi-websockets
          image: lanets-website/uwsgi-websockets

      uwsgi-django:
          build: docker/uwsgi-django
          image: lanets-website/uwsgi-django

      mariadb:
        cpu_percent: 70
        image: mariadb
        environment:
            MYSQL_USER: lanets
            MYSQL_PASSWORD: lanets
            MYSQL_DATABASE: lanets
            MYSQL_ROOT_PASSWORD: lanets
        volumes:
          - db_data:/var/lib/mysql
volumes:
  db_data:
```

CRÉER VOTRE ENVIRONNEMENT DE DEV

SÉPARER LES ÉLÉMENTS DE VOTRE ENVIRONNEMENT

1 processus = 1 conteneur (idéalement)

STACK WEB CLASSIQUE

- Serveur Web (Apache, Nginx, ...)
- Serveur Applicatif (php-fpm, uwsgi, tomcat, ...)
- Base de données (mariadb, mongodb, ...)

CRÉER SON DOCKER-COMPOSE.YML

1. Choisir ses images ou créer des Dockerfile
2. Définir les services
3. Déclarer les variables d'environnement
4. Déclarer les ports et les liens entre les services
5. Déclarer les volumes des services pour le code de l'application et les fichiers de configuration
6. Déclarer les volumes persistant (si nécessaire)

EXEMPLE MULTI-CONTENEURS

```
version: '2'

services:
  php_apache:
    image: php:5-apache
    ports:
      - "8000:80"
    links:
      - db
    volumes:
      - ./evolix.ca/:/var/www/html/:ro
      - ./config/php.ini:/etc/php5/apache2/php.ini:ro
      - ./config/000-default.conf:/etc/apache2/sites-available/000-default.conf:ro

  db:
    image: "mysql:5"
    volumes:
      - ./mysql:/etc/mysql/conf.d
    environment:
      MYSQL_ROOT_PASSWORD: phpapptest
      MYSQL_DATABASE: phpapp
```

EXEMPLE EVOLIX.CA

- Stack simple
- Réplication d'une platforme existante
- Apache + PHP
- Utilisation d'une image existante + configuration

DOCKER-COMPOSE.YML

```
version: '3'

services:
  php_apache:
    image: php:5-apache
    ports:
      - "8000:80"
    volumes:
      - ./evolix.ca/:/var/www/html/:ro
      - ./config/php.ini:/etc/php5/apache2/php.ini:ro
      - ./config/000-default.conf:/etc/apache2/sites-available/000-default.conf:ro
```