

DevOps

Cours 2: Intégration Continue (CI)

Antoine Balliet
antoine.balliet@dauphine.psl.eu

Dans l'épisode précédent ...

Cours

- Présentation générale
 - La culture “DevOps”
 - Les enjeux du déploiement en informatique
 - Notions abordées dans ce cours

Docker

- Conteneurisation
 - Repertoire d'images : <https://hub.docker.com/>
 - Dockerfile
 - Configuration
 - *build* étape par étape et cache
 - Isolation
 - Immutabilité
- Docker CLI

Dans l'épisode précédent ...

- Reproductibilité
 - Dockerfile

```
# Use an official Node runtime as the parent image
FROM node:lts
# Set the working directory in the container to /app
WORKDIR /app
# Copy the current directory contents into the container at /app
ADD . /app
# Make the container's port 80 available to the outside world
EXPOSE 80
# Run app.js using node when the container launches
CMD ["node", "app.js"]
```

- docker build -t xyz .
- docker run ...

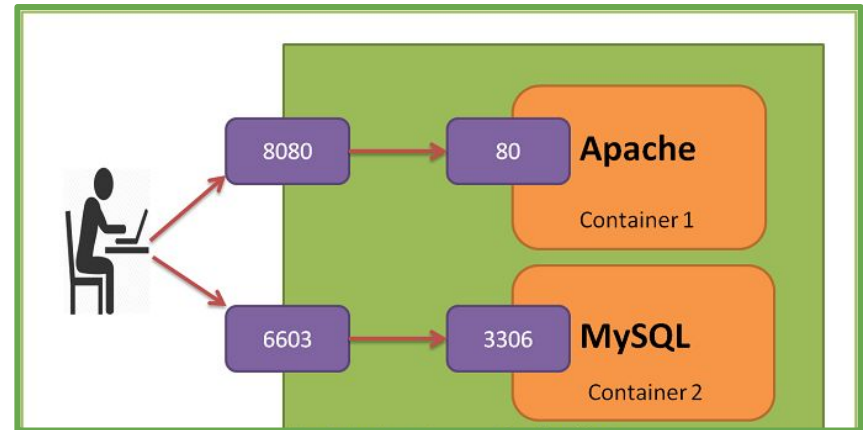
Dans l'épisode précédent ...

- Isolation
 - MAIS “binding” de port

Partie 3 Executer un conteneur [🔗](#)

Dans ce module, vous exécutez des conteneurs basés sur l'i

```
docker run -p 4000:80 --name my-app node-app:0.1
```

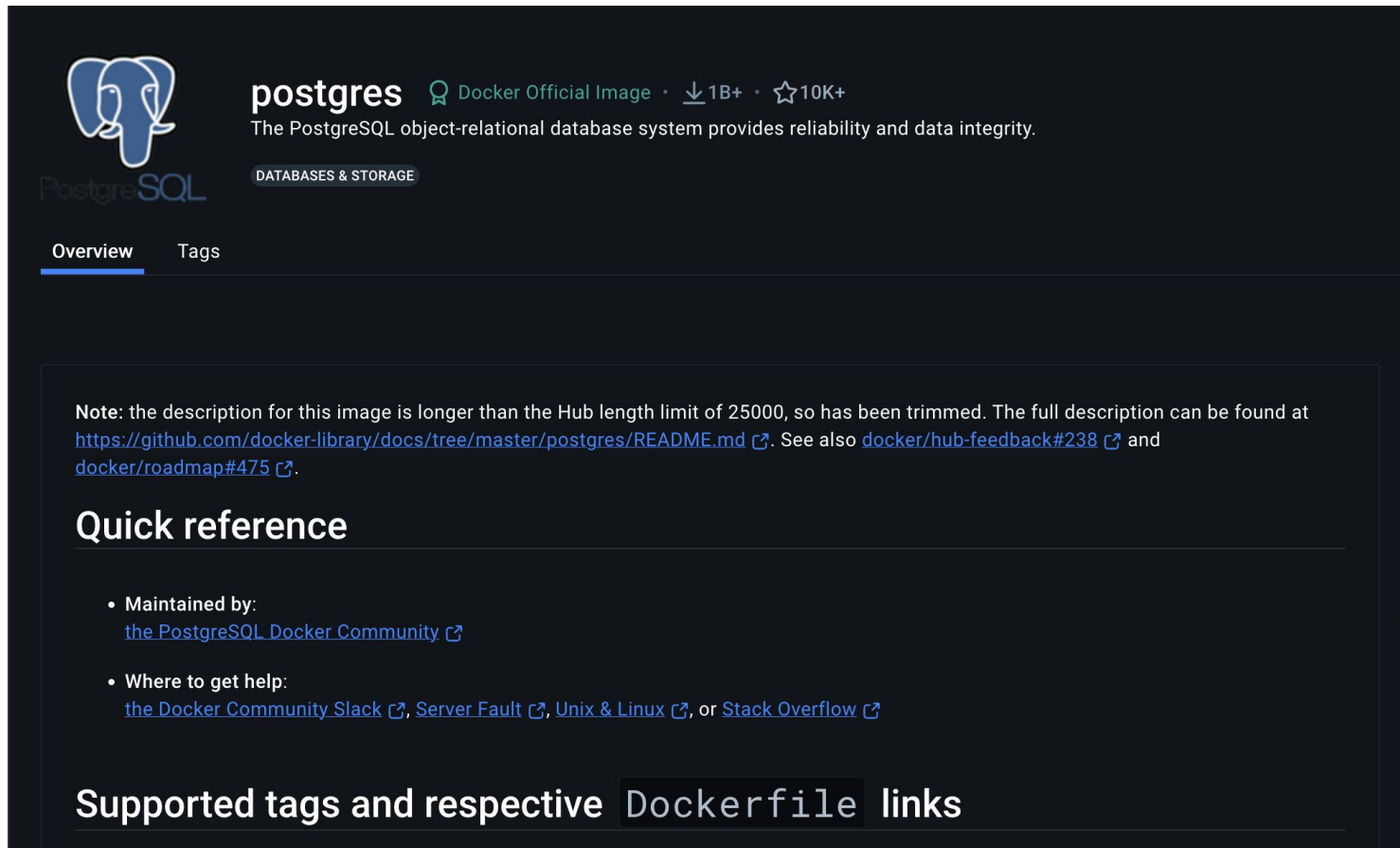


- MAIS montage de volume





```
docker run -d \
  --name postgres-container \
  -e POSTGRES_USER=your_username \
  -e POSTGRES_PASSWORD=your_password \
  -e POSTGRES_DB=your_database \
  -v ~/postgres_data:/var/lib/postgresql/data \
  -p 5432:5432 \
  postgres
```

Dans l'épisode précédent ...

- Réutilisation de code
 - Container postgres avec DockerHub



The screenshot shows the Docker Hub page for the 'postgres' image. At the top left is the PostgreSQL logo (an elephant) and the text 'PostgreSQL'. To the right, the word 'postgres' is displayed in a large font, followed by 'Docker Official Image', download statistics '1B+', and star statistics '10K+'. Below this is a description: 'The PostgreSQL object-relational database system provides reliability and data integrity.' and a category tag 'DATABASES & STORAGE'. There are two tabs: 'Overview' (selected) and 'Tags'. A note states: 'Note: the description for this image is longer than the Hub length limit of 25000, so has been trimmed. The full description can be found at <https://github.com/docker-library/docs/tree/master/postgres/README.md>. See also [docker/hub-feedback#238](#) and [docker/roadmap#475](#).' Below the note is a 'Quick reference' section with two bullet points: 'Maintained by: [the PostgreSQL Docker Community](#)' and 'Where to get help: [the Docker Community Slack](#), [Server Fault](#), [Unix & Linux](#), or [Stack Overflow](#)'. At the bottom, the text 'Supported tags and respective Dockerfile links' is shown.

 **postgres**  Docker Official Image ·  1B+ ·  10K+

The PostgreSQL object-relational database system provides reliability and data integrity.

DATABASES & STORAGE

Overview Tags

Note: the description for this image is longer than the Hub length limit of 25000, so has been trimmed. The full description can be found at <https://github.com/docker-library/docs/tree/master/postgres/README.md>. See also [docker/hub-feedback#238](#) and [docker/roadmap#475](#).



Quick reference


- Maintained by:
[the PostgreSQL Docker Community](#)
- Where to get help:
[the Docker Community Slack](#), [Server Fault](#), [Unix & Linux](#), or [Stack Overflow](#)


Supported tags and respective Dockerfile links


Dans l'épisode précédent ...

- Artifact Registry
 - Stockage de container

 Artifact Registry 

 Dépôts

 Paramètres

 Créer un dépôt

Format

- ☒ Docker
- ☐ Maven
- ☐ npm
- ☐ Python
- ☐ Apt
- ☐ Yum
- ☐ Kubeflow Pipelines
- ☐ Go
- ☐ Générique BÊTA

Correction partie 2 du TP docker (en live !)

```
docker run -d \  
  --name postgres-container \  
  -e POSTGRES_USER=your_username \  
  -e POSTGRES_PASSWORD=your_password \  
  -e POSTGRES_DB=your_database \  
  -p 5432:5432 \  
  postgres
```

Création d'un container
postgres : on run l'image
"postgres"

```
antoine_balliet@cloudshell:~$ docker ps  
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                    NAMES  
57cf81201623   postgres  "docker-entrypoint.s..." 23 minutes ago Up 23 minutes  0.0.0.0:5432->5432/tcp   postgres-container  
antoine_balliet@cloudshell:~$
```

Depuis l'intérieur de mon instance docker ...

```
PGPASSWORD=your_password psql -U your_username -d your_database
```

Directement sur mon cloud shell grâce au binding du port ...

```
PGPASSWORD=your_password psql -h 0.0.0.0 -U your_username -d your_database
```

Correction partie 2 du TP docker (en live !)

On liste les tables

```
\dt
your_database=# \dt
Did not find any relations.
your_database=#
```

On crée une table dans notre base de donnée

```
CREATE TABLE techo (id VARCHAR NOT NULL PRIMARY
KEY, name varchar(50) NOT NULL UNIQUE);
```

```
your_database=# \dt
Did not find any relations.
your_database=# CREATE TABLE techo (id VARCHAR NOT NULL PRIMARY KEY, name varchar(50) NOT NULL UNIQUE);
CREATE TABLE
your_database=# \dt
List of relations
 Schema | Name   | Type  | Owner
-----+-----+-----+-----
 public | techo  | table | your_username
(1 row)

your_database=#
```


Correction partie 2 du TP docker (en live !)

On restart le container => pas de changement

```
your_database=# \dt
               List of relations
 Schema | Name  | Type  | Owner
-----+-----+-----+-----
 public | techo | table | your_username
(1 row)

your_database=# █
```

On le supprime et on le recrée => perte de la table

Si on utilise un volume => pas de perte de la table :)

```
docker run -d \
  --name postgres-container \
  -e POSTGRES_USER=your_username \
  -e POSTGRES_PASSWORD=your_password \
  -e POSTGRES_DB=your_database \
  -v ~/postgres_data:/var/lib/postgresql/data \
  -p 5432:5432 \
  postgres
```

On supprime le dossier monté en volume contenant notre base avec la commande :

```
sudo rm -r postgres_data/
```

Dans l'épisode précédent ...

Cloud Provider

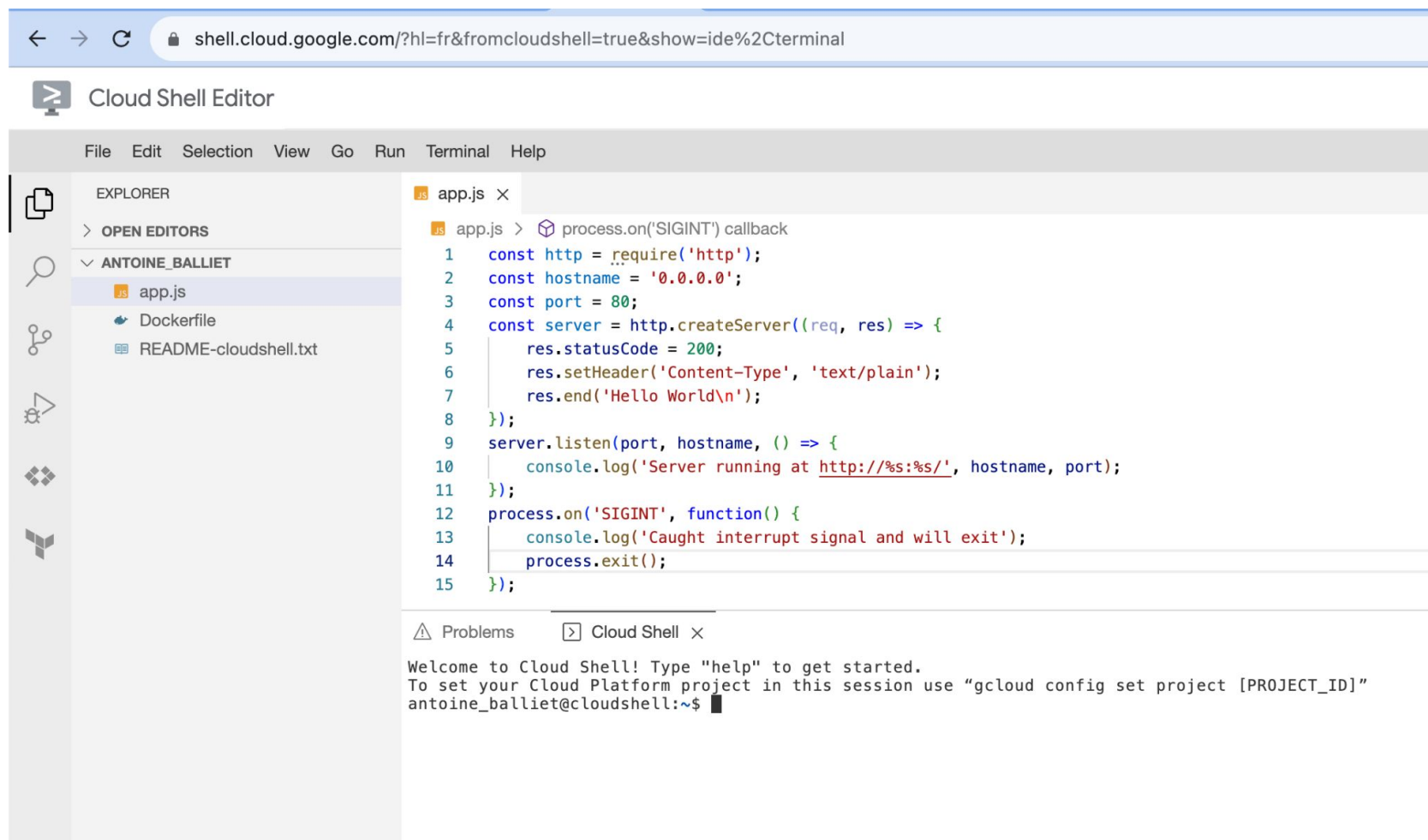
- Familiarisation avec Google Cloud Platform (GCP)
 - Cloud Shell : Terminal / éditeur de texte
 - Google Console
 - Coupons (?)



 Ouvrir l'éditeur

Dans l'épisode précédent ...

<https://shell.cloud.google.com/?hl=fr&fromcloudshell=true&show=ide%2Cterminal>



The screenshot displays the Google Cloud Shell Editor interface. At the top, the browser address bar shows the URL `https://shell.cloud.google.com/?hl=fr&fromcloudshell=true&show=ide%2Cterminal`. Below the address bar, the "Cloud Shell Editor" title is visible. The interface is divided into three main sections: a left sidebar, a central editor, and a bottom terminal.

Left Sidebar: Contains the "EXPLORER" view showing the file structure. Under "OPEN EDITORS", the file `app.js` is listed. Below it, the file explorer shows `ANTOINE_BALLIET` with sub-items `app.js`, `Dockerfile`, and `README-cloudshell.txt`.

Central Editor: Displays the code for `app.js`. The code is as follows:

```
1  const http = require('http');
2  const hostname = '0.0.0.0';
3  const port = 80;
4  const server = http.createServer((req, res) => {
5    res.statusCode = 200;
6    res.setHeader('Content-Type', 'text/plain');
7    res.end('Hello World\n');
8  });
9  server.listen(port, hostname, () => {
10   console.log('Server running at http://%s:%s/', hostname, port);
11 });
12 process.on('SIGINT', function() {
13   console.log('Caught interrupt signal and will exit');
14   process.exit();
15 });
```

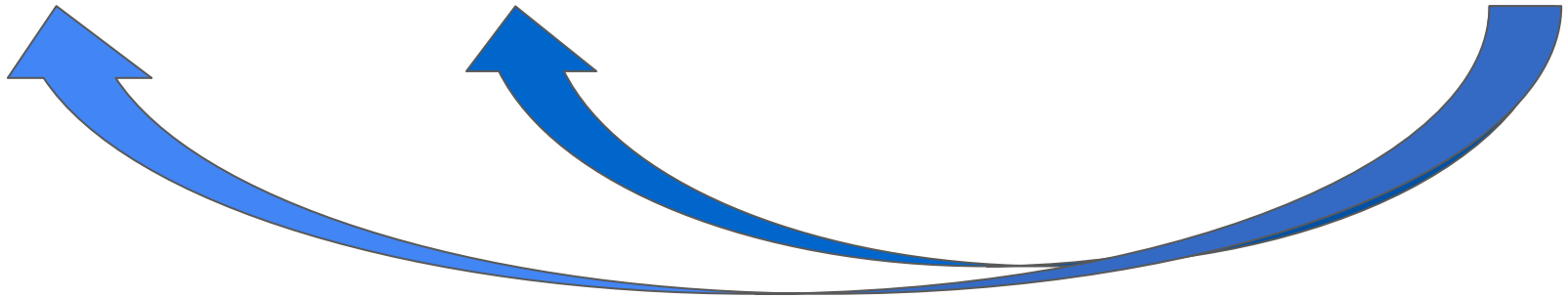
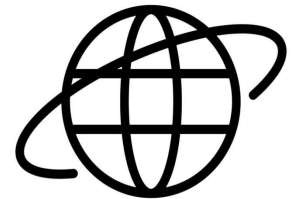
Bottom Terminal: Shows the "Problems" and "Cloud Shell" tabs. The "Cloud Shell" tab is active, displaying the following text:

```
Welcome to Cloud Shell! Type "help" to get started.
To set your Cloud Platform project in this session use "gcloud config set project [PROJECT_ID]"
antoine_balliet@cloudshell:~$
```

Prérequis

- 1 ordinateur pour 2
- Connaissance git ?
- Compte GitHub + GCP 🕶️
- Curiosité : n'hésitez pas à chercher sur internet :)

Vélocité 🚀



Critique du TP docker

Commandes
fastidieuses



Se souvenir de ce
qu'on a fait 🙄

Quels problèmes avons-nous constaté
pendant le TP docker ?

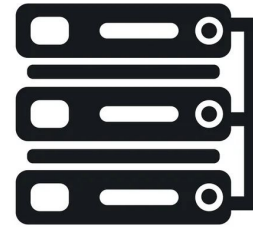
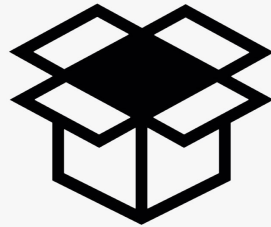
Documentation pas
simple 🙄

Passage de
paramètres 😡

Limitations visibles

- Build en local ?
- Configuration ?
- Collaboratif ?
- Stockage ?
- Version ?
- Automatisation ?

Processus de livraison d'une application



- Installer les dépendances
- Configurer Firewall
- Déployer l'application
- Vérifier son comportement
- Adapter les ressources à la charge
- ...

Approche DevOps

“You build it, you run it” (2006) - Werner Vogels
(CTO & VP @Amazon)

“The most powerful tool we have as developers is automation”
Scott Hanselman (VP @Microsoft)

“Deployment celebrations should be about the value of the new features, not joyful relief that nothing went horribly wrong”
Rebecca Parsons (CTO @Thoughtworks)

Continuous Integration & Continuous Delivery



Continuous Integration

Continuous Delivery

Les outils de gestion de versions

- Infrastructure ?
 - Pool de machines interne à l'entreprise
 - Offre SaaS / Produit d'entreprise : GitHub
 - Cloud provider : hébergement sur GCP
- Taille des fichiers ?
 - Code source : léger (10 Mo MAX)
 - Binaires / Docker images : 10 Mo - 10 Go
 - Modèle de Machine Learning (500 Go+)
- Fonctionnalités ?
 - Différences, réorganisation des changements
 - Fusion de versions
 - Rapidité de téléchargement
 - Compatibilité entre les outils

Les outils de gestion de versions

Max 2 GB par repo



Cloud Source
Repositories

Max 500 GB par repo



Amazon ECR



JFrog
ARTIFACTORY

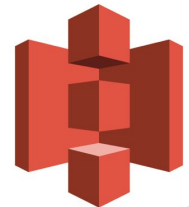
Max 5TB par fichier



Drive



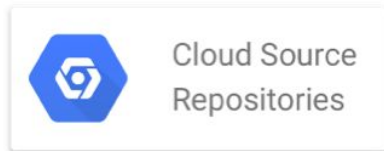
Google Cloud Storage



Amazon S3

Repo Code Source

Max 2 GB par repo



Repo images Docker, Binaires, Packages

Max 500 GB par repo



Version de fichiers

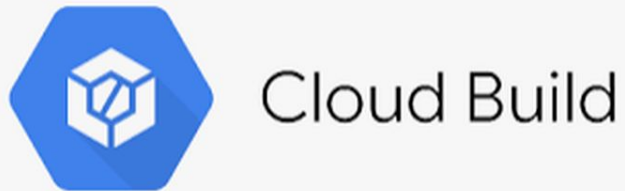
Max 5TB par fichier



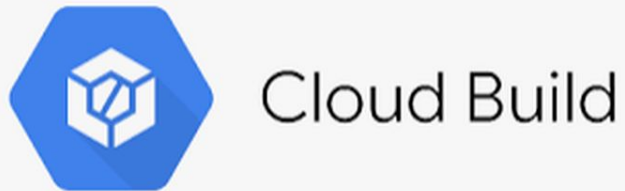
Les outils d'automatisation

- Infrastructure ?
 - Pool de machines interne à l'entreprise
 - Cloud Provider
 - Runner dans système de versioning
- Langage ?
 - Impératif : du code
 - Déclaratif : YAML
- Déclencheur & résultat ?
 - Événement sur le repository
 - Sorties multiples : email, teams, GitHub ...

Les outils d'automatisation



SaaS (serverless)



A herberger soi-même



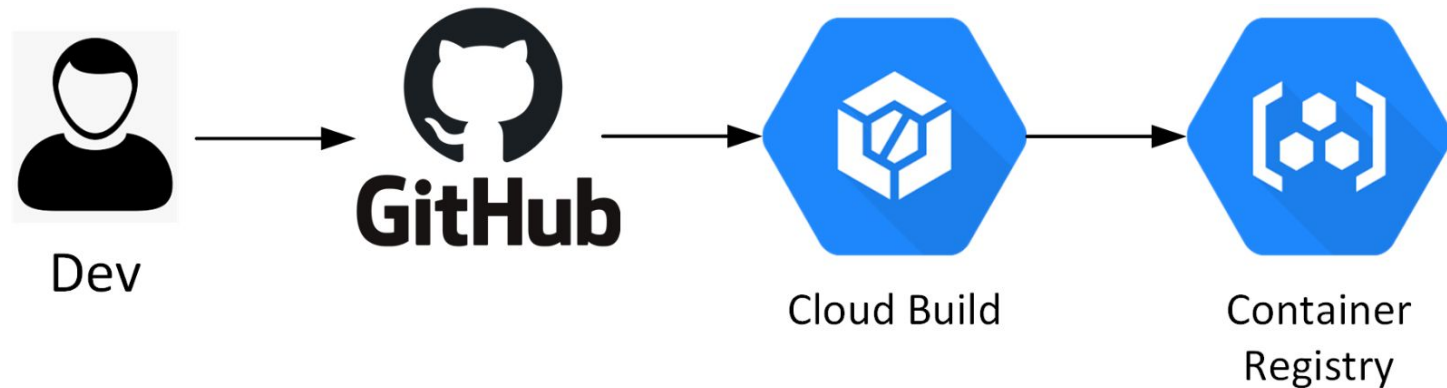
Déclaratif ?

YAML : texte structuré, pas de dépendances, pas d'exécution tel quel

Similaire au JSON, HTML

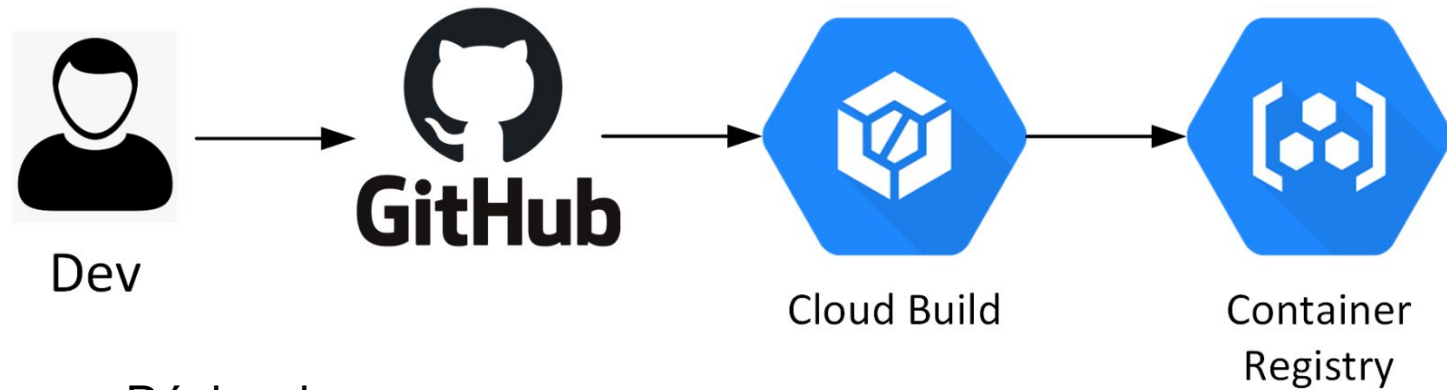
```
1 # This is a comment
2 title: This is some YAML
3 publisher: ConvertSimple Books
4 pages: 250
5 chapters: 21
6 time_to_read: 12 hours
7 descriptors:
8   - fun
9   - entertaining
10  - exciting
11 contributors:
12   author: Mark Templeton
13   editor: Cindy Johnson
```

Pipeline



- Vérification : style de code, tests unitaires du code, compilation ...
- *docker build*
- *docker tag*
- *docker push*
- Déploiement ...

Collaboratif ?



- Déclencheurs
 - Commit
 - Pull requests
- Règle d'automatisation par branches
- Tag des images

Pour aller plus loin

- Semver : <https://semver.org/lang/fr/>
- CI / CD chez Spotify
<https://engineering.atspotify.com/2020/08/how-we-improved-developer-productivity-for-our-devops-teams/> (en Anglais)
- CI /CD chez Netflix
<https://www.bunnysHELL.com/blog/how-netflix-does-devops/> (en Anglais)
- CI / CD chez Facebook
<https://www.youtube.com/watch?t=1896&v=X0VH78ye4yY> (en Anglais)