PARIS TEST CONF
10/10/2023

Accélérer l'automatisation par le DevOps

Jean-François FRESI

WerinGroup
QUALITY ENGINEERING

Conférence

Au-delà de l’automatisation

“Simply put, things always had to be in a production-ready state: if you wrote it, you darn well had to be there to get it running! “

Mike Miller , General Partner at Liquid 2 Ventures

“The most powerful tool we have as developers is automation.”

Scott Hanselman , teacher and programmer

Qu’est-ce que le DevOps

Je veux des changements

Dev

mur de la confusion

Je veux de la stabilité

Ops

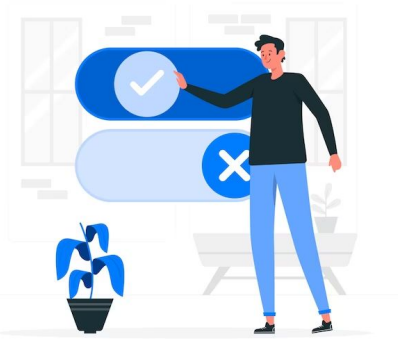
automatisation

culture

Le DevOps et le test

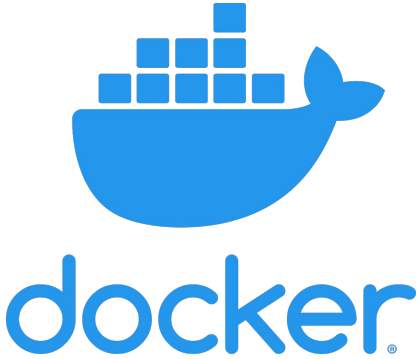
Installation hétérogène

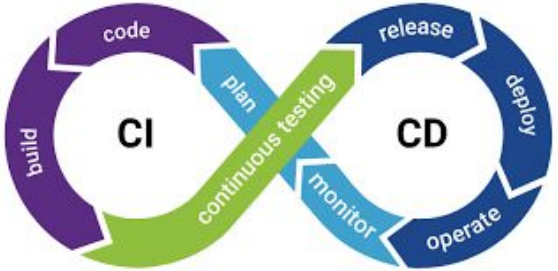
Complexité de configuration



Déploiement et test en continu

Les outils du DevOps à votre secours





Pipeline

Pourquoi Docker ?

Contexte

- Installation longue et fastidieuse
- Accélérer le time to market
- Configurations hétérogènes
- Ça ne marche pas sur ma machine

Enjeux

- Éviter la perte de temps
- Harmoniser des environnements
- Intégration continue facilitée
- Développement standardisé

Challenges

- Multi-navigateurs
- Debug et test des scripts
- Synchro code et des rapports
- Facilité d'utilisation

Une installation complexe ?

Une installation en plusieurs étapes

Installer Python 3.X et PIP	Installer RobotFramework via PIP	Installer les librairies nécessaires	Télécharger les Webdrivers	Paramétrer les Variables d'ENV
https://www.python.org/downloads Sélectionner la version en fonction de votre OS Ajouter Python au PATH (variable env)	Lancer dans un terminal la commande : pip install robotframework --upgrade	Lancer dans un terminal les commandes types : pip install robotframework-SeleniumLibrary	Télécharger les webdrivers du navigateur souhaité	Paramétrer les variables d'environnement afin que votre OS reconnaisse les commandes robotframework

Docker avantages et contraintes

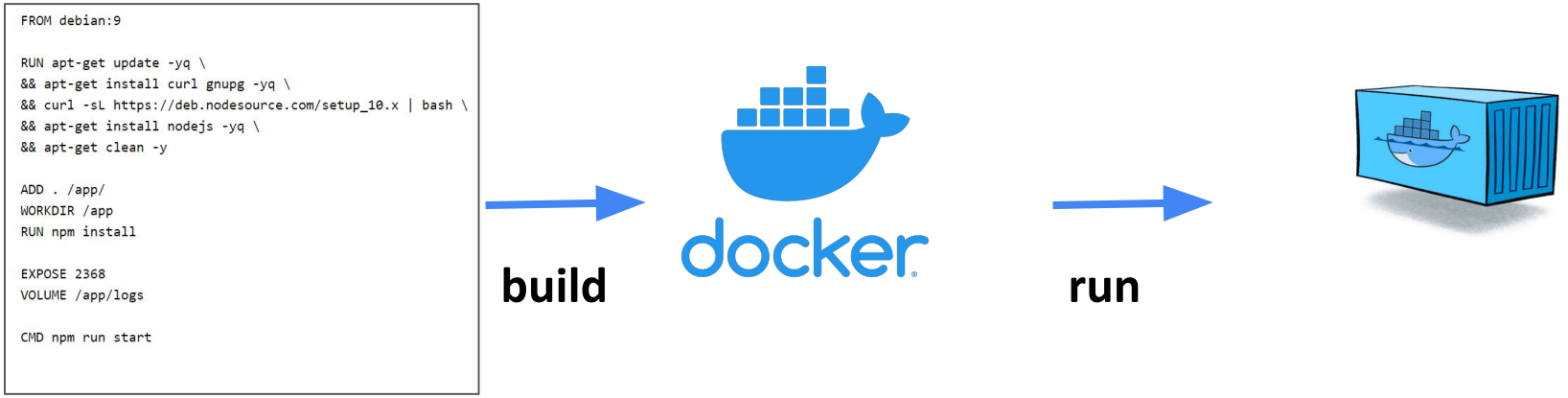
Avantages

- Isolation
- Répétabilité des tests
- Gestion simplifiée des environnements
- Productivité
- Portabilité
- Flexibilité

Contraintes

- Complexité initiale
- Compétences internes nécessaires
- Surcharge des ressources
- Investissement dans des machines adaptées

C’est quoi un container ?



Source: https://cto.ai/blog/docker-image-vs-container-vs-dockerfile/

Utiliser Docker

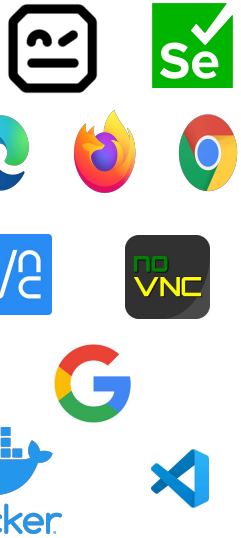
- **Installer** Docker
- **Créer une Image** RobotFramework
  - À partir image ubuntu
  - Installation navigateurs et drivers associés
  - Installation python
  - Installation selenium
  - Installation outils VNC et No VNC
  - Configuration des ports et des volumes

```
FROM ubuntu:18.04
RUN apt-get update && apt-get upgrade

# Essential tools and xvfb
RUN apt-get update && apt-get install -y \
    software-properties-common \
    unzip \
    curl \
    novnc \
    websockify \
    xvfb \
    x11vnc \
    fluxbox \
    nano && \
    rm -rf /var/lib/apt/lists/*

# Chrome browser to run the tests
ARG CHROME_VERSION="google-chrome-stable"
RUN curl https://dl-ssl.google.com/linux/linux_signing_key.pub -o /tmp/google.pub \
    && cat /tmp/google.pub | apt-key add -; rm /tmp/google.pub \
    && echo 'deb http://dl.google.com/linux/chrome/deb/ stable main' > /etc/apt/sources.list.d/google.list \
    && mkdir -p /usr/share/desktop-directories \
    && apt-get -y update && apt-get install -y  $CHROME_VERSION
# Disable the SUID sandbox so that chrome can launch without being in a privileged container
RUN dpkg-divert --add --rename --divert /opt/google/chrome/google-chrome.real /opt/google/chrome/google-chrome
    && echo "#!/bin/bash\nexec /opt/google/chrome/google-chrome.real --no-sandbox --disable-setuid-sandbox \"\$
    && chmod 755 /opt/google/chrome/google-chrome

#---------------------------------------------------------------------
# Chrome Driver
ARG CHROME_DRIVER_VERSION
RUN if [ -z "$CHROME_DRIVER_VERSION" ]; \
    then CHROME_MAJOR_VERSION=$(google-chrome --version | sed -E "s/.* ([0-9]+)(\.[0-9]+){3}.*/\1/") \
    && NO_SUCH_KEY=$(curl -ls https://chromedriver.storage.googleapis.com/LATEST_RELEASE_${CHROME_MAJOR_VERSION
    if [ -n "$NO_SUCH_KEY" ]; then \
    echo "No Chromedriver for version $CHROME_MAJOR_VERSION. Use previous major version instead" \
```

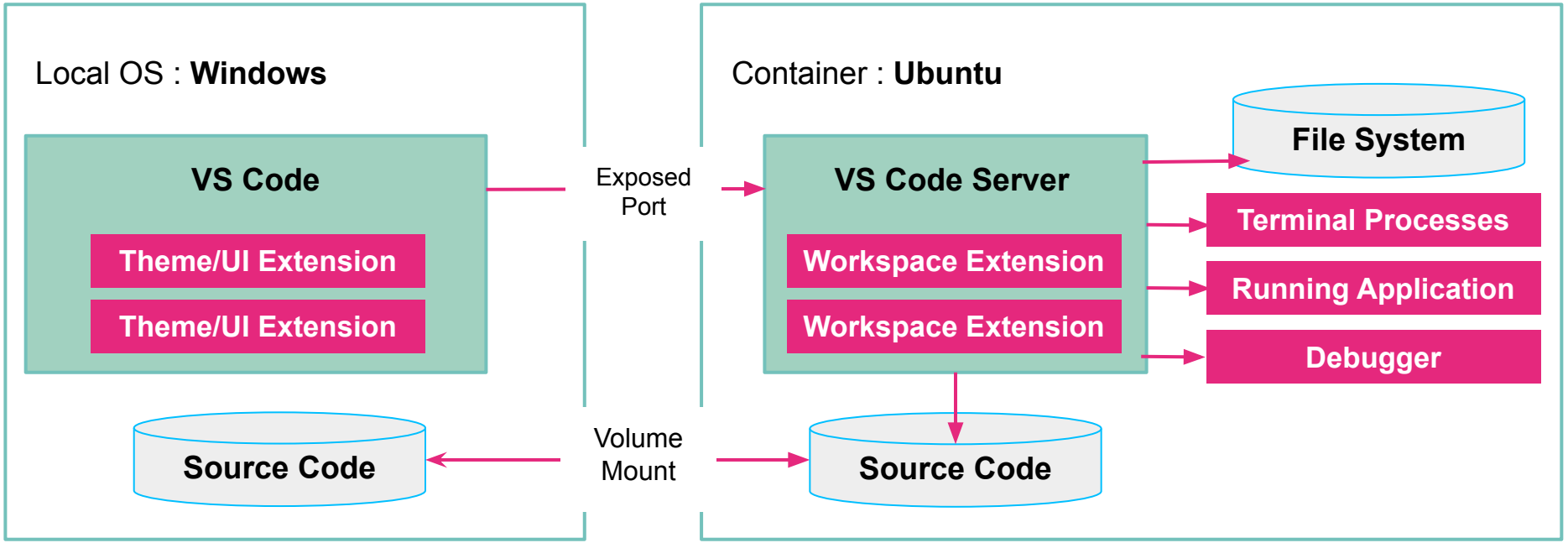
Contexte de la démo docker

Contexte de la démo

- **Robot Framework / Selenium**
- Navigateurs **Chrome, Firefox et Edge**
- Debug / voir les tests s'exécuter **: VNC / noVNC**
- **SUT :** home page de google
- **Editeur : VS code**



Comment automatiser dans un container ?



Source:https://code.visualstudio.com/docs/devcontainers/containers

Docker : Comment est créé le container ?

Création de l'image

docker build -t robotsel-containervsept13 **.**

nom de l'image

Création du container

docker run -it -p 5900:5900 -p 6080:6080 --name robotsel-sept13 -v "$(pwd):/home/Workspace/" robotsel-containervsept13

ports utilisés

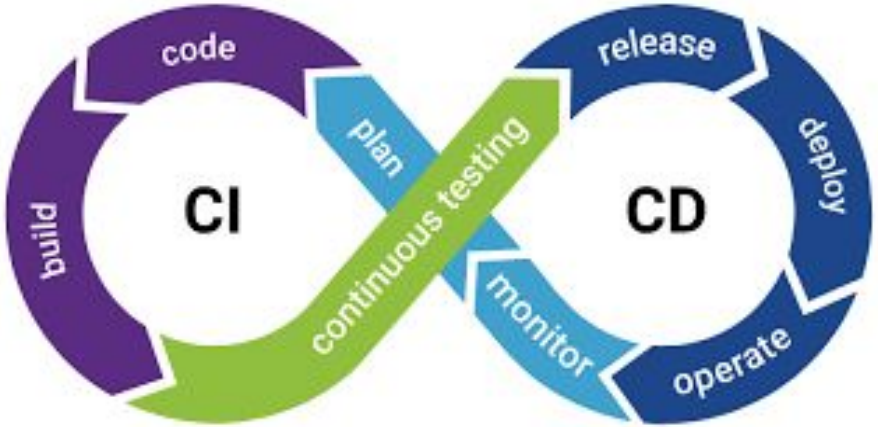
nom container

L'image

Volume

Démo Docker

Et maintenant le CI/CD



Pourquoi CI/CD ?

Nos attentes

- Détecter plus rapidement les problèmes
- Accélérer le time to market et les tests
- Utiliser efficacement l'ensemble des ressources
- Partager les résultats à 360
- Améliorer la productivité

Enjeux

- Éviter la perte de temps
- Feedback immédiat
- Amélioration de la qualité
- Industrialisation processus et outillage

CI/CD avantages et contraintes

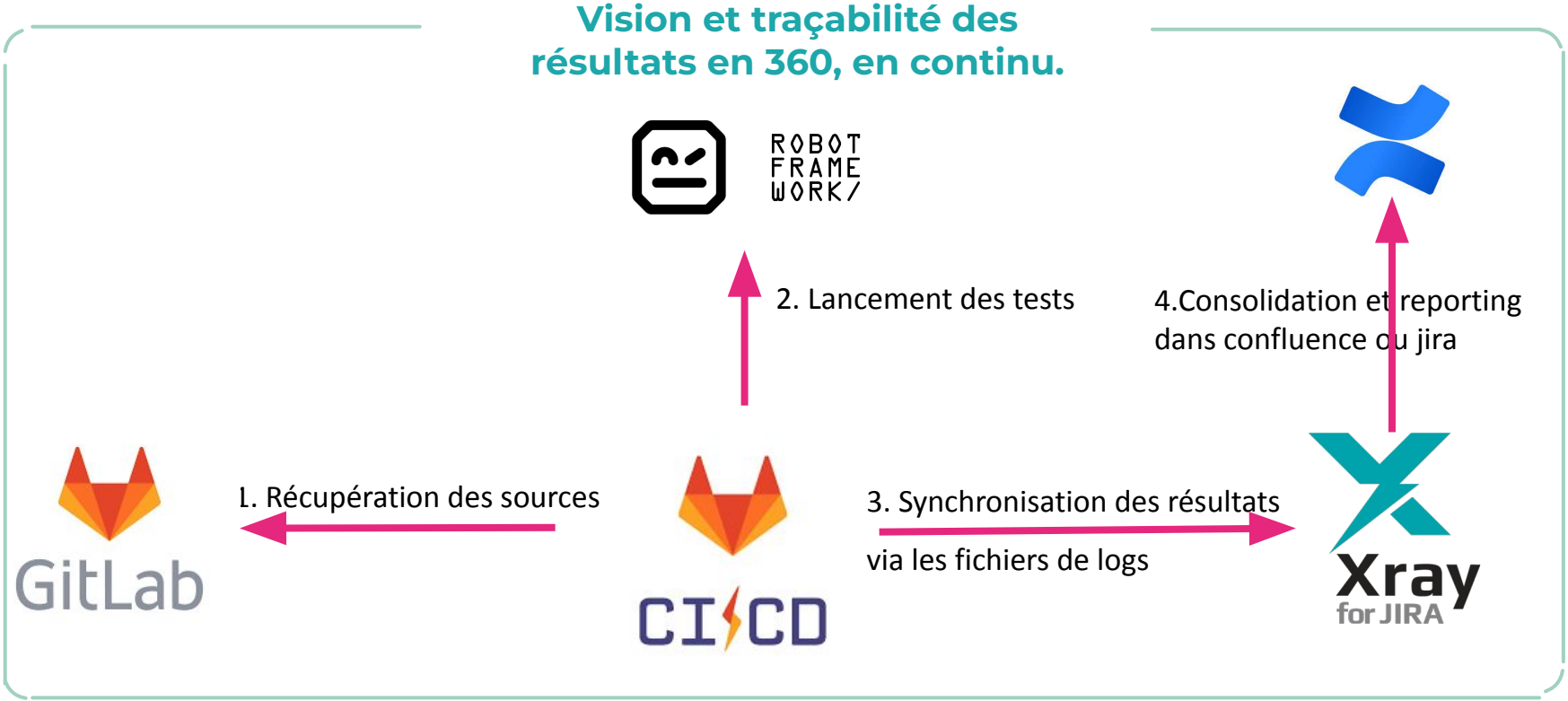
Avantages

- Feedbacks rapides
- Tester en parallèles
- Valider plusieurs types de configurations
- Inclure des tests non fonctionnels

Contraintes

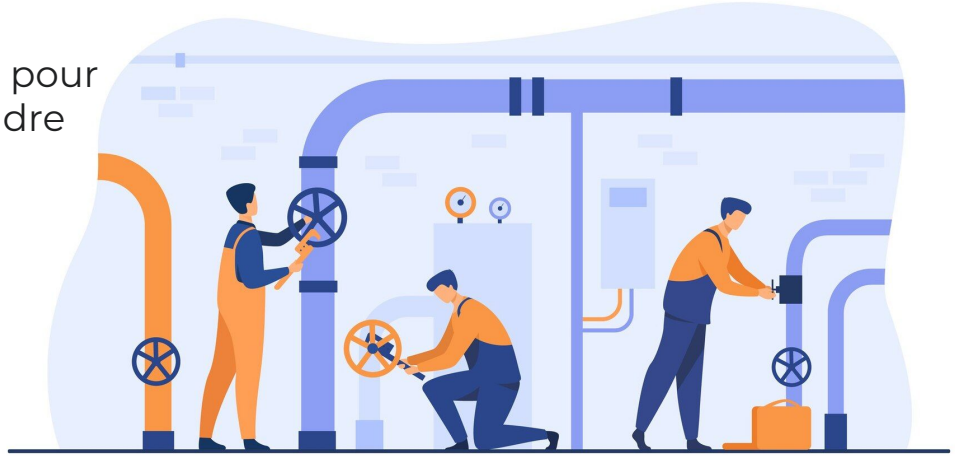
- Complexité de la configuration initiale
- Maintenance et évolutivité
- Compétences internes nécessaires
- Investissement dans des licences et des environnements

CI/CD pour le testing - un exemple



CI/ CD - le YAML

- Acronyme de **Yet Another Markup Language**
- Devenu **Ain't Markup Language ;** ce n’est pas un langage de balisage
- **Langage de sérialisation de données** utilisé pour écrire des fichiers de configuration dans le cadre d'un déploiement d'infrastructures



Le pipeline - yaml exemple

.gitlab-ci.yml 533 bytes

Edit in pipeline editor

```yaml
stages:
    - compose
    - test

compose:
    stage: compose
    image: docker/compose:latest
    services:
        - docker:dind
    before_script:
        - docker version
        - docker-compose version
    script:
        - docker-compose down
        - docker-compose up -d

test:
    stage: test
    image: gitlab.weringroup.com:5050/mywerin/templates-modeles/robotframework/roboframework-browserlib-image
    script:
        - python3 --version
        - python3 -m pip install -r requirements.txt
        - robot tests/

```

Le pipeline Gitlab CI

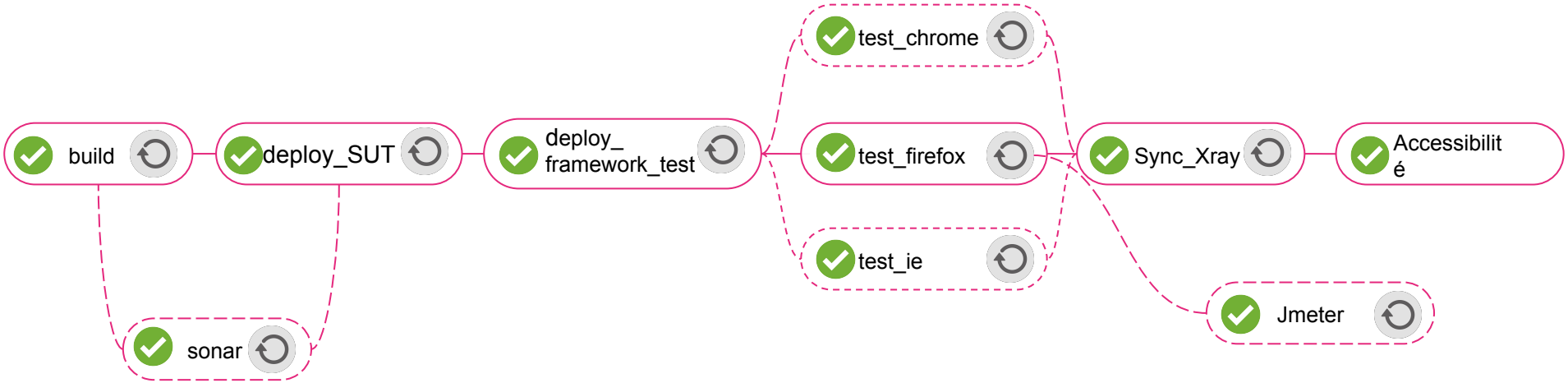
Contexte de la démo CI/CD

Pilotage du pipeline avec Gitlab CI

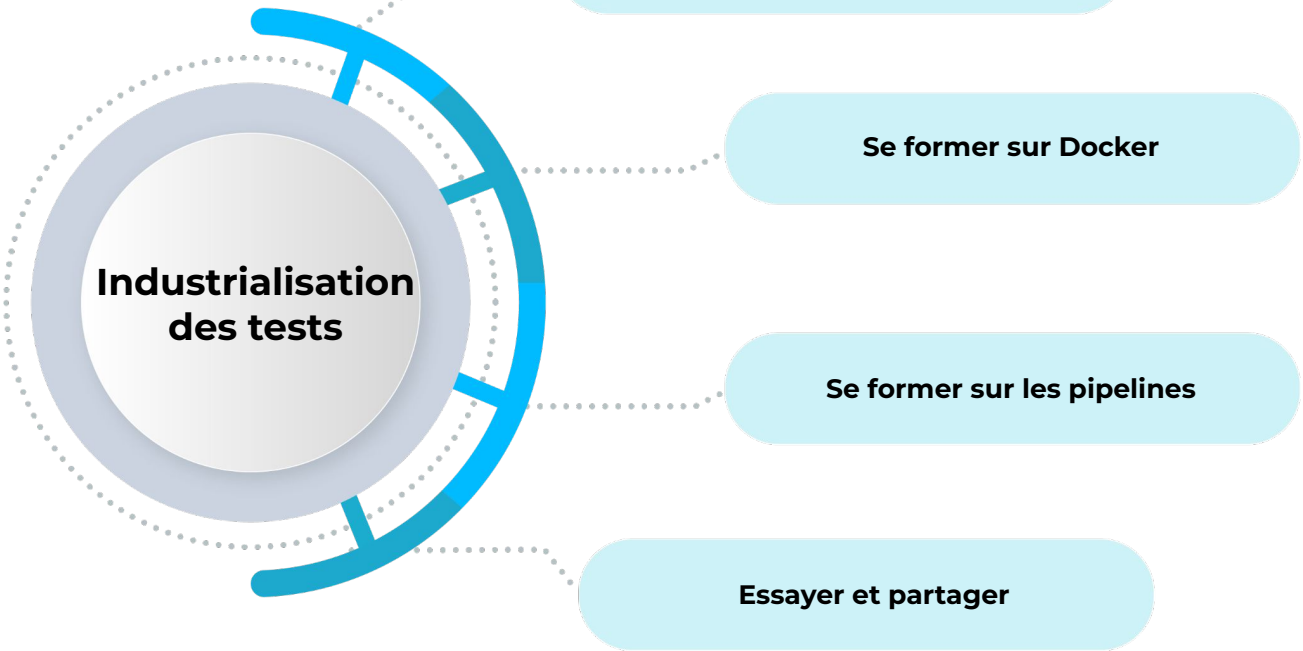
Déploiement SUT	Déploiement Framework de test	Exécution tests	Synchro Xray	Tests non fonctionnels
opencart docker	Se docker	GitLab	Xray	Google Lighthouse

Démo CI/CD

Mais aussi ..


build
deploy_SUT
deploy_
framework_test
test_chrome
test_firefox
test_ie
Sync_Xray
Accessibilit
é
sonar
Jmeter

Pour aller plus loin …



PARIS TEST CONF
10/10/2023

Merci de votre écoute

Jean-François FRESI

WerinGroup
QUALITY ENGINEERING