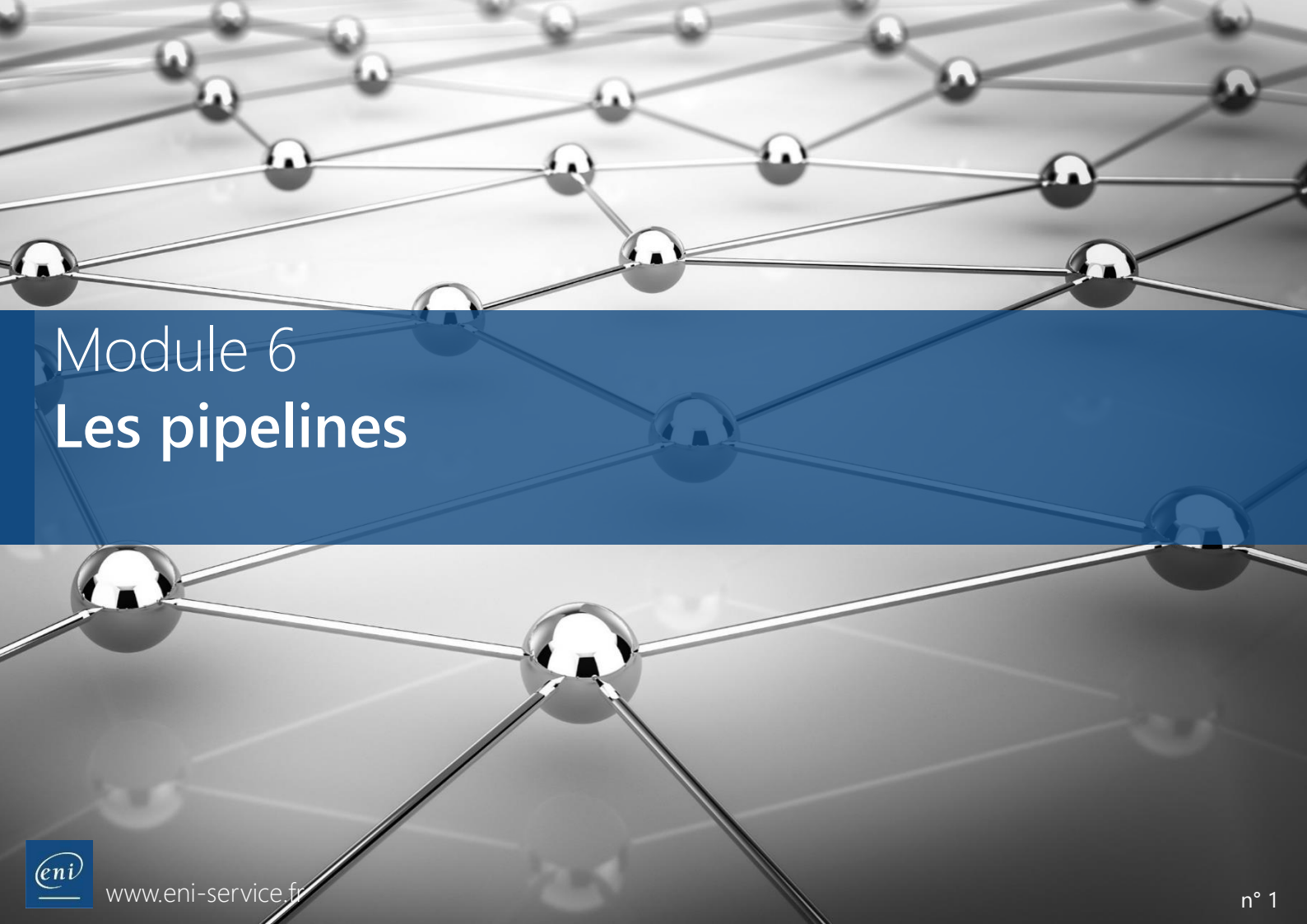


Module 6
Les pipelines

Contenu du module

- La problématique de la construction en étapes
  - Les besoins de conditionnement d'une construction.
  - Intégration, livraison et déploiement continue.
- La prise en charge des pipelines dans Jenkins
  - Projet natif.
  - Extension du support par les plugins.
- Les projets de pipelines
  - Création et déclaration du projet.
  - Vue d'ensemble du DSL de déclaration des pipelines.
  - Assistance à la création des fichiers de pipelines.
- Le plugin Blue Ocean
  - Présentation et installation.
  - Déclaration de pipelines avec Blue Ocean.

Qu'est ce qu'un pipeline ?

- Un pipeline Jenkins, c'est un job qui va être décomposé sous forme :
  - d'étapes qui se succèdent (stage)
  - d'étapes s'exécutant en parallèle (parallel)

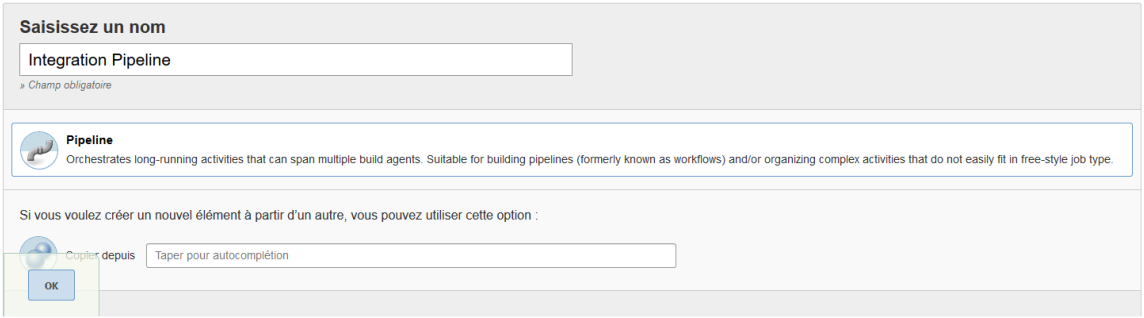
- Les pipelines donnent également la possibilité de choisir précisément quels agents pourront exécuter telle partie de code.
- Les pipelines sont décrits par du code
  - Un DSL (Domain Specific Language) de Groovy.
  - Dans un fichier nommé « JenkinsFile »
    - Potentiellement stocké dans le SCM !!!

- node
  - Un nœud Jenkins pour exécuter le job
- stage
  - Une étape

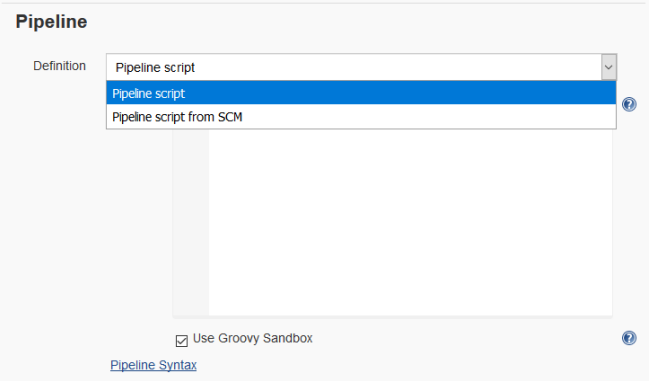
```
node {
    stage('Build') {
        ...
    }
    stage('Test') {
        ...
    }
    stage('Deploy') {
        ...
    }
}
```

Création d'un pipeline

- Menu « Nouveau Item »

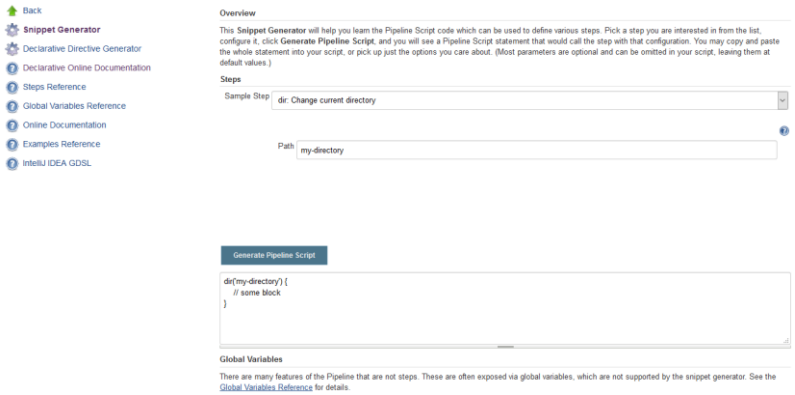


- Il est ensuite possible de créer un pipeline avec un script Groovy, ou bien de référencer un JenkinsFile stocké dans un SCM



Le DSL des pipelines

- La document Jenkins possède un chapitre dédié à la création des pipelines
  - https://jenkins.io/doc/book/pipeline/
- De plus, l'éditeur de pipeline sur un Job de ce type, propose un raccourci permettant d'aider à l'écriture des scripts !



- Un référentiel d'exemples est également disponible à l'adresse : https://github.com/jenkinsci/pipeline-examples

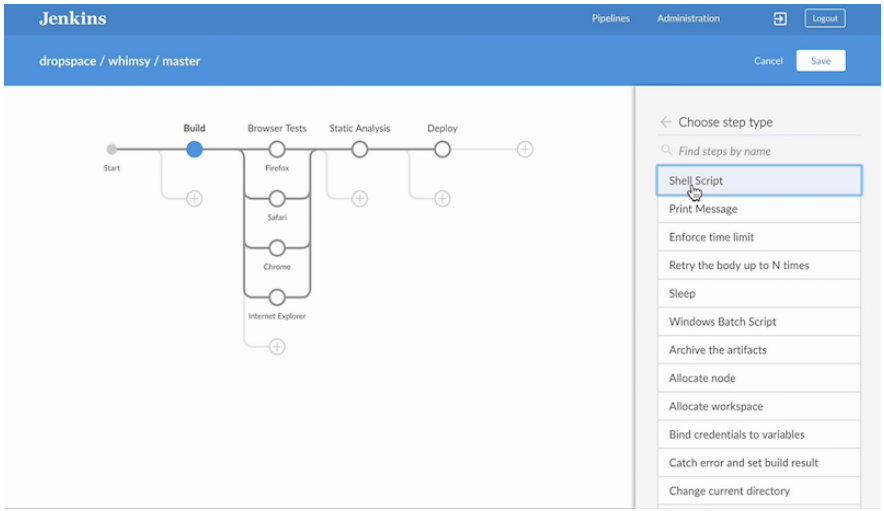
Pipeline

- Exemple de Pipeline
- 4 étapes :
  - Construction
  - Déploiement dans un conteneur
  - Tests fonctionnels
  - Arrêt du conteneur

```
node {
    stage('Construction du projet') {
        // On récupère le code de GitHub
        git 'https://github.com/elanglet/projet-banque.git'
        // On exécute maven sur le projet 'banque'
        withMaven(
            maven: 'Maven 3.6.3',
            jdk: 'JDK 11'
        ) {
            dir('banque/') {
                bat('mvn deploy')
            }
        }
        // On récupère les tests
        junit '**/target/surefire-reports/TEST-*.xml'
    }
    stage('Déploiement dans Docker') {
        dir('docker/') {
            bat('docker build -t tomcat-test:1.0 .')
            bat('docker run  -d --rm -p9000:8080 --name tomcat-test tomcat-test:1.0')
        }
    }
    stage('Execution des tests Web') {
        git 'https://github.com/elanglet/projet-banque-selenium.git'
        withMaven(
            maven: 'Maven 3.6.3',
            jdk: 'JDK 11'
        ) {
            dir('.') {
                bat('mvn test')
            }
        }
    }
    stage('Arrêt du conteneur Docker') {
        bat('docker stop tomcat-test')
    }
}
```

Le plugin Blue Ocean

- Blue Ocean est un plugin Jenkins permettant la construction graphique de pipeline
- Le tout sera stocké dans un JenkinsFile hébergé dans un SCM
- Le plugin Blue Ocean n'est pas installé par défaut, il est donc nécessaire de le faire pour pouvoir exploiter cette interface dédiée aux pipelines



- https://jenkins.io/projects/blueocean/

Travaux Pratiques