

## Chainage de jobs (Pipeline) en Jenkins

Vous avez vu précédemment comment définir des jobs avec Jenkins. Maintenant vous allez voir comment automatiser le chainage de plusieurs jobs (réaliser une séquence de jobs automatiquement). En effet, l'objectif est de définir un ensemble de jobs et qu'ils soient déclenchés automatiquement. Par exemple, suite à une modification dans Git, Jenkins déclenche automatiquement un build, si le build réussit, déploie automatiquement sur l'environnement de test, déclenche les jobs de tests et si les tests réussissent, déploie automatiquement.

### 1. Création d'une séquence d'un ensemble de jobs

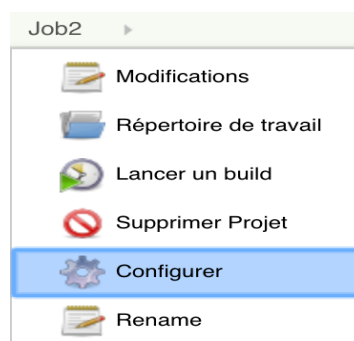
Créer trois projets « Job1 », « Job2 », et « Job3 ». Ici vous allez créer des jobs simples (de type free style et non pas maven) qui affiche une chaîne de caractère.

Job1 lors du build, il va afficher la chaîne « Job1 fini avec succès ».



Faites pareil pour Job2 et Job3

Une fois les trois jobs sont créés, retourner à Job2. Vous allez le configurer pour qu'il se lance automatiquement après la fin du build du Job1. Pour cela, cliquez sur « Configurer ».



Allez dans « Ce qui déclenche le build », et cochez « Construire après le build sur d'autres projets » et indiquez *Job1*. On veut que le build du Job2 soit déclenché juste après le build de Job1. On veut que ça soit le cas que si le build de Job1 réussit. Cochez l'option correspondante.

### Ce qui déclenche le build

☐ Déclencher les builds à distance (Par exemple, à partir de scripts)

☒ Construire après le build sur d'autres projets

Projet à surveiller

☒ Déclencher que si la construction est stable

☐ Déclencher même si la construction est instable

☐ Déclencher même si la construction échoue

☐ Construire périodiquement

☐ Scrutation de l'outil de gestion de version

Faites pareil pour Job3 qui doit être déclenché juste après Job2.

Lancez le build de Job1. Vous allez constater que Job2 et Job3 seront consécutivement déclenchés. Vous pouvez par exemple aller dans Job3 pour voir le build. Vous remarquerez que le Job3 a été lancé suite au build du Job2, qui à son tour a été lancé par Job1 :

## Construction #1 (9 nov. 2021 à 14...

 [Ajouter une description](#)



Aucun changement.



Lancé par le projet amont [Job2](#) avec le numéro de construction [2](#)  
Originellement lancé par:

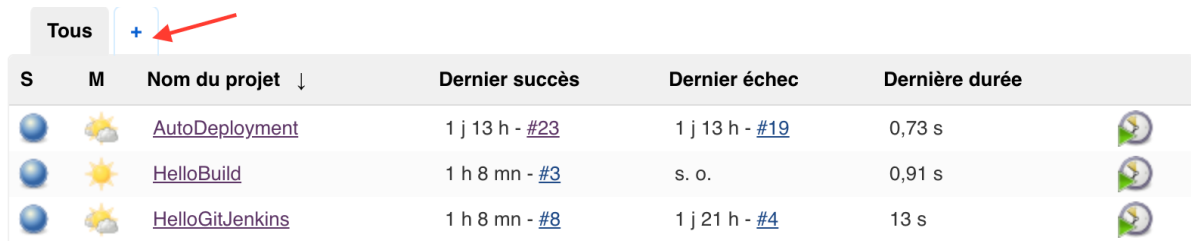
- Lancé par le projet amont [Job1](#) avec le numéro de construction [4](#)  
Originellement lancé par:
  - Lancé par l'utilisateur [Administrateur](#)

Concrètement, Job1 serait par exemple déclenché suite à des modifications dans Git. Ici il a été déclenché manuellement par l'administrateur.

## 2. Création de vues de pipeline :

Afin de pouvoir visualiser graphiquement la chaîne des jobs créés, vous allez utiliser le plugin « Delivery Pipeline ». Ajoutez-le.

Retourner à Jenkins, cliquez sur + pour créer une vue.



S	M	Nom du projet ↓	Dernier succès	Dernier échec	Dernière durée	
		<a href="#">AutoDeployment</a>	1 j 13 h - <a href="#">#23</a>	1 j 13 h - <a href="#">#19</a>	0,73 s	
		<a href="#">HelloBuild</a>	1 h 8 mn - <a href="#">#3</a>	s. o.	0,91 s	
		<a href="#">HelloGitJenkins</a>	1 h 8 mn - <a href="#">#8</a>	1 j 21 h - <a href="#">#4</a>	13 s	

Choisir « Delivery Pipeline View » et donnez un nom à votre vue. Ici on l'appelle AutoDeliveryPipeline.

Nom de la vue

AutoDeploymentPipeline

☒ **Delivery Pipeline View**

Continuous Delivery pipelines, perfect for visualization on information radiators. Shows one or more delivery pipeline instances, based on traditional Jenkins jobs with upstream/downstream dependencies.

☐ **Delivery Pipeline View for Jenkins Pipelines**

Continuous Delivery pipelines, perfect for visualization on information radiators. Shows one or more delivery pipeline instances, based on Jenkins pipelines (created using the Pipeline or Workflow plugin).

☐ **Ma vue**

Cette vue affiche automatiquement tous les jobs auxquels l'utilisateur a accès.

☐ **Vue liste**

Montre les jobs dans une simple liste. Vous pouvez choisir les jobs à afficher dans chaque vue.

OK

Une fois validé, cochez les trois options comme montré sur la figure suivante pour permettre de lancer des builds depuis le pipeline, permettre de relancer le build de job partiellement sans relancer toute la chaîne, et aussi pour pouvoir visualiser le temps des différents jobs.

Number of pipeline instances per pipeline	3	?
Display aggregated pipeline for each pipeline	<input type="checkbox"/>	?
Display aggregated changelog in aggregated view	<input type="checkbox"/>	?
Group aggregated changelog by regular expression		?
Number of columns	1	?
Sorting	None	?
Max number of pipelines	-1	?
Update interval	2	?
Enable start of new pipeline build	<input checked="" type="checkbox"/>	?
Enable manual triggers	<input type="checkbox"/>	?
Enable rebuild	<input checked="" type="checkbox"/>	?
Show avatars	<input type="checkbox"/>	?
Show commit messages	<input type="checkbox"/>	?
Show job description	<input type="checkbox"/>	?
Show job promotions	<input type="checkbox"/>	?
Show test results	<input type="checkbox"/>	?
Show static analysis results	<input type="checkbox"/>	?
Show total build time	<input checked="" type="checkbox"/>	?
Use relative links for easier navigation	<input type="checkbox"/>	?
Use direct link to console log	<input type="checkbox"/>	?

Ensuite, tout en bas, ajoutez des composants (Components). Chaque composant va correspondre à un job. Vous allez indiquer le premier Job de la chaîne que vous avez créé.

#### Pipelines

Components

Ajouter

Regular Expression

Ajouter

OK

Appliquer

Donnez un nom à votre composant et indiquez que le premier Job est Job1.

#### Pipelines

Components

Component  
Name

Build1

Initial Job

Job1

Final Job (optional)

Show upstream

☐

Supprimer

Ajouter

Regular Expression

Ajouter

OK

Appliquer

Après validation, vous obtenez l'historique des builds qui ont été lancés. Vous pouvez relancer le build de la séquence depuis la vue (voir la flèche rouge). Aussi, vous pouvez lancer le build des jobs seuls (voir flèche bleue).



### 3. Exercice :

Précédemment, vous avez créé des jobs pour un build automatique depuis Git et un pour le déploiement. Créez un pipeline de ces deux jobs.