

Jenkins et Git

Vous avez vu précédemment comment définir des jobs de build en utilisant les codes sources localement. Comme vous le savez, dans un projet informatique, il est indispensable d'utiliser un outil de versionning, comme Git. Il est donc essentiel d'associer l'outil d'intégration utilisé avec le dépôt correspondant. Le but de cette partie est donc de vous montrer comment associer Git à Jenkins.

1. Git et Jenkins

Vous allez ajouter d'abord le plugin Git à Jenkins. Allez dans Administrer Jenkins > Gestion des Plugins > Disponibles. Cherchez le plugin Git :

The screenshot shows the Jenkins 'Available Plugins' page. The search bar at the top contains 'Git'. The 'Git' plugin is checked for installation. The table lists various plugins with their names, versions, and release dates. The 'Git' plugin is the first one listed and is checked. Other plugins include 'Git client', 'GIT server', 'GitHub API', 'GitHub', 'GitHub Branch Source', 'Pipeline: GitHub Groovy Libraries', 'OkHttp', and 'Git Pipeline for Blue Ocean'.

Install	Name	Version	Released
<input checked="" type="checkbox"/>	Git git Source Code Management This plugin integrates Git with Jenkins.	4.9.0	18 j ago
<input type="checkbox"/>	Git client api-plugin Library plugins (for use by other plugins) Utility plugin for Git support in Jenkins	3.10.0	26 j ago
<input type="checkbox"/>	GIT server api-plugin git Library plugins (for use by other plugins) Allows Jenkins to act as a Git server.	1.10	3 mo. 21 j ago
<input type="checkbox"/>	GitHub API api-plugin github Library plugins (for use by other plugins) This plugin provides GitHub API for other plugins.	1.133	1 mo. 2 j ago
<input type="checkbox"/>	GitHub External Site/Tool Integrations github This plugin integrates GitHub to Jenkins.	1.34.1	1 mo. 24 j ago
<input type="checkbox"/>	GitHub Branch Source github pipeline Source Code Management Multibranch projects and organization folders from GitHub. Maintained by CloudBees, Inc.	2.11.3	28 j ago
<input type="checkbox"/>	Pipeline: GitHub Groovy Libraries github pipeline Allows Pipeline Groovy libraries to be loaded on the fly from GitHub.	1.0	4 an. 9 mo. ago
<input type="checkbox"/>	OkHttp This plugin provides OkHttp for other plugins.	3.14.9	1 an. 4 mo. ago
<input type="checkbox"/>	Git Pipeline for Blue Ocean External Site/Tool Integrations User Interface BlueOcean Git SCM pipeline creator	1.25.1	5 j 7 h ago

Buttons at the bottom: Install without restart, Download now and install after restart, Update information obtained: 1 j 4 h ago, Vérifier maintenant

Cliquez sur « Installer without restart ». L'installation s'effectue. Cochez « Redémarrer Jenkins quand l'installation est terminée et qu'aucun job n'est en cours ».

Une fois l'ajout du plugin Git à Jenkins est effectué, créez un Job (projet Maven) comme fait précédemment sur Jenkins. Ici on l'appelle *SimpleMavenGitWebProject*.

Vous allez configurer Jenkins avec Git. Autrement dit, Jenkins va considérer les codes sources disponibles sur votre repository et non pas sur votre machine. Nous allons utiliser ici le projet Web précédemment développé et mis sur le repository GitHub.

Dans Gestion de code source, choisir Git, puis renseignez l'URL de votre repository github :

Gestion de code source

☐ Aucune
☒ Git

Repositories

Repository URL

Credentials
- aucun -

Branches to build

Branch Specifier (blank for 'any')

Puis plus bas dans « *Ce qui déclenche le build* », cochez « *Scrutation de l'outil de gestion de version* ». Dans le champ « Planning », il faut écrire l'expression permettant à Jenkins de savoir comment il va vérifier les changements dans Git. Vous pouvez cliquer sur « ? » de planning pour voir comment écrire une expression. Ici par exemple, on veut que ça soit chaque minute (*****):

Ce qui déclenche le build

☐ Lance un build à chaque fois qu'une dépendance SNAPSHOT est construite
☐ Déclencher les builds à distance (Par exemple, à partir de scripts)
☐ Construire après le build sur d'autres projets
☐ Construire périodiquement
☒ Scrutation de l'outil de gestion de version

Planning

⚠ Voulez-vous vraiment dire "chaque minute" avec l'expression "*****"? Peut-être voulez-vous dire "H*****"?
Aaurait été lancé à mercredi 27 octobre 2021 à 09:58:41 heure d'été d'Europe centrale; prochaine exécution à mercredi 27 octobre 2021 à 09:58:41 heure d'été d'Europe centrale.

Plus bas dans « Build », on veut à chaque fois qu'il y a un changement dans Git, compiler le projet que Jenkins aurait cherché de Git.

Remarque :

Ici, on ne met plus le chemin absolu du fichier pom.xml de votre projet en local comme vous l'aviez fait précédemment. En effet, Jenkins va récupérer le projet disponible sur GitHub et donc le fichier pom.xml va être présent dans le répertoire demoWebApp (nom du projet présent dans le repository GitHub) dans le workspace de Jenkins. Sauvegardez.


Build

POM Racine
demoWebApp/pom.xml

Goals et options
clean compile

Jenkins va scruter votre dépôt toute les minutes. Ne lancez pas manuellement le projet Jenkins.

Faites des modifications à votre projet (par exemple dans votre servlet ajoutez un message) et poussez-les sur votre dépôt. Retournez à Jenkins et cliquez sur votre job « SimpleMavenGitWebProject » et ne faites rien. Juste observez le lancement d'un build automatiquement, suite à un changement dans votre repository Git. Vérifier l'historique des builds:

 **Jenkins**

rechercher

Tableau de bord > SimpleMavenGitWebProject >

Retour au tableau de bord

État

Modifications

Répertoire de travail

Lancer un build

Configurer

Supprimer Maven project

Modules

Log du dernier accès à Git

Rename

Historique des builds

tendance

find

#4

27 oct. 2021 10:22

#3

27 oct. 2021 10:12

#2

27 oct. 2021 10:08

#1

27 oct. 2021 10:06

Maven project SimpleMavenGitWebProject

Espace de travail

Changements récents

Liens permanents

- Dernier build (#4), il y a 2 mn 29 s
- Dernier build stable (#4), il y a 2 mn 29 s
- Dernier build avec succès (#4), il y a 2 mn 29 s
- Dernier build en échec (#2), il y a 18 mn
- Dernier build non réussi (#2), il y a 18 mn
- Last completed build (#4), il y a 2 mn 29 s

Regardez le log. Vous pouvez voir les changements qui ont été poussés (le fichier concerné et le commit réalisé avec son auteur) en cliquant sur *details*.

Construction #4 (27 oct. 2021 à 10:22:17)

 [Ajouter une description](#)



Changes

1. modification de la servlet ([details](#) / [githubweb](#))



[Lancé par un changement dans la base de code](#)



Revision: 320d20fcc8cc3737acdadb8a8d4f7367db1ee5

Repository: <https://github.com/nggel/DemoGit.git>

- refs/remotes/origin/master

Construction du module

 [demoWebApp Maven Webapp](#) 1,6 s