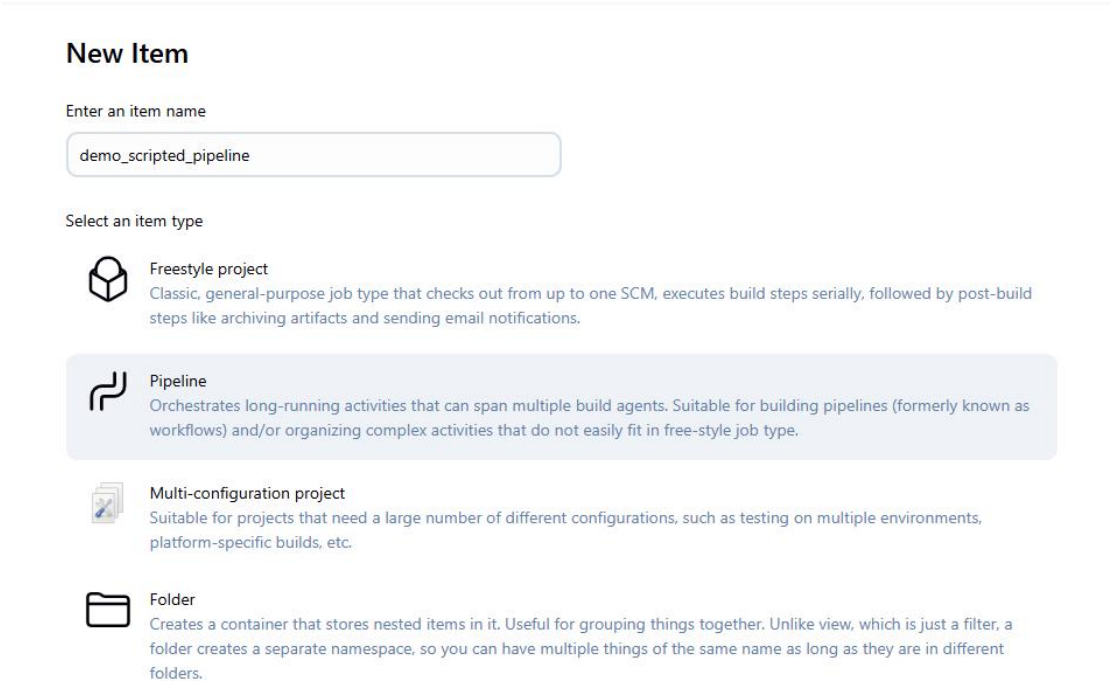
Jenkins Scripted Pipeline (Groovy)
--> For scripted pipeline, we will be using Groovy language
--> Flexibility
--> Customization
--> Reusable components

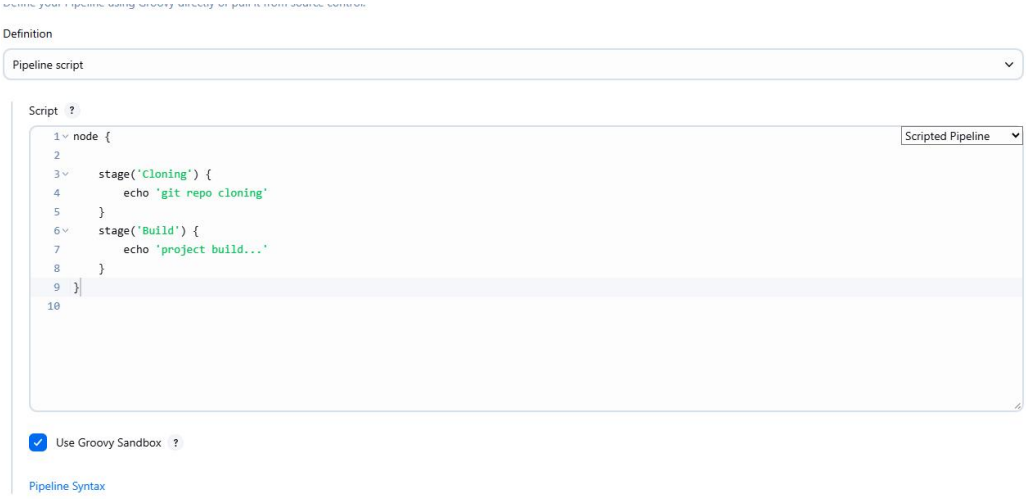
Syntax:

```
node {
        stage('git clone'){
                echo 'git cloning'
        }

        stage('build'){
                echo 'build...'
        }

}
```

```
-rwxr-x--- 1 ubuntu ubuntu   1908 Jun  5 17:49 version.sh
ubuntu@ip-172-31-11-116:~/apache-tomcat-11.0.8/bin$ sh startup.sh
Using CATALINA_BASE:   /home/ubuntu/apache-tomcat-11.0.8
Using CATALINA_HOME:   /home/ubuntu/apache-tomcat-11.0.8
Using CATALINA_TMPDIR: /home/ubuntu/apache-tomcat-11.0.8/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /home/ubuntu/apache-tomcat-11.0.8/bin/bootstrap.jar:/home
Using CATALINA_OPTS:
Tomcat started.
ubuntu@ip-172-31-11-116:~/apache-tomcat-11.0.8/bin$
```




```
node {

   stage('Cloning') {
      echo 'git repo cloning'
   }
   stage('Build') {
      echo 'project build...'
   }
}
```

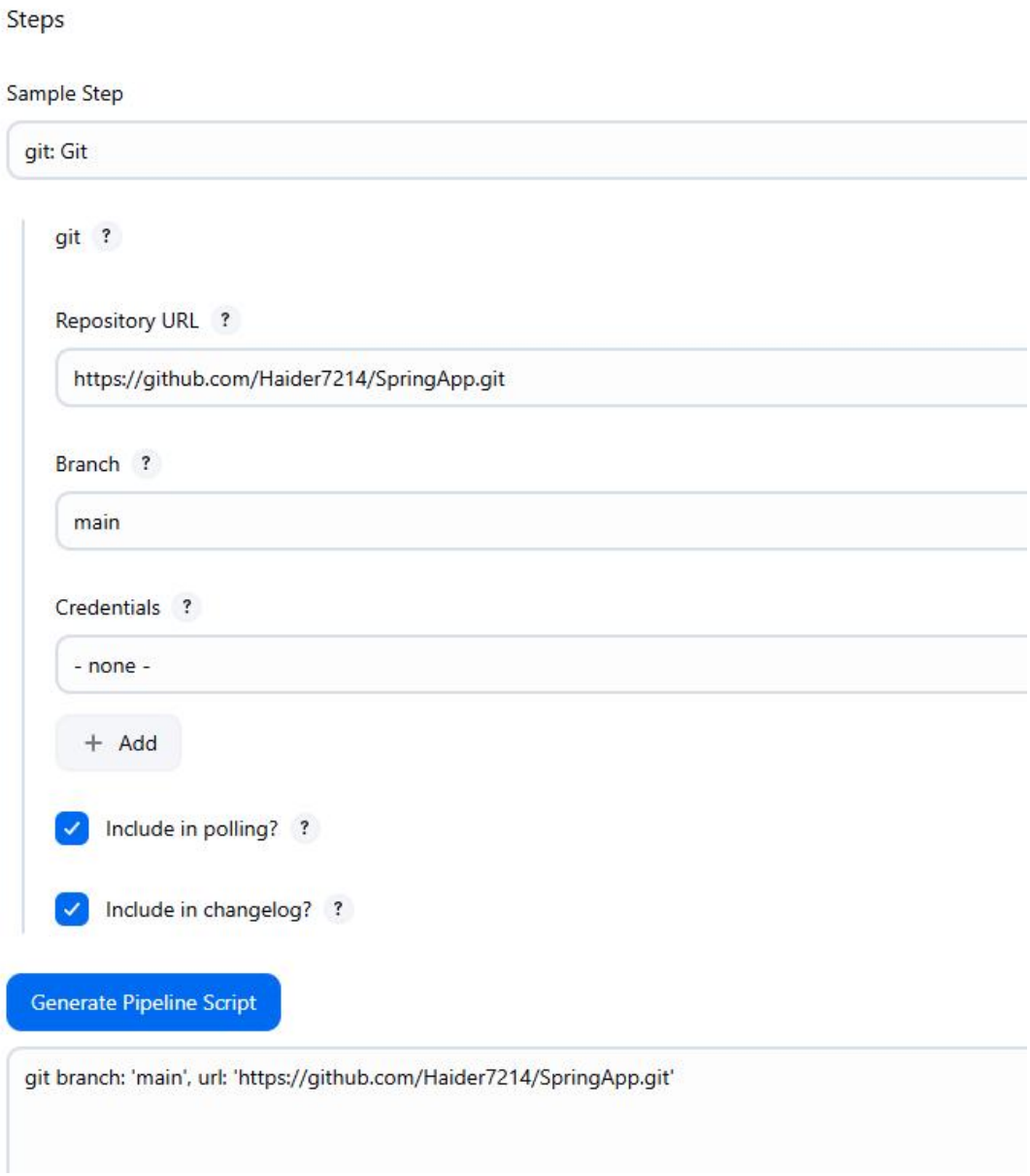


Apply and Save

Console Output

```
Started by user demo
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/demo_scripted_pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Cloning)
[Pipeline] echo
git repo cloning
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Build)
[Pipeline] echo
project build...
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Go back to Pipeline --> Configure



For Declarative it is different, for Scripted it is totally different
git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'

```
node {
   def mvnPath
   stage('git clone') {
      git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
   }
   stage('Maven Build') {
      def mvnHome = tool name:'maven:3.9.10', type='maven';
      mvnPath = "${mvnHome}/bin/mvn";
      sh "${mvnPath} clean package";
      echo 'maven build success';
   }
}
```

Define your Pipeline using Groovy directly or pull it from source control.

Definition

Pipeline script

Script ?

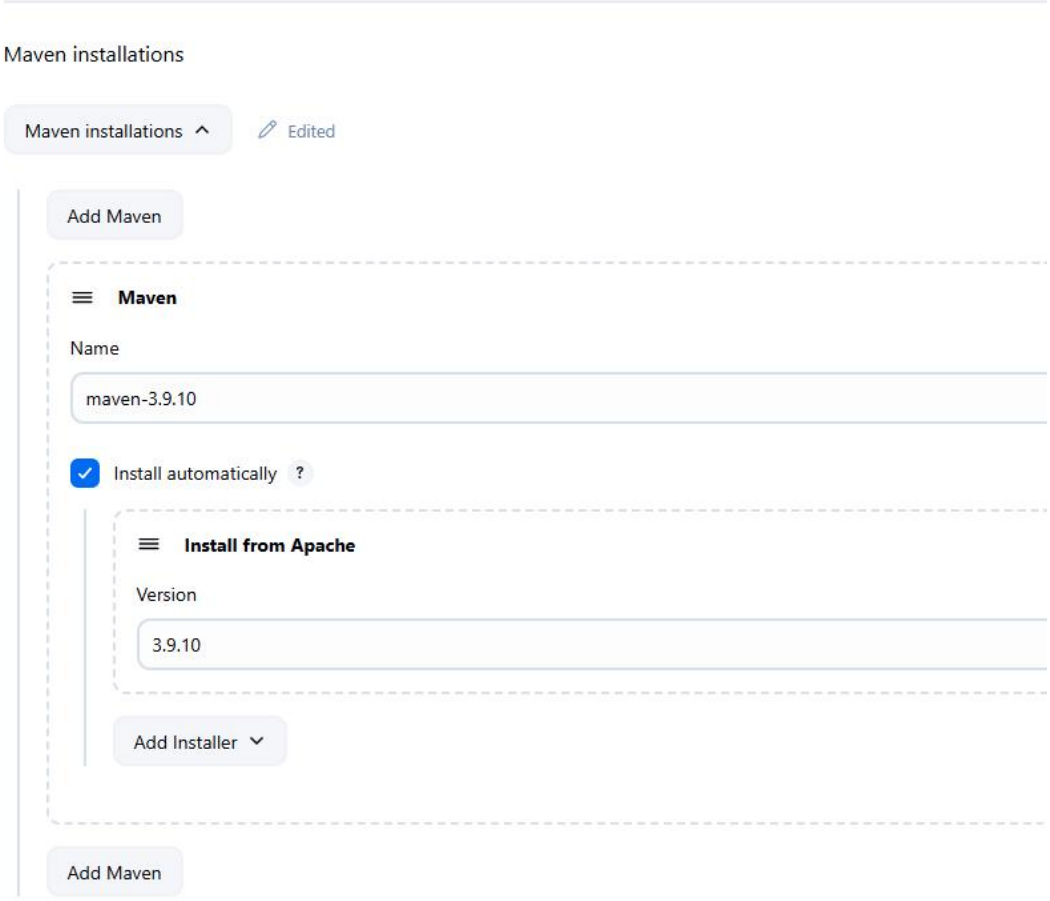
```
node {
    def mvnPath
    stage('git clone') {
        git branch: 'main', url: 'https://github.com/Haider7214/SpringApp.git'
    }
    stage('Maven Build') {
        def mvnHome = tool name:'maven:3.9.10', type='maven';
        mvnPath = "${mvnHome}/bin/mvn";
        sh "${mvnPath} clean package";
        echo 'maven build success';
    }
}
```

Use Groovy Sandbox ?

Pipeline Syntax

Apply and Save

Manage Jenkins --> Tools
That's the name we have given for Maven



Failed

```
        at PluginClassLoader for workflow-cps//com.cloudbees.groovy.cps.impl.FunctionCallBlock$ContinuationImpl.fixArg(FunctionCa
        at java.base/jdk.internal.reflect.DirectMethodHandleAccessor.invoke(DirectMethodHandleAccessor.java:103)
        at java.base/java.lang.reflect.Method.invoke(Method.java:580)
        at PluginClassLoader for workflow-cps//com.cloudbees.groovy.cps.impl.ContinuationPtr$ContinuationImpl.receive(Continuatio
        at PluginClassLoader for workflow-cps//com.cloudbees.groovy.cps.impl.ConstantBlock.eval(ConstantBlock.java:21)
        at PluginClassLoader for workflow-cps//com.cloudbees.groovy.cps.Next.step(Next.java:83)
        at PluginClassLoader for workflow-cps//com.cloudbees.groovy.cps.Continuable.run0(Continuable.java:147)
        at PluginClassLoader for workflow-cps//org.jenkinsci.plugins.workflow.cps.SandboxContinuable.access$001(SandboxContinuabl
        at PluginClassLoader for workflow-cps//org.jenkinsci.plugins.workflow.cps.SandboxContinuable.run0(SandboxContinuable.java
        at PluginClassLoader for workflow-cps//org.jenkinsci.plugins.workflow.cps.CpsThread.runNextChunk(CpsThread.java:180)
        at PluginClassLoader for workflow-cps//org.jenkinsci.plugins.workflow.cps.CpsThreadGroup.run(CpsThreadGroup.java:419)
        at PluginClassLoader for workflow-cps//org.jenkinsci.plugins.workflow.cps.CpsThreadGroup$2.call(CpsThreadGroup.java:327)
        at PluginClassLoader for workflow-cps//org.jenkinsci.plugins.workflow.cps.CpsThreadGroup$2.call(CpsThreadGroup.java:292)
        at PluginClassLoader for workflow-cps//org.jenkinsci.plugins.workflow.cps.CpsVmExecutorService.lambda$wrap$4(CpsVmExecut
        at java.base/java.util.concurrent.FutureTask.run(FutureTask.java:317)
        at hudson.remoting.SingleLaneExecutorService$1.run(SingleLaneExecutorService.java:139)
        at jenkins.util.ContextResettingExecutorService$1.run(ContextResettingExecutorService.java:28)
        at jenkins.security.ImpersonatingExecutorService$1.run(ImpersonatingExecutorService.java:68)
        at jenkins.util.ErrorLoggingExecutorService.lambda$wrap$0(ErrorLoggingExecutorService.java:51)
        at java.base/java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:572)
        at java.base/java.util.concurrent.FutureTask.run(FutureTask.java:317)
        at java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1144)
        at java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecutor.java:642)
        at PluginClassLoader for workflow-cps//org.jenkinsci.plugins.workflow.cps.CpsVmExecutorService$1.call(CpsVmExecutorServic
        at PluginClassLoader for workflow-cps//org.jenkinsci.plugins.workflow.cps.CpsVmExecutorService$1.call(CpsVmExecutorServic
        at org.codehaus.groovy.runtime.GroovyCategorySupport$ThreadCategoryInfo.use(GroovyCategorySupport.java:136)
        at org.codehaus.groovy.runtime.GroovyCategorySupport.use(GroovyCategorySupport.java:275)
        at PluginClassLoader for workflow-cps//org.jenkinsci.plugins.workflow.cps.CpsVmExecutorService.lambda$categoryThreadFacto
        at java.base/java.lang.Thread.run(Thread.java:1583)
Finished: FAILURE
```

I change the script

```
node {
   def mvnPath
   stage('git clone') {
      git branch: 'main', url: 'https://github.com/Haider7214/SpringBoot.git'
   }
   stage('Maven Build') {
      def mvnHome = tool name:'maven-3.9.10', type='maven';
      mvnPath = "${mvnHome}/bin/mvn";
      sh "${mvnPath} clean package";
      echo 'maven build success';
   }
}
```

Again Build failed

```
node {
   def mvnPath

   stage('git clone') {
      git branch: 'main', url: 'https://github.com/Haider7214/SpringBoot.git'
   }
   stage('Maven Build') {
      def mvnHome=tool name:'maven-3.9.10', type:'maven';
      mvnPath="${mvnHome}/bin/mvn";
      sh "${mvnPath} clean package";
      echo 'maven build success';
   }
}
```

This time build was successful

```
Progress (1): 357/465 kB
Progress (1): 374/465 kB
Progress (1): 390/465 kB
Progress (1): 406/465 kB
Progress (1): 423/465 kB
Progress (1): 439/465 kB
Progress (1): 456/465 kB
Progress (1): 465 kB

Downloaded from central: https://repo.maven.apache.org/maven2/org/springframework/bo
[INFO] Replacing main artifact /var/lib/jenkins/workspace/demo_scripted_pipeline/tar
[INFO] The original artifact has been renamed to /var/lib/jenkins/workspace/demo_scr
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  13.591 s
[INFO] Finished at: 2025-07-05T17:05:47Z
[INFO] ------------------------------------------------------------------------
[Pipeline] echo
maven build success
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Jenkins Scripted (Groovy) Pipeline with Git + Maven

```
node {
   def mvnPath

   stage('git clone') {
      git branch: 'main', url: 'https://github.com/Haider7214/SpringBoot.git'
   }
   stage('Maven Build') {
      def mvnHome=tool name:'maven-3.9.10', type:'maven';
      mvnPath="${mvnHome}/bin/mvn";
      sh "${mvnPath} clean package";
      echo 'maven build success';
   }
}
```

Multi Branch Pipeline in Jenkins

In Git repo, there will be multiple branches -> main branch, master, developer, release, feature

During git cloning, we got to specify which branch we are cloning from. Same Jenkins pipeline cannot work for all branches then if we are specifying the branch name explicitly

--> Creating different Jenkins pipeline for every branch would be difficult

--> We can create one common pipeline and build the code from multiple branches at a time with the help of “Multi-branch pipeline” concept

--> When we create Multi-branch pipeline, it will scan all the branches in given Github repo and execute pipeline for all the branches

Whichever branch only has the code change, only that branch will be built

Note: Whenever we run Multi-branch pipeline for second time, it will execute the pipeline for which code changes/commits have been done.