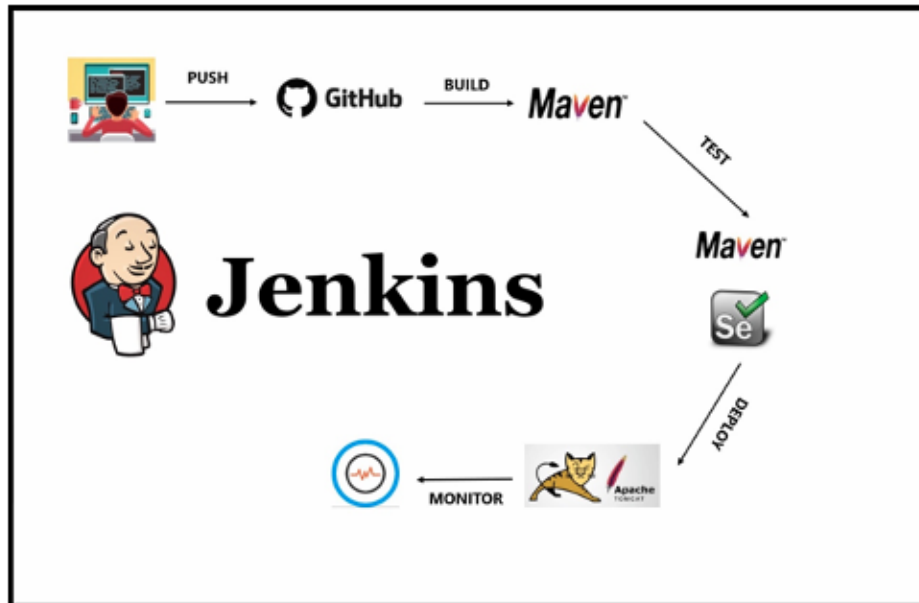


Creación de Pipeline en distintos lenguajes en Jenkins



Jesús Joel Meneses Meneses
2º DAW A
DPL---Despliegue de Aplicaciones Web

Índice

1. Requisitos básicos

2. Creación de Pipelines

- ### 2.1 Java ###

- ### 2.2 Node.js ###
- ### 2.3 Ruby ###
- ### 2.4 Python ###
- ### 2.5 PHP ###
- ### 2.6 GO ###

1. Requisitos básicos

- Tener jenkins instalado
- Tener los contenedores de docker jenkins parados, para eso utilizamos el metodo cortar por lo sano paramos todos los contenedores docker y así no tenemos ningún tipo de problema con el comando

Si tenemos problemas de permisos ejecutaremos el siguiente comando

```
joel@joel-joel:~$ sudo chmod 666 /var/run/docker.sock
```

Luego los paramos

```
joel@joel-joel:~$ sudo docker stop $(docker ps -a -q)
6bccbebe6dca
6e63079b0e10
18677fcfe0de
d4a50b285c13
14e42e37afbf
1ea7fe794488
c00d994cd6a1
8b8aba04fd06
11754f536b17
e18a2aec2ce7
3f7a1e58075c
```

Posteriormente reiniciamos el servicio apache y jenkins

Luego reiniciamos nuestra máquina virtual

```
joel@joel-joel:~$ sudo systemctl restart apache2.service
joel@joel-joel:~$ sudo systemctl reload apache2.service
```

2. Creación del Pipelines

Accedemos a jenkins



Welcome to Jenkins!

joel

Sign in

☐ Keep me signed in

Se mostrarán los diferentes pasos para la creación de los tipos de lenguajes

2.1. Java

Creación de Pipeline en Java

Clickeamos en nueva tarea

← → ↻ www.jenkins.joel.com:8080 ☆

Jenkins 🔍 búsqueda ⓘ 1 👤 joel 🚪 Desconectar

Panel de Control ▶

- Nueva Tarea
- Personas
- Historial de trabajos
- Administrar Jenkins
- Mis vistas
- Lockable Resources
- New View

Trabajos en la cola ^

No hay trabajos en la cola

Estado del ejecutor de construcciones ^

1 Inactivo

¡Bienvenido a Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job →

Set up a distributed build

Set up an agent →

Configure a cloud →

Learn more about distributed builds ↗

[añadir descripción](#)

Definimos el nombre del pipeline


← → ↻ www.jenkins.joel.com:8080/view/all/newJob ☆ ⓘ 1 joel Desconectar

Panel de Control ▸ Todo ▸


Enter an item name

Jenkins Java


» Required field

**Crear un proyecto de estilo libre**

Esta es la característica principal de Jenkins, la de ejecutar el proyecto combinando cualquier tipo de repositorio de software (SCM) con cualquier modo de construcción o ejecución (make, ant, mvn, rake, script ...). Por tanto se podrá tanto compilar y empaquetar software, como ejecutar cualquier proceso que requiera monitorización.

**Pipeline**

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

**Crear un proyecto multi-configuración**

Adecuado para proyectos que requieran un gran número de configuraciones diferentes, como testear en multiples entornos, ejecutar mas concretas, etc.

OK

En la opcion de pipeline, nos vamos a script y pegamos el script de la siguiente imagen

← → ↻ www.jenkins.joel.com:8080/job/Jenkins Java/configure ☆ ⓘ

Panel de Control ▸ Jenkins Java ▸

General Build Triggers Advanced Project Options **Pipeline**

Definition

Pipeline script

Script

```
1 pipeline {
2   agent { docker { image 'maven:3.8.4-openjdk-11-slim' } }
3   stages {
4     stage('build') {
5       steps {
6         sh 'mvn --version'
7       }
8     }
9   }
10 }
```

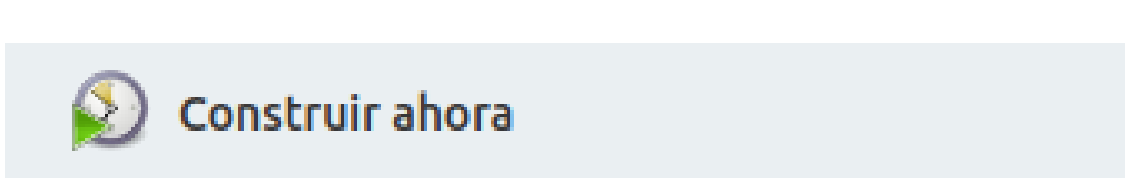
try sample Pipeline...

☒ Use Groovy Sandbox



[Pipeline Syntax](#)


Guardar Apply


Una vez guardado, pulsamos la opción construir ahora



Nos dara un error en el proceso

 **Historia de tareas** **Tendencia** 

 Filter builds...


 **#1** **23 ene. 2022 16:19**





← → ↻

www.jenkins.joel.com:8080


☆


🔒


 **Jenkins**


   joel  Desconectar


Panel de Control ▶


 Nueva Tarea


 Personas

 Historial de trabajos

 Administrar Jenkins

 Mis vistas

 Lockable Resources

 New View

Trabajos en la cola

No hay trabajos en la cola



Estado del ejecutor de construcciones

1 Inactivo

2 Inactivo


añadir descripción


Todo +


S	W	Nombre	Último Éxito	Último Fallo	Última Duración
		Jenkins Java	N/D	42 Seg - #1	0,83 Seg

Icono: S M L

Guía de iconos

 Atom feed para todos

 Atom feed para fallas

 Atom feed para los más recientes

Este es el error que sale por consola

Salida de consola

Lanzada por el usuario [joel](#)

org.codehaus.groovy.control.MultipleCompilationErrorsException: startup failed:

WorkflowScript: 2: Invalid agent type "docker" specified. Must be one of [any, label, none] @ line 2, column 13.
agent { docker { image 'maven:3.8.4-openjdk-11-slim' } }
 ^

1 error

```
at org.codehaus.groovy.control.ErrorCollector.failIfErrors(ErrorCollector.java:310)
at org.codehaus.groovy.control.CompilationUnit.applyToPrimaryClassNodes(CompilationUnit.java:1085)
at org.codehaus.groovy.control.CompilationUnit.doPhaseOperation(CompilationUnit.java:603)
at org.codehaus.groovy.control.CompilationUnit.processPhaseOperations(CompilationUnit.java:581)
at org.codehaus.groovy.control.CompilationUnit.compile(CompilationUnit.java:558)
at groovy.lang.GroovyClassLoader.doParseClass(GroovyClassLoader.java:298)
at groovy.lang.GroovyClassLoader.parseClass(GroovyClassLoader.java:268)
at groovy.lang.GroovyShell.parseClass(GroovyShell.java:688)
at groovy.lang.GroovyShell.parse(GroovyShell.java:700)
at org.jenkinsci.plugins.workflow.cps.CpsGroovyShell.doParse(CpsGroovyShell.java:142)
at org.jenkinsci.plugins.workflow.cps.CpsGroovyShell.reparse(CpsGroovyShell.java:127)
at org.jenkinsci.plugins.workflow.cps.CpsFlowExecution.parseScript(CpsFlowExecution.java:571)
at org.jenkinsci.plugins.workflow.cps.CpsFlowExecution.start(CpsFlowExecution.java:523)
at org.jenkinsci.plugins.workflow.job.WorkflowRun.run(WorkflowRun.java:334)
at hudson.model.ResourceController.execute(ResourceController.java:99)
at hudson.model.Executor.run(Executor.java:432)
```

Finished: FAILURE

Solución del error

Nos vamos al panel de control de jenkins, administrar jenkins

Panel de Control ▶



Nueva Tarea



Personas



Historial de trabajos



Administrar Jenkins



Mis vistas



Lockable Resources



New View

Nos vamos a la opción de administrar plugins

Panel de Control

- Nueva Tarea
- Personas
- Historial de trabajos
- Administrar Jenkins**
- Mis vistas
- Lockable Resources
- New View

Trabajos en la cola

No hay trabajos en la cola

Estado del ejecutor de construcciones

- Inactivo
- Inactivo

Administrar Jenkins

Parece que la configuración de proxy inverso está mal [Más información](#) [Ocultar](#)

Building on the built-in node can be a security issue. You should set up distributed builds. See [the documentation](#). [Set up agent](#) [Set up cloud](#) [Dismiss](#)

System Configuration

- Configurar el Sistema**
Configurar variables globales y rutas.
- Global Tool Configuration**
Configure tools, their locations and automatic installers.
- Administrar Plugins**
Añadir, borrar, desactivar y activar plugins que extienden la funcionalidad de Jenkins. Existen actualizaciones disponibles
- Administrar Nodos**
Añadir, borrar, gestionar y monitorizar los nodos sobre los que Jenkins ejecuta tareas.

Nos vamos a la pestaña todos los plugins y en el cuadro de búsqueda introducimos docker, y seleccionamos las siguientes opciones

Panel de Control

- Volver al Panel de control
- Administrar Jenkins**

Buscar: docker

Actualizaciones disponibles **Todos los plugins** Plugins instalados Configuración avanzada

Install	Name	Version	Released
<input checked="" type="checkbox"/>	Docker Cloud Providers Plugins para la administración de clusters y trabajos distribuidos docker This plugin integrates Jenkins with Docker	1.2.6	Hace 1 Mes 11 días
<input type="checkbox"/>	Docker Commons api-plugin docker Library plugins (for use by other plugins) Provides the common shared functionality for various Docker-related plugins.	1.18	Hace 1 Mes 13 días
<input checked="" type="checkbox"/>	Docker Pipeline Deployment DevOps docker pipeline Build and use Docker containers from pipelines.	1.27	Hace 1 Mes 13 días
<input type="checkbox"/>	Docker API api-plugin docker This plugin provides docker-java API for other plugins. ¡Este plugin está en adopción! Buscamos nuevos mantenedores. Visita nuestro sitio de la iniciativa Adopte un plugin para obtener más información.	3.1.5.2	Hace 1 Año 9 Mes
<input type="checkbox"/>	docker-build-step Plugins relacionados con la forma de ejecutar trabajos docker This plugin allows to add various docker commands to your job as build steps.	2.8	Hace 6 Mes 28 días

Pulsamos en descargar ahora e instalar después de reiniciar

Nos aparecerá una ventana similar a la siguiente, seleccionamos la opción de reiniciar jenkins cuando termine la instalación..

← → ↻ www.jenkins.joel.com:8080/updateCenter/ ☆ 🔒

Jenkins 🔍 búsqueda ⓘ ! 1 👤 joel 🚪 Desconectar

Panel de Control ▸ Update Center

🏠 Volver al Panel de control

⚙️ Administrar Jenkins

🧩 Administrar 'plugins'

Instalando/Actualizando plugins

Preparación

- Probando conectividad con Internet
- Probando conectividad con jenkins-ci.org
- Correcto

Authentication Tokens API	✅ Actualizado
Docker Commons	✅ Actualizado
Docker API	✅ Actualizado
Docker	✅ Actualizado
Docker Pipeline	✅ Actualizado
Loading plugin extensions	✅ Success

➡ [Volver al inicio de la página](#)
(puedes empezar a usar los plugins instalados inmediatamente)

➡ ☐ Reiniciar Jenkins cuando termine la instalación y no queden trabajos en ejecución

Nos aparecerá la siguiente ventana



Por favor espera hasta que Jenkins acabe de reiniciarse. ...

Su navegador recargará esta página cuando Jenkins esté listo.

Volvemos a pulsar la opción de construir ahora y se nos mostrará otro error distinto

Panel de Control

- Nueva Tarea
- Personas
- Historial de trabajos
- Administrar Jenkins
- Mis vistas
- Lockable Resources
- New View

Trabajos en la cola

No hay trabajos en la cola

Estado del ejecutor de construcciones

1 Inactivo

2 Inactivo

S	W	Nombre	Último Éxito	Último Fallo	Última Duración
		Jenkins Java	N/D	14 Min - #1	0,83 Seg

Icono: S M L

Guía de iconos

Atom feed para todos

Atom feed para fallas

Atom feed para los más recientes

Se muestra una ventana similar a la siguiente, de error de permisos

Salida de consola

```
Lanzada por el usuario joel
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/Jenkins Java
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] isUnix
[Pipeline] sh
+ docker inspect -f . maven:3.8.4-openjdk-11-slim

Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get
"http://%2Fvar%2Frun%2Fdocker.sock/v1.24/containers/maven:3.8.4-openjdk-11-slim/json": dial unix /var/run
/docker.sock: connect: permission denied
[Pipeline] withEnv
[Pipeline] {
[Pipeline] isUnix
[Pipeline] sh
+ docker pull maven:3.8.4-openjdk-11-slim
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post
"http://%2Fvar%2Frun%2Fdocker.sock/v1.24/images/create?fromImage=maven&tag=3.8.4-openjdk-11-slim": dial unix
/var/run/docker.sock: connect: permission denied
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
ERROR: script returned exit code 1
Finished: FAILURE
```

Añadimos permisos de usermod, con el comando...

```
sudo usermod -a -G docker jenkins
```

```
joel@joel-joel:~$ sudo usermod -a -G docker jenkins
```

Ahora nos vamos a la terminal para añadir el usuario de jenkins al grupo de sudoers

```
joel@joel-joel:~$ sudo usermod -a -G docker jenkins
joel@joel-joel:~$ sudo nano /etc/sudoers/

GNU nano 4.8 /etc/sudoers Modificado
#
# This file MUST be edited with the 'visudo' command as root.
#
# Please consider adding local content in /etc/sudoers.d/ instead of
# directly modifying this file.
#
# See the man page for details on how to write a sudoers file.
#
Defaults        env_reset
Defaults        mail_badpass
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
#root    ALL=(ALL:ALL) ALL
jenkins  ALL=(ALL) NOPASSWD: ALL
# Members of the admin group may gain root privileges
%admin    ALL=(ALL) ALL

# Allow members of group sudo to execute any command
%sudo    ALL=(ALL:ALL) ALL

# See sudoers(5) for more information on "#include" directives:

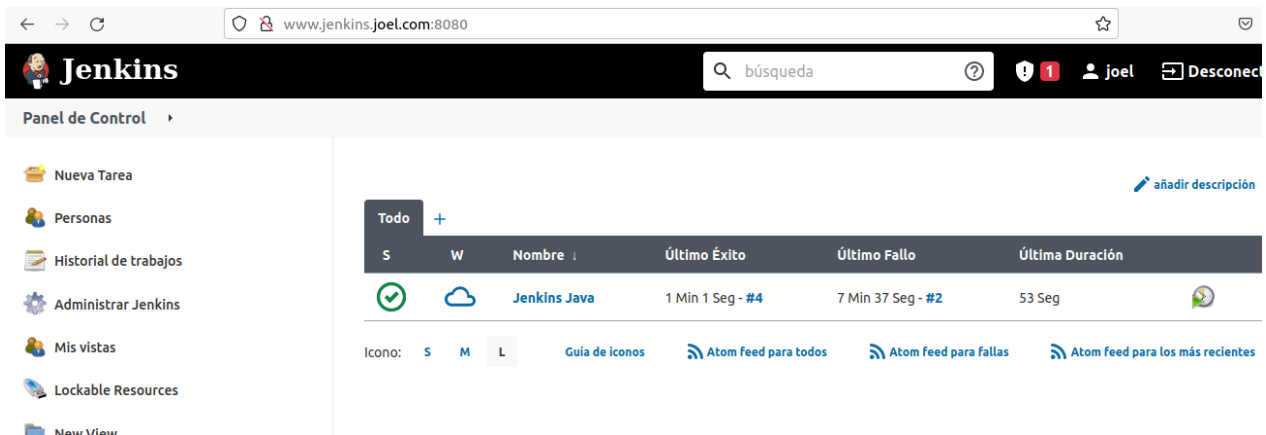
#include_dir /etc/sudoers.d

^G Ver ayuda  ^O Guardar   ^W Buscar    ^K Cortar Texto ^J Justificar  ^C Posición   M-U Deshacer  M-A Marcar texto
^X Salir      ^R Leer fich. ^\ Reemplazar ^U Pegar       ^T Ortografía  ^I Ir a línea  M-E Rehacer   M-G Copiar
```

Reiniciamos los servicios de apache y de jenkins

```
joel@joel-joel:~$ sudo systemctl restart apache2.service
joel@joel-joel:~$ sudo systemctl restart jenkins.service
```

Volvemos a ejecutar pipeline y vemos que se ejecuta

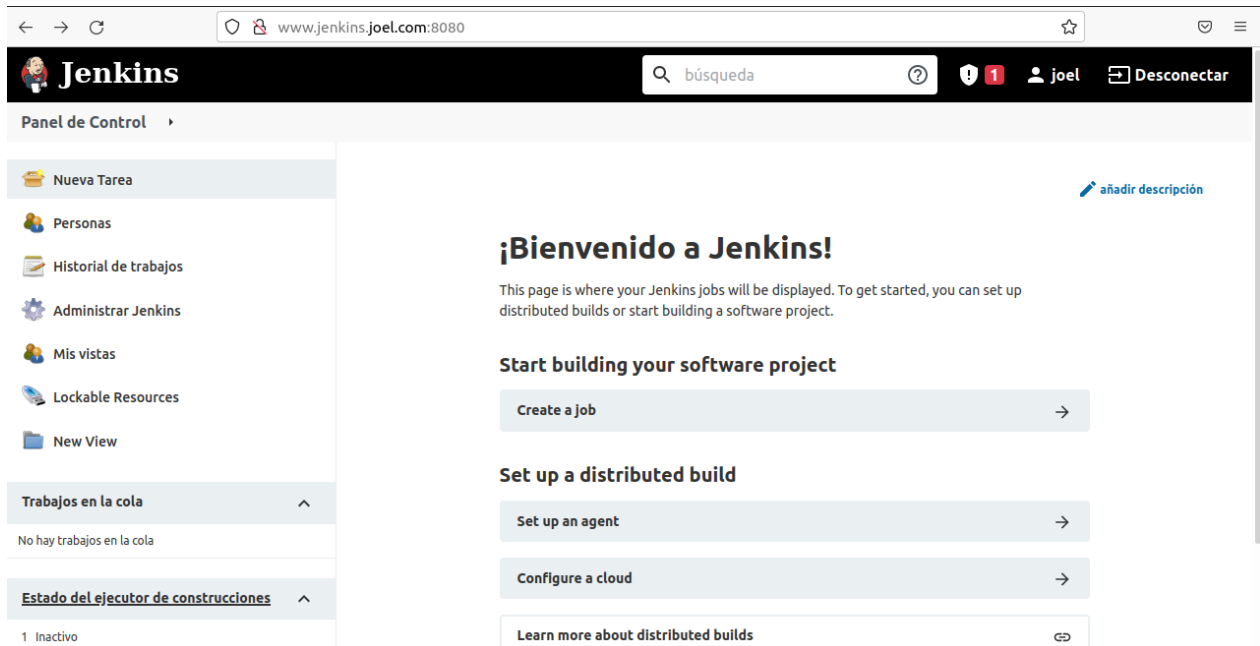


Ahora tenemos el pipeline de java instalado, hacemos el proceso con los siguientes lenguajes a instalar

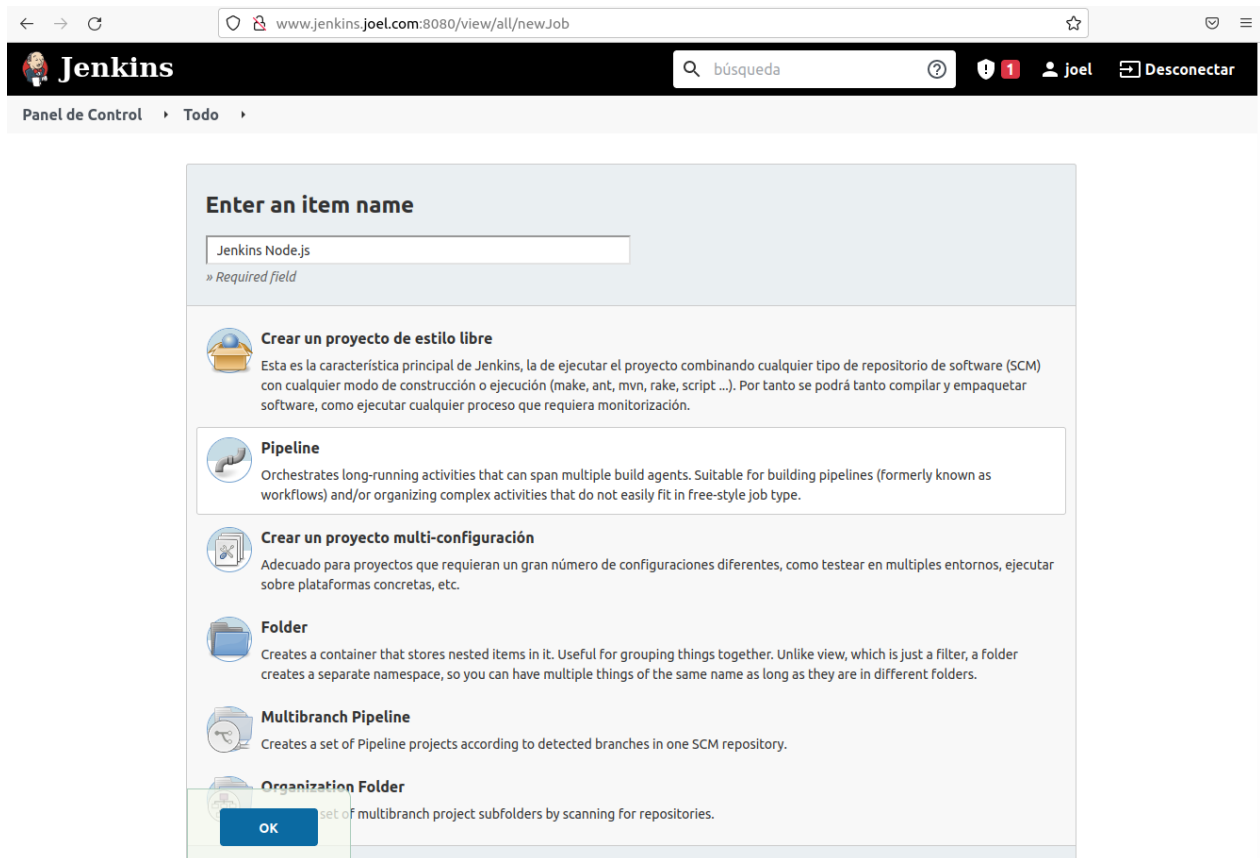
2.2. Node.js

Creación de Pipeline en Node.js

Clickeamos en nueva tarea



Definimos el nombre del pipeline



En la opción de pipeline, nos vamos a script y pegamos el script de la siguiente imagen

← → ↻ www.jenkins.joel.com:8080/job/Jenkins Node.js/configure ☆ 🔔 ☰

Panel de Control ▶ Jenkins Node.js ▶

General Build Triggers Advanced Project Options **Pipeline**

☐ Lanzar ejecuciones remotas (ejemplo: vue-cli scripts)

Advanced Project Options

Avanzado...

Pipeline

Definition

Pipeline script ▼

Script

```
1 pipeline {
2   agent { docker { image 'node:16.13.1-alpine' } }
3   stages {
4     stage('build') {
5       steps {
6         sh 'node --version'
7       }
8     }
9   }
10 }
```

try sample Pipeline... ▼

☒ Use Groovy Sandbox ?

Pipeline Syntax

Guardar Apply

Una vez guardado, pulsamos la opción construir ahora

⋮ ⚙ Jenkins Node.js N/D N/D N/D 🔄

Estado del ejecutor de construcciones ^

1 Inactivo

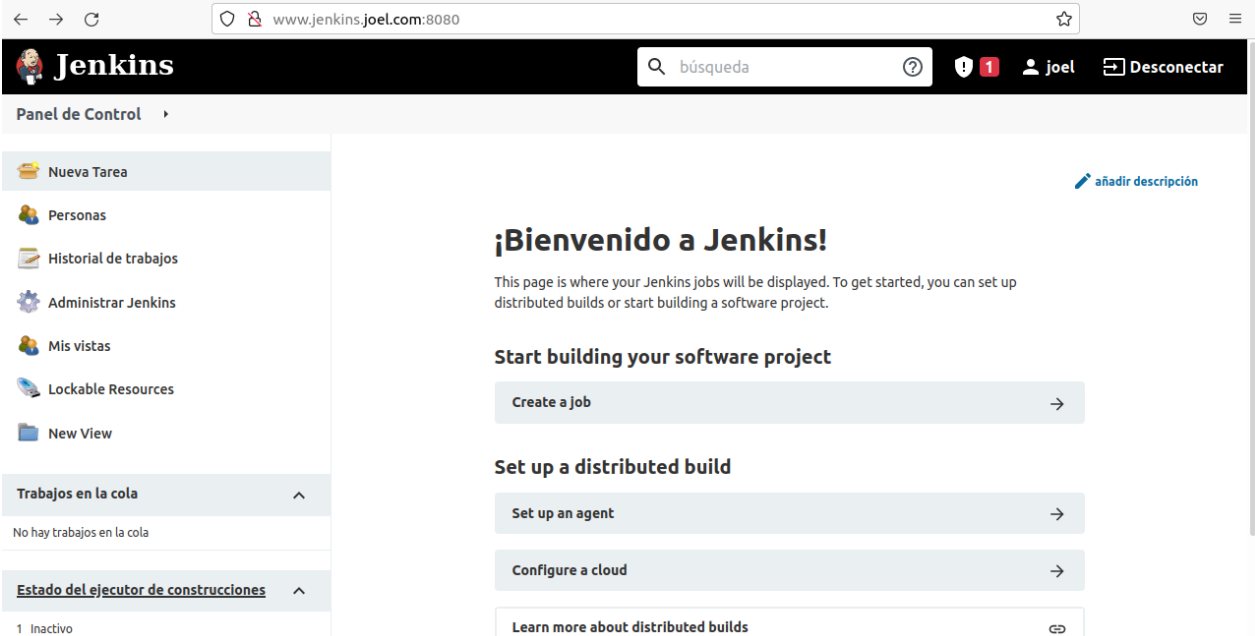
2 Jenkins Node.js #1 (part) ❌

Finalmente se ha instalado el pipeline

2.3. Ruby

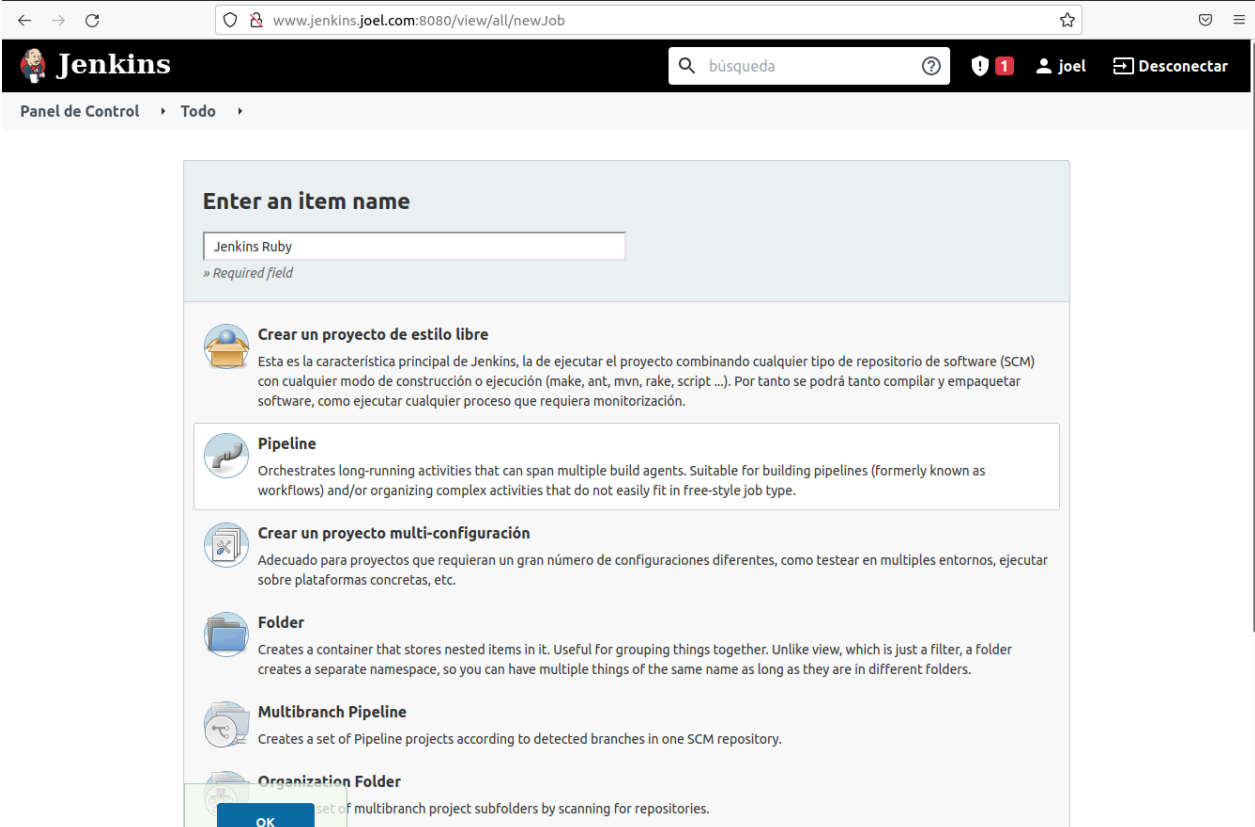
Creación de Pipeline en Ruby

Clickeamos en nueva tarea



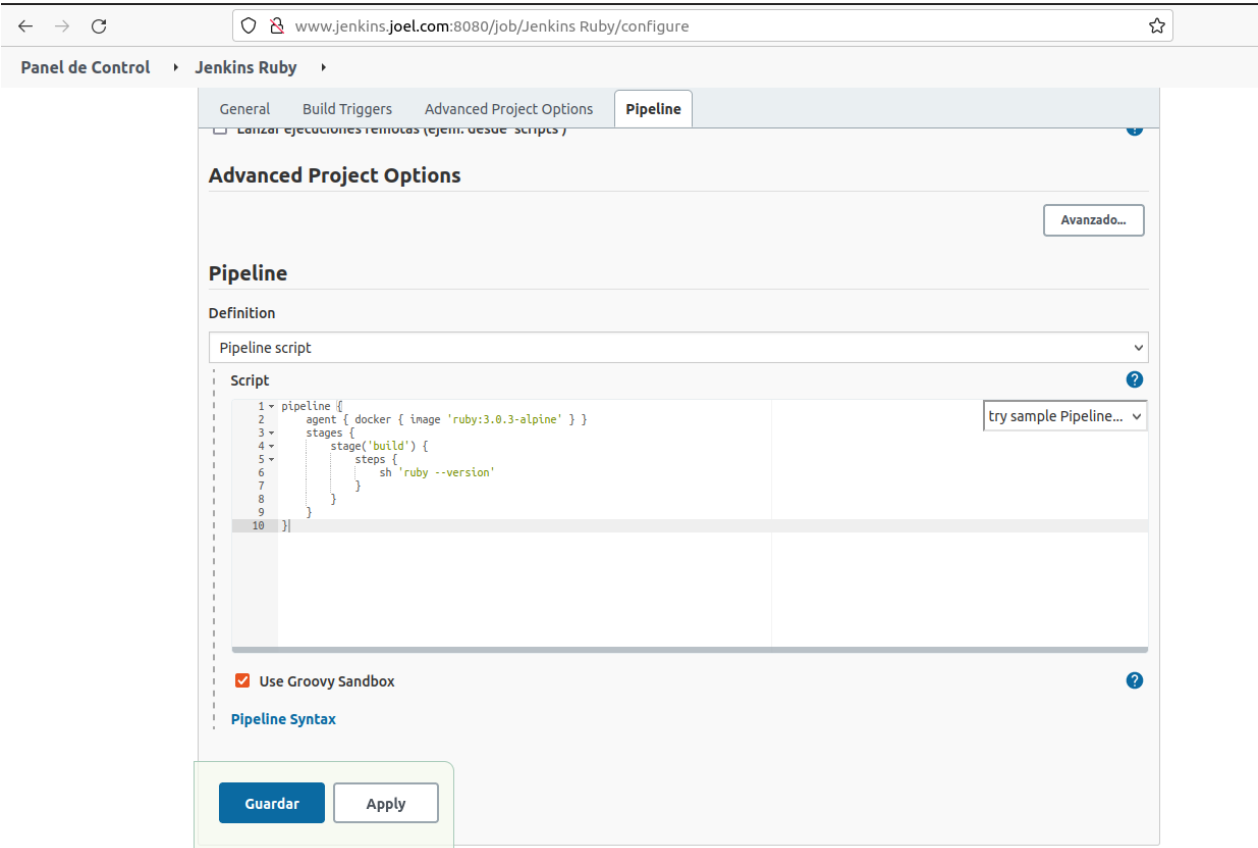
The screenshot shows the Jenkins dashboard at www.jenkins.joel.com:8080. The left sidebar contains a 'Panel de Control' with links to 'Nueva Tarea', 'Personas', 'Historial de trabajos', 'Administrar Jenkins', 'Mis vistas', 'Lockable Resources', and 'New View'. Below this, 'Trabajos en la cola' shows 'No hay trabajos en la cola', and 'Estado del ejecutor de construcciones' shows '1 Inactivo'. The main content area displays a welcome message '¡Bienvenido a Jenkins!' and instructions on how to get started. It includes a 'Start building your software project' section with a 'Create a job' button, and a 'Set up a distributed build' section with buttons for 'Set up an agent', 'Configure a cloud', and a link to 'Learn more about distributed builds'.

Definimos el nombre del pipeline

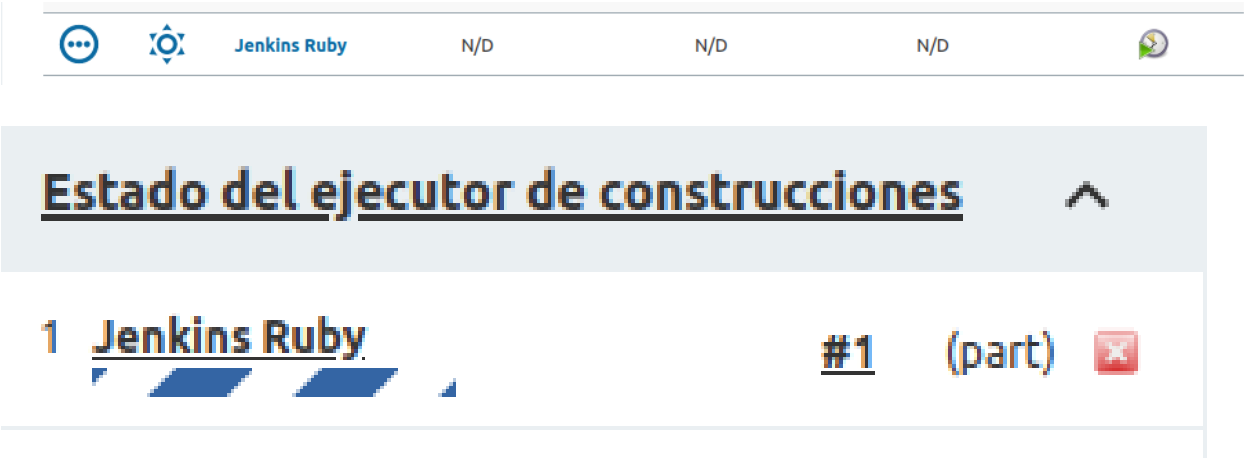


The screenshot shows the 'New Job' configuration page in Jenkins at www.jenkins.joel.com:8080/view/all/newJob. The 'Enter an item name' field is filled with 'Jenkins Ruby'. Below this, there are several job type options: 'Crear un proyecto de estilo libre', 'Pipeline', 'Crear un proyecto multi-configuración', 'Folder', 'Multibranch Pipeline', and 'Organization Folder'. Each option has a brief description. The 'Pipeline' option is highlighted, indicating it is the selected job type. At the bottom, there is an 'OK' button.

En la opción de pipeline, nos vamos a script y pegamos el script de la siguiente imagen



Una vez guardado, pulsamos la opción construir ahora



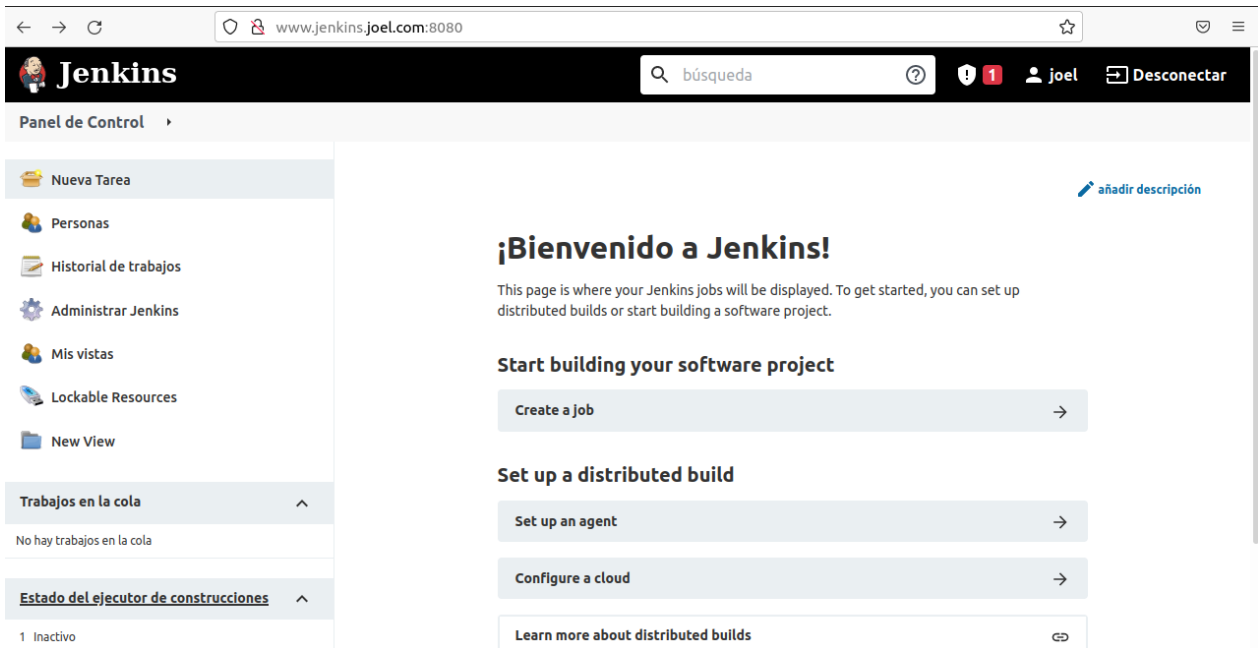
Finalmente se ha instalado el pipeline



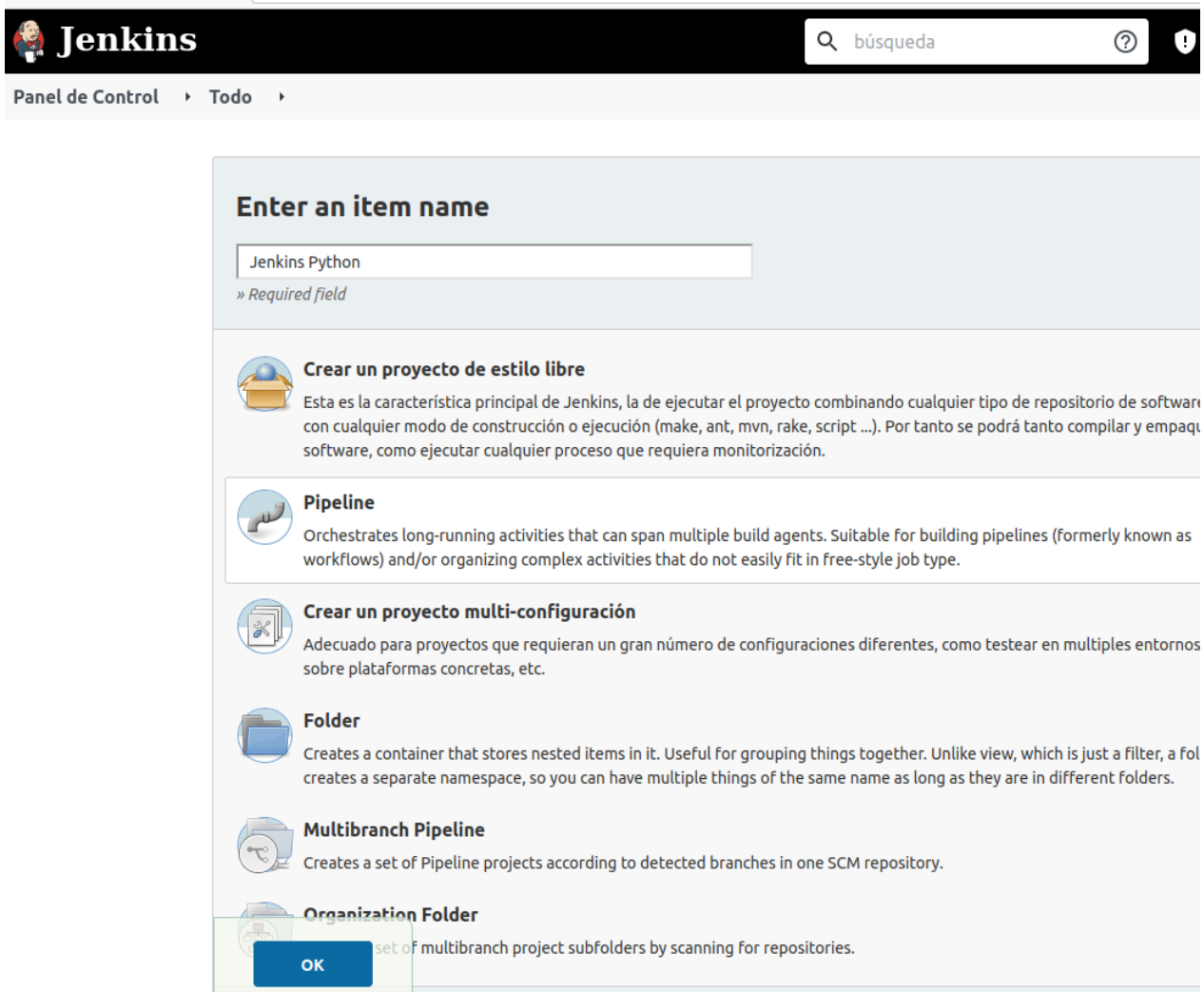
2.4. Python

Creación de Pipeline en Python

Clickeamos en nueva tarea



Definimos el nombre del pipeline



En la opción de pipeline, nos vamos a script y pegamos el script de la siguiente imagen

Panel de Control > Jenkins Python >

General Build Triggers Advanced Project Options **Pipeline**

Avanzado...

Pipeline

Definition

Pipeline script

Script

```

1 pipeline {
2   agent { docker { image 'python:3.10.1-alpine' } }
3   stages {
4     stage('build') {
5       steps {
6         sh 'python --version'
7       }
8     }
9   }
10 }

```

try sample Pipeline...

☒ Use Groovy Sandbox

[Pipeline Syntax](#)

Guardar Apply

Una vez guardado, pulsamos la opción construir ahora

⋮ ⚙ Jenkins Python N/D N/D N/D 🔄

1 Jenkins Python **#1** (part) ❌

Finalmente se ha instalado el pipeline

✅ ⚙ Jenkins Python 23 Seg - #1 N/D 17 Seg 🔄

2.5. PHP

Creación de Pipeline en PHP

Clickeamos en nueva tarea

← → ↻ www.jenkins.joel.com:8080 ☆ 🔔 1 👤 joel 🚪 Desconectar

Jenkins

Panel de Control ▸

Nueva Tarea

Personas

Historial de trabajos

Administrar Jenkins

Mis vistas

Lockable Resources

New View

Trabajos en la cola ^

No hay trabajos en la cola

Estado del ejecutor de construcciones ^

1 Inactivo

añadir descripción

¡Bienvenido a Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job →

Set up a distributed build

Set up an agent →

Configure a cloud →

Learn more about distributed builds ↗

Definimos el nombre del pipeline

← → ↻ www.jenkins.joel.com:8080/view/all/newJob 🔍 búsqueda

Jenkins

Panel de Control ▸ Todo ▸

Enter an item name

Jenkins PHP

» Required field

Crear un proyecto de estilo libre

Esta es la característica principal de Jenkins, la de ejecutar el proyecto combinando cualquier tipo de repositorio con cualquier modo de construcción o ejecución (make, ant, mvn, rake, script ...). Por tanto se podrá tanto compilar software, como ejecutar cualquier proceso que requiera monitorización.

Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (former workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Crear un proyecto multi-configuración

Adecuado para proyectos que requieran un gran número de configuraciones diferentes, como testear en múltiples plataformas concretas, etc.

Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just creates a separate namespace, so you can have multiple things of the same name as long as they are in different

Multibranch Pipeline

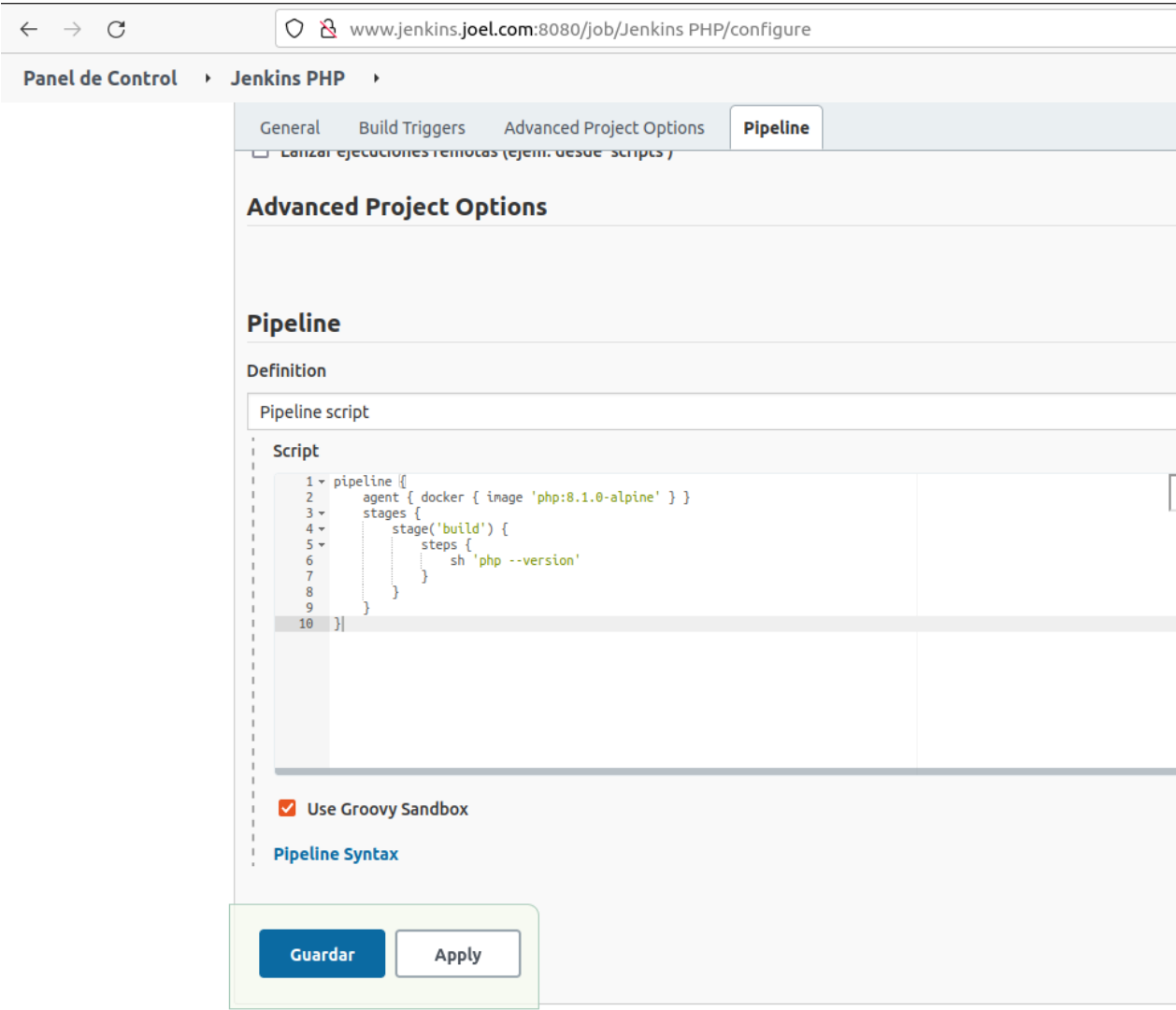
Creates a set of Pipeline projects according to detected branches in one SCM repository.

Organization Folder

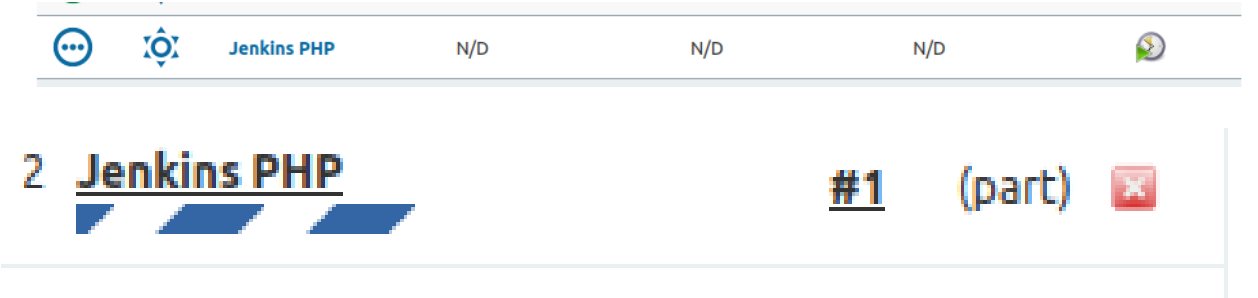
Creates a set of multibranch project subfolders by scanning for repositories.

OK

En la opción de pipeline, nos vamos a script y pegamos el script de la siguiente imagen



Una vez guardado, pulsamos la opción construir ahora



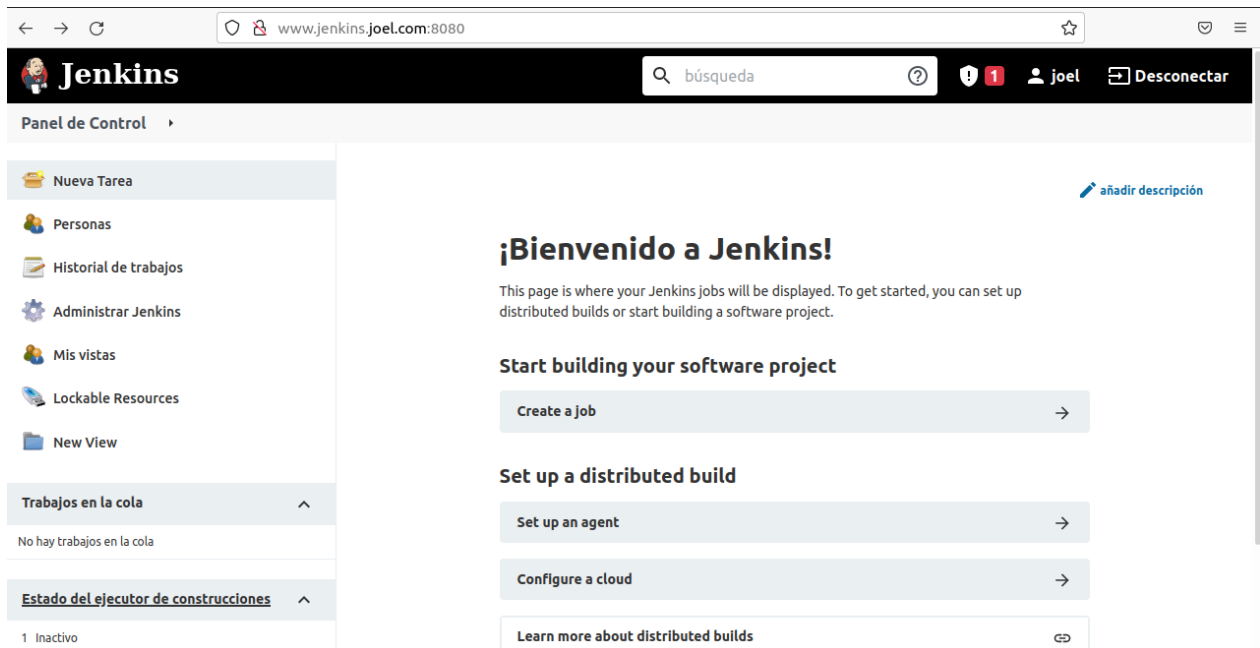
Finalmente se ha instalado el pipeline



2.6. GO

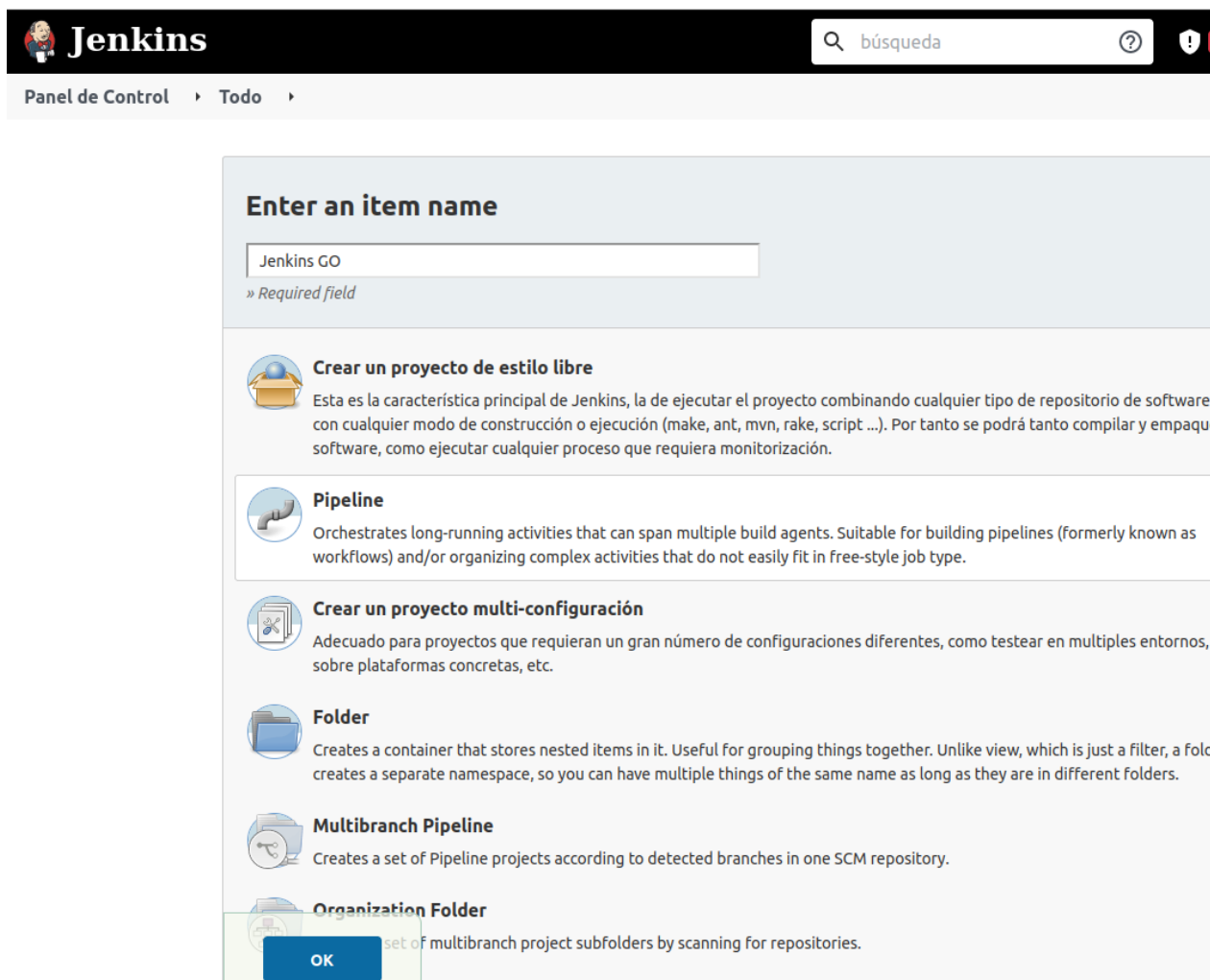
Creación de Pipeline en GO

Clickeamos en nueva tarea



The screenshot shows the Jenkins dashboard in a web browser. The address bar displays 'www.jenkins.joel.com:8080'. The Jenkins logo is in the top left, and a search bar with the text 'búsqueda' is in the top right. Below the header, there's a 'Panel de Control' section with a left sidebar containing links like 'Nueva Tarea', 'Personas', 'Historial de trabajos', 'Administrar Jenkins', 'Mis vistas', 'Lockable Resources', and 'New View'. The main content area has a large heading '¡Bienvenido a Jenkins!' followed by a paragraph: 'This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.' Below this, there's a section 'Start building your software project' with a 'Create a job' button. Another section 'Set up a distributed build' includes buttons for 'Set up an agent', 'Configure a cloud', and a link 'Learn more about distributed builds'.

Definimos el nombre del pipeline



The screenshot shows the 'Enter an item name' dialog in Jenkins. At the top, there's a text input field containing 'Jenkins GO' and a label '» Required field'. Below this, there are several options with icons and descriptions:

- Crear un proyecto de estilo libre**: Esta es la característica principal de Jenkins, la de ejecutar el proyecto combinando cualquier tipo de repositorio de software con cualquier modo de construcción o ejecución (make, ant, mvn, rake, script ...). Por tanto se podrá tanto compilar y empaquetar software, como ejecutar cualquier proceso que requiera monitorización.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Crear un proyecto multi-configuración**: Adecuado para proyectos que requieran un gran número de configuraciones diferentes, como testear en multiples entornos, sobre plataformas concretas, etc.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- Multibranch Pipeline**: Creates a set of Pipeline projects according to detected branches in one SCM repository.
- Organization Folder**: Creates a set of multibranch project subfolders by scanning for repositories.

At the bottom, there is a blue 'OK' button.

En la opción de pipeline, nos vamos a script y pegamos el script de la siguiente imagen

← → ↻ [www.jenkins.joel.com:8080/job/Jenkins GO/configure](http://www.jenkins.joel.com:8080/job/Jenkins%20GO/configure)

Panel de Control ▸ Jenkins GO ▸

General Build Triggers Advanced Project Options **Pipeline**

☐ Lanzar ejecuciones remotas (ejempl. desde scripts)

Advanced Project Options

Pipeline

Definition **Pipeline**

Pipeline script

Script

```
1 pipeline {
2   agent { docker { image 'golang:1.17.5-alpine' } }
3   stages {
4     stage('build') {
5       steps {
6         sh 'go version'
7       }
8     }
9   }
10 }
```

[try sample](#)

☒ Use Groovy Sandbox


[Pipeline Syntax](#)

Guardar **Apply**

Una vez guardado, pulsamos la opción construir ahora

⋮	⚙️	Jenkins GO	N/D	N/D	N/D	🔄
---	----	------------	-----	-----	-----	---

2 **Jenkins GO**

#1 (part) 

Finalmente se ha instalado el pipeline

✅	⚙️	Jenkins GO	53 Seg - #1	N/D	47 Seg	🔄
---	----	------------	-------------	-----	--------	---

Se muestran los distintos lenguajes instalados en jenkins pipeline

Todo +						
S	W	Nombre ↓	Último Éxito	Último Fallo	Última Duración	
		Jenkins GO	53 Seg - #1	N/D	47 Seg	
		Jenkins Java	1 Hor 5 Min - #4	1 Hor 12 Min - #2	53 Seg	
		Jenkins Node.js	14 Min - #1	N/D	22 Seg	
		Jenkins PHP	2 Min 43 Seg - #1	N/D	19 Seg	
		Jenkins Python	4 Min 57 Seg - #1	N/D	17 Seg	
		Jenkins Ruby	7 Min 15 Seg - #1	N/D	21 Seg	

Enlace github: [enlace](#)