

# Creación de los Pipeline en Java en Jenkins

---



IES  
Puerto  
de la  
Cruz  
Telesforo Bravo

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

---

## Índice

1. Requisitos básicos

2. Creacion de repositorio

---

### 3. Ficheros de configuración

---

### 4. Creación pipeline en java

---

#### 1. Requisitos básicos

- Tener jenkins instalado
- Haber realizado el ejercicio del siguiente [enlace](#)

#### 2. Creación de repositorios

Creamos el repositorio

# Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

## Repository template

Start your repository with a template repository's contents.

No template ▾

Owner \*



joel92MM ▾

/

Repository name \*

hello-word-java-apache-tomcat



Great repository names are short and memorable. Need inspiration? How about [glowing-meme?](#)

Description (optional)



**Public**

Anyone on the internet can see this repository. You choose who can commit.



**Private**

You choose who can see and commit to this repository.

## Initialize this repository with:

Skip this step if you're importing an existing repository.



**Add a README file**

This is where you can write a long description for your project. [Learn more.](#)




**Add .gitignore**

Choose which files not to track from a list of templates. [Learn more.](#)



**Choose a license**

A license tells others what they can and can't do with your code. [Learn more.](#)

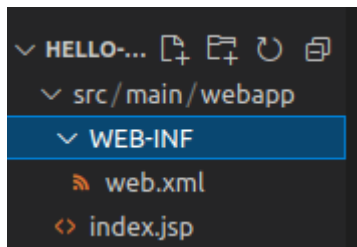
This will set  `main` as the default branch. Change the default name in your [settings](#).

Paramos todos los contenedores de docker

```
joel@joel-joel:~/Documentos/GitHub/hello-word-java-apache-tomcat$ sudo ch
mod 666 /var/run/docker.sock
joel@joel-joel:~/Documentos/GitHub/hello-word-java-apache-tomcat$ sudo do
cker stop $(docker ps -a -q)
6bccbebe6dca
6e63079b0e10
18677fcfe0de
d4a50b285c13
14e42e37afbf
1ea7fe794488
c00d994cd6a1
8b8aba04fd06
11754f536b17
e18a2aec2ce7
3f7a1e58075c
```

Clonamos el proyecto del siguiente [enlace]

(<https://github.com/jpexposito/docencia/blob/master/DPL/ARQUITECTURAS/tareas/despliegue-jsp-apache-tomcat.md>).



### 3. Ficheros de configuración

Creamos nuestro DockerFile, con la siguiente estructura...

```
FROM tomcat:version-seleccionada

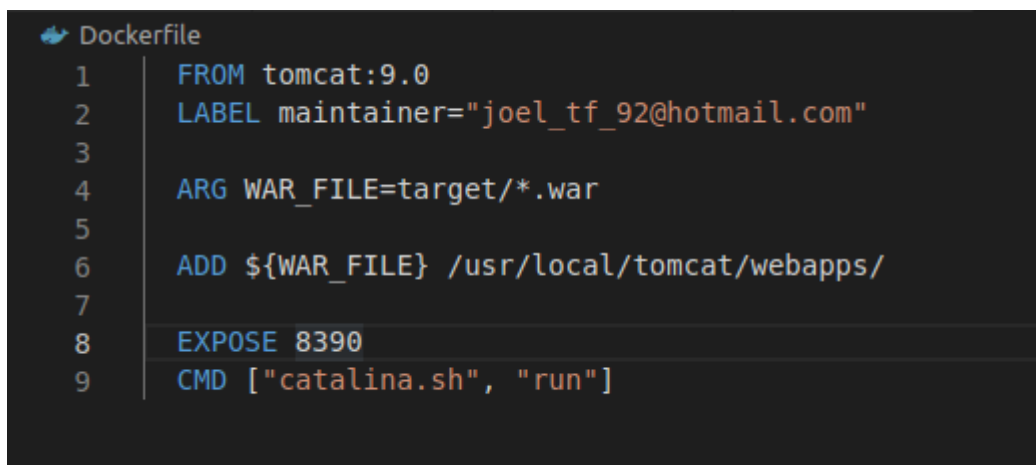
LABEL maintainer="emailalumno@iespuerto.es"

ARG WAR_FILE=target/*.war

ADD ${WAR_FILE} /usr/local/tomcat/webapps/

EXPOSE 8082

CMD ["catalina.sh", "run"]
```



Creamos el fichero Jenkinsfile, con el siguiente contenido...

- State Test Junit. Debe de ejecutar el comando:

```
mvn clean test
```

- State Build. Debe de ejecutar el comando:

```
mvn clean package
```

- State Deploy. Debe de ejecutar el comando:

```
Sentencias Docker
```

- Test Integration. Debe de ejecutar el comando:

Verificar que la aplicación esta desplegada a través de un (wget -m <http://www.example.com>).

Verificar que el fichero descargado contiene el nombre del alumno.

<https://ubunlog.com/buscar-cadenas-o-patrones-texto-sin-formato/>

```
Jenkinsfile
1  pipeline {
2      agent any
3
4      stages {
5          stage('Test Junit') {
6              steps {
7                  sh 'mvn clean test'
8                  echo 'Testing Junit..'
9              }
10         }
11         stage('Build') {
12             steps {
13                 sh 'mvn clean package'
14                 echo 'Building application..'
15             }
16         }
17         stage('Deploy') {
18             steps {
19
20                 sh 'docker build -t hello-word-java-apache-tomcat .'
21                 sh 'docker run -d --rm -p 8390:8080 hello-word-java-apache-tomcat'
22                 echo 'Deploying application....'
23             }
24         }
25         stage('Test Integration') {
26             steps {
27                 sh 'grep Joel | wget -m http://localhost:8390/app-web-demo'
28                 echo 'Testing integration..'
29             }
30         }
31     }
32 }
33
```

Creamos el target, con el siguiente comando

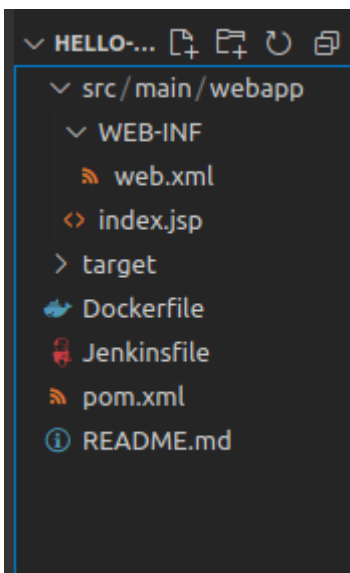
```
mvn clean install
```

```

joel@joel-joel:~/Documentos/GitHub/hello-word-java-apache-tomcat$ mvn clean install
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by com.google.inject.internal.cglib.core.$ReflectUtils$1 (file:/usr/share/maven/lib/guice.jar) to method java.lang.Cl
assLoader.defineClass(java.lang.String,byte[],int,int,java.security.ProtectionDomain)
WARNING: Please consider reporting this to the maintainers of com.google.inject.internal.cglib.core.$ReflectUtils$1
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
[INFO] Scanning for projects...
[INFO]
[INFO] -----< es.system.alumno:app-web-demo >-----
[INFO] Building app-web-demo 1.0-SNAPSHOT
[INFO] -----[ war ]-----
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ app-web-demo ---
[INFO] Deleting /home/joel/Documentos/GitHub/hello-word-java-apache-tomcat/target
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ app-web-demo ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/joel/Documentos/GitHub/hello-word-java-apache-tomcat/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.0:compile (default-compile) @ app-web-demo ---
[INFO] No sources to compile
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ app-web-demo ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /home/joel/Documentos/GitHub/hello-word-java-apache-tomcat/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.0:testCompile (default-testCompile) @ app-web-demo ---
[INFO] No sources to compile
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ app-web-demo ---
[INFO] No tests to run.
[INFO]
[INFO] --- maven-war-plugin:3.3.1:war (default-war) @ app-web-demo ---
[INFO] Packaging webapp
[INFO] Assembling webapp [app-web-demo] in [/home/joel/Documentos/GitHub/hello-word-java-apache-tomcat/target/app-web-demo]
[INFO] Processing war project
[INFO] Copying webapp resources [/home/joel/Documentos/GitHub/hello-word-java-apache-tomcat/src/main/webapp]
[INFO] Building war: /home/joel/Documentos/GitHub/hello-word-java-apache-tomcat/target/app-web-demo.war

```

Quedaría una estructura similar a la siguiente



Subimos el proyecto a nuestro GitHub

main
1 branch
0 tags
Go to file
Add file
Code

**joel92MM**
ed

ce3b415 1 minute ago 24 commits

src/main/webapp	ed	11 minutes ago
target	ed	11 minutes ago
.DS_Store	ed	17 hours ago
Dockerfile	ed	6 hours ago
Jenkinsfile	ed	2 hours ago
README.md	Initial commit	yesterday
pom.xml	edi	6 hours ago

README.md

# hello-word-java-apache-tomcat

#### 4. Creación pipeline en java

En el Jenkins nos creamos una nueva tarea pipeline nueva

[←](#)
[→](#)
[🔄](#)

[🔒](#)
[🌐](#)
[🔍](#)
[🌟](#)

[🔍](#)
[🔍](#)
[🔍](#)

[🔍](#)
[🔍](#)
[🔍](#)

**Jenkins**

[🔍](#)
[🔍](#)
[🔍](#)

[🔍](#)
[🔍](#)
[🔍](#)

[Panel de Control](#)
[Todo](#)

[🔍](#)
[🔍](#)

Enter an item name

» 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 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.

If you want to create a new item from other existing, you can use this option:

OK

Luego le indicamos nuestro repositorio Git con las credenciales

← → ↻ www.jenkins.joel.com:8080/job/Pipeline hello-word-java/configure

Panel de Control » Pipeline hello-word-java »

General Build Triggers Advanced Project Options Pipeline

### Build Triggers

- ☐ Construir tras otros proyectos
- ☐ Consultar repositorio (SCM)
- ☐ Ejecutar periódicamente
- ☐ GitHub hook trigger for GITScm polling
- ☐ Desactivar la ejecución
- ☐ Período de espera
- ☐ Lanzar ejecuciones remotas (ejem: desde 'scripts')

### Advanced Project Options

Avanzado...

### Pipeline

Definition

Pipeline script from SCM

SCM

Git

Repositories

Repository URL

https://github.com/joel92MM/hello-word-java-apache-tomcat

Credentials

joel92MM/\*\*\*\*\* Add

Avanzado...

add Repository

Guardar Apply

Branches to build

Branch Specifier (blank for 'any')

\*

Add Branch

Navegador del repositorio

(Auto)

Additional Behaviours

Añadir

Script Path

Jenkinsfile

☐ Lightweight checkout

Pipeline Syntax

Guardar Apply

Vemos el pipeline creado, lo ejecutamos

...		Pipeline hello-word-java	N/D	N/D	N/D	
-----	--	--------------------------	-----	-----	-----	--

Vemos el proceso de construcción



Panel de Control

Pipeline hello-word-java

Back to Dashboard

Status

Changes

Construir ahora

Configurar

Borrar Pipeline

Full Stage View

Rename

Pipeline Syntax

Historia de tareas

Tendencia

Filter builds...

#1

26 ene, 2022 18:30

Atom feed Para Todos

Atom feed para los errores

Pipeline Pipeline hello-word-java

añadir descripción

Desactivar el Proyecto

Recent Changes

Stage View

Declarative: Checkout SCM

Test Junit

Average stage times:

2s

2s

2s

Jan 26 18:30

No Changes

Enlaces permanentes

Última ejecución (#1) hace 1,2 Seg"

Si nos da el siguiente mensaje de error

## Salida de consola

```
Lanzada por el usuario joel
Checking out git https://github.com/joel92MM/hello-word-java-apache-tomcat into /var/lib/jenkins/workspace/Pipeline hello-word-java@script to read Jenkinsfile
The recommended git tool is: NONE
using credential 8932a364-e70f-4b8f-8e2c-0a6b1fa98f4e
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Pipeline hello-word-java@script/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/joel92MM/hello-word-java-apache-tomcat # timeout=10
Fetching upstream changes from https://github.com/joel92MM/hello-word-java-apache-tomcat
> git --version # timeout=10
> git --version # 'git version 2.25.1'
using GIT_ASKPASS to set credentials
> git fetch --tags --force --progress -- https://github.com/joel92MM/hello-word-java-apache-tomcat +refs/heads/*:refs/remotes/origin/* # timeout=10
Seen branch in repository origin/main
Seen 1 remote branch
> git show-ref --tags -d # timeout=10
Checking out Revision ce3b41568943298192d5537b7e3675221907f63f (origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f ce3b41568943298192d5537b7e3675221907f63f # timeout=10
Commit message: "ed"
First time build. Skipping changelog.
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/Pipeline hello-word-java
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
The recommended git tool is: NONE
using credential 8932a364-e70f-4b8f-8e2c-0a6b1fa98f4e
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Pipeline hello-word-java/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/joel92MM/hello-word-java-apache-tomcat # timeout=10

Removing intermediate container 9ccb80ffb71
---> c7ab284427ab
Successfully built c7ab284427ab
Successfully tagged hello-word-java-apache-tomcat:latest
[Pipeline] sh
+ docker run -d --rm -p 8390:8080 hello-word-java-apache-tomcat
39e7e0a0c934ceef796a8534de66e9c9dd15dd10772bd703a40b62aa7c5f26d3
docker: Error response from daemon: driver failed programming external connectivity on endpoint elegant_almeida
(6e463c7b3b5ecff57bce209489fd9c9cddc278286d9689e13c36cb0f697a5f0e7): Bind for 0.0.0.0:8390 failed: port is already allocated.
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
```

Nos iremos a la terminal y vemos los puertos ocupados

```
joel@joel-joel:~/Documentos/GitHub/hello-word-java-apache-tomcat$ netstat -putona
(No todos los procesos pueden ser identificados, no hay información de propiedad del proceso
no se mostrarán, necesita ser superusuario para verlos todos.)
Conexiones activas de Internet (servidores y establecidos)
Proto Recib Envíad Dirección local Dirección remota Estado PID/Program name Temporizador
tcp 0 0 0.0.0.0:8082 0.0.0.0:* ESCUCHAR - apagado (0.00/0/0)
tcp 0 0 127.0.0.53:53 0.0.0.0:* ESCUCHAR - apagado (0.00/0/0)
tcp 0 0 0.0.0.0:22 0.0.0.0:* ESCUCHAR - apagado (0.00/0/0)
tcp 0 0 127.0.0.1:631 0.0.0.0:* ESCUCHAR - apagado (0.00/0/0)
tcp 0 0 0.0.0.0:8443 0.0.0.0:* ESCUCHAR - apagado (0.00/0/0)
tcp 0 0 127.0.0.1:38561 0.0.0.0:* ESCUCHAR 5758/app --node-int apagado (0.00/0/0)
tcp 0 0 127.0.0.1:33060 0.0.0.0:* ESCUCHAR - apagado (0.00/0/0)
tcp 0 0 0.0.0.0:8390 0.0.0.0:* ESCUCHAR - apagado (0.00/0/0)
tcp 0 0 127.0.0.1:9990 0.0.0.0:* ESCUCHAR - apagado (0.00/0/0)
tcp 0 0 127.0.0.1:3306 0.0.0.0:* ESCUCHAR - apagado (0.00/0/0)
tcp 0 0 10.0.2.15:60636 140.82.114.26:443 ESTABLECIDO 3366/firefox mantener vivo (591,50/0/0)
tcp 0 0 127.0.0.1:54746 127.0.1.1:8080 TIME WAIT - tiempo de espera (26,04/0/0)
tcp 0 0 10.0.2.15:49452 140.82.121.3:443 TIME WAIT - tiempo de espera (50,49/0/0)
tcp 0 0 10.0.2.15:59032 34.120.208.123:443 ESTABLECIDO 3366/firefox apagado (0.00/0/0)
tcp 0 0 127.0.0.1:54748 127.0.1.1:8080 TIME WAIT - tiempo de espera (53,50/0/0)
tcp 0 0 10.0.2.15:60706 34.210.39.83:443 ESTABLECIDO 3366/firefox mantener vivo (295,08/0/0)
tcp6 0 0 :::8081 :::* ESCUCHAR - apagado (0.00/0/0)
tcp6 0 0 :::21 :::* ESCUCHAR - apagado (0.00/0/0)
tcp6 0 0 :::22 :::* ESCUCHAR - apagado (0.00/0/0)
tcp6 0 0 :::1:631 :::* ESCUCHAR - apagado (0.00/0/0)
tcp6 0 0 127.0.0.1:8005 :::* ESCUCHAR - apagado (0.00/0/0)
tcp6 0 0 :::8390 :::* ESCUCHAR - apagado (0.00/0/0)
tcp6 0 0 :::8080 :::* ESCUCHAR - apagado (0.00/0/0)
tcp6 0 0 :::80 :::* ESCUCHAR - apagado (0.00/0/0)
tcp6 0 0 127.0.0.1:8080 127.0.0.1:54742 TIME WAIT - tiempo de espera (21,01/0/0)
udp 0 0 127.0.0.53:53 0.0.0.0:* - apagado (0.00/0/0)
udp 0 0 10.0.2.15:68 10.0.2.2:67 ESTABLECIDO - apagado (0.00/0/0)
udp 0 0 0.0.0.0:631 0.0.0.0:* - apagado (0.00/0/0)
udp 0 0 0.0.0.0:5353 0.0.0.0:* - apagado (0.00/0/0)
udp 0 0 127.0.0.1:40232 127.0.0.53:53 ESTABLECIDO 8800/code apagado (0.00/0/0)
udp 0 0 0.0.0.0:57094 0.0.0.0:* - apagado (0.00/0/0)
udp6 0 0 :::52161 :::* - apagado (0.00/0/0)
udp6 0 0 :::5353 :::* - apagado (0.00/0/0)
```

Vemos el contenedor que esta ocupando ese puerto

```
joel@joel-joel:~/Documentos/GitHub/hello-word-java-apache-tomcat$ sudo docker ps -a
[sudo] contraseña para joel:
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAME
1649eabdd34   d42104e5e13d  "catalina.sh run"       About an hour ago    Up About an hour    8390/tcp, 0.0.0.0:8390->8080/tcp, :::8390->8080/tcp    nift_y_sanderson
```

Luego paramos el contenedor

```
joel@joel-joel:~/Documentos/GitHub/hello-word-java-apache-tomcat$ sudo docker stop 1649eabdd34
1649eabdd34
```

Volvemos a ejecutar el pipeline, y finalmente se ejecuta

Panel de Control

Pipeline hello-word-java

Back to Dashboard

Status

Changes

Construir ahora

Configurar

Borrar Pipeline

Full Stage View

Rename

Pipeline Syntax

Historia de tareas

Tendencia

Filter builds...

#2 26 ene. 2022 18:34

#1 26 ene. 2022 18:30

Atom feed Para Todos

Atom feed para los errores

Pipeline Pipeline hello-word-java

Recent Changes

Stage View

	Declarative: Checkout SCM	Test Junit	Build	Deploy	Test Integration
Average stage times: (Average full run time: ~54s)	2s	6s	9s	20s	6s
#2 Jan 26 18:34 No Changes	2s	6s	6s	18s	13s
#1 Jan 26 18:30 No Changes	2s	7s	11s	22s failed	784ms failed

Enlaces permanentes

- "Última ejecución (#2) hace 2 Seg"
- "Última ejecución fallida (#1) hace 3 Min 39 Seg"
- "Última ejecución fallida (#1) hace 3 Min 39 Seg"
- "Last completed build (#1) hace 3 Min 39 Seg"

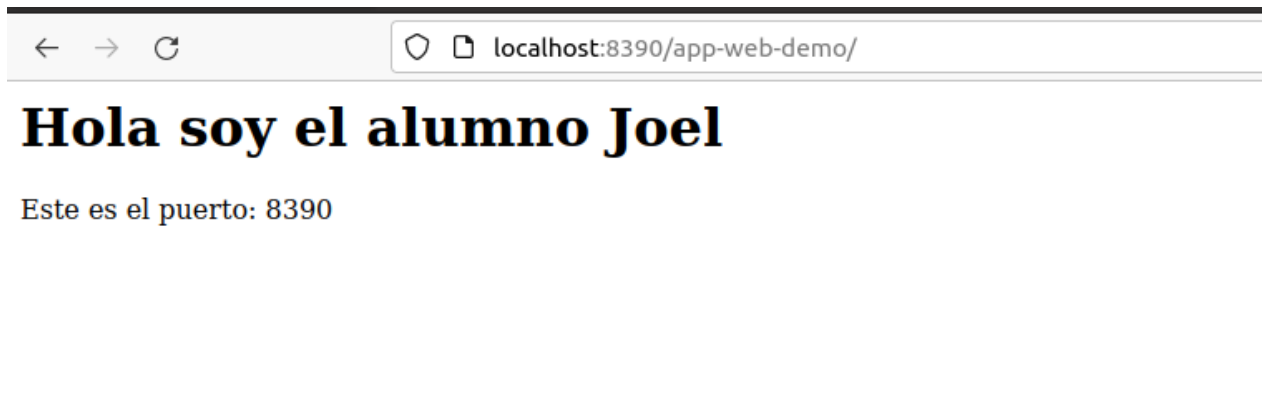
La salida por consola es correcta



## Salida de consola

```
Lanzada por el usuario joel
Checking out git https://github.com/joel92MM/hello-word-java-apache-tomcat into /var/lib/jenkins/workspace/Pipeline hello-word-java@script to read Jenkinsfile
The recommended git tool is: NONE
using credential 8932a364-e70f-4b8f-8e2c-0a6b1fa98f4e
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Pipeline hello-word-java@script/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/joel92MM/hello-word-java-apache-tomcat # timeout=10
Fetching upstream changes from https://github.com/joel92MM/hello-word-java-apache-tomcat
> git --version # timeout=10
> git --version # 'git version 2.25.1'
using GIT_ASKPASS to set credentials
> git fetch --tags --force --progress -- https://github.com/joel92MM/hello-word-java-apache-tomcat +refs/heads/*:refs/remotes/origin/* # timeout=10
Seen branch in repository origin/main
Seen 1 remote branch
> git show-ref --tags -d # timeout=10
Checking out Revision ce3b41568943298192d5537b7e3675221907f63f (origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f ce3b41568943298192d5537b7e3675221907f63f # timeout=10
Commit message: "ed"
> git rev-list --no-walk ce3b41568943298192d5537b7e3675221907f63f # timeout=10
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/Pipeline hello-word-java
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
The recommended git tool is: NONE
using credential 8932a364-e70f-4b8f-8e2c-0a6b1fa98f4e
```

Si mostramos la direccion en el navegador, se mostraria el resultado siguiente



Enlace github: [enlace](https://github.com/joel92MM/hello-word-java-apache-tomcat)