

Tarea 25: Creación de Pipeline en Java



Jenkins

Índice

1. Clonación del repositorio github
2. Creación de los archivos dockerFile y Jenkinfile
3. Actualización de los cambios del repositorio
4. Creación del Pipeline en PHP dentro de Jenkins
5. Resultado del Pipeline construido

1. Clonación del repositorio github

Primero vamos a clonar el proyecto proporcionado a nuestro repositorio de github en local

```
-----
usuario@ubuntu-20:~/hello-word-php-apache$ ls
app-web-demo  Dockerfile  Jenkinfile  src
usuario@ubuntu-20:~/hello-word-php-apache$
```

Una vez clonado lo renombramos a hello-word-java-apache-tomcat

```
usuario@ubuntu-20:~/hello-word-php-apache$ sudo mv app-web-demo hello-word-java-
apache-tomcat
[sudo] contraseña para usuario:
usuario@ubuntu-20:~/hello-word-php-apache$ ls
Dockerfile  hello-word-java-apache-tomcat  Jenkinfile  src
usuario@ubuntu-20:~/hello-word-php-apache$
```

2. Creación de los archivos DockerFile y Jenkinfile

Ahora Modificaremos el archivo Dockerfile con el siguiente contenido

```
usuario@ubuntu-20:~/hello-word-php-apache$ cd hello-word-java-apache-tomcat
usuario@ubuntu-20:~/hello-word-php-apache/hello-word-java-apache-tomcat$ ls
docker-compose.yml  Dockerfile  pom.xml  src
usuario@ubuntu-20:~/hello-word-php-apache/hello-word-java-apache-tomcat$ sudo na
no Dockerfile
usuario@ubuntu-20:~/hello-word-php-apache/hello-word-java-apache-tomcat$
```

```

GNU nano 5.2 Dockerfile
FROM jboss/wildfly

ARG WAR_FILE=target/*.war

ADD ${ARG} /opt/jboss/wildfly/standalone/deployments/

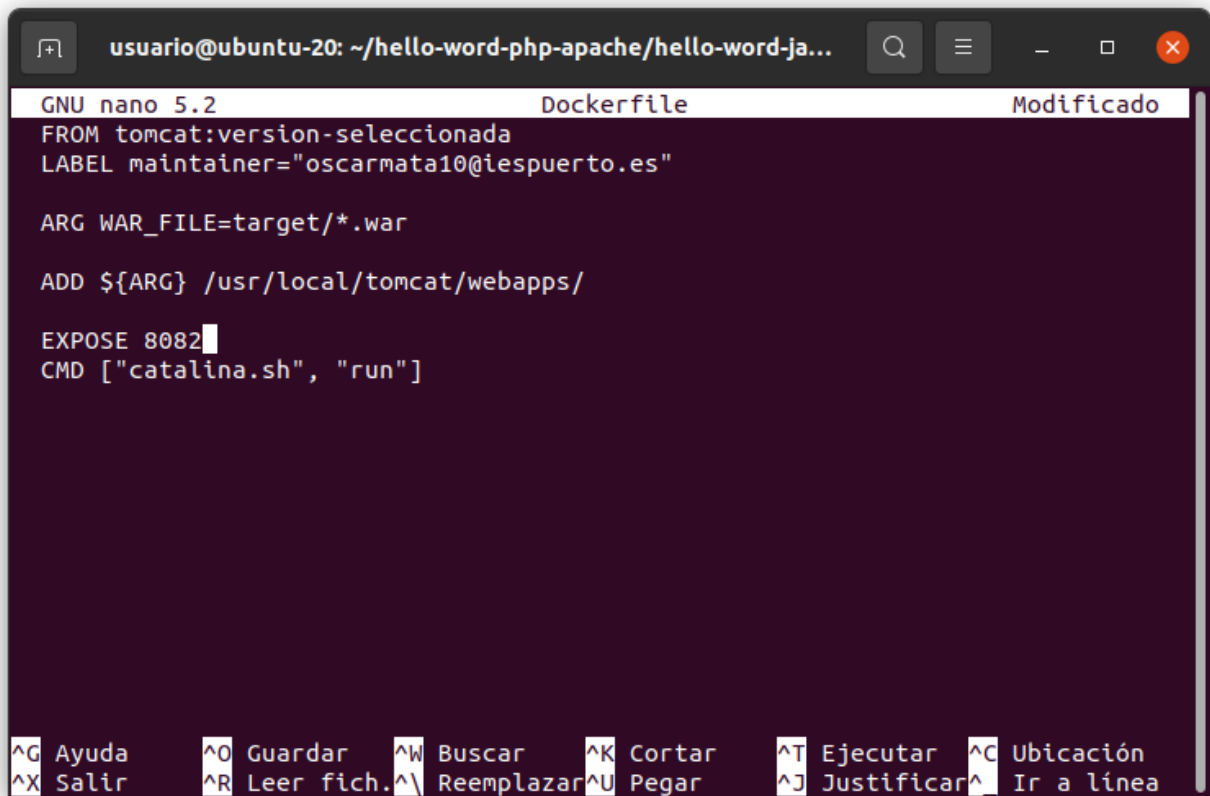
ARG WILDFLY_NAME
ARG CLUSTER_PW

ENV WILDFLY_NAME=${WILDFLY_NAME}
ENV CLUSTER_PW=${CLUSTER_PW}

ENTRYPOINT /opt/jboss/wildfly/bin/standalone.sh -b=0.0.0.0 -bmanagement=0.0.0.0
  
```

[14 líneas leídas]

^G Ayuda ^O Guardar ^W Buscar ^K Cortar ^T Ejecutar ^C Ubicación
 ^X Salir ^R Leer fich. ^U Reemplazar ^V Pegar ^J Justificar ^_ Ir a línea



The screenshot shows a terminal window with the title bar "usuario@ubuntu-20: ~/hello-word-php-apache/hello-word-ja...". The terminal is running the GNU nano 5.2 editor, editing a file named "Dockerfile". The content of the Dockerfile is as follows:

```

FROM tomcat:version-seleccionada
LABEL maintainer="oscarмата10@iespuerto.es"

ARG WAR_FILE=target/*.war

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

EXPOSE 8082
CMD ["catalina.sh", "run"]

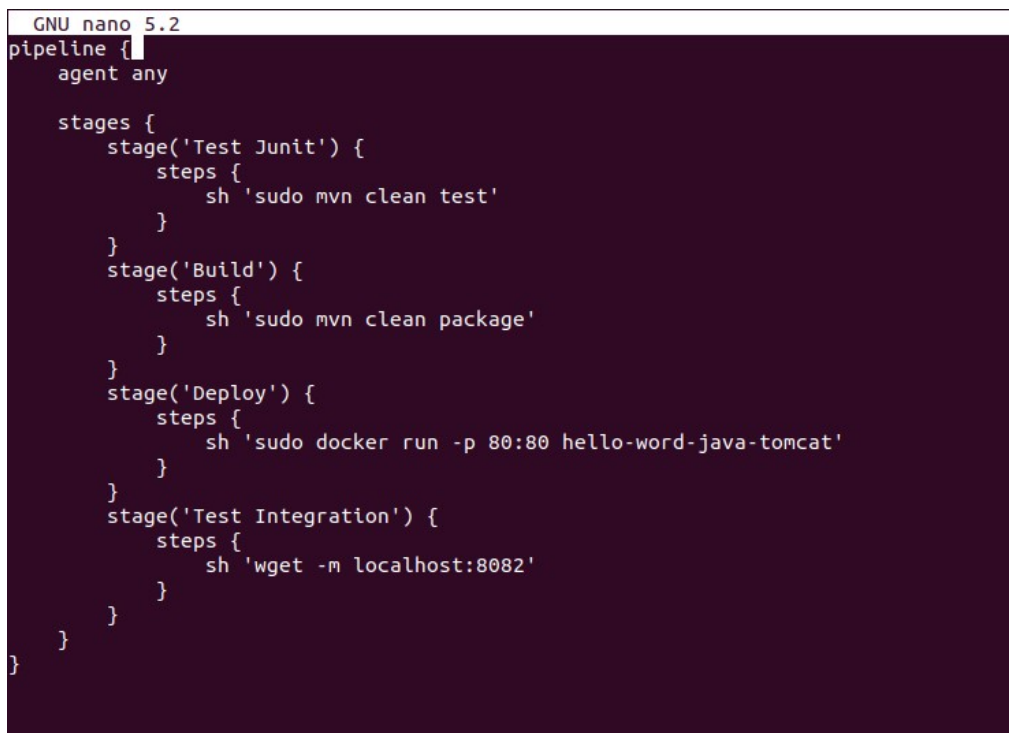
```

At the bottom of the terminal, there is a status bar with keyboard shortcuts for various nano editor functions:

^G Ayuda	^O Guardar	^W Buscar	^K Cortar	^T Ejecutar	^C Ubicación
^X Salir	^R Leer fich.	^L Reemplazar	^U Pegar	^J Justificar	^_ Ir a línea

Y luego crearemos un archivo Jenkinsfile con el siguiente contenido

```
usuario@ubuntu-20:~/hello-word-php-apache/hello-word-java-apache-tomcat$ sudo nano Jenkinsfile
```



The screenshot shows a terminal window with the title bar "GNU nano 5.2". The terminal is running the GNU nano 5.2 editor, editing a file named "Jenkinsfile". The content of the Jenkinsfile is as follows:

```

pipeline {
    agent any

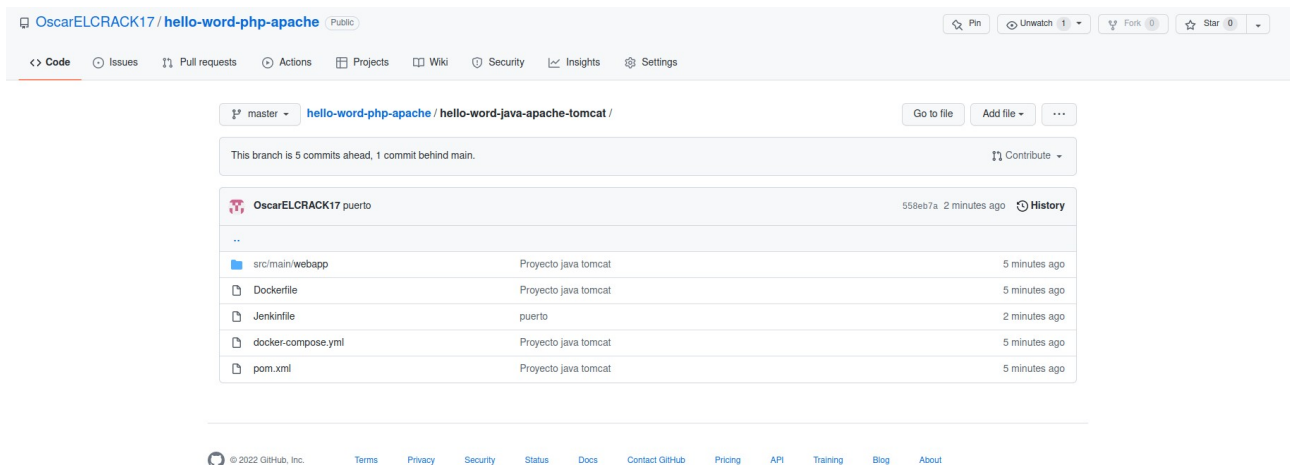
    stages {
        stage('Test Junit') {
            steps {
                sh 'sudo mvn clean test'
            }
        }
        stage('Build') {
            steps {
                sh 'sudo mvn clean package'
            }
        }
        stage('Deploy') {
            steps {
                sh 'sudo docker run -p 80:80 hello-word-java-tomcat'
            }
        }
        stage('Test Integration') {
            steps {
                sh 'wget -m localhost:8082'
            }
        }
    }
}

```

Nota: En el Jenkinsfile se debe poner 80:82 para que luego no de error (en el siguiente paso ya está cambiado).

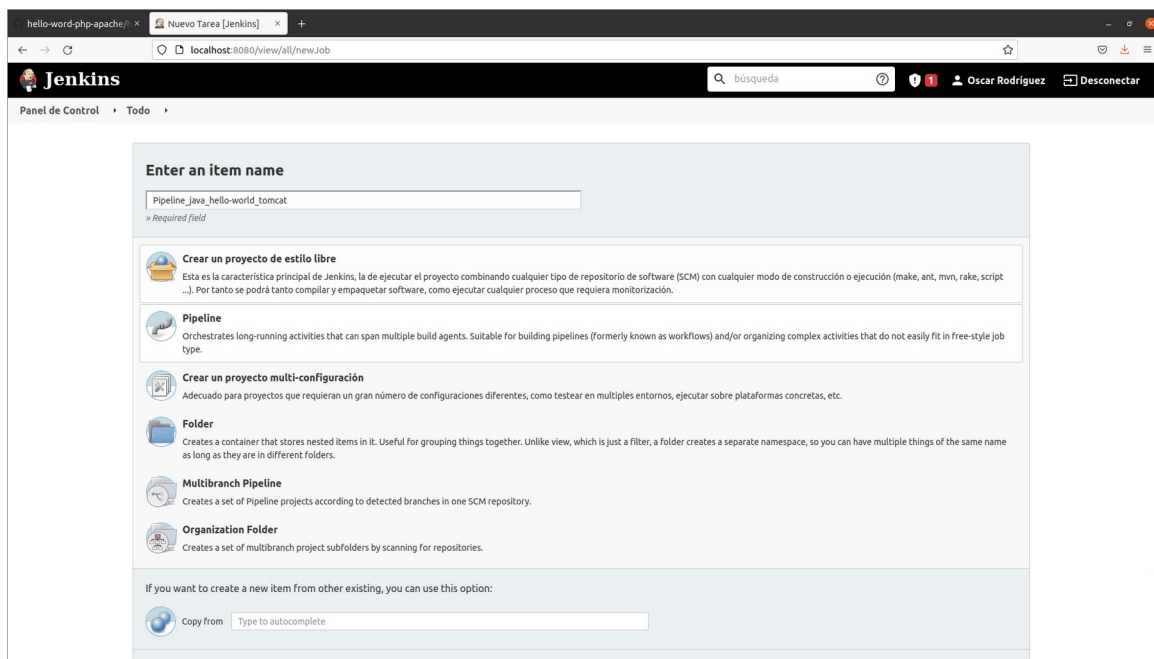
3. Actualización de los cambios del repositorio github

Ahora hacemos un push para subir todos los cambios a nuestro repositorio y este sería el resultado



4. Creación del Pipeline en PHP en Jenkins

Ahora al igual que en la práctica anterior utilizaremos Jenkins para crear nuestro pipeline en Java, pero cuando lo creamos lo vinculamos con nuestro repositorio de GitHub que hemos creado.



General Build Triggers Advanced Project Options Pipeline

Descripción

[Plain text] Visualizar

- ☐ Desechar ejecuciones antiguas
- ☐ Do not allow concurrent builds
- ☐ Do not allow the pipeline to resume if the controller restarts
- ☐ Esta ejecución debe parametrizarse
- ☒ GitHub project

Project url

`https://github.com/OscarELCRACK17/hello-word-php-apache/tree/master/hello-word-java-apache-tomcat`

Avanzado...

- ☐ Pipeline speed/durability override
- ☐ Preserve stashes from completed builds
- ☐ Throttle builds

Build Triggers

- ☐ Construir tras otros proyectos
- ☐ Consultar repositorio (SCM)
- ☐ Ejecutar periódicamente
- ☒ GitHub hook trigger for GITSCM polling
- ☐ Desactivar la ejecución

5. Resultado del Pipeline construido

Ahora construiremos el Pipeline y comprobaremos que el resultado sea el correcto

		Pipeline_java_hello-world_tomcat	26 Seg - #1	N/D	3,3 Seg	
--	--	----------------------------------	-------------	-----	---------	--



Recent Changes

Stage View

