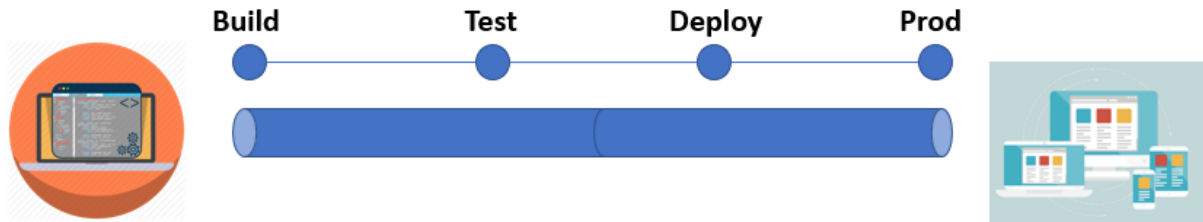


Crear una pipeline de Jenkins con multiples etapas

Así es como se ve un flujo de pipeline de Jenkins, que consta de múltiples etapas entre los desarrolladores que crean un software y el software entregado en producción.



Creemos una pipeline declarativa.

En el panel de Jenkins, haga clic en Nuevo elemento. Luego ingrese un nombre de elemento, por ejemplo, 'Primera pipeline' y seleccione el proyecto 'Pipeline'. Luego haga clic en Aceptar.

Enter an item name

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

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.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Haga clic en la pestaña Pipeline como se muestra en la imagen a continuación y coloque su código JenkinsFile (Código Groovy) aquí.

General

Build Triggers

Advanced Project Options

Pipeline

Pipeline

Definition

Pipeline script

Script

```
1 pipeline {  
2     agent any  
3     stages {  
4         stage('Build') {  
5             steps {  
6                 echo 'Hi, GeekFlare. Sarting to build the App.'  
7             }  
8         }  
9         stage('Test') {  
10            steps {  
11                input('Do you want to proceed?')  
12            }  
13        }  
14        stage('Deploy') {  
15            parallel {  
16                stage('Deploy start ') {  
17                    steps {  
18                        echo "Start the deploy .."  
19                    }  
20                }  
21            }  
22        }  
23    }  
24 }
```

☒ Use Groovy Sandbox

[Pipeline Syntax](#)

Save

Apply

```
pipeline {  
    agent any  
    stages {  
        stage('Build') {  
            steps {  
                echo 'Hi. Starting to build the App.'  
            }  
        }  
        stage('Test') {  
            steps {  
                input('Do you want to proceed?')  
            }  
        }  
        stage('Deploy') {  
            parallel {  
                stage('Deploy start ') {  
                    steps {  
                        echo "Start the deploy .."  
                    }  
                }  
                stage('Deploying now') {  
                    steps {  
                        echo "'Simulated Docker' Created"  
                    }  
                }  
            }  
        }  
        stage('Prod') {  
            steps {  
                echo "App is Prod Ready"  
            }  
        }  
    }  
}
```

```
}  
}
```

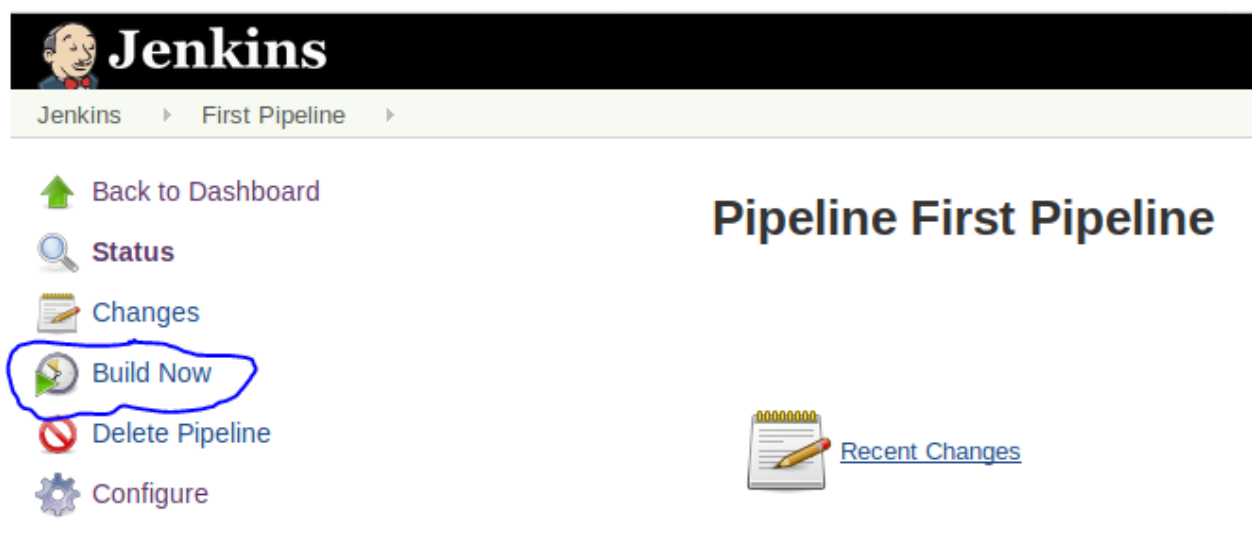
Déjame explicarte los bloques anteriores.

- El bloque **de pipeline** consta de todas las instrucciones para crear, probar y entregar software. Es el componente clave de Jenkins Pipeline.
- Se asigna un **agente** para ejecutar la canalización en un nodo y asignar un espacio de trabajo para la canalización.
- Una **etapa** es un bloque que tiene pasos para construir, probar e implementar la aplicación. Las etapas se utilizan para visualizar los procesos de Jenkins Pipeline.
- Un **paso** es una única tarea que se debe realizar, por ejemplo, crear un directorio, ejecutar una imagen de la ventana acoplable, eliminar un archivo, etc.

El código Groovy anterior lo estoy usando para JenkinsFile. Cualquier agente disponible será asignado al pipeline. Luego defino la etapa de compilación y realizo un paso de eco simple. Luego definí la etapa de Prueba donde el paso pregunta si desea continuar o no. Después de eso, creé una etapa de Implementación, que tiene dos etapas más ejecutándose en paralelo. La etapa de inicio de implementación tiene un paso con el comando echo, y la implementación ahora otro eco. Finalmente, hay una etapa Prod con un simple paso de eco.

El proceso explicado anteriormente tiene etapas que tienen pasos simples para que usted comprenda cómo funciona. Una vez que aprenda a crear una canalización, podrá agregar más complejidad y crear canalizaciones complejas también.

Una vez que tenga el código en la pestaña Pipeline, haga clic en Aplicar y Guardar. Finalmente, haga clic en Construir ahora para comenzar a construir el Jenkins Pipeline que acaba de crear.



Así es como se verá la interfaz de usuario cuando se ejecute la canalización. Si ha seguido todos los pasos correctamente, su compilación será exitosa y mostrará un color azul para la compilación (#27). Si hay errores en la construcción, le dará color rojo a la construcción (#27).

Pipeline First Pipeline

 [add description](#)

[Disable Project](#)




[Recent Changes](#)

Stage View


		Build	Test	Deploy	Deploy start	Deploying now	Prod
Average stage times: (Average full run time: ~35s)		361ms	483ms	396ms	4s	18s	539ms
#27	Nov 09 16:25 No Changes	310ms	366ms (paused for 13s)	375ms	2s	13s	415ms

Ahora haga clic en la compilación n.º 27 y luego haga clic en Salida de la consola para verificar qué sucedió exactamente en el back-end cuando se ejecutó.

 [Back to Project](#)

 [Status](#)

 [Changes](#)

 [Console Output](#)

 [Edit Build Information](#)

 [Delete build '#27'](#)

 [Docker Fingerprints](#)

 [Restart from Stage](#)

 [Replay](#)

 [Pipeline Steps](#)

 [Workspaces](#)

 [Previous Build](#)

 [Next Build](#)

 **Build #27 (Nov 9, 2019 4:25:48 PM)**



Started by user [geekflare](#)



This was approved by user [geekflare](#).

Started by user [Juan Pablo Arias Mora](#)

[Pipeline] Start of Pipeline

[Pipeline] node

Running on [Jenkins](#) in /var/lib/jenkins/workspace/Hello World 3

[Pipeline] {

```
[Pipeline] stage
[Pipeline] { (Build)
[Pipeline] echo
Hi, GeekFlare. Starting to build the App.
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Test)
[Pipeline] input
Do you want to proceed?
Proceed or Abort
Approved by Juan Pablo Arias Mora
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Deploy)
[Pipeline] parallel
[Pipeline] { (Branch: Deploy start )
[Pipeline] { (Branch: Deploying now)
[Pipeline] stage
[Pipeline] { (Deploy start )
[Pipeline] stage
[Pipeline] { (Deploying now)
[Pipeline] echo
Start the deploy ..
[Pipeline] }
[Pipeline] echo
'Simulated Docker' Created
[Pipeline] }
[Pipeline] // stage
[Pipeline] // stage
[Pipeline] }
[Pipeline] }
[Pipeline] // parallel
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Prod)
[Pipeline] echo
App is Prod Ready
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```