

Integracion Continua con Jenkins

=====

Table of Contents

1. Integración Continua	1
1.1. Introducción	1
1.2. Buenas practicas para la CI	1
1.3. ¿Porque Integracion Continua?	2
2. Jenkins	2
2.1. Introducción	2
2.2. Instalación	2
2.3. Configuración	4
2.4. Seguridad y Gestion de usuarios.....	7
2.5. Tareas (Jobs)	9
2.6. Plugins.....	13
2.7. Scripting con Jenkins CLI	19
2.8. Consola de Script Integrada	20
2.9. API de acceso remoto	22
2.10. Ejecución parametrizada	22
2.11. Tarea Multiconfiguración	23
2.12. Dependencias entre proyectos	24
2.13. Ejecución Distribuida	24

1. Integración Continua

1.1. Introducción

Modelo propuesto inicialmente por [Martin Fowler](#) que consiste en automatizar un proceso sobre los proyectos, comprendiendo este proceso los siguientes pasos

- Descarga de fuentes desde el SCM.
- Compilación del código.
- Ejecución de las pruebas.
- Validación de informes.

Este proceso persigue la vigilancia del código, para la detección temprana de errores.

Para esto se utilizan aplicaciones como

- Bamboo
- Continuum
- Hudson
- Jenkins
- CruiseControl
- Team Foundation Server

1.2. Buenas practicas para la CI

Según Martin Fowler, se deben seguir las siguientes buenas practicas

- Mantener el código versionado con un SCM (Git, SVN, CVS, ...).
- Automatizar la construcción (Jenkins, Bamboo, ...)
- Crear Test que permitan tener confianza en el código generado (JUnit, Selenium, SoapUI, ...).
- Commits diarios al SCM, para que el servidor de CI, ofrezca una imagen real del desarrollo.
- Cada commit debería forzar un build en el servidor de CI.
- Aviso inmediato al autor/equipo del commit que introduce una inestabilidad.
- Mantener una construcción rápida, que el proceso de CI no sea vital, no quiere decir que pueda tardar mucho en ejecutarse.
- Pruebas sobre un clon de producción.
- Obtención fácil de las construcciones (Releases, snapshot) a través de servidores de artefactos.
- Visibilidad del proceso de IC y de los reportes para todos los miembros del equipo.

1.3. ¿Porque Integracion Continua?

Normalmente en los proyectos las entregas son el punto mas caliente, no solo porque supone culminar un trabajo, sino porque que se llega a ellas con poca información del estado del proyecto.

- ¿Cuanto se tarda en desplegar?
- ¿Tiene defectos la aplicación?
- ¿Cuántos tests han pasado?¿De que tipo son?
- ¿Cómo es el código de robusto?

Aplicando la IC, se puede obtener este tipo de onformación desde el principio y tener una visión de la evolución.

2. Jenkins

2.1. Introducción

Servidor de Integracion Continua (CI), basado en Hudson.

Creado por Kohsuke Kawaguchi. Esta liberado bajo licencia MIT.

Jenkins tiene la posibilidad de ser extendido mediante Plugins, existiend multitud de ellos disponibles, mas información [aquí](#)

2.2. Instalación

Desde la [pagina oficial](#) se puede realizar la descarga de Jenkins en multiples modalidades.

Si se descarga el **war**, este puede ser desplegado en el servidor de aplicaciones deseado.

Tambien se puede auto ejecutar, ya que lleva embebido un Jetty.

```
java -jar jenkins.war
```

Si se desea cambiar el puerto donde escucha **Jenkins**, que por defecto es el **8080**

```
java -jar jenkins.war --httpPort=8081
```

Otra opción es por ejemplo el instalador de Jenkins para Windows, que crea un servicio de Windows para poder manejar Jenkins.

Cuando se arranca el servicio por defecto Jenkins escucha en **localhost:8080**

Independientemente de como se ejecute, **Jenkins** necesita un directorio de trabajo, que por defecto tiene los siguientes valores para las distintas plataformas con un usuario **admin**

- **Windows 7** → **C:\Users\admin\.jenkins**
- **Windows XP** → **C:\Documents and Settings\admin\.jenkins**
- **Linux** → **/home/admin/.jenkins**

Si se desea cambiar, lo unico que habrá que hacer será definir la variable de entorno **JENKINS_HOME** con la ubicación deseada.

Si por ejemplo se despliega en un **Tomcat**, este puede ser el encargado de definir dicha variable para su ejecución, para ello se ha de crear un fichero **jenkins.xml** en el directorio **\$CATALINA_BASE/conf/localhost**, con el siguiente contenido

```
<Context docBase="../../jenkins.war">
  <Environment name="JENKINS_HOME" type="java.lang.String" value="/data/jenkins"
  override="true"/>
</Context>
```

Tambien se puede cambiar a nivel de la **JVM**

```
java -jar -DJENKINS_HOME=D:\utilidades\jenkins jenkins.war
```

En el proceso de instalación, se pide la definición de un usuario administrador

Getting Started

×

Create First Admin User

Usuario:

Contraseña:

Confirma la contraseña:

Nombre completo:

Dirección de email:

[Continue as admin](#) [Save and Finish](#)

Para el curso se establecerá **admin/admin**.

2.3. Configuración

La zona de configuración se accede a través del enlace **Administrar Jenkins**

Jenkins admin | [Desconectar](#)

[ACTIVAR AUTO-REFRESCO](#)

Administrar Jenkins

- [Configurar el Sistema](#)
Configurar variables globales y rutas.
- [Configuración global de la seguridad](#)
Seguridad en Jenkins. Define quién tiene acceso al sistema (autenticación) y qué puede hacer (autorización)
- [Global Tool Configuration](#)
Configure tools, their locations and automatic installers.
- [Actualizar configuración desde el disco duro](#)
Descartar todos los datos cargados en memoria y actualizar todo nuevamente desde los ficheros del sistema. Útil cuando se modifican ficheros de configuración directamente en el disco duro.
- [Administrar Plugins](#)
Añadir, borrar, desactivar y activar plugins que extienden la funcionalidad de Jenkins.
- [Información del sistema](#)
Muestra información del entorno que puedan ayudar a la solución de problemas.
- [System Log](#)
El log del sistema captura la salidas de la clase java.util.logging en todo lo relacionado con Jenkins.
- [Estadísticas de Carga](#)
Comprobar la utilización de los recursos y comprobar si es necesario añadir nuevos nodos para la ejecución de tareas.
- [Jenkins CLI](#)
Accede y administra Jenkins desde la consola, o desde scripts
- [Consola de scripts](#)
Ejecutar script para la administración, diagnóstico y solución de problemas.
- [Administrar Nodos](#)
Añadir, borrar, gestionar y monitorizar los nodos sobre los que Jenkins ejecuta tareas.
- [Gestión de credenciales](#)
Crear/borrar/modificar las credenciales que pueden ser usadas por Jenkins y por las tareas que se ejecutan en él para conectar con servicios de terceros
- [Acerca de Jenkins](#)
Eche un vistazo a la información sobre la versión y la licencia.
- [Datos antiguos](#)
Scrub configuration files to remove remnants from old plugins and earlier versions.
- [Gestión de usuarios](#)
Crear/borrar/editar usuarios que puedan utilizar Jenkins
- [In-process Script Approval](#)
Allows a Jenkins administrator to review proposed scripts (written e.g. in Groovy) which run inside the Jenkins process and so could bypass security restrictions.
- [Preparar Jenkins para apagar el contenedor](#)
Detener la ejecución de nuevas tareas para que el sistema pueda apagarse de manera segura.

Desde aquí se puede entre otras cosas acceder a

- Configurar el sistema
- Configurar herramientas
- Instalar Plugins
- Consola de Scripts
- Gestión de usuarios

Ya vienen instalados unos cuantos plugins, que habrá que configurar, como son

- Maven
- Git
- SVN
- CVS

Lo primero será configurar la JDK, para ello entrar en **Administrar Jenkins** → **Configuración del sistema** → **JDK**

JDK

instalaciones de JDK

JDK	
Nombre	<input type="text" value="JDK8"/>
JAVA_HOME	<input type="text" value="C:\Program Files\Java\jdk1.8.0_91"/>
<input type="checkbox"/>	Instalar automáticamente

Borrar JDK

Lo siguiente será configurar **Maven**, para ello entrar en **Administrar Jenkins** → **Configuración del sistema** → **Maven**

Maven

instalaciones de Maven

Maven	
Nombre	<input type="text" value="Maven 3.3.9"/>
MAVEN_HOME	<input type="text" value="D:\utilidades\apache-maven-3.3.9"/>
<input type="checkbox"/>	Instalar automáticamente

Borrar Maven

Añadir Maven

Listado de instalaciones de Maven en este sistema

Lo siguiente a configurar puede ser el servidor de SMTP que se desea emplear para las notificaciones de los eventos

Notificación por correo electrónico

Servidor de correo saliente (SMTP)

sufrido de email por defecto

☐ Probar la configuración enviando un correo de prueba

Avanzado...

De forma nativa **Jenkins** soporta **CVS** y **SVN**, pero no **Git**, por lo que si se quiere trabajar con **Git**, habrá que instalar un plugin, el **Git Plugin**.

<input type="checkbox"/>	CMVC Plugin This plugin integrates CMVC to Hudson.	0.3
<input type="checkbox"/>	Darcs Plugin This plugin integrates Darcs version control system to Jenkins. The plugin requires the Darcs binary (darcs) to be installed on the target machine.	0.3.5
<input type="checkbox"/>	Dimensions Plugin This plugin integrates Hudson with Dimensions , the Serena SCM solution.	0.8.1
<input type="checkbox"/>	File System SCM Use File System as SCM.	1.10
<input checked="" type="checkbox"/>	Git Plugin This plugin allows use of GIT as a build SCM. Git 1.3.3 or newer is required.	1.1.6
<input type="checkbox"/>	Harvest Plugin This plugin allows you to use CA Harvest as a SCM.	0.4

Una vez instalado y para el correcto funcionamiento de **Git** desde **Jenkins**, se ha de configurar el plugin, para ello se ha de acceder a **Administrar Jenkins** → **Configurar el sistema** → **Git Plugin** y allí

añadir el nombre de usuario y el mail.

The screenshot shows the Jenkins Configuration page with the following sections and fields:

- System Admin e-mail address:** A text field with the value "http://localhost:8080/" and a warning message: "Escriba un nombre de servidor correcto en lugar de 'localhost'". Below it, another field shows "Dirección no configurada todavía <nobody@nowhere>".
- SSH Server:** A section with a radio button for "Arreglado" (selected), a text field, and radio buttons for "Aleatoria" and "Desactivar".
- GitHub:** A section with a dropdown menu for "Add GitHub Server".
- GitHub Enterprise Servers:** A section with an "Añadir" button.
- Plugin de tiempo máximo de ejecución > Acción del paso de ejecución:** A checkbox for "Habilitar acción del paso de ejecución".
- Git plugin:** A section with two text fields: "Global Config user.name Value" (containing "Victor Herrero Cazurro") and "Global Config user.email Value" (containing "victorherreroazurro@gmail.com"). There is also a checkbox for "Create new accounts base on author/commmitter's email".
- Subversion:** A section with a dropdown menu for "Subversion Workspace Version" (set to "1.4") and a text field for "Nombre de exclusión 'revprop'".
- Línea de comandos:** A section with a text field for "Ejecutable para la línea de comandos (shell)".
- Extended E-mail Notification:** A section with two text fields: "SMTP server" and "Default user E-mail suffix".
- Default Content Type:** A dropdown menu set to "Plain Text (text/plain)".

At the bottom, there are two buttons: "Guardar" and "Apply".


2.4. Seguridad y Gestion de usuarios

Por defecto Jenkins permite acceder en modo anonimo a todas las tareas, pudiendo ver la información asociada a ellas, aunque no se permite iniciar la construcción.

La seguridad se puede activar en **Administrar Jenkins/Configuración Global de la Seguridad**, donde se puede elegir la forma de autenticar

- Contenedor de servlets
- LDAP
- Base de datos de Jenkins

Tambien se puede gestionar la autorización, ya que por defecto todos los usuarios autenticados tienen permisos para hacer de todo, pero se pueden establecer planes para todo Jenkins o para los proyectos.

 **Configuración global de la seguridad**

☒ Activar seguridad

Puerto TCP de JNLP para los agentes en los nodos secundarios

☐ Arreglado : ☐ Aleatoria☒ Desactivar

☐ Disable remember me

Control de acceso

Seguridad

☐ Delegar seguridad al contenedor de servlets

☐ LDAP

☒ Usar base de datos de Jenkins

☐ Permitir que los usuarios se registren.

Autorización

☐ Configuración de seguridad

☐ Cualquiera puede hacer cualquier acción

☐ Estrategia de seguridad para el proyecto

☐ Modo 'legacy'

☒ Usuarios autenticados tienen privilegios para todo

☒ Allow anonymous read access

Markup Formatter

Plain text

Treats all input as plain text. HTML unsafe characters like < and & are escaped to their respective character entities.

☒ Prevenir ataques "Cross site request forgery"

Camino (Crumbs)

Plugin Manager

☒ Enable Slave --> Master Access Control


☐ Use browser for metadata download

Rules can be tweaked [here](#)

Guardar


Apply


De establecerse otros criterios de seguridad, será conveniente dar de alta usuarios, para ello se ha de acceder a la seccion **Administrar Jenkins/Gestión de usuarios**


 Jenkins

admin | Desconectar

Jenkins > Usar base de datos de Jenkins



 Volver al Panel de control

 Administrar Jenkins

 Crear un usuario

Usuarios

Estos usuarios pueden entrar en Jenkins. Este es un subconjunto de [esta lista](#), que tambien incluyen usuarios creados automáticamente porque hayan hecho 'commits' a proyectos. Los usuarios creados automáticamente no tienen acceso directo a Jenkins.


ID usuario	Nombre
 admin	admin 

2.5. Tareas (Jobs)

Representan los trabajos que se pretenden automatizar, luego deberán ejecutar los siguientes pasos

- Descarga de fuentes desde el SCM.
- Compilación del código.
- Ejecución de las pruebas.
- Validación de informes.

Es normal que se delegue en una herramienta de gestion de ciclo de vida del proyecto, como Maven, ANT o Gradle, el control de este proceso, aunque existen otras opciones, para crear una tarea de estas características, se ha de seleccionar **Crear un proyecto de estilo libre**.


 Jenkins


admin | Desconectar


Jenkins > 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 slaves. 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 múltiples entornos, ejecutar sobre plataformas concretas, etc.


**External Job**
Este tipo de tareas te permite registrar la ejecución de un proceso externo a Jenkins, incluso en una máquina remota. Está diseñado para usar Jenkins como un panel de control de tu sistema de automatización. Para más información consulta esta [página](#).

**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.

**GitHub Organization**
Scans a GitHub organization (or user account) for all repositories matching some defined markers.

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

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

 Copy from

OK

Lo primero en la creación de la tarea, será definir el origen del código, es decir la conexión con el SCM.

10

Jenkins > Proyecto >

General **Configurar el origen del código fuente** Disparadores de ejecuciones Entorno de ejecución Ejecutar Acciones para ejecutar después.

☐ Desactivar la ejecución
☐ Lanzar ejecuciones concurrentes en caso de ser necesario

Avanzado...

Configurar el origen del código fuente

☐ Ninguno
☐ Git
☒ Subversion

Módulos

Repository URL:

Credentials: Add

Local module directory:

Repository depth:

Ignore externals: ☒

Add module...

Additional Credentials: Add additional credentials...

Check-out Strategy:
 Use 'svn update' whenever possible, making the build faster. But this causes the artifacts from the previous build to remain when a new build starts.

Navegador del repositorio:

Avanzado...

Disparadores de ejecuciones

☐ Lanzar ejecuciones remotas (ejem: desde 'scripts')
☐ Build after other projects are built
☐ Build when a change is pushed to GitHub

Guardar Apply

Se podrá definir un disparador (Trigger) que inicie la ejecución de la tarea, hay varios tipos

- Ejecución temporal empleando una expresion de Cron.
- Comprobar periodicamente si el estado del SCM no ha cambiado, y si cambia construir, se emplea una expresion de Cron.
- Construir cuando haya cambios en Github, este SCM, permite definir un Hook, que establece una comunicación bidireccional entre Github y Jenkins, pudiendo Github indicar cuando hay cambios para que Jenkins construya.
- Construir cuando otros proyectos se construyan.

Disparadores de ejecuciones

☐ Lanzar ejecuciones remotas (ejem: desde 'scripts')
☐ Build after other projects are built
☐ Build when a change is pushed to GitHub
☐ Consultar repositorio (SCM)
☐ Ejecutar periódicamente

Habrá que definir una tarea o conjunto de tareas a realizar una vez se tenga el código fuente, una de las más habituales es una tarea Maven.

Ejecutar

Ejecutar tareas 'maven' de nivel superior

Version de Maven

Maven 3.3.9

Goles

install

Avanzado...

Añadir un nuevo paso

Ejecutar Ant

Ejecutar línea de comandos (shell)

Ejecutar tareas 'maven' de nivel superior

Ejecutar un comando de Windows

Invoke Gradle script

Process Job DSLs

Set build status to "pending" on GitHub commit

Se pueden definir pasos posteriores a la tarea, como por ejemplo la **publicación de los resultados de los test de JUnit**

Acciones para ejecutar después.

Publicar los resultados de tests JUnit

Ficheros XML con los informes de tests

**/target/surefire-reports/*.xml

El atributo '@includes' de la etiqueta 'fileset' especifica dónde están los ficheros XML generados, por ejemplo: 'myproject/target/test-reports/*.xml'. El directorio base para la etiqueta 'fileset' es el directorio [raíz](#) del proyecto

☐ Guardar la salida estándar y de error aunque sea muy larga.

Health report amplification factor

1,0

1% failing tests scores as 99% health. 5% failing tests scores as 95% health

Borrar

También se puede configurar el archivado de los artefactos producidos por la tarea

Guardar los archivos generados

Ficheros para guardar

**/target*.jar

Avanzado...

Borrar

O la publicación de los **Javadoc generados**

Publicar Javadoc

Directorio para los javadoc

target/site/apidocs

Directorio relativo al 'workspace' del proyecto, ejemplo: 'myproject/build/javadoc'

☐ Conservar los javadoc para todas las ejecuciones correctas

Borrar

2.5.1. Resultado de la ejecución

La ejecución de la tarea, se mostrará con un círculo de color

- azul. La ejecución ha ido bien.

- amarillo. Ha habido un problema con los Test o con la Cobertura.
- rojo. Ha habido un error en ejecución.

La ejecución de la tarea puede ofrecer como resultado

- Sol (0/5)
- Nubes (1-2/5)
- Lluvia (3-4/5)
- Tormenta (5-5)

Todo +						
S	W	Name ↓	Último éxito	Último fallo	Última duración	
		CleanDailyReleases	2 días 12 Hor (#12)	N/D	10 Seg	
		Deploy Daily Build	4 Hor 47 Min (#104)	6 días 0 Hor (#100)	2 Min 13 Seg	
		Liquibase-Update	4 Hor 48 Min (#123)	4 Hor 48 Min (#123)	9,9 Seg	
		Liquibase-Update-SQL	4 Hor 48 Min (#95)	1 Mes 17 días (#42)	31 Seg	
		Power Desk Web	4 Hor 59 Min (#93)	21 Hor (#91)	10 Min	
		Restore-DB-IC	4 Hor 48 Min (#96)	2 Mes 7 días (#7)	2,2 Seg	

2.6. Plugins

2.6.1. Maven Plugin

Se ha de configurar Maven en Jenkins, para ello se ha de acceder a **Administrar Jenkins/Global Tool Configuration** y allí crear una nueva configuracion de Maven, indicando o bien **MAVEN_HOME**, o bien que se descargue la versión de Maven deseada.

Jenkins > Global Tool Configuration

Name: Default

Path to Git executable: git.exe

☐ Instalar automáticamente

[Delete Git](#)

[Add Git](#)

description

Gradle

instalaciones de Gradle

[Añadir Gradle](#)

Listado de instalaciones de Gradle en este sistema

Ant

instalaciones de Ant

[Añadir Ant](#)

Listado de instalaciones de Ant en este sistema

Maven

instalaciones de Maven

Maven

Nombre: Maven 3.3.9

MAVEN_HOME: D:\utilidades\apache-maven-3.3.9

☒ Instalar automáticamente

[Instalar desde Apache](#)

Versión: 3.3.9

[Borrar un instalador](#)

[Añadir un instalador](#)

[Añadir Maven](#)

Listado de instalaciones de Maven en este sistema

[Save](#) [Apply](#)

Página generada: 27-abr-2016 20:59:31 CEST [Jenkins ver 2.0](#)



Una vez configurado Maven, se ha de asegurar que los proyectos emplean esta configuración, en versiones de Jenkins ocurre que se selecciona la version de Maven por defecto y de esta forma no funciona la construcción

Jenkins > Proyecto >

General

Configurar el origen del código fuente

Disparadores de ejecuciones

Entorno de ejecución

Ejecutar

Acciones para ejecutar después.

Disparadores de ejecuciones

☐ Lanzar ejecuciones remotas (ejem: desde 'scripts')
 ☐ Build after other projects are built
 ☐ Build when a change is pushed to GitHub
 ☐ Consultar repositorio (SCM)
 ☐ Ejecutar periódicamente

Entorno de ejecución

☐ Delete workspace before build starts
 ☐ Abortar la ejecución si se atasca
 ☐ Add timestamps to the Console Output
 ☐ Use secret text(s) or file(s)

Ejecutar

Ejecutar tareas 'maven' de nivel superior

Version de Maven

Maven 3.3.9

Goles

(Por defecto)

Maven 3.3.9

Avanzado...

Añadir un nuevo paso

Acciones para ejecutar después.

Añadir una acción

Guardar

Apply

Página generada: 27-abr-2016 21:01:54 CEST

REST API

Jenkins ver. 2.0

2.6.2. Plugin Sonarqube

Es un pugin que permite conectar Jenkins con Sonar.

Jenkins

búsqueda

admin | Desconectar

Jenkins > Gestor de plugins

Volver al Panel de control

Administrar Jenkins

Actualizaciones disponibles

Todos los plugins

Plugins instalados

Configuración avanzada

Filtrar: sonarqube

Instalar	Nombre	Versión
<input type="checkbox"/>	Mashup Portlets Additional portlets: 'Generic JS Portlet' flexibly displays any content from another source. 'Recent Changes' and 'Test Results' show useful job information that would otherwise only be available by drilling down into the job. The 'SonarQube Portlet' shows important issues from Sonar directly in Jenkins.	1.0.6
<input checked="" type="checkbox"/>	SonarQube Plugin This plugin allow easy integration of SonarQube™ , the open source platform for Continuous Inspection of code quality.	2.4

Instalar sin reiniciar

Descargar ahora e instalar después de reiniciar

Update information obtained: 1 día 0 Hor ago

Comprobar ahora

Se ha de configurar el servidor Sonar en **Administrar Jenkins/Configurar el sistema**

SonarQube servers

Environment variables

☒ Enable injection of SonarQube server configuration as build environment variables

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

Instalaciones de SonarQube

Name

Sonar local

URL del servidor

localhost:9000

Server version

5.3 or higher

Server authentication token

SonarQube account login

SonarQube account password

Por defecto es http://localhost:9000

Configuration fields depend on the SonarQube server version.

SonarQube authentication token. Mandatory when anonymous access is disabled.

SonarQube account used to perform analysis. Mandatory when anonymous access is disabled. No longer used since SonarQube 5.3.

SonarQube account used to perform analysis. Mandatory when anonymous access is disabled. No longer used since SonarQube 5.3.

Avanzado...

Delete SonarQube

Add SonarQube

Listado de instalaciones SonarQube

Se ha de configurar el Sonarqube Scanner en **Global Tool Configuration**

SonarQube Scanner

Instalaciones de SonarQube Scanner

SonarQube Scanner

Name

Sonarqube scanner

☒ Instalar automáticamente

Install from Maven Central

Versión

SonarQube Scanner 2.5.1

Añadir un instalador

Añadir SonarQube Scanner

Listado de instalaciones de SonarQube Scanner en este sistema

Borrar un instalador

Borrar SonarQube Scanner

Este plugin proporciona un nuevo ejecutable a incluir en la ejecución de la tarea.

Ejecutar

Ejecutar tareas 'maven' de nivel superior

Version de Maven

Maven 3.3.9

Goles

install

Avanzado...

Execute SonarQube Scanner

Task to run

JDK

(Inherit From Job)

JDK to be used for this sonar analysis

Path to project properties

Analysis properties

Additional arguments

JVM Options

Añadir un nuevo paso ▾

2.6.3. Cobertura Plugin

Plugin que permite visualizar los resultados del analisis estatico de código que realiza Cobertura, así como la definición de los limites en los cuales se considera una Cobertura aceptable.

☒ Publish Cobertura Coverage Report

Cobertura xml report pattern

This is a file name pattern that can be used to locate the cobertura xml report files (for example with Maven2 use ****/target/site/cobertura/coverage.xml**). The path is relative to the module root unless you have configured your SCM with multiple modules, in which case it is relative to the workspace root. Note that the module root is SCM-specific, and may not be the same as the workspace root. Cobertura must be configured to generate XML reports for this plugin to function.

Consider only stable builds ☐

Include only stable builds, i.e. exclude unstable and failed ones.

Coverage Metric Targets

Conditionals	<input type="button" value="Delete"/>	98	75	75
Lines	<input type="button" value="Delete"/>	98	75	75
Methods	<input type="button" value="Delete"/>	100	80	80
Packages	<input type="button" value="Delete"/>	100	95	95
<input type="button" value="Add"/>				

Configure health reporting thresholds.

For the row, leave blank to use the default value (i.e. 80).

For the and rows, leave blank to use the default values (i.e. 0).

2.6.4. Deploy To Container Plugin

Es un **Plugin**, que permite desplegar una aplicación empresarial en un servidor de aplicaciones, como

Tomcat, JBoss, WebSphere, Weblogic, ...

☒ Deploy war/ear to a container

WAR/EAR files

Container

Manager user name

Manager password

Tomcat URL

2.6.5. Copy Artifact plugin

Permite copiar uno o varios ficheros de un **Job** a otro.

Build

Copy artifacts from another project

Project name

Which build

☐ Stable build only

Artifacts to copy

Target directory

☒ Flatten directories ☐ Optional

Delete

2.6.6. Disk Usage Plugin

Permite monitorizar el uso del disco.

2.6.7. Backup Plugin

Aunque el Backup de Jenkins es facil de realiza, basta con hacer el backup de la carpeta **JENKINS_HOME**, este plugin facilita la tarea, permitiendo configurar que partes del directorio se van a guardar, ya que por ejemplo la carpeta de **workspace** es una carpeta innecesaria a la hora del backup y que puede ocupar bastante, ya que contiene el proyecto entero.

2.6.8. Dependency Graph Viewer Plugin

Permite visualizar las dependencias configuradas entre los **Jobs** definidos en **Jenkins**.

Este plugin emplea **graphviz**, el cual habra que tener instalado en el equipo.

2.6.9. Maven Release Plug-in

Permite publicar una release empleando el plugin de release de **Maven**, siendo configurado por **Jenkins**

2.6.10. Plugin Job DSL

Este plugin, permite definir la tarea como un script DSL de Groovy, se puede encontrar un tutorial que crea una tarea a partir de una tarea de tipo Job DSL [aquí](#)

2.6.11. Plugin Project Template

Permite reutilizar las configuraciones de un proyecto en otro



Dentro de la solución de pago de **CloudBees**, se proporcionan plugins para crear plantillas no solo de **Jobs**, sino tambien incluso de **Builds**

2.6.12. Plugin Pipeline

Permite definir un script con las fases de un Job.

El Script se escribe en groovy

2.7. Scripting con Jenkins CLI

Descargar el siguiente jar

```
http://localhost:8080/jnlpJars/jenkins-cli.jar
```

Ejecutar el comando **login**, para que CLI recuerde el login hasta que se cierre la sesion.

```
java -jar jenkins-cli.jar -s http://localhost:8080 login --username admin --password admin
```

Ejecutar el comando **groovy** indicando el path de un fichero Groovy, para ejecutar scripts de **Groovy**.

```
java -jar jenkins-cli.jar -s http://localhost:8080 groovy fichero_script.groovy
```

Un script de Groovy de ejemplo, que recorre los fichers de la instalación, indicando aquellos de gran tamaño podria ser.

```

root = jenkins.model.Jenkins.instance.getRootDir()
count = 0
size = 0
maxsize = 1024*1024*32
root.eachFileRecurse() { file ->
    count++
    size += file.size();
    if (file.size() > maxsize) {
        println "Thinking about deleting: ${file.getPath()}"
    }
}
println "Space used ${size/(1024*1024)} MB Number of files ${count}"

```

Otro script de Groovy de ejemplo, que recorre los **Jobs** creados en Jenkins, comprobando si la última construcción correcta es del año en curso.

```

def warning = '<font color=\''red\'>[ARCHIVE]</font> '
def now = new Date()

for (job in hudson.model.Hudson.instance.items) {
    println "\nName: ${job.name}"
    Run lastSuccessfulBuild = job.getLastSuccessfulBuild()
    if (lastSuccessfulBuild != null) {
        def time = lastSuccessfulBuild.getTimestamp().getTime()
        if (now.year.equals(time.year)){
            println("Project has same year as build");
        }else {
            if (job.description.startsWith(warning)){
                println("Description has already been changed");
            }else{
                job.setDescription("${warning}${job.description}")
            }
        }
    }
}
}


```

Ejecutar el comando **logout**, para que CLI olvide el login.

```
java -jar jenkins-cli.jar -s http://localhost:8080 logout.
```

2.8. Consola de Script Integrada

En la administración de Jenkins, hay una consola integrada, que permite ejecutar scripts de Groovy.

 Jenkins

admin | Desconectar

Jenkins

Nueva Tarea

Personas

Historial de trabajos

Administrar Jenkins

Mis vistas

Credentials

Trabajos en la cola


No hay trabajos en la cola


Estado del ejecutor de construcciones


1 Inactivo


2 Inactivo

Administrar Jenkins

 [Configurar el Sistema](#)
Configurar variables globales y rutas.


 [Configuración global de la seguridad](#)
Seguridad en Jenkins. Define quién tiene acceso al sistema (autenticación) y qué puede hacer (autorización)


 [Global Tool Configuration](#)
Configure tools, their locations and automatic installers.

 [Actualizar configuración desde el disco duro.](#)
Descartar todos los datos cargados en memoria y actualizar todo nuevamente desde los ficheros del sistema. Útil cuando se modifican ficheros de configuración directamente en el disco duro.


 [Administrar Plugins](#)
Añadir, borrar, desactivar y activar plugins que extienden la funcionalidad de Jenkins.


 [Información del sistema](#)
Muestra información del entorno que puedan ayudar a la solución de problemas.


 [System Log](#)
El log del sistema captura la salidad de la clase java.util.logging en todo lo relacionado con Jenkins.


 [Estadísticas de Carga](#)
Comprobar la utilización de los recursos y comprobar si es necesario añadir nuevos nodos para la ejecución de tareas.


 [Jenkins CLI](#)
Accede y administra Jenkins desde la consola, o desde scripts


 [Consola de scripts](#)
Ejecutar script para la administración, diagnóstico y solución de problemas.


 [Administrar Nodos](#)
Añadir, borrar, gestionar y monitorizar los nodos sobre los que Jenkins ejecuta tareas.


 [Gestión de credenciales](#)
Crear/borrar/modificar las credenciales que pueden ser usadas por Jenkins y por las tareas que se ejecutan en él para conectar con servicios de terceros


 [Acerca de Jenkins](#)
Eche un vistazo a la información sobre la versión y la licencia.

 [Datos antiguos](#)
Scrub configuration files to remove remnants from old plugins and earlier versions.

 [Gestión de usuarios](#)
Crear/borrar/editar usuarios que puedan utilizar Jenkins

 [In-process Script Approval](#)
Allows a Jenkins administrator to review proposed scripts (written e.g. in Groovy) which run inside the Jenkins process and so could bypass security restrictions.

 [Preparar Jenkins para apagar el contenedor](#)
Detener la ejecución de nuevas tareas para que el sistema pueda apagarse de manera segura.

 Jenkins

admin | Desconectar

Jenkins

Nueva Tarea

Personas

Historial de trabajos

Administrar Jenkins

Mis vistas

Credentials


Trabajos en la cola

No hay trabajos en la cola

Estado del ejecutor de construcciones

1 Inactivo

2 Inactivo



Consola de scripts

Escribe un 'script' [Groovy script](#) y ejecutalo en el servidor. Es útil para depurar e investigar problemas. Usa 'println' para ver la salida (si usas System.out, se escribirá en la salida 'stdout' del servidor, lo que es más difícil de visualizar). Ejemplo:

```
println(Jenkins.instance.pluginManager.plugins)
```

Todas las clases de todos los plugins son visibles. Los paquetes: jenkins.*, jenkins.model.*, hudson.*, y hudson.model.*, se importarán automáticamente.

```
1 root = jenkins.model.Jenkins.instance.getRootDir()
2 count = 0
3 size = 0
4 maxSize=1024*1024*32
5 root.eachFileRecurse() { file ->
6     count++
7     size+=file.size();
8     if (file.size() >maxsize) {
9         println "Thinking about deleting: ${file.getPath()}"
10    }
11 }
12 println "Space used ${size/(1024*1024)} MB Number of files ${count}"
```

Ejecutar

Resultado

```
Thinking about deleting: D:\utilidades\jenkins\jenkins.war
Thinking about deleting: D:\utilidades\jenkins\jre\bin\jfxwebkit.dll
Thinking about deleting: D:\utilidades\jenkins\jre\lib\rt.jar
Space used 384.2653446197509765625 MB Number of files 3351
```

2.9. API de acceso remoto

Desde el API de acceso remoto, se puede entre otras cosas,

- Lanzar un build de un tarea.
- Deshabilitar/Habilitar una tarea
- Borrar una tarea.


Se puede acceder desde


```
http://localhost:8080/job/Proyecto/api/
```


2.10. Ejecución parametrizada


Se pueden definir variables en la construcción de las tareas en Jenkins, que permitan cambiar el comportamiento de la construcción en cada momento.


Para ello se ha de definir el parametro en la sección inicial **Esta ejecución debe parametrizarse**.

☒ Esta ejecución debe parametrizarse 

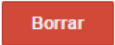
Parámetro de cadena 


Nombre 

Valor por defecto 

Descripción 

[Plain text] [Visualizar](#)





Una vez definido el parametro, este se puede incluir en cualquier zona de la configuración, empleando \$

Proyecto

Fichero POM raíz 

Goles y opciones 



Lo mas habitual con **Tareas Maven** es emplear los parametros para seleccionar el **profile** de Maven


```
mvn clean install -P produccion
```

Se puede lanzar la Tarea parametrizada de forma remota, indicando

```
http://localhost:8080/job/MiTarea/buildWithParameters?GOAL=clean
```

Los parametros empleados en cada una de las ejecuciones de la tarea, se almacenan en la propia Tarea



2.11. Tarea Multiconfiguración

Este tipo de proyectos incluyen la **Matriz de Configuración**, que permite definir un parametro de configuración, con los posibles valores que puede tomar, y por cada uno de los valores definidos, se creará una **SubTarea**.

Por defecto las **SubTareas** se ejecutarán de forma paralela, pero en ocasiones esto no será recomendable, ya que pueden necesitar el mismo recurso de forma simultanea, y el código puede no contemplar la concurrencia (porque no tenga sentido, son en realidad el mismo proyecto corriendo con distintas configuraciones), en este caso, se puede marcar **Run each configuration sequentially**, que ejecutará las **Subtareas** de forma secuencial.

Si se define mas de un **Eje** (variable), se ejecutarán todas las posibles combinaciones con los valores de los **Ejes**, sino se desea que se ejecuten todas las posibles combinaciones, se deberá definir un **Filtro de combinación**

Los **Filtros de combinación** definen lo que se ha de cumplir para que se cree una **SubTarea**

```
(browser=="firefox") || (browser=="iexplorer" && os=="windows") || (browser=="chrome" && os != "linux")
```

Los **Ejes** definidos, se pasan como parametros Maven a la construcción (-D<nombre del parametro>=<valor del parametro>), además de poder ser empleados en la configuración de la **Tarea** de **Jenkins**, ya que son parametros de Jenkins (\$<nombre del parametro>).

2.12. Dependencias entre proyectos

Dentro de la configuración de una **Tarea**, se puede indicar que se ejecute otra **Tarea** al finalizar la actual, para ello se acude **Acciones a ejecutar después** y se incluye una referencia al proyecto hijo.

Acciones para ejecutar después.

⌵ Ejecutar otros proyectos



Proyectos a ejecutar

MiTarea

- ☒ Trigger only if build is stable
- ☐ Lanzar incluso si el resultado de la ejecución fué inestable.
- ☐ Lanzar incluso si la ejecución acabó con errores.

Borrar

Las **Tareas Hijas**, se ejecutarán dependiendo del resultado de la ejecución de la **Tarea Padre** y de la configuración establecida, pudiendo ser esta:

- Lanzar solo si la ejecución es estable.
- Lanzar aunque la ejecución no sea estable.
- Lanzar aunque la ejecución haya finalizado con errores.

2.13. Ejecución Distribuida

Se pueden definir nodos secundarios sobre los que delegar la ejecución de las tareas, para ello, se ha de definir el nodo secundario en el nodo principal desde **Administrar Jenkins** → **Administrar Nodos** → **Nuevo nodo**

Lo primero es indicar el tipo de nodo, solo podrá ser **Pasivo**.

Nombre del nodo

Esclavo-Windows

☒ Secundario pasivo

Añadir un esclavo pasivo a Jenkins. Es llamado 'pasivo' porque Jenkins no provee ningún tipo de integración de alto nivel con estos esclavos, como pueda ser aprovisionamiento dinámico. Selecciona este tipo si no hay ningún otro tipo mas adecuado. Por ejemplo cuando se añaden maquinas físicas o virtuales gestionadas desde fuera de Jenkins, etc.

☐ Copiar un nodo existente

Copiar desde

OK

Una vez definido, se ha de configurar indicando:

- Numero de ejecutores.
- Directorio Raiz remoto.
- Cuando usar.


- Usar tanto como sea posible
- Usar solo con tareas asociadas directamente a el.
- Modo de ejecución.
 - Arrancar agente remotos Linux, via SSH.
 - Arrancar con un comando desde el nodo principal.
 - Ejecutar empelando JNLP
 - Permitir al esclavo que se inicie como servicio windows
- Disponibilidad.
 - Mantener el nodo en linea todo lo que sa posible.
 - Poner en linea cuando se necesite.
 - Programar cuando esta en linea.

En Windows se suele emplear la opcion de **Modo de ejecucion** la de **Ejecutar empleando JNLP**, para arrancarlo, se ha de ejecutar

```
java -jar slave.jar -jnlpUrl http://localhost:8081/computer/<Nombre de esclavo>/slave-agent.jnlp
```

Donde el fichero **slave.jar** esta en %JENKINS_HOME%\war\WEB-INF\slave.jar.

Una vez arrancado, se vera como sincronizado

S	Nombre ↓	Arquitectura	Diferencia entre los relojes	Espacio de disco libre	Espacio de intercambio libre	Espacio temporal libre	Tiempo de respuesta
	Esclavo	Windows 10 (amd64)	Sincronizados	N/A	4,49 GB	142,94 GB	4041ms 
	principal	Windows 10 (amd64)	Sincronizados	142,94 GB	4,49 GB	142,94 GB	0ms 
Data obtained		12 Min	12 Min	12 Min	12 Min	12 Min	12 Min

Actualizar el estado

El siguiente paso será configurarlo para que se ejecuten las tareas en el, por un lado habra que configurar el **Nodo** con etiquetas, que definan para que se ha de emplear.

Nombre	<input type="text" value="Esclavo"/>	
Descripción	<input type="text"/>	
Número de ejecutores	<input type="text" value="1"/>	
Directorio raiz remoto	<input type="text" value="C:\slave"/>	
Etiquetas	<input type="text" value="performance integration-test"/>	

Y por otro, en las tareas, activando la opción **Restringir dónde se puede ejecutar este proyecto**,

indicar las etiquetas que indicaran en que nodo se ha de ejecutar la tarea.

image::jenkins_nodo_esclavo_seleccion_de_nodo_en_tarea_por_etiqueta.png[]

En este campo, se pueden emplear expresiones booleanas como las siguientes

```
performance //Nodos con la etiqueta performance
```

```
!performance //Nodos sin la etiqueta performance
```

```
linux && postgres //Nodos con las etiquetas linux y postgres
```

```
"Windows 7" || "Windows XP" //Nodos con las etiquetas "Windows 7" o "Windows XP"
```

```
windows -> "Windows 7" //Si existe la etiqueta "windows", debe existir la etiqueta "Windows 7"
```

```
windows <-> "Windows 7" //Si existe la etiqueta "windows", debe existir la etiqueta "Windows 7", pero sino existe windows, tampoco puede existir "Windows 7"
```