

Integración y entrega continua de código.

En muchas industrias, el éxito de una organización depende en gran parte de la capacidad de su software. La web, el cómputo móvil y las aplicaciones embebidas definen la forma en que los clientes perciben a una marca. La tolerancia a fallas por parte de los usuarios es cada vez menor, el no cumplir con sus expectativas de servicio es un factor determinante para el auge o fracaso de cualquier empresa.

Uno de los principios de la metodología *Agile* nos menciona que es necesario “satisfacer al cliente por medio de la entrega temprana y continua de software valioso”, lo cual es posible acelerando los ciclos de retroalimentación, alineando frecuentemente el software construido con las necesidades del cliente o del mercado.

DevOps surge como un modelo de operación que acerca a todos los involucrados en la cadena de producción de software para colaborativamente perseguir y alcanzar metas de negocio conjuntas, tales como entregar un nuevo producto de alta calidad que opere de manera estable. A través de estandarizar procesos, configuración de versiones y aumentar la colaboración; *DevOps* fortalece a los equipos, habilita la confianza y transparenta la coordinación entre ellos para trabajar juntos de manera eficaz y eficiente, incrementando la velocidad y predictibilidad.

La Integración Continua (*CI* por sus siglas en inglés) es una piedra angular en la práctica actual de desarrollo de software, mediante la cual los *developers* combinan los cambios en el código en un repositorio central de forma periódica, tras ello, versiones y pruebas automáticas son ejecutadas; se refiere en su mayoría a la fase de creación o integración del proceso de publicación de software. Los objetivos clave de la integración continua consisten en encontrar y arreglar errores con mayor rapidez, mejorar la calidad del software y reducir el tiempo que se tarda en validar y publicar nuevas actualizaciones del mismo.

La Entrega Continua (*CD*) vendría siendo el siguiente paso, después de la Integración Continua. De una forma sencilla implica que todo cambio subido al repositorio y cuyas pruebas sean exitosas, pasarán a un servidor, donde el conjunto de todos los cambios será compilado, probado y verificado. Al final, el servidor de pruebas indica si éstas fueron aprobadas o no, en cuyo caso el mismo sistema debe notificar del resultado a los equipos involucrados. Es importante no confundir éste concepto con el de Despliegue Continuo, el cual, además de generar un artefacto (*build*) exitoso, se encarga de propagarlo en uno o mas de los diversos ambientes que estén siendo administrados, por ejemplo: *development*, *test*, *stage*, *production*, etc.

Jenkins es un servidor *CI/CD* de código abierto, basado en el proyecto Hudson. Mediante el uso de distintos *Plugins* su funcionalidad es extendida; es capaz de desencadenar una serie incremental de procesos a partir de un simple *commit* en un repositorio, a esto se le conoce como *Pipeline*, dicha secuencia puede incluir diversas operaciones, tales como: la evaluación de librerías utilizadas dentro del código, la generación del artefacto, la ejecución automática de pruebas para asegurar la calidad del producto, hasta el despliegue del mismo en un servidor de aplicaciones; el envío de mensajes para notificar el estatus de cada una de éstas fases a los interesados (*developers*, *testers*, *managers*, entre otros) es configurable durante toda la cadena de acciones, no solo por correo sino inclusive mediante otro tipo de mensajeros instantáneos, comúnmente empleados dentro de las organizaciones, como puede ser Skype, Slack, HipChat, etc.

Instalación de Jenkins

La documentación oficial <https://jenkins.io/download/> contiene instrucciones y paquetes para cada uno de los distintos sistemas operativos soportados. A continuación, se describe el proceso de instalación para sistemas Red Hat/Fedora/CentOS.

Es necesario añadir el siguiente repositorio e importar la llave desde la terminal.

```
$ sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo
$ sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
```

Java es un requerimiento, la versión dependerá del release de Jenkins que será instalado.

Jenkins	Java
2.54 (2017-04) y nuevos	Java 8
1.612 (2015-05) y nuevos	Java 7

```
$ sudo yum install java-1.8.0-openjdk.x86_64
```

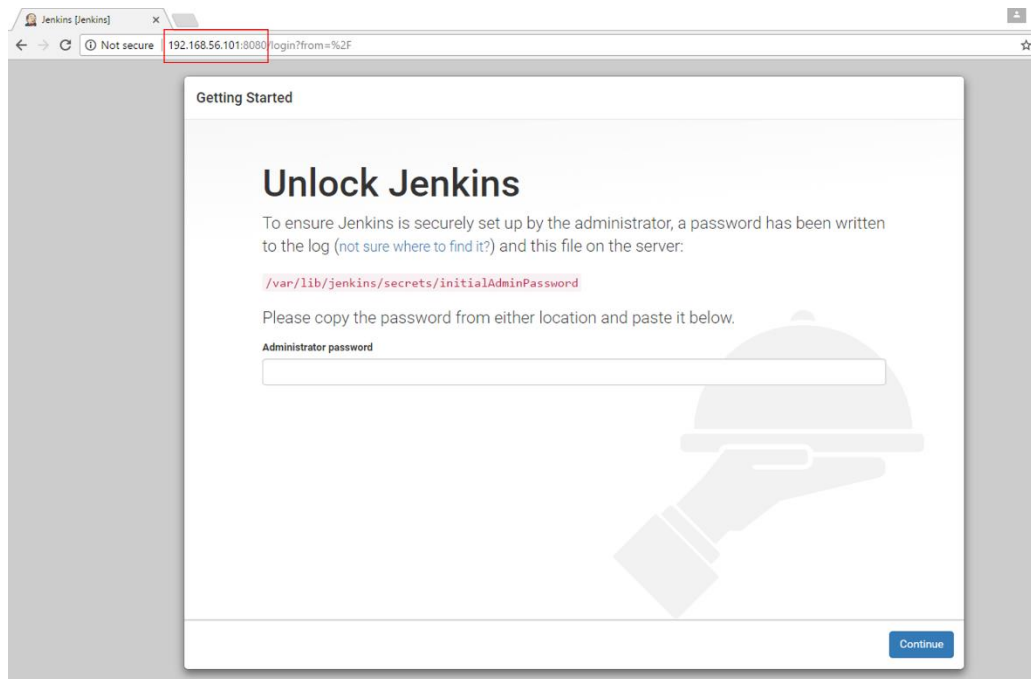
Después, Jenkins puede ser instalado con el comando:

```
$ sudo yum install jenkins
```

Iniciar el servicio y establecer su arranque automático.

```
$ sudo systemctl start jenkins.service
$ sudo systemctl enable jenkins.service
```

Jenkins por default corre en el puerto 8080, su interfaz gráfica es alcanzable desde cualquier navegador web escribiendo su dirección IP seguida del puerto.



La primera vez que Jenkins es iniciado será necesario desbloquearlo, por seguridad solicitará una contraseña generada aleatoriamente, ésta se encuentra en el directorio indicado en letra rojas (la ruta puede variar dependiendo del sistema operativo donde haya sido instalado).

Desde la terminal la contraseña puede ser vista utilizando el comando *cat*, seguido de la ruta del archivo. Es necesario introducir ésta contraseña en la pantalla previa.

```
[root@ip-172-31-1-192 ~]# cat /var/lib/jenkins/secrets/initialAdminPassword
0326848445814872b1f62ff22be042f0
[root@ip-172-31-1-192 ~]#
```

Jenkins ofrece la posibilidad de extender su funcionalidad mediante la instalación de más de 1000 *Plugins* disponibles, los cuales son generados y soportados por la comunidad; Ruby, Android, Java, MySQL y Git, son solo algunos de éstos complementos; el listado y sus principales características puede ser consultado en el siguiente enlace: <https://plugins.jenkins.io/>.

Bienvenido a Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

De ser necesario es posible registrar más de un Administrador; o bien, continuar con la instalación y hacerlo después.

Getting Started

Create First Admin User

Usuario:	<input type="text" value="jenkins"/>
Contraseña:	<input type="password" value="....."/>
Confirma la contraseña:	<input type="password" value="....."/>
Nombre completo:	<input type="text" value="Ana Victoria Rodriguez"/>
Dirección de email:	<input type="text" value="ana_devops@epam.com"/>

Jenkins 2.46.3

[Continue as admin](#)

[Save and Finish](#)

Luego de guardar los cambios, Jenkins estará listo para ser utilizado.

Getting Started

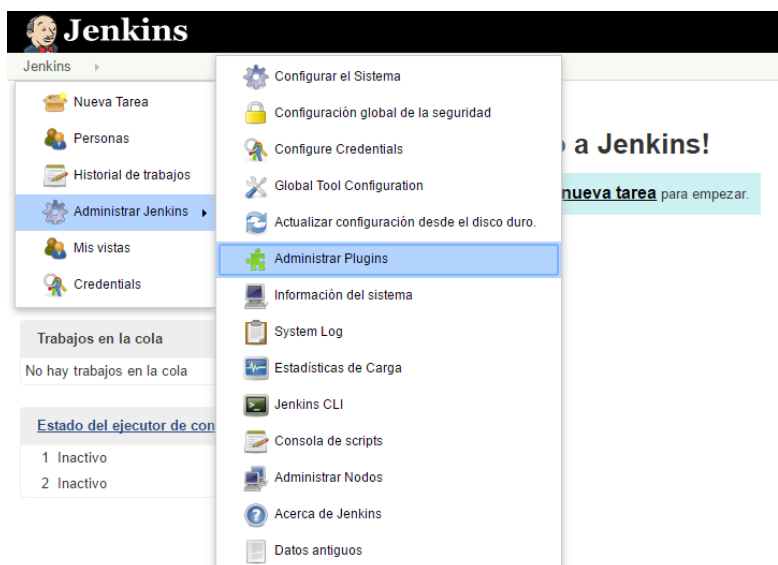
Jenkins is ready!

Your Jenkins setup is complete.

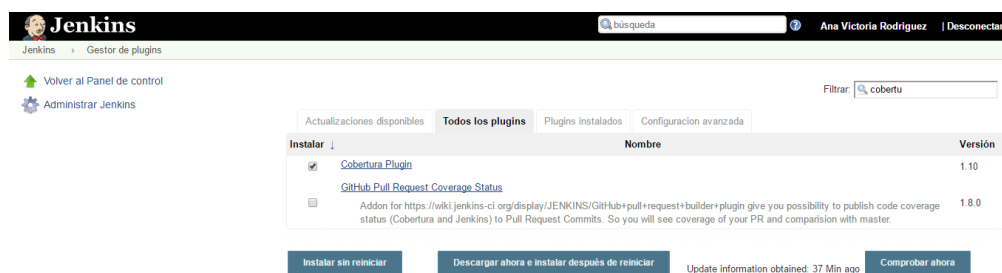
[Start using Jenkins](#)

A continuación, instalaremos el Plugin de Cobertura, el procedimiento es el mismo para instalar cualquier otro.

Hacer clic en el botón **Jenkins > Administrar Jenkins > Administrar Plugins**



En la pantalla siguiente, seleccionando la pestaña **Todos los plugins**, filtrar por “cobertura”, seleccionar **Cobertura Plugin**. Después hacer clic en el botón **Instalar sin reiniciar**.



Una vez instalado el *plugin*, seleccionar la casilla **Reiniciar Jenkins** en la parte inferior. Cuando el servicio se restablezca será necesario firmarse de nuevo en la consola web.



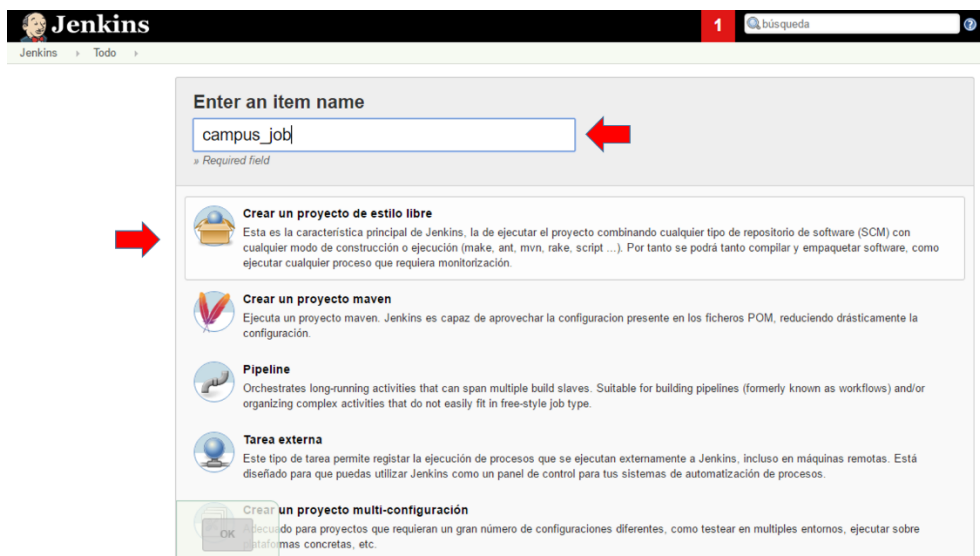
The screenshot shows the Jenkins Update Center interface. At the top, there's a search bar and navigation links for 'Jenkins' and 'Update Center'. Below this, there are three links: 'Volver al Panel de control', 'Administrar Jenkins', and 'Administrar 'plugins''. The main heading is 'Instalando/Actualizando plugins'. Underneath, it shows the status of the 'Cobertura Plugin' as 'Actualizado'. A list of steps is shown: 'Preparación' (Checking internet connectivity, Checking update center connectivity, Success) and 'Cobertura Plugin' (Actualizado). At the bottom, there's a link 'Volver al inicio de la página' and a checkbox 'Reiniciar Jenkins cuando termine la instalación y no queden trabajos en ejecución' which is checked. A red arrow points to this checkbox.

Crear un nuevo Job haciendo clic en **Jenkins > Nueva Tarea**.



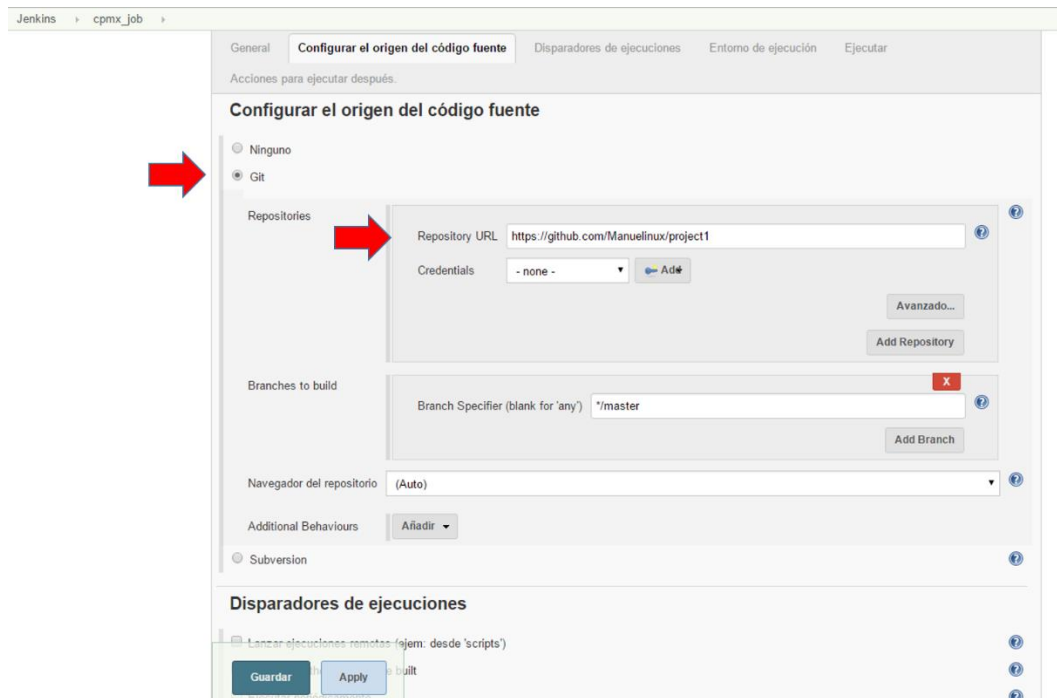
The screenshot shows the Jenkins 'Nueva Tarea' (New Job) dropdown menu. The menu is open, showing options: 'Nueva Tarea', 'Personas', 'Historial de trabajos', 'Relacion entre proyectos', 'Comprobar firma de archivos', 'Administrar Jenkins', 'Mis vistas', and 'Credentials'. A red arrow points to the 'Nueva Tarea' option.

En la siguiente pantalla ingresar el nombre del *job* en la caja de texto, seleccionar **Crear un proyecto de estilo libre** y hacer clic en el botón **OK**.

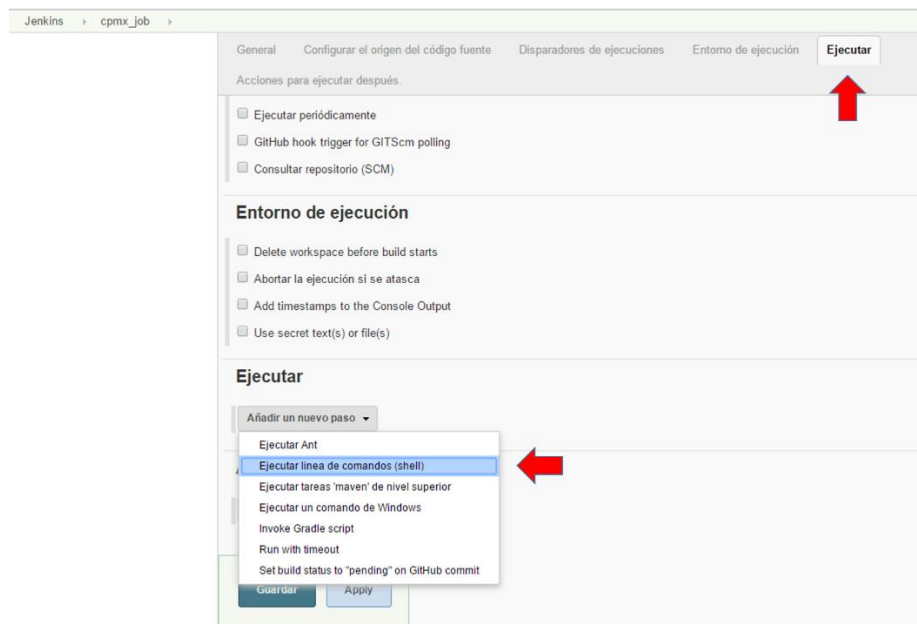


The screenshot shows the Jenkins 'Enter an item name' screen. The text 'campus_job' is entered in the input field. Below the input field, there are four options: 'Crear un proyecto de estilo libre', 'Crear un proyecto maven', 'Pipeline', and 'Tarea externa'. The 'Crear un proyecto de estilo libre' option is selected, and a red arrow points to it. At the bottom, there is an 'OK' button. A red arrow points to the 'OK' button.

Es una buena práctica introducir una breve descripción de cada *job*. Seleccionar la pestaña **Configurar el origen del código fuente**, hacer clic en **Git** e introducir **https://github.com/Manuelinux/project1** en el campo **Repository URL**.



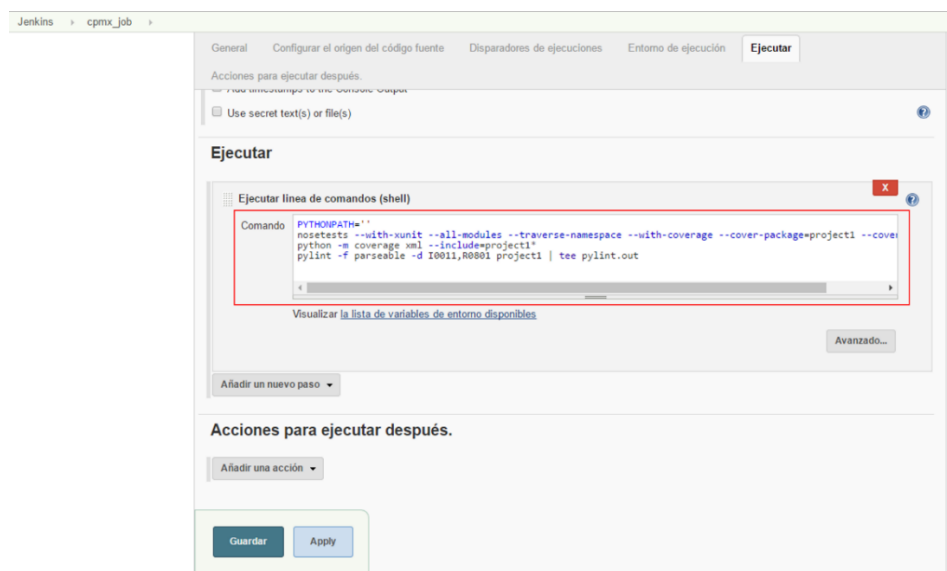
Después, seleccionar **Añadir nuevo paso > Ejecutar línea de comandos (Shell)** en la pestaña **Ejecutar**.



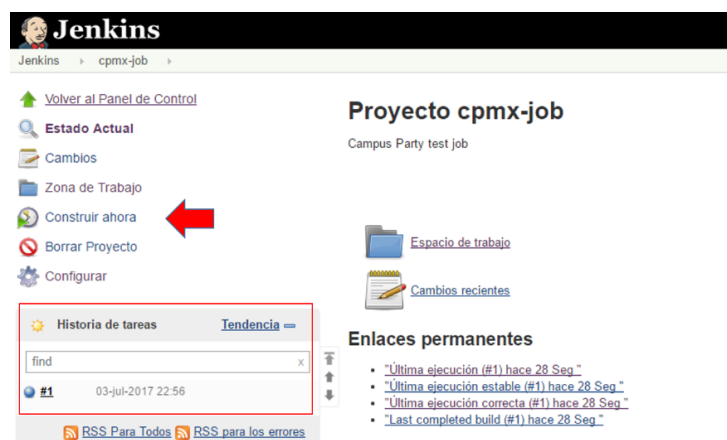
Copiar el código desde <https://raw.githubusercontent.com/Manuelinux/project1/master/README.md>



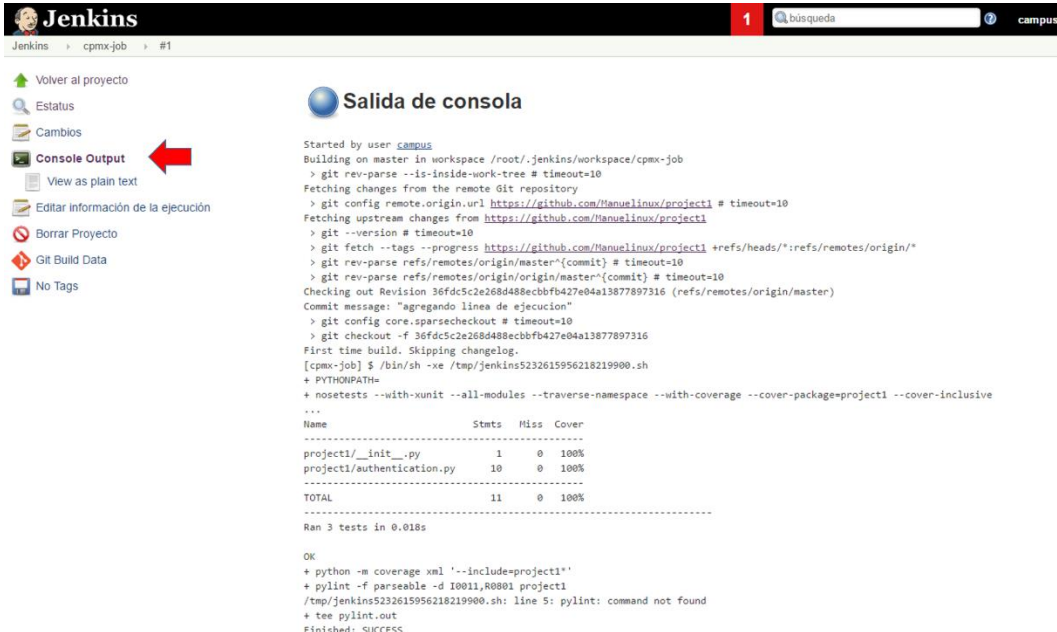
Y pegar en el campo *Comando*, de la sección **Ejecutar**. Finalmente hacer clic en **Apply** y luego **Guardar**.



El *job* está listo para ejecutarse, hacer clic en **Construir ahora** para iniciarlo. El progreso del mismo puede ser monitoreado en la sección **Historial de tareas**.



Un círculo azul a la izquierda del número de *job* indica que finalizó sin errores, un círculo rojo señalará lo opuesto. Es posible verificar los detalles de salida de los comandos ejecutados, haciendo clic en **Console Output**.



The screenshot shows the Jenkins web interface. On the left sidebar, the 'Console Output' link is highlighted with a red arrow. The main panel displays the console output for the 'cpmx-job'.

Salida de consola

```

Started by user campus
Building on master in workspace /root/.jenkins/workspace/cpmx-job
> git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/Manuelinux/project1 # timeout=10
Fetching upstream changes from https://github.com/Manuelinux/project1
> git --version # timeout=10
> git fetch --tags --progress https://github.com/Manuelinux/project1 +refs/heads/*:refs/remotes/origin/*
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision 36fdc5c2e268d488ecbbf427e04a13877897316 (refs/remotes/origin/master)
Commit message: "agregando linea de ejecucion"
> git config core.sparsecheckout # timeout=10
> git checkout -f 36fdc5c2e268d488ecbbf427e04a13877897316
First time build. Skipping changelog.
[cpmx-job] $ /bin/sh -xe /tmp/jenkins5232615956218219900.sh
+ PYTHONPATH=
+ nosetests --with-xunit --all-modules --traverse-namespace --with-coverage --cover-package=project1 --cover-inclusive
...
Name                               Stmts  Miss  Cover
-----
project1/__init__.py                 1     0   100%
project1/authentication.py          10     0   100%
TOTAL                               11     0   100%
-----
Ran 3 tests in 0.018s

OK
+ python -m coverage xml '--include=project1'
+ pylint -f parseable -d I0011,R0801 project1
/tmp/jenkins5232615956218219900.sh: line 5: pylint: command not found
+ tee pylint.out
Finished: SUCCESS

```



Oops!

Ahora crea un nuevo job, **cpmx-fail** el cual fallará durante las pruebas. Sigue el procedimiento previo, solo reemplaza el repositorio por <https://github.com/Manuelinux/projectfail>