Alan Arturo Loya Fabela 15040110

Samanta Castro Hernández 15040102

Beatriz Miranda Miranda 15040112

Jesús Albino Calderón 15040099

|  |  |
| --- | --- |
| Resultado de imagen para jenkins  instalación y configuración de jenkins | equipo |

**¿Qué es la integración continua? (Aquella buena práctica que facilita Jenkins)**

La integración continua es una práctica de desarrollo software donde los miembros del equipo integran su trabajo frecuentemente (como mínimo una vez al día, aunque normalmente se realizan múltiples integraciones diarias).

Cada integración se verifica compilando el código fuente y obteniendo un ejecutable (a esto se le llama build, y debe hacerse de forma automatizada).Además también se pasan las pruebas y métricas de calidad para detectar los errores tan pronto como sea posible.

Es muy recomendable hacer builds periódicamente y comprobar que funcionen correctamente, para conseguir un producto final más fiable, con menos fallos en producción.

Al integrar frecuentemente el código, y con la ayuda de herramientas como Jenkins, puedes saber el estado del software en todo momento. Sabes qué funciona, qué no y qué errores hay.

También puedes monitorizar la calidad del código y su cobertura de pruebas. La integración continua incluso puede ayudarte a reducir la deuda técnica y mantener los costes bajos.

Así que… ¿Cómo conseguir todo esto? ¡Veamos qué es Jenkins!

**Jenkins**

Jenkins es un servidor de integración continua, gratuito, open-source y actualmente uno de los más empleados para esta función. Además es muy fácil de utilizar.

Esta herramienta, proviene de otra similar llamada Hudson, ideada por Kohsuke Kawaguchi, que trabajaba en Sun. Unos años después de que Oracle comprara Sun, la comunidad de Hudson decidió renombrar el proyecto a Jenkins, migrar el código a Github y continuar el trabajo desde ahí. No obstante, Oracle ha seguido desde entonces manteniendo y trabajando en Hudson.

**¿Qué papel juega Jenkins dentro del proceso de integración continua?**

La base de Jenkins son las tareas, donde indicamos qué es lo que hay que hacer en un build. Por ejemplo, podríamos programar una tarea en la que se compruebe el repositorio de control de versiones cada cierto tiempo, y cuando un desarrollador quiera subir su código al control de versiones, este se compile y se ejecuten las pruebas.

Si el resultado no es el esperado o hay algún error, Jenkins notificará al desarrollador, al equipo de QA, por email o cualquier otro medio, para que lo solucione. Si el build es correcto, podremos indicar a Jenkins que intente integrar el código y subirlo al repositorio de control de versiones.

Pero la cosa no queda ahí. Una de las cosas buenas que tiene Jenkins es que además de poder ayudarte a integrar el código periódicamente, puede actuar como herramienta que sirva de enlace en todo el proceso de desarrollo.

Desde Jenkins podrás indicar que se lancen métricas de calidad y visualizar los resultados dentro de la misma herramienta. También podrás ver el resultado de los tests, generar y visualizar la documentación del proyecto o incluso pasar una versión estable del software al entorno de QA para ser probado, a pre-producción o producción.

Por ejemplo, en esta imagen puedes ver una zona del cuadro de mando de Jenkins, donde aparecen las distintas tareas que se han llevado a cabo, la duración, cuándo se produjo el último fallo.

[](https://www.javiergarzas.com/wp-content/uploads/2014/05/jenkins.png)

**Terminando: Cosas a tener en cuenta para llevar a cabo la integración continúa con Jenkins**

En primer lugar, es imprescindible tener un repositorio de control de versiones (mercurial, git, svn, plastic, etc). Todo lo necesario para realizar el build debe estar allí (código, scripts de test, librerías de terceros…).

Sin esto, no podremos sacar el máximo partido a Jenkins, ya que uno de sus puntos fuertes es que es capaz de monitorizar el control de versiones y actuar ante cualquier cambio.

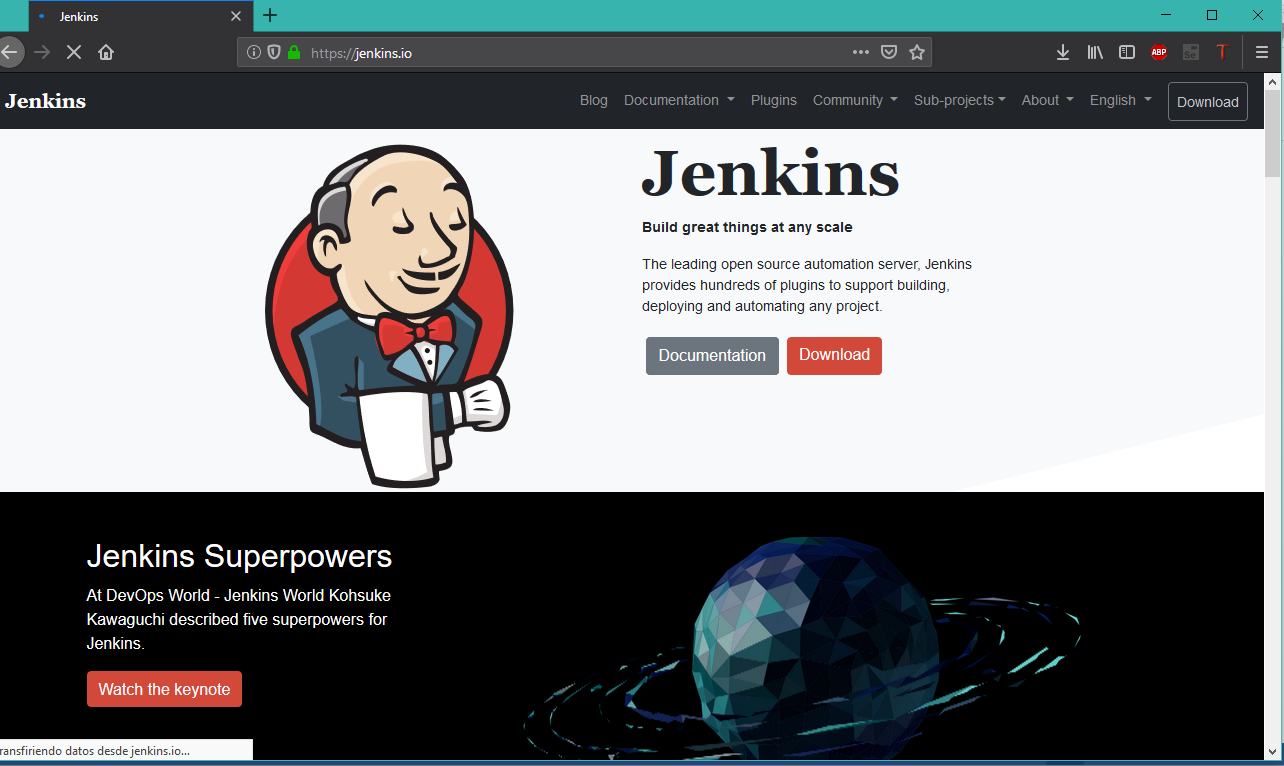
Además, para que la integración continua funcione, es imprescindible que el equipo esté mentalizado y comprometido.

Por ejemplo (y puede que de tanto repetirlo se haga pesado, pero créeme, hay veces que esto no se sigue. Todo el código debe subirse al control de versiones. Los desarrolladores deben subir su trabajo periódicamente al repositorio. Además los proyectos tienen que tener un proceso de build automático, y si un biild falla, lo más prioritario debe ser arreglarlo.

Aun así, y sin dudarlo, con todo lo que nos aporta esta buena práctica, merece la pena llevarla a cabo. Y merece la pena utilizar Jenkins.

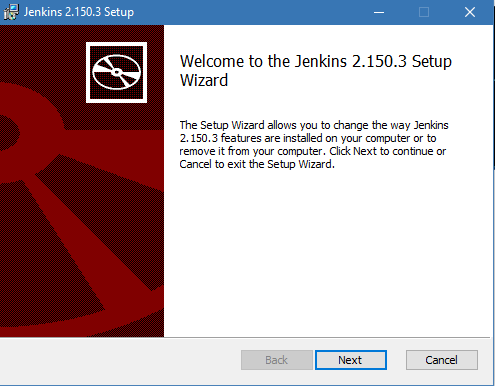
**Descargar Jenkins**

Ir a la página oficial de Jenkins en <https://jenkins.io/>



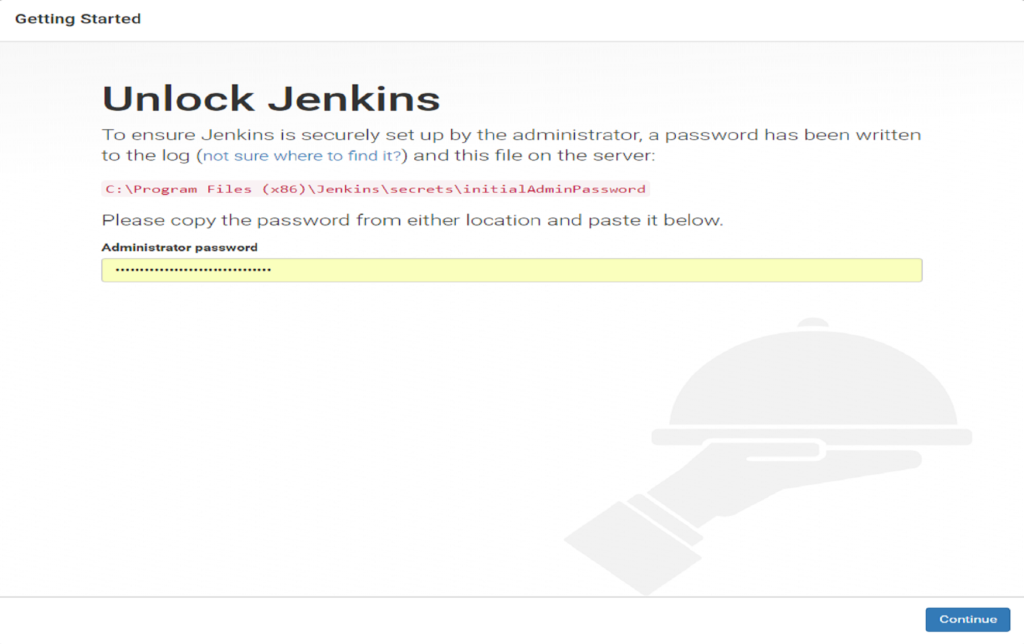
**Instalación de Jenkins**

1. Damos clic en siguiente hasta terminar la instalación.

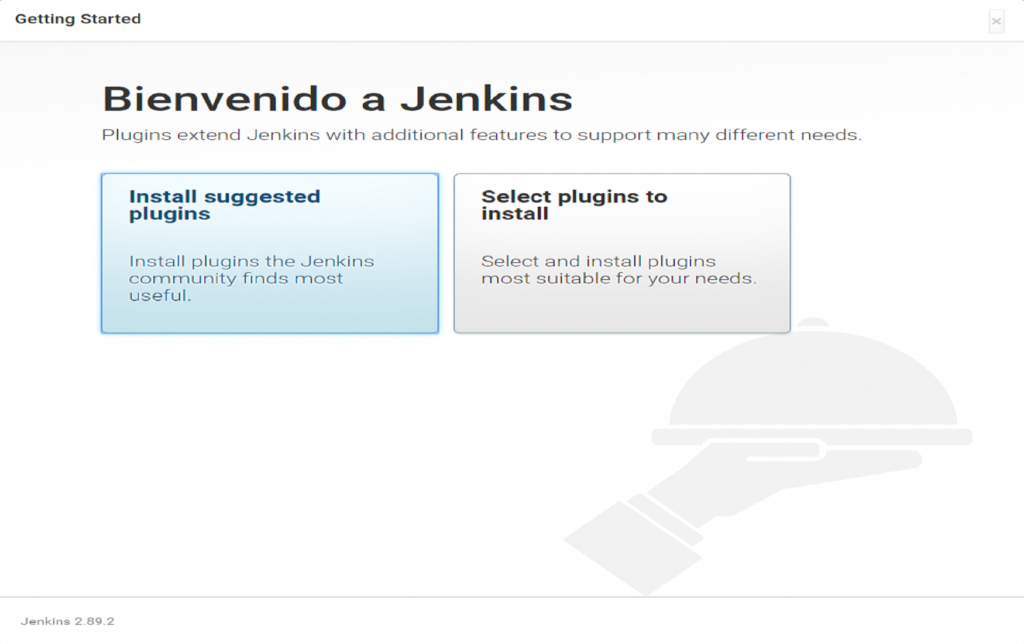
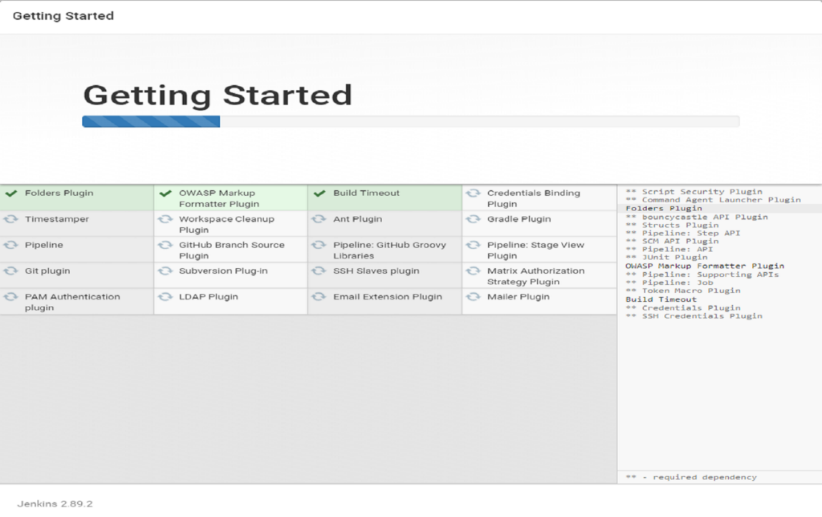


1. Con esto ya tendríamos instalado Jenkins. Para comprobarlo, abrimos un navegador y tecleamos **“http://localhost:8080”**. Lo primero que nos aparece es una pantalla de Jenkins nos indica la ubicación en el disco duro de la clave de administrador generada por defecto.

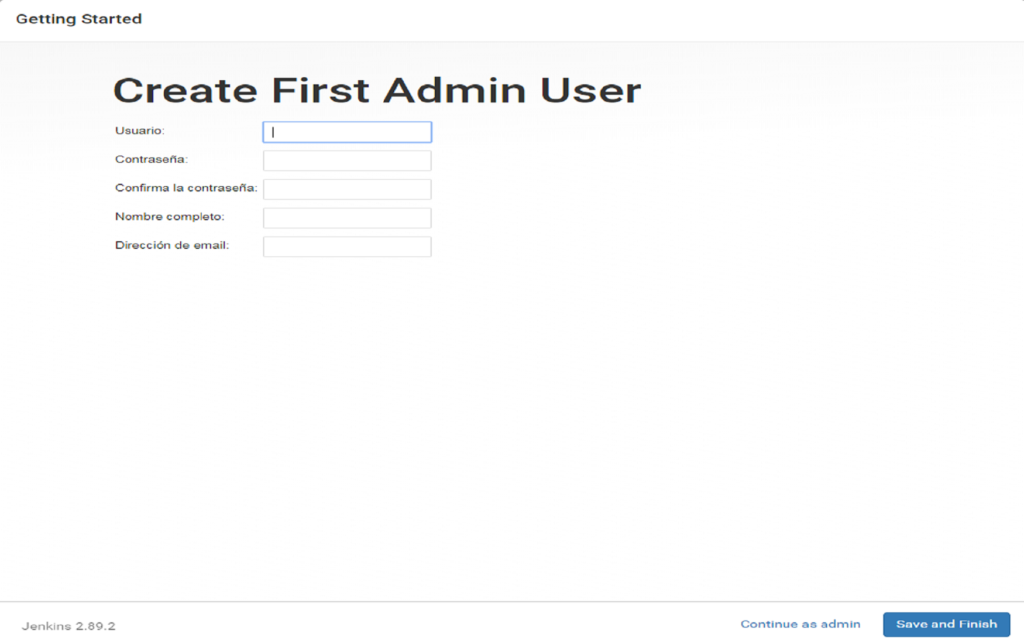
Accedemos a dicha ubicación, copiamos la clave indicada y la pegamos en la cajita donde pone **“Administrator password”**. Pulsamos sobre **“Continue”**. Ahora nos aparece la pantalla de bienvenida de Jenkins.



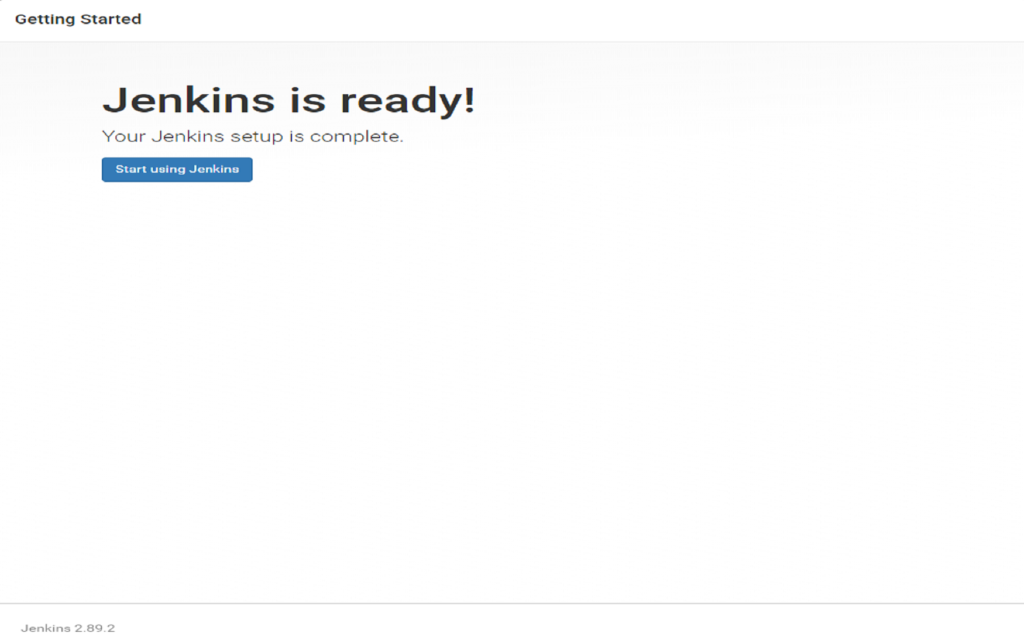
1. Vamos instalar los **plugins de Jenkins más populares**. Para ello, pulsamos sobre la caja de la izquierda y esperamos que los plugins se vayan instalando.

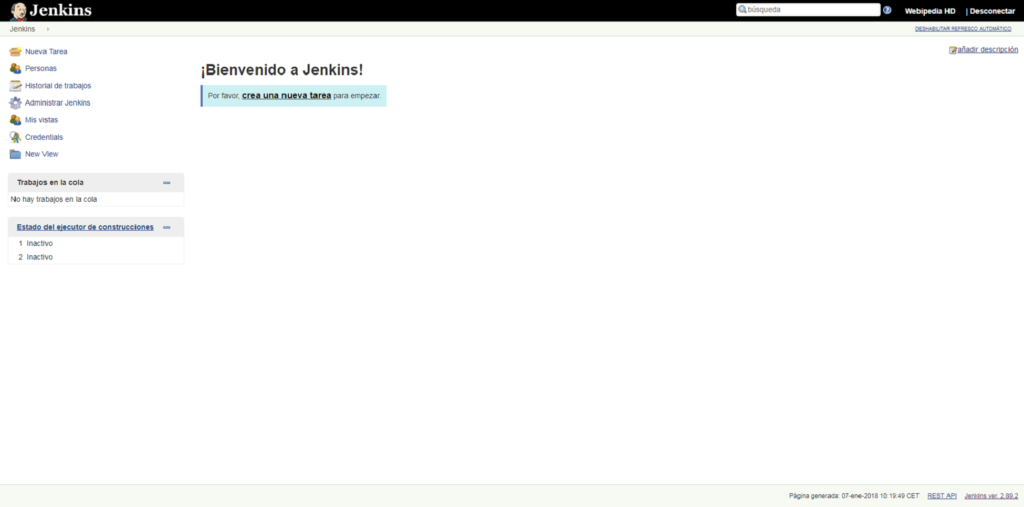
1. Una vez instalados, nos aparece la **ventana de creación del usuario administrador**. Rellenamos los datos y pulsamos sobre **“Save and Finish”**.



1. Y ya estaría instalado **Jenkins**, así que ya solo quedaría ponernos manos a la obra y empezar a utilizarlo.

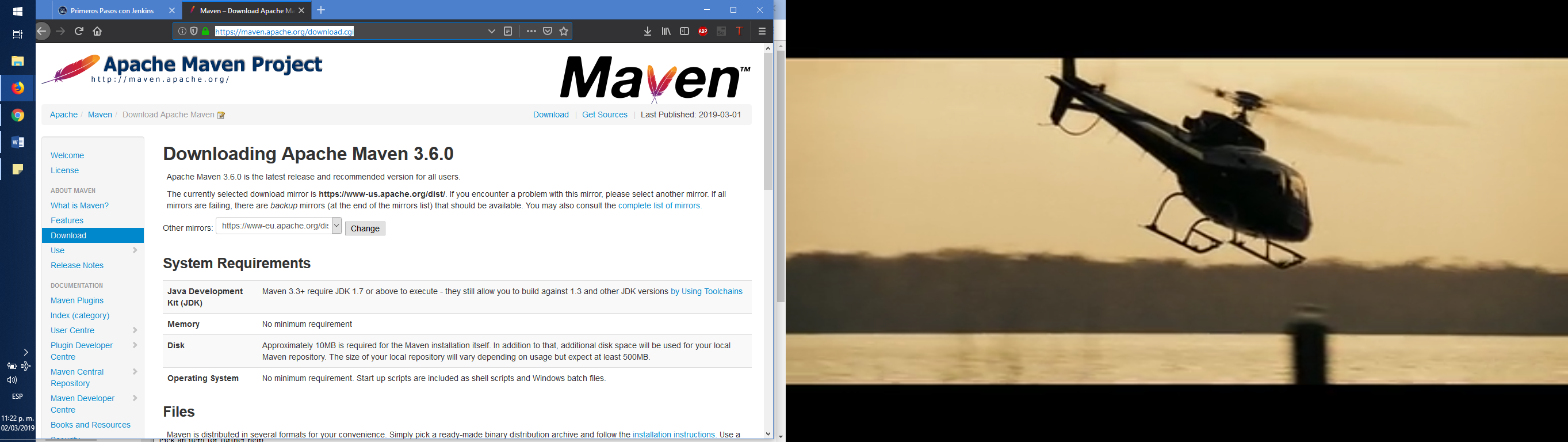


1. Esta sería la **pantalla de inicio** de Jenkins, a la que se accedería con la URL **“http://localhost:8080”**.

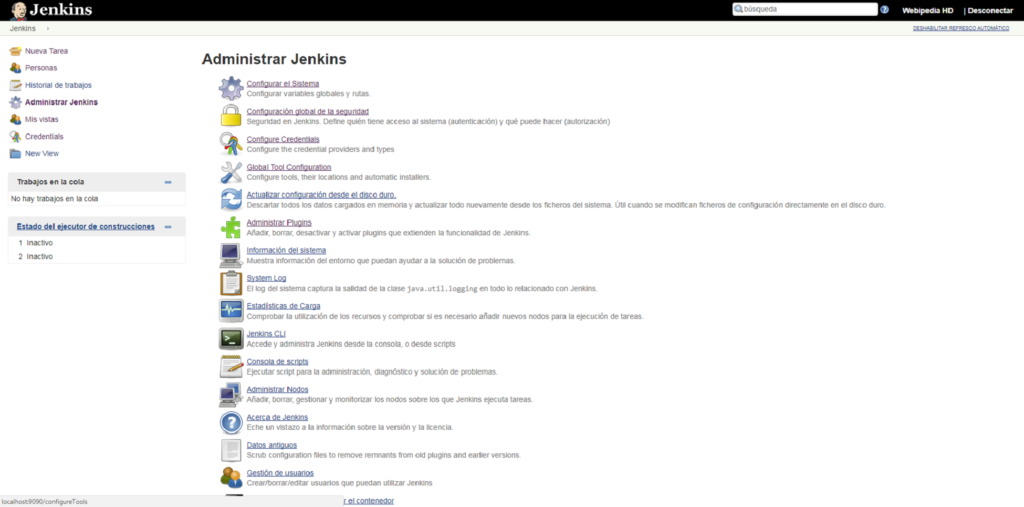


**Configuración de Jenkins**

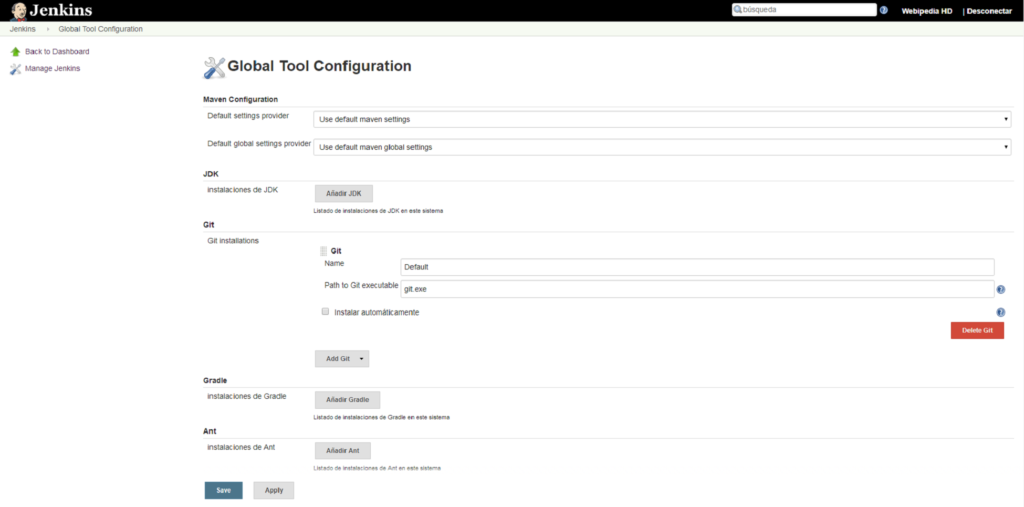
1. Tenemos que descargar apache Maven en <https://maven.apache.org/download.cgi>



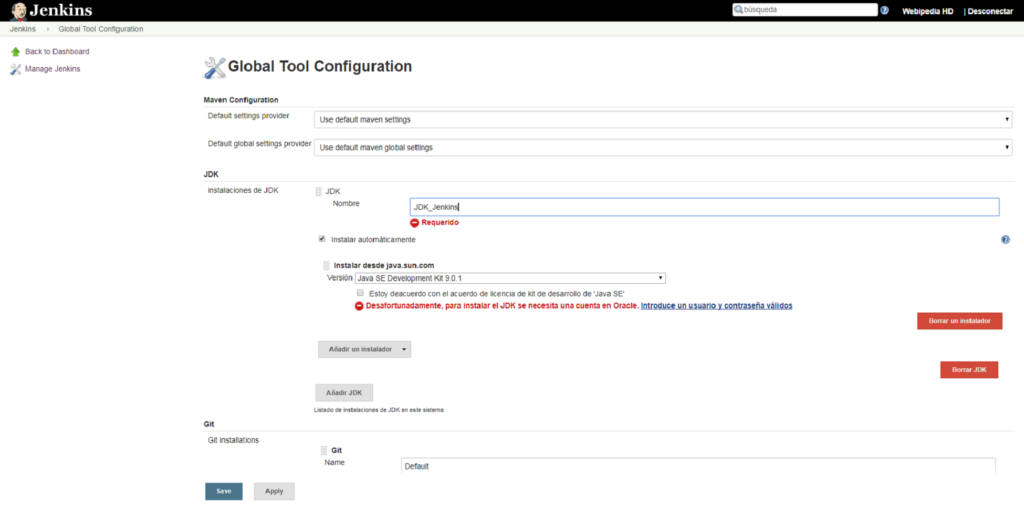
1. Lo descomprimimos en el disco local c y guardamos la dirección en la que se encuentra
2. Tenemos que tener descargado e instalado el jdk se puede obtener de <https://www.oracle.com>
3. Tenemos que tener descargado e instalado el Git se puede obtener de <https://git-scm.com/downloads>
4. Dentro de Jenkins pulsamos sobre **“Administrar Jenkins**” y después en **“Global Tool Configuration”**.



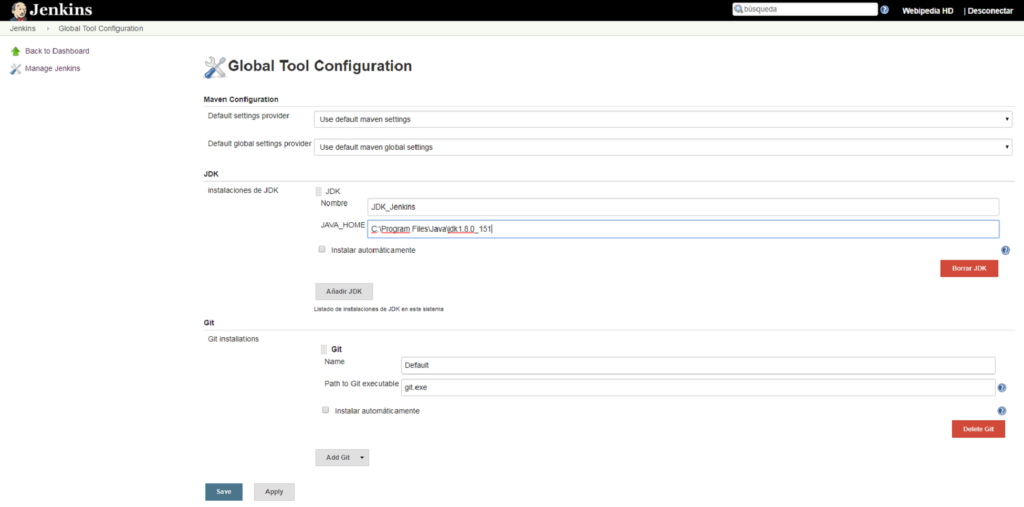
1. Desde esta pantalla configuraremos lo que podemos llamar el “núcleo” de nuestro Jenkins: **JDK** (para la creación de programas en Java), **Git** (para control de versiones) y **Maven** (para la gestión y construcción de proyectos). Para cada uno de estos aspectos, existe la correspondiente sección en esta pantalla, además de otras para **Gradle**, **Ant** y **Docker**.



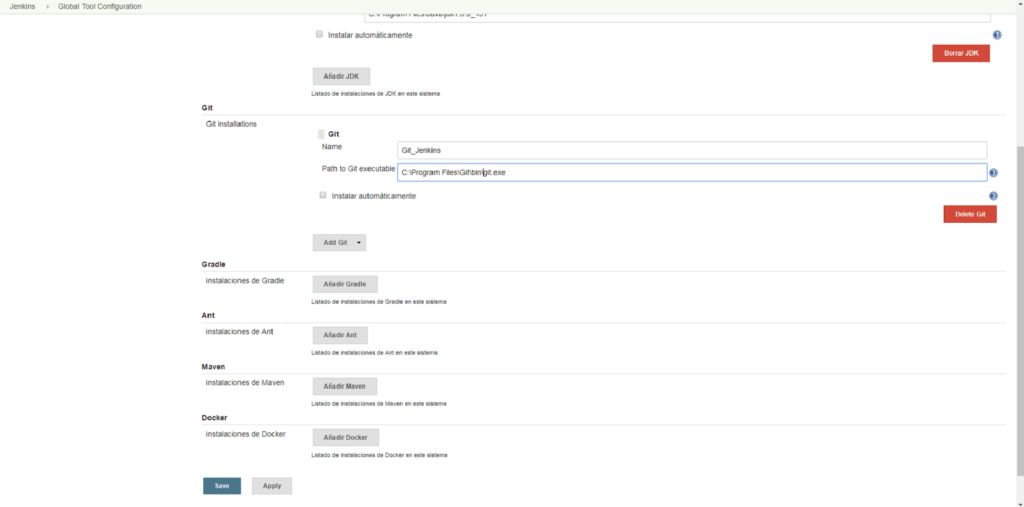
1. En la sección **“JDK”**, pulsamos sobre **“Añadir JDK”**. Tecleamos en primer lugar un nombre para nuestro JDK, por ejemplo **“JDK\_Jenkins”**.



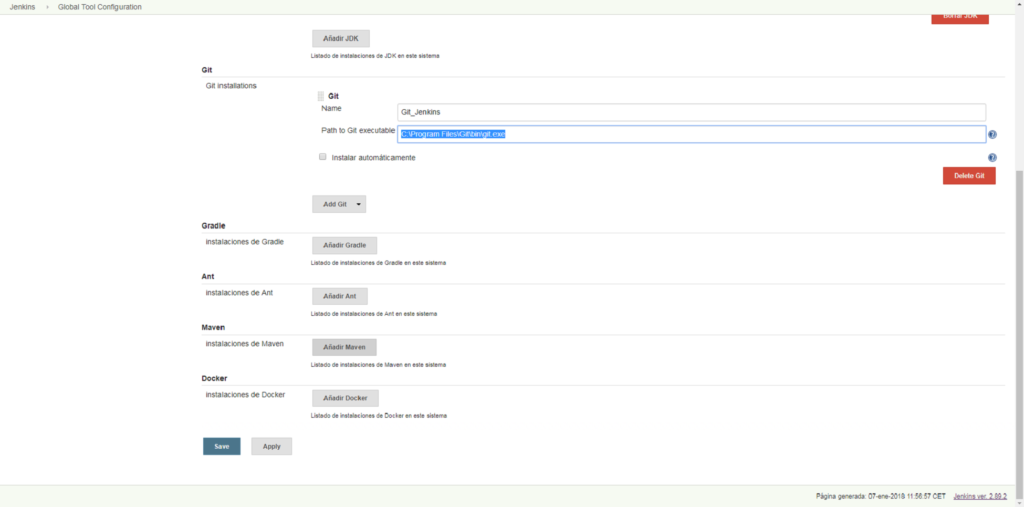
1. Vemos que aparece marcado el check de **“Instalar automáticamente”**, desmarcar esta opción y configurar la versión de Java la que tengo instalada. Al desmarcar la opción de descarga, se habilita una cajita para introducir el **JAVA\_HOME**. Se teclea la ruta correspondiente al jdk.



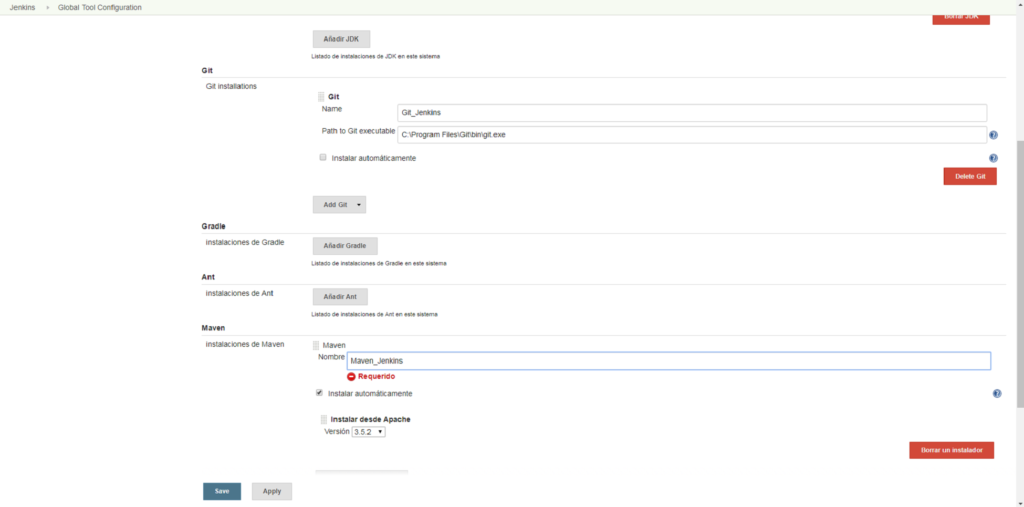
1. Pasamos a la siguiente sección, la de **Git**. Nos ofrece un sistema muy práctico de gestión de repositorios para nuestros proyectos (ver **“https://github.com/”**). Aquí simplemente tecleamos un nombre (por ejemplo **“Git\_Jenkins”**) e indicamos la ruta del ejecutable **“Git.exe”** en nuestra máquina.



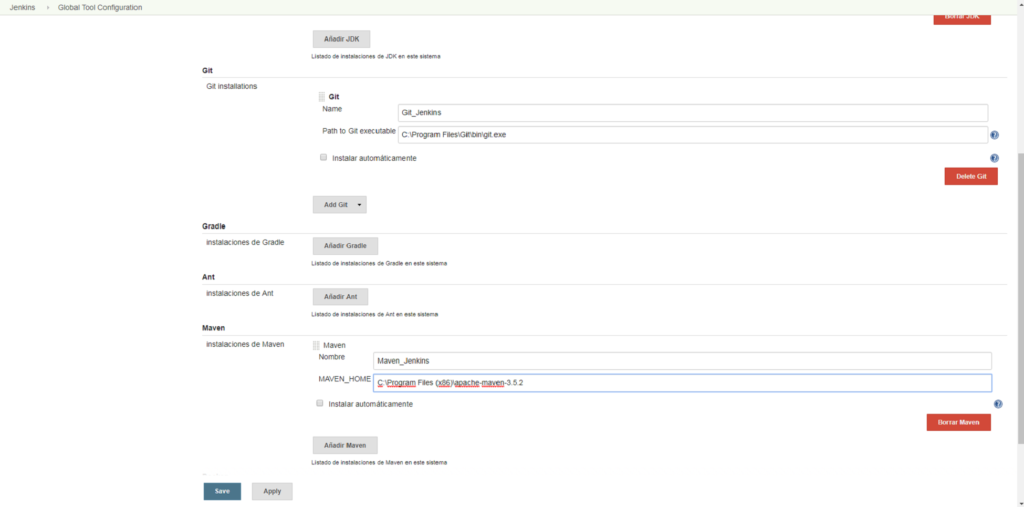
1. Por último, configuraremos nuestro **Maven**. En la sección correspondiente, pulsamos sobre **“Añadir Maven”**.



1. Tecleamos en primer lugar un nombre, como por ejemplo **“Maven\_Jenkins”**. Como vemos, al igual que con el JDK, aparece marcado un check **“Instalar automáticamente”**, que nos da la opción de instalar automáticamente desde Apache la **versión 3.5.2**.



1. Como ya la tenemos instalada, lo que se hará será **desmarcar esta casilla** e introducir la ruta en mi sistema del home de Maven.



1. Y ya estaría todo. Pulsamos **“Save”** en la parte inferior de nuestra pantalla de configuración, lo que nos llevaría de nuevo a la **pantalla de administración de Jenkins**.