5. CI/CD

Sistemas de automatización de la construcción

Gradle, npm, composer, nuget

Víctor Ponz

Institut Educació Secundària
El Caminàs

Curso de especialización en Ciberseguridad

Contenidos

Qué aprenderemos

- Qué son los sistemas de automatización de la construcción
- Herramientas principales: gradle, nmp, composer, etc.

Sistemas de automatización de contrucción

Pasar del código fuente a una versión ejecutable es una tarea que requiere tiempo y capacidad de cómputo, además de realizar otras tareas como movimiento de ficheros y carpetas, generación automática de documentación o análisis de cumplimiento de normas de codificación. Esta tarea se encuentra dentro de la integración continua (CI) que se basa en automatizar la compilación y el empaquetamiento.

Junto con Unix y el lenguaje C nace la primera herramienta de automatización de construcción, el famoso **Make**, que por defecto utiliza el fichero `makefile` que se encuentre en la ruta en que se ejecute el comando, en este fichero se indica el proceso para la generación del código objeto y el ejecutable.

Si bien **make** es una herramienta muy potente no se pensó para la gestión de dependencias, llegando a ser compleja la inclusión de los elementos externos necesarios para la construcción de la aplicación al realizarse de forma manual, la inclusión de versiones concretas de librerías y paquetes, la generación de documentación o realización de pruebas.

Con la popularización de Java aparece **Apache Ant** similar a **make** pero multiplataforma y usando formato `XML` para describir el proceso de compilación, la siguiente evolución fue **Maven** en la que también se define el proceso de gestión, compilación y construcción de un proyecto Java en formato `XML`, añadiendo nuevos conceptos como un repositorio desde el cual se pueden descargar *plugins* y librerías necesarias, incluso indicar la versión que se desea, con lo que teniendo el código fuentes y el fichero de configuración el proceso de construcción se simplifica de forma exponencial, además establece un ciclo de vida para los proyectos **Maven**.

Actualmente prácticamente cada lenguaje de programación tiene asociado un sistema de automatización de construcción, siendo su funcionalidad muy similar entre estos, en concreto:

- Definición de proyecto en fichero con formato estructurado: XML y/o JSON.
- Gestión de repositorios externos para librerías de terceros.
- Gestión con línea de comandos.
- Posibilidad de integrar otras herramientas o fases como ejecución de pruebas o desplegado de aplicaciones.

Algunas de las **herramientas de automatización** de construcción:

- **Gradle** Gradle Build Tool. Evolución de Ant y Maven, define su propio lenguaje para definir cómo construir el proyecto. Diseñado para grandes proyectos, en especial el desarrollo y despliegue. Usado por AndroidStudio, inicialmente centrado en Java y lenguajes relacionados como Google o Scala aunque se ha ampliado a otros como C++ o Swift.
- **npm** NPM (Naicely Pointed Mandibles). Sistema de gestión de paquetes para Node.js, aunque también se utiliza en el desarrollo de aplicaciones web en cliente, se basa en el análisis y ejecución del fichero package.json en el que se define información del proyecto.Algunas de las **secciones** del fichero **package.json**:
  - **license:** Tipo de licencia del proyecto.
  - **files:** Similar a ***.gitinore***, pero en este caso se indican los ficheros o directorios a incluir.
  - **bin:** Ficheros ejecutables necesarios para el proyecto que se incluyen en el **PATH**.
  - **scripts:** Permite definir pequeños guiones para la automatización de ciertas tareas, como limpiar datos temporales o general la aplicación para producción en el caso de **Angular.**
  - **Dependencies:** Esta sección define las dependencias de librerías externas necesarias para construir el proyecto, por ejemplo, al clonar un proyecto en **GitHub** es necesario instalar las dependencias indicadas en esta sección del fichero, en concreto el comando ***npm install***.

  Además provee de una interfaz de línea de **comandos CLI** para la gestión del proyecto, en concreto el manejo del fichero `package.json` con **comandos** como:
  - **npm install.** Instala las dependencias.
  - **npm upate.** Actualiza las librerías.
  - **npm exec.** Ejecuta el guión identificado por el nombre que se le facilita como parámetro.
  - **npm init.** Crea fichero y esqueleto del fichero ***package.json***.
- **Comporer** Composer es el equivalente a **Npm** para **PHP**. Se centra también en la gestión de paquetes. La configuración se encuentra en **composer.json** y es muy similar a **Npm**, posee sección de **dependencias**, **línea de comandos y definición** y **ejecución de *scripts***.
- **Nuget** Nuget es usado por la plataforma **.NET** para crear, compartir y construir aplicaciones, se centra en la gestión de dependencias y paquetes, en especial de las diferentes versiones de los mismos. Entre sus características destaca la de poder crear y administrar repositorios privados. Usa **ficheros XML** con formato muy similar a **NPM**. Al igual que los anteriores posee una **interfaz de comandos**.