INVETIGACIÓN GITHUB, MAVEN Y GRADLE

Trabajo 1

Andrés Elías Carrascal Verona, Luis Carlos Rendon Cardona, Wulfram Polo Casteñeda

Universidad de Antioquia

Introducción

Mediante el siguiente trabajo, vamos a conocer acerca de GitHub, Maven y Gradle, conoceremos como están compuestos cada uno y cuáles son sus principales funciones y usos que día a día utilizan los desarrolladores para llevar a cabo una buena creación de software. Hablaremos de las herramientas que se utilizan para estos 3 componentes y el uso adeacuado de estas, generando así una información en especie de guía para profundizar y afianzar los conocimientos previos.

Objetivos generales

- Profundizar sobre los conceptos, beneficios y funcionalidades de distintas tecnologías como lo son Gradle, Maven y Github.
- Fomentar el aprendizaje mediante la indagación de tecnologías aplicables a la vida cotidiana del programador.
- Analizar la importancia de cada una de estas tecnologías para el desarrollo de software.
- Aprender de qué manera se pueden utilizar o aplicar estas tecnologías a los entornos del desarrollo de software.

Objetivos específico

- Poder implementar la herramienta GitHub para el desarrollo de aplicaciones.
- Ahondar en la investigación de los pros y contras de Gradle.
- Investigar sobre las herramientas o lugins que se pueden utilizar mediante Gradle.
- Indagar sobre las diferencias existentes entre los precursores de Gradle, que son Maven y Ant.
- Confirmar si herramientas como Maven son multi-plataforma.

GitHub

¿Qué es GitHub y para qué se utiliza?

GitHub es un sistema de gestión de proyectos y control de versiones de código, así como una plataforma de red social diseñada para desarrolladores. ¿Pero para qué se usa GitHub? Bueno, en general, permite trabajar en colaboración con otras personas de todo el mundo, planificar proyectos y realizar un seguimiento del trabajo. Para hablar de GitHub primero debemos conocer acerca de Git y Hub

¿Qué es Git?

Git es un sistema de control de versiones desarrollado por Linus Torvalds. cuando se hace un nuevo proyecto, siempre continúan haciéndole modificaciones al código. Incluso después de la puesta en marcha de los proyectos, todavía necesitan actualizar las versiones, corregir errores, agregar nuevas funciones, etc.

El sistema de control de versiones ayuda a registrar los cambios realizados al código. Aún más, registra quién realizó los cambios y puede restaurar el código borrado o modificado.

¿Qué es Hub?

El hub de GitHub es lo que convierte una línea de comandos. Además de contribuir a un determinado proyecto, GitHub permite a los usuarios socializar con personas de ideas afines. Puedes seguir a las personas y ver qué hacen o con quién se conectan.

¿Cómo está compuesto GitHub?

Repository

Un repositorio o “repo” es un directorio donde se almacenan los archivos del proyecto. Puede estar ubicado en el almacenamiento de GitHub o en un repositorio local en las computadoras. Puede almacenar archivos de código, imágenes, audios o todo lo relacionado con el proyecto en el repositorio.

Branch

Branch, que se traduce como rama, es una copia del repositorio. Se puede utilizar la rama cuando se requiere hacer desarrollo de forma aislada. Trabajar en una rama no afectará el repositorio central u otras ramas. Si se ha completado el trabajo, se puede combinar esa rama con otras ramas y con el repositorio central mediante una pull request.

Pull Request

Pull request significa que se le informa a los demás que se ha enviado al repositorio principal el cambio que se hizo en una rama. Los colaboradores del repositorio pueden aceptar o rechazar una pull request.

Bifurcar un repositorio

Bifurcar un repositorio significa crear un nuevo proyecto basado en el repositorio existente. En términos simples, esto significa que se copia un repositorio existente, se hacen los cambios necesarios, se almacena la nueva versión como un nuevo repositorio y se nombra como un proyecto propio. Esta es una muy buena función que propulsa el desarrollo del proyecto. Debido a que es un proyecto totalmente nuevo, el repositorio central no se verá afectado. Si el repositorio principal es actualizado, también se aplica esa actualización a la bifurcación actual.

Maven

¿Qué es Maven?

Anteriormente las formas de compilación y generación de ejecutables era un proceso no tan fácil de realizar, para llevar a cabo estos proceso, era necesario generar ejecutables de los proyectos, la gestión de las librerías era algo complejo de manejar ya que se debía saber dónde incluirlas y también qué dependencias de compilación había en el proyecto; fue en el año 2001 llega una herramienta que ayudaría a gestionar todas estos procesos de una manera mucho más fácil, ésta herramienta es Maven, capaz de gestionar los proceso de compilación y ejecución a partir del código fuente de una manera rápida. Algo que también es importante, es que esta herramienta es Open-source.

Lo que en un pasado tomaba más de 1 hora para generar un build de un proyecto, pasó a realizarse en unos pocos minutos independientemente de cuantos módulos, dependencias o librerías tenga el proyecto, es por esto que esta herramienta ha tomado tanta fuerza en los últimos años.

¿Qué más puede hacer Maven aparte de generar builds?

Lo cierto es que Maven ayuda demasiado en el proceso de realización de builds, pero no solo funciona para eso, esta herramienta también es capaz de realizar la gestión de proyectos de software, tanto en la comprobación de si el código es correcto hasta la realización de despliegue en aplicaciones, también permite realizar informes, ejecutar pruebas entre otros.

¿Cuáles son las etapas de un proyecto Maven?

*Validación (validate): Valida que en el proyecto se encuentre todo correcto

*Compilación (compile): Permite realizar el proceso de conversión a lenguaje de máquina.

*Test (test): Probar del código fuente utilizando algún framework que permita la realización de pruebas unitarias.

*Empaquetar (package): Permite realizar la transformación de código compilado y transformado a los formatos de .jar o .war

*Pruebas de integración (integration-test): Procesa y despliega el código en algún entorno que permita la ejecución de pruebas de integración.

*Verificar (verify): Permite validar que el código empaquetado sea válido y cumpla con los requisitos de calidad

*Instalar (install): Permite traer y utilizar el código local de Maven para usarlo en los proyectos que se deseen.

*Desplegar (deploy): Permite desplegar el código en un entorno

Para la utilización de algunos de los códigos de Maven nombrados anteriormente, solo se utiliza la palabra mvn y luego alguna de las palabras en color rojo del menú anterior, por ejemplo, **mvn install**.

¿Cómo se realiza la gestión de las librerías?

Para realizar la gestión de las librerías, es necesario ir al proyecto y buscar un archivo llamada POM, ahí es donde las librerías se descargan desde el repositorio Maven central que es donde se encuentran la mayoría de librerías utilizadas en todo el mundo, a continuación. El artefacto está compuesto por los siguientes datos; Nombre, grupo, versión y dependencia, a continuación, hay un ejemplo de cómo es el archivo.

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-
4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>
<groupId>com.genbetadev.proyecto1</groupId>
<artifactId>proyecto1</artifactId>
<version>0.0.1-SNAPSHOT</version>
<packaging>jar</packaging>
<dependencies>
<dependency>
<groupId>log4j</groupId>
<artifactId>log4j</artifactId>
<version>1.2.17</version>
</dependency>
</dependencies>
</project>
```

Gradle

¿Qué es Gradle?

Antes de ahondar en las definiciones concretas de la utilización de Gradle y sus funcionalidades vamos a hacer un pequeño énfasis en por qué y el cómo surge esta magnífica herramienta de trabajo, es bueno saber que gradle tiene dos precursores claves y que ambos se juntan para formar una mezcla perfecta que sintetiza lo mejor de cada uno de ellos, estos precursores son Ant y Maven y el su producto final posee tanto la fuerza y flexibilidad del primero como la facilidad de uso que posee el segundo. Teniendo en cuenta la evolución tecnológica y adicional a esto la evolución misma de la programación existe una materia que necesita siempre poseer distintos avances de desarrollo en el mundo de la automatización de builds.

Gradle posee un excelente manejo y flexibilidad, además de que nos permite ejecutar distintos tipos de lenguajes y no nos limita a trabajar con uno solo, posee también un sistema de gestión de dependencias bastante estable. También es muy rápido, pues ejecuta todas las tareas de manera eficaz haciendo uso de la reutilización de las salidas de las ejecuciones anteriores, y procesando solamente las entradas que presentan cambios en paralelo

En resumen, podemos decir entonces que Gradle posee en particular 6 características que lo identifican de manera acertada y que lo describen muy bien, estas características son:

- Flexibilidad
- Construcción incremental
- Diseño de repositorio personalizado
- Dependencias transitivas

- Compilación incremental para java

También podemos tener en cuenta que Gradle implementa de manera eficiente la depuración colaborativa, pues permite compartir los resultados de la compilación para resolver en equipo de forma eficiente todos los posibles problemas que puedan llegar a aparecer en algún momento. Para hacer uso de esta magnífica herramienta es necesario haber instalado previamente el JDK 8 o una versión superior a esta, Gradle se organiza en diferentes tipos de proyectos, donde estos pueden llegar a ser un producto de lo que hagas, como una librería jar por ejemplo o una aplicación web, cada proyecto estará compuesto por una o más tareas, estas tareas representan de manera conjunta la parte más fundamental en una construcción y proporciona una biblioteca de distintos tipos de tarea, en donde cada una de estas realiza una operación, entre estas tareas encontramos la herramienta encargada e listar los proyectos y las tareas. Para esto se hace uso de una terminal.

Complementos de Gradle

Gradle ayuda a los equipos a desarrollar, automatizar software de calidad muy poco tiempo, hay que recalcar que gradle posee infinidad de complementos que lo hacen más eficiente y que mejoran de gran manera la manera en la que se trabaja.

- javamuc.gradle-semantic-build-versioning
- io.freefair.maven-publish-war
- io.freefair.maven-publish-java
- org.mozilla.rust-android-gradle.rust-android
- com.bmuschko.docker-remote-api

Conclusión

En conclusión, las tecnologías habladas anteriormente le han facilitado la vida a muchas personas gracias a la rapidez y agilidad de resolver procesos que anteriormente eran muy complejos de realizar o que no se podían realizar bien, al inicio puede ser complejo la forma de la implementación que tienen, pero bata con leer un poco la información y estructura que tienen para darse cuenta que son fáciles de manejar, siempre y cuando se tenga la información de cada una a la mano.