LEJOS

Reporte de ejecución de IDE de sociedad de agentes

Resumen

En este trabajo se explican detalladamente los pasos realizados para lograr la correcta ejecución del IDE de sociedad de agentes InAge2000. Este es un objetivo fundamental para la correcta realización de nuestro trabajo asignado en la materia de Programación Orientada a Objetos 2, el cual es mejorar la implementación del código de LeJOS para la programación de una sociedad de agentes en un robot de Lego Mindstorm.

Este IDE se encuentra diseñado en lenguaje Java, el cual es un lenguaje puramente orientado a objetos. La interfaz gráfica de usuario fue realizada en JavaFX, esto es una tecnología de Java especialmente creada para el diseño de GUI. Por último, cabe destacar que este proyecto tiene una complejidad mayor a una simple aplicación de Java, por lo que fue gestionado con la herramienta Maven.

Palabras clave: Lego, LeJOS, sociedad de agentes, JavaFX, Maven.

1. Introducción

Como ha sido mencionado, el IDE InAge está gestionado con Maven. Maven es una herramienta de software para la gestión y construcción de proyectos Java que pertenece a la empresa Apache.

Normalmente cuando nosotros trabajamos con Java/JavaEE el uso de librerías es algo común como en cualquier otro lenguaje de programación. Por otro lado, una librería puede depender de otras librerías para funcionar de forma correcta. Así pues necesitamos más información para gestionarlo todo de forma correcta. Maven solventa este problema a través del concepto de Artefacto.

Un Artefacto puede verse como una librería mejorada (aunque agrupa más conceptos). Contiene las clases propias de la librería pero además incluye toda la información necesaria para su correcta gestión (grupo, versión, dependencias etc). Para definir un Artefacto necesitamos crear un fichero POM.xml (Proyect Object Model) que es el encargado de almacenar toda la información que hemos comentado anteriormente

Una vez definidos correctamente todos los Artefactos que necesitamos, Maven nos provee de un Repositorio donde alojar, mantener y distribuir estos. Permitiéndonos una gestión correcta de nuestra librerías, proyectos y dependencias.

2. Desarrollo

Para ejecutar el IDE InAge, utilizamos la herramienta Eclipse, ya que está programado en Java. Preferimos utilizar Eclipse en lugar de Netbeans ya que Eclipse permite importar un proyecto de Maven de forma más amigable.

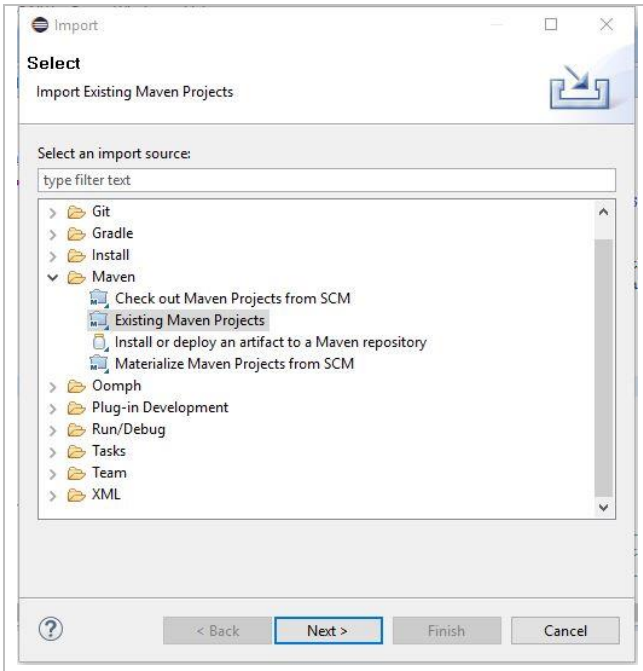

Fig. 1. Importación del proyecto de Maven en Eclipse

Después de realizar este paso, ya tenemos el proyecto cargado en Eclipse, podemos ver en el 'Project Explorer' que todo se encuentra sin anomalías, excepto por el archivo "pom" mencionado anteriormente.

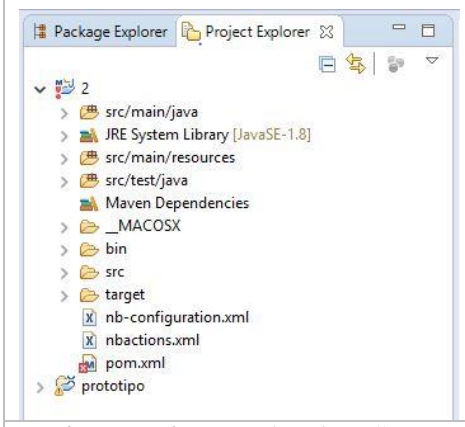

Fig 2. Desperfecto en el archivo "pom.xml"

Sin embargo, y como veremos después, la ejecución del IDE se puede realizar aun con este fallo. Es importante mencionar que se trató de corregir este error e incluso se sigue trabajando para corregirlo.

Después de esto, se localiza y se abre el archivo Java donde se encuentra la aplicación principal o Main, la cual está localizada en la carpeta **src/main/java**, en el 'package' **com.compimac.fx**, dicho archivo tiene por nombre **MainApp.java.**

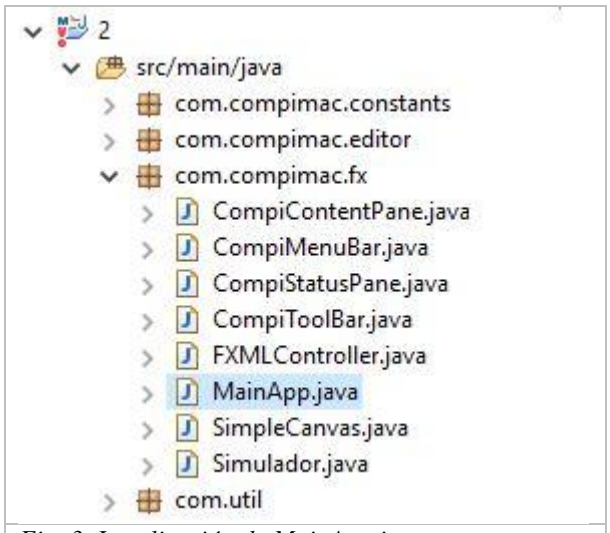

Fig. 3. Localización de MainApp.java

Después de abrir este archivo se lleva a cabo la compilación y ejecución en Eclipse, sin embargo, se ven los efectos de los errores mostrados anteriormente, es importante resaltar que no son errores de compilación, por lo tanto, damos clic en "Proceed" y la ejecución continúa.

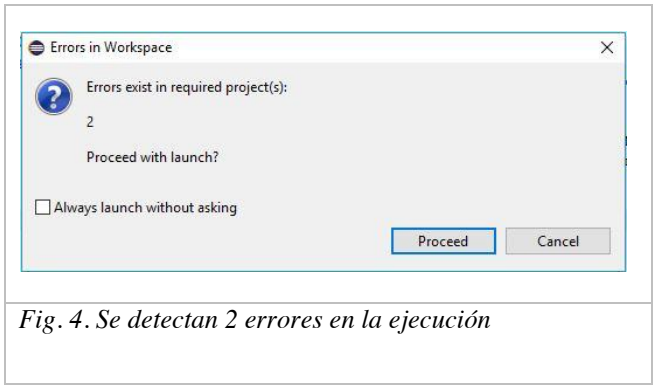

Fig. 4. Se detectan 2 errores en la ejecución

Por último, vemos por fin la interfaz del IDE InAge:

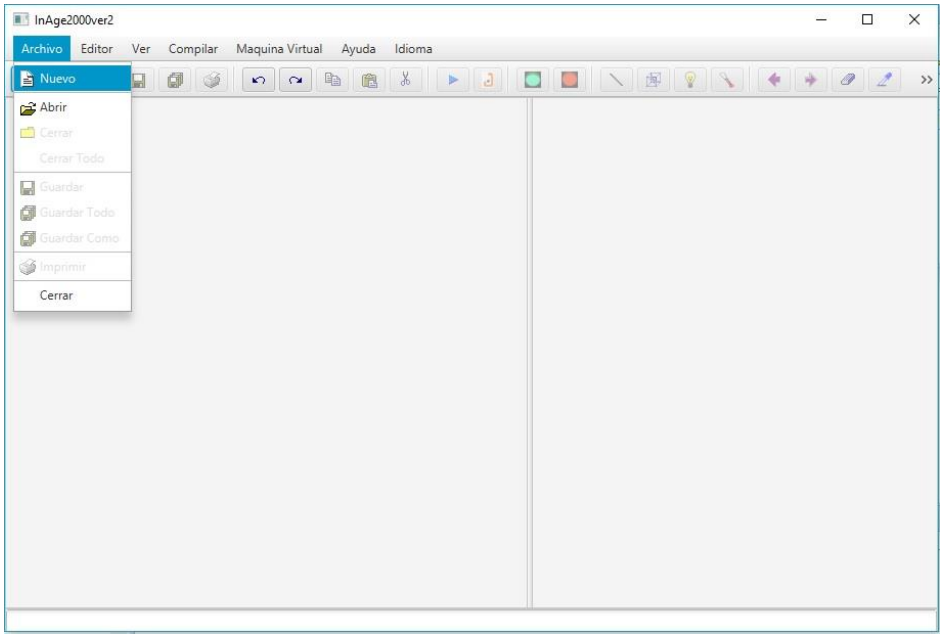

Fig. 5. Interfaz gráfica de usuario del IDE InAge.

3. Resultados

Después de lograr compilarlo, tratamos de ejecutar un programa ya hecho en el lenguaje InAge2000 y vemos que efectivamente se compiló correctamente.

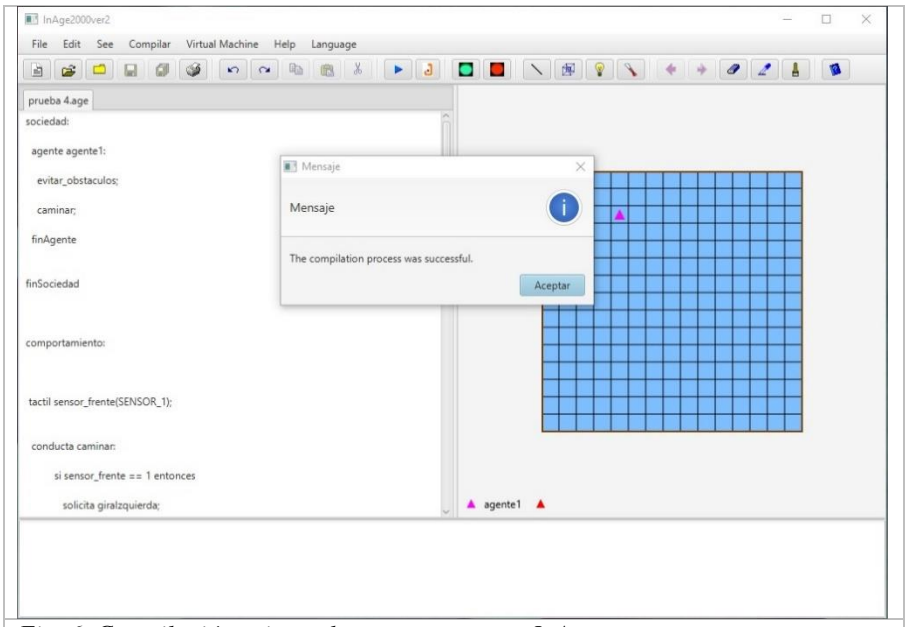

Fig. 6. Compilación exitosa de un programa en InAge

4. Conclusiones.

Pudimos observar el IDE se ejecuta correctamente, aunque hay aspectos referentes a la interfaz gráfica de usuario que se deben mejorar. Esto se abordará con los prototipos de interfaces por caso de uso.

Por último, vemos también que hay errores en la traducción de código InAge2000 a lenguaje LeJOS, el lenguaje nativo de los Lego Mindstorm. Esto es precisamente el trabajo que al equipo le fue asignado, corregir estos errores.