

Instituto Politécnico Nacional

Escuela Superior de Cómputo

Compiladores

Tecla Parra Roberto

Manual técnico - Logo

Integrantes

Rafael Landa Aguirre

José Leonardo Juárez Bouchan

Jesús Paniagua Hernández

Grupo: 3CM8

Ciudad de México, lunes 11 de junio de 2018

Índice

1. Introducción	2
2. Requerimientos técnicos	3
2.1. Java	3
2.1.1. Java Development Kit (JDK)	3
2.1.2. Java Runtime Edition (JRE)	3
2.2. Herramientas de desarrollo	4
2.2.1. Maven	4
3. Instalaciones de plataforma	5
3.1. Instalación de Java	5
3.2. Eclipse	12
4. Ejecución del aplicativo	15
4.1. Ejecución en entornos de desarrollo	15
4.1.1. Eclipse	15
4.1.2. Ejecución via terminal	17
5. Casos de uso	18
6. Modelo de información	19
7. Prototipos de pantallas del aplicativo	20

1. Introducción

Logo es un lenguaje de programación de alto nivel es, en parte funcional, en parte estructurado; de muy fácil aprendizaje, razón por la cual suele ser el lenguaje de programación preferido para trabajar con niños y jóvenes. Fue diseñado con fines didácticos por Danny Bobrow, Wally Feurzeig, Seymour Papert y Cynthia Solomon, los cuales se basaron en las características del lenguaje Lisp. Logo fue creado con la finalidad de usarlo para enseñar programación y puede usarse para enseñar la mayoría de los principales conceptos de la programación, ya que proporciona soporte para manejo de listas, archivos y entrada/salida. Logo cuenta con varias versiones.

Una característica más explotada de Logo es poder producir «gráficos tortuga», es decir, poder en dar instrucciones a una tortuga virtual, un cursor gráfico usado para crear dibujos, que en algunas versiones es un triángulo, en otras tiene la figura de una tortuga vista desde arriba. Esta tortuga o cursor se maneja mediante palabras que representan instrucciones.

El desarrollo de este proyecto consistirá en una interfaz gráfica para el usuario con un apartado en donde el pueda agregar código con la finalidad de dibujar desde las figuras más sencillas como cuadrados y círculos, hasta espirales, árboles , estrellas, etc, con la posibilidad de agregar colores a las figuras dibujadas, esto se mostrará en una vista en el mismo programa.

Para el desarrollo de la interfaz gráfica se cuenta con la API de JAVA: AWT, para aplicaciones de escritorio en Windows/Linux.

2. Requerimientos técnicos

2.1. Java

plataforma informática comercializada por primera vez en 1995 por Sun Microsystems. Hay muchas aplicaciones y sitios web que no funcionarán a menos que tenga Java instalado y cada día se crean más. Java es rápido, seguro y fiable. Desde portátiles hasta centros de datos, desde consolas para juegos hasta súper computadoras, desde teléfonos móviles hasta Internet, Java está en todas partes.

Java es la base para prácticamente todos los tipos de aplicaciones de red, además del estándar global para desarrollar y distribuir aplicaciones móviles y embebidas, juegos, contenido basado en web y software de empresa. Con más de 9 millones de desarrolladores en todo el mundo, Java le permite desarrollar, implementar y utilizar de forma eficaz interesantes aplicaciones y servicios. Desde portátiles hasta centros de datos, desde consolas para juegos hasta súper computadoras, desde teléfonos móviles hasta Internet, Java está en todas partes.

2.1.1. Java Development Kit (JDK)

El JDK es un entorno de desarrollo para crear aplicaciones, applets y componentes utilizando el lenguaje de programación Java. El JDK incluye herramientas útiles para desarrollar y probar programas escritos en el lenguaje de programación Java y que se ejecutan en la plataforma Java.

2.1.2. Java Runtime Edition (JRE)

Java Runtime Environment (JRE) es lo que se obtiene al descargar el software de Java. JRE está formado por Java Virtual Machine (JVM), clases del núcleo de la plataforma Java y bibliotecas de la plataforma Java de soporte. JRE es la parte de tiempo de ejecución del software de Java, que es todo lo que necesita para ejecutarlo en el explorador web.

2.2. Herramientas de desarrollo

2.2.1. Maven

Maven es una herramienta principalmente utilizada en el desarrollo de software Java. Aparece ante la necesidad de modelar el concepto de proyecto y artefacto en forma estándar independientemente del IDE de desarrollo.

Maven es una herramienta de software para la gestión y construcción de proyectos Java creada por Jason van Zyl, de Sonatype, en 2002. Es similar en funcionalidad a Apache Ant (y en menor medida a PEAR de PHP y CPAN de Perl), pero tiene un modelo de configuración de construcción más simple, basado en un formato XML. Estuvo integrado inicialmente dentro del proyecto Jakarta pero ahora ya es un proyecto de nivel superior de la Apache Software Foundation.

Maven utiliza un Project Object Model (POM) para describir el proyecto de software a construir, sus dependencias de otros módulos y componentes externos, y el orden de construcción de los elementos. Viene con objetivos predefinidos para realizar ciertas tareas claramente definidas, como la compilación del código y su empaquetado.

Se utiliza como herramienta de SCM (Software Configuration Management) para controlar:

- El versionado de nuestros proyectos.
- Las dependencias con proyectos nuestros y librerías de terceros.
- Automatización de tareas relativas al ambiente de desarrollo y construcción del artefacto (compilación, generación de código, empaquetado, etc.).

3. Instalaciones de plataforma

3.1. Instalación de Java

Descargado, recuerde que no importa la versión del JDK que haya descargado, puede ser una versión anterior o posterior (Recomendamos la *versión 8*).

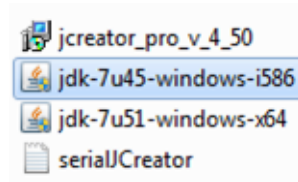


Figura 1: Archivo ejecutable o instalador.

Al ejecutar este archivo nos desplegará la siguiente ventana de bienvenidos al instalador de actualización del JDK.



Figura 2: Ventana inicial para comenzar la instalación.

A los pocos segundos nos mostrara la imagen de bienvenida al instalador de JDK, el procedimiento es verificar si existe en su equipo de cómputo una versión anterior a instalar, si existe dicha versión de JDK solo actualizara su JDK de lo contrario será una instalación desde cero, en la que deberemos dar clic en aceptar para que inicie el proceso de instalación.



Figura 3: Guía de instalación de Java.

Una vez leído el archivo nos desplegará la ventana inicial al instalar el JDK, donde debemos elegir las opciones que queremos instalar en nuestro pc, regularmente se instala todo pero usted puede elegir que instalar (JVM, Código, Ejemplos).

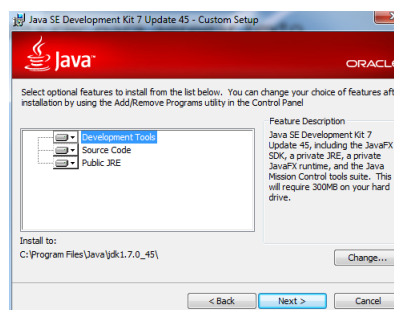


Figura 4: Selección de herramientas de instalación.

Al dar clic en siguiente nos muestra una pantalla del progreso de descarga según a instalar Productos de Java que es la primer ventana del menú, lo cual desplegará la siguiente ventana.

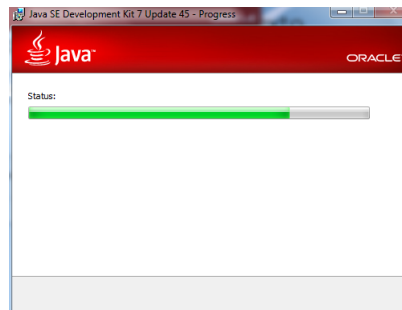


Figura 5: Estatus de descarga del aplicativo.

Posteriormente comenzará a instalarse la máquina virtual de java que habíamos descargado en los pasos anteriores, este procedimiento puede tardar dependiendo de los recursos de nuestro equipo de cómputo.

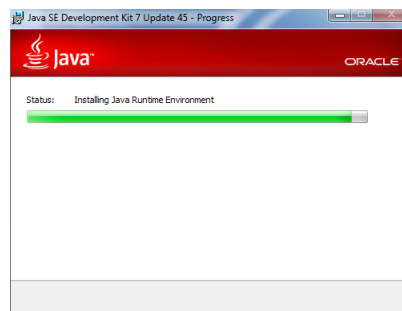


Figura 6: Proceso de instalación de Java runtime Edition.

En el siguiente paso nos preguntará si deseamos instalar el JRE de la máquina virtual, obviamente decimos que sí.

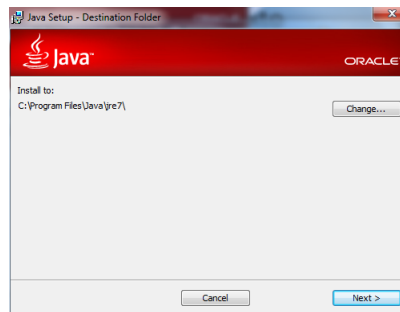


Figura 7: Selección de la carpeta de instalación.

Comenzará a instalar los componentes del JDK, esto puede tardar varios minutos así que no desesperen y vayan a prepararse una taza de café.



Figura 8: Proceso de instalación.

Hasta este momento tenemos instalado nuestro JDK, pero esto no es suficiente si queremos que nuestro equipo de cómputo pueda realizar sistemas informáticos, por esa razón tenemos que decirle a nuestro pc que pueda generar archivos .class de la siguiente manera. Primero damos clic derecho en equipo y elegimos propiedades, como en la siguiente imagen:

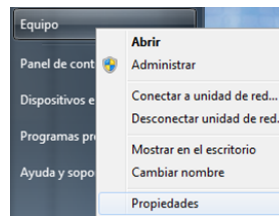


Figura 9: Menú de propiedades de Windows.

Nos mostrara un cuadro de dialogo en el cual del lado izquierdo contiene un menú donde nosotros seleccionaremos Configuración Avanzada del Sistema.



Figura 10: Selección de la configuración del sistema de Windows.

Nuevamente nos abrirá un cuadro de dialogo donde seleccionamos la pestaña Opciones Avanzadas y elegimos el botón variables de entorno.

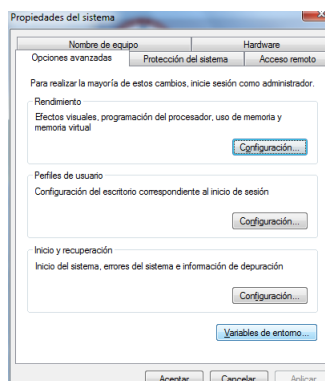


Figura 11: Ventana para seleccionar la configuración de variables de entorno.

En el siguiente cuadro de diálogo tenemos que tener mucho cuidado ya que cada vez que iniciamos nuestro pc estas variables de entorno son las que se encargan de cargar nuestro sistema operativo así es que si llegáramos a borrar una variable de esta sección podría no cargar la próxima vez nuestro sistema operativo. Primero debemos de ubicar la variable Path, la seleccionamos y damos clic en el botón Editar.

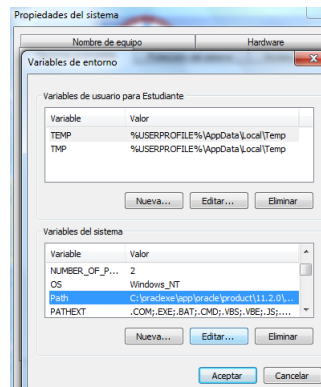


Figura 12: Configuración de la variable *Path* de Windows.

Debemos de copiar la ruta donde se instaló nuestro JDK para poder agregarla a las variables de entorno así que vamos a copiar la ruta.

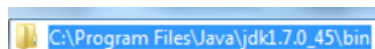


Figura 13: Carpeta de instalación de java/bin.

En las variables de entorno vamos a agregar un punto y coma ";" pegamos la ruta que copiamos en el paso anterior y agregamos nuevamente punto y coma ";" como nota importante no debemos de borrar ninguna variable de entorno por lo explicado anteriormente.

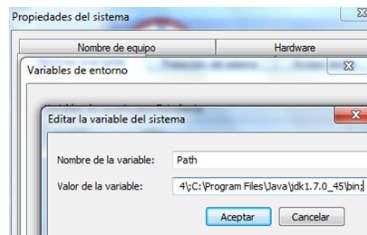


Figura 14: Actualización de la variable *Path* de Windows con la dirección de la carpeta de instalación de Java.

Aceptamos todas las ventanas que abrimos anteriormente.

Una vez realizado los pasos descritos anteriormente es hora de verificar que los hayamos realizado correctamente. Abrimos una ventana de consola para ello vamos a inicio « accesorios » símbolo de sistema y escribimos el comando `java` y damos enter. Nos debe de mostrar muchas líneas describiendo el comando `java` como se muestra a continuación. Si nos arroja un error repetir los pasos anteriores hasta lograr el objetivo.

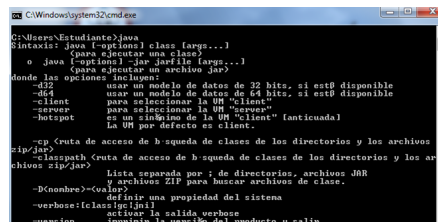
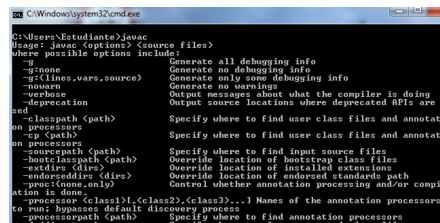


Figura 15: Verificación del comando *java* en la terminal.

Ahora escribiremos el comando `javac` y de la misma manera que el comando anterior nos debe de describir este comando.



```
C:\Windows\system32\cmd.exe
C:\Users\Estudiente>javac
Usage: javac <options> <source files>
where possible options include:
  -g             Generate all debugging info
  -g:none        Generate no debugging info
  -g{lines,vars,source} Generate only some debugging info
  -m:none        Generate no warnings
  -verbose       Output messages about what the compiler is doing
  -deprecation  Output source locations where deprecated APIs are used
  -classpath <path> Specify where to find user class files and annotations
  -cp <path>       Specify where to find user class files and annotations
  -processor <name> Specify where to find input source files
  -sourcepath <path> Override location of bootstrap class files
  -bootclasspath <path> Override location of installed extensions
  -extdirs <dirs>   Override location of endorsed standards path
  -endorseddirs <dirs> Override location of endorsed standards path
  -proc:none      Control whether annotation processing and/or compilation is done
  -processor <class1> [<class2>...<classN>] Names of the annotation processors to run; bypasses default discovery process
  -proc:all       Specify where to find annotation processors
```

Figura 16: Verificación del comando *javac* en la terminal.

3.2. Eclipse

Eclipse. Ve a: <http://www.eclipse.org/downloads/>

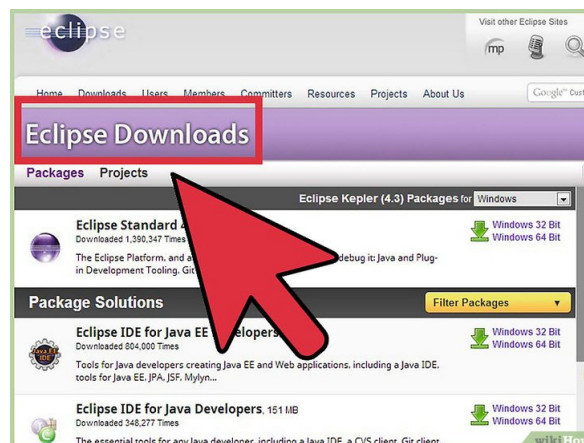


Figura 17: Descarga de eclipse desde la página oficial de eclipse.

Para los usuarios de Windows, tendrás que saber qué versión de sistema operativo tienes. Si tu computadora es de 64-bit, selecciona Windows 64 y si es de 32-bit, selecciona Windows 32 bit.

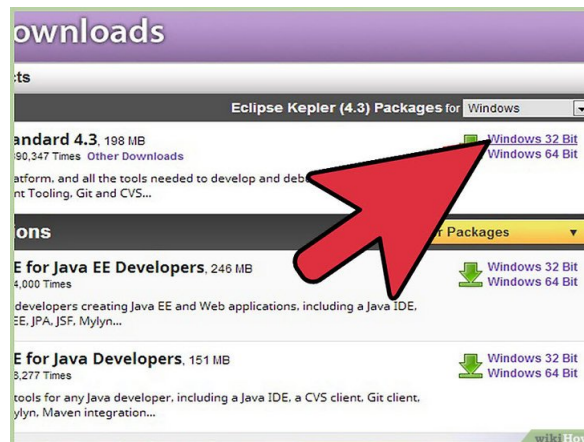


Figura 18: Selección del tipo de sistema operativo dónde se ejecutará la aplicación.

Una vez que descargues el archivo de Eclipse, necesitarás descomprimir el archivo Zip, el cual creará una carpeta de Eclipse sin comprimir. Debes extraer el archivo a la raíz de la unidad C: , así creando la carpeta C: eclipse , o sólo muévelo a esa carpeta después de extraerlo. Ya que Eclipse no tiene algún instalador, habrá un archivo dentro de la carpeta de Eclipse llamado eclipse.exe (). Puedes hacer doble clic en el archivo para ejecutar Eclipse.

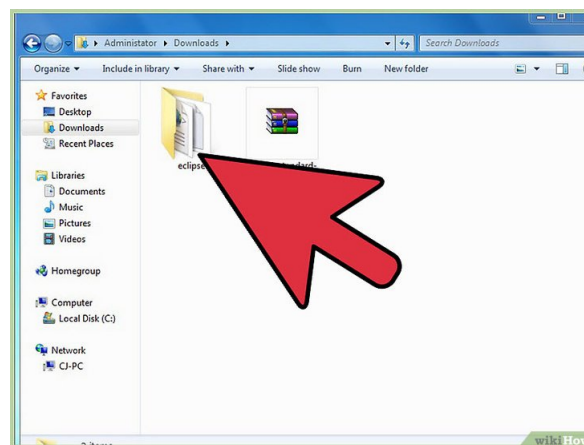


Figura 19: Archivo descomprimido de la descarga.

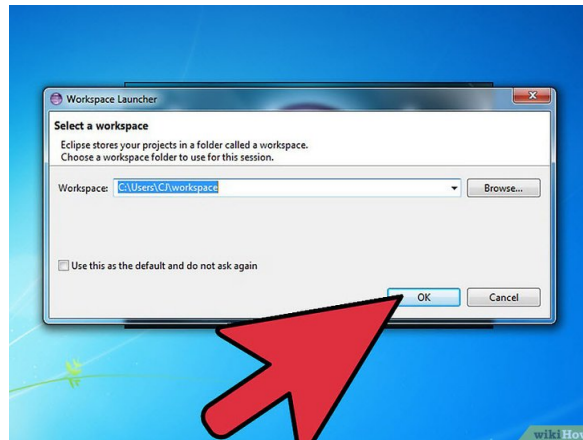


Figura 20: Inicio de un nuevo entorno de trabajo.

4. Ejecución del aplicativo

4.1. Ejecución en entornos de desarrollo

4.1.1. Eclipse

Para ejecutar la aplicación en un entorno de ejecución como para este caso Eclipse.

- Primero se abrirá el entorno de trabajo en la carpeta donde está instalada la aplicación.

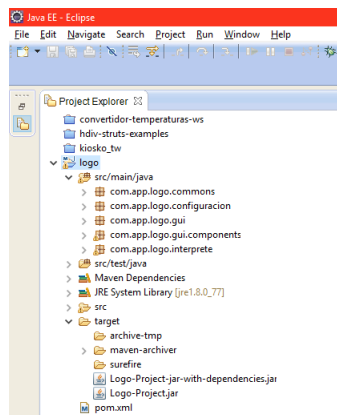


Figura 21: Menú de selección para ejecutar el proyecto.

- Hacer click derecho del icono del proyecto en el panel de 'Proyectos'.

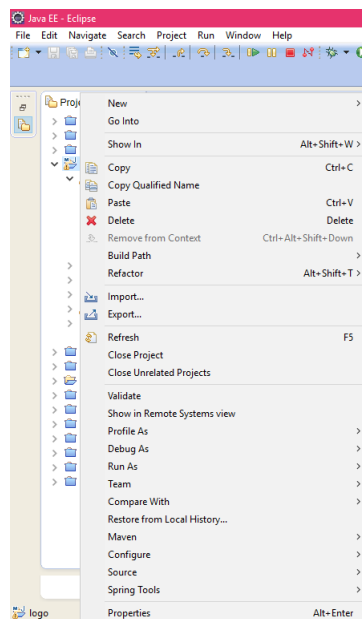


Figura 22: Menú de selección para ejecutar el proyecto.

- Seleccionar la opción: *Run As*; Finalmente, seleccionar la opción *Maven Install*

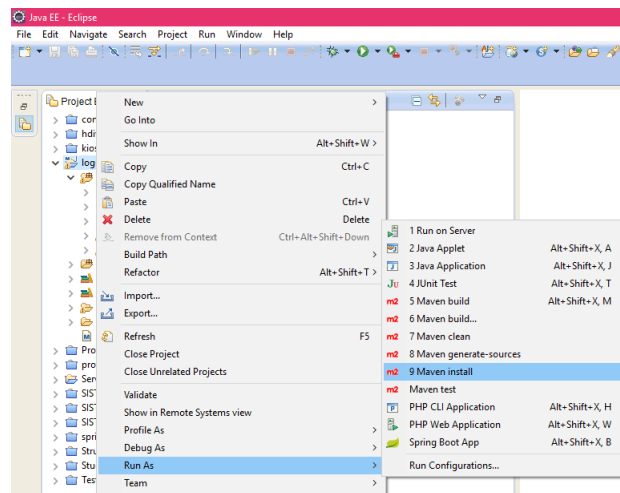


Figura 23: Ejecutar opción *Maven Install*.

- A continuación, se observa la generación de archivos JAR compilados del proyecto.

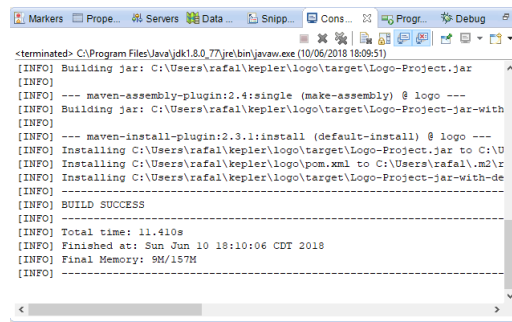


Figura 24: Generación del proyecto exitosa.

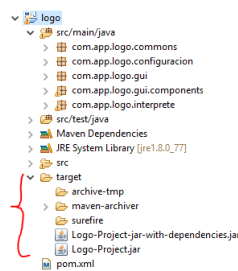


Figura 25: Carpeta target contiene archivos ejecutables.

4.1.2. Ejecución via terminal

Para ejecutar la aplicación desde una terminal o CMD, debemos realizar los siguientes pasos:

- Dirigirse a la carpeta donde está instalado el archivo <JAR>.
- Ejecutar el comando: `java -jar <nombre-archivo>.jar`

5. Casos de uso

En la figura 26 se muestra un diagrama de casos de uso que describe las acciones que realiza el usuario en la aplicación.

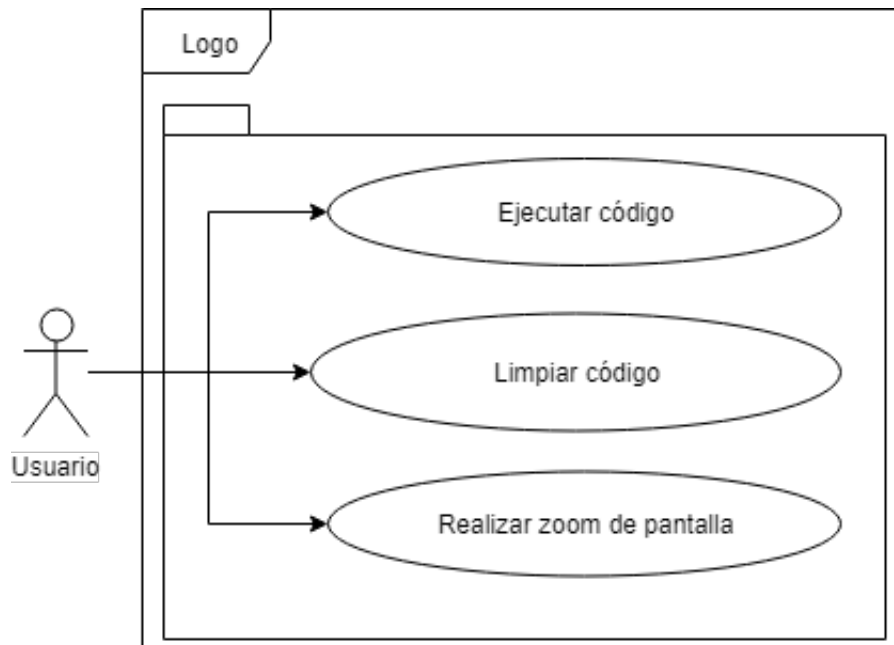


Figura 26: Acciones del usuario que puede hacer en la aplicación.

6. Modelo de información

En la figura 27 se muestra un diagrama de casos de uso que describe las acciones que realiza el usuario en la aplicación.

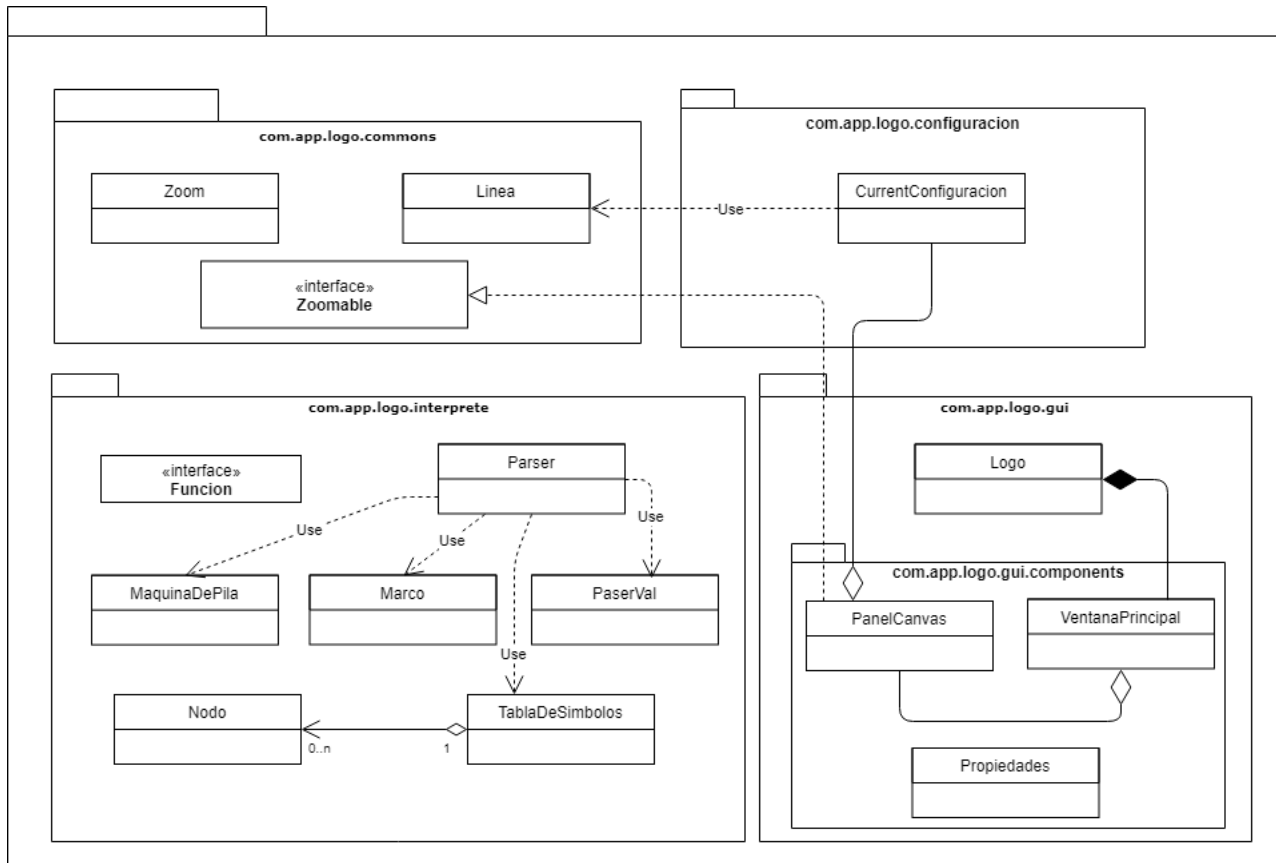


Figura 27: Arquitectura de paquetes de la aplicación.

7. Prototipo de pantalla del aplicativo

En la figura 28 se muestra la pantalla de inicio de la aplicación **Logo**.

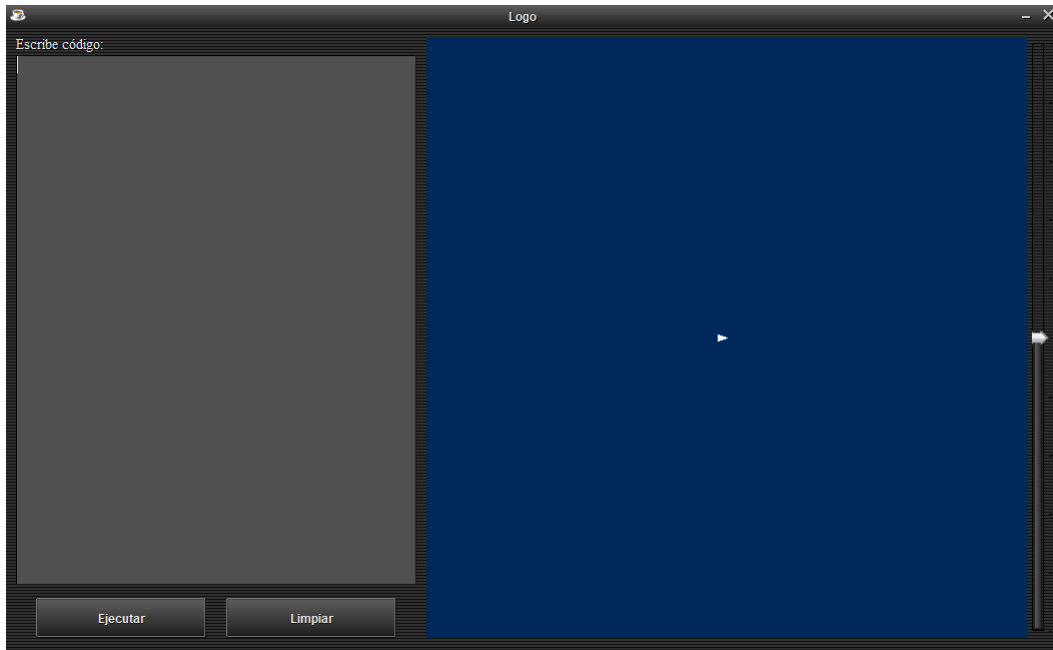


Figura 28: Pantalla principal.