



INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO



PROYECTO. LOGO
Manual Técnico
Compiladores
5CM2

Alumno:

2024630040 Manrique Godínez Daniel Alejandro

2024630188 Aguilar Bautista César Fernando

Profesor: Tecla Parra Roberto

ÍNDICE

INTRODUCCIÓN	3
REQUERIMIENTOS TÉCNICOS	4
Java	4
Java Development Kit	4
Java Runtime Environment	4
INSTALACIÓN DE JAVA EN WINDOWS	4
EJECUCIÓN DEL PROYECTO	4
Ejecución por consola	4
Ejecución por medio del Ejecutable	5
DIAGRAMA DE CLASES	5
GRAMÁTICA	6

INTRODUCCIÓN

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.

Una característica de Logo es poder producir gráficos tortuga, es decir, poder 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, como lo son:

- **AVANZAR[n]**: Permite dibujar una línea recta avanzando n pixeles en la dirección hacia la que apunta el pincel que se encuentra en el panel.
- **GIRAR[n]**: Gira la dirección del pincel n grados en el panel.
- **SUBIR[n]**: Sube el pincel que se encuentra en el panel.
- **BAJAR[n]**: Baja el pincel que se encuentra en el panel
- **COLOR[n1, n2, n3]**: Modifica el color del trazo de dibujo siguiendo los parámetros en RGB ($n1, n2, n3$)

El desarrollo de este proyecto consistirá en una interfaz gráfica con un apartado en donde él 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**

REQUERIMIENTOS TÉCNICOS

Java

Java es un lenguaje de programación y una 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 Development Kit

Java Development Kit (JDK) es un software para desarrolladores de Java. Incluye el intérprete Java, clases Java y herramientas de desarrollo Java (JDT): compilador, depurador, desensamblador, visor de applets, generador de archivos de apéndice y generador de documentación. El JDK le permite escribir aplicaciones que se desarrollan una sola vez y se ejecutan en cualquier lugar de cualquier máquina virtual Java. Las aplicaciones Java desarrolladas con el JDK en un sistema se pueden usar en otro sistema sin tener que cambiar ni recompilar el código. Los archivos de clase Java son portables a cualquier máquina virtual Java estándar.

Java Runtime Environment

Java Runtime Environment, o JRE, es una capa de software que se ejecuta encima del software del sistema operativo de una computadora y proporciona las bibliotecas de clases y otros recursos que un programa java específico necesita ejecutar. JRE es uno de los tres componentes interrelacionados para desarrollar y ejecutar programas Java

INSTALACIÓN DE JAVA EN WINDOWS

1. Descargar el jdk y el jre
2. Dale doble clic y sigue los pasos de instalación.
3. Recuerda la carpeta donde se instalaron.
4. En el buscador de Windows escribe variables de entorno y dale clic.
5. Da clic en variables de entorno.
6. Y agregar al path la dirección donde se instaló jdk.
7. Da clic en aplicar y aceptar

EJECUCIÓN DEL PROYECTO

El proyecto se puede abrir de dos formas, ya sea directamente desde consola o desde el ejecutable (.jar)

Ejecución por consola

Primero debemos de compilar todos nuestros archivos con extensión *.java*.

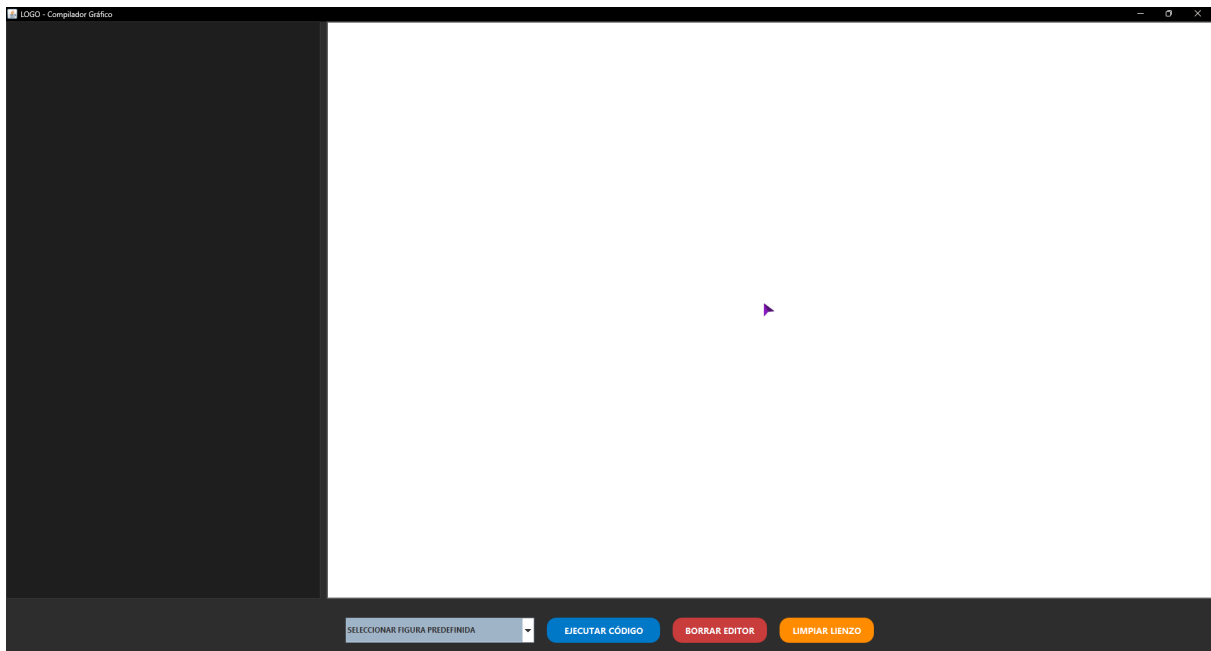
Ejecutaremos el comando:

\$ javac *.java

dentro de nuestra carpeta **CodigoJAVA**

Al compilar todos nuestros archivos *.java* nos creará varios archivos con extensión *.class*

Posteriormente, tendremos que ejecutar nuestro programa, para ello usaremos el comando **\$ java Main** y debería de abrirse una ventana cómo la siguiente:

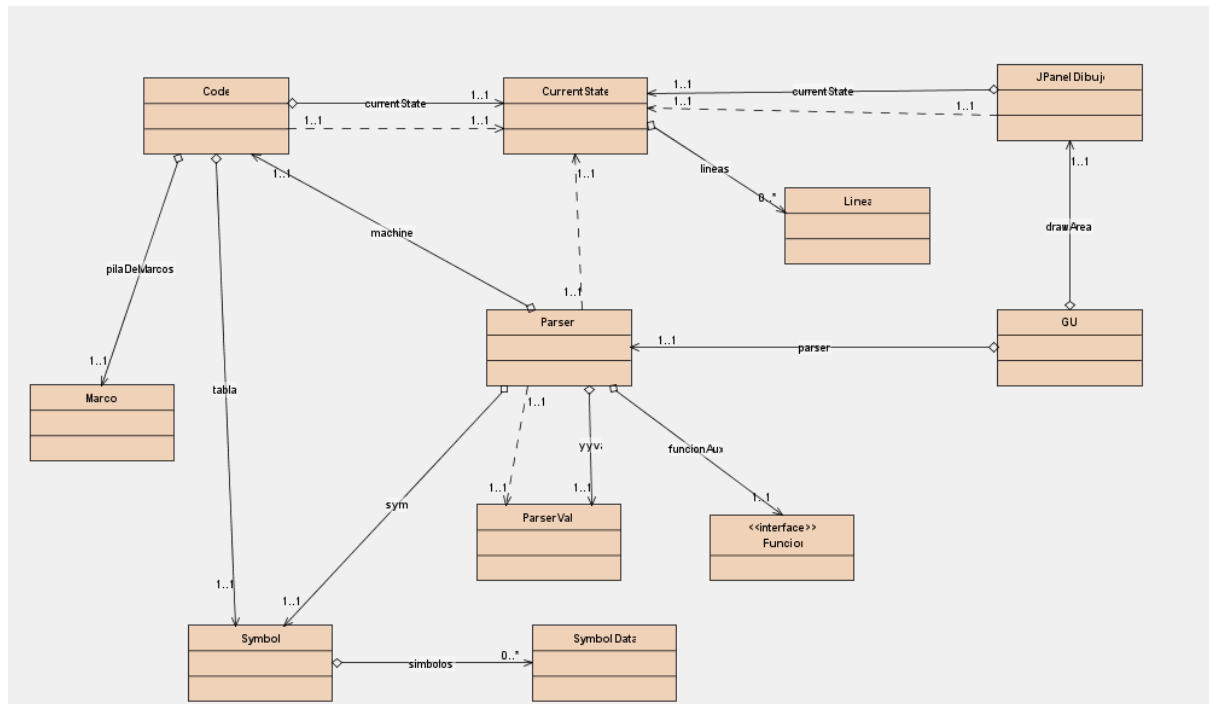


Ejecución por medio del Ejecutable

La otra forma es dar doble clic a **LogoCompiler.jar**, y debe abrir la misma ventana que se abrió anteriormente.



DIAGRAMA DE CLASES



GRAMÁTICA

```

%token IF ELSE WHILE FOR FUNC RETURN PARAMETRO PROC FNCT NUMBER VAR
%token AND OR COMP DIFERENTES MAY MEN MAYI MENI
%right '='
%left '+' '-'
%left '*'
%left '/'
%left COMP
%left DIFERENTES
%left MAY
%left MAYI
%left MEN
%left MENI
%left '!'
%left AND
%left OR
%right RETURN
%%

```

```

list:
    | list'n'
    | list linea 'n'
    ;

```

```

linea: exp ':' {$$ = $1;}
      |stmt {$$ = $1;}
      |linea exp ':' {$$ = $1;}
      |linea stmt {$$ = $1;}
      ;

```

```

exp: VAR { $$ = new ParserVal(machine.agregarOperacion("varPush_Eval"));
          machine.agregar($1.sval);
        }
    | '-' exp {
          $$ = new ParserVal(machine.agregarOperacion("negativo"));
        }
    | NUMBER {
          $$ = new ParserVal(machine.agregarOperacion("constPush"));
          machine.agregar($1.dval);
        }
    | VAR '=' exp {
          $$ = new ParserVal($3.ival);
          machine.agregarOperacion("varPush");
          machine.agregar($1.sval);
          machine.agregarOperacion("asignar");
          machine.agregarOperacion("varPush_Eval");
          machine.agregar($1.sval);
        }
    | exp '*' exp {
          $$ = new ParserVal($1.ival);
          machine.agregarOperacion("MUL");
        }
    | exp '+' exp {
          $$ = new ParserVal($1.ival);
          machine.agregarOperacion("SUM");
        }
    | exp '-' exp {
          $$ = new ParserVal($1.ival);
          machine.agregarOperacion("RES");
        }
    | '(' exp ')' {
          $$ = new ParserVal($2.ival);
        }
    | exp COMP exp {
          machine.agregarOperacion("EQ");
          $$ = $1;
        }
    | exp DIFERENTES exp {
          machine.agregarOperacion("NE");
          $$ = $1;
        }
    | exp MEN exp {
          machine.agregarOperacion("LE");
          $$ = $1;
        }
    | exp MENI exp {
          machine.agregarOperacion("LQ");
          $$ = $1;
        }
    | exp MAY exp {
          machine.agregarOperacion("GR");
          $$ = $1;
        }
    | exp MAYI exp {
          machine.agregarOperacion("GE");
          $$ = $1;
        }
    | exp AND exp {
          machine.agregarOperacion("AND");
          $$ = $1;
        }
    | exp OR exp {
          machine.agregarOperacion("OR");
          $$ = $1;
        }
    | '!' exp {
          machine.agregarOperacion("NOT");
          $$ = $2;
        }
    | RETURN exp { $$ = $2; machine.agregarOperacion("_return"); }
    | PARAMETRO { $$ = new ParserVal(machine.agregarOperacion("push_parametro")); machine.agregar((int)$1.ival); }

[nombreProc '!' arglist ']' { $$ = new ParserVal(machine.agregarOperacionEn("invocar",($1.ival))); machine.agregar(null); }
//instrucciones tiene la estructura necesaria para la lista de argumentos

//Lamada a Procedimiento

```

```

arglist:
|exp {$$ = $1; machine.agregar("Limite");}
|arglist ',' exp {$$ = $1; machine.agregar("Limite");}
;

nop: {$$ = new ParserVal(machine.agregarOperacion("nop"));}
;

stmt: if '(' exp stop ')' '(' linea stop ')' ELSE '(' linea stop ')' {
    $$ = $1;
    machine.agregar($7.ival, $1.ival + 1);
    machine.agregar($12.ival, $1.ival + 2);
    machine.agregar(machine.numeroDeElementos() - 1, $1.ival + 3);
}
| if '(' exp stop ')' '(' linea stop ')' nop stop{
    $$ = $1;
    machine.agregar($7.ival, $1.ival + 1);
    machine.agregar($10.ival, $1.ival + 2);
    machine.agregar(machine.numeroDeElementos() - 1, $1.ival + 3);
}
| while '(' exp stop ')' '(' linea stop ')' stop{
    $$ = $1;
    machine.agregar($7.ival, $1.ival + 1);
    machine.agregar($10.ival, $1.ival + 2);
}
| for '(' instrucciones stop ';' exp stop ';' instrucciones stop ')' '(' linea stop ')' stop{
    $$ = $1;
    machine.agregar($6.ival, $1.ival + 1);
    machine.agregar($9.ival, $1.ival + 2);
    machine.agregar($13.ival, $1.ival + 3);
    machine.agregar($16.ival, $1.ival + 4);
}
| funcion nombreProc '(' ')' '(' linea null ')'
| procedimiento nombreProc '(' ')' '(' linea null ')'
| instruccion '[' arglist ']' ';' {
    $$ = new ParserVal($1.ival);
    machine.agregar(null);
}
;

instruccion: FNCT {
    $$ = new ParserVal(machine.agregar((Funcion)($1.obj)));

}
;

```



```

procedimiento: PROC { machine.agregarOperacion("declaracion"); }
;

funcion: FUNC { machine.agregarOperacion("declaracion"); }
;

nombreProc: VAR { $$ = new ParserVal(machine.agregar($1.sval)); }
;

null: { machine.agregar(null); }
;

stop: { $$ = new ParserVal(machine.agregarOperacion("stop")); }
;

if: IF {
    $$ = new ParserVal(machine.agregarOperacion("IF_ELSE"));
    machine.agregarOperacion("stop");//then
    machine.agregarOperacion("stop");//else
    machine.agregarOperacion("stop");//siguiente comando
}
;

while: WHILE {
    $$ = new ParserVal(machine.agregarOperacion("WHILE"));
    machine.agregarOperacion("stop");//cuerpo
    machine.agregarOperacion("stop");//final
}
;

for : FOR {
    $$ = new ParserVal(machine.agregarOperacion("FOR"));
    machine.agregarOperacion("stop");//condicion
    machine.agregarOperacion("stop");//instrucción final
    machine.agregarOperacion("stop");//cuerpo
    machine.agregarOperacion("stop");//final
}

instrucciones: { $$ = new ParserVal(machine.agregarOperacion("nop")); }
| exp { $$ = $1; }
| instrucciones ',' exp { $$ = $1; }
;

```