UNIVERSIDAD UTE

FACULTAD DE CIENCIAS DE LA INGENIERIA E INDUSTRIAS

CARRERA COMPUTACIÓN

CUARTO SEMESTRE

LENGUAJES Y COMPILADORES

MANUAL DE USUARIO

REALIZADO POR

STALIN MINDA

MIGUEL MONAR

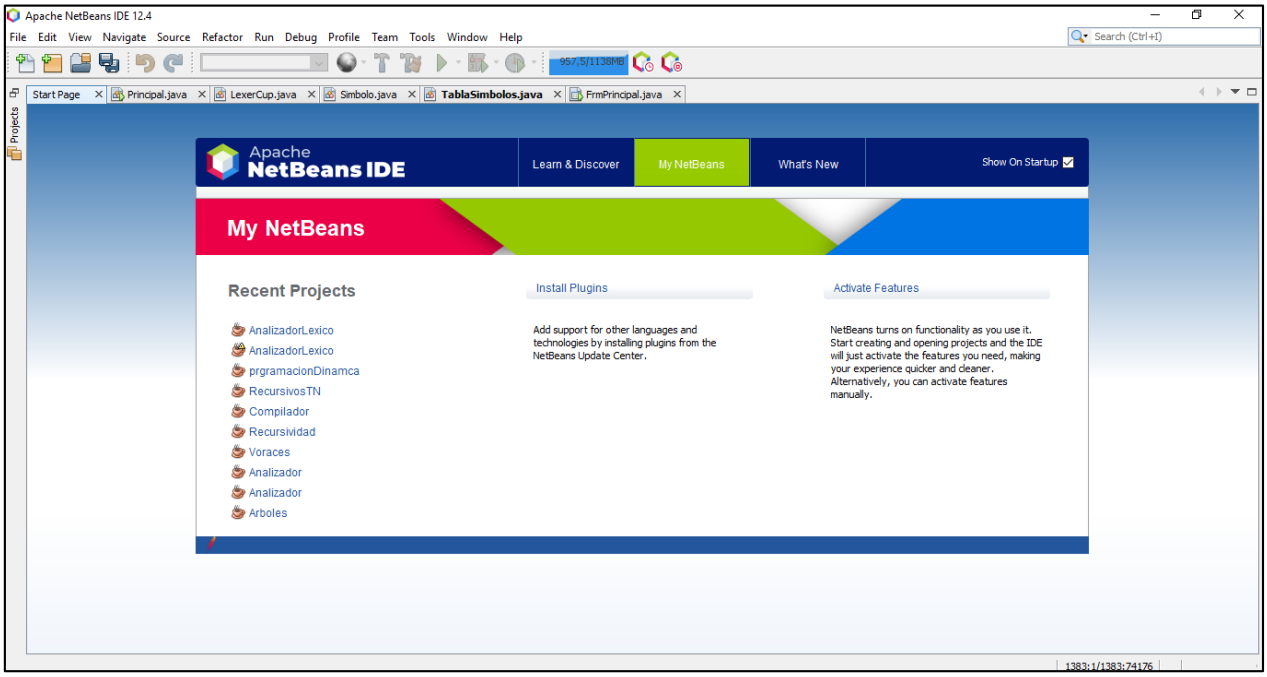
10 de agosto de 2021

Contenido

Requisitos

Netbeans

Para el uso de este programa es necesario el IDE NetBeans.12.4 o una versión superior.

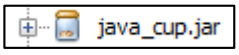


JFlex

Se utilizó la librería JFLex para realizar el analizador sintáctico.

JavaCup

Se utilizaron la librería java_cup para el análisis sintáctico



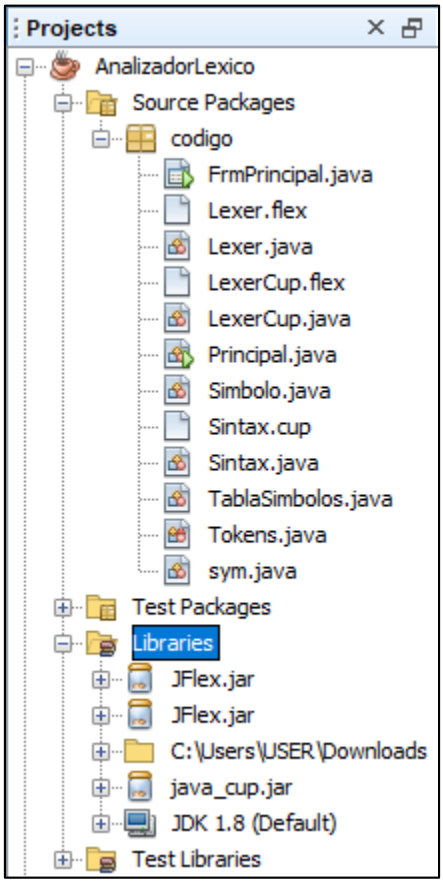
Acciones previas a la ejecución

Conflictos de librerías

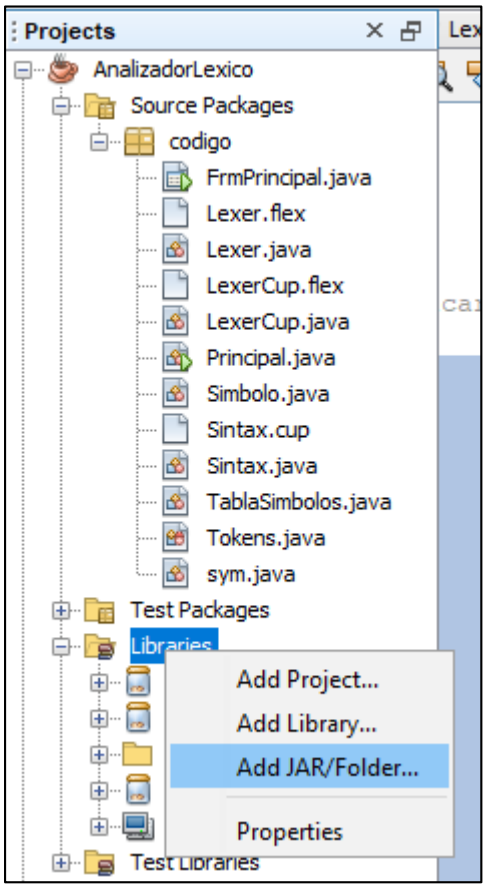
Tras haber importado el proyecto y haberlo abierto en NetBeans, es importante tener descargadas las librerías que se han mencionado en los requisitos.

- JFLex.jar
- JavaCup.jar

Se debe verificar que estas dos librerías se encuentren actualizadas en la carpeta ***Libraries*** del proyecto para que funcione correctamente.



Si no se encuentran en esta sección, debe descargarlas, e importarlas.



Actualizando rutas

Una vez que hemos resuelto el paso anterior, es importante ir a la clase principal para actualizar las rutas de los archivos con el directorio en el cual usted ha descomprimido el proyecto.

Para ello bastará con actualizar toda la ruta que se encuentra antes de "**/AnalizadorLexico/src/codigo/Lexer.flex**" puesto que esto se mantendrá igual.

Por lo que usted deberá reconocer el directorio en el cual ha descomprimido el proyecto, y luego deberá cambiar esta parte en cada una de las direcciones de la clase principal, como se muestra a continuación:

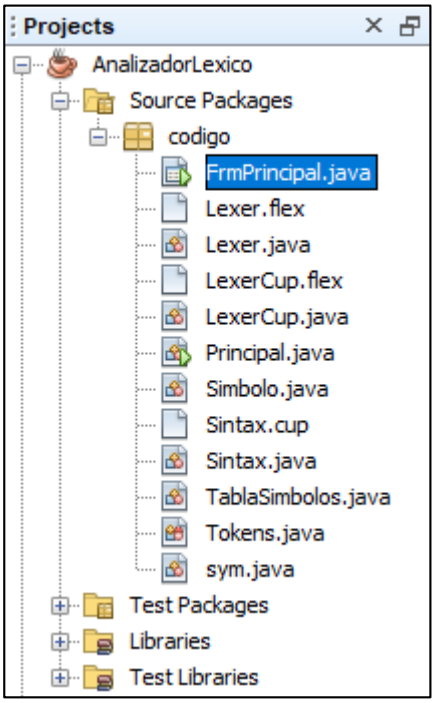
```java
public class Principal {
    public static void main(String[] args) throws Exception {

        String ruta1 = "C:/Users/USER/Desktop/Final - copia/AnalizadorLexico/src/codigo/Lexer.flex";
        String ruta2 = "C:/Users/USER/Desktop/Final - copia/AnalizadorLexico/src/codigo/LexerCup.flex";
        String[] rutaS = {"-parser", "Sintax", "C:/Users/USER/Desktop/Final - copia/AnalizadorLexico/src/codigo/Sintax.cup"};
        generar(ruta1, ruta2, rutaS);
    }
    public static void generar(String ruta1, String ruta2, String[] rutaS) throws IOException, Exception{
        File archivo;
        archivo = new File(ruta1);
        JFlex.Main.generate(archivo);
        archivo = new File(ruta2);
        JFlex.Main.generate(archivo);
        java_cup.Main.main(rutaS);

        Path rutaSym = Paths.get("C:/Users/USER/Desktop/Final - copia/AnalizadorLexico/src/codigo/sym.java");
        if (Files.exists(rutaSym)) {
            Files.delete(rutaSym);
        }
        Files.move(
                Paths.get("C:/Users/USER/Desktop/Final - copia/AnalizadorLexico/sym.java"),
                Paths.get("C:/Users/USER/Desktop/Final - copia/AnalizadorLexico/src/codigo/sym.java")
        );
        Path rutaSin = Paths.get("C:/Users/USER/Desktop/Final - copia/AnalizadorLexico/codigo/Sintax.java");
        if (Files.exists(rutaSin)) {
            Files.delete(rutaSin);
        }
        Files.move(
                Paths.get("C:/Users/USER/Desktop/Final - copia/AnalizadorLexico/Sintax.java"),
                Paths.get("C:/Users/USER/Desktop/Final - copia/AnalizadorLexico/src/codigo/Sintax.java")
        );
    }
}
```

Hecho esto, debe ejecutar la clase principal, y el programa estará listo para usarse.

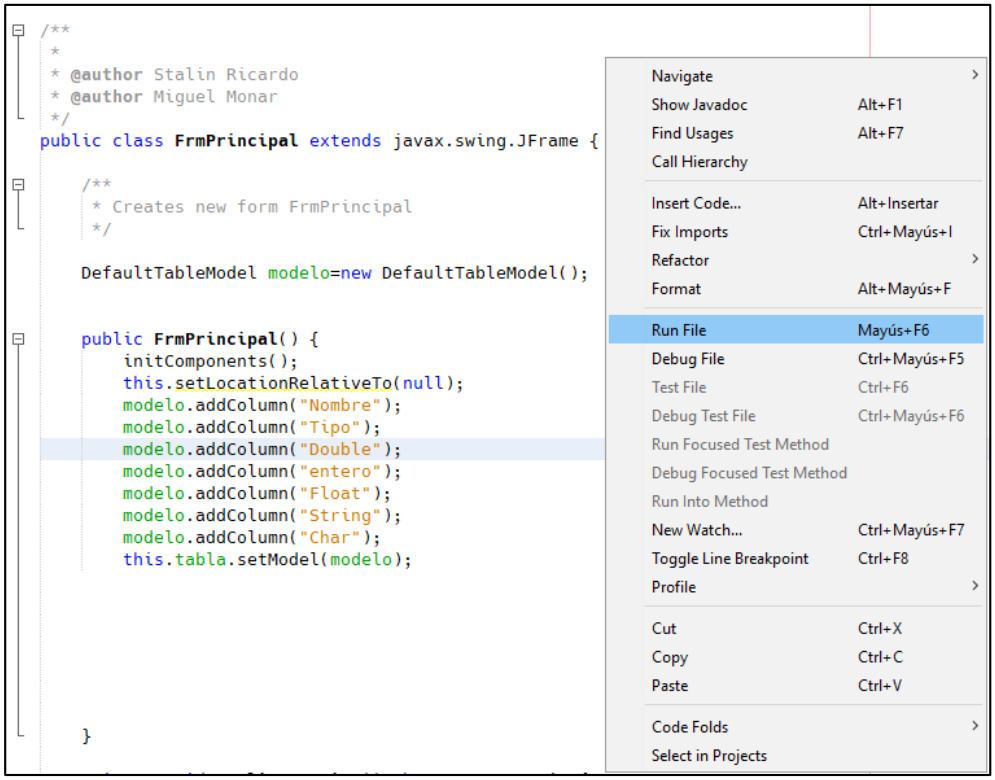
Ejecución del programa

FrmPrincipal.java

Para ejecutar el programa debemos ir a la clase **FrmPrincipal.java.**



Una vez dentro de la clase **FrmPrincipal.java** solo debemos dar clic derecho sobre el código y seleccionamos la opción ***Run File.***



Sentencias reconocidas

No se han incluido todas las sentencias del lenguaje Java, pero se ha intentado implementar aquellas con las cuales es posible realizar los 3 tipos de análisis y evidenciar el llenado de tabla de símbolos y el manejo de errores sintáctico y semánticos. Estas sentencias son:

SENTENCIA	EJEMPLO	ANÁLISIS		
		LÉXICO	SINTÁXTICO	SEMÁNTICO
Declarar una variable entera	`int a;`	x	x	
Declarar una variable flotante	`float a;`	x	x	
Declarar una variable double	`double a;`	x	x	
Definir una variable entera;	`int a = 12;`	x	x	x
Definir una variable flotante;	`float a = 12,5;`	x		x
Definir una variable doublé;	`double a = 3,14151648;`	x		x
Definir una variable String	`String cadena = “Hola mundo”;`	x		x
Definir una variable String vacia	`String cadena = “”;`	x	x	
Definir una variable char	`char c = ‘x’;`	x		x
Definir una variable entera en función de otra.	`int a  = 13;` `int b = a;`	x	x	x
Definir una variable flotante en función de otra	`float a = 12,5;` `float c = a;`	x		x
Definir una variable double en función de otra	`double pi = 3,1415616;` `double copia = pi;`	x		x
Definir una variable String en función de otra	`Stirng c1 = “perro”;` `String c2 = c1;`	x		x
Definir una variable char en función de otra	`char c1 = ‘gg’;` `char c2 = c1;`	x		x

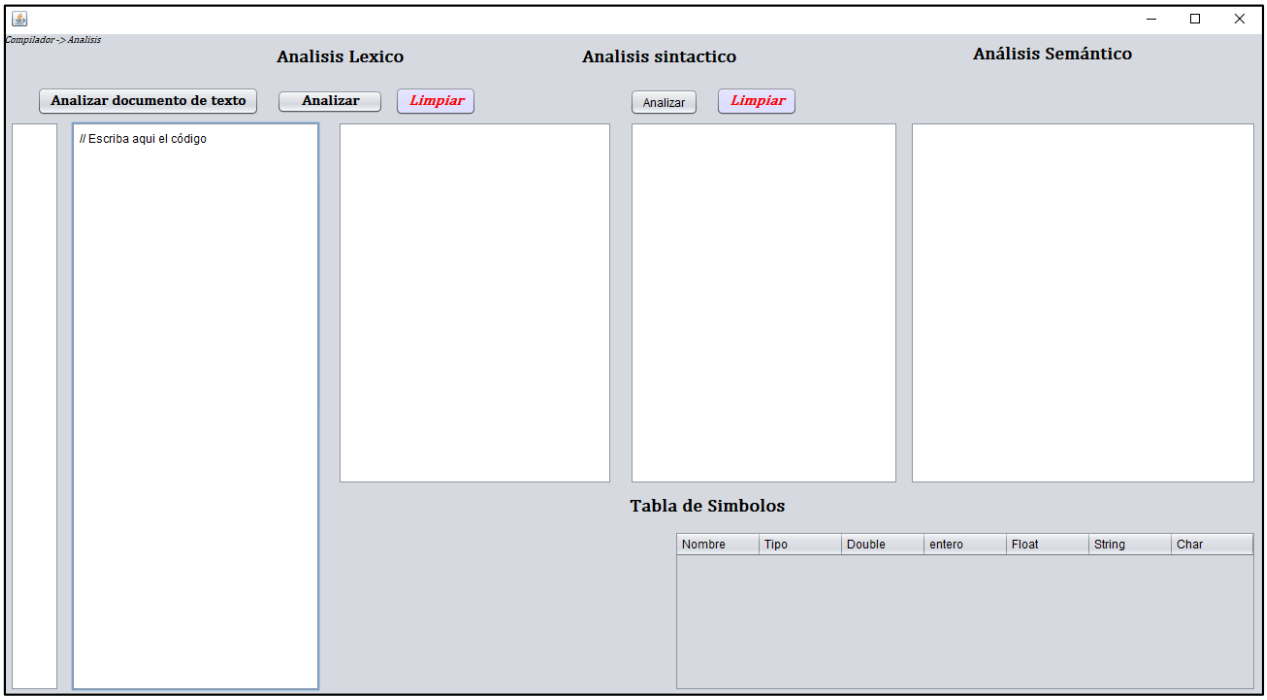
Usar un método main	`int main (){`
	`    int a = 2;`
	`}`
		x	x	x

if	if (a<=b){
	
	}
		x	x	

boolean	boolean bandera;	x	x	

Interfaz gráfica de usuario

Al ejecutar la clase **FrmPrincipal.java,** se le aparecerá la siguiente ventana:



Aquí usted puede elegir entre escribir código directamente en el cuadro de texto, o hacer clic en el botón **“Analizar documento de texto”,** para leer un fichero **.txt** que contenga código dentro de su ordenador.

Luego de haber elegido un fichero, o haber escrito el código manualmente, usted puede hacer clic en cualquiera de los dos botones **“Analizar”** para proceder con los análisis léxico, o sintáctico. Recuerde que el análisis semántico se realiza al mismo tiempo que se llena la tabla de símbolos por lo que no debe preocuparse si no le aparece nada en el cuadro de texto de este análisis, ya que esto indicará que no hay errores semánticos siempre que las sentencias se encuentren dentro de las sentencias permitidas.

Una ejecución se vería de la siguiente manera: