



Universidad de San Carlos de Guatemala

Facultad de Ingeniería

Escuela de Ingeniería en ciencias de sistemas.

Manual de usuario

Nombre del estudiante

José Luis Espinoza Jolón - 202202182

Guatemala, septiembre de 17 de 2024

Contenido

Descripción	3
Entorno de trabajo	4
• Crear Archivo	4
• Abrir Archivo.....	5
Otras funcionalidades	6
• Guardar	6
• Ejecutar	6
• Reportes de errores	6
• Reporte de tabla de símbolos.....	6
• Área de entrada	6
• Area de consola	6
Uso de lenguaje	7
• Comentarios.....	7
• Variables	7
• Operadores Aritméticos	8
• Operadores Lógicos	8
• Sentencias de control de flujo	8
• Estructura de datos	9
• Stucts	9
• Funciones	9
• Ejemplo de ejecución del programa	10
• Ejemplo de reportes	11

Descripción

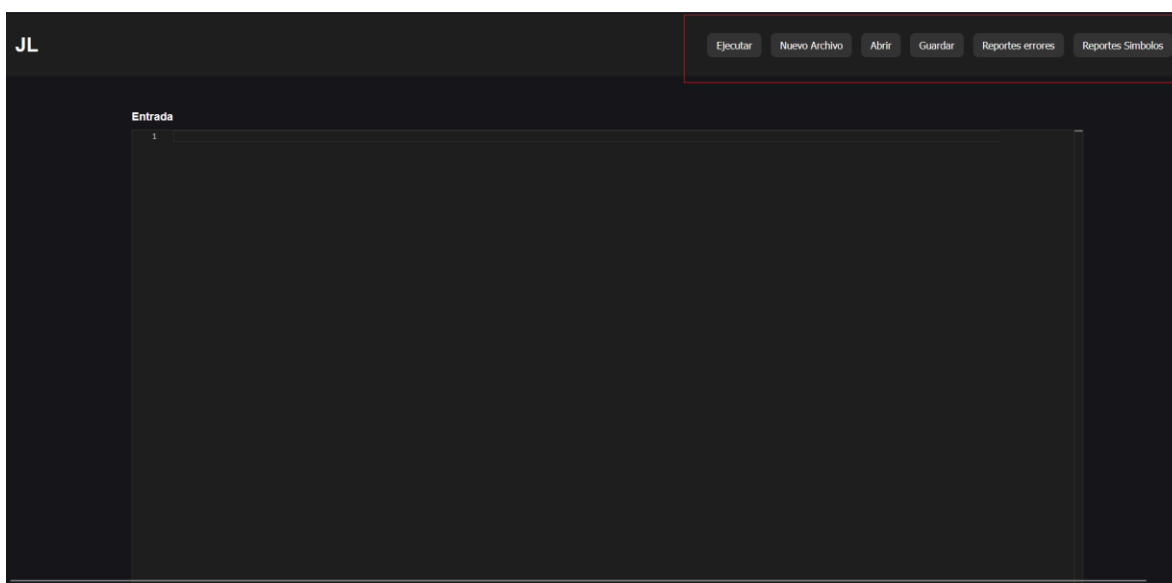
desarrolle un intérprete para el lenguaje de programación OakLand, el cual toma inspiración de la sintaxis de Java, aunque esta no es su característica principal.

OakLand sobresale por su capacidad para soportar diversos paradigmas de programación, como la orientación a objetos, la programación funcional y la programación procedimental.

Adicionalmente, se deberá crear una plataforma simple pero robusta para permitir la creación, apertura, edición e interpretación de código en OakLand. Este entorno de desarrollo (IDE) se implementará utilizando JavaScript puro y será alojado en Github Pages.

Entorno de trabajo

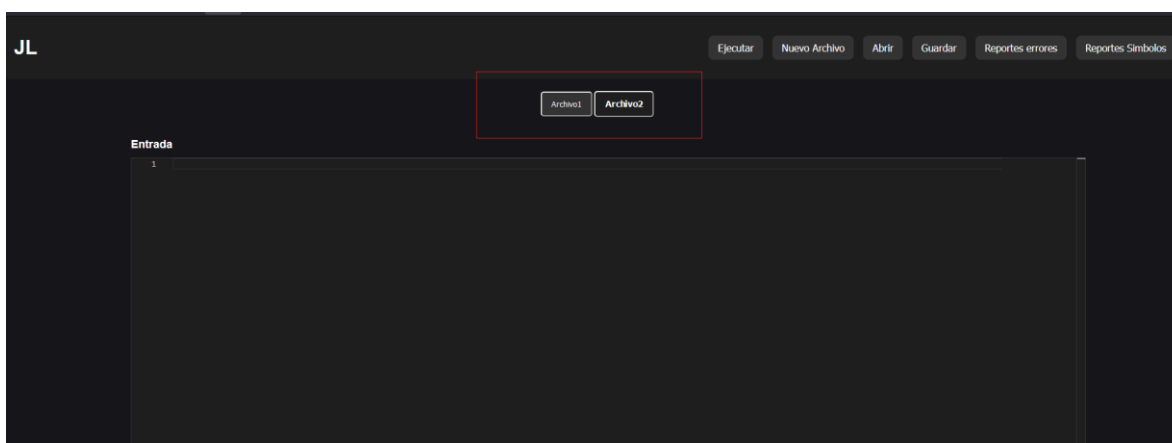
El usuario al ejecutar el programa visualizara la siguiente vista



Las funciones que puede usar el usuario son la siguiente.

- [Crear Archivo](#)

El editor tiene la capacidad de crear múltiples archivos



- Abrir Archivo

El editor tiene la capacidad de abrir archivos con las extensiones. oak cuyo contenido se deberá mostrar en el área de entrada en una pestaña nueva con el nombre del archivo.

Nombre	Fecha de modificación	Tipo	Tamaño
codigo	11/09/2024 12:10	Carpeta de archivos	
OLC2_252024	7/09/2024 10:07	Carpeta de archivos	
Proyecto1 aa	10/08/2024 22:23	Carpeta de archivos	
Usac	12/09/2024 09:19	Carpeta de archivos	
prueba1.oak	12/09/2024 10:21	Archivo OAK	18 KB
prueba2.oak	11/09/2024 22:07	Archivo OAK	4 KB
prueba3.oak	11/09/2024 21:48	Archivo OAK	4 KB
prueba4.oak	11/09/2024 16:32	Archivo OAK	18 KB

Al seleccionar el archivo, automáticamente lo abre en la consola de entrada de esta manera.

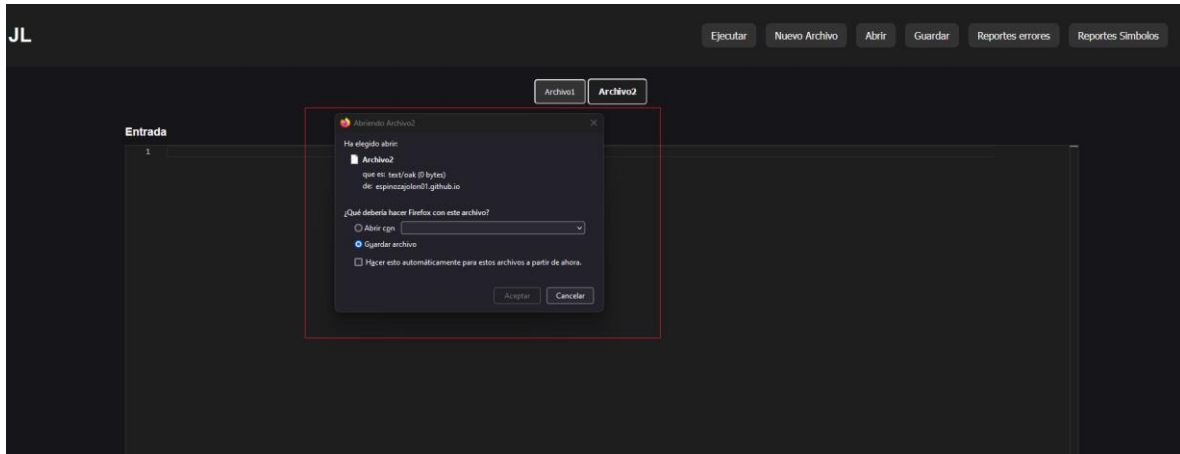
The screenshot shows the JL editor interface. At the top, there are buttons for 'Ejecutar', 'Nuevo Archivo', 'Abrir', 'Guardar', 'Reportes errores', and 'Reportes Símbolos'. Below these buttons, there is a tab bar with three tabs: 'Archivo1', 'Archivo2', and 'prueba1.oak'. The 'prueba1.oak' tab is selected. The main area of the editor is labeled 'Entrada' and contains the following Java code:

```
1 void FactorialIterativo(int n2){
2
3     int numeroFactorial = n2;
4
5     while (numeroFactorial > -1) {
6         mostrarFactorial(numeroFactorial);
7         numeroFactorial -- 1;
8     }
9
10
11 void mostrarFactorial(int n2){
12     int fact = 1;
13     string cadena1 = "El factorial de: " + toString(n2) + " = ";
14     if (n2 != 0) {
15         for (int i = n2; i > 0; i -- 1) {
16             fact = fact * i;
17             cadena1 = cadena1 + toString(i);
18             if (i > 1) {
19                 cadena1 = cadena1 + " * ";
20             }
21         }
22     }
23     System.out.println(cadena1 + fact);
24 }
```

Otras funcionalidades

- Guardar

El editor debe de tener la capacidad de guardar el estado del archivo en el que se está trabajando



- Ejecutar

Se envía la entrada de la pestaña actualmente seleccionada al interprete con la finalidad de realizar todo el analisis de las instrucciones dadas

- Reportes de errores

Se mostrarán todos los errores que encuentra el programa

- Reporte de tabla de símbolos

Se mostrará todos los símbolos encontrados durante la ejecución

- Área de entrada

Se va a visualizar el código generado por el usuario

- Area de consola

Se mostrará el resultado y mensajes generado por el programa

Uso de lenguaje

El lenguaje OakLand está inspirado en la sintaxis del lenguaje Java, por lo tanto, se conforma por un subconjunto de instrucciones de este, con la diferencia de que OakLand

tendrá una sintaxis más reducida, pero sin perder las funcionalidades que caracterizan al

lenguaje original

- **Comentarios**

Un comentario es un componente léxico del lenguaje que no es tomado en cuenta para el

análisis sintáctico de la entrada. Existirán dos tipos de comentarios:

- Los comentarios de una línea que serán delimitados al inicio con el símbolo de // y al

final como un carácter de finalización de línea.

- Los comentarios con múltiples líneas que empezarán con los símbolos /* y terminarán con los símbolos */

- **Variables**

Una variable es un elemento de datos cuyo valor puede cambiar durante el curso de la

ejecución de un programa siempre y cuando sea el mismo tipo de dato. Una variable

cuenta con un nombre y un valor, los nombres de variables no pueden coincidir con una

palabra reservada.

Para poder utilizar una variable se tiene que definir previamente, la declaración nos permite

crear una variable y asignarle un valor o sin valor

- **Operadores Aritméticos**

Los operadores aritméticos toman valores numéricos de expresiones y retornan un valor

numérico único de un determinado tipo. Los operadores aritméticos estándar son adición

o suma +, sustracción o resta -, multiplicación *, y división /, adicionalmente vamos a

trabajar el módulo %.

- **Operadores Lógicos**

Los operadores lógicos comprueban la veracidad de alguna condición. Al igual que los

operadores de comparación, devuelven el tipo de dato boolean con el valor true ó false.

- **Sentencias de control de flujo**

Las estructuras de control permiten regular el flujo de la ejecución del programa. Este flujo

de ejecución se puede controlar mediante sentencias condicionales que realicen ramificaciones e iteraciones. Se debe considerar que estas sentencias se encontrarán

únicamente dentro funciones.

- Estructura de datos

Las estructuras de datos en el lenguaje OakLand son los componentes que nos permiten

almacenar un conjunto de valores agrupados de forma ordenada, las estructuras básicas

que incluye el lenguaje son los Array

- Structs

El lenguaje OakLand tiene la capacidad de permitir al programador en crear sus propios

tipos compuestos personalizados, estos elementos se les denomina structs, los structs

permiten la creación de estructuras de datos y manipulación de información de una manera

más versátil. Estos están compuestos por tipos primitivos o por otros structs.

Un struct que contiene otro struct como una de sus propiedades no puede ser del mismo

tipo que el struct que lo contiene.

- Funciones

En términos generales, una función es un "subprograma" que puede ser llamado por código

externo (o Interno en caso de recursión) a la función. Al igual que el programa en sí mismo,

una función se compone de una secuencia de declaraciones y sentencias, que conforman el

cuerpo de la función.

Se pueden pasar valores a una función y la función puede devolver un valor. Para devolver.

un valor específico, una función debe tener una sentencia return, que especifique el valor a

devolver. Además, la función debe tener especificado el tipo de valor que va a retornar.

- Ejemplo de ejecución del programa

```

1 // ***** Funciones no recursivas sin parámetros *****
2
3 System.out.println("***** Funciones no recursivas sin parámetros *****");
4
5 System.out.println("1. void");
6
7 void saludar() {
8     System.out.println("Hola");
9 }
10
11 saludar();
12 System.out.println("");
13
14 System.out.println("2. Con retorno");
15
16 int sumar() {
17     return 10 + 20;
18 }
19
20 System.out.println(sumar());
21
22 // ***** Funciones no recursivas con parámetros *****
23
24 System.out.println("***** Funciones no recursivas con parámetros *****");
25
26 System.out.println("1. void");
27
28 void saludar2(string nombre) {
29     System.out.println("Hola " + nombre);
30 }
31
32 saludar2("Mundo");
33
34 System.out.println("");
35
36 System.out.println("2. Con retorno");
37
38 int sumar2(int a, int b) {
39     return a + b;

```

```

35 2. parseFloat
36 10.0
37 10.5
38
39 3. toString
40 10
41 10.510
42 php es el mejor lenguaje: false
43
44 4. toLowerCase
45 hola mundo
46
47 5. toUpperCase
48 HOLA MUNDO
49
50 6. typeof
51 int
52 float
53 string
54 char
55 boolean
56
57 Error: El valor 'Hola' no es un número válido. at line 116, column 19
58 Error: El tipo del valor no coincide con el tipo int at line 116, column 1
59 Error: No válido ese tipo de dato en parseInt at line 117, column 20
60 Error: El tipo del valor no coincide con el tipo int at line 117, column 1
61 Error: El valor 'Hola' no es un número válido. at line 129, column 22
62 Error: El tipo del valor no coincide con el tipo float at line 129, column 1
63 Error: Error: No es válida esa operación en toLowerCase at line 143, column 21
64 Error: El tipo del valor no coincide con el tipo string at line 143, column 1
65 Error: Error: No es válida esa operación en toUpperCase at line 151, column 21
66 Error: El tipo del valor no coincide con el tipo string at line 151, column 1

```

- Ejemplo de reportes

JL

Ejecutar Nuevo Archivo Abrir Guardar Reportes errores Reportes Simbolos

Entrada

```
1 // =====
```

Tabla de Simbolos

ID	Tipo simbolo	Tipo dato	Ámbito	Línea	Columna
saludar	Funcion	void	Global	7	1
sumar	Funcion	int	Global	16	1
saludar2	Funcion	void	Global	28	1
sumar2	Funcion	int	Global	38	1
factorial	Funcion	int	Global	50	1
fibonacci	Funcion	int	Global	63	1
hanoi	Funcion	void	Global	76	1
total	Funcion	int	Global	92	1
c	variable	int	Global	94	5
sumar4	Funcion	int	Global	96	5
numero10	variable	int	Global	112	1
numeroDecimal	variable	int	Global	113	1
numero	variable	int	Global	156	1
decimal	variable	float	Global	157	1
cadena	variable	string	Global	158	1
letra	variable	char	Global	159	1

JL

Ejecutar Nuevo Archivo Abrir Guardar Reportes errores Reportes Simbolos

Entrada

```
1 // =====
2
3 System.out.println("***** Funciones no recursivas sin parámetros *****");
4 System.out.println("1. void");
5
6
7 void saludar() {
```

Reporte de Errores

No.	Descripción	Línea	Columna	Tipo
1	El valor 'Hola' no es un número válido.	116	19	SEMANTICO
2	El tipo del valor no coincide con el tipo int	116	1	SEMANTICO
3	No valido ese tipo de dato en parseInt	117	20	SEMANTICO
4	El tipo del valor no coincide con el tipo int	117	1	SEMANTICO
5	El valor 'Hola' no es un número válido.	129	22	SEMANTICO
6	El tipo del valor no coincide con el tipo float	129	1	SEMANTICO
7	Error. No es valida esa operacion en toLowerCase	143	21	SEMANTICO
8	El tipo del valor no coincide con el tipo string	143	1	SEMANTICO
9	Error. No es valida esa operacion en toUpperCase	151	21	SEMANTICO
10	El tipo del valor no coincide con el tipo string	151	1	SEMANTICO

```
15
16 System.out.println("2. Con retorno");
17
18 int sumar2(int a, int b) {
19     return a + b;
20 }
```