MANUAL DE USUARIO

Oacklang

1. Introducción

Bienvenido al entorno de desarrollo integrado (IDE) para el lenguaje de programación OakLand. Este manual de usuario está diseñado para guiarte a través de las funcionalidades del IDE, desde la creación y edición de archivos hasta la ejecución y depuración de código.

2. Inicio Rápido

2.1. Acceso al IDE

Para acceder al IDE de OakLand, abre tu navegador web y navega a la página del proyecto alojada en GitHub Pages. El IDE es completamente accesible desde cualquier navegador moderno sin necesidad de instalaciones adicionales.

2.2. Interfaz de Usuario

La interfaz del IDE se divide en varias secciones principales:

- Editor de Código: Área donde puedes escribir y editar el código.
- Consola: Muestra los resultados de la ejecución del código, así como mensajes de error y advertencias.
- Controles: Botones para crear, abrir, guardar archivos y ejecutar el código.

3. Uso del Editor de Código

3.1. Creación de Archivos

1. Haz clic en el botón "Crear Archivo".

2. Se añadirá un nuevo área de texto en el editor donde podrás comenzar a escribir tu código.

3.2. Apertura de Archivos

1. Haz clic en el botón "Abrir Archivo" y selecciona un archivo con extensión `.oak` desde tu computadora.

2. El contenido del archivo se cargará en una nueva área de texto en el editor.

3.3. Guardar Archivos

1. Haz clic en el botón "Guardar Archivo" para descargar el contenido del área de texto como un archivo `.oak` en tu computadora.

3.4. Edición de Código

- Puedes escribir, editar y modificar código en las áreas de texto del editor.

- El editor soporta múltiples archivos abiertos a la vez, permitiéndote trabajar en diferentes partes de tu proyecto simultáneamente.

4**. Ejecución de Código**

4.1. Ejecutar Código

1. Una vez que hayas escrito o cargado tu código en el editor, haz clic en el botón "Ejecutar".

2. El intérprete procesará tu código, ejecutando las instrucciones y mostrando los resultados en la consola.

4.2. Ver Resultados en la Consola

- La consola mostrará los resultados de la ejecución, incluyendo cualquier salida generada por instrucciones como `System.out.println`.

- Cualquier error léxico, sintáctico o semántico también se mostrará en la consola.

4.3. Manejo de Errores

- Si tu código contiene errores, estos se mostrarán en una tercera consola de Errores con detalles específicos del problema, incluyendo la línea y columna donde ocurrió el error.

5. Funcionalidades Avanzadas

5.1. Reporte de Tabla de Símbolos

1. Haz clic en el botón "Generar Reporte" para crear un informe detallado de todas las variables, funciones y métodos declarados en tu código.

2. El reporte aparecerá en una consola debajo de la consola principal, proporcionando detalles sobre cada símbolo en el programa, incluyendo su entorno, tipo y valor.

5.2. Depuración Básica

- Break y Continue: Puedes utilizar las sentencias `break` y `continue` para controlar el flujo dentro de los bucles. Estos comandos se manejarán correctamente, permitiendo una ejecución controlada.

- Return: Usa la sentencia `return` para devolver un valor desde una función, lo que se reflejará correctamente en la ejecución del programa.

6. Uso de Funciones Embebidas

6.1. `System.out.println`

- Imprime texto o variables en la consola.

- Ejemplo de uso:

```
oak
int a = 5;
System.out.println("El valor de a es ", a);
```

6.2. `parseInt` y `parseFloat`

- parseInt: Convierte una cadena a un entero.

- parseFloat: Convierte una cadena a un número de punto flotante.

- Ejemplo de uso:

```
oak
int numero = parseInt("123");
float decimal = parseFloat("3.14");
```

6.3. `toString

- Convierte cualquier tipo de dato a su representación en cadena.

- Ejemplo de uso:

```
oak
int numero = 10;
string texto = toString(numero);
```

6.4. `toLowerCase` y `toUpperCase`

- toLowerCase: Convierte una cadena a minúsculas.
- toUpperCase: Convierte una cadena a mayúsculas.
- Ejemplo de uso:

```
oak
string texto = "Hola Mundo";
string minusculas = toLowerCase(texto);
string mayusculas = toUpperCase(texto);
```

6.5. `typeof`

- Devuelve el tipo de dato de una variable.
- Ejemplo de uso:

```
oak
int numero = 10;
string tipo = typeof(numero); // Devuelve "int"
```

7. Ejemplos de Código

7.1. Bucle `For`

```
for (int i = 0; i < 10; i++) {
  System.out.println("Iteración: ", i);
}
```

7.2. Bucle `While`

```
int contador = 0;
while (contador < 5) {
  System.out.println("Contador: ", contador);
  contador++;
}
```

7.3. Uso de Funciones oak

```
int suma(int a, int b) {
  return a + b;
}

int resultado = suma(3, 4);
System.out.println("Resultado de la suma: ", resultado);
```

7.4. Manejo de `If-Else`

oak

```
int numero = 10;

if (numero > 5) {

  System.out.println("El número es mayor que 5");

} else {

  System.out.println("El número es 5 o menor");

}
```

8. Soporte y Ayuda

Si encuentras problemas o necesitas asistencia adicional, puedes acceder a la documentación completa del proyecto en el repositorio de GitHub. Además, se pueden reportar problemas o solicitar características nuevas a través de los Issues en GitHub.

9. Conclusión

El IDE OakLand es una herramienta poderosa para escribir, depurar y ejecutar código en el lenguaje OakLand. Con este manual, deberías tener una base sólida para comenzar a explorar y desarrollar proyectos utilizando este lenguaje. ¡Disfruta programando en OakLand!