Manual Técnico

NodeLex v2.0

Descripción General

NodeLex v2.0 es una herramienta de análisis léxico y sintáctico con capacidades de cálculo matemático, diseñada para procesar un lenguaje exclusivo definido en archivos con extensión .nlex. Este proyecto incluye una interfaz web moderna basada en React, junto con una implementación robusta de los analizadores léxicos y sintácticos.

Componentes Principales

1. Analizador Léxico

El analizador léxico (Lexer) procesa el código fuente, identifica tokens y reporta errores léxicos.

Características:

- Reconoce tokens como palabras clave, identificadores, operadores, números, cadenas y símbolos.
- Soporte para comentarios de una línea (//) y multilínea (/* */).
- Mecanismo de manejo de errores para detectar símbolos inválidos o cadenas sin cerrar.

Métodos Clave:

- **advance()**: Avanza al siguiente carácter del código fuente.
- **tokenize()**: Itera por el código y genera una lista de tokens.
- **lexNumber(), lexWord(), lexString(), lexSymbol()**: Métodos especializados para procesar diferentes tipos de tokens.

Entrada:

Código fuente en formato .nlex.

Salida:

- Lista de tokens válidos.
- Lista de errores léxicos.

---

2. Analizador Sintáctico

El analizador sintáctico (SyntacticAnalyzer) construye un Árbol de Sintaxis Abstracta (AST) y reporta errores de sintaxis.

Características:

- Implementa una gramática libre de contexto para validar estructuras del lenguaje.
- Construcción del AST para operaciones, configuraciones y funciones.

Métodos Clave:

- **analizar()**: Procesa los tokens y construye el AST.
- **analizarOperaciones()**: Analiza bloques de operaciones.
- **analizarConfiguracion()**: Valida configuraciones de Lex y Parser.
- **analizarFuncion()**: Procesa funciones como imprimir o promedio.
- **registrarError()**: Captura errores de sintaxis.

Entrada:

Lista de tokens generados por el Lexer.

Salida:

- AST.
- Lista de errores sintácticos.

---

3. Resolución de Operaciones

El módulo OperationResolver evalúa operaciones matemáticas definidas en el AST.

Características:

- Soporte para operaciones matemáticas estándar (suma, resta, multiplicación, etc.) y funciones trigonométricas.
- Resolución de operaciones anidadas.
- Generación de diagramas en formato Graphviz.

Métodos Clave:

- **resolve()**: Procesa el AST y calcula resultados.
- **generateGraphvizDiagram()**: Genera un archivo DOT para visualizar las operaciones.

Entrada:

AST generado por el analizador sintáctico.

Salida:

- Resultados de las operaciones.
- Archivo DOT con el diagrama de operaciones.

---

4. Interfaz Web

La interfaz está construida con React para ofrecer una experiencia interactiva y fácil de usar.

Características:

- Editor de texto para escribir o cargar archivos .nlex.
- Botones para análisis, resolución, y exportación de archivos.
- Visualización de resultados, tokens, AST, errores y diagramas.

Funciones Clave:

- **Carga de archivos**: Permite cargar archivos desde el sistema.
- **Exportación**: Guarda el código analizado como .nlex.
- **Generación de diagramas**: Descarga automáticamente el archivo DOT.

Librerías Utilizadas:

- **React**: Framework principal.
- **Viz.js**: Generación de diagramas SVG a partir de archivos DOT.

Instalación y Configuración

Requisitos Previos:

- Node.js (v16 o superior).
- Un navegador moderno (Chrome, Firefox, etc.).

Pasos de Instalación:

1. Clonar el repositorio:
2. git clone <repositorio>
3. Instalar dependencias:
4. npm install
5. Iniciar la aplicación:
6. npm start
7. Acceder a la interfaz en http://localhost:3000.

Uso de NodeLex

```
ConfiguracionesLex = [fondo: "#FFFFFF", fuente: "#000000"]
ConfiguracionesParser = [forma: "ellipse", tipoFuente: "Arial"]
Operaciones = [
  { nombre: "op1", operacion: "suma", valor1: 5, valor2: 10 },
  { operacion: "potencia", valor1: 2, valor2: 3 }
]
```

Archivo de Entrada

Formato .nlex:

ConfiguracionesLex = [fondo: "#FFFFFF", fuente: "#000000"]

ConfiguracionesParser = [forma: "ellipse", tipoFuente: "Arial"]

Operaciones = [

{ nombre: "op1", operacion: "suma", valor1: 5, valor2: 10 },

{ operacion: "potencia", valor1: 2, valor2: 3 }

]

Análisis Léxico y Sintáctico

1. Escribir o cargar un archivo .nlex en el editor.

2. Presionar el botón **Analizar**.
3. Visualizar tokens, AST y errores en sus respectivas secciones.

Resolución de Operaciones

1. Después del análisis, presionar **Resolver Operaciones**.
2. Consultar los resultados y diagramas generados.

Exportación

- Guardar el archivo actual: **Guardar**.
- Guardar como un nuevo archivo: **Guardar como**.

Generación de Reportes

1. **Tabla de Tokens**: Generada automáticamente al analizar.
2. **Errores Léxicos y Sintácticos**: Visualizados en secciones separadas.
3. **Diagrama de Operaciones**: Generado en formato DOT y SVG.

Diagramas

Generación de Diagramas

El método generateGraphvizDiagram de OperationResolver genera diagramas DOT que se convierten a SVG mediante Viz.js. Los nodos y enlaces están personalizados según configuraciones del archivo .nlex.

Estructura del Proyecto

```
├── src
│   ├── components
│   │   ├── App.js
│   │   ├── Lexer.js
│   │   ├── SyntacticAnalyzer.js
```

```
│   │   └── OperationResolver.js
│   └── styles
│       └── App.css
├── public
│   ├── index.html
└── package.json
```

Consideraciones Finales

- **Requisitos obligatorios:** Uso de React, generación de diagramas con Graphviz.
- **Entregables:** Repositorio privado con documentación técnica y de usuario.
- **Calidad:** Validación robusta de errores léxicos y sintácticos.
