EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE MORELIA
"José María Morelos y Pavón"

INSTITUTO TECNOLÓGICO DE MORELIA

DIVISIÓN DE ESTUDIOS PROFESIONALES

Lenguajes y Autómatas II

Proyecto Final

QUE PARA OBTENER LA CALIFICACIÓN DE

Proyecto Final

PRESENTA:

EDGAR TAPIA MARTINEZ

LUIS FERNANDO CHAVEZ MARTINEZ

MARIO EDUARDO SANCHEZ MEJIA

DOCENTE: ALMA LILIA NUÑEZ GONZALES

MORELIA, MICHOACÁN

TranslatorApp: Intérprete y Traductor de ALMA

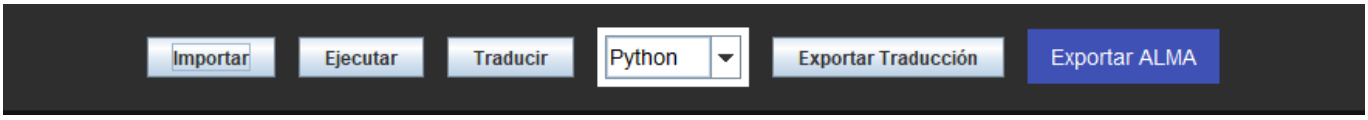
Descripción General

Este código implementa una aplicación de traducción e interpretación para un lenguaje personalizado denominado **ALMA**. La aplicación ofrece una interfaz gráfica moderna que permite a los usuarios editar código ALMA, ejecutarlo en tiempo real, y traducirlo a diversos lenguajes de programación como Python, Java y Assembly.

Arquitectura de la Interfaz

La interfaz gráfica está diseñada con un enfoque moderno y funcional:

Panel Superior



- **Barra de Control** — Alberga los controles principales de la aplicación:
  - `Importar` — Carga archivos ALMA (.alma)
  - `Ejecutar` — Interpreta y ejecuta el código ALMA actual
  - `Traducir` — Convierte el código al lenguaje seleccionado
  - `Selector de Lenguaje` — Permite elegir el lenguaje destino
  - `Exportar Traducción` — Guarda el código traducido
  - `Exportar ALMA` — Guarda el código original

Panel Principal

- **División Horizontal** — Separa la interfaz en dos secciones principales:
  - **Editor (Izquierda)** — Área para editar código ALMA
  - **Panel de Resultados (Derecha)** — Organizado verticalmente:
    - *Área de Salida* — Muestra el código traducido
    - *Consola* — Presenta mensajes del sistema y resultados de ejecución

---

Componentes Funcionales

1. Sistema de Edición

El editor está optimizado para trabajar con código ALMA, con un diseño visual oscuro que reduce la fatiga visual durante sesiones prolongadas de programación.

2. Motor de Procesamiento

La aplicación procesa el código ALMA mediante una secuencia sofisticada:

- **Análisis Léxico → Análisis Sintáctico → Interpretación → Traducción**

3. Funciones Principales

- `importCode()`
  - Facilita la selección e importación de archivos ALMA
  - Actualiza el editor y notifica al usuario
- `executeCode()`
  - Transforma el código en tokens mediante `LanguageLexer`
  - Genera un árbol sintáctico con `LanguageParser`
  - Procesa el árbol utilizando `LanguageCustomVisitor`
  - Ejecuta secuencialmente cada instrucción generada
  - Muestra los resultados en la consola
- `translateCode()`
  - Sigue un proceso similar al de ejecución
  - Utiliza `CodeTranslator` para convertir las estructuras a otro lenguaje
  - Despliega el resultado en el área de salida
- `exportCode()` **y** `exportOriginalCode()`
  - Permiten guardar tanto el código traducido como el original
  - Aseguran el uso de extensiones de archivo apropiadas

Aspectos Técnicos Destacados

- **Framework ANTLR4** — Proporciona capacidades robustas de análisis léxico y sintáctico
- **Patrón Visitor** — Facilita el recorrido y procesamiento del árbol sintáctico

- **Java Swing** — Ofrece componentes de interfaz gráfica versátiles
- **Diseño Modular** — Separa claramente las responsabilidades entre interfaz, análisis y traducción
- **Manejo de Excepciones** — Garantiza respuestas adecuadas ante errores de sintaxis o ejecución

---

Flujo de Trabajo del Usuario

1. El usuario **escribe o importa** código ALMA
2. **Ejecuta** el código para verificar su comportamiento
3. **Traduce** el código al lenguaje deseado
4. Examina la traducción y **exporta** los resultados

La aplicación proporciona retroalimentación constante a través de la consola, informando sobre el éxito o fracaso de cada operación realizada.