Documentación del Compilador Python to x86_64

Índice de Documentación

Este archivo contiene la documentación completa del proyecto **Compilador Python a x86_64**. Los documentos están organizados para proporcionar información desde conceptos básicos hasta detalles técnicos avanzados.

📋 Documentos Disponibles

Documento	Descripción	Audiencia
architecture.md	Arquitectura general del sistema, componentes principales y patrones de diseño	Desarrolladores, Arquitectos
user-guide.md	Manual de instalación, uso y sintaxis soportada	Usuarios finales, Estudiantes
grammar-specification.md	Especificación completa de la gramática ANTLR, tokens y reglas de producción	Desarrolladores de compiladores
development.md	Guía para contribuir al proyecto, extender funcionalidad y convenciones de código	Contribuidores, Desarrolladores
examples.md	Ejemplos de código, casos de uso y limitaciones actuales	Usuarios, Testers

🎯 Guía de Lectura Recomendada

Para Usuarios Nuevos:

1. Comenzar con **user-guide.md** para instalación y uso básico
2. Revisar **examples.md** para ver casos de uso prácticos
3. Consultar **architecture.md** si desea entender el funcionamiento interno

Para Desarrolladores:

1. Leer **architecture.md** para comprender el diseño
2. Estudiar **grammar-specification.md** para entender la gramática
3. Seguir **development.md** para contribuir al proyecto
4. Usar **examples.md** como referencia para testing

Para Estudiantes de Compiladores:

1. Iniciar con **grammar-specification.md** para comprender ANTLR
2. Continuar con **architecture.md** para el patrón AST/Visitor
3. Experimentar con **examples.md** para casos prácticos
4. Explorar **development.md** para extensiones

📖 Resumen del Proyecto

El **Compilador Python to x86_64** es una implementación educativa que transforma un subconjunto del lenguaje Python en código ensamblador x86_64. El proyecto demuestra los conceptos fundamentales de construcción de compiladores utilizando tecnologías modernas.

Características Principales:

- ☑ **Sintaxis Python**: Soporte completo para indentación, variables, expresiones
- ☑ **Estructuras de Control**: Ciclos `for` y `while`, condicionales `if/elif/else`
- ☑ **Tipos de Datos**: Enteros, strings literales y booleanos
- ☑ **Expresiones**: Aritméticas (+, -, *, /, %, **), lógicas (and, or, not) y comparaciones (==, !=, <, >, <=, >=)
- ☑ **Operadores Unarios**: Negación aritmética y lógica (-, not, +)
- ☑ **Generación ASM**: Código x86_64 optimizado para Linux con gestión de strings

Stack Tecnológico:

- **ANTLR 4.13.2**: Generación de parsers
- **Java 8+**: Lenguaje de implementación
- **x86_64 Assembly**: Target de compilación
- **Linux ABI**: Convenciones de llamadas del sistema

🗀 Estructura de Documentación

```
documentation/
├── README.md                    # Este archivo - índice general
├── architecture.md              # Diseño y componentes del sistema
├── user-guide.md               # Manual de usuario e instalación
├── grammar-specification.md     # Especificación completa de gramática ANTLR
├── development.md              # Guía de desarrollo y contribución
└── examples.md                 # Ejemplos de código y casos de uso
```

🚀 Quick Start

Si desea empezar rápidamente:

1. **Instalación Básica**: Ver user-guide.md#instalación
2. **Primer Ejemplo**: Consultar examples.md#básicos
3. **Compilar Código**: Seguir user-guide.md#uso

🔧 Para Contribuidores

Si desea contribuir al proyecto:

1. **Setup de Desarrollo**: development.md#setup
2. **Agregar Features**: development.md#extensiones
3. **Testing**: development.md#testing
4. **Code Style**: development.md#convenciones

Referencias Adicionales

Documentación Externa:

- ANTLR 4 Documentation
- x86_64 Assembly Reference
- System V ABI

Recursos Académicos:

- **Compiladores**: "Engineering a Compiler" por Cooper & Torczon
- **ANTLR**: "The Definitive ANTLR 4 Reference" por Terence Parr
- **Assembly**: "Programming from the Ground Up" por Jonathan Bartlett

Reporte de Issues

Para reportar problemas o sugerir mejoras:

1. **Bugs**: Seguir template en development.md#bug-reports
2. **Feature Requests**: Crear issue con etiqueta "enhancement"
3. **Preguntas**: Usar etiqueta "question" en issues

Licencia y Uso

Este proyecto está desarrollado con fines educativos. Consulte el archivo `LICENSE` en el directorio raíz para información sobre términos de uso.

Estado del Proyecto

Componente	Estado	Notas
Lexer/Parser	☑ Completo	ANTLR 4.13.2
AST	☑ Completo	Patrón Visitor
CodeGen	☑ Completo	x86_64 básico
Testing	☑ Funcional	Casos de prueba
Documentation	☑ Completo	Esta documentación
If/Elif/Else	☑ Completo	Condicionales completos
While Loops	☑ Completo	Ciclos while
For Loops	☑ Completo	Ciclos for con range()
Logical Ops	☑ Completo	and, or, not
Features Pendientes		
Functions	⏳ Pendiente	Funciones definidas por usuario
Arrays/Lists	⏳ Pendiente	Estructuras de datos

Componente	Estado	Notas
Range Args	⏳ Pendiente	range(start, stop, step) completo

🔄 Última Actualización

Versión: 2.0 **Fecha**: Noviembre 2025 **Autores**:

- **Melo Reséndiz Jorge**
- **Guerrero Serrano Jafeth Oswaldo**
- **Paniagua Rico Juan Julián**

Estado: Funcional y documentado con características completas de control de flujo

Para información sobre versiones y changelog, ver el historial de commits en el repositorio principal.

💡 **Tip**: Mantenga esta documentación actualizada conforme evolucione el proyecto. Cada nueva característica debe incluir ejemplos en `examples.md` y consideraciones de desarrollo en `development.md`.