

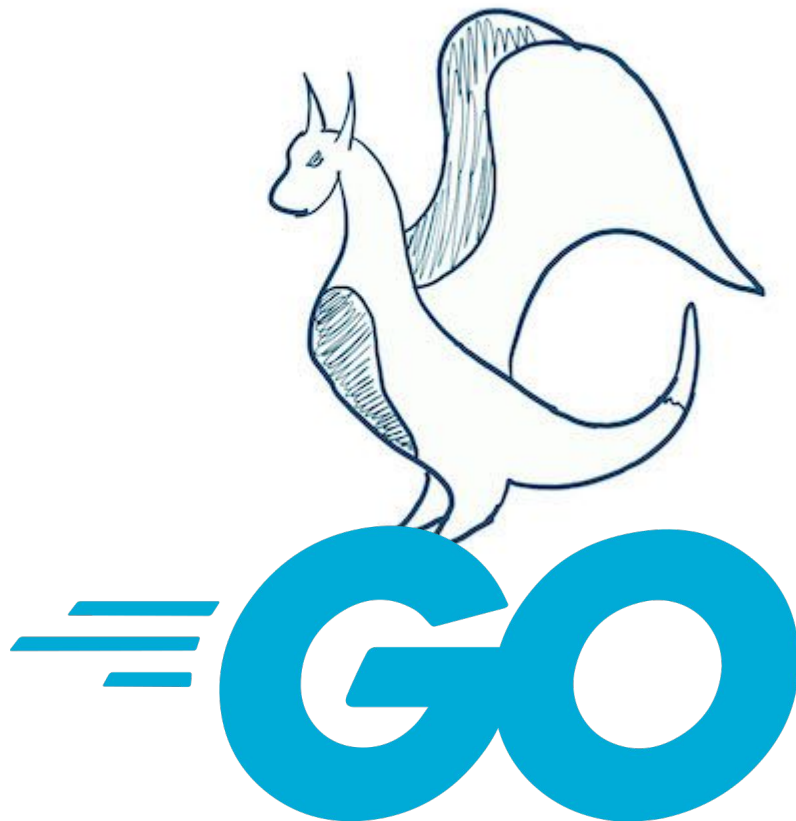
# Compilador para Go

Juan Lara & Santiago Jimenez

# Introducción

- Un lenguaje de programación de código abierto apoyado por Google.
- Concurrente, compilado, estructurado, orientado a objetos y con tipado estático.
- Inspirado en C (i386, amd64 y ARM)
- Seguridad de memoria y recolección de basura.
- Creado en **2009** por:

*Robert Griesemer, Rob Pike y Ken Thompson*



## Companies using Go

Organizations in every industry use Go to power their software and services [View all stories](#)



¿Todo



eso?



No

# The Go Programming Language Specification

Version of June 29, 2022

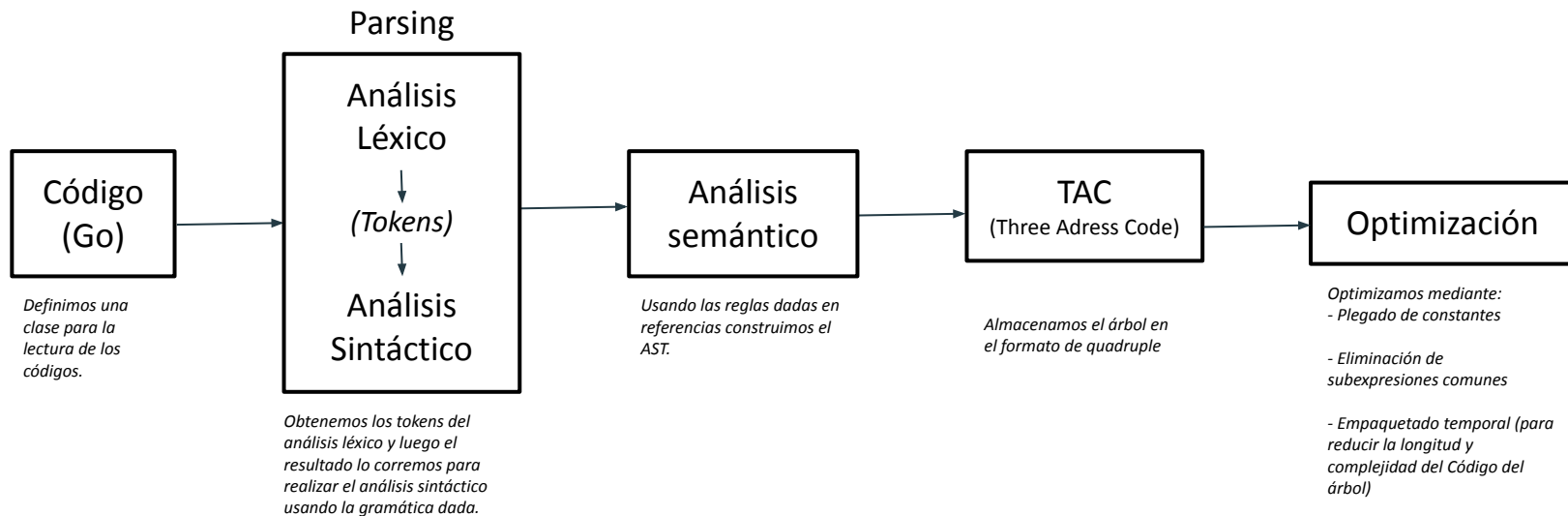
## Table of Contents

Introduction	Index expressions
Notation	Slice expressions
Source code representation	Type assertions
Characters	Calls
Letters and digits	Passing arguments to ... parameters
Lexical elements	Instantiations
Comments	Type inference
Tokens	Operators
Semicolons	Arithmetic operators
Identifiers	Comparison operators
Keywords	Logical operators
Operators and punctuation	Address operators
Integer literals	Receive operator
Floating-point literals	Conversions
Imaginary literals	Constant expressions
Rune literals	Order of evaluation
String literals	Statements

Constants	Terminating statements
Variables	Empty statements
Types	Labeled statements
Boolean types	Expression statements
Numeric types	Send statements
String types	IncDec statements
Array types	Assignment statements
Slice types	If statements
Struct types	Switch statements
Pointer types	For statements
Function types	Go statements
Interface types	Select statements
Map types	Return statements
Channel types	Break statements
Properties of types and values	Continue statements
Underlying types	Goto statements
Core types	Fallthrough statements
Type identity	Defer statements
Assignability	Built-in functions
Representability	Close
Method sets	Length and capacity
Blocks	Allocation
Declarations and scope	Making slices, maps and channels

Label scopes	Appending to and copying slices
Blank identifier	Deletion of map elements
Predeclared identifiers	Manipulating complex numbers
Exported identifiers	Handling panics
Uniqueness of identifiers	Bootstrapping
Constant declarations	Packages
Iota	Source file organization
Type declarations	Package clause
Type parameter declarations	Import declarations
Variable declarations	An example package
Short variable declarations	Program initialization and execution
Function declarations	The zero value
Method declarations	Package initialization
Expressions	Program execution
Operands	Errors
Qualified identifiers	Run-time panics
Composite literals	System considerations
Function literals	Package unsafe
Primary expressions	Size and alignment guarantees
Selectors	
Method expressions	
Method values	

# Diagrama de solución

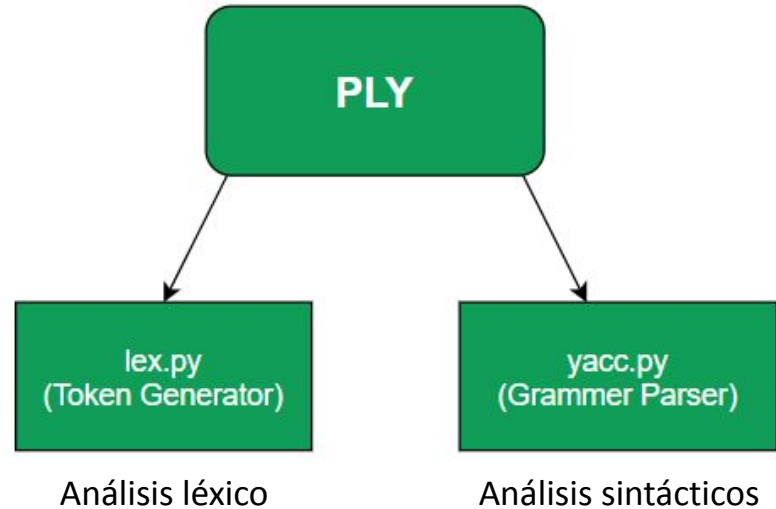


*Basado en referencias en las cuales se define la gramática.*

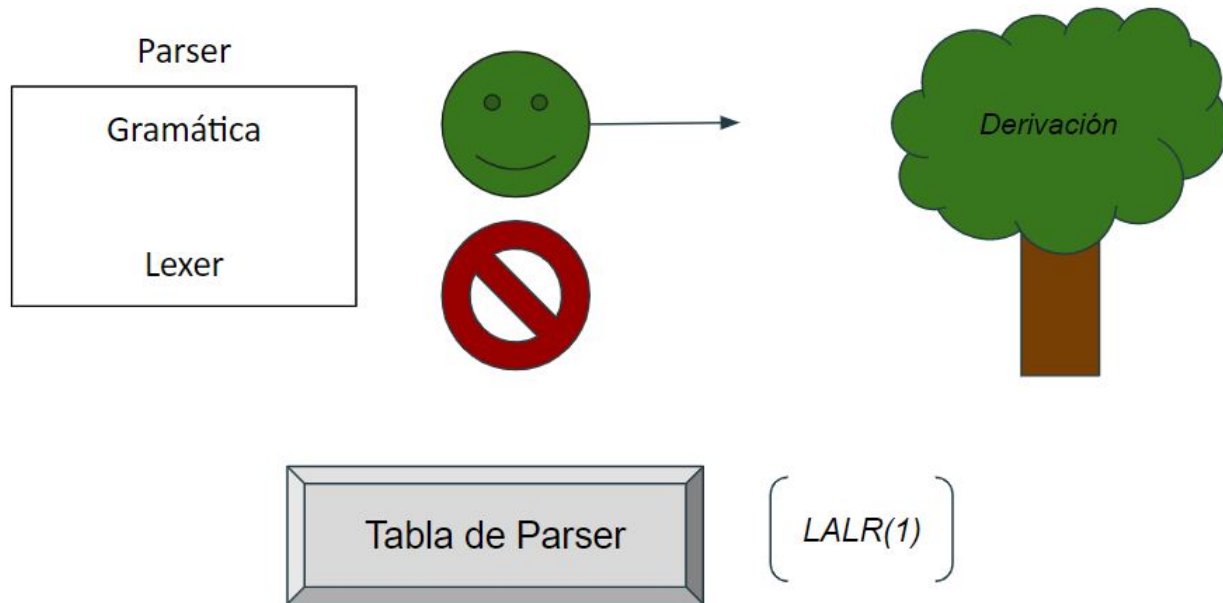


# Para ello usamos

- Referencias:  
Go (CFG, reglas que definen el lenguaje).
- **PLY (Python Lex-Yacc)**



# Acerca del parser



# Ejemplo

.Go

```
package main

import "fmt"

func main() {
    sum := 243
    for sum < 1000 {
        sum += sum
    }
}
```



```
['func', 'main', '', '']
[':=', 'sum', '243', '']
['label_0']
['<', 'temp_0', 'sum', '1000']
['ifgotoeq', 'temp_0', 0, 'label_1']
['+=', 'sum', 'sum', '']
['goto', 'label_0']
['label_1']
```

## OPTIMIZATION

```
['func', 'main', '', '']
['label_0']
['<', 'temp_0', '243', '1000']
['ifgotoeq', 'temp_0', 0, 'label_1']
['goto', 'label_0']
['label_1']
```

## SYMBOL TABLE

NAME	TYPE	SCOPE	VALUE	EXPR
sum	INT	1	243	None
temp_0	INT	1	1	sum<1000

```
package main
```

```
import "fmt"
```

```
func main() {
    sum := 243
    for sum < 1000 {
        sum += sum
    }
}
```

# Referencias

1. Aho, A. V., & Aho, A. V. (Eds.). (2007). Compilers: Principles, techniques, & tools (2nd ed). Pearson/Addison Wesley.
2. Aho, A. V., Sethi, R., & Ullman, J. D. (1986). Compilers, principles, techniques, and tools. Addison-Wesley Pub. Co.
3. Codewalk: First-class functions in go - the go programming language. (n.d.). Retrieved 3 December 2022, from <https://go.dev/doc/codewalk/functions/>
4. Go—Operators precedence. (n.d.). Retrieved 3 December 2022, from [https://www.tutorialspoint.com/go/go\\_operators\\_precedence.htm](https://www.tutorialspoint.com/go/go_operators_precedence.htm)
5. PLY (Python lex-yacc)—Ply 4.0 documentation. (n.d.). Retrieved 3 December 2022, from <https://ply.readthedocs.io/en/latest/>
6. The go programming language specification—The go programming language. (n.d.). Retrieved 3 December 2022, from <https://go.dev/ref/spec>
7. Three address code in Compiler. (2018, May 21). GeeksforGeeks. <https://www.geeksforgeeks.org/three-address-code-compiler/>