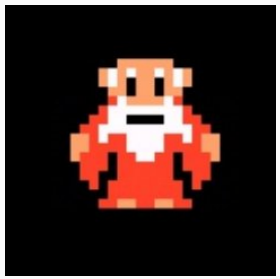
Golang and Domain Specific Languages

Lorenzo Fontana

March 24, 2017

About Me



Lorenzo Fontana

DevOps Expert @Kiratech

Docker Maintainer

`http://fntlnz.wtf`

`https://github.com/fntlnz`

`https://twitter.com/fntlnz`

Background on DSLs

Background on DSLs: Examples

- HTML

Background on DSLs: Examples

- HTML
- SQL

Background on DSLs: Examples

- HTML
- SQL
- GraphQL

Background on DSLs: Examples

- HTML
- SQL
- GraphQL
- Protobuf

Background on DSLs: Examples

- HTML
- SQL
- GraphQL
- Protobuf
- Regex

Background on DSLs: Examples

- HTML
- SQL
- GraphQL
- Protobuf
- Regex
- Jenkinsfile

Background on DSLs: Examples

- HTML
- SQL
- GraphQL
- Protobuf
- Regex
- Jenkinsfile
- Dockerfile

Background on DSLs: Examples

- HTML
- SQL
- GraphQL
- Protobuf
- Regex
- Jenkinsfile
- Dockerfile
- Make

Background on DSLs: Examples

- HTML
- SQL
- GraphQL
- Protobuf
- Regex
- Jenkinsfile
- Dockerfile
- Make
- CSS

Background on DSLs: Examples

- HTML
- SQL
- GraphQL
- Protobuf
- Regex
- Jenkinsfile
- Dockerfile
- Make
- CSS
- TeX

Background on DSLs: Examples

- HTML
- SQL
- GraphQL
- Protobuf
- Regex
- Jenkinsfile
- Dockerfile
- Make
- CSS
- TeX
- And a lot more

Background on DSLs: Examples

- HTML
- SQL
- GraphQL
- Protobuf
- Regex
- Jenkinsfile
- Dockerfile
- Make
- CSS
- TeX
- And a lot more, really

Background on DSLs: Examples

- HTML
- SQL
- GraphQL
- Protobuf
- Regex
- Jenkinsfile
- Dockerfile
- Make
- CSS
- TeX
- And a lot more, really, **trust me**

Background on DSLs: Terminology

Internal DSL : a DSL which exposes a particular form of host general purpose language to fit domain specific needs, for their nature, this kind of DSLs are easier to implement but limited by the design of the host language.

External DSL : a DSL which is parsed independently of the host general purpose language.

Background on DSLs: External or internal ?

Example from: Ginkgo, BDD testing framework for Go

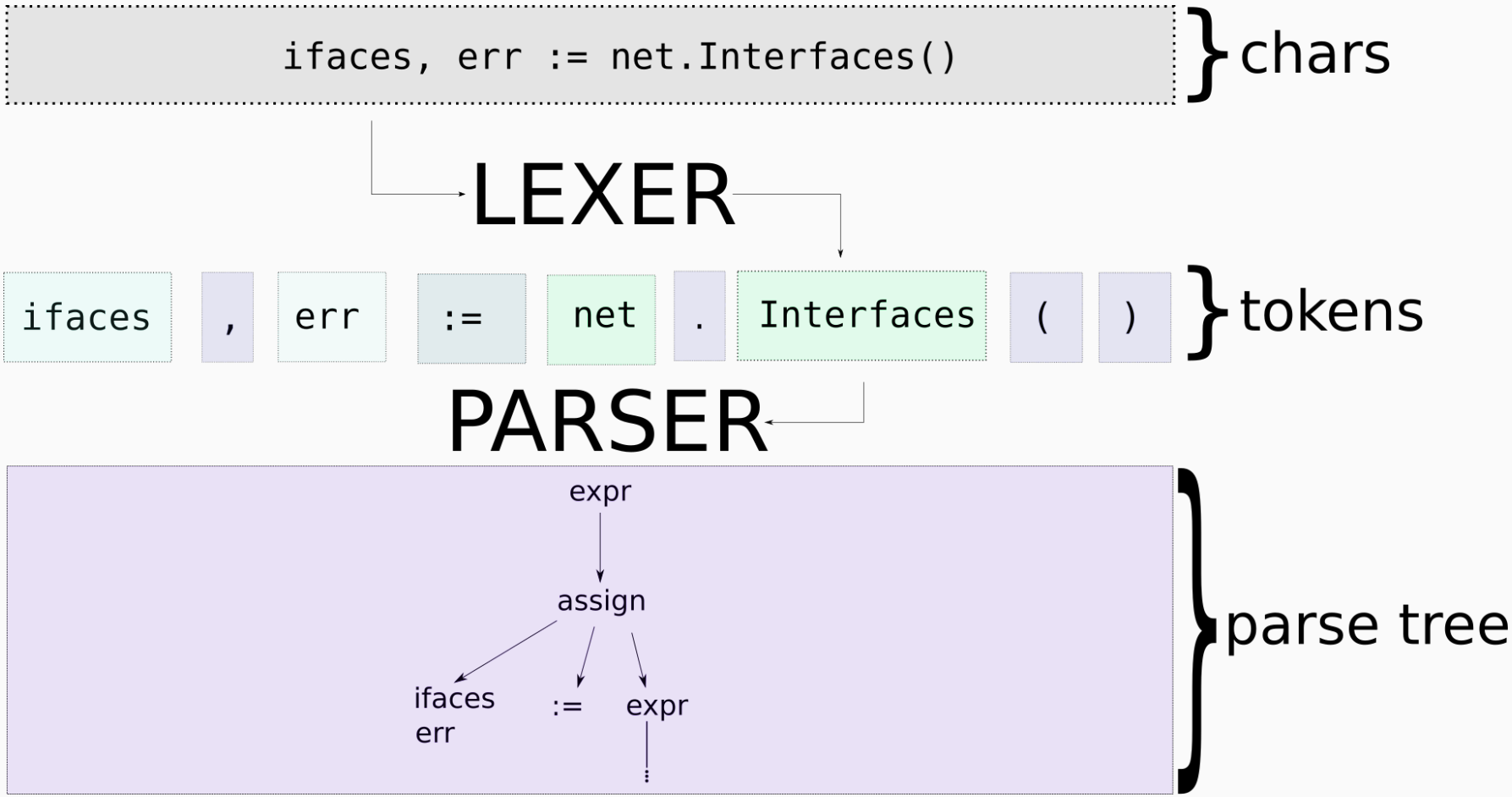
```
It("should fail in all the possible ways", func() {
        session := startGinkgo(pathToTest, "--noColor")
        Eventually(session).Should(gexec.Exit(1))
        output := string(session.Out.Contents())
          (output).ShouldNot(
                ContainSubstring("NEVER SEE THIS")
        )
}
```

Background on DSLs: External or internal?

Makefile

```
.PHONY: check-gopath

check-gopath:
ifndef GOPATH
        $(error GOPATH is not set)
endif

lint: check-gopath
        @echo "checking lint"
        @./.tool/lint

gofmt:
        @./hack/verify-gofmt.sh

conmon:
        $(MAKE) -C $@
```

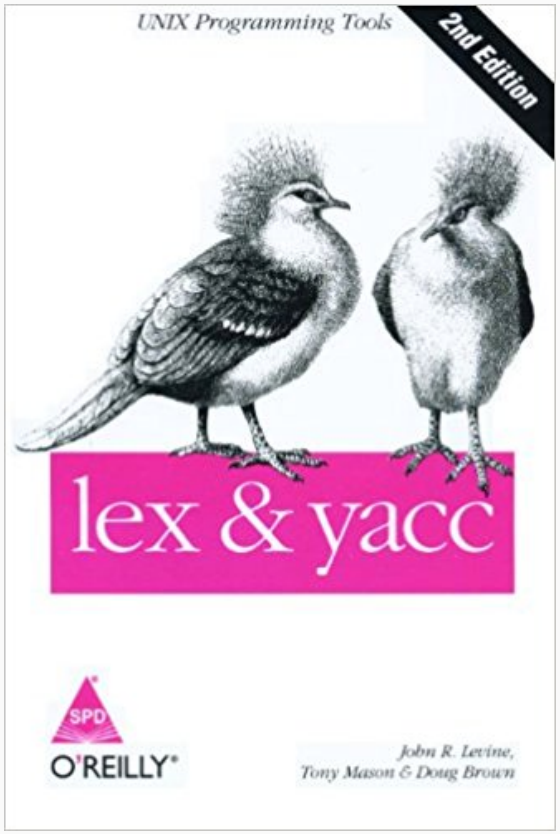
Background on DSLs: Lexical analyzers and parser generators

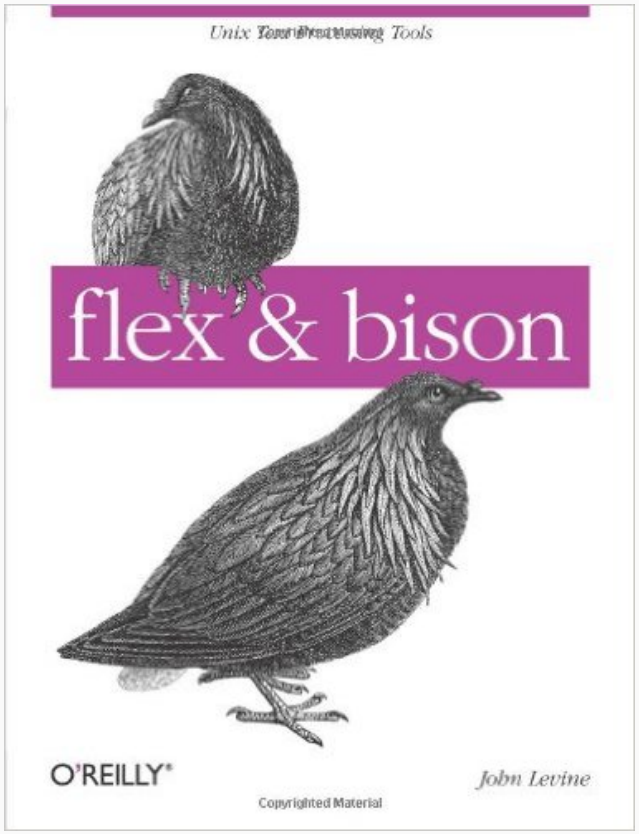


Background on DSLs: Backus-Naur Form (BNF)

```
<expr> ::= <term> "+" <expr>
         |  <term>

<term> ::= <factor> "*" <term>
         |  <factor>

<factor> ::= "(" <expr> ")"
           |  <const>

<const> ::= integer
```

Background on DSLs: lex and yacc - flex and bison





What about Go?

Go: goyacc

GoDoc Home About Search

tools: golang.org/x/tools/cmd/goyacc Files

Command goyacc

Goyacc is a version of yacc for Go. It is written in Go and generates parsers written in Go.

Usage:

```
goyacc args...
```

It is largely transliterated from the Inferno version written in Limbo which in turn was largely transliterated from the Plan 9 version written in C and documented at

```
https://9p.io/magic/man2html/1/yacc
```

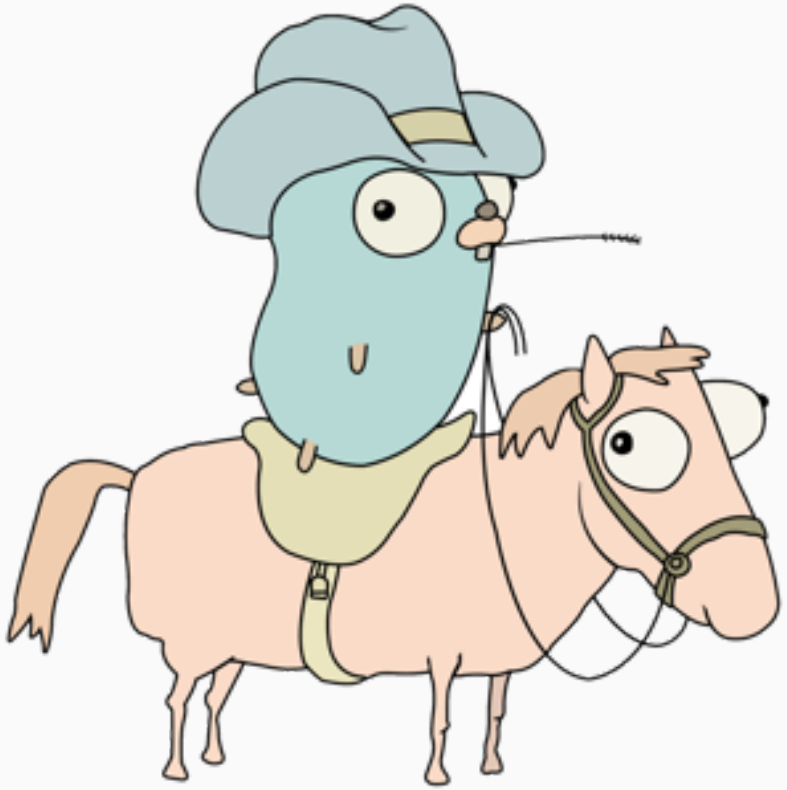
Adepts of the original yacc will have no trouble adapting to this form of the tool.

The directory $GOPATH/src/golang.org/x/tools/cmd/goyacc/testdata/expr is a yacc program for a very simple expression parser. See expr.y and main.go in that directory for examples of how to write and build goyacc programs.

The generated parser is reentrant. The parsing function yyParse expects to be given an argument that conforms to the following interface:

```
type yyLexer interface {
        Lex(lval *yySymType) int
        Error(e string)
}
```

Go: SHOW US THE CODE!



Who am I to not **put a gopher** in my slides?

Conclusion

Conclusion: Happy promotion!

Kiratech, the company I work for, is **hiring!**

drop me a line at lo@linux.com

Questions?

Thanks for listening!

And thanks to all the organizers!