Dann schreibe ich mal eben eine DSL

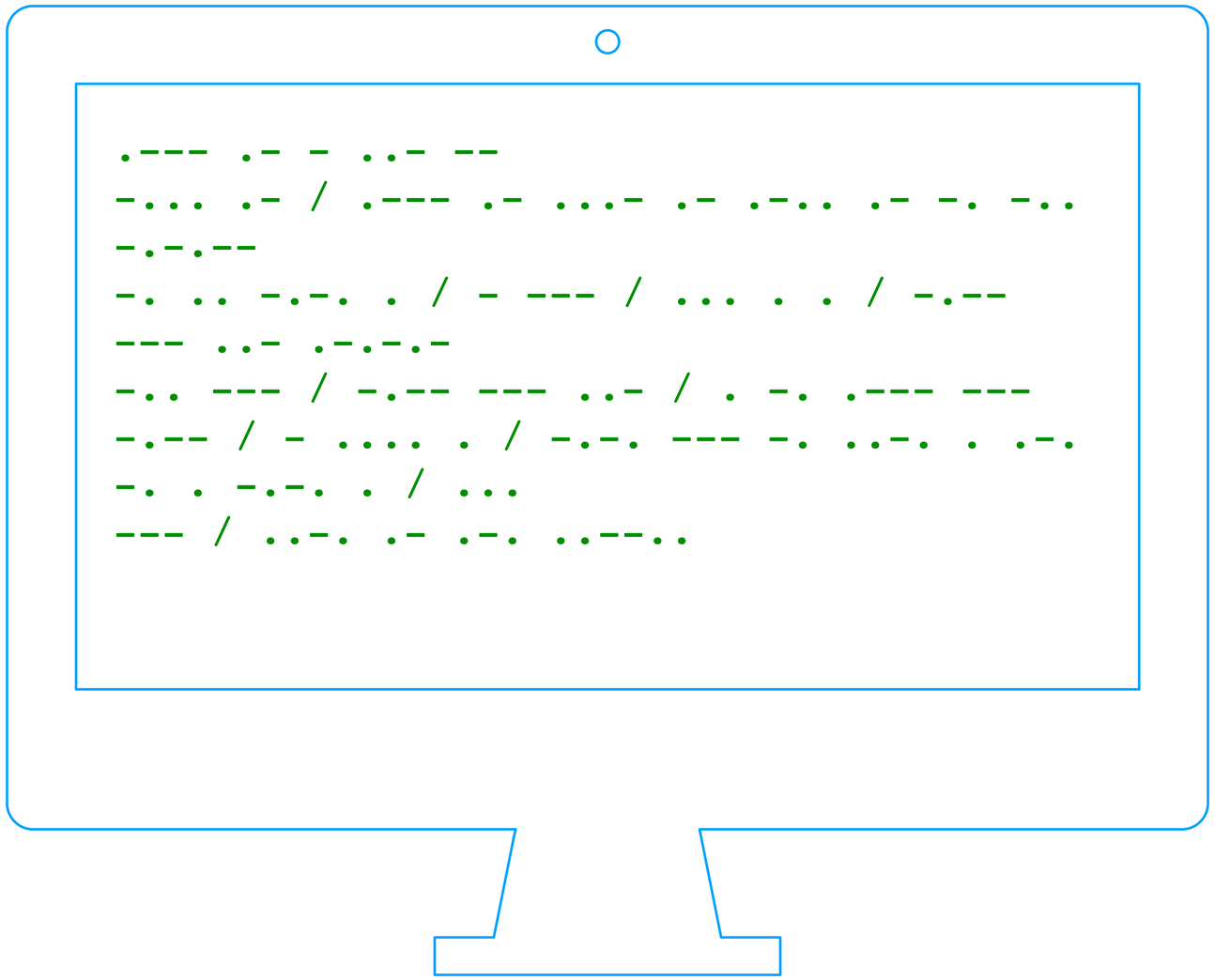
Javaland 2025, 03.04.202, Birgit Kratz

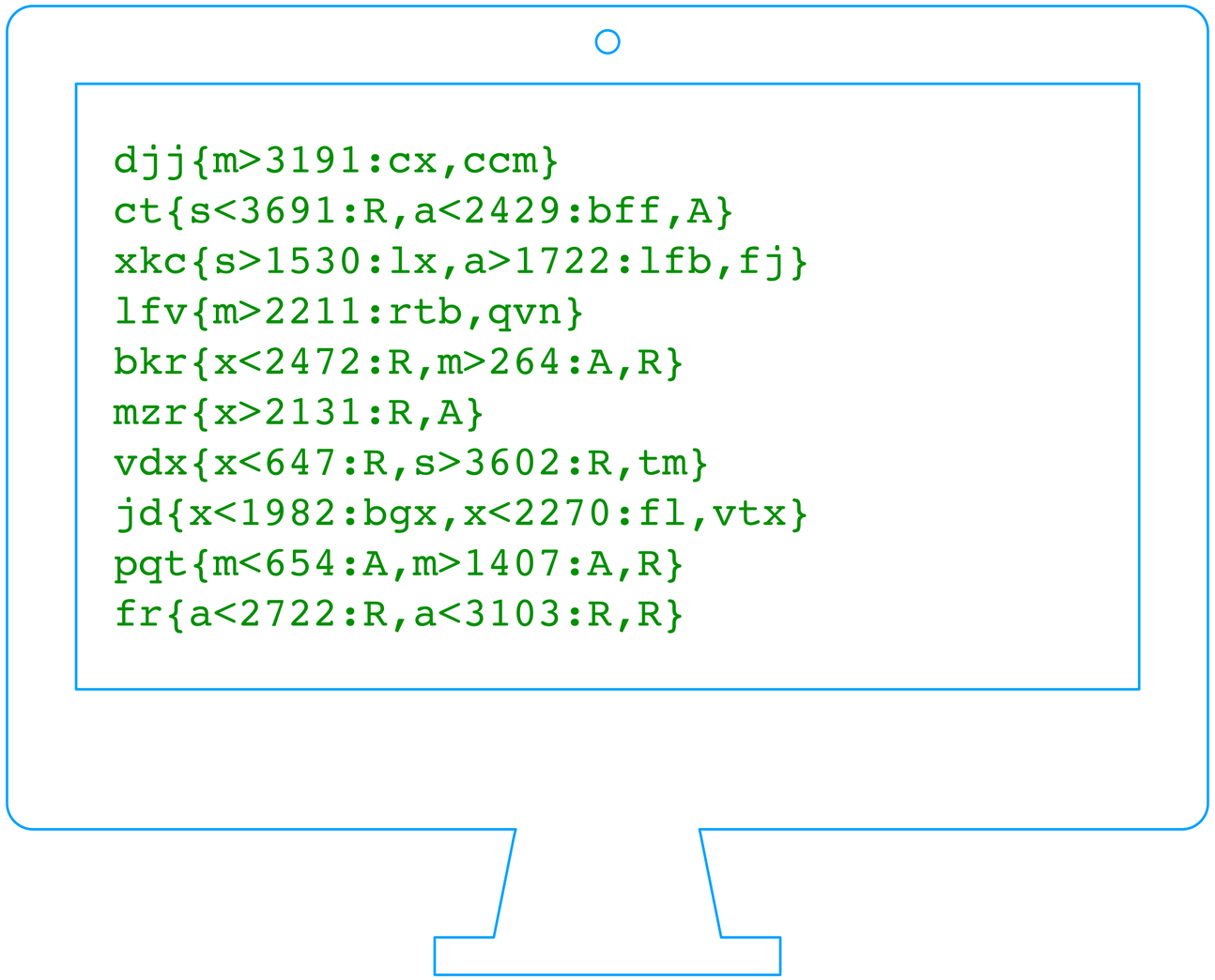
About me

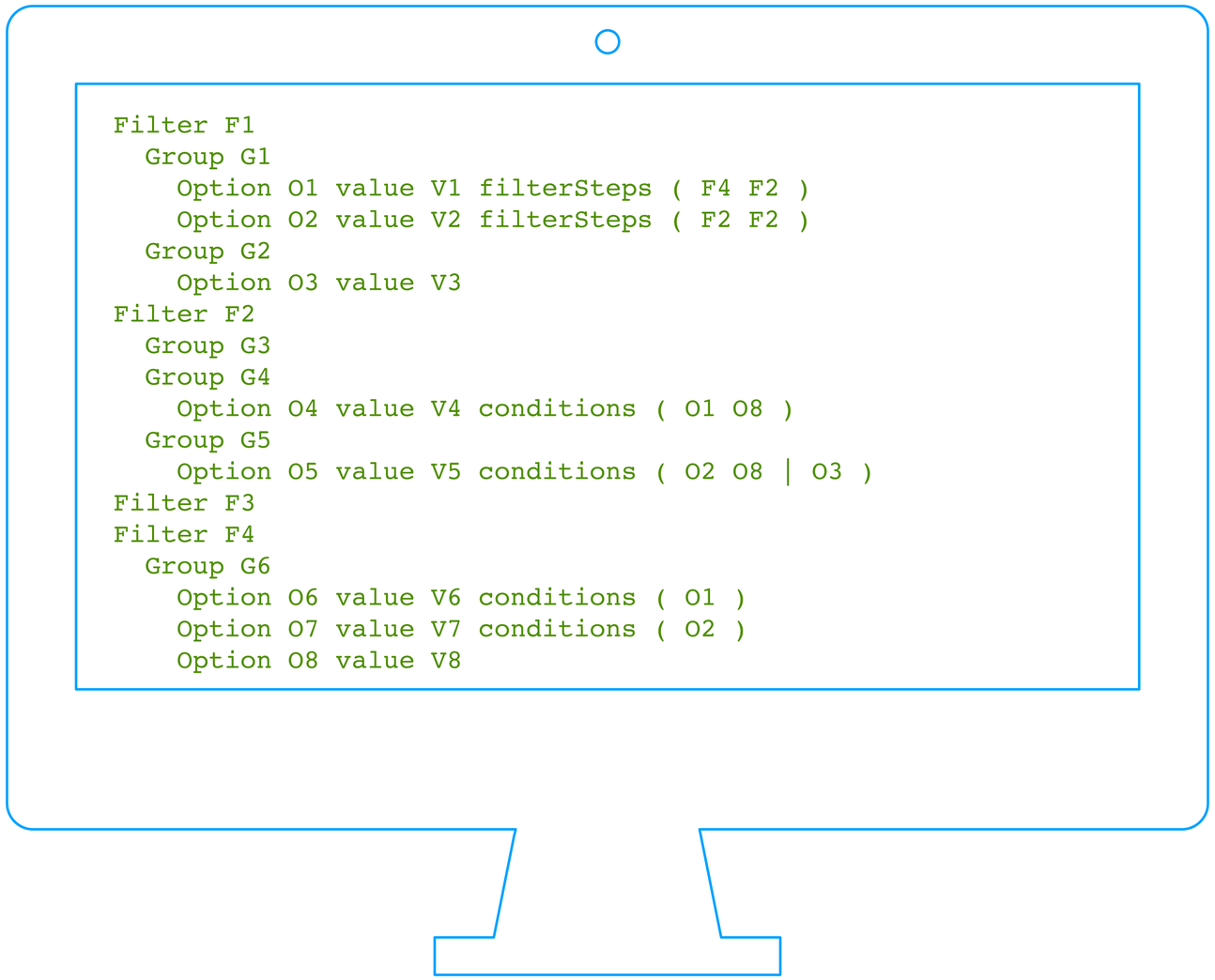
Birgit Kratz

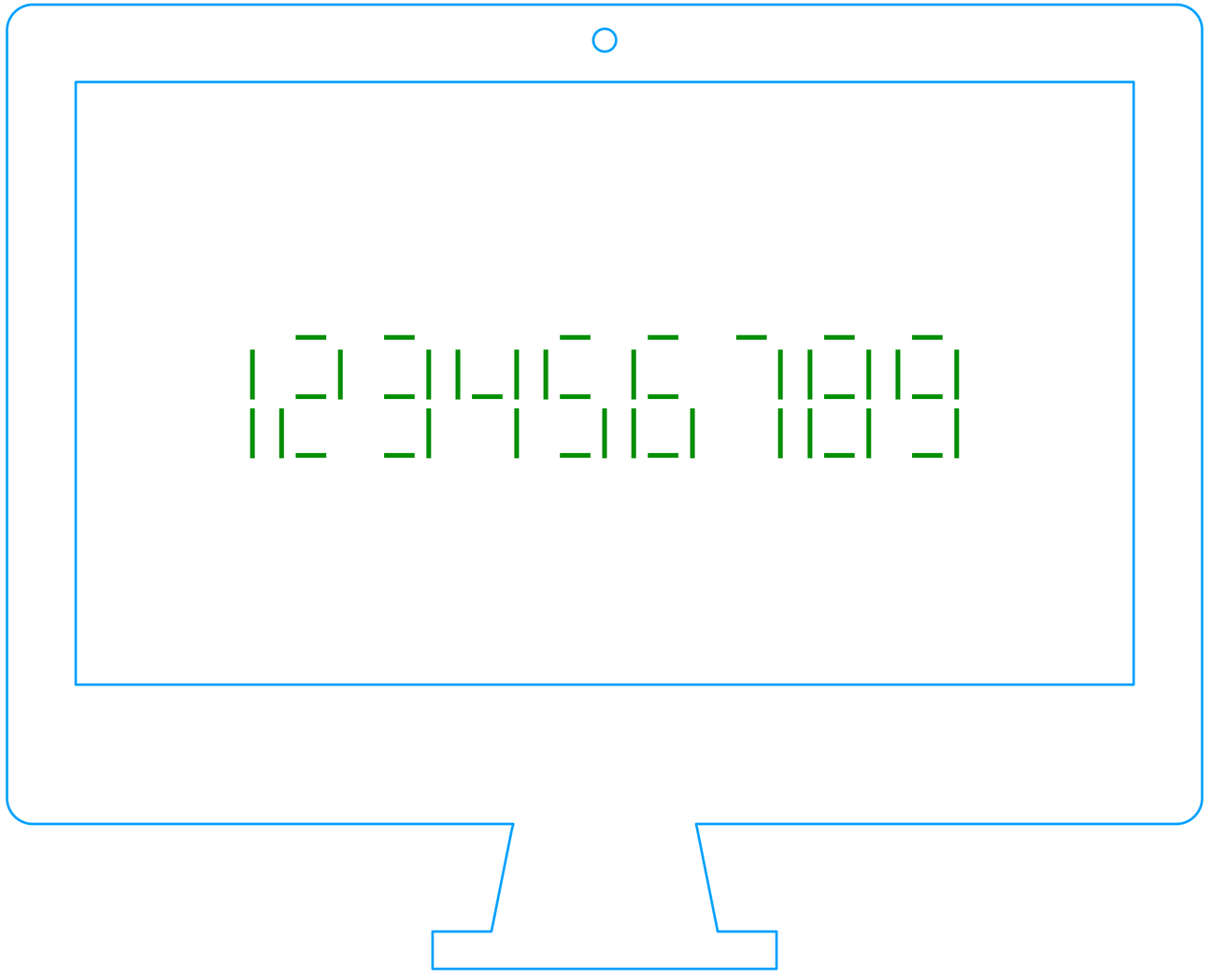
- Freelancing IT Consultant
- Java-Backend
- More than 25 years experience
- Co-Organizer of Softwerkskammer in Düsseldorf and Köln (Cologne)
- Co-Organizer of SoCraTes-Conf Germany
- Email: mail@birgitkratz.de
- Mastodon: @birgitkratz@jvm.social
- Github: https://github.com/bkratz
- Web: https://www.birgitkratz.de

What is the Problem?




djj{m>3191:cx,ccm}
ct{s<3691:R,a<2429:bff,A}
xkc{s>1530:lx,a>1722:lfb,fj}
lfv{m>2211:rtb,qvn}
bkr{x<2472:R,m>264:A,R}
mzr{x>2131:R,A}
vdx{x<647:R,s>3602:R,tm}
jd{x<1982:bgx,x<2270:fl,vtx}
pqt{m<654:A,m>1407:A,R}
fr{a<2722:R,a<3103:R,R}


Filter F1
Group G1
Option O1 value V1 filterSteps (F4 F2)
Option O2 value V2 filterSteps (F2 F2)
Group G2
Option O3 value V3
Filter F2
Group G3
Group G4
Option O4 value V4 conditions (O1 O8)
Group G5
Option O5 value V5 conditions (O2 O8 | O3)
Filter F3
Filter F4
Group G6
Option O6 value V6 conditions (O1)
Option O7 value V7 conditions (O2)
Option O8 value V8



Let’s Get Meta!

To implement a **language**, we have to build an application that **reads sentences** and **reacts appropriately** to the phrases and input symbols it discovers.*

If an application computes or executes sentences, we call that application an **INTERPRETER**.

If an application converts sentences from one language to another, we call that application a **TRANSLATOR**.

* from The Definite ANTLR 4 Reference by Terence Parr, Chapter 2

2 Main Tasks

- read sentences
  - using Lexer and Parser
- react appropriately
  - using programming (i.e. with the help of Listeners or Visitors)

What is a Lexer?

Processes and groups characters into words or symbols

- lexical analysis (**Tokenising**)

Tokens have 2 pieces of information:

- type
- text

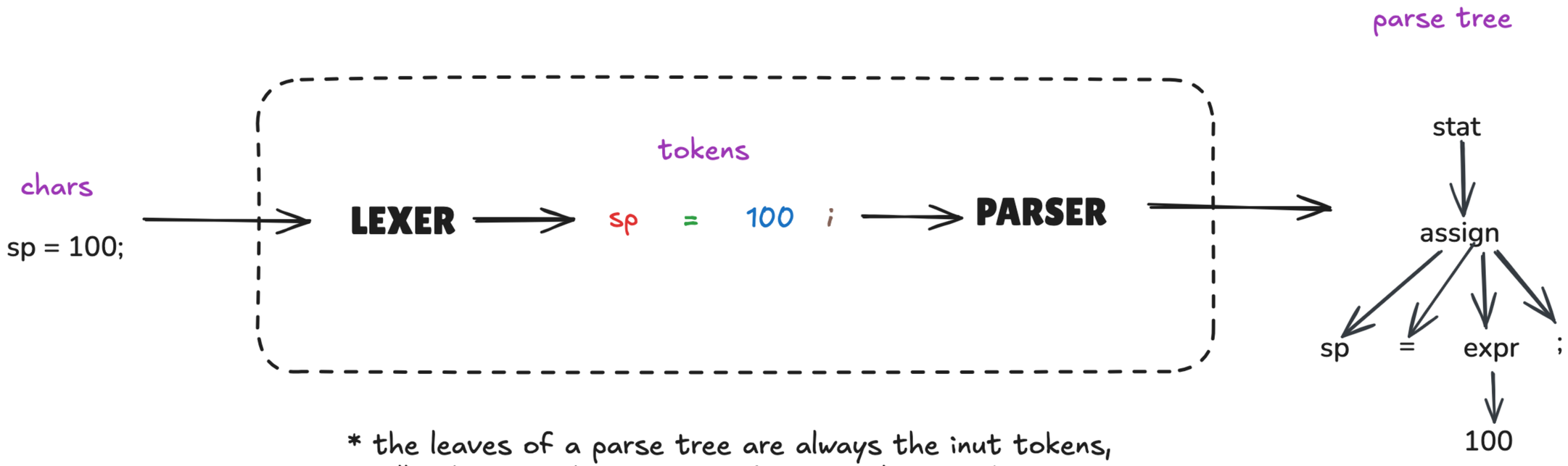
What is a Parser?

Program that checks a sentence's structure against the rules of a grammar.

Syntax: refers to **rules** governing language membership

Grammar: specification of a language syntax

How do Lexer and Parser work together?



Source: The Definite ANTLR 4 Reference by Terence Parr, page 11

Is there something that can help us doing it?

Yes

ANTLR

ANother **T**ool for **L**anguage **R**ecognition

What is ANTLR?

ANTLR is a program that writes other programs based on a **Grammar**

- Lexer
- (recursive-descent) Parser

} **Reading Sentences**

- Listeners
- Visitors

} **React appropriately**

Lexer, Parser, Listeners, Visitor can be generated in several languages:
Java (default), C#, Python 3, JavaScript, TypeScript, Go, C++, Swift, PHP, Dart

What is an ANTLR grammar?

- is derived from a language pattern like recurring grammatical structures (sequence, choice, token dependance, nested phrase)
- written down in a text file containing
  - Lexer rules
  - Parser rules

Grammar Structure

- grammar Name
- options {…} - optional
- import {…} - optional
- tokens {…} - optional
- @actionName {…} - optional,
  i.e. @header, @members, @after
- Lexer rules - starting with uppercase
- Parser rules - starting with lowercase

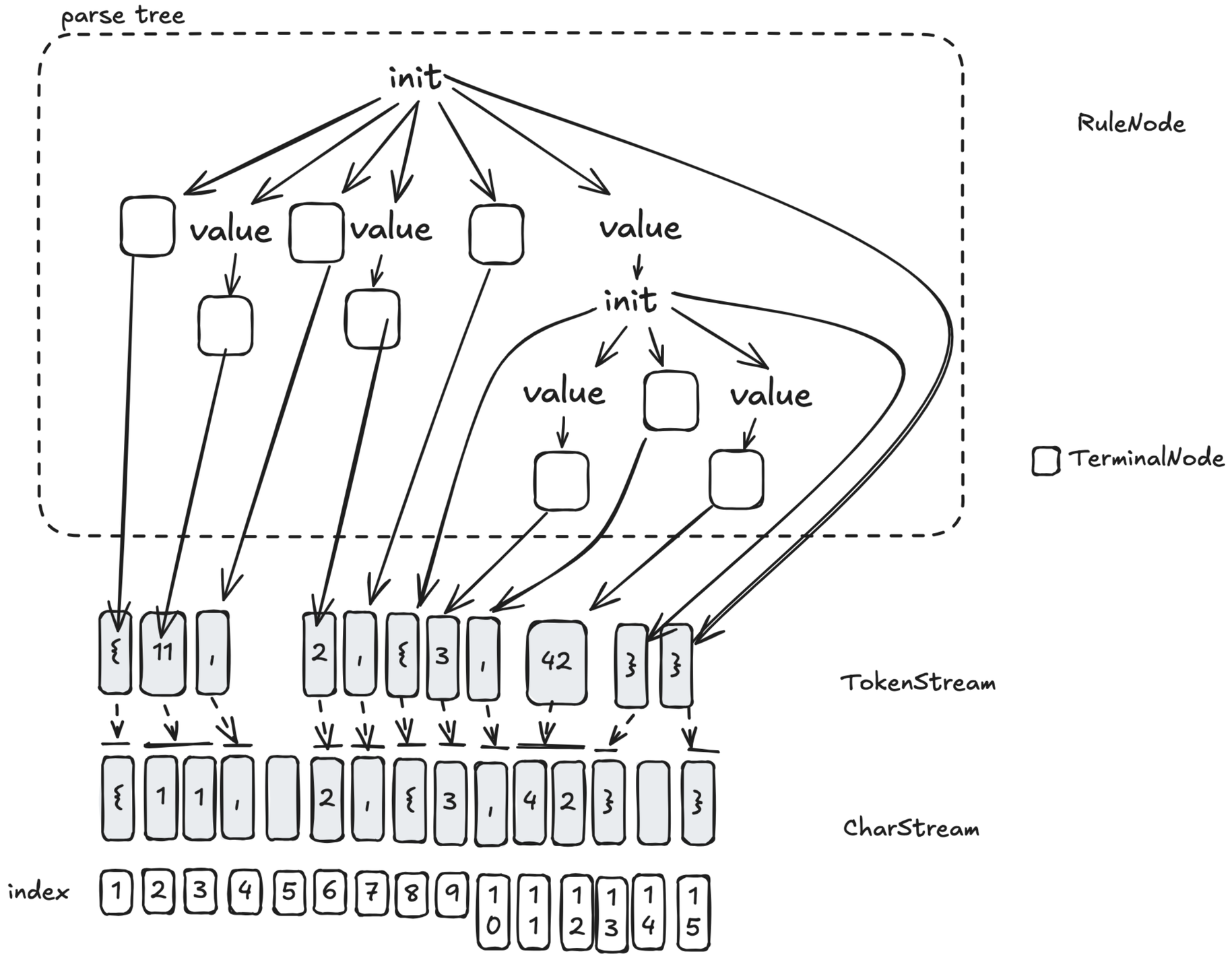
Grammar Constructs

- Comments
- Lexer rules
- Parser rules - greedy, non-greedy
- Literals
- Actions - code blocks written in target language
- Rule element labels
- Lexical Modes
- Lexer Commands - i.e. skip, more, channel, mode, …

```
/** Grammars always start with a grammar header.
*   This grammar is called ArrayInit and must match the filename: ArrayInit.g4
*/
grammar ArrayInit;

/** a rule called 'init' that matches comma-sparated values between {...} */
init: '{' value (',' value)* '}';

/** a value can be either a nested array/struct or a simple integer (INT) */
value:  init
      | INT
      ;

// parser rules start with lowercase letters, lexer rules with uppercase
INT: [0-9]+;                    // Define token INT as one or more digits
WS: [ \t\r\n]+ -> skip;         // Define whitespace rule, toss it out
```

Reading Sentences

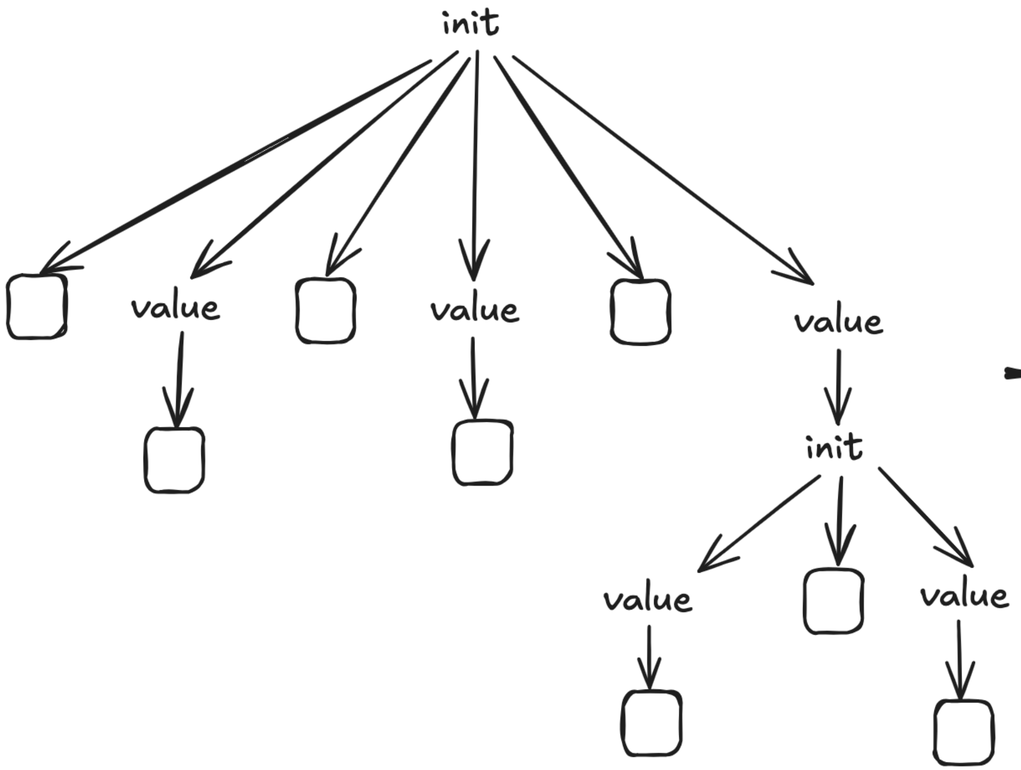
Lexer and Parser


parse tree
init
value
value
value
init
value
value
RuleNode
TerminalNode
{ 11 , 2 , { 3 , 42 } }
TokenStream
{ 1 1 , 2 , { 3 , 4 2 } }
CharStream
index
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

React appropriately

Listeners and Visitors

Listener



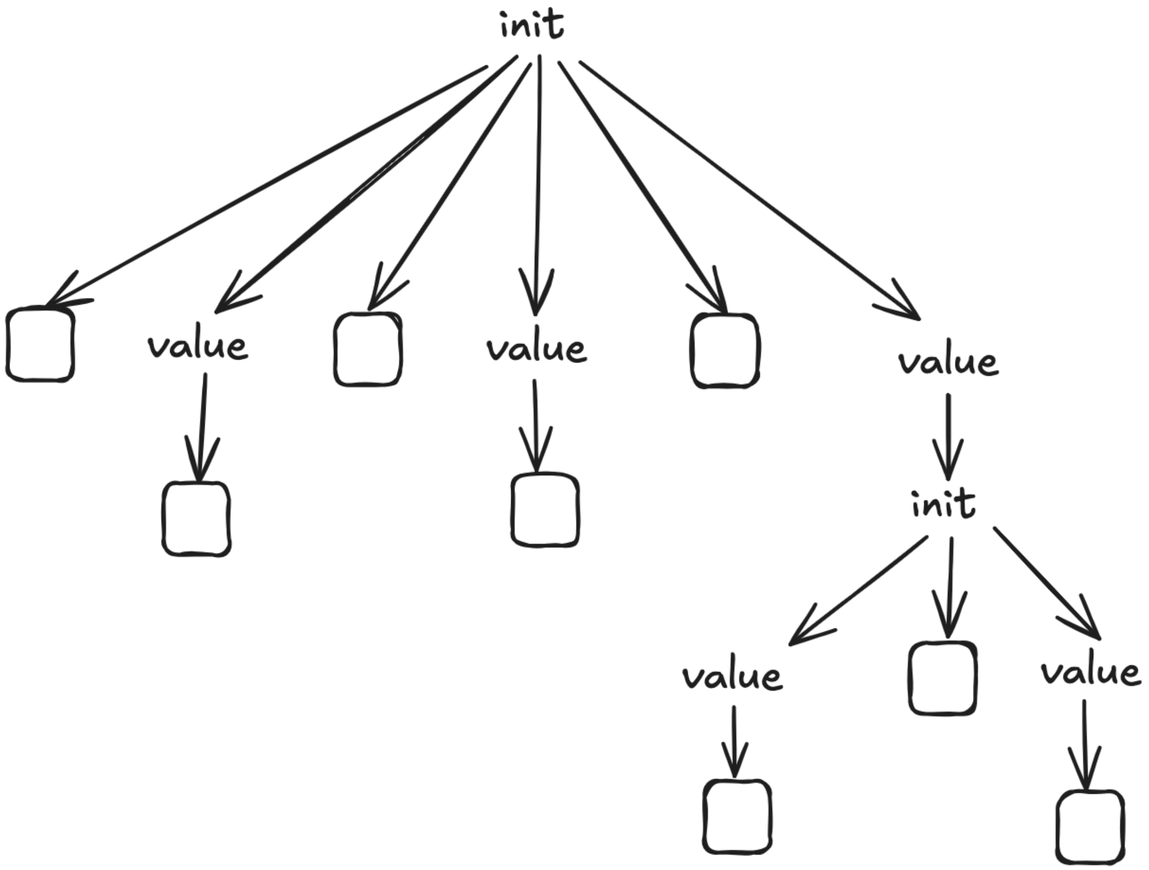
PARSE TREE WALKER

enterInit(InitContext)
visitTerminal(TerminalNode)
enterValue(ValueContext)
visitTerminal(TerminalNode)
exitValue(ValueContext)
visitTerminal(TerminalNode)
enterValue(ValueContext)
visitTerminal(TerminalNode)
exitValue(ValueContext)
visitTerminal(TerminalNode)
enterValue(ValueContext)
enterInit(InitContext)
enterValue(ValueContext)
visitTerminal(TerminalNode)
exitValue(ValueContext)
visitTerminal(TerminalNode)
enterValue(ValueContext)
visitTerminal(TerminalNode)
exitValue(ValueContext)
exitInit(InitContext)
exitValue(ValueContext)
exitInit(InitContext)

APIs

Rest of Application

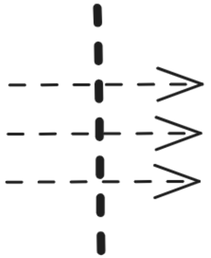
Visitor



MyVisitor

visitInit(InitContext)
visitValue(ValueContext)
visitTerminal(TerminalNode)

APIs

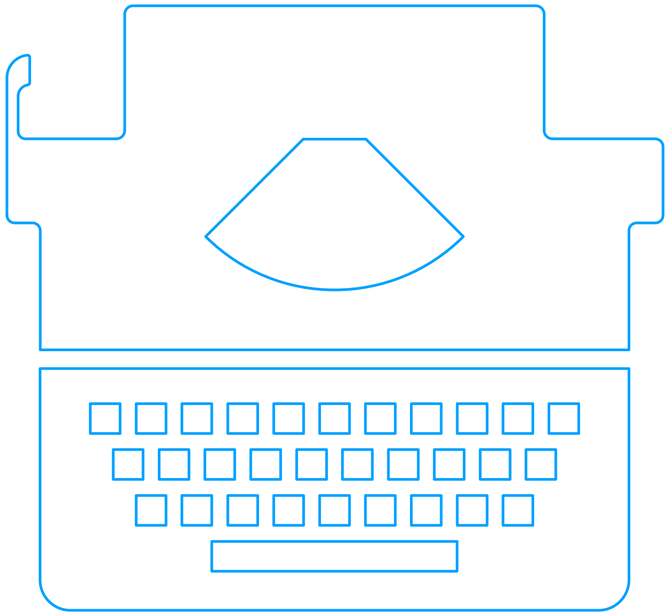


Rest of Application

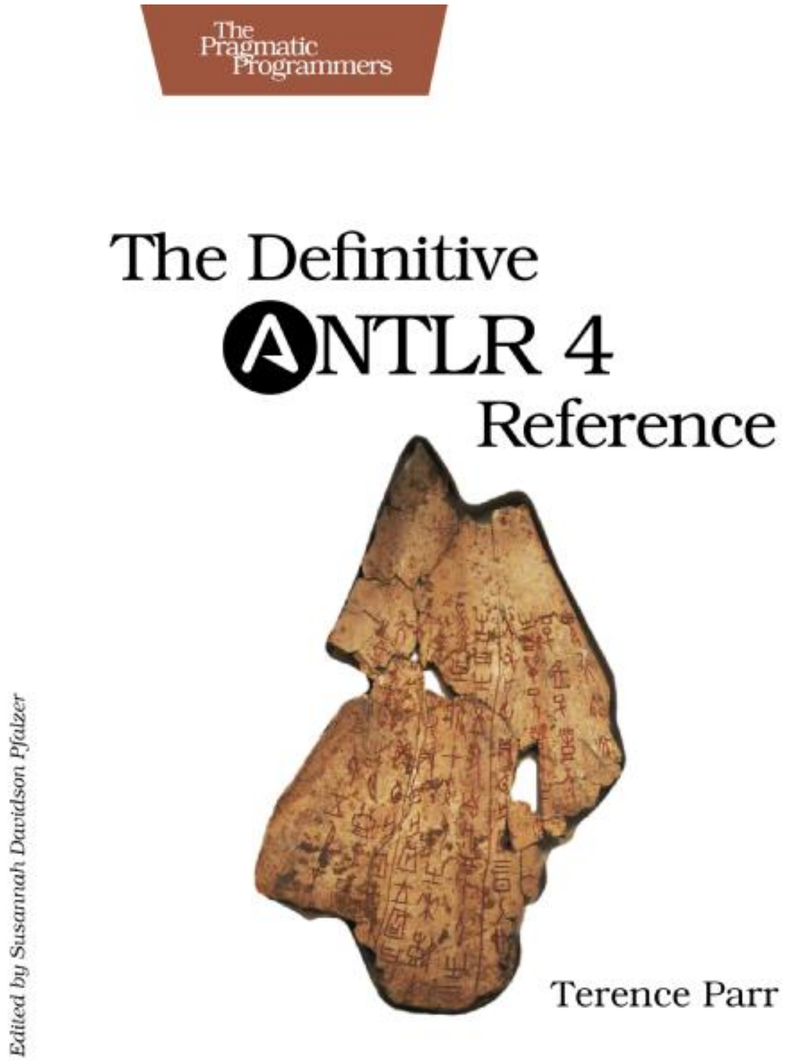
Where is ANTLR used?

- SpringDataJPA - to generate SQL Queries out of @Query annotations
- OpenRewrite - i.e. in Modul ‘rewrite-yaml’ to Parse JSONPath
- Your Project?

Demo



?



https://www.antlr.org/
https://pragprog.com/titles/tpantlr2/the-definitive-antlr-4-reference/

Thank you

Slides: https://www.birgitkratz.de/uploads/Javaland_2025_Antlr.pdf

Sample code repositories:
https://github.com/bkratz/antlr4

- Email: mail@birgitkratz.de
- Mastodon: @birgitkratz@jvm.social
- Github: https://github.com/bkratz
- Web: https://www.birgitkratz.de