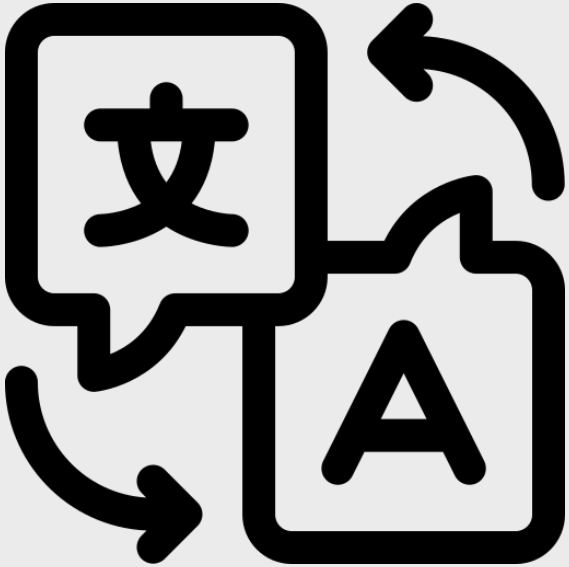


GRAMMARIZER

Catalina Aldana Mongui
Cristian Camilo Garcia
Diego Armando Velasquez

OBJETIVO

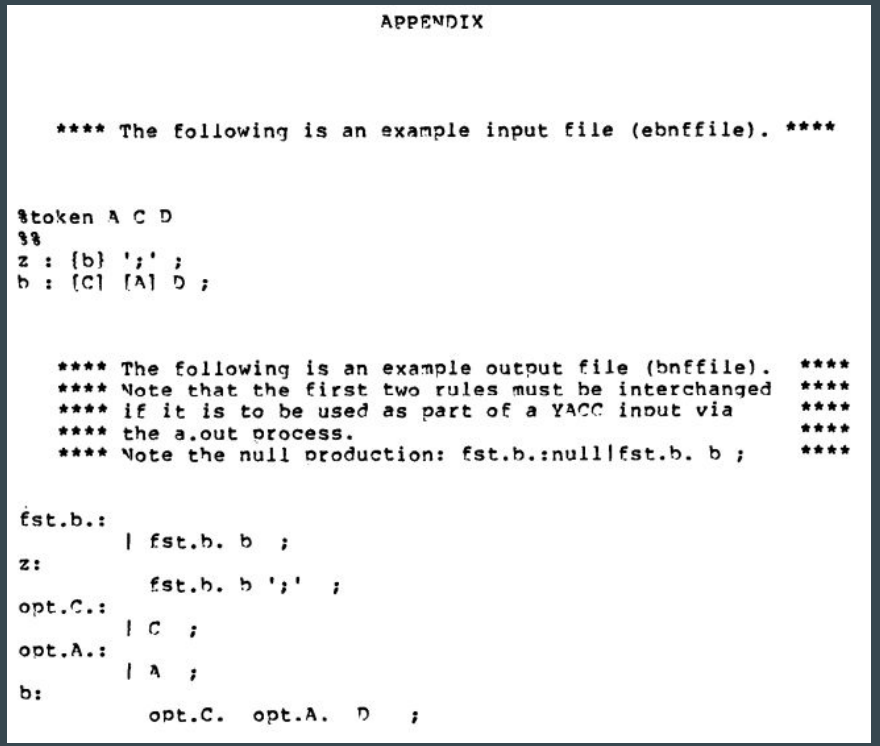
Ofrecer una herramienta sencilla que convierta una gramática de la forma EBNF a BNF

ANTECEDENTES

A PROGRAM FOR THE CONVERSION OF PRODUCTIONS IN AN EXTENDED BACKUS-NAUR-FORM TO AN EQUIVALENT BACKUS-NAUR-FORM

EARL E. MCCOY AND THOMAS WETMORE III

UNIVERSITY OF CONNECTICUT - JULY 1980



APPENDIX

**** The following is an example input file (ebnffile). ****

```
%token A C D
%%
z : {b} ';' ;
b : [C] [A] D ;
```

```
**** The following is an example output file (bnffile).  ****
**** Note that the first two rules must be interchanged  ****
**** if it is to be used as part of a YACC input via     ****
**** the a.out process.                                  ****
**** Note the null production: fst.b.:null|fst.b. b ;    ****

fst.b.:
        | fst.b. b  ;
z:
          fst.b. b ';'  ;
opt.C.:
        | C  ;
opt.A.:
        | A  ;
b:
          opt.C.  opt.A.  D   ;
```

PROPUESTA

GRAMÁTICA GENERALIZADA EBNF

⇩

DISEÑO ANTLR

Gramática

⇩

IMPLEMENTACIÓN ALGORITMO

Implementación de las clases

⇩

INTERFAZ WEB

INPUT (EBNF) → OUTPUT (BNF)

IMPLEMENTACIÓN

ANTLR 4

Python

HTML

ANTLR

```
grammar grammarizer;

grammarFile: rule* EOF;
rule: name=GRAMMAR_RULE_NAME ':' body ';';
body: alternative ('|' alternative)*;
alternative: unit+;
unit: (grouping | primitive) MULTIPLICITY_OP?;
grouping: '(' body ')';
primitive: TK_LITERAL | GRAMMAR_RULE_NAME | LEXICAL_RULE_NAME;

LEXICAL_RULE_NAME: [A-Z][A-Za-z_]*;
GRAMMAR_RULE_NAME: [a-z][A-Za-z_]*;
TK_LITERAL: '\'' .*? '\'';
MULTIPLICITY_OP: '*' | '+' | '?';
WHITESPACE: [ \t\r\n]+ -> skip;
```

GRAMÁTICA GENERALIZADA EBNF

PYTHON

```python
def visitUnit(self, ctx: Parser.UnitContext) -> Node:
    multiplied_node = (self.visitPrimitive(ctx.primitive())
                       if ctx.primitive()
                       else self.visitGrouping(ctx.grouping()))
    if not ctx.MULTIPLICITY_OP():
        return multiplied_node
    multiplier = ctx.MULTIPLICITY_OP().getText()
    if multiplier == '?':
        alternatives = [multiplied_node, ValueNode(value='')]
        return BranchingNode(alternatives=alternatives)
    aux_rule_name = f"$aux_rule{next(self.counter)}"
    aux_rule_node = ValueNode(value=aux_rule_name)
    aux_rule_alternative = SequenceNode(sub_rules=[multiplied_node, aux_rule_node])
    aux_rule_tree = BranchingNode(alternatives=[multiplied_node, aux_rule_alternative])
    self.rules_map[aux_rule_name].append(aux_rule_tree)
    if multiplier == '*':
        return BranchingNode(alternatives=[aux_rule_node, ValueNode(value='')])
    if multiplier == '+':
        return aux_rule_node
    raise ValueError(f"unhandled multiplier {multiplier}")
```

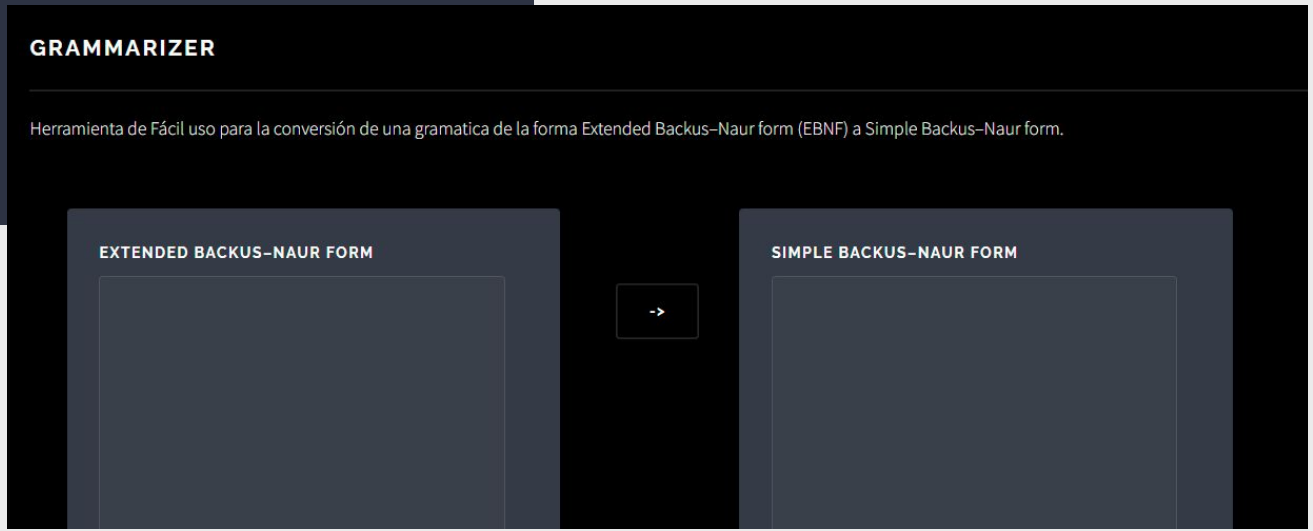
IMPLEMENTACIÓN SOBRE LOS OPERADORES

HTML

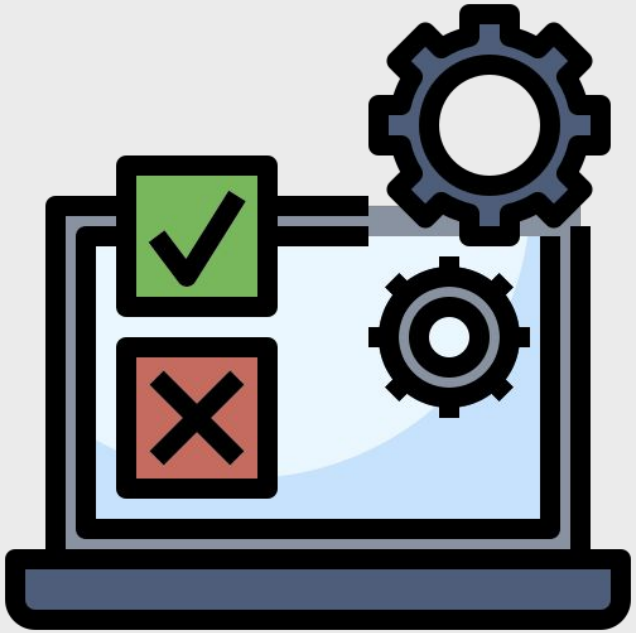
GRAMMARIZER

- Catalina Aldana Monguí
- Cristian Camilo Garcia Barrera
- Diego Armando Velasquez Vargas

PREVIEW INTERFAZ



PRUEBAS



PRUEBAS

GRAMÁTICA DE EJEMPLO

PRUEBAS

GRAMÁTICA DE SCALA

CONCLUSIONES

- Logramos desarrollar una herramienta que traduce una gramática **EBNF** a una gramática **BNF**
- Se podría mejorar el traductor aceptando comentarios y caracteres especiales como la gramática de Python
- **ANTLR** permite construir gramáticas de forma práctica y sencilla

REFERENCIAS

- Crafting Interpreters. (n.d.). Crafting Interpreters. Retrieved June 19, 2022, from http://craftinginterpreters.com/
- Parr Terence. The Definitive ANTLR 4 Reference. The pragmatic bookshelf. 2012.
- https://athena.ecs.csus.edu/~gordonvs/135/resources/05ebnfSyntaxDiagrams.pdf. (n.d.).
- A program for the conversion of productions in an extended backus-naur-form to an equivalent backus-naur-form https://apps.dtic.mil/sti/pdfs/ADA089932.pdf

GRACIAS

¿Tienen alguna pregunta?