Relatório Analisador Sintático X+++

Equipe

14104255 - Bruno Aurélio Rôzza de Moura Campos
14101370 - Fabiano Pereira de Oliveira
14101383 - Laís Ferrigo Perazzolo
14101398 - Thary Correia

Papeis no Desenvolvimento

Houve 3 encontros com **todos** os membros participando do desenvolvimento da segunda parte do trabalho.

Alterações que foram realizadas sobre o projeto sugerido nos capítulos 4 e 5 de Delamaro (2004)

- Adicionado configuração de saída na geração dos arquivos

```
OUTPUT_DIRECTORY = "parser";
```

- Criação do método `accessOperation()` com os tipos de acesso

```
void accessOperation(): {} {
    // Os qualificadores de acesso devem ser opcionais
   [<PUBLIC> | <PRIVATE> | <PROTECTED>]
}
```

- Criação do método `typeOperation()` que define a tipagem primitíva

```
void typeOperation(): {} {
    // Tipos de variáveis e literais
   <INT> | <STRING> | <BYTE> | <SHORT> | <LONG> | <FLOAT>
}
```

- Alterado método `vardecl()`
  - Adicionado token opcional `[<FINAL>]`
  - adicionado `typeOperation()` e `accessOperation()`
  - Adicionado atribuição de valor à uma variável `[<ASSIGN> factor()]`

```
void vardecl(RecoverySet g) throws ParseEOFException : {} {
    try{
        [<FINAL>]   // variavel pode ser ou não FINAL
```

```
        accessOperation() (typeOperation() | <IDENT> )
        <IDENT>
        (<LBRACKET> <RBRACKET>)*
        (<COMMA> <IDENT> ( <LBRACKET> <RBRACKET>)* )*
        [<ASSIGN> factor()]     // Atribuição de valor à uma variável
    } catch (ParseException e) {
        consumeUntil(g, e, "vardecl");
    }
}
```

- Alterado método `methoddecl()`
  - Adicionado `typeOperation()` e `accessOperation()`

```
void methoddecl(RecoverySet g) throws ParseEOFException : {} {
    try {
        (typeOperation() | accessOperation() | <IDENT>)
        (<LBRACKET> <RBRACKET>)*
        <IDENT> methodbody(g)
    } catch (ParseException e) {
       consumeUntil(g, e, "methoddecl");
    }
}
```

- Alterado método `paramlist()`
  - Adicionado `typeOperation()` e `accessOperation()`

```
void paramlist(RecoverySet g) throws ParseEOFException : {} {
    try {
        [
            (typeOperation() | accessOperation() | <IDENT> )
            <IDENT>
            (<LBRACKET> <RBRACKET>)*
            (<COMMA> (typeOperation() | <IDENT>) <IDENT> (<LBRACKET>
<RBRACKET>)* )*
        ]
    } catch (ParseException e) {
       consumeUntil(g, e, "paramlist");
    }
}
```

- Alterado método `numexpr()`
  - Adicionado tokens `<STAR> | <SLASH> | <REM>`

```
void numexpr() throws ParseEOFException : {} {
    logicalOp() ((<PLUS> | <MINUS> | <STAR> | <SLASH> | <REM> )
logicalOp())*
}
```

- Criado método `logicalOp()`

```
void logicalOp() throws ParseEOFException : {} {
    unaryexpr() (( <OR> | <AND> | <XOR> ) unaryexpr())*
}
```

- Alterado método `unaryexpr()`
  - Adicionado token `NOT`

```
void unaryexpr() throws ParseEOFException : {} {
   [(<PLUS> | <MINUS> | <NOT>)] factor()
}
```

- Alterado método `vardecl()`
  - Adicionado `<long_constant>`, `<short_constant>` e `<float_constant>`

```
void factor() throws ParseEOFException : {} {
    (
        <int_constant>
        | <string_constant>
        | <long_constant>
        | <short_constant>
        | <float_constant>
        | <null_constant>
        | lvalue(null)
        | <LPAREN> expression(null) <RPAREN>
    )
}
```

Comandos utilizados

```
sudo apt install javacc
```

- Geração do parser

```
javacc parser/langX+++.jj
```

- Geração dos arquivos `.class`

```
javac parser/langX.java
```

- Testes

```
java parser.langX testes_e_logs/teste_com_erro_classbody.x
java parser.langX testes_e_logs/teste_expressoes_logicas.x
```

- Debug Analisador Sintático

```
java parser.langX -debug_AS testes_e_logs/debugAS.x
```

Notas

- O arquivo `langX+++.jj` foi identado com 4 espaços,
- Todo o trabalho foi versionado usando a ferramenta git,
- Encoding dos arquivos: US-ASCII