Paradigmas e Linguagens de Programação

Prof. Augusto Sampaio

Aluno: Marcio Alexandre (maps3@cin.ufpe.br)

Cin – UFPE (01/04/2017)

- Passo a Passo -

Como adicionar um novo tipo e uma expressão unária/binária na Linguagem de Expressão 1.

1. Instala o javacc no eclipse
    a. Help >> Eclipse MarketPlace >> "javacc"...
2. Importa projeto: http://cin.ufpe.br/~in1007/linguagens/Expressoes1/expressao1.html
3. Adicionar no projeto nova expressão/função (ex.: Dizer o valor Asc II de um char):
    a. Esta função é uma Expressão Unária (ExpUnaria.java).
4. Editar arquivo Expressoes1.jj:
    a. Estrutura do arquivo:
        i. Option (não mexe em nada)
        ii. Parser (Begin e End) (não mexe em nada)
        iii. Tokens (onde adiciona os novos componentes da linguagem)
        iv. Valor (Define o novo valor da linguagem)
        v. Valor PValor() (adiciona o nome valor no PValor() padrão)
        vi. Expressao (onde adiciona nova expressao (unária ou binária) da linguagem)
        vii. Expressao PExpUnaria() (adiciona nova expressão como retorno, SE unária)
        viii. Expressao PExpUnaria() (adiciona nova expressão como retorno, SE binária)
        ix. Expressao PExpressao() (não mexe em nada)
        x. Programa PPrograma() (não mexe em nada)
5. **Se adicionar a nova expressao/função (ascii, length, ...).**
    a. Cria uma nova classe no pacote "le1.plp.expressions1.expression"
    b. Nome no padrão Exp"Nome da Função".java
    c. Editando nova classe Java:
        Ex.: adicionando expressão/função AscII:

```
public ExpAscii(Expressao exp) {   // (construtor)
        super(exp, "ascii");    // (nome definido para a função/expressao lá no arquivo ".jj")
}
public Valor avaliar(AmbienteExecucao amb) {  // (executa a expressão da linguagem)
         return new ValorInteiro ((int) ((ValorChar)getExp().avaliar(amb)).valor());
}
public Tipo getTipo(AmbienteCompilacao amb) {
        return TipoPrimitivo.INTEIRO;  // (o tipo do retorno da função)
}
protected boolean checaTipoElementoTerminal(AmbienteCompilacao amb) {
        return (getExp().getTipo(amb).eChar());  // (define o tipo de dado do parâmetro da expressão)
}
```

6. **Se adicionar uma novo tipo de dados (integer, Double, string, ...)**
    a. Edita a interface "Tipo.java" no pacote "le1.plp.expressions1.util".

```
public abstract boolean eInteiro();
public abstract boolean eBooleano();
public abstract boolean eChar(); // adiciona um nome método referenciando o novo tipo
```

    b. Edita o enumerador "TipoPrimitivo.java" no pacote "le1.plp.expressions1.util".

Ex.: adiconar o tipo Char.

```
INTEIRO("INTEIRO"),
BOOLEANO("BOOLEANO"),
STRING("STRING"),
CHAR("CHAR");  //adiciona o novo tipo ao enumerador

public boolean eChar() {  //adiciona o método booleano do novo tipo
        return this.eIgual(CHAR);
}
```

c. Cria uma nova classe no pacote “le1.plp.expressions1.expression” do novo tipo.

d. Nome no padrão Valor”Nome do Tipo”.java

```
public ValorChar(Character valor) { // (construtor com o novo tipo)
        super(valor);
}
public Tipo getTipo(AmbienteCompilacao amb) { // método que retorno o novo tipo
        return TipoPrimitivo.CHAR;
}
```