Grammar for Rz language

a study of rz parser 27 Sept 2010

pass -> dcl pass | tkEOF

// declaration is

// - function definition f(arg){ body }

// - variable dcl a, b, c[12] . . . ;

dcl -> tkIDENTIFIER dcl2

dcl2 -> tkLPAREN formal tkRPAREN block | var

// formal is an action routine, scans formal parameters

// rewrite it into grammar 27 Sept 2010

formal -> tkIDENTIFIER formal2

formal2 -> tkCOMMA tkIDENTIFIER formal2 | nil

var -> tkLBRACKET tkNUMBER tkRBRACKET var2 | var2

var2 -> tkCOMMA tkIDENTIFIER var | tkSEMICOLON

block -> tkLBRACE stmts tkRBRACE

stmt -> block | stmt1

stmts -> stmt1 stmts | nil

// statement is

// - ;

// - if( expr ) stmt <else stmt> optional

// - while( expr ) stmt

// - return <expr> ;

// - print( ... ) ;

// - \*name = expr ;

// - name = expr ;

// - name[ expr ] = expr ;

// - name( ... ) ; function call

stmt1 ->

 tkSEMICOLON |

 tkIF tkLPAREN expr tkRPAREN stmt elsest |

 tkWHILE tkLPAREN expr tkRPAREN stmt |

 tkRETURN returnst |

 tkPRINT tkLPAREN prlist tkRPAREN tkSEMICOLON |

 tkSTAR tkIDENTIFIER tkEQ expr tkSEMICOLON |

 tkIDENTIFIER stmt2

elsest -> tkELSE stmt | nil

returnst -> tkSEMICOLON | expr tkSEMICOLON

prlist -> p1 prlist2

prlist2 -> tkCOMMA p1 prlist2 | nil

p1 -> tkSTRING | expr

stmt2 ->

 tkEQ expr tkSEMICOLON |

 tkLBRACKET expr tkRBRACKET tkEQ expr tkSEMICOLON |

 tkLPAREN tkRPAREN tkSEMICOLON

// an expression is

// - term op term op . . . term

// - op has priority: (highest) unary \* / + - compare && ||

// - compare: lt le eq neq ge gt

// - unary: - ! \* &

// - term: number, name, name[expr], name(...)

// - ( expr ) parenthesis has highest priority

expr -> term exprs | nil

exprs -> tkOROR term exprs | nil

// flatten f1 -> f2 f1s, f2 -> f3 f2s, f3 -> item f3s

// to f1 -> item f3s f2s f1s

// term -> f1 terms

// terms -> tkANDAND f1 terms | nil

term -> item f3s f2s f1s terms

terms -> tkANDAND item f3s f2s f1s terms | nil

// f1 -> f2 f1s

f1s -> logicop f1s | nil

logicop ->

 tkLT item f3s f2s | tkLE item f3s f2s |

 tkEQEQ item f3s f2s | tkNE item f3s f2s |

 tkGE item f3s f2s | tkGT item f3s f2s

// f2 -> f3 f2s

f2s -> plusminusop f2s | nil

plusminusop -> tkPLUS item f3s | tkMINUS item f3s

// f3 -> item f3s

f3s -> muldivop f3s | nil

muldivop -> tkSTAR item | tkSLASH item

item ->

 tkMINUS t1 | tkNOT t1 |

 tkSTAR t1 | tkAND tkIDENTIFIER mod2 |

 t1

t1 ->

 tkIDENTIFIER mod | tkNUMBER |

 tkLPAREN expr tkRPAREN

mod ->

 tkLBRACKET expr tkRBRACKET |

 tkLPAREN tkRPAREN | nil

mod2 -> tkLBRACKET expr tkRBRACKET | nil

END