MSU Compilers Class 2015

Team 4 (A-Team): Todd Beckman, Jacob Barthelmeh, Robert Lewis

Generated by Doxygen 1.8.8

Sat May 2 2015 23:30:19

Contents

1 Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

2 Class Documentation

2.1 compiler.Compiler Class Reference

Static Public Member Functions

- static void **main** (String[] args)

Static Public Attributes

- static boolean **DEBUG** = false

The documentation for this class was generated from the following file:

- Compiler.java

2.2 scanner.Dispatcher Class Reference

Public Member Functions

- Token **nextToken** ()
- void close ()

  close file
- **Dispatcher** (String filename)

The documentation for this class was generated from the following file:

- Dispatcher.java

2.3 scanner.FSA Class Reference

Static Public Member Functions

- static Token TEST_DIGIT (Reader reader, int row, int col)

  Test a digit and return the token.
- static Token TEST_LETTER (Reader reader, int row, int col)

  Tests to see if this is a keyword or identifier.
- static Token TEST_STRING_LIT (Reader reader, int row, int col)

  Tests to see if this is a string literal.
- static Token TEST_COLON (Reader reader, int row, int col)

  Tests to see if this is an := or :
- static Token TEST_LTHAN (Reader reader, int row, int col)

  Tests to see if this is < or <= or <>
- static Token TEST_GTHAN (Reader reader, int row, int col)

  Tests to see if this is < or <=.

2.3.1 Member Function Documentation

2.3.1.1 static Token scanner.FSA.TEST_COLON (Reader *reader,* int *row,* int *col*) [static]

Tests to see if this is an := or :

Parameters

reader	The file reader
row	The row of the occurrence
col	The col of the occurrence

Returns

The token containing the result

2.3.1.2 static Token scanner.FSA.TEST_DIGIT (Reader *reader,* int *row,* int *col*) [static]

Test a digit and return the token.

Parameters

reader	The file reader
row	The row of the occurrence
col	The col of the occurrence

Returns

The token containing the number

2.3.1.3 static Token scanner.FSA.TEST_GTHAN (Reader *reader,* int *row,* int *col*) [static]

Tests to see if this is < or <=.

Parameters

reader	The file reader
row	The row of the occurrence
col	The col of the occurrence

Returns

The result

2.3.1.4 static Token scanner.FSA.TEST_LETTER (Reader *reader,* int *row,* int *col*) [static]

Tests to see if this is a keyword or identifier.

Parameters

reader	The file reader
row	The row of the occurrence
col	The col of the occurrence

Returns

The token containing the keyword or identifier

2.3.1.5 static Token scanner.FSA.TEST_LTHAN (Reader *reader,* int *row,* int *col*) [static]

Tests to see if this is < or <= or <>

Parameters

reader	The file reader
row	The row of the occurrence
col	The col of the occurrence

Returns

The token containing the result

2.3.1.6 static Token scanner.FSA.TEST_STRING_LIT (Reader *reader,* int *row,* int *col*) [static]

Tests to see if this is a string literal.

Parameters

reader	The file to read from
row	The row of the occurrence
col	The col of the occurrence

Returns

The token with the string

The documentation for this class was generated from the following file:

- FSA.java

2.4 util.Kind Enum Reference

Public Attributes

- **VARIABLE**

- **INVARIABLE**
- **INOUTVARIABLE**
- **INPARAMETER**
- **INOUTPARAMETER**
- **PROCEDURE**
- **FUNCTION**
- **NOKIND**

The documentation for this enum was generated from the following file:

- Kind.java

2.5 util.NonTerminal Enum Reference

Public Attributes

- **SystemGoal**
- **Program**
- **ProgramHeading**
- **Block**
- **VariableDeclarationPart**
- **VariableDeclarationTail**
- **VariableDeclaration**
- **Type**
- **ProcedureAndFunctionDeclarationPart**
- **ProcedureDeclaration**
- **FunctionDeclaration**
- **ProcedureHeading**
- **FunctionHeading**
- **OptionalFormalParameterList**
- **FormalParameterSectionTail**
- **FormalParameterSection**
- **ValueParameterSection**
- **VariableParameterSection**
- **StatementPart**
- **CompoundStatement**
- **StatementSequence**
- **StatementTail**
- **Statement**
- **EmptyStatement**
- **ReadStatement**
- **ReadParameterTail**
- **ReadParameter**
- **WriteStatement**
- **WriteParameterTail**
- **WriteParameter**
- **AssignmentStatement**
- **IfStatement**
- **OptionalElsePart**
- **RepeatStatement**
- **WhileStatement**
- **ForStatement**
- **ControlVariable**
- **InitialValue**

- **StepValue**
- **FinalValue**
- **ProcedureStatement**
- **OptionalActualParameterList**
- **ActualParameterTail**
- **ActualParameter**
- **Expression**
- **OptionalRelationalPart**
- **RelationalOperator**
- **SimpleExpression**
- **TermTail**
- **OptionalSign**
- **AddingOperator**
- **Term**
- **FactorTail**
- **MultiplyingOperator**
- **Factor**
- **ProgramIdentifier**
- **VariableIdentifier**
- **ProcedureIdentifier**
- **FunctionIdentifier**
- **BooleanExpression**
- **OrdinalExpression**
- **IdentifierList**
- **IdentifierTail**

The documentation for this enum was generated from the following file:

- NonTerminal.java

2.6 util.Operator Enum Reference

Public Member Functions

- **Operator** (int p, String code)

Public Attributes

- **ADDITION** =(0, "ADDS")
- **SUBTRACTION** =(0, "SUBS")
- **OR** =(0, "ORS")
- **MULTIPLICATION** =(1, "MULS")
- **DIVISION** =(1, "DIVS")
- **MODULO** =(1, "MODS")
- **NEGATION** =(1, "NEGS")
- **AND** =(1, "ANDS")
- **NOT** =(1, "NOTS")
- **EQUAL** =(2, "CMPEQS")
- **NEQUAL** =(2, "CMPNEQS")
- **GEQUAL** =(2, "CMPGES")
- **LEQUAL** =(2, "CMPLES")
- **LTHAN** =(2, "CMPLTS")
- **GTHAN** =(2, "CMPGTS")
- **NOOP** =(-1, "NOOP")

- int **precedence**
- String **code**

The documentation for this enum was generated from the following file:

- Operator.java

2.7 parser.Parser Class Reference

Public Member Functions

- Parser (SemanticAnalyzer sa)

  genRead in csv ll1 table
- boolean parseFile (String in)

  Set a file to parse.
- void setRuleOutputFile (String in)

  Used to set the output file for the rule tree created when parsing.

2.7.1 Constructor & Destructor Documentation

2.7.1.1 parser.Parser.Parser (SemanticAnalyzer *sa*)

genRead in csv ll1 table

Parameters

sa	semantic analyzer object

2.7.2 Member Function Documentation

2.7.2.1 boolean parser.Parser.parseFile (String *in*)

Set a file to parse.

Parameters

in	file to be parsed

Returns

0 on success

2.7.2.2 void parser.Parser.setRuleOutputFile (String *in*)

Used to set the output file for the rule tree created when parsing.

Parameters

in	name of file or directory

The documentation for this class was generated from the following file:

- Parser.java

2.8 util.Reader Class Reference

Public Member Functions

- **Reader** (String filename)
- void **close** ()
- char peekChar ()

  Reads the next character without progressing the file pointer.
- char nextChar ()

  Reads the next character and progresses the file pointer.
- void ungetChar (char c)

  Unreads a character.

Public Attributes

- int **linenumber**

2.8.1 Member Function Documentation

2.8.1.1 char util.Reader.nextChar ( )

Reads the next character and progresses the file pointer.

Returns

The character found

2.8.1.2 char util.Reader.peekChar ( )

Reads the next character without progressing the file pointer.

Returns

The character found

2.8.1.3 void util.Reader.ungetChar (char *c*)

Unreads a character.

Parameters

c	

The documentation for this class was generated from the following file:

- Reader.java

2.9 scanner.Scanner Class Reference

Public Member Functions

- Token nextToken ()

  Gets the next token from the file.
- void close ()

  Close file scanning from.
- Scanner (String filename)

  Constructs a dispatcher to read from a file.

2.9.1 Constructor & Destructor Documentation

2.9.1.1 scanner.Scanner.Scanner (String *filename*)

Constructs a dispatcher to read from a file.

Parameters

filename	The name of the file to search for

2.9.2 Member Function Documentation

2.9.2.1 Token scanner.Scanner.nextToken ( )

Gets the next token from the file.

Be sure to check for Error tokens.

Returns

The next token in the file

The documentation for this class was generated from the following file:

- Scanner.java

2.10 semanticanalyzer.SemanticAnalyzer Class Reference

Public Member Functions

- void error (String err)

  Produce an error and cancel compile.
- void genPush (SemanticRecord rec)

  Push something to the stack.
- void genStoreRegisters (int nestingL)

  Used for storing register values on the stack when starting the program.
- void genRestoreRegisters (int nestingL)

  Used for restoring register values after program run.
- void genHalt ()

  Halt the program and close the output file.
- void genMove (String from, String to)

  Move something from one place to another.
- boolean handleArithCasts (SemanticRecord left, SemanticRecord right)

  Handles casting arithmetic properly to float if necessary.
- boolean genArithOperator_S (SemanticRecord left, Operator opp, SemanticRecord right)

  Generate an arithmetic operation given two operands.
- void genLogicalOperator_S (SemanticRecord left, Operator opp, SemanticRecord right)

  Generate a logical operation given two operands.
- void genNot_S ()

  Generate a negation of a boolean.
- void genNegation_S (SemanticRecord rec)

  The record that is being negated.
- void genAssignment (SemanticRecord into, SemanticRecord from)

  Assign a variable to a destination.
- void genRead (SemanticRecord rec)

Read user input into the program.

- void startWrite (boolean writingLine)

  Flag the begin of a write statement.
- void genWrite_S (Token t)

  Generate a write statement using current writing mode.
- int newLabel ()

  Create a new label.
- void putLabel (int l)

  Put a label into the program.
- void genBranch (int l)

  Put an unconditional branch to a label.
- void genBranchFalse_S (int l)

  Put a conditional branch to a label.
- void genForInitialize (SemanticRecord control, SemanticRecord initial)

  Generate the initialize part of a for statement.
- void genForAlter (SemanticRecord control, boolean increment)

  Generate the alter part of a for statement.
- void genForTest (SemanticRecord control, boolean increment, SemanticRecord end)

  Generate the test part of a for statement.
- void onStartFormalCall (Symbol callLocation)

  Begins the logic at the start of a procedure or function call.
- void removeLocals (ArrayList< Symbol > locals)

  Removes the local variables from the current scope.
- void onEndFormalCall (Symbol callLocation)

  Produces a return statement to the appropriate location.
- void onStartActualCall (Symbol callLocation, ArrayList< SemanticRecord > actual)

  Prepares a call with the parameters provided and then makes a call to the procedure or function.
- void padForVariable ()

  Provides padding on the stack to store a variable.
- **SemanticAnalyzer** (String filename, SymbolTableHandler sh)

Public Attributes

- boolean **noerrors** = true
- SymbolTableHandler **sh**
- boolean **funcCall**

2.10.1 Member Function Documentation

2.10.1.1 void semanticanalyzer.SemanticAnalyzer.error (String *err*)

Produce an error and cancel compile.

Parameters

err	The error message

2.10.1.2 boolean semanticanalyzer.SemanticAnalyzer.genArithOperator_S (SemanticRecord *left*, Operator *opp*, SemanticRecord *right*)

Generate an arithmetic operation given two operands.

Parameters

left	The left operand
opp	The operator
right	The right operand

Returns

Whether floating point arithmetic was used

2.10.1.3 void semanticanalyzer.SemanticAnalyzer.genAssignment (SemanticRecord *into,* SemanticRecord *from*)

Assign a variable to a destination.

Type casting is handled.

Parameters

into	The location to assign to
from	Where to find the value's type

2.10.1.4 void semanticanalyzer.SemanticAnalyzer.genBranch (int *l*)

Put an unconditional branch to a label.

Parameters

l	The label to branch to.

2.10.1.5 void semanticanalyzer.SemanticAnalyzer.genBranchFalse_S (int *l*)

Put a conditional branch to a label.

Parameters

l	The label to branch to if the condition is false.

2.10.1.6 void semanticanalyzer.SemanticAnalyzer.genForAlter (SemanticRecord *control,* boolean *increment*)

Generate the alter part of a for statement.

Parameters

control	The variable to iterate over
increment	The iterate direction (up = true, down = false)

2.10.1.7 void semanticanalyzer.SemanticAnalyzer.genForInitialize (SemanticRecord *control,* SemanticRecord *initial*)

Generate the initialize part of a for statement.

Parameters

control	The variable to iterate over
initial	The initial value of that variable

2.10.1.8 void semanticanalyzer.SemanticAnalyzer.genForTest (SemanticRecord *control,* boolean *increment,* SemanticRecord *end*)

Generate the test part of a for statement.

Parameters

control	The control variable to iterate over
increment	Whether to iterate up rather than down
end	The expected end value of the for loop

2.10.1.9 void semanticanalyzer.SemanticAnalyzer.genLogicalOperator_S (SemanticRecord *left,* Operator *opp,* SemanticRecord *right*)

Generate a logical operation given two operands.

Parameters

left	The left operand
opp	The operator
right	The right operand

2.10.1.10 void semanticanalyzer.SemanticAnalyzer.genMove (String *from,* String *to*)

Move something from one place to another.

Parameters

from	The place to move from
to	The place to move to

2.10.1.11 void semanticanalyzer.SemanticAnalyzer.genNegation_S (SemanticRecord *rec*)

The record that is being negated.

Parameters

rec	The integer or float record to negate

2.10.1.12 void semanticanalyzer.SemanticAnalyzer.genPush (SemanticRecord *rec*)

Push something to the stack.

Parameters

rec	What to push onto the stack

2.10.1.13 void semanticanalyzer.SemanticAnalyzer.genRead (SemanticRecord *rec*)

Read user input into the program.

Parameters

rec	The record containing the destination to read into

2.10.1.14 void semanticanalyzer.SemanticAnalyzer.genWrite_S (Token *t*)

Generate a write statement using current writing mode.

Parameters

t	The Token containing the desired write value to write

2.10.1.15 boolean semanticanalyzer.SemanticAnalyzer.handleArithCasts (SemanticRecord *left,* SemanticRecord *right*)

Handles casting arithmetic properly to float if necessary.

Parameters

left	The left operand
right	The right operand

Returns

Whether the result is dealing with floating point arithmetic

2.10.1.16 int semanticanalyzer.SemanticAnalyzer.newLabel ( )

Create a new label.

Returns

The new label value.

2.10.1.17 void semanticanalyzer.SemanticAnalyzer.onEndFormalCall (Symbol *callLocation*)

Produces a return statement to the appropriate location.

Parameters

callLocation	The location called to

2.10.1.18 void semanticanalyzer.SemanticAnalyzer.onStartActualCall (Symbol *callLocation,* ArrayList< SemanticRecord > *actual*)

Prepares a call with the parameters provided and then makes a call to the procedure or function.

Parameters

callLocation	The destination to call to
actual	The list of actual parameters provided

2.10.1.19 void semanticanalyzer.SemanticAnalyzer.onStartFormalCall (Symbol *callLocation*)

Begins the logic at the start of a procedure or function call.

Parameters

callLocation	The location called to

2.10.1.20 void semanticanalyzer.SemanticAnalyzer.putLabel (int *l*)

Put a label into the program.

Parameters

l	The label to put into the program.

2.10.1.21 void semanticanalyzer.SemanticAnalyzer.removeLocals (ArrayList< Symbol > *locals*)

Removes the local variables from the current scope.

Parameters

locals	The list of local variables

2.10.1.22 void semanticanalyzer.SemanticAnalyzer.startWrite (boolean *writingLine*)

Flag the begin of a write statement.

Parameters

writingLine	Whether to writeln rather than just write

The documentation for this class was generated from the following file:

- SemanticAnalyzer.java

2.11 semanticanalyzer.SemanticRecord Class Reference

Public Member Functions

- SemanticRecord (Token token, Symbol symbol, String code, String opp, String register, Type type)

  Generate everything in one initialize.
- SemanticRecord (Token token, Symbol symbol)

  Allow the record to generate its own contents.
- String **toString** ()

Public Attributes

- Token **token**
- Symbol **symbol**
- String **code**
- Type **type**
- String **opp**
- String **register**

2.11.1 Constructor & Destructor Documentation

2.11.1.1 semanticanalyzer.SemanticRecord.SemanticRecord (Token *token,* Symbol *symbol,* String *code,* String *opp,* String *register,* Type *type*)

Generate everything in one initialize.

Parameters

token	
symbol	
code	
opp	
register	
type	

2.11.1.2 semanticanalyzer.SemanticRecord.SemanticRecord (Token *token,* Symbol *symbol*)

Allow the record to generate its own contents.

Parameters

token	
symbol	

The documentation for this class was generated from the following file:

- SemanticRecord.java

2.12 util.Terminal Enum Reference

Public Attributes

- **AND**
- **BEGIN**
- **BOOLEAN**
- **DIV**
- **DO**
- **DOWNTO**
- **ELSE**
- **END**
- **FALSE**
- **FIXED**
- **FLOAT**
- **FOR**
- **FUNCTION**
- **IF**
- **INTEGER**
- **MOD**
- **NOT**
- **OR**
- **PROCEDURE**
- **PROGRAM**
- **READ**
- **REPEAT**
- **STRING**
- **THEN**
- **TO**
- **TRUE**
- **UNTIL**
- **VAR**
- **WHILE**
- **WRITE**
- **WRITELN**
- **IDENTIFIER**
- **INTEGER_LIT**
- **FLOAT_LIT**
- **STRING_LIT**
- **PERIOD**
- **COMMA**
- **SCOLON**
- **COLON**
- **LPAREN**
- **RPAREN**
- **EQUAL**

- **GTHAN**
- **LTHAN**
- **LEQUAL**
- **GEQUAL**
- **NEQUAL**
- **ASSIGN**
- **PLUS**
- **MINUS**
- **TIMES**
- **FLOAT_DIVIDE**
- **EOF**

The documentation for this enum was generated from the following file:

- Terminal.java

2.13 util.Test Class Reference

Static Public Member Functions

- static void **run** (String[] args)
- static boolean parser_test (String fIn, String fOut, String file)

  Function used to test the parser operations.

2.13.1 Member Function Documentation

2.13.1.1 static boolean util.Test.parser_test (String *fIn,* String *fOut,* String *file*) [static]

Function used to test the parser operations.

Parameters

fIn	name of the input file
fOut	name of the output file for debris such as rule tree
file	The .MP file to print the machine code to

Returns

true on success, false on fail

The documentation for this class was generated from the following file:

- Test.java

2.14 compiler.Token Class Reference

Public Member Functions

- Token (String contents, Terminal terminal, int line, int col)

  Constructs a token.
- String getContents ()

  Gets the semantic content of the token.
- Terminal getTerminal ()

  Gets the classification for the token.
- int **getLine** ()
- int **getCol** ()
- String **toString** ()

2.14.1 Constructor & Destructor Documentation

2.14.1.1 compiler.Token.Token (String *contents,* **Terminal** *terminal,* int *line,* int *col*)

Constructs a token.

Parameters

contents	The semantic contents of the token
terminal	The classification of the token
line	The line number of the token occurrence
col	The column number of the token's first character

2.14.2 Member Function Documentation

2.14.2.1 String compiler.Token.getContents ( )

Gets the semantic content of the token.

Returns

The content

2.14.2.2 **Terminal** compiler.Token.getTerminal ( )

Gets the classification for the token.

Returns

The classification

The documentation for this class was generated from the following file:

- Token.java

2.15 util.Type Enum Reference

Public Attributes

- **INTEGER**
- **FIXED**
- **FLOAT**
- **STRING**
- **BOOLEAN**
- **NOTYPE**

The documentation for this enum was generated from the following file:

- Type.java

2.16 util.Writer Class Reference

Public Member Functions

- void **writeLine** (String line)
- void **close** ()

- **Writer** (String filename)

The documentation for this class was generated from the following file:

- Writer.java