

## Interpreter

Generated by Doxygen 1.8.11



# Contents

<b>1</b>	<b>Interpreter Main Page</b>	<b>1</b>
1.1	Introduction . . . . .	1
1.1.1	Scanner . . . . .	1
1.1.2	Parser . . . . .	1
<b>2</b>	<b>Hierarchical Index</b>	<b>3</b>
2.1	Class Hierarchy . . . . .	3
<b>3</b>	<b>Class Index</b>	<b>5</b>
3.1	Class List . . . . .	5
<b>4</b>	<b>Class Documentation</b>	<b>7</b>
4.1	fcsl::ast::AssignLongStmt Class Reference . . . . .	7
4.1.1	Detailed Description . . . . .	8
4.1.2	Constructor & Destructor Documentation . . . . .	8
4.1.2.1	AssignLongStmt(VarName *var_name, Expr *expr, Expr *expr2, Expr *expr3) . . . . .	8
4.1.3	Member Function Documentation . . . . .	8
4.1.3.1	unparse() . . . . .	9
4.2	fcsl::ast::AssignStmt Class Reference . . . . .	9
4.2.1	Detailed Description . . . . .	10
4.2.2	Constructor & Destructor Documentation . . . . .	10
4.2.2.1	AssignStmt(VarName *var_name, Expr *expr) . . . . .	10
4.2.3	Member Function Documentation . . . . .	10
4.2.3.1	unparse() . . . . .	10
4.3	fcsl::ast::BinaryOp Class Reference . . . . .	11

4.3.1	Detailed Description	12
4.3.2	Constructor & Destructor Documentation	12
4.3.2.1	BinaryOp(Expr *expr, std::string *op, Expr *expr2)	12
4.3.3	Member Function Documentation	12
4.3.3.1	unparse()	12
4.4	fcal::ast::BlockStmt Class Reference	13
4.4.1	Detailed Description	14
4.4.2	Constructor & Destructor Documentation	14
4.4.2.1	BlockStmt(Stmts *stmts)	14
4.4.3	Member Function Documentation	14
4.4.3.1	unparse()	14
4.5	fcal::ast::BoolFalse Class Reference	14
4.5.1	Detailed Description	15
4.5.2	Member Function Documentation	15
4.5.2.1	unparse()	15
4.6	fcal::ast::BoolTrue Class Reference	16
4.6.1	Detailed Description	17
4.6.2	Member Function Documentation	17
4.6.2.1	unparse()	17
4.7	fcal::scanner::CharConstToken Class Reference	17
4.8	fcal::scanner::DashToken Class Reference	18
4.9	fcal::ast::Decl Class Reference	19
4.9.1	Detailed Description	19
4.10	fcal::ast::EmptyStmts Class Reference	20
4.10.1	Detailed Description	21
4.10.2	Member Function Documentation	21
4.10.2.1	unparse()	21
4.11	fcal::scanner::EndOfFileToken Class Reference	21
4.12	fcal::ast::EndStmt Class Reference	22
4.12.1	Detailed Description	23

4.12.2	Member Function Documentation	23
4.12.2.1	unparse()	23
4.13	fcal::ast::Expr Class Reference	24
4.13.1	Detailed Description	25
4.14	fcal::scanner::ExtToken Class Reference	26
4.15	fcal::scanner::FalseKwdToken Class Reference	27
4.16	fcal::scanner::FloatConstToken Class Reference	28
4.17	fcal::scanner::ForwardSlashToken Class Reference	29
4.18	fcal::ast::IfElseStmt Class Reference	30
4.18.1	Detailed Description	31
4.18.2	Constructor & Destructor Documentation	31
4.18.2.1	IfElseStmt(Expr *expr, Stmt *stmt, Stmt *stmt2)	31
4.18.3	Member Function Documentation	31
4.18.3.1	unparse()	31
4.19	fcal::ast::IfExpr Class Reference	32
4.19.1	Detailed Description	33
4.19.2	Constructor & Destructor Documentation	33
4.19.2.1	IfExpr(Expr *expr, Expr *expr2, Expr *expr3)	33
4.19.3	Member Function Documentation	33
4.19.3.1	unparse()	33
4.20	fcal::ast::IfStmt Class Reference	33
4.20.1	Detailed Description	34
4.20.2	Constructor & Destructor Documentation	35
4.20.2.1	IfStmt(Expr *expr, Stmt *stmt)	35
4.20.3	Member Function Documentation	35
4.20.3.1	unparse()	35
4.21	fcal::scanner::IfToken Class Reference	35
4.22	fcal::scanner::IntConstToken Class Reference	36
4.23	fcal::scanner::LeftParenToken Class Reference	37
4.24	fcal::ast::LetExpr Class Reference	38

4.24.1 Detailed Description . . . . .	39
4.24.2 Constructor & Destructor Documentation . . . . .	39
4.24.2.1 LetExpr(Stmts *stmts, Expr *expr) . . . . .	39
4.24.3 Member Function Documentation . . . . .	39
4.24.3.1 unparse() . . . . .	39
4.25 fcal::scanner::LetToken Class Reference . . . . .	40
4.26 fcal::ast::MatrixDecl Class Reference . . . . .	41
4.26.1 Detailed Description . . . . .	42
4.26.2 Constructor & Destructor Documentation . . . . .	42
4.26.2.1 MatrixDecl(VarName *var_name, Expr *expr) . . . . .	42
4.26.3 Member Function Documentation . . . . .	42
4.26.3.1 unparse() . . . . .	42
4.27 fcal::ast::MatrixLongDecl Class Reference . . . . .	42
4.27.1 Detailed Description . . . . .	43
4.27.2 Constructor & Destructor Documentation . . . . .	44
4.27.2.1 MatrixLongDecl(VarName *var_name, Expr *expr, Expr *expr2, VarName *var_name2, VarName *var_name3, Expr *expr3) . . . . .	44
4.27.3 Member Function Documentation . . . . .	44
4.27.3.1 unparse() . . . . .	44
4.28 fcal::ast::MatrixRef Class Reference . . . . .	44
4.28.1 Detailed Description . . . . .	45
4.28.2 Constructor & Destructor Documentation . . . . .	46
4.28.2.1 MatrixRef(VarName *var_name, Expr *expr, Expr *expr2) . . . . .	46
4.28.3 Member Function Documentation . . . . .	46
4.28.3.1 unparse() . . . . .	46
4.29 MySequence< T, N > Class Template Reference . . . . .	46
4.30 fcal::ast::NestedOrFuncCall Class Reference . . . . .	47
4.30.1 Detailed Description . . . . .	48
4.30.2 Constructor & Destructor Documentation . . . . .	48
4.30.2.1 NestedOrFuncCall(VarName *var_name, Expr *expr) . . . . .	48
4.30.3 Member Function Documentation . . . . .	48

4.30.3.1	<a href="#">unparse()</a>	48
4.31	<a href="#">fcal::ast::Node Class Reference</a>	48
4.31.1	<a href="#">Detailed Description</a>	50
4.32	<a href="#">fcal::ast::NotExpr Class Reference</a>	50
4.32.1	<a href="#">Detailed Description</a>	51
4.32.2	<a href="#">Constructor &amp; Destructor Documentation</a>	51
4.32.2.1	<a href="#">NotExpr(Expr *expr)</a>	51
4.32.3	<a href="#">Member Function Documentation</a>	51
4.32.3.1	<a href="#">unparse()</a>	51
4.33	<a href="#">fcal::scanner::NotOpToken Class Reference</a>	52
4.34	<a href="#">fcal::ast::ParenExpr Class Reference</a>	53
4.34.1	<a href="#">Detailed Description</a>	54
4.34.2	<a href="#">Constructor &amp; Destructor Documentation</a>	54
4.34.2.1	<a href="#">ParenExpr(Expr *expr)</a>	54
4.34.3	<a href="#">Member Function Documentation</a>	54
4.34.3.1	<a href="#">unparse()</a>	54
4.35	<a href="#">fcal::parser::Parser Class Reference</a>	54
4.35.1	<a href="#">Member Function Documentation</a>	55
4.35.1.1	<a href="#">parse_addition(ParseResult left)</a>	55
4.35.1.2	<a href="#">parse_decl()</a>	55
4.35.1.3	<a href="#">parse_division(ParseResult left)</a>	55
4.35.1.4	<a href="#">parse_false_kwd()</a>	55
4.35.1.5	<a href="#">parse_float_const()</a>	55
4.35.1.6	<a href="#">parse_if_expr()</a>	56
4.35.1.7	<a href="#">parse_int_const()</a>	56
4.35.1.8	<a href="#">parse_let_expr()</a>	56
4.35.1.9	<a href="#">parse_matrix_decl()</a>	56
4.35.1.10	<a href="#">parse_multiplication(ParseResult left)</a>	56
4.35.1.11	<a href="#">parse_nested_expr()</a>	56
4.35.1.12	<a href="#">parse_not_expr()</a>	56

4.35.1.13	<a href="#">parse_relational_expr(ParseResult left)</a>	56
4.35.1.14	<a href="#">parse_standard_decl()</a>	56
4.35.1.15	<a href="#">parse_stmt()</a>	57
4.35.1.16	<a href="#">parse_stmts()</a>	57
4.35.1.17	<a href="#">parse_string_const()</a>	57
4.35.1.18	<a href="#">parse_subtraction(ParseResult left)</a>	57
4.35.1.19	<a href="#">parse_true_kwd()</a>	57
4.35.1.20	<a href="#">parse_variable_name()</a>	57
4.36	<a href="#">fcal::parser::ParseResult Class Reference</a>	58
4.37	<a href="#">fcal::scanner::PlusSignToken Class Reference</a>	58
4.38	<a href="#">fcal::ast::PrintStmt Class Reference</a>	59
4.38.1	Detailed Description	60
4.38.2	Constructor & Destructor Documentation	60
4.38.2.1	<a href="#">PrintStmt(Expr *expr)</a>	60
4.38.3	Member Function Documentation	60
4.38.3.1	<a href="#">unparse()</a>	60
4.39	<a href="#">fcal::ast::Program Class Reference</a>	61
4.39.1	Detailed Description	62
4.39.2	Constructor & Destructor Documentation	62
4.39.2.1	<a href="#">Program(VarName *v, Stmts *s)</a>	62
4.39.2.2	<a href="#">~Program()</a>	62
4.39.3	Member Function Documentation	62
4.39.3.1	<a href="#">unparse()</a>	62
4.40	<a href="#">fcal::scanner::RelationalOpToken Class Reference</a>	63
4.41	<a href="#">fcal::ast::RepeatStmt Class Reference</a>	64
4.41.1	Detailed Description	65
4.41.2	Constructor & Destructor Documentation	65
4.41.2.1	<a href="#">RepeatStmt(VarName *var_name, Expr *expr, Expr *expr2, Stmt *stmt)</a>	65
4.41.3	Member Function Documentation	65
4.41.3.1	<a href="#">unparse()</a>	65



4.42	<a href="#">fcal::scanner::Scanner Class Reference</a>	65
4.42.1	<a href="#">Detailed Description</a>	66
4.42.2	<a href="#">Constructor &amp; Destructor Documentation</a>	66
4.42.2.1	<a href="#">Scanner()</a>	66
4.42.3	<a href="#">Member Function Documentation</a>	66
4.42.3.1	<a href="#">consume_whitespace_and_comments(regex_t *white_space, regex_t *block_↵comment, regex_t *single_comment, const char *text)</a>	66
4.42.3.2	<a href="#">InitRegexTokenArray()</a>	66
4.42.3.3	<a href="#">Scan(const char *text)</a>	66
4.43	<a href="#">fcal::ast::SeqStmts Class Reference</a>	67
4.43.1	<a href="#">Detailed Description</a>	68
4.43.2	<a href="#">Constructor &amp; Destructor Documentation</a>	68
4.43.2.1	<a href="#">SeqStmts Stmt *stmt, Stmts *stmts)</a>	68
4.43.3	<a href="#">Member Function Documentation</a>	68
4.43.3.1	<a href="#">unparse()</a>	68
4.44	<a href="#">fcal::scanner::StarToken Class Reference</a>	69
4.45	<a href="#">fcal::ast::Stmt Class Reference</a>	70
4.45.1	<a href="#">Detailed Description</a>	71
4.46	<a href="#">fcal::ast::StmtDecl Class Reference</a>	71
4.46.1	<a href="#">Detailed Description</a>	72
4.46.2	<a href="#">Constructor &amp; Destructor Documentation</a>	72
4.46.2.1	<a href="#">StmtDecl(Decl *decl)</a>	72
4.46.3	<a href="#">Member Function Documentation</a>	72
4.46.3.1	<a href="#">unparse()</a>	72
4.47	<a href="#">fcal::ast::Stmts Class Reference</a>	72
4.47.1	<a href="#">Detailed Description</a>	73
4.48	<a href="#">fcal::scanner::StringConstToken Class Reference</a>	74
4.49	<a href="#">fcal::scanner::Token Class Reference</a>	75
4.49.1	<a href="#">Detailed Description</a>	75
4.50	<a href="#">fcal::scanner::TrueKwdToken Class Reference</a>	76
4.51	<a href="#">fcal::ast::TypeConst Class Reference</a>	77

4.51.1 Detailed Description . . . . .	78
4.51.2 Constructor & Destructor Documentation . . . . .	78
4.51.2.1 TypeConst(std::string type_const) . . . . .	78
4.51.3 Member Function Documentation . . . . .	78
4.51.3.1 unparse() . . . . .	78
4.52 fcal::ast::TypeDecl Class Reference . . . . .	78
4.52.1 Detailed Description . . . . .	79
4.52.2 Constructor & Destructor Documentation . . . . .	80
4.52.2.1 TypeDecl(VarName *type, VarName *var_name) . . . . .	80
4.52.3 Member Function Documentation . . . . .	80
4.52.3.1 unparse() . . . . .	80
4.53 fcal::scanner::VariableNameToken Class Reference . . . . .	80
4.54 fcal::ast::VarName Class Reference . . . . .	81
4.54.1 Detailed Description . . . . .	82
4.54.2 Constructor & Destructor Documentation . . . . .	82
4.54.2.1 VarName(std::string lexeme) . . . . .	82
4.54.3 Member Function Documentation . . . . .	82
4.54.3.1 unparse() . . . . .	82
4.55 fcal::ast::WhileStmt Class Reference . . . . .	83
4.55.1 Detailed Description . . . . .	84
4.55.2 Constructor & Destructor Documentation . . . . .	84
4.55.2.1 WhileStmt(Expr *expr, Stmt *stmt) . . . . .	84
4.55.3 Member Function Documentation . . . . .	84
4.55.3.1 unparse() . . . . .	84
<b>Index</b>	<b>85</b>

# Chapter 1

## Interpreter Main Page

### 1.1 Introduction

This is the introduction to iteration 3 of the interpreter project. So far we have created the scanner and parser for the interpreter. The scanner will read from a file and create a linked list of tokens that all contain Enumerated Tokentypes and using these Enumerated Tokentypes the parser is then able to generate an Abstract Syntax Tree (AST). The linked list of tokens is passed to the parser and using the Tokentypes is able to parse them into an AST and with each Node in the AST is able to unparse which will generate c++ code equivalent to the FCAL language we are interpreting from

#### 1.1.1 Scanner

The scanner reads in characters from another file and using regex expressions the scanner is able to categorize which characters are which Enumerated Tokentype. At the same time the scanner is also scanning for white space which it gets rid of using the regex for white space and bypasses the white space by moving the pointer reading the input file. After each character is properly categorized it is placed as a Token type in a linked list.

#### 1.1.2 Parser

The Parser reads in the Token linked list from the scanner and goes through each Token in the linked list and generates a subclass according to the TokenType of each Token in the linked list. The first class generated is always the Root class which is the root of the AST that will be generated by the Parser. After this Root class has been generated other Stmt, Stmts, Expr, and Decl subclasses will be generated according to the TokenTypes of the rest of the Tokens in the Token linked list that was passed by the Scanner.



## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

fcsl::scanner::ExtToken . . . . .	26
fcsl::scanner::CharConstToken . . . . .	17
fcsl::scanner::DashToken . . . . .	18
fcsl::scanner::EndOfFileToken . . . . .	21
fcsl::scanner::FalseKwdToken . . . . .	27
fcsl::scanner::FloatConstToken . . . . .	28
fcsl::scanner::ForwardSlashToken . . . . .	29
fcsl::scanner::IfToken . . . . .	35
fcsl::scanner::IntConstToken . . . . .	36
fcsl::scanner::LeftParenToken . . . . .	37
fcsl::scanner::LetToken . . . . .	40
fcsl::scanner::NotOpToken . . . . .	52
fcsl::scanner::PlusSignToken . . . . .	58
fcsl::scanner::RelationalOpToken . . . . .	63
fcsl::scanner::StarToken . . . . .	69
fcsl::scanner::StringConstToken . . . . .	74
fcsl::scanner::TrueKwdToken . . . . .	76
fcsl::scanner::VariableNameToken . . . . .	80
MySequence< T, N > . . . . .	46
fcsl::ast::Node . . . . .	48
fcsl::ast::Decl . . . . .	19
fcsl::ast::MatrixDecl . . . . .	41
fcsl::ast::MatrixLongDecl . . . . .	42
fcsl::ast::TypeDecl . . . . .	78
fcsl::ast::Expr . . . . .	24
fcsl::ast::BinaryOp . . . . .	11
fcsl::ast::BoolFalse . . . . .	14
fcsl::ast::BoolTrue . . . . .	16
fcsl::ast::IfExpr . . . . .	32
fcsl::ast::LetExpr . . . . .	38
fcsl::ast::MatrixRef . . . . .	44
fcsl::ast::NestedOrFuncCall . . . . .	47
fcsl::ast::NotExpr . . . . .	50
fcsl::ast::ParenExpr . . . . .	53
fcsl::ast::TypeConst . . . . .	77

fcal::ast::VarName . . . . .	81
fcal::ast::Program . . . . .	61
fcal::ast::Stmt . . . . .	70
fcal::ast::AssignLongStmt . . . . .	7
fcal::ast::AssignStmt . . . . .	9
fcal::ast::BlockStmt . . . . .	13
fcal::ast::EndStmt . . . . .	22
fcal::ast::IfElseStmt . . . . .	30
fcal::ast::IfStmt . . . . .	33
fcal::ast::PrintStmt . . . . .	59
fcal::ast::RepeatStmt . . . . .	64
fcal::ast::StmtDecl . . . . .	71
fcal::ast::WhileStmt . . . . .	83
fcal::ast::Stmts . . . . .	72
fcal::ast::EmptyStmts . . . . .	20
fcal::ast::SeqStmts . . . . .	67
fcal::parser::Parser . . . . .	54
fcal::parser::ParseResult . . . . .	58
fcal::scanner::Scanner . . . . .	65
fcal::scanner::Token . . . . .	75

## Chapter 3

# Class Index

### 3.1 Class List

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

<a href="#">fcal::ast::AssignLongStmt</a>	7
Inherits directly from the abstract <a href="#">Stmt</a> parent	
<a href="#">fcal::ast::AssignStmt</a>	9
Inherits directly from the abstract <a href="#">Stmt</a> parent class	
<a href="#">fcal::ast::BinaryOp</a>	11
<a href="#">fcal::ast::BlockStmt</a>	13
Inherits directly from the abstract <a href="#">Stmt</a> parent class	
<a href="#">fcal::ast::BoolFalse</a>	14
Inherits directly from the abstract <a href="#">Expr</a> class	
<a href="#">fcal::ast::BoolTrue</a>	16
Inherits directly from the abstract <a href="#">Expr</a> class	
<a href="#">fcal::scanner::CharConstToken</a>	17
<a href="#">fcal::scanner::DashToken</a>	18
<a href="#">fcal::ast::Decl</a>	19
<a href="#">fcal::ast::EmptyStmts</a>	20
Inherits directly from the abstract <a href="#">Stmts</a> parent class	
<a href="#">fcal::scanner::EndOfFileToken</a>	21
<a href="#">fcal::ast::EndStmt</a>	22
Inherits directly from the abstract <a href="#">Stmt</a> parent class	
<a href="#">fcal::ast::Expr</a>	24
<a href="#">fcal::scanner::ExtToken</a>	26
<a href="#">fcal::scanner::FalseKwdToken</a>	27
<a href="#">fcal::scanner::FloatConstToken</a>	28
<a href="#">fcal::scanner::ForwardSlashToken</a>	29
<a href="#">fcal::ast::IfElseStmt</a>	30
Inherits directly from the abstract <a href="#">Stmt</a> parent class	
<a href="#">fcal::ast::IfExpr</a>	32
Inherits directly from the abstract <a href="#">Expr</a> class	
<a href="#">fcal::ast::IfStmt</a>	33
Inherits directly from the abstract <a href="#">Stmt</a> parent class	
<a href="#">fcal::scanner::IfToken</a>	35
<a href="#">fcal::scanner::IntConstToken</a>	36
<a href="#">fcal::scanner::LeftParenToken</a>	37
<a href="#">fcal::ast::LetExpr</a>	38
Inherits directly from the abstract <a href="#">Expr</a> class	

<a href="#">fcal::scanner::LetToken</a>	40
<a href="#">fcal::ast::MatrixDecl</a>	
Inherits directly from the abstract <a href="#">Decl</a> class	41
<a href="#">fcal::ast::MatrixLongDecl</a>	
Inherits directly from the abstract <a href="#">Decl</a> class	42
<a href="#">fcal::ast::MatrixRef</a>	
Inherits directly from the abstract <a href="#">Expr</a> class	44
<a href="#">MySequence&lt; T, N &gt;</a>	46
<a href="#">fcal::ast::NestedOrFuncCall</a>	
Inherits directly from the abstract <a href="#">Expr</a> class	47
<a href="#">fcal::ast::Node</a>	48
<a href="#">fcal::ast::NotExpr</a>	
Inherits directly from the abstract <a href="#">Expr</a> class	50
<a href="#">fcal::scanner::NotOpToken</a>	52
<a href="#">fcal::ast::ParenExpr</a>	
Inherits directly from the abstract <a href="#">Expr</a> class	53
<a href="#">fcal::parser::Parser</a>	54
<a href="#">fcal::parser::ParseResult</a>	58
<a href="#">fcal::scanner::PlusSignToken</a>	58
<a href="#">fcal::ast::PrintStmt</a>	
Inherits directly from the abstract <a href="#">Stmt</a> parent class	59
<a href="#">fcal::ast::Program</a>	61
<a href="#">fcal::scanner::RelationalOpToken</a>	63
<a href="#">fcal::ast::RepeatStmt</a>	
Inherits directly from the abstract <a href="#">Stmt</a> parent class	64
<a href="#">fcal::scanner::Scanner</a>	65
<a href="#">fcal::ast::SeqStmts</a>	
Inherits directly from the abstract <a href="#">Stmts</a> parent class	67
<a href="#">fcal::scanner::StarToken</a>	69
<a href="#">fcal::ast::Stmt</a>	70
<a href="#">fcal::ast::StmtDecl</a>	
Inherits directly from the abstract <a href="#">Stmt</a> parent class	71
<a href="#">fcal::ast::Stmts</a>	72
<a href="#">fcal::scanner::StringConstToken</a>	74
<a href="#">fcal::scanner::Token</a>	75
<a href="#">fcal::scanner::TrueKwdToken</a>	76
<a href="#">fcal::ast::TypeConst</a>	77
<a href="#">fcal::ast::TypeDecl</a>	78
<a href="#">fcal::scanner::VariableNameToken</a>	80
<a href="#">fcal::ast::VarName</a>	81
<a href="#">fcal::ast::WhileStmt</a>	
Inherits directly from the abstract <a href="#">Stmt</a> parent class	83



## Chapter 4

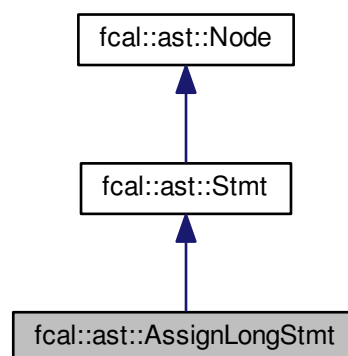
# Class Documentation

### 4.1 fcal::ast::AssignLongStmt Class Reference

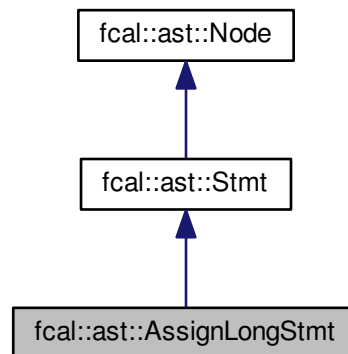
The [AssignLongStmt](#) class inherits directly from the abstract [Stmt](#) parent.

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::AssignLongStmt:



Collaboration diagram for `fcval::ast::AssignLongStmt`:



## Public Member Functions

- [AssignLongStmt](#) ([VarName](#) \*var\_name, [Expr](#) \*expr, [Expr](#) \*expr2, [Expr](#) \*expr3)
- `std::string unparse ()`  
[AssignLongStmt unparse\(\)](#) method.
- `std::string cpp_code ()`

### 4.1.1 Detailed Description

The [AssignLongStmt](#) class inherits directly from the abstract [Stmt](#) parent.

### 4.1.2 Constructor & Destructor Documentation

4.1.2.1 `fcval::ast::AssignLongStmt::AssignLongStmt ( VarName * var_name, Expr * expr, Expr * expr2, Expr * expr3 )`  
`[inline], [explicit]`

[AssignLongStmt](#) production class takes the parameters: \*var\_name, \*expr, expr2, and \*expr3

#### Parameters

*var_name	is the name of the variable being assigned
*expr	is the first parameter in a matrix sequence
*expr2	is the second parameter in a matrix sequence
*expr3	is the expression being assigned to the specific matrix position

### 4.1.3 Member Function Documentation

4.1.3.1 `std::string fcal::ast::AssignLongStmt::unparse ( )` [virtual]

[AssignLongStmt unparse\(\)](#) method.

[AssignLongStmt unparse\(\)](#) returns `var_name_`, `expr_`, `expr2_` and `expr3_`.

Implements [fcal::ast::Node](#).

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

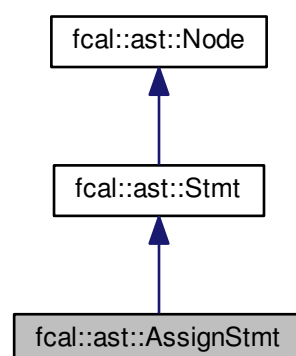
- `include/ast.h`
- `src/ast.cc`

## 4.2 fcal::ast::AssignStmt Class Reference

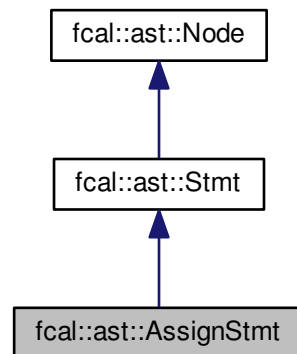
The [AssignStmt](#) class inherits directly from the abstract [Stmt](#) parent class.

```
#include <ast.h>
```

Inheritance diagram for `fcal::ast::AssignStmt`:



Collaboration diagram for `fcgal::ast::AssignStmt`:



## Public Member Functions

- [AssignStmt](#) ([VarName](#) \*var\_name, [Expr](#) \*expr)
- `std::string` [unparse](#) ()  
[AssignStmt unparse\(\)](#) method.
- `std::string` [cpp\\_code](#) ()

### 4.2.1 Detailed Description

The [AssignStmt](#) class inherits directly from the abstract [Stmt](#) parent class.

### 4.2.2 Constructor & Destructor Documentation

4.2.2.1 `fcgal::ast::AssignStmt::AssignStmt ( VarName * var_name, Expr * expr )` `[inline]`, `[explicit]`

[AssignStmt](#) production class takes the parameters: \*var\_name and \*expr

#### Parameters

*var_name	is the name of the variable being assigned
*expr	is the expression being assigned to the variable name

### 4.2.3 Member Function Documentation

4.2.3.1 `std::string` `fcgal::ast::AssignStmt::unparse ( )` `[virtual]`

[AssignStmt unparse\(\)](#) method.

[AssignStmt unparse\(\)](#) returns `var_name_`, `expr_`.

Implements [fcal::ast::Node](#).

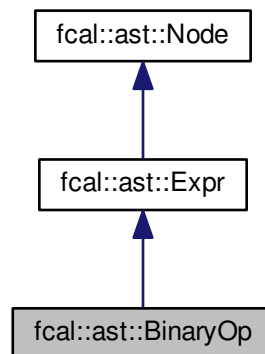
The documentation for this class was generated from the following files:

- `include/ast.h`
- `src/ast.cc`

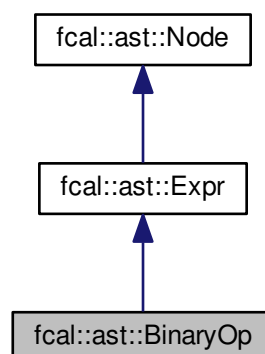
## 4.3 fcal::ast::BinaryOp Class Reference

```
#include <ast.h>
```

Inheritance diagram for `fcal::ast::BinaryOp`:



Collaboration diagram for `fcal::ast::BinaryOp`:



## Public Member Functions

- [BinaryOp](#) ([Expr](#) \*expr, std::string \*op, [Expr](#) \*expr2)
- std::string [unparse](#) ()  
[BinaryOp unparse\(\)](#) method.
- std::string [cpp\\_code](#) ()

### 4.3.1 Detailed Description

The [BinaryOp](#) class inherits directly from the parent [Expr](#) class. The [BinaryOp](#) class combines the redundant nature of the implementing multiple production rule classes for the various binary operators including: \*, /, +, -, >, >=, <, <=, ==, !=, && and ||.

The constructor determines the type of operator associated with expression by defining the \*op to the lexeme of the prev\_token\_ for the matched signed.

### 4.3.2 Constructor & Destructor Documentation

4.3.2.1 `fcgal::ast::BinaryOp::BinaryOp ( Expr * expr, std::string * op, Expr * expr2 ) [inline],[explicit]`

[BinaryOp](#) production rules take the parameters: \*expr, \*op and \*expr2

#### Parameters

*expr	is the LHS expression
*op	is the binary operator
*expr2	is the RHS expression

### 4.3.3 Member Function Documentation

4.3.3.1 `std::string fcgal::ast::BinaryOp::unparse ( ) [virtual]`

[BinaryOp unparse\(\)](#) method.

[BinaryOp](#) returns the expr\_, op\_ and expr2\_.

Implements [fcgal::ast::Node](#).

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

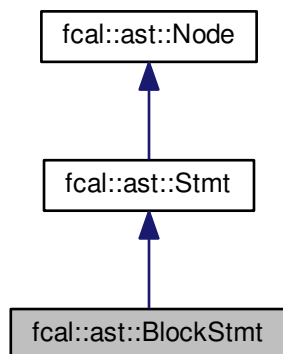
- include/ast.h
- src/ast.cc

## 4.4 fcal::ast::BlockStmt Class Reference

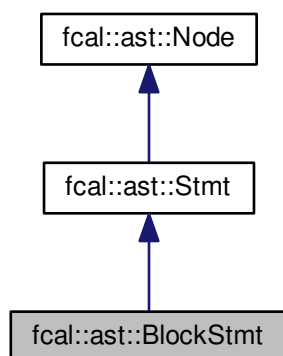
The [BlockStmt](#) class inherits directly from the abstract [Stmt](#) parent class.

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::BlockStmt:



Collaboration diagram for fcal::ast::BlockStmt:



### Public Member Functions

- [BlockStmt](#) ([Stmts](#) \*stmts)
- std::string [unparse](#) ()  
*[BlockStmt unparse\(\)](#) method.*
- std::string **cpp\_code** ()

#### 4.4.1 Detailed Description

The [BlockStmt](#) class inherits directly from the abstract [Stmt](#) parent class.

#### 4.4.2 Constructor & Destructor Documentation

4.4.2.1 `fcgal::ast::BlockStmt::BlockStmt ( Stmt* * stmts ) [inline],[explicit]`

[BlockStmt](#) production class takes a single parameter: stmts

Parameters

<code>*stmts</code>	statements
---------------------	------------

#### 4.4.3 Member Function Documentation

4.4.3.1 `std::string fcgal::ast::BlockStmt::unparse ( ) [virtual]`

[BlockStmt unparse\(\)](#) method.

[BlockStmt unparse\(\)](#) returns the stmts\_ parameter.

Implements [fcgal::ast::Node](#).

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

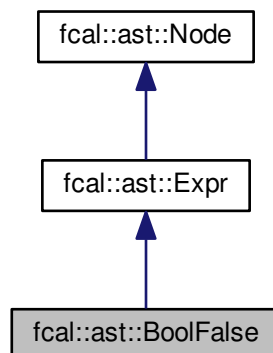
- include/ast.h
- src/ast.cc

### 4.5 fcgal::ast::BoolFalse Class Reference

The [BoolFalse](#) class inherits directly from the abstract [Expr](#) class.

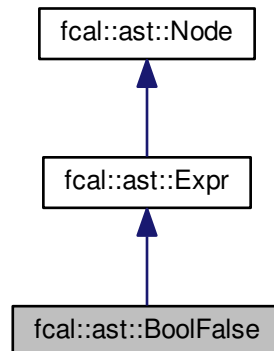
```
#include <ast.h>
```

Inheritance diagram for `fcgal::ast::BoolFalse`:





Collaboration diagram for fcal::ast::BoolFalse:



## Public Member Functions

- [BoolFalse\(\)](#)  
*BoolFalse() constructor.*
- `std::string` [unparse\(\)](#)  
*BoolFalse unparse() method.*
- `std::string` [cpp\\_code\(\)](#)

### 4.5.1 Detailed Description

The [BoolFalse](#) class inherits directly from the abstract [Expr](#) class.

### 4.5.2 Member Function Documentation

#### 4.5.2.1 `std::string fcal::ast::BoolFalse::unparse ( )` [virtual]

[BoolFalse unparse\(\)](#) method.

[BoolFalse](#) returns a "False" string for a boolean false.

Implements [fcal::ast::Node](#).

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

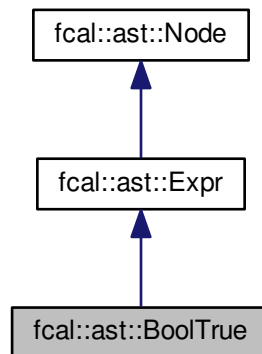
- `include/ast.h`
- `src/ast.cc`

## 4.6 fcal::ast::BoolTrue Class Reference

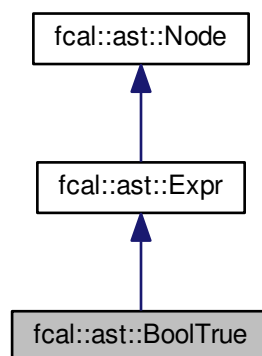
The `BoolTrue` class inherits directly from the abstract `Expr` class.

```
#include <ast.h>
```

Inheritance diagram for `fcal::ast::BoolTrue`:



Collaboration diagram for `fcal::ast::BoolTrue`:



### Public Member Functions

- `BoolTrue ()`  
*`BoolTrue()` constructor.*
- `std::string unparse ()`  
*`BoolTrue unparse()` method.*
- `std::string cpp_code ()`

### 4.6.1 Detailed Description

The [BoolTrue](#) class inherits directly from the abstract [Expr](#) class.

### 4.6.2 Member Function Documentation

#### 4.6.2.1 `std::string fcal::ast::BoolTrue::unparse ( )` `[virtual]`

[BoolTrue unparse\(\)](#) method.

[BoolTrue](#) returns the "True" string for a boolean truth.

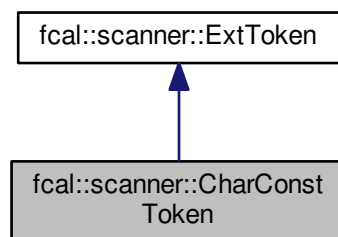
Implements [fcal::ast::Node](#).

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

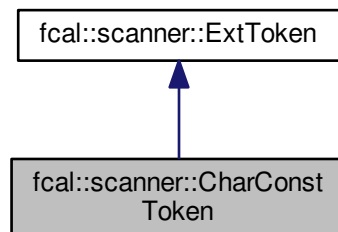
- include/ast.h
- src/ast.cc

## 4.7 fcal::scanner::CharConstToken Class Reference

Inheritance diagram for `fcal::scanner::CharConstToken`:



Collaboration diagram for `fcal::scanner::CharConstToken`:



## Public Member Functions

- **CharConstToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- [parser::ParseResult](#) **nud** ()
- std::string **description** ()

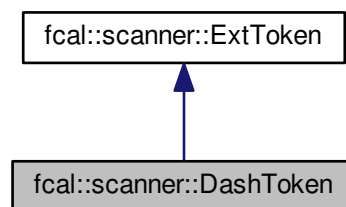
## Additional Inherited Members

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

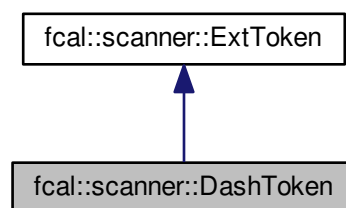
- include/ext\_token.h

## 4.8 fcal::scanner::DashToken Class Reference

Inheritance diagram for fcal::scanner::DashToken:



Collaboration diagram for fcal::scanner::DashToken:



## Public Member Functions

- **DashToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- [parser::ParseResult](#) **led** ([parser::ParseResult](#) left)
- std::string **description** ()
- int **lbp** ()

### Additional Inherited Members

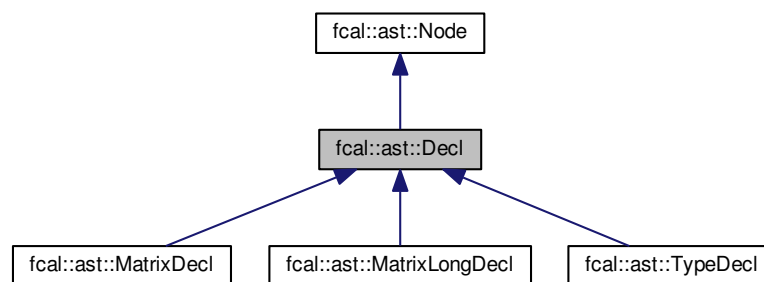
The documentation for this class was generated from the following file:

- include/ext\_token.h

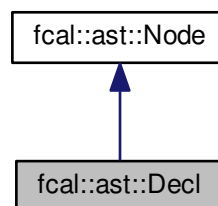
## 4.9 fcal::ast::Decl Class Reference

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::Decl:



Collaboration diagram for fcal::ast::Decl:



### Additional Inherited Members

#### 4.9.1 Detailed Description

This is an abstract [Decl](#) class that inherits directly from the parent [Node](#) class.

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

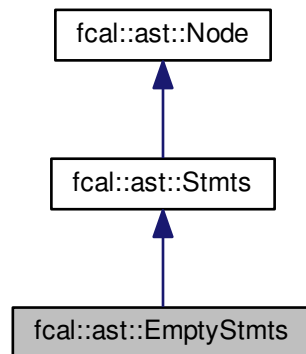
- include/ast.h

## 4.10 fcal::ast::EmptyStmts Class Reference

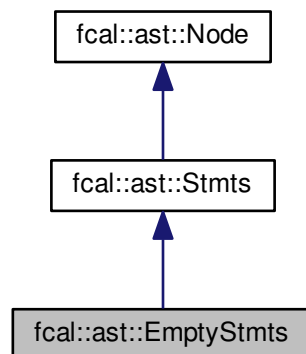
The [EmptyStmts](#) class inherits directly from the abstract [Stmts](#) parent class.

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::EmptyStmts:



Collaboration diagram for fcal::ast::EmptyStmts:



### Public Member Functions

- [EmptyStmts](#) ()  
*EmptyStmts Deconstructor.*
- `std::string` [unparse](#) ()  
*EmptyStmts unparse() method.*
- `std::string` **cpp\_code** ()

### 4.10.1 Detailed Description

The [EmptyStmts](#) class inherits directly from the abstract [Stmts](#) parent class.

### 4.10.2 Member Function Documentation

#### 4.10.2.1 `std::string fcal::ast::EmptyStmts::unparse ( )` `[virtual]`

[EmptyStmts unparse\(\)](#) method.

[EmptyStmts unparse\(\)](#) returns nothing.

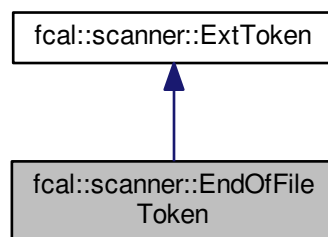
Implements [fcal::ast::Node](#).

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

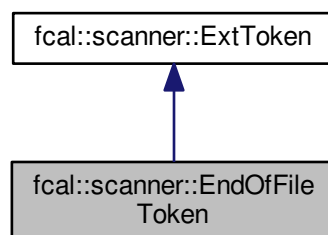
- include/ast.h
- src/ast.cc

## 4.11 fcal::scanner::EndOfFileToken Class Reference

Inheritance diagram for `fcal::scanner::EndOfFileToken`:



Collaboration diagram for `fcal::scanner::EndOfFileToken`:



## Public Member Functions

- **EndOfFileToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- `std::string description` ()

## Additional Inherited Members

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

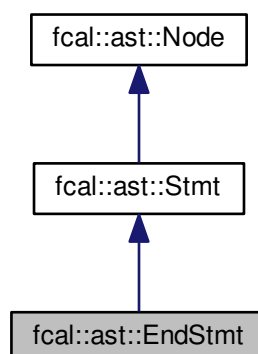
- `include/ext_token.h`

## 4.12 `fcgal::ast::EndStmt` Class Reference

The `EndStmt` class inherits directly from the abstract `Stmt` parent class.

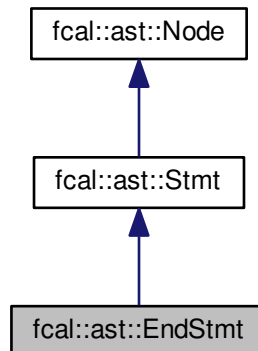
```
#include <ast.h>
```

Inheritance diagram for `fcgal::ast::EndStmt`:





Collaboration diagram for fcal::ast::EndStmt:



## Public Member Functions

- [EndStmt](#) ()  
*EndStmt() constructor.*
- `std::string` [unparse](#) ()  
*EndStmt unparse() method.*
- `std::string` **cpp\_code** ()

### 4.12.1 Detailed Description

The [EndStmt](#) class inherits directly from the abstract [Stmt](#) parent class.

### 4.12.2 Member Function Documentation

#### 4.12.2.1 `std::string fcal::ast::EndStmt::unparse ( )` [virtual]

[EndStmt unparse\(\)](#) method.

[EndStmt](#) returns a semicolon (;) for end of line.

Implements [fcal::ast::Node](#).

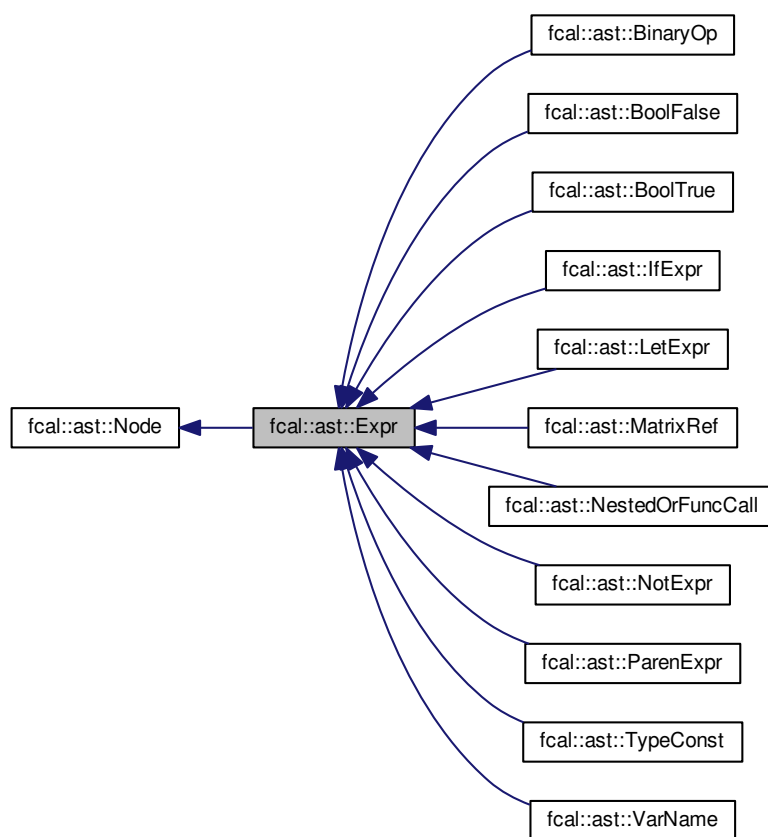
The documentation for this class was generated from the following files:

- `include/ast.h`
- `src/ast.cc`

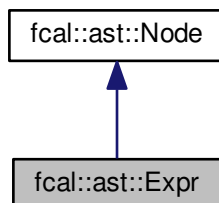
### 4.13 fcal::ast::Expr Class Reference

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::Expr:



Collaboration diagram for fcal::ast::Expr:



## Additional Inherited Members

### 4.13.1 Detailed Description

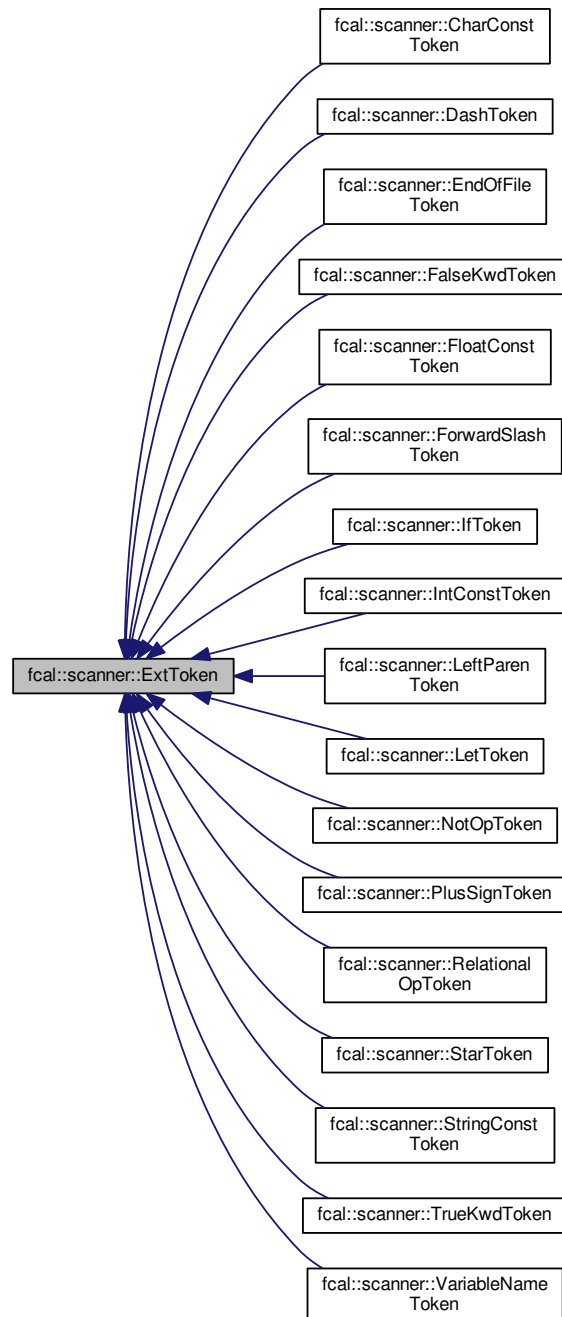
This is an abstract [Expr](#) class that inherits directly from the parent [Node](#) class.

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

- `include/ast.h`

## 4.14 fcal::scanner::ExtToken Class Reference

Inheritance diagram for fcal::scanner::ExtToken:



### Public Member Functions

- **ExtToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- **ExtToken** ([parser::Parser](#) \*p, [Token](#) \*t, std::string d)

- virtual [parser::ParseResult](#) **nud** (void)
- virtual [parser::ParseResult](#) **led** ([parser::ParseResult](#) left)
- [ExtToken](#) \* **ExtendToken** ([parser::Parser](#) \*p, [Token](#) \*tokens)
- [ExtToken](#) \* **ExtendTokenList** ([parser::Parser](#) \*p, [Token](#) \*tokens)
- virtual int **lbp** ()
- virtual std::string **description** ()
- std::string **lexeme** (void) const
- [ExtToken](#) \* **next** (void) const
- scanner::TokenType **terminal** (void) const

### Protected Member Functions

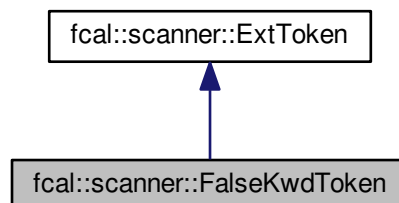
- [parser::Parser](#) \* **parser** (void)

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

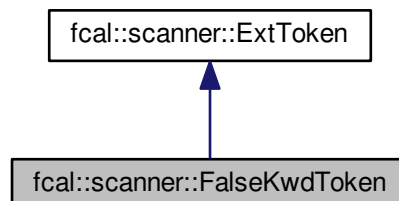
- include/ext\_token.h
- src/ext\_token.cc

## 4.15 fcal::scanner::FalseKwdToken Class Reference

Inheritance diagram for fcal::scanner::FalseKwdToken:



Collaboration diagram for fcal::scanner::FalseKwdToken:



## Public Member Functions

- **FalseKwdToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- [parser::ParseResult](#) **nud** ()
- [std::string](#) **description** ()

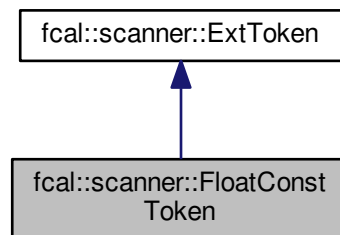
## Additional Inherited Members

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

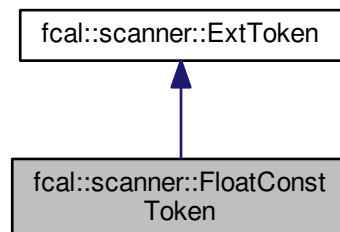
- [include/ext\\_token.h](#)

## 4.16 fcal::scanner::FloatConstToken Class Reference

Inheritance diagram for `fcal::scanner::FloatConstToken`:



Collaboration diagram for `fcal::scanner::FloatConstToken`:



### Public Member Functions

- **FloatConstToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- [parser::ParseResult](#) **nud** ()
- [std::string](#) **description** ()

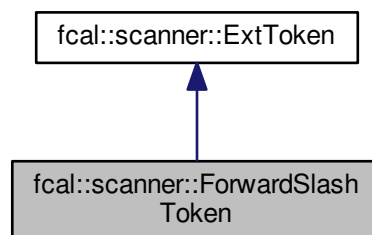
### Additional Inherited Members

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

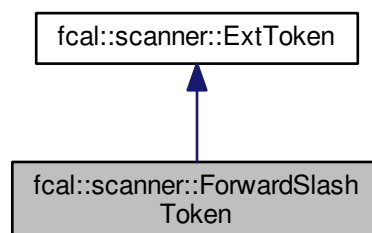
- [include/ext\\_token.h](#)

## 4.17 fcal::scanner::ForwardSlashToken Class Reference

Inheritance diagram for fcal::scanner::ForwardSlashToken:



Collaboration diagram for fcal::scanner::ForwardSlashToken:



## Public Member Functions

- **ForwardSlashToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- [parser::ParseResult](#) **led** ([parser::ParseResult](#) left)
- `std::string` **description** ()
- `int` **lbp** ()

## Additional Inherited Members

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

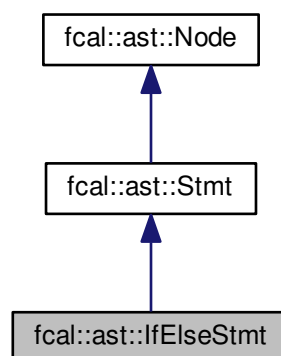
- `include/ext_token.h`

## 4.18 `fcgal::ast::IfElseStmt` Class Reference

The `IfElseStmt` class inherits directly from the abstract `Stmt` parent class.

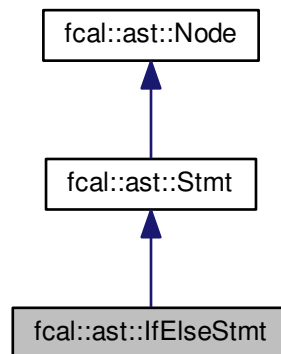
```
#include <ast.h>
```

Inheritance diagram for `fcgal::ast::IfElseStmt`:





Collaboration diagram for fcal::ast::IfElseStmt:



## Public Member Functions

- `IfElseStmt` (`Expr` \*expr, `Stmt` \*stmt, `Stmt` \*stmt2)
- `std::string` `unparse` ()  
*IfElseStmt* `unparse` method()
- `std::string` `cpp_code` ()

### 4.18.1 Detailed Description

The `IfElseStmt` class inherits directly from the abstract `Stmt` parent class.

### 4.18.2 Constructor & Destructor Documentation

4.18.2.1 `fcal::ast::IfElseStmt::IfElseStmt ( Expr * expr, Stmt * stmt, Stmt * stmt2 )` `[inline]`, `[explicit]`

`IfElseStmt` production class takes the parameters: \*expr, \*stmt and \*stmt2

#### Parameters

*expr	expression of the if statement
*stmt	statement of the then clause
*stmt2	statement of the else clause

### 4.18.3 Member Function Documentation

4.18.3.1 `std::string` `fcal::ast::IfElseStmt::unparse ( )` `[virtual]`

`IfElseStmt` `unparse` method()

`IfElseStmt unparsed()` returns the `expr_`, `stmt_` and `stmt2_` parameters.

Implements `fcal::ast::Node`.

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

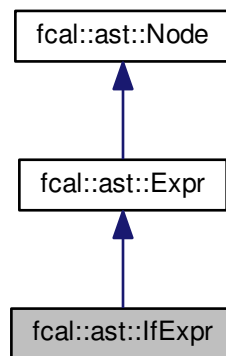
- `include/ast.h`
- `src/ast.cc`

## 4.19 `fcal::ast::IfExpr` Class Reference

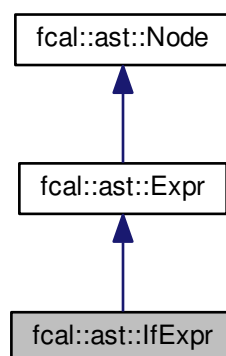
The `IfExpr` class inherits directly from the abstract `Expr` class.

```
#include <ast.h>
```

Inheritance diagram for `fcal::ast::IfExpr`:



Collaboration diagram for `fcal::ast::IfExpr`:



## Public Member Functions

- [IfExpr](#) ([Expr](#) \*expr, [Expr](#) \*expr2, [Expr](#) \*expr3)
- std::string [unparse](#) ()  
*[IfExpr unparse\(\)](#) method.*
- std::string [cpp\\_code](#) ()

### 4.19.1 Detailed Description

The [IfExpr](#) class inherits directly from the abstract [Expr](#) class.

### 4.19.2 Constructor & Destructor Documentation

4.19.2.1 `fcal::ast::IfExpr::IfExpr ( Expr * expr, Expr * expr2, Expr * expr3 )` `[inline], [explicit]`

[IfExpr](#) production rules take the paramters: \*expr, \*expr2 and \*expr3

#### Parameters

* <i>expr</i>	is the if expression
* <i>expr2</i>	is the then expression
* <i>expr3</i>	is the else expression

### 4.19.3 Member Function Documentation

4.19.3.1 `std::string fcal::ast::IfExpr::unparse ( )` `[virtual]`

[IfExpr unparse\(\)](#) method.

[IfExpr](#) returns expr\_, expr2\_ and expr3\_ paramters.

Implements [fcal::ast::Node](#).

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

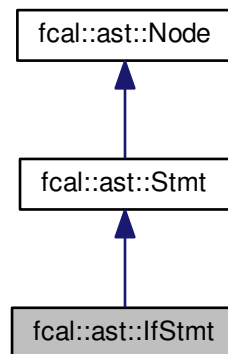
- include/ast.h
- src/ast.cc

## 4.20 fcal::ast::IfStmt Class Reference

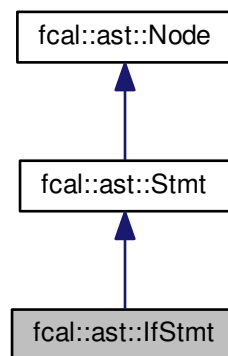
The [IfStmt](#) class inherits directly from the abstract [Stmt](#) parent class.

```
#include <ast.h>
```

Inheritance diagram for `fcal::ast::IfStmt`:



Collaboration diagram for `fcal::ast::IfStmt`:



## Public Member Functions

- `IfStmt` (`Expr` \*expr, `Stmt` \*stmt)
- `std::string unparse ()`  
*`IfStmt unparse()` method.*
- `std::string cpp_code ()`

### 4.20.1 Detailed Description

The `IfStmt` class inherits directly from the abstract `Stmt` parent class.

## 4.20.2 Constructor & Destructor Documentation

4.20.2.1 `fcal::ast::IfStmt::IfStmt ( Expr * expr, Stmt * stmt )` `[inline]`, `[explicit]`

`IfStmt` production class takes the parameters: `*expr` and `*stmt`

Parameters

<code>*<i>expr</i></code>	expression of the if statement
<code>*<i>stmt</i></code>	statement of the then clause

## 4.20.3 Member Function Documentation

4.20.3.1 `std::string fcal::ast::IfStmt::unparse ( )` `[virtual]`

`IfStmt unparse()` method.

`IfStmt unparse()` returns the `expr_` and `stmt_` parameters.

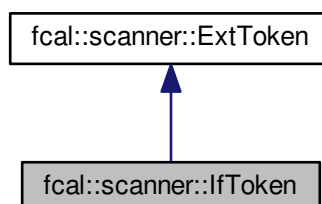
Implements `fcal::ast::Node`.

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

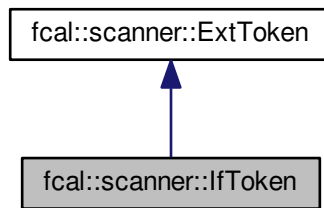
- `include/ast.h`
- `src/ast.cc`

## 4.21 fcal::scanner::IfToken Class Reference

Inheritance diagram for `fcal::scanner::IfToken`:



Collaboration diagram for `fcal::scanner::IfToken`:



### Public Member Functions

- **IfToken** (`parser::Parser *p`, `Token *t`)
- `parser::ParseResult nud` ()
- `std::string description` ()
- `int lbp` ()

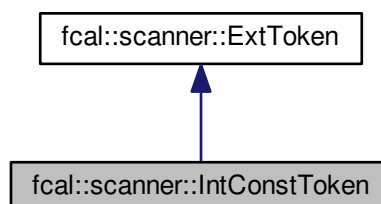
### Additional Inherited Members

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

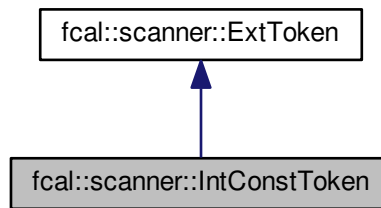
- `include/ext_token.h`

## 4.22 `fcal::scanner::IntConstToken` Class Reference

Inheritance diagram for `fcal::scanner::IntConstToken`:



Collaboration diagram for fcal::scanner::IntConstToken:



### Public Member Functions

- **IntConstToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- [parser::ParseResult](#) **nud** ()
- [std::string](#) **description** ()

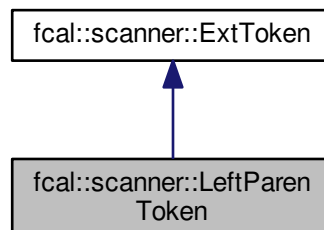
### Additional Inherited Members

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

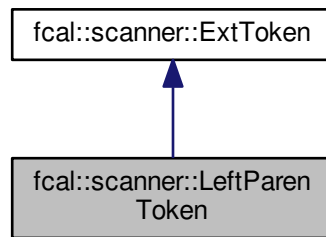
- `include/ext_token.h`

## 4.23 fcal::scanner::LeftParenToken Class Reference

Inheritance diagram for fcal::scanner::LeftParenToken:



Collaboration diagram for `fcal::scanner::LeftParenToken`:



### Public Member Functions

- **LeftParenToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- [parser::ParseResult](#) **nud** ()
- `std::string` **description** ()
- `int` **lbp** ()

### Additional Inherited Members

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

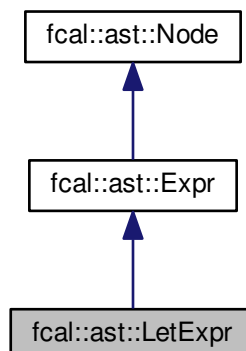
- `include/ext_token.h`

## 4.24 fcal::ast::LetExpr Class Reference

The [LetExpr](#) class inherits directly from the abstract [Expr](#) class.

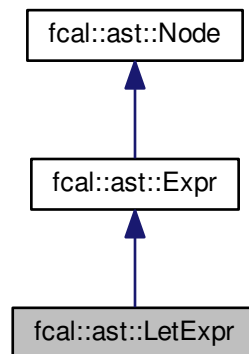
```
#include <ast.h>
```

Inheritance diagram for `fcal::ast::LetExpr`:





Collaboration diagram for fcal::ast::LetExpr:



## Public Member Functions

- [LetExpr](#) ([Stmts](#) \*stmts, [Expr](#) \*expr)
- `std::string` [unparse](#) ()  
[LetExpr unparse\(\)](#) method.
- `std::string` [cpp\\_code](#) ()

### 4.24.1 Detailed Description

The [LetExpr](#) class inherits directly from the abstract [Expr](#) class.

### 4.24.2 Constructor & Destructor Documentation

4.24.2.1 `fcal::ast::LetExpr::LetExpr ( Stmts * stmts, Expr * expr )` `[inline]`, `[explicit]`

[LetExpr](#) production rules take the parameters: \*stmts and \*expr

#### Parameters

*stmts	are the statements between let and in
*expr	is the expression after in and before end

### 4.24.3 Member Function Documentation

4.24.3.1 `std::string` `fcal::ast::LetExpr::unparse ( )` `[virtual]`

[LetExpr unparse\(\)](#) method.

[LetExpr](#) returns the `stmts_` and `expr_` parameters.

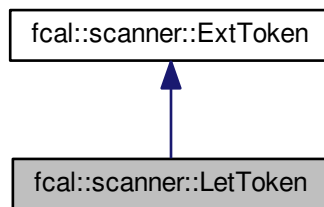
Implements [fcal::ast::Node](#).

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

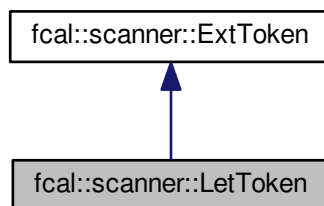
- `include/ast.h`
- `src/ast.cc`

## 4.25 fcal::scanner::LetToken Class Reference

Inheritance diagram for `fcal::scanner::LetToken`:



Collaboration diagram for `fcal::scanner::LetToken`:



### Public Member Functions

- **LetToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- [parser::ParseResult](#) **nud** ()
- `std::string` **description** ()
- `int` **lbp** ()

### Additional Inherited Members

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

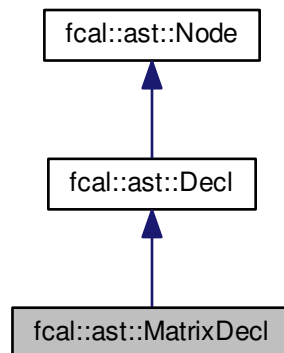
- include/ext\_token.h

## 4.26 fcal::ast::MatrixDecl Class Reference

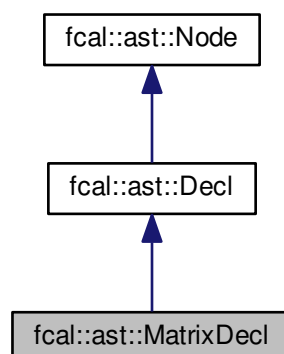
The [MatrixDecl](#) class inherits directly from the abstract [Decl](#) class.

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::MatrixDecl:



Collaboration diagram for fcal::ast::MatrixDecl:



## Public Member Functions

- [MatrixDecl](#) ([VarName](#) \*var\_name, [Expr](#) \*expr)
- std::string [unparse](#) ()  
[MatrixDecl unparse\(\)](#) method.
- std::string [cpp\\_code](#) ()

### 4.26.1 Detailed Description

The [MatrixDecl](#) class inherits directly from the abstract [Decl](#) class.

### 4.26.2 Constructor & Destructor Documentation

4.26.2.1 `fcgal::ast::MatrixDecl::MatrixDecl ( VarName * var_name, Expr * expr )` `[inline]`, `[explicit]`

[MatrixDecl](#) production class takes the parameters: \*var\_name and \*expr

#### Parameters

*var_name	is the name of the variable being assigned
*expr	is the expression being assigned to the variable

### 4.26.3 Member Function Documentation

4.26.3.1 `std::string fcgal::ast::MatrixDecl::unparse ( )` `[virtual]`

[MatrixDecl unparse\(\)](#) method.

[MatrixDecl](#) returns var\_name\_ and expr\_.

Implements [fcgal::ast::Node](#).

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

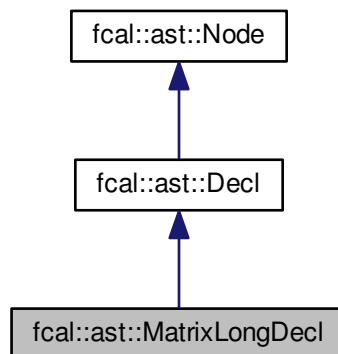
- include/ast.h
- src/ast.cc

## 4.27 fcgal::ast::MatrixLongDecl Class Reference

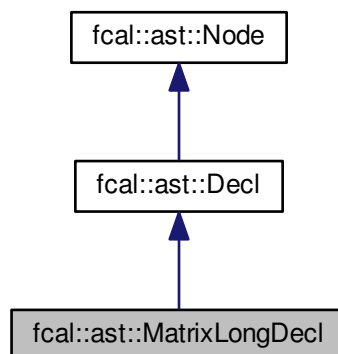
The [MatrixLongDecl](#) class inherits directly from the abstract [Decl](#) class.

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::MatrixLongDecl:



Collaboration diagram for fcal::ast::MatrixLongDecl:



### Public Member Functions

- [MatrixLongDecl](#) ([VarName](#) \*var\_name, [Expr](#) \*expr, [Expr](#) \*expr2, [VarName](#) \*var\_name2, [VarName](#) \*var\_name3, [Expr](#) \*expr3)
- [std::string](#) [unparse](#) ()  
[MatrixLongDecl unparse\(\)](#) method.
- [std::string](#) [cpp\\_code](#) ()

#### 4.27.1 Detailed Description

The [MatrixLongDecl](#) class inherits directly from the abstract [Decl](#) class.

## 4.27.2 Constructor & Destructor Documentation

4.27.2.1 `fcgal::ast::MatrixLongDecl::MatrixLongDecl ( VarName * var_name, Expr * expr, Expr * expr2, VarName * var_name2, VarName * var_name3, Expr * expr3 ) [inline],[explicit]`

MatrixLogDecl production class takes the parameters: \*var\_name, \*expr, expr2, \*var\_name2, \*var\_name3, and \*expr3

### Parameters

*var_name	names the variable referencing the matrix
*expr	first parameter of the matrix
*expr2	second parameter of the matrix
*var_name2	first variable reference
*var_name3	secondary variable reference
*expr3	expression being assigned

## 4.27.3 Member Function Documentation

4.27.3.1 `std::string fcal::ast::MatrixLongDecl::unparse ( ) [virtual]`

[MatrixLongDecl unparse\(\)](#) method.

[MatrixLongDecl](#) returns var\_name\_, expr\_, expr2\_, var\_name2\_, var\_name3\_, and expr3\_

Implements [fcgal::ast::Node](#).

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

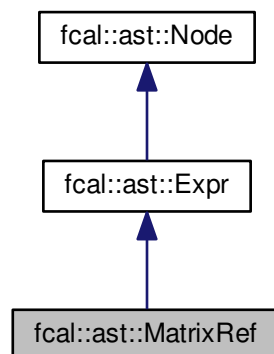
- include/ast.h
- src/ast.cc

## 4.28 fcal::ast::MatrixRef Class Reference

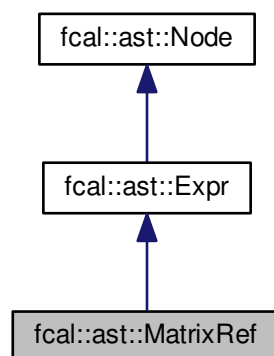
The [MatrixRef](#) class inherits directly from the abstract [Expr](#) class.

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::MatrixRef:



Collaboration diagram for fcal::ast::MatrixRef:



### Public Member Functions

- [MatrixRef](#) ([VarName](#) \*var\_name, [Expr](#) \*expr, [Expr](#) \*expr2)
- [std::string](#) [unparse](#) ()  
*[MatrixRef unparse\(\)](#) method.*
- [std::string](#) [cpp\\_code](#) ()

#### 4.28.1 Detailed Description

The [MatrixRef](#) class inherits directly from the abstract [Expr](#) class.

## 4.28.2 Constructor & Destructor Documentation

4.28.2.1 `fcfcal::ast::MatrixRef::MatrixRef ( VarName * var_name, Expr * expr, Expr * expr2 ) [inline], [explicit]`

[MatrixRef](#) production rules take the parameters: \*var\_name, \*expr, and expr2

### Parameters

*var_name	is the name of the matrix reference
*expr	is the first parameter
*expr2	is the second parameter

## 4.28.3 Member Function Documentation

4.28.3.1 `std::string fcfcal::ast::MatrixRef::unparse ( ) [virtual]`

[MatrixRef unparse\(\)](#) method.

[MatrixRef](#) returns the var\_name\_, expr\_ and expr2\_ parameters.

Implements [fcfcal::ast::Node](#).

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

- include/ast.h
- src/ast.cc

## 4.29 MySequence< T, N > Class Template Reference

### Public Member Functions

- void **set\_member** (int x, T value)
- T **get\_member** (int x)

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

- src/templates.cc

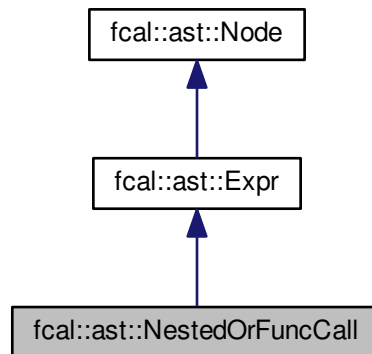


## 4.30 fcal::ast::NestedOrFuncCall Class Reference

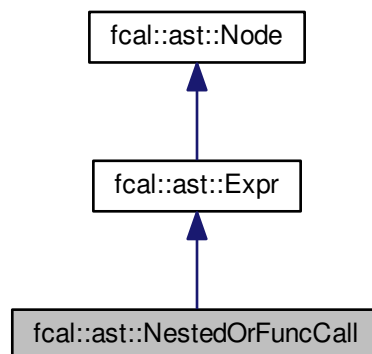
The [NestedOrFuncCall](#) class inherits directly from the abstract [Expr](#) class.

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::NestedOrFuncCall:



Collaboration diagram for fcal::ast::NestedOrFuncCall:



### Public Member Functions

- [NestedOrFuncCall](#) ([VarName](#) \*var\_name, [Expr](#) \*expr)
- `std::string unparse ()`  
*[NestedOrFuncCall](#) [unparse\(\)](#) method.*
- `std::string cpp\_code ()`

### 4.30.1 Detailed Description

The [NestedOrFuncCall](#) class inherits directly from the abstract [Expr](#) class.

### 4.30.2 Constructor & Destructor Documentation

4.30.2.1 `fcgal::ast::NestedOrFuncCall::NestedOrFuncCall ( VarName * var_name, Expr * expr ) [inline],  
[explicit]`

[NestedOrFuncCall](#) production rules take the parameters: *\*var\_name* and *\*expr*

Parameters

<i>*var_name</i>	is the variable name
<i>*expr</i>	is the nested expression

### 4.30.3 Member Function Documentation

4.30.3.1 `std::string fcal::ast::NestedOrFuncCall::unparse ( ) [virtual]`

[NestedOrFuncCall](#) `unparse()` method.

[NestedOrFuncCall](#) returns the *var\_name\_* and *expr\_* paramters.

Implements [fcgal::ast::Node](#).

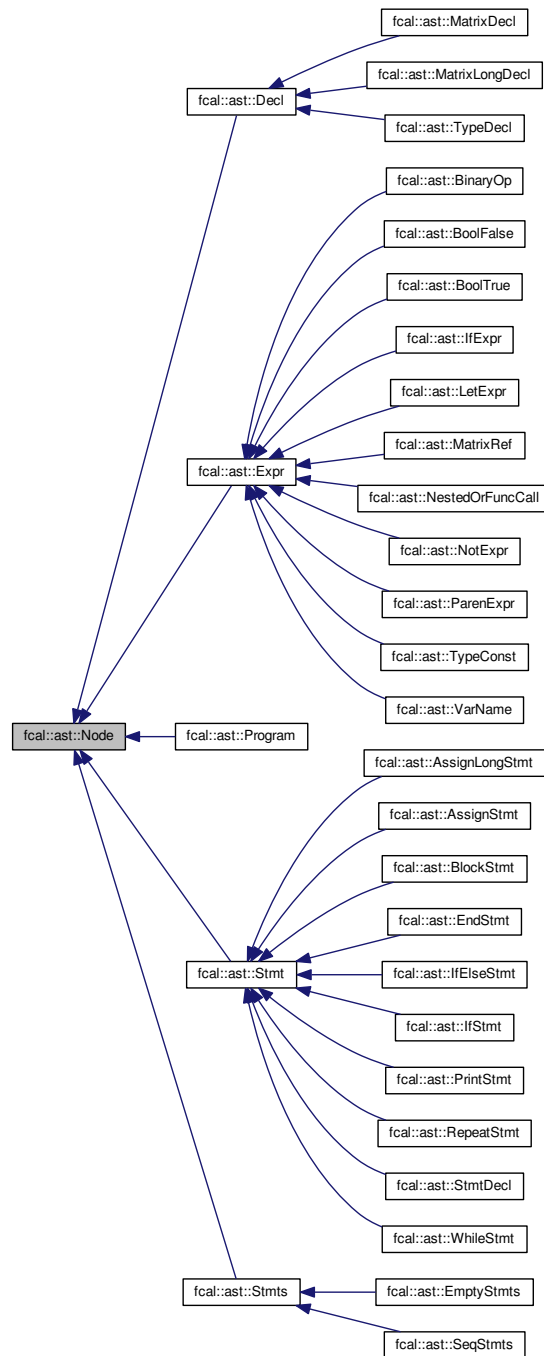
The documentation for this class was generated from the following files:

- `include/ast.h`
- `src/ast.cc`

## 4.31 fcal::ast::Node Class Reference

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::Node:



## Public Member Functions

- virtual std::string [unparse](#) ()=0  
virtual [unparse\(\)](#) method
- virtual [~Node](#) ()  
*Node()* destructor.

### 4.31.1 Detailed Description

The abstract [Node](#) base class is the parent to all classes within the production rules. All further classes will inherit the [unparse\(\)](#) function.

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

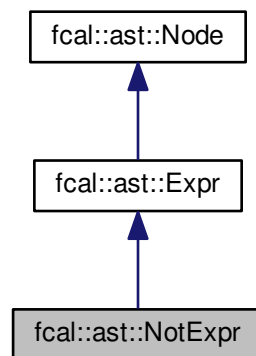
- include/ast.h

### 4.32 fcal::ast::NotExpr Class Reference

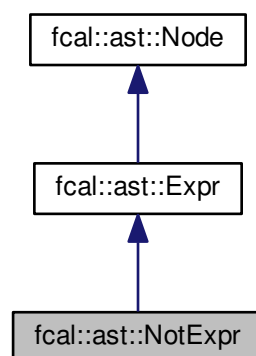
The [NotExpr](#) class inherits directly from the abstract [Expr](#) class.

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::NotExpr:



Collaboration diagram for fcal::ast::NotExpr:



## Public Member Functions

- [NotExpr](#) ([Expr](#) \**expr*)
- `std::string` [unparse](#) ()  
*NotExpr unparse()* method.
- `std::string` [cpp\\_code](#) ()

### 4.32.1 Detailed Description

The [NotExpr](#) class inherits directly from the abstract [Expr](#) class.

### 4.32.2 Constructor & Destructor Documentation

4.32.2.1 `fcal::ast::NotExpr::NotExpr ( Expr * expr )` `[inline]`, `[explicit]`

[NotExpr](#) production rules take the parameter: \**expr*

#### Parameters

* <i>expr</i>	is the expression being negated
---------------	---------------------------------

### 4.32.3 Member Function Documentation

4.32.3.1 `std::string` `fcal::ast::NotExpr::unparse ( )` `[virtual]`

[NotExpr unparse\(\)](#) method.

[NotExpr](#) returns a negated `expr_` parameter.

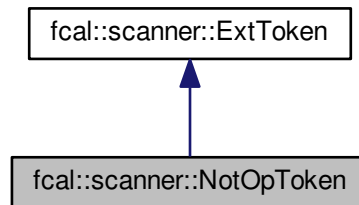
Implements [fcal::ast::Node](#).

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

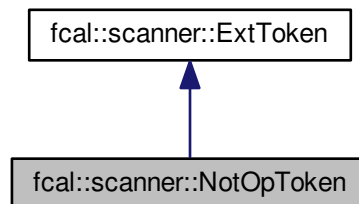
- `include/ast.h`
- `src/ast.cc`

### 4.33 fcal::scanner::NotOpToken Class Reference

Inheritance diagram for fcal::scanner::NotOpToken:



Collaboration diagram for fcal::scanner::NotOpToken:



#### Public Member Functions

- **NotOpToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- [parser::ParseResult](#) **nud** ()
- std::string **description** ()

#### Additional Inherited Members

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

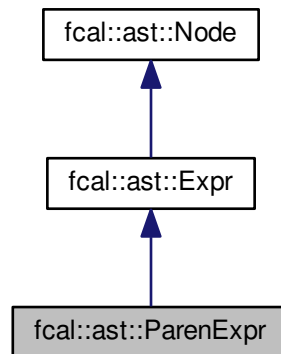
- include/ext\_token.h

## 4.34 fcal::ast::ParenExpr Class Reference

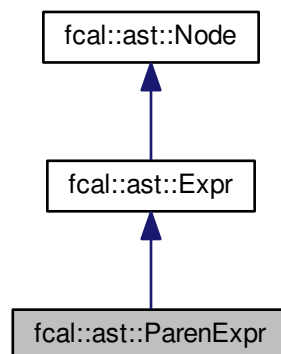
The [ParenExpr](#) class inherits directly from the abstract [Expr](#) class.

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::ParenExpr:



Collaboration diagram for fcal::ast::ParenExpr:



### Public Member Functions

- [ParenExpr](#) ([Expr](#) \*expr)
- std::string [unparse](#) ()  
*[ParenExpr unparse\(\)](#) method.*
- std::string [cpp\\_code](#) ()

### 4.34.1 Detailed Description

The [ParenExpr](#) class inherits directly from the abstract [Expr](#) class.

### 4.34.2 Constructor & Destructor Documentation

4.34.2.1 `fcgal::ast::ParenExpr::ParenExpr ( Expr * expr ) [inline],[explicit]`

[ParenExpr](#) production rules take a single parameter: `*expr`

Parameters

<code>*<i>expr</i></code>	is expression nested between parantheses
---------------------------	--

### 4.34.3 Member Function Documentation

4.34.3.1 `std::string fcal::ast::ParenExpr::unparse ( ) [virtual]`

[ParenExpr](#) `unparse()` method.

[ParenExpr](#) returns the `expr_` paramter.

Implements [fcgal::ast::Node](#).

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

- `include/ast.h`
- `src/ast.cc`

## 4.35 fcal::parser::Parser Class Reference

### Public Member Functions

- [~Parser](#) (void)  
*Parser* destructor function.
- [ParseResult Parse](#) (const char \*text)  
*Parse* constructor function.
- [ParseResult ParseProgram](#) ()  
*ParseProgram* creates the first node in the AST, the Program Node.
- [ParseResult parse\\_decl](#) ()
- [ParseResult parse\\_standard\\_decl](#) ()
- [ParseResult parse\\_matrix\\_decl](#) ()
- [ParseResult parse\\_stmts](#) ()
- [ParseResult parse\\_stmt](#) ()
- [ParseResult parse\\_expr](#) (int rbp)
- [ParseResult parse\\_true\\_kwd](#) ()



- [ParseResult parse\\_false\\_kwd \(\)](#)
- [ParseResult parse\\_int\\_const \(\)](#)
- [ParseResult parse\\_float\\_const \(\)](#)
- [ParseResult parse\\_string\\_const \(\)](#)
- [ParseResult parse\\_char\\_const \(\)](#)
- [ParseResult parse\\_variable\\_name \(\)](#)
- [ParseResult parse\\_nested\\_expr \(\)](#)
- [ParseResult parse\\_not\\_expr \(\)](#)
- [ParseResult parse\\_let\\_expr \(\)](#)
- [ParseResult parse\\_if\\_expr \(\)](#)
- [ParseResult parse\\_addition \(ParseResult left\)](#)
- [ParseResult parse\\_multiplication \(ParseResult left\)](#)
- [ParseResult parse\\_subtraction \(ParseResult left\)](#)
- [ParseResult parse\\_division \(ParseResult left\)](#)
- [ParseResult parse\\_relational\\_expr \(ParseResult left\)](#)
- void **match** (const scanner::TokenType &tt)
- bool **attempt\_match** (const scanner::TokenType &tt)
- bool **next\_is** (const scanner::TokenType &tt)
- void **next\_token** (void)

#### 4.35.1 Member Function Documentation

##### 4.35.1.1 ParseResult fcal::parser::Parser::parse\_addition ( ParseResult *prLeft* )

parse\_addition will generate a BinaryOp with parameters expr, "+", and expr2

##### 4.35.1.2 ParseResult fcal::parser::Parser::parse\_decl ( )

parse\_decl will categorize what type of declaration using the current Token types to either parse\_matrix\_decl or parse\_standard\_decl

##### 4.35.1.3 ParseResult fcal::parser::Parser::parse\_division ( ParseResult *prLeft* )

parse\_division will generate a BinaryOp with parameters expr, "/", and expr2

##### 4.35.1.4 ParseResult fcal::parser::Parser::parse\_false\_kwd ( )

parser\_false\_kwd identifies the current node's Token type and if it is kFalseKwd then it generates a BoolFalse subclass

##### 4.35.1.5 ParseResult fcal::parser::Parser::parse\_float\_const ( )

parse\_float\_const identifies the current node's Token type and if it is kFloatConst then it generates a TypeConst subclass and passes in a "float" lexeme with it

#### 4.35.1.6 **ParseResult** fcal::parser::Parser::parse\_if\_expr ( )

parse\_if\_expr will generate an IfExpr subclass with parameters expr, expr2, and expr3

#### 4.35.1.7 **ParseResult** fcal::parser::Parser::parse\_int\_const ( )

parse\_int\_const identifies the current node's Token type and if it is kIntConst then it generates a TypeConst subclass and passes in a "int" lexeme with it

#### 4.35.1.8 **ParseResult** fcal::parser::Parser::parse\_let\_expr ( )

parse\_let\_expr will generate a LetExpr with parameters stmts and expr

#### 4.35.1.9 **ParseResult** fcal::parser::Parser::parse\_matrix\_decl ( )

parse\_matrix\_decl parses a matrix declaration. If the second token is a left square bracker then it will parse according to the MatrixLongDecl, but there is not left square bracket then it will parse according to the regular MatrixDecl

#### 4.35.1.10 **ParseResult** fcal::parser::Parser::parse\_multiplication ( **ParseResult** *prLeft* )

parse\_multiplication will generate a BinaryOp with parameters expr, "\*", and expr2

#### 4.35.1.11 **ParseResult** fcal::parser::Parser::parse\_nested\_expr ( )

parse\_nested\_expr will generate a ParenExpr subclass with parameter expr

#### 4.35.1.12 **ParseResult** fcal::parser::Parser::parse\_not\_expr ( )

parse\_not\_expr will generate a NotExpr with parameter expr

#### 4.35.1.13 **ParseResult** fcal::parser::Parser::parse\_relational\_expr ( **ParseResult** *prLeft* )

parse\_relational\_expr will generate a BinaryOp with expr, whichever relational expression, and expr2

#### 4.35.1.14 **ParseResult** fcal::parser::Parser::parse\_standard\_decl ( )

parse\_standard\_decl parses a type declaration. The decl\_type will be passed to the general TypeDecl subclass and the decl\_type will be placed in front of the varName ensuring correct parsing

#### 4.35.1.15 ParseResult fcal::parser::Parser::parse\_stmt ( )

parse\_stmt will categorize the type of statement by identifying the keyword and will create the according subclass for it. If the current token is a keyword associated with declarations; kIntKwd, kFloatKwd, etc. it will create a StmtDecl subclass. If the current token is the keyword kLeftCurly then a BlockStmt subclass will be created. If the current token is the keyword kIfKwd then a IfStmt subclass will be created, but if there is a token after that is kElseKwd then the subclass IfElseStmt will be created instead. If the current token is the keyword kVariableName and the next token is of type kLeftSquare then a AssignLongStmt subclass will be created, but if then tokens are just kVariableName and kAssign then an AssignStmt will be created. If the current token is the keyword kPrintKwd then a PrintStmt will be created. If the current token is the keyword kRepeatKwd then a RepeatStmt will be created. If the current token is the keyword kWhileKwd then a WhileStmt will be created. If the current token is the keyword kSemiColon then an EndStmt will be created. If there is current token then throw an error message

#### 4.35.1.16 ParseResult fcal::parser::Parser::parse\_stmts ( )

parse\_stmts will parse EmptyStmts if it is the last Node of the AST, but if the next Node in the AST is neither a kRightCurly or a kInKwd then it will continue parsing with SeqStmts

#### 4.35.1.17 ParseResult fcal::parser::Parser::parse\_string\_const ( )

parse\_string\_const identifies the current node's Token type and if it is kStringConst then it generates a TypeConst subclass and passes in a "string" lexeme with it

#### 4.35.1.18 ParseResult fcal::parser::Parser::parse\_subtraction ( ParseResult prLeft )

parse\_subtraction will generate a BinaryOp with parameters expr, "-", and expr2

#### 4.35.1.19 ParseResult fcal::parser::Parser::parse\_true\_kwd ( )

parse\_true\_kwd identifies the current node's Token type and if it is kTrueKwd then it generates a BoolTrue subclass

#### 4.35.1.20 ParseResult fcal::parser::Parser::parse\_variable\_name ( )

parse\_variable\_name identifies the current token's type and creates a subclass according to it. If the current token is the keyword kLeftSquare then a MatrixRef subclass will be created. If the current token is the keyword kLeftParen then a NestedOrFuncCall will be created. Else if the current token matches none of these then it creates a VarName subclass

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

- include/parser.h
- src/parser.cc

## 4.36 fcal::parser::ParseResult Class Reference

### Public Member Functions

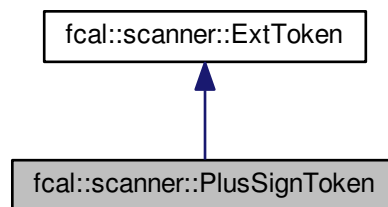
- bool **ok** (void) const
- void **ok** (bool result\_in)
- std::string **errors** (void) const
- void **errors** (const std::string str\_in)
- [ast::Node](#) \* **ast** (void)
- void **ast** ([ast::Node](#) \*Node\_ptr)

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

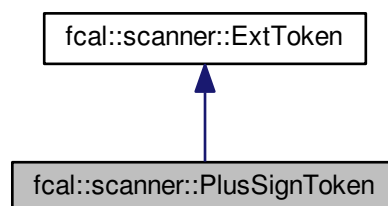
- include/parse\_result.h

## 4.37 fcal::scanner::PlusSignToken Class Reference

Inheritance diagram for fcal::scanner::PlusSignToken:



Collaboration diagram for fcal::scanner::PlusSignToken:



## Public Member Functions

- **PlusSignToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- [parser::ParseResult](#) **led** ([parser::ParseResult](#) left)
- std::string **description** ()
- int **lbp** ()

## Additional Inherited Members

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

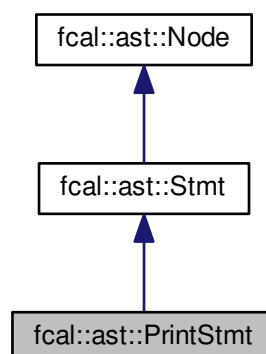
- include/ext\_token.h

## 4.38 fcal::ast::PrintStmt Class Reference

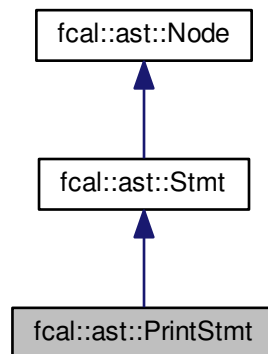
The [PrintStmt](#) class inherits directly from the abstract [Stmt](#) parent class.

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::PrintStmt:



Collaboration diagram for `fcal::ast::PrintStmt`:



## Public Member Functions

- [PrintStmt](#) ([Expr](#) \**expr*)
- `std::string` [unparse](#) ()  
[PrintStmt unparse\(\)](#) method.
- `std::string` [cpp\\_code](#) ()

### 4.38.1 Detailed Description

The [PrintStmt](#) class inherits directly from the abstract [Stmt](#) parent class.

### 4.38.2 Constructor & Destructor Documentation

**4.38.2.1** `fcal::ast::PrintStmt::PrintStmt ( Expr * expr )` `[inline]`, `[explicit]`

[PrintStmt](#) production class takes a single parameter: \**expr*

#### Parameters

<i>*expr</i>	parameter of the printing expression
--------------	--------------------------------------

### 4.38.3 Member Function Documentation

**4.38.3.1** `std::string` `fcal::ast::PrintStmt::unparse ( )` `[virtual]`

[PrintStmt unparse\(\)](#) method.

[PrintStmt unparsed\(\)](#) returns the `expr_` parameter.

Implements [fcal::ast::Node](#).

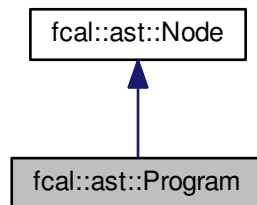
The documentation for this class was generated from the following files:

- `include/ast.h`
- `src/ast.cc`

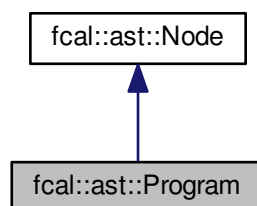
## 4.39 fcal::ast::Program Class Reference

```
#include <ast.h>
```

Inheritance diagram for `fcal::ast::Program`:



Collaboration diagram for `fcal::ast::Program`:



### Public Member Functions

- [Program](#) (`VarName *v`, `Stmts *s`)
- `std::string unparsed ()`  
*Program unparsed() method.*
- `std::string cpp_code ()`
- `virtual ~Program ()`  
*Program() destructor.*

### 4.39.1 Detailed Description

The [Program](#) class, otherwise known as the Root class, inherits directly from the abstract parent [Node](#) class. The [Program](#) class starts the production rules to build the AST.

### 4.39.2 Constructor & Destructor Documentation

4.39.2.1 `fcgal::ast::Program::Program ( VarName * v, Stmts * s ) [inline],[explicit]`

[Program](#) production class takes two parameters: \*v and \*s

Parameters

*v	the name of the program
*s	statements on the RHS of the tree

4.39.2.2 `fcgal::ast::Program::~~Program ( ) [virtual]`

[Program](#)() destructor.

[Program](#) destructor method.

### 4.39.3 Member Function Documentation

4.39.3.1 `std::string fcgal::ast::Program::unparse ( ) [virtual]`

[Program unparse\(\)](#) method.

[Program unparse\(\)](#) returns the var\_name\_ and stmts\_ parameters.

Implements [fcgal::ast::Node](#).

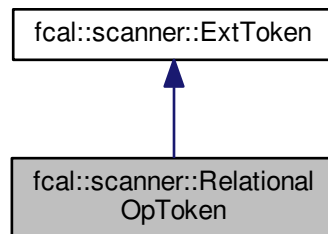
The documentation for this class was generated from the following files:

- include/ast.h
- src/ast.cc

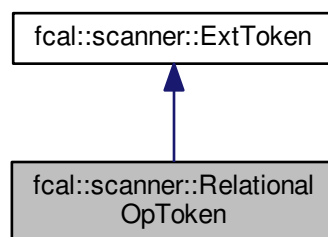


## 4.40 fcal::scanner::RelationalOpToken Class Reference

Inheritance diagram for fcal::scanner::RelationalOpToken:



Collaboration diagram for fcal::scanner::RelationalOpToken:



### Public Member Functions

- **RelationalOpToken** ([parser::Parser](#) \*p, [Token](#) \*t, std::string d)
- [parser::ParseResult](#) **led** ([parser::ParseResult](#) left)
- int **lbp** ()

### Additional Inherited Members

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

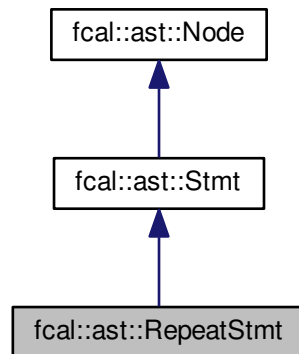
- include/ext\_token.h

## 4.41 fcal::ast::RepeatStmt Class Reference

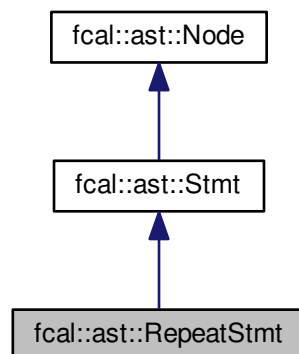
The [RepeatStmt](#) class inherits directly from the abstract [Stmt](#) parent class.

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::RepeatStmt:



Collaboration diagram for fcal::ast::RepeatStmt:



### Public Member Functions

- [RepeatStmt](#) ([VarName](#) \*var\_name, [Expr](#) \*expr, [Expr](#) \*expr2, [Stmt](#) \*stmt)
- std::string [unparse](#) ()  
     [RepeatStmt unparse\(\)](#) method.
- std::string [cpp\\_code](#) ()

### 4.41.1 Detailed Description

The [RepeatStmt](#) class inherits directly from the abstract [Stmt](#) parent class.

### 4.41.2 Constructor & Destructor Documentation

4.41.2.1 `fcal::ast::RepeatStmt::RepeatStmt ( VarName * var_name, Expr * expr, Expr * expr2, Stmt * stmt )`  
`[inline], [explicit]`

[RepeatStmt](#) production class takes the parameters: \*var\_name, \*expr, expr2, and \*stmt

#### Parameters

*var_name	is the name of the variable being assigned
*expr	is the start parameter
*expr2	is the end parameter
*stmt	is the statement being repeated

### 4.41.3 Member Function Documentation

4.41.3.1 `std::string fcal::ast::RepeatStmt::unparse ( )` `[virtual]`

[RepeatStmt unparse\(\)](#) method.

[RepeatStmt](#) returns var\_name\_, expr\_, expr2\_ and stmt\_.

Implements [fcal::ast::Node](#).

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

- include/ast.h
- src/ast.cc

## 4.42 fcal::scanner::Scanner Class Reference

```
#include <scanner.h>
```

### Public Member Functions

- [Scanner](#) ()  
[Scanner\(\)](#) constructor.
- [~Scanner](#) ()  
[Scanner\(\)](#) destructor.
- void [InitRegexTokenArray](#) ()  
Initializes an array of regular expressions associated with its token type.
- [Token \\* Scan](#) (const char \*text)  
Scan muethod that reads through a file to determine token type matches.
- int [consume\\_whitespace\\_and\\_comments](#) (regex\_t \*white\_space, regex\_t \*block\_comment, regex\_t \*single\_comment, const char \*text)

## Public Attributes

- `regex_t * regex_token_array` [45]

### 4.42.1 Detailed Description

The [Scanner](#) class defines the various methods and attributes associated with scanning in a file to determine the token types.

### 4.42.2 Constructor & Destructor Documentation

#### 4.42.2.1 `fcsl::scanner::Scanner::Scanner ( )`

[Scanner\(\)](#) constructor.

[Scanner\(\)](#) constructor; calls to `InitRegexTokenArray` to initialize array.

### 4.42.3 Member Function Documentation

#### 4.42.3.1 `int fcsl::scanner::Scanner::consume_whitespace_and_comments ( regex_t * white_space, regex_t * block_comment, regex_t * single_comment, const char * text )`

The `consume_whitespace_and_comments` method scans through a file or string and looks for comment lines, block comments and white spaces and removes them from consumption so that the `Scan` method can pass through the file without mistaking one of them for a token type.

#### 4.42.3.2 `void fcsl::scanner::Scanner::InitRegexTokenArray ( )`

Initializes an array of regular expressions associated with its token type.

`InitRegexTokenArray` creates an array of regular expressions that matches to the token types, and the array is indexed based on the enum `kTokenEnumType` variable names.

#### 4.42.3.3 `Token * fcsl::scanner::Scanner::Scan ( const char * text )`

`Scan` method that reads through a file to determine token type matches.

The `Scan` method reads in a file or string, but if the file or string is determined to be `NULL`, then the `Scan` method will return `NULL` and terminate. If the `Scan` continues then it will check for an EOF character before continuing to scan in all the characters. As the `Scan` method scans the file, it will iterate through the `InitRegexTokenArray` to determine possible matches to the string; as it finds a given match, it stores the longest `max_num_matched_chars` and once it's done iterating through the string, it will return the token `match_type`. As it returns the `match_type` it will push the determined string of characters to an array list of tokens, which stores the token type, string and a pointer to the next token on an array. Once the scan reaches an EOF character, it will return the array list of tokens.

## Parameters

<i>*text</i>	is a string or file that is read by the scanner
--------------	---

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

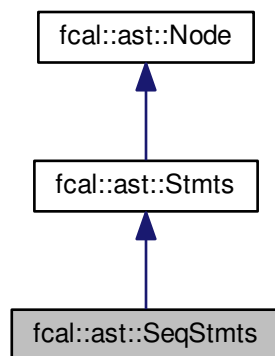
- include/scanner.h
- src/scanner.cc

## 4.43 fcal::ast::SeqStmts Class Reference

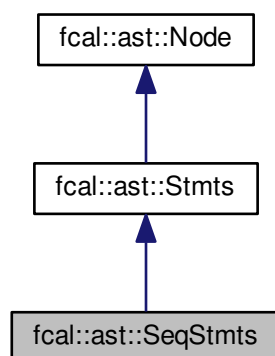
The [SeqStmts](#) class inherits directly from the abstract [Stmts](#) parent class.

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::SeqStmts:



Collaboration diagram for fcal::ast::SeqStmts:



## Public Member Functions

- [SeqStmts](#) ([Stmt](#) \*stmt, [Stmts](#) \*stmts)
- `std::string` [unparse](#) ()  
[SeqStmts](#) *unparse()* method.
- `std::string` [cpp\\_code](#) ()

### 4.43.1 Detailed Description

The [SeqStmts](#) class inherits directly from the abstract [Stmts](#) parent class.

### 4.43.2 Constructor & Destructor Documentation

4.43.2.1 `fcgal::ast::SeqStmts::SeqStmts ( Stmt * stmt, Stmts * stmts )` `[inline]`, `[explicit]`

[SeqStmts](#) production class takes two paramters: \*stmt, and \*stmts

#### Parameters

* <i>stmt</i>	the statement on the LHS within the sequence of statements
* <i>stmts</i>	the statements on the RHS within the sequence of statements

### 4.43.3 Member Function Documentation

4.43.3.1 `std::string fcgal::ast::SeqStmts::unparse ( )` `[virtual]`

[SeqStmts](#) *unparse()* method.

[SeqStmts](#) *unparse()* returns the stmt\_ and stmts\_ parameters.

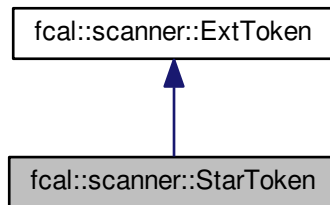
Implements [fcgal::ast::Node](#).

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

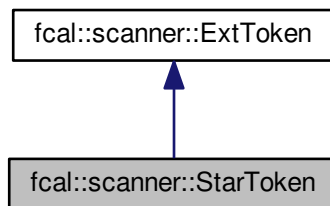
- include/ast.h
- src/ast.cc

## 4.44 fcal::scanner::StarToken Class Reference

Inheritance diagram for fcal::scanner::StarToken:



Collaboration diagram for fcal::scanner::StarToken:



### Public Member Functions

- **StarToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- [parser::ParseResult](#) **led** ([parser::ParseResult](#) left)
- std::string **description** ()
- int **lbp** ()

### Additional Inherited Members

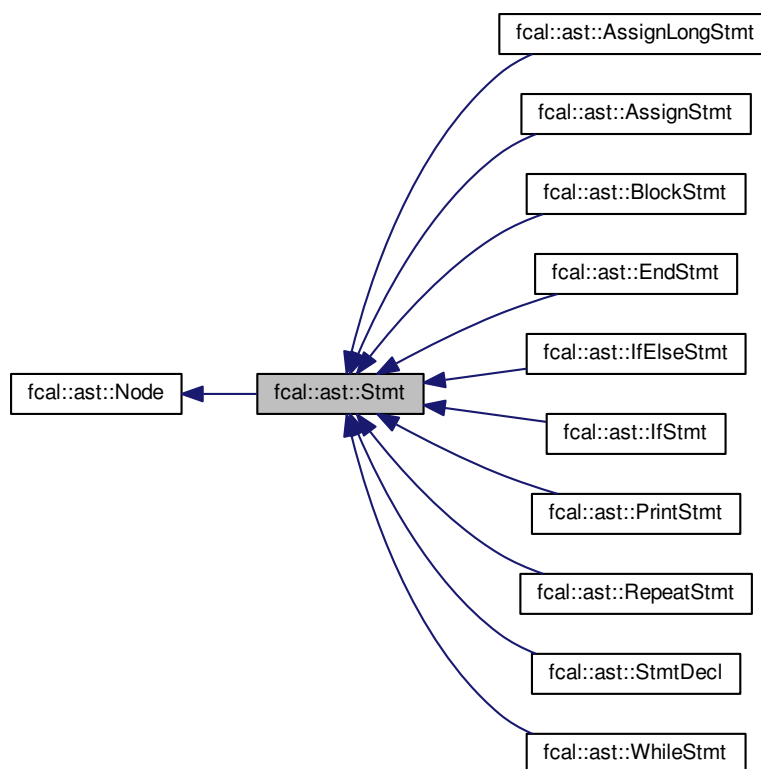
The documentation for this class was generated from the following file:

- include/ext\_token.h

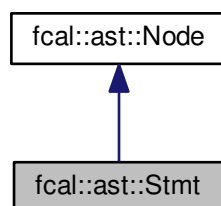
## 4.45 fcal::ast::Stmt Class Reference

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::Stmt:



Collaboration diagram for fcal::ast::Stmt:





## Additional Inherited Members

### 4.45.1 Detailed Description

This is an abstract [Stmt](#) class that inherits directly from the parent [Node](#) class.

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

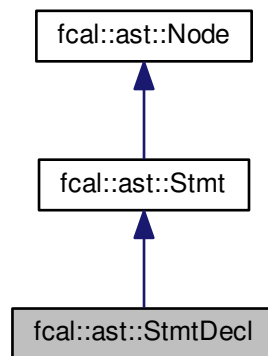
- include/ast.h

## 4.46 fcal::ast::StmtDecl Class Reference

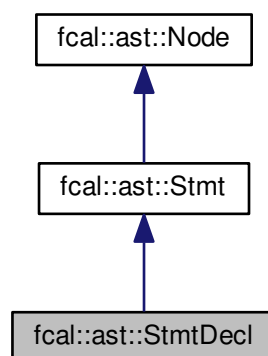
The [StmtDecl](#) class inherits directly from the abstract [Stmt](#) parent class.

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::StmtDecl:



Collaboration diagram for fcal::ast::StmtDecl:



## Public Member Functions

- [StmtDecl](#) ([Decl](#) \*decl)
- `std::string` [unparse](#) ()  
[StmtDecl unparse\(\)](#) method.
- `std::string` [cpp\\_code](#) ()

### 4.46.1 Detailed Description

The [StmtDecl](#) class inherits directly from the abstract [Stmt](#) parent class.

### 4.46.2 Constructor & Destructor Documentation

4.46.2.1 `fcgal::ast::StmtDecl::StmtDecl ( Decl * decl ) [inline],[explicit]`

[StmtDecl](#) production class takes a single parameter: decl

#### Parameters

<code>*decl</code>	is a declaration found with in a single statement
--------------------	---

### 4.46.3 Member Function Documentation

4.46.3.1 `std::string fcgal::ast::StmtDecl::unparse ( ) [virtual]`

[StmtDecl unparse\(\)](#) method.

[StmtDecl unparse\(\)](#) returns the decl\_ parameter.

Implements [fcgal::ast::Node](#).

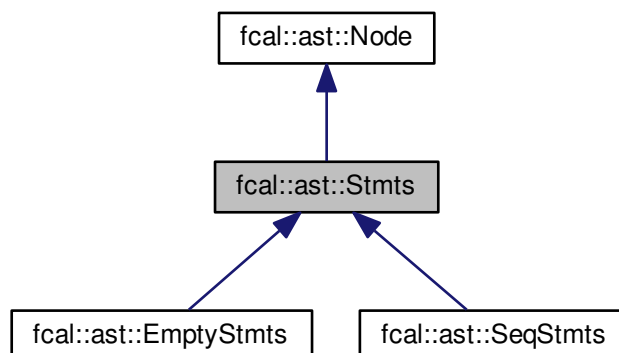
The documentation for this class was generated from the following files:

- include/ast.h
- src/ast.cc

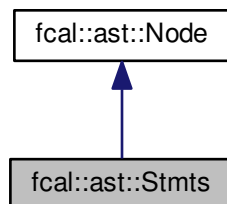
## 4.47 fcgal::ast::Stmts Class Reference

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::Stmts:



Collaboration diagram for fcal::ast::Stmts:



## Additional Inherited Members

### 4.47.1 Detailed Description

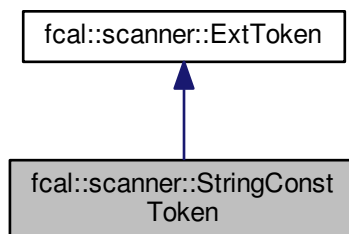
This is an abstract [Stmts](#) class that inherits directly from the parent [Node](#) class.

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

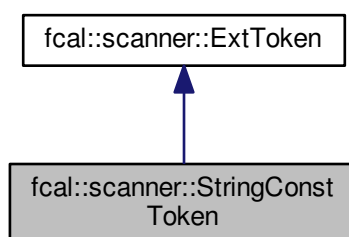
- `include/ast.h`

## 4.48 fcal::scanner::StringConstToken Class Reference

Inheritance diagram for fcal::scanner::StringConstToken:



Collaboration diagram for fcal::scanner::StringConstToken:



### Public Member Functions

- **StringConstToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- [parser::ParseResult](#) **nud** ()
- std::string **description** ()

### Additional Inherited Members

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

- include/ext\_token.h

## 4.49 fcal::scanner::Token Class Reference

```
#include <scanner.h>
```

### Public Member Functions

- [Token](#) ()  
*Token() constructor.*
- [Token](#) (TokenType terminal, std::string lexeme, [Token](#) \*next)  
*Token parameters: terminal, lexeme and next.*
- [~Token](#) ()  
*Token() destructor.*
- TokenType [get\\_terminal\\_](#) ()  
*Token terminal\_ accessor method.*
- void [set\\_terminal\\_](#) (TokenType terminal)  
*Token terminal\_ mutator method.*
- std::string [get\\_lexeme\\_](#) ()  
*Token lexeme\_ accessor method.*
- void [set\\_lexeme\\_](#) (std::string lexeme)  
*Token lexeme\_ mutator method.*
- [Token](#) \* [get\\_next\\_](#) ()  
*Token next\_ accessor method.*
- void [set\\_next\\_](#) ([Token](#) \*next)  
*Token next\_ mutator method.*
- void [return\\_terminal\\_](#) ()
- void [return\\_lexeme\\_](#) ()
- void [return\\_next\\_](#) ()

### Public Attributes

- int [length\\_of\\_lexeme\\_](#)

#### 4.49.1 Detailed Description

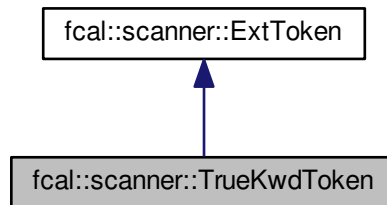
The [Token](#) class defines the various methods and attributes associated with the token types listed in the enum `kTokenEnumType`.

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

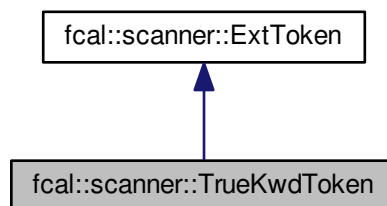
- include/scanner.h
- src/scanner.cc

## 4.50 fcal::scanner::TrueKwdToken Class Reference

Inheritance diagram for fcal::scanner::TrueKwdToken:



Collaboration diagram for fcal::scanner::TrueKwdToken:



### Public Member Functions

- **TrueKwdToken** ([parser::Parser](#) \*p, [Token](#) \*t)
- [parser::ParseResult](#) **nud** ()
- std::string **description** ()

### Additional Inherited Members

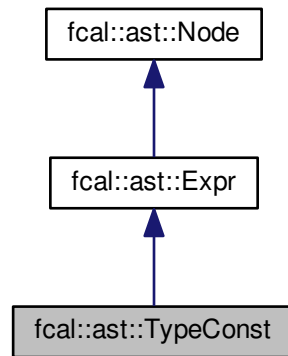
The documentation for this class was generated from the following file:

- include/ext\_token.h

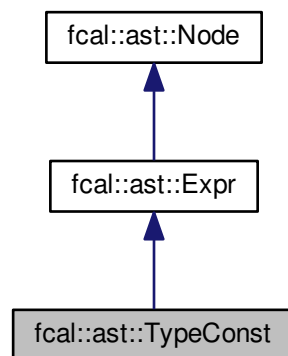
## 4.51 fcal::ast::TypeConst Class Reference

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::TypeConst:



Collaboration diagram for fcal::ast::TypeConst:



### Public Member Functions

- [TypeConst](#) (std::string type\_const)
- std::string [unparse](#) ()  
*TypeConst unparse() method.*
- std::string [cpp\\_code](#) ()

### 4.51.1 Detailed Description

The [TypeConst](#) class inherits directly from the parent [Expr](#) class. The [TypeConst](#) class combines the redundant nature of the implementing multiple production rule classes to define integer, float, and string constants.

The integer, float, and string constants are defined by referencing the lexeme member of the `prev_token_`.

### 4.51.2 Constructor & Destructor Documentation

4.51.2.1 `fcal::ast::TypeConst::TypeConst ( std::string type_const ) [inline],[explicit]`

[TypeConst](#) production rules take the parameter: `type_const`

Parameters

<code>type_const</code>	refers to the constant of a given data type
-------------------------	---

### 4.51.3 Member Function Documentation

4.51.3.1 `std::string fcal::ast::TypeConst::unparse ( ) [virtual]`

[TypeConst unparse\(\)](#) method.

[TypeConst](#) returns the `type_const` of a data type.

Implements [fcal::ast::Node](#).

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

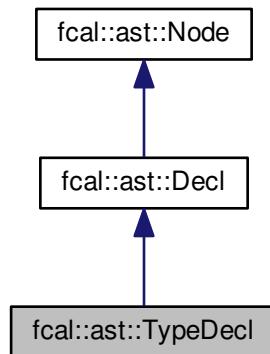
- `include/ast.h`
- `src/ast.cc`

## 4.52 fcal::ast::TypeDecl Class Reference

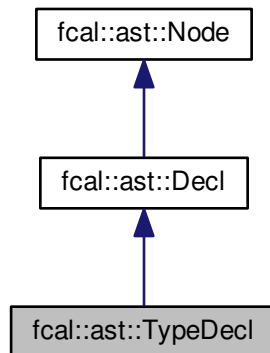
```
#include <ast.h>
```



Inheritance diagram for fcal::ast::TypeDecl:



Collaboration diagram for fcal::ast::TypeDecl:



### Public Member Functions

- [TypeDecl](#) ([VarName](#) \*type, [VarName](#) \*var\_name)
- `std::string` [unparse](#) ()  
*[TypeDecl unparse\(\)](#) method.*
- `std::string` [cpp\\_code](#) ()

#### 4.52.1 Detailed Description

The [TypeDecl](#) class inherits directly from the parent [Decl](#) class. The [TypeDecl](#) class combines the redundant nature of the implementing multiple production rule classes to define integer, float, string and boolean data types.

The integer, float, string and data types is defined by the \*type parameter that is passed to the constructor.

## 4.52.2 Constructor & Destructor Documentation

4.52.2.1 `fcgal::ast::TypeDecl::TypeDecl ( VarName * type, VarName * var_name ) [inline], [explicit]`

[TypeDecl](#) production class takes the parameters: `*type` and `*var_name`

Parameters

<code>*type</code>	defines the data type of the declaration
<code>*var_name</code>	defines the variable name of the specific data type

## 4.52.3 Member Function Documentation

4.52.3.1 `std::string fcal::ast::TypeDecl::unparse ( ) [virtual]`

[TypeDecl](#) `unparse()` method.

[TypeDecl](#) returns the `type_` and `var_name_` of a declaration.

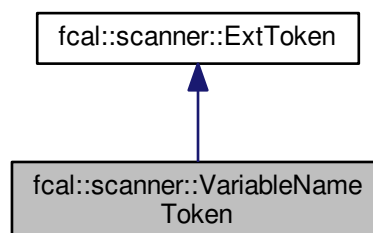
Implements [fcgal::ast::Node](#).

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

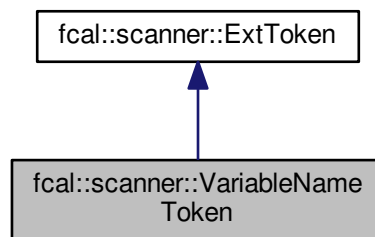
- `include/ast.h`
- `src/ast.cc`

## 4.53 fcal::scanner::VariableNameToken Class Reference

Inheritance diagram for `fcgal::scanner::VariableNameToken`:



Collaboration diagram for fcal::scanner::VariableNameToken:



### Public Member Functions

- **VariableNameToken** (`parser::Parser *p`, `Token *t`)
- `parser::ParseResult nud` ()
- `std::string description` ()

### Additional Inherited Members

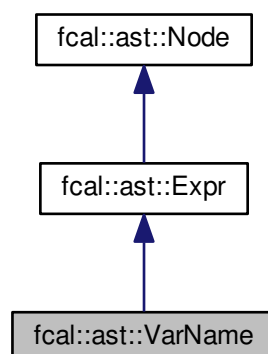
The documentation for this class was generated from the following file:

- `include/ext_token.h`

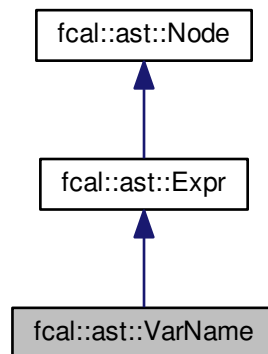
## 4.54 fcal::ast::VarName Class Reference

```
#include <ast.h>
```

Inheritance diagram for fcal::ast::VarName:



Collaboration diagram for `fcgal::ast::VarName`:



## Public Member Functions

- [VarName](#) (`std::string lexeme`)
- `std::string` [unparse](#) ()  
     [VarName unparse\(\)](#) method.
- `std::string` [cpp\\_code](#) ()

### 4.54.1 Detailed Description

The [VarName](#) class is actually a terminal type that is constantly referenced by the other production rules. The [VarName](#) class is defined as a child of the [Expr](#) only because of the specifications of the production rules. Otherwise, [VarName](#) could also be defined as a class inheriting directly from the [Node](#) class.

### 4.54.2 Constructor & Destructor Documentation

4.54.2.1 `fcgal::ast::VarName::VarName ( std::string lexeme )` `[inline]`, `[explicit]`

[VarName](#) production rules take the parameter: `lexeme`

#### Parameters

<code>lexeme</code>	is the lexeme string of a given token
---------------------	---------------------------------------

### 4.54.3 Member Function Documentation

4.54.3.1 `std::string fcgal::ast::VarName::unparse ( )` `[virtual]`

[VarName unparse\(\)](#) method.

[VarName](#) `unparse()` returns the `lexeme_` parameter.

Implements [fcal::ast::Node](#).

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

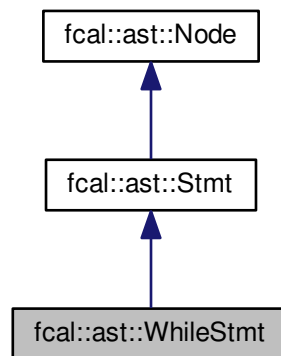
- `include/ast.h`
- `src/ast.cc`

## 4.55 fcal::ast::WhileStmt Class Reference

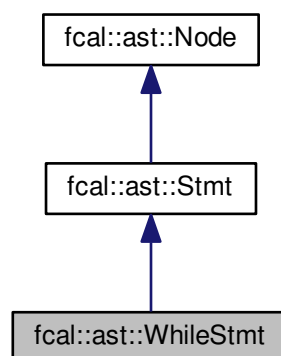
The [WhileStmt](#) class inherits directly from the abstract [Stmt](#) parent class.

```
#include <ast.h>
```

Inheritance diagram for `fcal::ast::WhileStmt`:



Collaboration diagram for `fcal::ast::WhileStmt`:



## Public Member Functions

- [WhileStmt](#) ([Expr](#) \*expr, [Stmt](#) \*stmt)
- std::string [unparse](#) ()  
[WhileStmt unparse\(\)](#) method.
- std::string [cpp\\_code](#) ()

### 4.55.1 Detailed Description

The [WhileStmt](#) class inherits directly from the abstract [Stmt](#) parent class.

### 4.55.2 Constructor & Destructor Documentation

4.55.2.1 `fcgal::ast::WhileStmt::WhileStmt ( Expr * expr, Stmt * stmt ) [inline],[explicit]`

[WhileStmt](#) production class takes the parameters: \*expr and \*stmt

#### Parameters

* <i>expr</i>	the expression being evaluated to continue looping
* <i>stmt</i>	the statement to be looped

### 4.55.3 Member Function Documentation

4.55.3.1 `std::string fcal::ast::WhileStmt::unparse ( ) [virtual]`

[WhileStmt unparse\(\)](#) method.

[WhileStmt](#) returns expr\_ and stmt\_ parameters.

Implements [fcgal::ast::Node](#).

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

- include/ast.h
- src/ast.cc

# Index

~Program  
  fcal::ast::Program, 62

AssignLongStmt  
  fcal::ast::AssignLongStmt, 8

AssignStmt  
  fcal::ast::AssignStmt, 10

BinaryOp  
  fcal::ast::BinaryOp, 12

BlockStmt  
  fcal::ast::BlockStmt, 14

consume\_whitespace\_and\_comments  
  fcal::scanner::Scanner, 66

fcal::ast::AssignLongStmt, 7  
  AssignLongStmt, 8  
  unparse, 8

fcal::ast::AssignStmt, 9  
  AssignStmt, 10  
  unparse, 10

fcal::ast::BinaryOp, 11  
  BinaryOp, 12  
  unparse, 12

fcal::ast::BlockStmt, 13  
  BlockStmt, 14  
  unparse, 14

fcal::ast::BoolFalse, 14  
  unparse, 15

fcal::ast::BoolTrue, 16  
  unparse, 17

fcal::ast::Decl, 19

fcal::ast::EmptyStmts, 20  
  unparse, 21

fcal::ast::EndStmt, 22  
  unparse, 23

fcal::ast::Expr, 24

fcal::ast::IfElseStmt, 30  
  IfElseStmt, 31  
  unparse, 31

fcal::ast::IfExpr, 32  
  IfExpr, 33  
  unparse, 33

fcal::ast::IfStmt, 33  
  IfStmt, 35  
  unparse, 35

fcal::ast::LetExpr, 38  
  LetExpr, 39  
  unparse, 39

fcal::ast::MatrixDecl, 41  
  MatrixDecl, 42  
  unparse, 42

fcal::ast::MatrixLongDecl, 42  
  MatrixLongDecl, 44  
  unparse, 44

fcal::ast::MatrixRef, 44  
  MatrixRef, 46  
  unparse, 46

fcal::ast::NestedOrFuncCall, 47  
  NestedOrFuncCall, 48  
  unparse, 48

fcal::ast::Node, 48

fcal::ast::NotExpr, 50  
  NotExpr, 51  
  unparse, 51

fcal::ast::ParenExpr, 53  
  ParenExpr, 54  
  unparse, 54

fcal::ast::PrintStmt, 59  
  PrintStmt, 60  
  unparse, 60

fcal::ast::Program, 61  
  ~Program, 62  
  Program, 62  
  unparse, 62

fcal::ast::RepeatStmt, 64  
  RepeatStmt, 65  
  unparse, 65

fcal::ast::SeqStmts, 67  
  SeqStmts, 68  
  unparse, 68

fcal::ast::Stmt, 70

fcal::ast::StmtDecl, 71  
  StmtDecl, 72  
  unparse, 72

fcal::ast::Stmts, 72

fcal::ast::TypeConst, 77  
  TypeConst, 78  
  unparse, 78

fcal::ast::TypeDecl, 78  
  TypeDecl, 80  
  unparse, 80

fcal::ast::VarName, 81  
  unparse, 82  
  VarName, 82

fcal::ast::WhileStmt, 83  
  unparse, 84  
  WhileStmt, 84

- fc`al::`parser`::`ParseResult, 58
- fc`al::`parser`::`Parser, 54
  - parse\_addition, 55
  - parse\_decl, 55
  - parse\_division, 55
  - parse\_false\_kwd, 55
  - parse\_float\_const, 55
  - parse\_if\_expr, 55
  - parse\_int\_const, 56
  - parse\_let\_expr, 56
  - parse\_matrix\_decl, 56
  - parse\_multiplication, 56
  - parse\_nested\_expr, 56
  - parse\_not\_expr, 56
  - parse\_relational\_expr, 56
  - parse\_standard\_decl, 56
  - parse\_stmt, 56
  - parse\_stmts, 57
  - parse\_string\_const, 57
  - parse\_subtraction, 57
  - parse\_true\_kwd, 57
  - parse\_variable\_name, 57
- fc`al::`scanner`::`CharConstToken, 17
- fc`al::`scanner`::`DashToken, 18
- fc`al::`scanner`::`EndOfFileToken, 21
- fc`al::`scanner`::`ExtToken, 26
- fc`al::`scanner`::`FalseKwdToken, 27
- fc`al::`scanner`::`FloatConstToken, 28
- fc`al::`scanner`::`ForwardSlashToken, 29
- fc`al::`scanner`::`IfToken, 35
- fc`al::`scanner`::`IntConstToken, 36
- fc`al::`scanner`::`LeftParenToken, 37
- fc`al::`scanner`::`LetToken, 40
- fc`al::`scanner`::`NotOpToken, 52
- fc`al::`scanner`::`PlusSignToken, 58
- fc`al::`scanner`::`RelationalOpToken, 63
- fc`al::`scanner`::`Scanner, 65
  - consume\_whitespace\_and\_comments, 66
  - InitRegexTokenArray, 66
  - Scan, 66
  - Scanner, 66
- fc`al::`scanner`::`StarToken, 69
- fc`al::`scanner`::`StringConstToken, 74
- fc`al::`scanner`::`Token, 75
- fc`al::`scanner`::`TrueKwdToken, 76
- fc`al::`scanner`::`VariableNameToken, 80
- IfElseStmt
  - fc`al::`ast`::`IfElseStmt, 31
- IfExpr
  - fc`al::`ast`::`IfExpr, 33
- IfStmt
  - fc`al::`ast`::`IfStmt, 35
- InitRegexTokenArray
  - fc`al::`scanner`::`Scanner, 66
- LetExpr
  - fc`al::`ast`::`LetExpr, 39
- MatrixDecl
  - fc`al::`ast`::`MatrixDecl, 42
- MatrixLongDecl
  - fc`al::`ast`::`MatrixLongDecl, 44
- MatrixRef
  - fc`al::`ast`::`MatrixRef, 46
- MySequence< T, N >, 46
- NestedOrFuncCall
  - fc`al::`ast`::`NestedOrFuncCall, 48
- NotExpr
  - fc`al::`ast`::`NotExpr, 51
- ParenExpr
  - fc`al::`ast`::`ParenExpr, 54
- parse\_addition
  - fc`al::`parser`::`Parser, 55
- parse\_decl
  - fc`al::`parser`::`Parser, 55
- parse\_division
  - fc`al::`parser`::`Parser, 55
- parse\_false\_kwd
  - fc`al::`parser`::`Parser, 55
- parse\_float\_const
  - fc`al::`parser`::`Parser, 55
- parse\_if\_expr
  - fc`al::`parser`::`Parser, 55
- parse\_int\_const
  - fc`al::`parser`::`Parser, 56
- parse\_let\_expr
  - fc`al::`parser`::`Parser, 56
- parse\_matrix\_decl
  - fc`al::`parser`::`Parser, 56
- parse\_multiplication
  - fc`al::`parser`::`Parser, 56
- parse\_nested\_expr
  - fc`al::`parser`::`Parser, 56
- parse\_not\_expr
  - fc`al::`parser`::`Parser, 56
- parse\_relational\_expr
  - fc`al::`parser`::`Parser, 56
- parse\_standard\_decl
  - fc`al::`parser`::`Parser, 56
- parse\_stmt
  - fc`al::`parser`::`Parser, 56
- parse\_stmts
  - fc`al::`parser`::`Parser, 57
- parse\_string\_const
  - fc`al::`parser`::`Parser, 57
- parse\_subtraction
  - fc`al::`parser`::`Parser, 57
- parse\_true\_kwd
  - fc`al::`parser`::`Parser, 57
- parse\_variable\_name
  - fc`al::`parser`::`Parser, 57
- PrintStmt
  - fc`al::`ast`::`PrintStmt, 60
- Program
  - fc`al::`ast`::`Program, 62



RepeatStmt  
  fcal::ast::RepeatStmt, [65](#)

Scan  
  fcal::scanner::Scanner, [66](#)

Scanner  
  fcal::scanner::Scanner, [66](#)

SeqStmts  
  fcal::ast::SeqStmts, [68](#)

StmtDecl  
  fcal::ast::StmtDecl, [72](#)

TypeConst  
  fcal::ast::TypeConst, [78](#)

TypeDecl  
  fcal::ast::TypeDecl, [80](#)

unparse  
  fcal::ast::AssignLongStmt, [8](#)  
  fcal::ast::AssignStmt, [10](#)  
  fcal::ast::BinaryOp, [12](#)  
  fcal::ast::BlockStmt, [14](#)  
  fcal::ast::BoolFalse, [15](#)  
  fcal::ast::BoolTrue, [17](#)  
  fcal::ast::EmptyStmts, [21](#)  
  fcal::ast::EndStmt, [23](#)  
  fcal::ast::IfElseStmt, [31](#)  
  fcal::ast::IfExpr, [33](#)  
  fcal::ast::IfStmt, [35](#)  
  fcal::ast::LetExpr, [39](#)  
  fcal::ast::MatrixDecl, [42](#)  
  fcal::ast::MatrixLongDecl, [44](#)  
  fcal::ast::MatrixRef, [46](#)  
  fcal::ast::NestedOrFuncCall, [48](#)  
  fcal::ast::NotExpr, [51](#)  
  fcal::ast::ParenExpr, [54](#)  
  fcal::ast::PrintStmt, [60](#)  
  fcal::ast::Program, [62](#)  
  fcal::ast::RepeatStmt, [65](#)  
  fcal::ast::SeqStmts, [68](#)  
  fcal::ast::StmtDecl, [72](#)  
  fcal::ast::TypeConst, [78](#)  
  fcal::ast::TypeDecl, [80](#)  
  fcal::ast::VarName, [82](#)  
  fcal::ast::WhileStmt, [84](#)

VarName  
  fcal::ast::VarName, [82](#)

WhileStmt  
  fcal::ast::WhileStmt, [84](#)