msdscript

Generated by Doxygen 1.10.0

1 Hierarchical Index **1**

1.1 Class Hierarchy . . . 1

2 Class Index **3**

2.1 Class List . . . 3

3 File Index **5**

3.1 File List . . . 5

4 Class Documentation **7**

4.1 Add Class Reference . . . 7

4.1.1 Constructor & Destructor Documentation . . . 8

4.1.1.1 Add() . . . 8

4.1.2 Member Function Documentation . . . 8

4.1.2.1 equals() . . . 8

4.1.2.2 has_variable() . . . 8

4.1.2.3 interp() . . . 8

4.1.2.4 pretty_print() . . . 8

4.1.2.5 pretty_print_at() . . . 9

4.1.2.6 print() . . . 9

4.1.2.7 subst() . . . 9

4.1.3 Member Data Documentation . . . 9

4.1.3.1 lhs_m . . . 9

4.1.3.2 rhs_m . . . 9

4.2 Bool Class Reference . . . 10

4.2.1 Constructor & Destructor Documentation . . . 10

4.2.1.1 Bool() . . . 10

4.2.2 Member Function Documentation . . . 11

4.2.2.1 equals() . . . 11

4.2.2.2 has_variable() . . . 11

4.2.2.3 interp() . . . 11

4.2.2.4 print() . . . 11

4.2.2.5 subst() . . . 11

4.2.3 Member Data Documentation . . . 11

4.2.3.1 bool_m . . . 11

4.3 BoolVal Class Reference . . . 12

4.3.1 Constructor & Destructor Documentation . . . 12

4.3.1.1 BoolVal() . . . 12

4.3.2 Member Function Documentation . . . 12

4.3.2.1 add_to() . . . 12

4.3.2.2 equals() . . . 13

4.3.2.3 is_true() . . . 13

4.3.2.4 mult_with() . . . 13

4.3.2.5 print() 13
4.3.2.6 to_expr() 13
4.3.3 Member Data Documentation 13
4.3.3.1 bool_m 13
4.4 Eq Class Reference 14
4.4.1 Constructor & Destructor Documentation 14
4.4.1.1 Eq() 14
4.4.2 Member Function Documentation 15
4.4.2.1 equals() 15
4.4.2.2 has_variable() 15
4.4.2.3 interp() 15
4.4.2.4 pretty_print() 15
4.4.2.5 pretty_print_at() 15
4.4.2.6 print() 15
4.4.2.7 subst() 16
4.4.3 Member Data Documentation 16
4.4.3.1 lhs_m 16
4.4.3.2 rhs_m 16
4.5 Expr Class Reference 16
4.5.1 Member Function Documentation 17
4.5.1.1 equals() 17
4.5.1.2 has_variable() 17
4.5.1.3 interp() 17
4.5.1.4 pretty_print() 17
4.5.1.5 pretty_print_at() 17
4.5.1.6 print() 18
4.5.1.7 subst() 18
4.5.1.8 to_pretty_string() 18
4.5.1.9 to_string() 18
4.6 If Class Reference 18
4.6.1 Constructor & Destructor Documentation 19
4.6.1.1 If() 19
4.6.2 Member Function Documentation 19
4.6.2.1 equals() 19
4.6.2.2 has_variable() 19
4.6.2.3 interp() 20
4.6.2.4 pretty_print() 20
4.6.2.5 pretty_print_at() 20
4.6.2.6 print() 20
4.6.2.7 subst() 20
4.6.3 Member Data Documentation 20
4.6.3.1 else_m 20

4.6.3.2 test_m 21
4.6.3.3 then_m 21
4.7 Let Class Reference 21
4.7.1 Constructor & Destructor Documentation 22
4.7.1.1 Let() 22
4.7.2 Member Function Documentation 22
4.7.2.1 equals() 22
4.7.2.2 has_variable() 22
4.7.2.3 interp() 22
4.7.2.4 pretty_print() 22
4.7.2.5 pretty_print_at() 23
4.7.2.6 print() 23
4.7.2.7 subst() 23
4.7.3 Member Data Documentation 23
4.7.3.1 body_m 23
4.7.3.2 lhs_m 23
4.7.3.3 rhs_m 23
4.8 Mult Class Reference 24
4.8.1 Constructor & Destructor Documentation 24
4.8.1.1 Mult() 24
4.8.2 Member Function Documentation 25
4.8.2.1 equals() 25
4.8.2.2 has_variable() 25
4.8.2.3 interp() 25
4.8.2.4 pretty_print() 25
4.8.2.5 pretty_print_at() 25
4.8.2.6 print() 25
4.8.2.7 subst() 26
4.8.3 Member Data Documentation 26
4.8.3.1 lhs_m 26
4.8.3.2 rhs_m 26
4.9 Num Class Reference 26
4.9.1 Constructor & Destructor Documentation 27
4.9.1.1 Num() 27
4.9.2 Member Function Documentation 27
4.9.2.1 equals() 27
4.9.2.2 has_variable() 27
4.9.2.3 interp() 27
4.9.2.4 print() 28
4.9.2.5 subst() 28
4.9.3 Member Data Documentation 28
4.9.3.1 int_m 28

4.10 NumVal Class Reference 28
4.10.1 Constructor & Destructor Documentation 29
4.10.1.1 NumVal() 29
4.10.2 Member Function Documentation 29
4.10.2.1 add_to() 29
4.10.2.2 equals() 29
4.10.2.3 is_true() 29
4.10.2.4 mult_with() 29
4.10.2.5 print() 30
4.10.2.6 to_expr() 30
4.10.3 Member Data Documentation 30
4.10.3.1 int_m 30
4.11 Val Class Reference 30
4.11.1 Member Function Documentation 31
4.11.1.1 add_to() 31
4.11.1.2 equals() 31
4.11.1.3 is_true() 31
4.11.1.4 mult_with() 31
4.11.1.5 print() 31
4.11.1.6 to_expr() 31
4.11.1.7 to_string() 31
4.12 Var Class Reference 32
4.12.1 Constructor & Destructor Documentation 32
4.12.1.1 Var() 32
4.12.2 Member Function Documentation 33
4.12.2.1 equals() 33
4.12.2.2 has_variable() 33
4.12.2.3 interp() 33
4.12.2.4 print() 33
4.12.2.5 subst() 33
4.12.3 Member Data Documentation 33
4.12.3.1 str_m 33

5 File Documentation **35**
5.1 /Users/u0858882/Desktop/msdscript/msdscript/src/cmdline.cpp File Reference 35
5.1.1 Macro Definition Documentation 35
5.1.1.1 CATCH_CONFIG_RUNNER 35
5.1.2 Function Documentation 35
5.1.2.1 handle_cin() 35
5.1.2.2 if_help() 36
5.1.2.3 if_interp() 36
5.1.2.4 if_pretty_print() 36

5.1.2.5 if_print() 36
5.1.2.6 if_test() 36
5.1.2.7 use_arguments() 36
5.2 /Users/u0858882/Desktop/msdscript/msdscript/src/cmdline.h File Reference 36
5.2.1 Function Documentation 37
5.2.1.1 use_arguments() 37
5.3 cmdline.h 37
5.4 /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.cpp File Reference 37
5.5 /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.h File Reference 37
5.5.1 Enumeration Type Documentation 38
5.5.1.1 precedence_t 38
5.6 Expr.h 38
5.7 /Users/u0858882/Desktop/msdscript/msdscript/src/main.cpp File Reference 42
5.7.1 Function Documentation 42
5.7.1.1 main() 42
5.8 /Users/u0858882/Desktop/msdscript/msdscript/src/parse.cpp File Reference 43
5.8.1 Function Documentation 43
5.8.1.1 build_number() 43
5.8.1.2 consume() [1/2] 43
5.8.1.3 consume() [2/2] 43
5.8.1.4 consume_whitespace() 43
5.8.1.5 parse_adds() 44
5.8.1.6 parse_bool() 44
5.8.1.7 parse_eqs() 44
5.8.1.8 parse_expr() [1/2] 44
5.8.1.9 parse_expr() [2/2] 44
5.8.1.10 parse_if() 44
5.8.1.11 parse_let() 44
5.8.1.12 parse_mults() 44
5.8.1.13 parse_num() 44
5.8.1.14 parse_paren() 45
5.8.1.15 parse_unary_and_ternary_exprs() 45
5.8.1.16 parse_var() 45
5.8.1.17 peek_keyword() 45
5.9 /Users/u0858882/Desktop/msdscript/msdscript/src/parse.h File Reference 45
5.9.1 Function Documentation 45
5.9.1.1 parse_expr() 45
5.10 parse.h 46
5.11 /Users/u0858882/Desktop/msdscript/msdscript/src/Val.cpp File Reference 46
5.12 /Users/u0858882/Desktop/msdscript/msdscript/src/Val.h File Reference 46
5.13 Val.h 46
5.14 /Users/u0858882/Desktop/msdscript/test_msdscript/src/test_msdscript.cpp File Reference 47

5.14.1 Function Documentation 48
5.14.1.1 compare_IO() 48
5.14.1.2 compare_programs() 48
5.14.1.3 main() 48
5.14.1.4 write_report_header() 48
5.14.1.5 write_results() [1/2] 49
5.14.1.6 write_results() [2/2] 49
5.14.2 Variable Documentation 49
5.14.2.1 EXECS_DIR 49
5.14.2.2 OUTPUT_DIR 49
5.14.2.3 TEST_ITER 49

Index **51**

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

- Expr 16
  - Add 7
  - Bool 10
  - Eq 14
  - If 18
  - Let 21
  - Mult 24
  - Num 26
  - Var 32
- Val 30
  - BoolVal 12
  - NumVal 28

Chapter 2

Class Index

2.1 Class List

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

Add 7
Bool 10
BoolVal 12
Eq 14
Expr 16
If 18
Let 21
Mult 24
Num 26
NumVal 28
Val 30
Var 32

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

/Users/u0858882/Desktop/msdscript/msdscript/src/cmdline.cpp 35
/Users/u0858882/Desktop/msdscript/msdscript/src/cmdline.h 36
/Users/u0858882/Desktop/msdscript/msdscript/src/Expr.cpp 37
/Users/u0858882/Desktop/msdscript/msdscript/src/Expr.h 37
/Users/u0858882/Desktop/msdscript/msdscript/src/main.cpp 42
/Users/u0858882/Desktop/msdscript/msdscript/src/parse.cpp 43
/Users/u0858882/Desktop/msdscript/msdscript/src/parse.h 45
/Users/u0858882/Desktop/msdscript/msdscript/src/Val.cpp 46
/Users/u0858882/Desktop/msdscript/msdscript/src/Val.h 46
/Users/u0858882/Desktop/msdscript/test_msdscript/src/test_msdscript.cpp 47

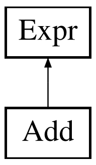
Chapter 4

Class Documentation

4.1 Add Class Reference

```
#include <Expr.h>
```

Inheritance diagram for Add:



Public Member Functions

- Add (Expr ∗lhs, Expr ∗rhs)
- bool equals (Expr ∗e) override
- Val ∗ interp () override
- bool has_variable () override
- Expr ∗ subst (std::string str, Expr ∗e) override

Public Member Functions inherited from Expr

- std::string to_string ()
- std::string to_pretty_string ()

Public Attributes

- Expr ∗ lhs_m

  The lhs operand of an addition operation.
- Expr ∗ rhs_m

  The rhs operand of an addition operation.

Private Member Functions

- void print (std::ostream &stream) override
- void pretty_print (std::ostream &stream) override
- void pretty_print_at (std::ostream &stream, precedence_t p, std::streampos &pos, bool paren) override

4.1.1 Constructor & Destructor Documentation

4.1.1.1 Add()

```
Add::Add (
            Expr * lhs,
            Expr * rhs )
```

4.1.2 Member Function Documentation

4.1.2.1 equals()

```
bool Add::equals (
            Expr * e )  [override], [virtual]
```

Implements Expr.

4.1.2.2 has_variable()

```
bool Add::has_variable ( )  [override], [virtual]
```

Implements Expr.

4.1.2.3 interp()

```
Val * Add::interp ( )  [override], [virtual]
```

Implements Expr.

4.1.2.4 pretty_print()

```
void Add::pretty_print (
            std::ostream & stream )  [override], [private], [virtual]
```

Reimplemented from Expr.

4.1.2.5 pretty_print_at()

```
void Add::pretty_print_at (
             std::ostream & stream,
             precedence_t p,
             std::streampos & pos,
             bool paren )  [override], [private], [virtual]
```

Reimplemented from Expr.

4.1.2.6 print()

```
void Add::print (
             std::ostream & stream )  [override], [private], [virtual]
```

Implements Expr.

4.1.2.7 subst()

```
Expr * Add::subst (
             std::string str,
             Expr * e )  [override], [virtual]
```

Implements Expr.

4.1.3 Member Data Documentation

4.1.3.1 lhs_m

```
Expr* Add::lhs_m
```

The lhs operand of an addition operation.

4.1.3.2 rhs_m

```
Expr* Add::rhs_m
```

The rhs operand of an addition operation.

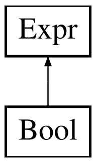
The documentation for this class was generated from the following files:

- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.h
- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.cpp

4.2 Bool Class Reference

```
#include <Expr.h>
```

Inheritance diagram for Bool:



Public Member Functions

- Bool (bool val)
- bool equals (Expr ∗e) override
- Val ∗ interp () override
- bool has_variable () override
- Expr ∗ subst (std::string str, Expr ∗e) override

Public Member Functions inherited from Expr

- std::string to_string ()
- std::string to_pretty_string ()
- virtual void pretty_print (std::ostream &stream)
- virtual void pretty_print_at (std::ostream &stream, precedence_t p, std::streampos &pos, bool paren)

Public Attributes

- bool bool_m

  The boolean value of the Bool object.

Private Member Functions

- void print (std::ostream &stream) override

4.2.1 Constructor & Destructor Documentation

4.2.1.1 Bool()

```
Bool::Bool (
            bool val )  [explicit]
```

4.2.2 Member Function Documentation

4.2.2.1 equals()

```
bool Bool::equals (
             Expr * e )  [override], [virtual]
```

Implements Expr.

4.2.2.2 has_variable()

```
bool Bool::has_variable ( )  [override], [virtual]
```

Implements Expr.

4.2.2.3 interp()

```
Val * Bool::interp ( )  [override], [virtual]
```

Implements Expr.

4.2.2.4 print()

```
void Bool::print (
             std::ostream & stream )  [override], [private], [virtual]
```

Implements Expr.

4.2.2.5 subst()

```
Expr * Bool::subst (
             std::string str,
             Expr * e )  [override], [virtual]
```

Implements Expr.

4.2.3 Member Data Documentation

4.2.3.1 bool_m

```
bool Bool::bool_m
```

The boolean value of the Bool object.

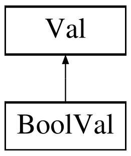
The documentation for this class was generated from the following files:

- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.h
- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.cpp

4.3 BoolVal Class Reference

```
#include <Val.h>
```

Inheritance diagram for BoolVal:



Public Member Functions

- BoolVal (bool val)
- Expr ∗ to_expr () override
- bool equals (Val ∗v) override
- Val ∗ add_to (Val ∗other_val) override
- Val ∗ mult_with (Val ∗other_val) override
- bool is_true () override
- void print (std::ostream &ostream) override

Public Member Functions inherited from Val

- std::string to_string ()

Public Attributes

- bool bool_m

4.3.1 Constructor & Destructor Documentation

4.3.1.1 BoolVal()

```
BoolVal::BoolVal (
            bool val )  [explicit]
```

4.3.2 Member Function Documentation

4.3.2.1 add_to()

```
Val ∗ BoolVal::add_to (
            Val ∗ other_val )  [override], [virtual]
```

Implements Val.

4.3.2.2 equals()

```
bool BoolVal::equals (
             Val * v )  [override], [virtual]
```

Implements Val.

4.3.2.3 is_true()

```
bool BoolVal::is_true ( )  [override], [virtual]
```

Implements Val.

4.3.2.4 mult_with()

```
Val * BoolVal::mult_with (
             Val * other_val )  [override], [virtual]
```

Implements Val.

4.3.2.5 print()

```
void BoolVal::print (
             std::ostream & ostream )  [override], [virtual]
```

Implements Val.

4.3.2.6 to_expr()

```
Expr * BoolVal::to_expr ( )  [override], [virtual]
```

Implements Val.

4.3.3 Member Data Documentation

4.3.3.1 bool_m

```
bool BoolVal::bool_m
```

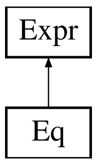
The documentation for this class was generated from the following files:

- /Users/u0858882/Desktop/msdscript/msdscript/src/Val.h
- /Users/u0858882/Desktop/msdscript/msdscript/src/Val.cpp

4.4 Eq Class Reference

```
#include <Expr.h>
```

Inheritance diagram for Eq:



Public Member Functions

- Eq (Expr ∗lhs, Expr ∗rhs)
- bool equals (Expr ∗e) override
- Val ∗ interp () override
- bool has_variable () override
- Expr ∗ subst (std::string str, Expr ∗e) override

Public Member Functions inherited from Expr

- std::string to_string ()
- std::string to_pretty_string ()

Public Attributes

- Expr ∗ lhs_m

  The lhs operand of an equality operation.
- Expr ∗ rhs_m

  The rhs operand of an equality operation.

Private Member Functions

- void print (std::ostream &stream) override
- void pretty_print (std::ostream &stream) override
- void pretty_print_at (std::ostream &stream, precedence_t p, std::streampos &pos, bool paren) override

4.4.1 Constructor & Destructor Documentation

4.4.1.1 Eq()

```
Eq::Eq (
            Expr * lhs,
            Expr * rhs )
```

4.4.2 Member Function Documentation

4.4.2.1 equals()

```
bool Eq::equals (
             Expr * e )  [override], [virtual]
```

Implements Expr.

4.4.2.2 has_variable()

```
bool Eq::has_variable ( )  [override], [virtual]
```

Implements Expr.

4.4.2.3 interp()

```
Val * Eq::interp ( )  [override], [virtual]
```

Implements Expr.

4.4.2.4 pretty_print()

```
void Eq::pretty_print (
             std::ostream & stream )  [override], [private], [virtual]
```

Reimplemented from Expr.

4.4.2.5 pretty_print_at()

```
void Eq::pretty_print_at (
             std::ostream & stream,
             precedence_t p,
             std::streampos & pos,
             bool paren )  [override], [private], [virtual]
```

Reimplemented from Expr.

4.4.2.6 print()

```
void Eq::print (
             std::ostream & stream )  [override], [private], [virtual]
```

Implements Expr.

4.4.2.7 subst()

```
Expr * Eq::subst (
             std::string str,
             Expr * e )  [override], [virtual]
```

Implements Expr.

4.4.3 Member Data Documentation

4.4.3.1 lhs_m

```
Expr* Eq::lhs_m
```

The lhs operand of an equality operation.

4.4.3.2 rhs_m

```
Expr* Eq::rhs_m
```

The rhs operand of an equality operation.

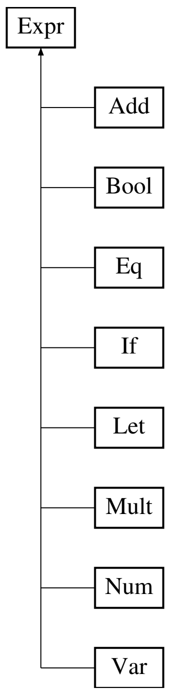
The documentation for this class was generated from the following files:

- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.h
- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.cpp

4.5 Expr Class Reference

```
#include <Expr.h>
```

Inheritance diagram for Expr:



Public Member Functions

- std::string to_string ()
- std::string to_pretty_string ()
- virtual bool equals (Expr ∗e)=0
- virtual Val ∗ interp ()=0
- virtual bool has_variable ()=0
- virtual Expr ∗ subst (std::string str, Expr ∗e)=0
- virtual void print (std::ostream &stream)=0
- virtual void pretty_print (std::ostream &stream)
- virtual void pretty_print_at (std::ostream &stream, precedence_t p, std::streampos &pos, bool paren)

4.5.1 Member Function Documentation

4.5.1.1 equals()

```
virtual bool Expr::equals (
             Expr * e )  [pure virtual]
```

Implemented in Num, Bool, Eq, Add, Mult, Var, Let, and If.

4.5.1.2 has_variable()

```
virtual bool Expr::has_variable ( )  [pure virtual]
```

Implemented in Num, Bool, Eq, Add, Mult, Var, Let, and If.

4.5.1.3 interp()

```
virtual Val * Expr::interp ( )  [pure virtual]
```

Implemented in Num, Bool, Eq, Add, Mult, Var, Let, and If.

4.5.1.4 pretty_print()

```
virtual void Expr::pretty_print (
             std::ostream & stream )  [inline], [virtual]
```

Reimplemented in Eq, Add, Mult, Let, and If.

4.5.1.5 pretty_print_at()

```
virtual void Expr::pretty_print_at (
             std::ostream & stream,
             precedence_t p,
             std::streampos & pos,
             bool paren )  [inline], [virtual]
```

Reimplemented in Let, If, Eq, Add, and Mult.

4.5.1.6 print()

```
virtual void Expr::print (
             std::ostream & stream )  [pure virtual]
```

Implemented in Num, Bool, Eq, Add, Mult, Var, Let, and If.

4.5.1.7 subst()

```
virtual Expr * Expr::subst (
             std::string str,
             Expr * e )  [pure virtual]
```

Implemented in Num, Bool, Eq, Add, Mult, Var, Let, and If.

4.5.1.8 to_pretty_string()

```
std::string Expr::to_pretty_string ( )
```

4.5.1.9 to_string()

```
std::string Expr::to_string ( )
```

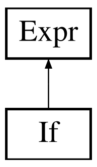
The documentation for this class was generated from the following files:

- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.h
- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.cpp

4.6 If Class Reference

```
#include <Expr.h>
```

Inheritance diagram for If:



Public Member Functions

- If (Expr ∗condition, Expr ∗first_branch, Expr ∗second_branch)
- bool equals (Expr ∗e) override
- Val ∗ interp () override
- bool has_variable () override
- Expr ∗ subst (std::string str, Expr ∗e) override

Public Member Functions inherited from Expr

- std::string to_string ()
- std::string to_pretty_string ()

Public Attributes

- Expr ∗ test_m

  The condition operand of an If expression.
- Expr ∗ then_m

  Branch 1 operand of an If expression.
- Expr ∗ else_m

  Branch 2 operand of an If expression.

Private Member Functions

- void print (std::ostream &stream) override
- void pretty_print (std::ostream &stream) override
- void pretty_print_at (std::ostream &stream, precedence_t p, std::streampos &caller_pos, bool has_paren) override

4.6.1 Constructor & Destructor Documentation

4.6.1.1 If()

```
If::If (
            Expr ∗ condition,
            Expr ∗ first_branch,
            Expr ∗ second_branch )
```

4.6.2 Member Function Documentation

4.6.2.1 equals()

```
bool If::equals (
            Expr ∗ e )  [override], [virtual]
```

Implements Expr.

4.6.2.2 has_variable()

```
bool If::has_variable ( )  [override], [virtual]
```

Implements Expr.

4.6.2.3 interp()

```
Val * If::interp ( )  [override], [virtual]
```

Implements Expr.

4.6.2.4 pretty_print()

```
void If::pretty_print (
             std::ostream & stream )  [override], [private], [virtual]
```

Reimplemented from Expr.

4.6.2.5 pretty_print_at()

```
void If::pretty_print_at (
             std::ostream & stream,
             precedence_t p,
             std::streampos & caller_pos,
             bool has_paren )  [override], [private], [virtual]
```

Reimplemented from Expr.

4.6.2.6 print()

```
void If::print (
             std::ostream & stream )  [override], [private], [virtual]
```

Implements Expr.

4.6.2.7 subst()

```
Expr * If::subst (
             std::string str,
             Expr * e )  [override], [virtual]
```

Implements Expr.

4.6.3 Member Data Documentation

4.6.3.1 else_m

```
Expr* If::else_m
```

Branch 2 operand of an If expression.

4.6.3.2 test_m

```
Expr* If::test_m
```

The condition operand of an If expression.

4.6.3.3 then_m

```
Expr* If::then_m
```

Branch 1 operand of an If expression.

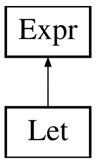
The documentation for this class was generated from the following files:

- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.h
- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.cpp

4.7 Let Class Reference

```
#include <Expr.h>
```

Inheritance diagram for Let:



Public Member Functions

- Let (std::string lhs, Expr ∗rhs, Expr ∗body)
- bool equals (Expr ∗e) override
- Val ∗ interp () override
- bool has_variable () override
- Expr ∗ subst (std::string str, Expr ∗e) override

Public Member Functions inherited from Expr

- std::string to_string ()
- std::string to_pretty_string ()

Public Attributes

- std::string lhs_m

  The Let object's variable name.
- Expr ∗ rhs_m

  The Let object's variable definition.
- Expr ∗ body_m

Private Member Functions

- void print (std::ostream &stream) override
- void pretty_print (std::ostream &stream) override
- void pretty_print_at (std::ostream &stream, precedence_t p, std::streampos &caller_pos, bool has_paren) override

4.7.1 Constructor & Destructor Documentation

4.7.1.1 Let()

```
Let::Let (
             std::string lhs,
             Expr * rhs,
             Expr * body )
```

4.7.2 Member Function Documentation

4.7.2.1 equals()

```
bool Let::equals (
             Expr * e )  [override], [virtual]
```

Implements Expr.

4.7.2.2 has_variable()

```
bool Let::has_variable ( )  [override], [virtual]
```

Implements Expr.

4.7.2.3 interp()

```
Val * Let::interp ( )  [override], [virtual]
```

Implements Expr.

4.7.2.4 pretty_print()

```
void Let::pretty_print (
             std::ostream & stream )  [override], [private], [virtual]
```

Reimplemented from Expr.

4.7.2.5 pretty_print_at()

```
void Let::pretty_print_at (
             std::ostream & stream,
             precedence_t p,
             std::streampos & caller_pos,
             bool has_paren )  [override], [private], [virtual]
```

Reimplemented from Expr.

4.7.2.6 print()

```
void Let::print (
             std::ostream & stream )  [override], [private], [virtual]
```

Implements Expr.

4.7.2.7 subst()

```
Expr * Let::subst (
             std::string str,
             Expr * e )  [override], [virtual]
```

Implements Expr.

4.7.3 Member Data Documentation

4.7.3.1 body_m

```
Expr* Let::body_m
```

The expression in which the variable declaration/definition applies

4.7.3.2 lhs_m

```
std::string Let::lhs_m
```

The Let object's variable name.

4.7.3.3 rhs_m

```
Expr* Let::rhs_m
```

The Let object's variable definition.

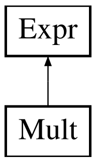
The documentation for this class was generated from the following files:

- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.h
- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.cpp

4.8 Mult Class Reference

```
#include <Expr.h>
```

Inheritance diagram for Mult:



Public Member Functions

- Mult (Expr ∗lhs, Expr ∗rhs)
- bool equals (Expr ∗e) override
- Val ∗ interp () override
- bool has_variable () override
- Expr ∗ subst (std::string str, Expr ∗e) override

Public Member Functions inherited from Expr

- std::string to_string ()
- std::string to_pretty_string ()

Public Attributes

- Expr ∗ lhs_m

  The lhs operand of a multiplication operation.
- Expr ∗ rhs_m

  The rhs operand of an multiplication operation.

Private Member Functions

- void print (std::ostream &stream) override
- void pretty_print (std::ostream &stream) override
- void pretty_print_at (std::ostream &stream, precedence_t p, std::streampos &pos, bool paren) override

4.8.1 Constructor & Destructor Documentation

4.8.1.1 Mult()

```
Mult::Mult (
            Expr ∗ lhs,
            Expr ∗ rhs )
```

4.8.2 Member Function Documentation

4.8.2.1 equals()

```
bool Mult::equals (
             Expr * e )  [override], [virtual]
```

Implements Expr.

4.8.2.2 has_variable()

```
bool Mult::has_variable ( )  [override], [virtual]
```

Implements Expr.

4.8.2.3 interp()

```
Val * Mult::interp ( )  [override], [virtual]
```

Implements Expr.

4.8.2.4 pretty_print()

```
void Mult::pretty_print (
             std::ostream & stream )  [override], [private], [virtual]
```

Reimplemented from Expr.

4.8.2.5 pretty_print_at()

```
void Mult::pretty_print_at (
             std::ostream & stream,
             precedence_t p,
             std::streampos & pos,
             bool paren )  [override], [private], [virtual]
```

Reimplemented from Expr.

4.8.2.6 print()

```
void Mult::print (
             std::ostream & stream )  [override], [private], [virtual]
```

Implements Expr.

4.8.2.7 subst()

```
Expr * Mult::subst (
            std::string str,
            Expr * e )  [override], [virtual]
```

Implements Expr.

4.8.3 Member Data Documentation

4.8.3.1 lhs_m

```
Expr* Mult::lhs_m
```

The lhs operand of a multiplication operation.

4.8.3.2 rhs_m

```
Expr* Mult::rhs_m
```

The rhs operand of an multiplication operation.

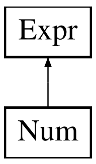
The documentation for this class was generated from the following files:

- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.h
- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.cpp

4.9 Num Class Reference

```
#include <Expr.h>
```

Inheritance diagram for Num:



Public Member Functions

- Num (int val)
- bool equals (Expr *e) override
- Val * interp () override
- bool has_variable () override
- Expr * subst (std::string str, Expr *e) override

Public Member Functions inherited from Expr

- std::string to_string ()
- std::string to_pretty_string ()
- virtual void pretty_print (std::ostream &stream)
- virtual void pretty_print_at (std::ostream &stream, precedence_t p, std::streampos &pos, bool paren)

Public Attributes

- int int_m

  The integer value of the Num object.

Private Member Functions

- void print (std::ostream &stream) override

4.9.1 Constructor & Destructor Documentation

4.9.1.1 Num()

```
Num::Num (
             int val )   [explicit]
```

4.9.2 Member Function Documentation

4.9.2.1 equals()

```
bool Num::equals (
             Expr * e )   [override], [virtual]
```

Implements Expr.

4.9.2.2 has_variable()

```
bool Num::has_variable ( )   [override], [virtual]
```

Implements Expr.

4.9.2.3 interp()

```
Val * Num::interp ( )   [override], [virtual]
```

Implements Expr.

4.9.2.4 print()

```
void Num::print (
            std::ostream & stream )  [override], [private], [virtual]
```

Implements Expr.

4.9.2.5 subst()

```
Expr * Num::subst (
            std::string str,
            Expr * e )  [override], [virtual]
```

Implements Expr.

4.9.3 Member Data Documentation

4.9.3.1 int_m

```
int Num::int_m
```

The integer value of the Num object.

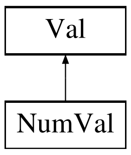
The documentation for this class was generated from the following files:

- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.h
- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.cpp

4.10 NumVal Class Reference

```
#include <Val.h>
```

Inheritance diagram for NumVal:



Public Member Functions

- NumVal (int val)
- Expr ∗ to_expr () override
- bool equals (Val ∗v) override
- Val ∗ add_to (Val ∗other_val) override
- Val ∗ mult_with (Val ∗other_val) override
- bool is_true () override
- void print (std::ostream &ostream) override

Public Member Functions inherited from Val

- std::string to_string ()

Public Attributes

- int int_m

4.10.1 Constructor & Destructor Documentation

4.10.1.1 NumVal()

```
NumVal::NumVal (
             int val )  [explicit]
```

4.10.2 Member Function Documentation

4.10.2.1 add_to()

```
Val * NumVal::add_to (
             Val * other_val )  [override], [virtual]
```

Implements Val.

4.10.2.2 equals()

```
bool NumVal::equals (
             Val * v )  [override], [virtual]
```

Implements Val.

4.10.2.3 is_true()

```
bool NumVal::is_true ( )  [override], [virtual]
```

Implements Val.

4.10.2.4 mult_with()

```
Val * NumVal::mult_with (
             Val * other_val )  [override], [virtual]
```

Implements Val.

4.10.2.5 print()

```
void NumVal::print (
            std::ostream & ostream )  [override], [virtual]
```

Implements Val.

4.10.2.6 to_expr()

```
Expr * NumVal::to_expr ( )  [override], [virtual]
```

Implements Val.

4.10.3 Member Data Documentation

4.10.3.1 int_m

```
int NumVal::int_m
```

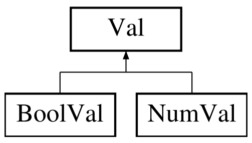
The documentation for this class was generated from the following files:

- /Users/u0858882/Desktop/msdscript/msdscript/src/Val.h
- /Users/u0858882/Desktop/msdscript/msdscript/src/Val.cpp

4.11 Val Class Reference

```
#include <Val.h>
```

Inheritance diagram for Val:



Public Member Functions

- std::string to_string ()
- virtual Expr ∗ to_expr ()=0
- virtual bool equals (Val ∗v)=0
- virtual Val ∗ add_to (Val ∗v)=0
- virtual Val ∗ mult_with (Val ∗v)=0
- virtual bool is_true ()=0
- virtual void print (std::ostream &stream)=0

4.11.1 Member Function Documentation

4.11.1.1 add_to()

```
virtual Val * Val::add_to (
            Val * v )  [pure virtual]
```

Implemented in NumVal, and BoolVal.

4.11.1.2 equals()

```
virtual bool Val::equals (
            Val * v )  [pure virtual]
```

Implemented in NumVal, and BoolVal.

4.11.1.3 is_true()

```
virtual bool Val::is_true ( )  [pure virtual]
```

Implemented in NumVal, and BoolVal.

4.11.1.4 mult_with()

```
virtual Val * Val::mult_with (
            Val * v )  [pure virtual]
```

Implemented in NumVal, and BoolVal.

4.11.1.5 print()

```
virtual void Val::print (
            std::ostream & stream )  [pure virtual]
```

Implemented in NumVal, and BoolVal.

4.11.1.6 to_expr()

```
virtual Expr * Val::to_expr ( )  [pure virtual]
```

Implemented in NumVal, and BoolVal.

4.11.1.7 to_string()

```
std::string Val::to_string ( )
```

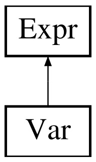
The documentation for this class was generated from the following files:

- /Users/u0858882/Desktop/msdscript/msdscript/src/Val.h
- /Users/u0858882/Desktop/msdscript/msdscript/src/Val.cpp

4.12 Var Class Reference

```
#include <Expr.h>
```

Inheritance diagram for Var:



Public Member Functions

- Var (std::string str)
- bool equals (Expr *e) override
- Val * interp () override
- bool has_variable () override
- Expr * subst (std::string str, Expr *e) override

Public Member Functions inherited from Expr

- std::string to_string ()
- std::string to_pretty_string ()
- virtual void pretty_print (std::ostream &stream)
- virtual void pretty_print_at (std::ostream &stream, precedence_t p, std::streampos &pos, bool paren)

Public Attributes

- std::string str_m

  The string value of the Var object.

Private Member Functions

- void print (std::ostream &stream) override

4.12.1 Constructor & Destructor Documentation

4.12.1.1 Var()

```
Var::Var (
             std::string str )   [explicit]
```

4.12.2 Member Function Documentation

4.12.2.1 equals()

```
bool Var::equals (
             Expr * e )  [override], [virtual]
```

Implements Expr.

4.12.2.2 has_variable()

```
bool Var::has_variable ( )  [override], [virtual]
```

Implements Expr.

4.12.2.3 interp()

```
Val * Var::interp ( )  [override], [virtual]
```

Implements Expr.

4.12.2.4 print()

```
void Var::print (
             std::ostream & stream )  [override], [private], [virtual]
```

Implements Expr.

4.12.2.5 subst()

```
Expr * Var::subst (
             std::string str,
             Expr * e )  [override], [virtual]
```

Implements Expr.

4.12.3 Member Data Documentation

4.12.3.1 str_m

```
std::string Var::str_m
```

The string value of the Var object.

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

- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.h
- /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.cpp

Chapter 5

File Documentation

5.1 /Users/u0858882/Desktop/msdscript/msdscript/src/cmdline.cpp File Reference

```
#include "catch.h"
#include "cmdline.h"
```

Macros

- #define CATCH_CONFIG_RUNNER /∗ Don't move either of these ∗/

Functions

- void if_help ()
- void if_test (char ∗∗argv)
- void if_interp ()
- void if_print ()
- void if_pretty_print ()
- void handle_cin (Expr ∗&e)
- int use_arguments (int argc, char ∗∗argv)

5.1.1 Macro Definition Documentation

5.1.1.1 CATCH_CONFIG_RUNNER

```
#define CATCH_CONFIG_RUNNER /∗ Don't move either of these ∗/
```

5.1.2 Function Documentation

5.1.2.1 handle_cin()

```
void handle_cin (
            Expr ∗& e )
```

std::cin helper

5.1.2.2 if_help()

```
void if_help ( )
```

Argument handling functions

5.1.2.3 if_interp()

```
void if_interp ( )
```

5.1.2.4 if_pretty_print()

```
void if_pretty_print ( )
```

5.1.2.5 if_print()

```
void if_print ( )
```

5.1.2.6 if_test()

```
void if_test (
             char ** argv )
```

5.1.2.7 use_arguments()

```
static int use_arguments (
             int argc,
             char ** argv )
```

5.2 /Users/u0858882/Desktop/msdscript/msdscript/src/cmdline.h File Reference

```
#include <iostream>
#include "catch.h"
#include "Expr.h"
#include "parse.h"
#include "Val.h"
```

Functions

- int use_arguments (int argc, char ∗∗argv)

5.2.1 Function Documentation

5.2.1.1 use_arguments()

```
int use_arguments (
            int argc,
            char ** argv )
```

5.3 cmdline.h

Go to the documentation of this file.

```
/*******************************************************************************
 * \brief Command line argument-handling functions declarations
 *
 * \file cmdline.h
 * \author Jake Dame
 ******************************************************************************/

#pragma once

#include<iostream> /* Console I/O */

#include "catch.h" /* Catch2 testing framework */

#include "Expr.h"
#include "parse.h"
#include "Val.h"

int use_arguments( int argc, char ** argv );
```

5.4 /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.cpp File Reference

```
#include "Expr.h"
#include "Val.h"
```

5.5 /Users/u0858882/Desktop/msdscript/msdscript/src/Expr.h File Reference

```
#include <iostream>
#include <stdexcept>
#include <sstream>
#include <utility>
```

Classes

- class Expr
- class Num
- class Bool
- class Eq
- class Add
- class Mult
- class Var
- class Let
- class If

Enumerations

- enum precedence_t { prec_none = 0 , prec_A = 1 , prec_M = 2 }

5.5.1 Enumeration Type Documentation

5.5.1.1 precedence_t

```
enum precedence_t
```

Enumerator

prec_none	default precedence for Num and Var
prec_A	default precedence for Add
prec_M	default precedence for Mult

5.6 Expr.h

Go to the documentation of this file.

```
/*****************************************************************************
 * \brief Expr bass class + derived class declarations
 *
 * \file Expr.h
 * \author Jake Dame
 *****************************************************************************/

#pragma once

#include <iostream>     /* Console I/O */
#include <stdexcept>    /* exceptions handling */
#include <sstream>      /* std::stringstream */
#include <utility>      /* std::move (for Var constructor) */

class Val;              /* Val class for Expr::interp() */

/*****************************************************************************
 * \typedef precedence_t
 * \brief Assists in nested/parenthetical expression precedence typing
 *****************************************************************************/
typedef enum
{
    prec_none = 0,
    prec_A = 1,
    prec_M = 2
} precedence_t;

/*****************************************************************************
 * \class Expr
 * \brief An abstract, base class representing a mathematical expression.
 *
 * The Expression class is an abstract class that defines regular and
 * virtual functions used to perform various mathematical operations. These
 * include basic number and variable expressions, as well as operational
 * expressions such as addition, multiplication, and let substitution. All
 * classes that inherit from the Expression class are also able to print as a
 * string in two different styles.
 *****************************************************************************/
class Expr
{
public:

    /*
     * Non-virtual methods
     */
    std::string to_string();

    std::string to_pretty_string();

    /*
     * Pure virtual methods
     */
    virtual bool equals( Expr * e ) = 0;

    virtual Val * interp() = 0;

    virtual bool has_variable() = 0;

    virtual Expr * subst( std::string str, Expr * e ) = 0;

    virtual void print( std::ostream & stream ) = 0;

    /*
     * Regular virtual methods
     */
    virtual void pretty_print( std::ostream & stream )
    {
        print( stream );
    }

    virtual void pretty_print_at( std::ostream & stream,
                                  precedence_t p,
                                  std::streampos & pos,
                                  bool paren )
    {
        pretty_print( stream );
    }
};

/*******************************************************************************
 * \class Num
 * \brief An Expr derived class representing a basic integer
 ******************************************************************************/
class Num : public Expr
{

public:

    int int_m;

    explicit Num( int val );

    bool equals( Expr * e ) override;

    Val * interp() override;

    bool has_variable() override;

    Expr * subst( std::string str, Expr * e ) override;

private:

    void print( std::ostream & stream ) override;
};

/*******************************************************************************
 * \class Bool
 * \brief An Expr derived class representing a basic boolean value
 ******************************************************************************/
class Bool : public Expr
{

public:

    bool bool_m;

    explicit Bool( bool val );

    bool equals( Expr * e ) override;

    Val * interp() override;

    bool has_variable() override;

    Expr * subst( std::string str, Expr * e ) override;

private:

    void print( std::ostream & stream ) override;
};

/*******************************************************************************
 * \class Eq
 * \brief An Expr derived class representing an equality operation/comparison
 *
 * The Eq class compares two other Expr objects, and checks for equality. Its
 * value is defined by a BoolVal object. E.g. "Expr == Expr". If the Expr
 * objects are equal, an Eq object will have a BoolVal of "_true". If they are
 * not, it will be "_false."
 *****************************************************************************/
class Eq : public Expr
{

public:

    Expr * lhs_m;
    Expr * rhs_m;

    Eq( Expr * lhs, Expr * rhs );

    bool equals( Expr * e ) override;

    Val * interp() override;

    bool has_variable() override;

    Expr * subst( std::string str, Expr * e ) override;

private:

    void print( std::ostream & stream ) override;

    void pretty_print( std::ostream & stream ) override;

    void pretty_print_at( std::ostream & stream,
                                  precedence_t p,
                                  std::streampos & pos,
                                  bool paren ) override;
};

/******************************************************************************
 * \class Add
 * \brief An Expr derived class representing an addition operation
 *
 * The Add class is constructed with two values: a left-hand side (lhs) value,
 * and a right-hand side (rhs) value. These values can be either a Number
 * (object), or a Variable (object), or another Add or Multiplication object
 * (nested). If both values are Numbers, they can be interpreted to be their
 * sum. The Add object supports Variable substitution, and precedence-based
 * printing of parentheses.
 *****************************************************************************/
class Add : public Expr
{

public:

    Expr * lhs_m;
    Expr * rhs_m;

    Add( Expr * lhs, Expr * rhs );

    bool equals( Expr * e ) override;

    Val * interp() override;

    bool has_variable() override;

    Expr * subst( std::string str, Expr * e ) override;

private:

    void print( std::ostream & stream ) override;

    void pretty_print( std::ostream & stream ) override;

    void pretty_print_at( std::ostream & stream,
                          precedence_t p,
                          std::streampos & pos,
                          bool paren ) override;
};

/******************************************************************************
 * \class Mult
 * \brief An Expr derived class representing a multiplication operation
 *
 * The Multiplication class is constructed with two values: a left-hand side
 * (lhs) value, and a right-hand side (rhs) value. These values can be either
 * a Number (object), or a Variable (object), or another Add or Multiplication
 * object (nested). If both values are Numbers, they can be interpreted to be
 * their product. The Multiplication object supports Variable substitution,
 * and precedence-based printing of parentheses.
 *****************************************************************************/
```

```
class Mult : public Expr
{

public:

    Expr * lhs_m;
    Expr * rhs_m;

    Mult( Expr * lhs, Expr * rhs );

    bool equals( Expr * e ) override;

    Val * interp() override;

    bool has_variable() override;

    Expr * subst( std::string str, Expr * e ) override;

private:

    void print( std::ostream & stream ) override;

    void pretty_print( std::ostream & stream ) override;

    void pretty_print_at( std::ostream & stream,
                          precedence_t p,
                          std::streampos & pos,
                          bool paren ) override;
};

/*******************************************************************************
 * \class Var
 * \brief An Expr derived class representing a string placeholder (variable)
 *
 * The Variable class is ultimately a representation of the value
 * of it's int_m member variable. It can be wrapped in other Expression
 * classes when performing operations such as addition and multiplication,
 * but cannot be interpreted to an integer value -- unless it is substituted.
 ******************************************************************************/
class Var : public Expr
{

public:

    std::string str_m;

    explicit Var( std::string str );

    bool equals( Expr * e ) override;

    Val * interp() override;

    bool has_variable() override;

    Expr * subst( std::string str, Expr * e ) override;

private:

    void print( std::ostream & stream ) override;
};

/*******************************************************************************
 * \class Let
 * \brief An Expr derived class supporting let binding
 *
 * The Let class allows for let binding, which allows for the declaration
 * of a variable and its definition with an Expression, within the scope of
 * Let's "body" Expression. This can be utilized in Expressions that have
 * variables to declare/define a variable, without calling any other functions
 * for substitution, etc. Example: _let x = 5 _in 3 * x
 ******************************************************************************/
class Let : public Expr
{

public:

    std::string lhs_m;
    Expr * rhs_m;
    Expr * body_m;

    Let( std::string lhs, Expr * rhs, Expr * body );

    bool equals( Expr * e ) override;

    Val * interp() override;

    bool has_variable() override;

    Expr * subst( std::string str, Expr * e ) override;

private:

    void print( std::ostream & stream ) override;

    void pretty_print( std::ostream & stream ) override;

    void pretty_print_at( std::ostream & stream,
                          precedence_t p,
                          std::streampos & caller_pos,
                          bool has_paren ) override;
};

/*******************************************************************************
 * \class If
 * \brief An Expr derived class representing a conditional operation expression
 *
 * An If object has a condition operand, and two branch operands (i.e. "then"
 * and "else"). It can "evaluate" the condition operand, and then embody a
 * value based on the result of that evaluation. If the condition has a BoolVal
 * of "_true", the If object will come to have a value equal to its then_m
 * operand; the opposite is true for the else_m operand.
 ******************************************************************************/
class If : public Expr
{

public:

    Expr * test_m;
    Expr * then_m;
    Expr * else_m;

    If( Expr * condition, Expr * first_branch, Expr * second_branch );

    bool equals( Expr * e ) override;

    Val * interp() override;

    bool has_variable() override;

    Expr * subst( std::string str, Expr * e ) override;

private:

    void print( std::ostream & stream ) override;

    void pretty_print( std::ostream & stream ) override;

    void pretty_print_at( std::ostream & stream,
                          precedence_t p,
                          std::streampos & caller_pos,
                          bool has_paren ) override;
};
```

5.7 /Users/u0858882/Desktop/msdscript/msdscript/src/main.cpp File Reference

```
#include "cmdline.h"
```

Functions

- int main (int argc, char **argv)

5.7.1 Function Documentation

5.7.1.1 main()

```
int main (
        int argc,
        char ** argv )
```

5.8 /Users/u0858882/Desktop/msdscript/msdscript/src/parse.cpp File Reference

```
#include "parse.h"
```

Functions

- Expr ∗ parse_expr (std::istream &stream)
- Expr ∗ parse_eqs (std::istream &stream)
- Expr ∗ parse_adds (std::istream &stream)
- Expr ∗ parse_mults (std::istream &stream)
- Expr ∗ parse_unary_and_ternary_exprs (std::istream &stream)
- Expr ∗ parse_num (std::istream &stream)
- Expr ∗ parse_bool (std::istream &stream)
- Expr ∗ parse_var (std::istream &stream)
- Expr ∗ parse_let (std::istream &stream)
- Expr ∗ parse_if (std::istream &stream)
- Expr ∗ parse_paren (std::istream &stream)
- int build_number (std::istream &stream)
- std::string peek_keyword (std::istream &stream)
- void consume (std::istream &stream, int expect)
- void consume (std::istream &stream, const std::string &str)
- void consume_whitespace (std::istream &stream)
- Expr ∗ parse_expr (const std::string &str)

5.8.1 Function Documentation

5.8.1.1 build_number()

```
int build_number (
            std::istream & stream )
```

5.8.1.2 consume() [1/2]

```
void consume (
            std::istream & stream,
            const std::string & str )
```

5.8.1.3 consume() [2/2]

```
void consume (
            std::istream & stream,
            int expect )
```

5.8.1.4 consume_whitespace()

```
void consume_whitespace (
            std::istream & stream )
```

5.8.1.5 parse_adds()

```
Expr * parse_adds (
            std::istream & stream )
```

5.8.1.6 parse_bool()

```
Expr * parse_bool (
            std::istream & stream )
```

5.8.1.7 parse_eqs()

```
Expr * parse_eqs (
            std::istream & stream )
```

5.8.1.8 parse_expr() [1/2]

```
Expr * parse_expr (
            const std::string & str )
```

5.8.1.9 parse_expr() [2/2]

```
Expr * parse_expr (
            std::istream & stream )
```

5.8.1.10 parse_if()

```
Expr * parse_if (
            std::istream & stream )
```

5.8.1.11 parse_let()

```
Expr * parse_let (
            std::istream & stream )
```

5.8.1.12 parse_mults()

```
Expr * parse_mults (
            std::istream & stream )
```

5.8.1.13 parse_num()

```
Expr * parse_num (
            std::istream & stream )
```

5.8.1.14 parse_paren()

```
Expr * parse_paren (
             std::istream & stream )
```

5.8.1.15 parse_unary_and_ternary_exprs()

```
Expr * parse_unary_and_ternary_exprs (
             std::istream & stream )
```

5.8.1.16 parse_var()

```
Expr * parse_var (
             std::istream & stream )
```

5.8.1.17 peek_keyword()

```
std::string peek_keyword (
             std::istream & stream )
```

5.9 /Users/u0858882/Desktop/msdscript/msdscript/src/parse.h File Reference

```
#include <iostream>
#include "Expr.h"
```

Functions

- Expr ∗ parse_expr (const std::string &str)

5.9.1 Function Documentation

5.9.1.1 parse_expr()

```
Expr * parse_expr (
             const std::string & str )
```

5.10 parse.h

Go to the documentation of this file.

```
/*******************************************************************************
 * \brief Parsing functions declarations
 *
 * \file parse.h
 * \author Jake Dame
 ******************************************************************************/

#pragma once

#include <iostream> /* Console I/O */

#include "Expr.h"

Expr * parse_expr( const std::string & str );
```

5.11 /Users/u0858882/Desktop/msdscript/msdscript/src/Val.cpp File Reference

```
#include "Expr.h"
#include "Val.h"
```

5.12 /Users/u0858882/Desktop/msdscript/msdscript/src/Val.h File Reference

Classes

- class Val
- class NumVal
- class BoolVal

5.13 Val.h

Go to the documentation of this file.

```
/*******************************************************************************
 * \brief Val bass class + derived class declarations
 *
 * \file Val.h
 * \author Jake Dame
 ******************************************************************************/

#pragma once

class Expr; /* Expr class for Val::to_expr() */

/*******************************************************************************
 * \class Val
 * \brief An abstract, base class representing a the value of an expression
 *
 * The Val class is an abstract class that defines regular and
 * virtual functions that can be used to represent two mathematical
 * values -- boolean values (BoolVal class) and integer values (NumVal class)
 * -- of mathematical expressions (Expr class in Expr.h); calling interp() on
 * an Expr object returns a Val object.
 *
 * The Val class  handles base-level addition ( add_to() )and multiplication
 * ( mult_with() ); it supports comparison between Val objects ( equals() ),
 * and conversion to analogous Expr objects as well ( to_expr() ).
 ******************************************************************************/
class Val
{
public:

    /*
     * Non-virtual methods
     */
    std::string to_string();

    /*
     * Pure virtual methods
     */
    virtual Expr * to_expr() = 0;
    virtual bool equals( Val * v ) = 0;
    virtual Val * add_to( Val * v ) = 0;
    virtual Val * mult_with( Val * v ) = 0;
    virtual bool is_true() = 0;
    virtual void print( std::ostream & stream ) = 0;
};

/*******************************************************************************
 * \class NumVal
 * \brief A Val derived class representing an integer value
 *
 * A NumVal object represents an integer value of a mathematical expression.
 ******************************************************************************/
class NumVal : public Val
{
public:
    int int_m;

    explicit NumVal( int val );

    Expr * to_expr() override;
    bool equals( Val * v ) override;
    Val * add_to( Val * other_val ) override;
    Val * mult_with( Val * other_val ) override;
    bool is_true() override;
    void print( std::ostream & ostream ) override;
};

/*******************************************************************************
 * \class BoolVal
 * \brief A Val derived class representing a boolean value
 *
 * A BoolVal object represents a boolean value of a mathematical expression.
 ******************************************************************************/
class BoolVal : public Val
{
public:
    bool bool_m;

    explicit BoolVal( bool val );

    Expr * to_expr() override;
    bool equals( Val * v ) override;
    Val * add_to( Val * other_val ) override;
    Val * mult_with( Val * other_val ) override;
    bool is_true() override;
    void print( std::ostream & ostream ) override;
};
```

5.14 /Users/u0858882/Desktop/msdscript/test_msdscript/src/test_msdscript.cpp File Reference

```
#include <ctime>
#include <iostream>
#include <fstream>
#include <sstream>
#include "exec.h"
```

Functions

- void write_report_header (std::ofstream &output_file, const std::string &exec_name, const int &error_count)

- void write_results (std::stringstream &stream, const std::string &label, const std::vector< ExecResult > &exec_results, const std::string &input, const int &iteration, int &error_count)
- void compare_IO (const std::string &exec_name)
- void compare_programs (const std::string &exec_name_1, const std::string &exec_name_2)
- int main (int argc, char **argv)
- void write_results (std::stringstream &stream, const std::string &label, const ExecResult &er, const std::string &input, int &error_count)

Variables

- const std::string EXECS_DIR

  Directory where the executables to test are. USE ABS PATH.
- const std::string OUTPUT_DIR

  Directory where error reports (.txt) should go. USE ABS PATH.
- const int TEST_ITER = 30

  Number of inputs to test per executable.

5.14.1 Function Documentation

5.14.1.1 compare_IO()

```
void compare_IO (
             const std::string & exec_name )
```

Testing functions

5.14.1.2 compare_programs()

```
void compare_programs (
             const std::string & exec_name_1,
             const std::string & exec_name_2 )
```

5.14.1.3 main()

```
int main (
             int argc,
             char ** argv )
```

5.14.1.4 write_report_header()

```
void write_report_header (
             std::ofstream & output_file,
             const std::string & exec_name,
             const int & error_count )
```

Error-report-building functions

5.14.1.5 write_results() [1/2]

```
void write_results (
            std::stringstream & stream,
            const std::string & label,
            const ExecResult & er,
            const std::string & input,
            int & error_count )
```

5.14.1.6 write_results() [2/2]

```
void write_results (
            std::stringstream & stream,
            const std::string & label,
            const std::vector< ExecResult > & exec_results,
            const std::string & input,
            const int & iteration,
            int & error_count )
```

5.14.2 Variable Documentation

5.14.2.1 EXECS_DIR

```
const std::string EXECS_DIR
```

Initial value:

```
=
       "/Users/u0858882/Desktop/msdscript/test_msdscript/testers/"
```

Directory where the executables to test are. USE ABS PATH.

CONSTANTS

5.14.2.2 OUTPUT_DIR

```
const std::string OUTPUT_DIR
```

Initial value:

```
=
       "/Users/u0858882/Desktop/msdscript/test_msdscript/reports/"
```

Directory where error reports (.txt) should go. USE ABS PATH.

5.14.2.3 TEST_ITER

```
const int TEST_ITER = 30
```

Number of inputs to test per executable.

Index

/Users/u0858882/Desktop/msdscript/msdscript/src/Expr.cpp, 37
/Users/u0858882/Desktop/msdscript/msdscript/src/Expr.h, 37, 38
/Users/u0858882/Desktop/msdscript/msdscript/src/Val.cpp, 46
/Users/u0858882/Desktop/msdscript/msdscript/src/Val.h, 46
/Users/u0858882/Desktop/msdscript/msdscript/src/cmdline.cpp, 35
/Users/u0858882/Desktop/msdscript/msdscript/src/cmdline.h, 36, 37
/Users/u0858882/Desktop/msdscript/msdscript/src/main.cpp, 42
/Users/u0858882/Desktop/msdscript/msdscript/src/parse.cpp, 43
/Users/u0858882/Desktop/msdscript/msdscript/src/parse.h, 45, 46
/Users/u0858882/Desktop/msdscript/test_msdscript/src/test_msdscript.cpp, 47

Add, 7
- Add, 8
- equals, 8
- has_variable, 8
- interp, 8
- lhs_m, 9
- pretty_print, 8
- pretty_print_at, 8
- print, 9
- rhs_m, 9
- subst, 9

add_to
- BoolVal, 12
- NumVal, 29
- Val, 31

body_m
- Let, 23

Bool, 10
- Bool, 10
- bool_m, 11
- equals, 11
- has_variable, 11
- interp, 11
- print, 11
- subst, 11

bool_m
- Bool, 11
- BoolVal, 13

BoolVal, 12
- add_to, 12
- bool_m, 13
- BoolVal, 12
- equals, 12
- is_true, 13
- mult_with, 13
- print, 13
- to_expr, 13

build_number
- parse.cpp, 43

CATCH_CONFIG_RUNNER
- cmdline.cpp, 35

cmdline.cpp
- CATCH_CONFIG_RUNNER, 35
- handle_cin, 35
- if_help, 35
- if_interp, 36
- if_pretty_print, 36
- if_print, 36
- if_test, 36
- use_arguments, 36

cmdline.h
- use_arguments, 37

compare_IO
- test_msdscript.cpp, 48

compare_programs
- test_msdscript.cpp, 48

consume
- parse.cpp, 43

consume_whitespace
- parse.cpp, 43

else_m
- If, 20

Eq, 14
- Eq, 14
- equals, 15
- has_variable, 15
- interp, 15
- lhs_m, 16
- pretty_print, 15
- pretty_print_at, 15
- print, 15
- rhs_m, 16
- subst, 15

equals
- Add, 8
- Bool, 11

BoolVal, 12
Eq, 15
Expr, 17
If, 19
Let, 22
Mult, 25
Num, 27
NumVal, 29
Val, 31
Var, 33
EXECS_DIR
test_msdscript.cpp, 49
Expr, 16
equals, 17
has_variable, 17
interp, 17
pretty_print, 17
pretty_print_at, 17
print, 17
subst, 18
to_pretty_string, 18
to_string, 18
Expr.h
prec_A, 38
prec_M, 38
prec_none, 38
precedence_t, 38

handle_cin
cmdline.cpp, 35
has_variable
Add, 8
Bool, 11
Eq, 15
Expr, 17
If, 19
Let, 22
Mult, 25
Num, 27
Var, 33

If, 18
else_m, 20
equals, 19
has_variable, 19
If, 19
interp, 19
pretty_print, 20
pretty_print_at, 20
print, 20
subst, 20
test_m, 20
then_m, 21
if_help
cmdline.cpp, 35
if_interp
cmdline.cpp, 36
if_pretty_print
cmdline.cpp, 36
if_print
cmdline.cpp, 36
if_test
cmdline.cpp, 36
int_m
Num, 28
NumVal, 30
interp
Add, 8
Bool, 11
Eq, 15
Expr, 17
If, 19
Let, 22
Mult, 25
Num, 27
Var, 33
is_true
BoolVal, 13
NumVal, 29
Val, 31

Let, 21
body_m, 23
equals, 22
has_variable, 22
interp, 22
Let, 22
lhs_m, 23
pretty_print, 22
pretty_print_at, 22
print, 23
rhs_m, 23
subst, 23
lhs_m
Add, 9
Eq, 16
Let, 23
Mult, 26

main
main.cpp, 42
test_msdscript.cpp, 48
main.cpp
main, 42
Mult, 24
equals, 25
has_variable, 25
interp, 25
lhs_m, 26
Mult, 24
pretty_print, 25
pretty_print_at, 25
print, 25
rhs_m, 26
subst, 25
mult_with
BoolVal, 13
NumVal, 29

Val, 31

Num, 26
equals, 27
has_variable, 27
int_m, 28
interp, 27
Num, 27
print, 27
subst, 28
NumVal, 28
add_to, 29
equals, 29
int_m, 30
is_true, 29
mult_with, 29
NumVal, 29
print, 29
to_expr, 30

OUTPUT_DIR
test_msdscript.cpp, 49

parse.cpp
build_number, 43
consume, 43
consume_whitespace, 43
parse_adds, 43
parse_bool, 44
parse_eqs, 44
parse_expr, 44
parse_if, 44
parse_let, 44
parse_mults, 44
parse_num, 44
parse_paren, 44
parse_unary_and_ternary_exprs, 45
parse_var, 45
peek_keyword, 45
parse.h
parse_expr, 45
parse_adds
parse.cpp, 43
parse_bool
parse.cpp, 44
parse_eqs
parse.cpp, 44
parse_expr
parse.cpp, 44
parse.h, 45
parse_if
parse.cpp, 44
parse_let
parse.cpp, 44
parse_mults
parse.cpp, 44
parse_num
parse.cpp, 44
parse_paren
parse.cpp, 44
parse_unary_and_ternary_exprs
parse.cpp, 45
parse_var
parse.cpp, 45
peek_keyword
parse.cpp, 45
prec_A
Expr.h, 38
prec_M
Expr.h, 38
prec_none
Expr.h, 38
precedence_t
Expr.h, 38
pretty_print
Add, 8
Eq, 15
Expr, 17
If, 20
Let, 22
Mult, 25
pretty_print_at
Add, 8
Eq, 15
Expr, 17
If, 20
Let, 22
Mult, 25
print
Add, 9
Bool, 11
BoolVal, 13
Eq, 15
Expr, 17
If, 20
Let, 23
Mult, 25
Num, 27
NumVal, 29
Val, 31
Var, 33

rhs_m
Add, 9
Eq, 16
Let, 23
Mult, 26

str_m
Var, 33
subst
Add, 9
Bool, 11
Eq, 15
Expr, 18
If, 20
Let, 23
Mult, 25

Num, 28
Var, 33

TEST_ITER
test_msdscript.cpp, 49
test_m
If, 20
test_msdscript.cpp
compare_IO, 48
compare_programs, 48
EXECS_DIR, 49
main, 48
OUTPUT_DIR, 49
TEST_ITER, 49
write_report_header, 48
write_results, 48, 49
then_m
If, 21
to_expr
BoolVal, 13
NumVal, 30
Val, 31
to_pretty_string
Expr, 18
to_string
Expr, 18
Val, 31

use_arguments
cmdline.cpp, 36
cmdline.h, 37

Val, 30
add_to, 31
equals, 31
is_true, 31
mult_with, 31
print, 31
to_expr, 31
to_string, 31
Var, 32
equals, 33
has_variable, 33
interp, 33
print, 33
str_m, 33
subst, 33
Var, 32

write_report_header
test_msdscript.cpp, 48
write_results
test_msdscript.cpp, 48, 49