MSDScript

Generated by Doxygen 1.9.6

Chapter 1

MSDScript

Author

Yue Sun

Date

02-07-2023

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 5

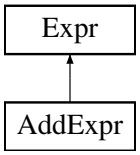
Class Documentation

5.1 AddExpr Class Reference

AddExpr class - for add expressions.

`#include <expr.hpp>`

Inheritance diagram for AddExpr:



Public Member Functions

- AddExpr (Expr *left, Expr *right)
- AddExpr (int left, int right)
- AddExpr (std::string left, int right)
- AddExpr (int left, std::string right)
- AddExpr (std::string left, std::string right)
- bool equals (Expr *expr)
- int interp ()
- bool has_variable ()
- Expr * subst (std::string s, Expr *expr)
- void print (std::ostream &out)
- void pretty_print (std::ostream &out)

Public Member Functions inherited from Expr

- virtual bool equals (Expr *expr)=0
- virtual int interp ()=0
- virtual bool has_variable ()=0
- virtual Expr * subst (std::string s, Expr *expr)=0
- virtual void print (std::ostream &out)=0
- virtual void pretty_print (std::ostream &out)=0
- std::string to_string ()
- precedence_t **get_precedence** ()
- std::string to_pretty_string ()

Additional Inherited Members

Public Attributes inherited from Expr

- precedence_t **prec**

5.1.1 Detailed Description

AddExpr class - for add expressions.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 AddExpr() [1/5]

```
AddExpr::AddExpr (
            Expr * left,
            Expr * right )
```

Constructs a AddExpr object with the specified left and right expressions.

Parameters

left	the specified left expression
right	the specified right expression

5.1.2.2 AddExpr() [2/5]

```
AddExpr::AddExpr (
            int left,
            int right )
```

Constructs a AddExpr object with the specified int values.

Parameters

left	the specified int value in the left expression
right	the specified int value in the right expression

5.1.2.3 AddExpr() [3/5]

```
AddExpr::AddExpr (
            std::string left,
            int right )
```

Constructs a AddExpr object with a specified string as the left expression and a specified int value as the right expression.

Parameters

left	the specified string in the left expression
right	the specified int value in the right expression

5.1.2.4 AddExpr() [4/5]

```
AddExpr::AddExpr (
            int left,
            std::string right )
```

Constructs a AddExpr object with a specified int as the left expression and a specified string value as the right expression.

Parameters

left	the specified int value in the left expression
right	the specified string in the right expression

5.1.2.5 AddExpr() [5/5]

```
AddExpr::AddExpr (
            std::string left,
            std::string right )
```

Constructs a AddExpr object with the specified strings.

Parameters

left	the specified string in the left VarExpr expression
right	the specified string in the right VarExpr expression

5.1.3 Member Function Documentation

5.1.3.1 equals()

```
bool AddExpr::equals (
            Expr * expr )  [virtual]
```

Compared the specified expression with this object for equality.

Parameters

expr	the expression to be compared for equality with this object

Returns

true if the specified expression is a AddExpr object and it has the same value with this object. Otherwise returns false.

Implements Expr.

5.1.3.2 has_variable()

```
bool AddExpr::has_variable ( )  [virtual]
```

Check if this AddExpr expression object contains a VarExpr object.

Returns

true if either its left or right expression contains a VarExpr object

Implements Expr.

5.1.3.3 interp()

```
int AddExpr::interp ( )  [virtual]
```

Add the left and right expressions and returns the int value of the result.

Returns

the int value of the addition of the left and right expressions

Implements Expr.

5.1.3.4 pretty_print()

```
void AddExpr::pretty_print (
            std::ostream & out )  [virtual]
```

Print this object in a prettier way. To specify, we'll omit redundant parentheses and add a space around operators.

Parameters

out	the output stream used to print this object

Implements Expr.

5.1.3.5 print()

```
void AddExpr::print (
            std::ostream & out )  [virtual]
```

Print this object with parentheses and a add sign.

Parameters

out	the output stream used to print this object

Implements Expr.

5.1.3.6 subst()

```
Expr * AddExpr::subst (
            std::string s,
            Expr * expr )  [virtual]
```

Substitute the specified string in this object with a specified expression

Parameters

s	the specified string to be substituted
expr	the specified expression used to substitute the string

Returns

a new AddExpr object with the specified strings in its left and right expressions have been substituted

Implements Expr.

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

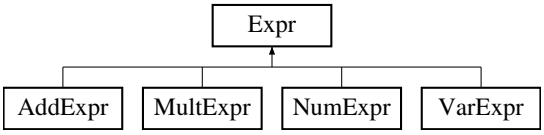
- /Users/sonia/Documents/MSD/CS6015GitRepo/MSDScript/expr.hpp
- /Users/sonia/Documents/MSD/CS6015GitRepo/MSDScript/expr.cpp

5.2 Expr Class Reference

Expr class - the base class for all expressions.

`#include <expr.hpp>`

Inheritance diagram for Expr:



Public Member Functions

- virtual bool equals (Expr ∗expr)=0
- virtual int interp ()=0
- virtual bool has_variable ()=0
- virtual Expr ∗ subst (std::string s, Expr ∗expr)=0
- virtual void print (std::ostream &out)=0
- virtual void pretty_print (std::ostream &out)=0
- std::string to_string ()
- precedence_t **get_precedence** ()
- std::string to_pretty_string ()

Public Attributes

- precedence_t **prec**

5.2.1 Detailed Description

Expr class - the base class for all expressions.

5.2.2 Member Function Documentation

5.2.2.1 equals()

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

Implemented in NumExpr, AddExpr, MultExpr, and VarExpr.

5.2.2.2 has_variable()

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

Implemented in NumExpr, AddExpr, MultExpr, and VarExpr.

5.2.2.3 interp()

```
virtual int Expr::interp ( )  [pure virtual]
```

Implemented in NumExpr, AddExpr, MultExpr, and VarExpr.

5.2.2.4 pretty_print()

```
virtual void Expr::pretty_print (
            std::ostream & out )  [pure virtual]
```

Implemented in NumExpr, AddExpr, MultExpr, and VarExpr.

5.2.2.5 print()

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

Implemented in NumExpr, AddExpr, MultExpr, and VarExpr.

5.2.2.6 subst()

```
virtual Expr * Expr::subst (
            std::string s,
            Expr * expr )  [pure virtual]
```

Implemented in NumExpr, AddExpr, MultExpr, and VarExpr.

5.2.2.7 to_pretty_string()

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

Store the result of pretty_print method into a string.

Returns

the string representation of the pretty_print method

5.2.2.8 to_string()

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

Store the result of print method into a string.

Returns

the string representation of the print method

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

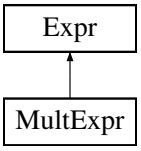
- /Users/sonia/Documents/MSD/CS6015GitRepo/MSDScript/expr.hpp
- /Users/sonia/Documents/MSD/CS6015GitRepo/MSDScript/expr.cpp

5.3 MultExpr Class Reference

MultExpr class - for multiplication expressions.

```
#include <expr.hpp>
```

Inheritance diagram for MultExpr:



Public Member Functions

- MultExpr (Expr *left, Expr *right)
- MultExpr (int left, int right)
- MultExpr (std::string left, int right)
- MultExpr (int left, std::string right)
- MultExpr (std::string left, std::string right)
- bool equals (Expr *expr)
- int interp ()
- bool has_variable ()
- Expr * subst (std::string s, Expr *expr)
- void print (std::ostream &out)
- void pretty_print (std::ostream &out)

Public Member Functions inherited from Expr

- virtual bool equals (Expr ∗expr)=0
- virtual int interp ()=0
- virtual bool has_variable ()=0
- virtual Expr ∗ subst (std::string s, Expr ∗expr)=0
- virtual void print (std::ostream &out)=0
- virtual void pretty_print (std::ostream &out)=0
- std::string to_string ()
- precedence_t **get_precedence** ()
- std::string to_pretty_string ()

Public Attributes

- Expr ∗ **lhs**
- Expr ∗ **rhs**

Public Attributes inherited from Expr

- precedence_t **prec**

5.3.1 Detailed Description

MultExpr class - for multiplication expressions.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 MultExpr() [1/5]

```
MultExpr::MultExpr (
            Expr * left,
            Expr * right )
```

Constructs a MultExpr object with the specified left and right expressions.

Parameters

left	the specified left expression
right	the specified right expression

5.3.2.2 MultExpr() [2/5]

```
MultExpr::MultExpr (
```

```
            int left,
            int right )
```

Constructs a MultExpr object with the specified int values.

Parameters

left	the specified int value in the left expression
right	the specified int value in the right expression

5.3.2.3 MultExpr() [3/5]

```
MultExpr::MultExpr (
            std::string left,
            int right )
```

Constructs a MultExpr object with a specified string as the left expression and a specified int value as the right expression.

Parameters

left	the specified string in the left expression
right	the specified int value in the right expression

5.3.2.4 MultExpr() [4/5]

```
MultExpr::MultExpr (
            int left,
            std::string right )
```

Constructs a MultExpr object with a specified int as the left expression and a specified string value as the right expression.

Parameters

left	the specified int value in the left expression
right	the specified string in the right expression

5.3.2.5 MultExpr() [5/5]

```
MultExpr::MultExpr (
            std::string left,
            std::string right )
```

Constructs a MultExpr object with the specified strings.

Parameters

left	the specified string in the left VarExpr expression
right	the specified string in the right VarExpr expression

5.3.3 Member Function Documentation

5.3.3.1 equals()

```
bool MultExpr::equals (
            Expr * expr )  [virtual]
```

Compared the specified expression with this object for equality.

Parameters

expr	the expression to be compared for equality with this object

Returns

true if the specified expression is a MultExpr object and it has the same value with this object. Otherwise returns false.

Implements Expr.

5.3.3.2 has_variable()

```
bool MultExpr::has_variable ( )  [virtual]
```

Check if this MultExpr expression object contains a VarExpr object.

Returns

true if either its left or right expression contains a VarExpr object

Implements Expr.

5.3.3.3 interp()

```
int MultExpr::interp ( )  [virtual]
```

Multiply the left and right expressions and returns the int value of the result.

Returns

the int value of the multiplication of the left and right expressions

Implements Expr.

5.3.3.4 pretty_print()

```
void MultExpr::pretty_print (
            std::ostream & out )  [virtual]
```

Print this object in a prettier way. To specify, we'll omit redundant parentheses and add a space around operators.

Parameters

out	the output stream used to print this object

Implements Expr.

5.3.3.5 print()

```
void MultExpr::print (
            std::ostream & out )  [virtual]
```

Print this object with parentheses and a add sign.

Parameters

out	the output stream used to print this object

Implements Expr.

5.3.3.6 subst()

```
Expr * MultExpr::subst (
            std::string s,
            Expr * expr )  [virtual]
```

Substitute the specified string in this object with a specified expression

Parameters

s	the specified string to be substituted
expr	the specified expression used to substitute the string

Returns

a new MultExpr object with the specified strings in its left and right expressions have been substituted

Implements Expr.

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

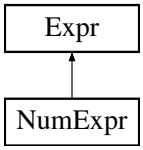
- /Users/sonia/Documents/MSD/CS6015GitRepo/MSDScript/expr.hpp
- /Users/sonia/Documents/MSD/CS6015GitRepo/MSDScript/expr.cpp

5.4 NumExpr Class Reference

NumExpr class - for number expressions.

`#include <expr.hpp>`

Inheritance diagram for NumExpr:



Public Member Functions

- NumExpr (int v)
- bool equals (Expr ∗expr)
- int interp ()
- bool has_variable ()
- Expr ∗ subst (std::string s, Expr ∗expr)
- void print (std::ostream &out)
- void pretty_print (std::ostream &out)

Public Member Functions inherited from Expr

- virtual bool equals (Expr ∗expr)=0
- virtual int interp ()=0
- virtual bool has_variable ()=0
- virtual Expr ∗ subst (std::string s, Expr ∗expr)=0
- virtual void print (std::ostream &out)=0
- virtual void pretty_print (std::ostream &out)=0
- std::string to_string ()
- precedence_t **get_precedence** ()
- std::string to_pretty_string ()

Additional Inherited Members

Public Attributes inherited from Expr

- precedence_t **prec**

5.4.1 Detailed Description

NumExpr class - for number expressions.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 NumExpr()

```
NumExpr::NumExpr (
            int v )
```

Constructs a NumExpr with the specified value and the precedence 0.

Parameters

v	the specified value

5.4.3 Member Function Documentation

5.4.3.1 equals()

```
bool NumExpr::equals (
            Expr * expr )  [virtual]
```

Compared the specified expression with this object for equality.

Parameters

expr	the expression to be compared for equality with this object

Returns

true if the specified expression is a NumExpr object and it has the same value with this object. Otherwise returns false.

Implements Expr.

5.4.3.2 has_variable()

```
bool NumExpr::has_variable ( )  [virtual]
```

Check if this NumExpr expression object contains a VarExpr object.

Returns

false since a NumExpr object won't have a varibale

Implements Expr.

5.4.3.3 interp()

```
int NumExpr::interp ( )  [virtual]
```

Evaluated the expression and returns the int value.

Returns

the int value of the member variable val

Implements Expr.

5.4.3.4 pretty_print()

```
void NumExpr::pretty_print (
            std::ostream & out )  [virtual]
```

Print this object in a prettier way. To specify, we'll omit redundant parentheses and add space around operators.

Parameters

out	the output stream used to print this object

Implements Expr.

5.4.3.5 print()

```
void NumExpr::print (
            std::ostream & out )  [virtual]
```

Print this object.

Parameters

out	the output stream used to print this object

Implements Expr.

5.4.3.6 subst()

```
Expr * NumExpr::subst (
            std::string s,
            Expr * expr )  [virtual]
```

Substitute the specified string in this object with a specified expression

Parameters

s	the specified string to be substituted
expr	the specified expression used to substitute the string

Returns

a new NumExpr object with the same value as this object

Implements Expr.

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

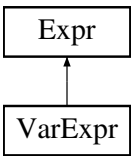
- /Users/sonia/Documents/MSD/CS6015GitRepo/MSDScript/expr.hpp
- /Users/sonia/Documents/MSD/CS6015GitRepo/MSDScript/expr.cpp

5.5 VarExpr Class Reference

VarExpr class - for variable expressions.

```
#include <expr.hpp>
```

Inheritance diagram for VarExpr:



Public Member Functions

- VarExpr (std::string s)
- bool equals (Expr *expr)
- int interp ()
- bool has_variable ()
- Expr * subst (std::string s, Expr *expr)
- void print (std::ostream &out)
- void pretty_print (std::ostream &out)

Public Member Functions inherited from Expr

- virtual bool equals (Expr *expr)=0
- virtual int interp ()=0
- virtual bool has_variable ()=0
- virtual Expr * subst (std::string s, Expr *expr)=0
- virtual void print (std::ostream &out)=0
- virtual void pretty_print (std::ostream &out)=0
- std::string to_string ()
- precedence_t **get_precedence** ()
- std::string to_pretty_string ()

Additional Inherited Members

Public Attributes inherited from Expr

- precedence_t **prec**

5.5.1 Detailed Description

VarExpr class - for variable expressions.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 VarExpr()

```
VarExpr::VarExpr (
            std::string s )
```

Constructs a NumExpr with the specified string and the precedence 0.

Parameters

s	the specified string

5.5.3 Member Function Documentation

5.5.3.1 equals()

```
bool VarExpr::equals (
            Expr * expr )  [virtual]
```

Compared the specified expression with this object for equality.

Parameters

expr	the expression to be compared for equality with this object

Returns

true if the specified expression is a VarExpr object and it has the same value with this object. Otherwise returns false.

Implements Expr.

5.5.3.2 has_variable()

```
bool VarExpr::has_variable ( )  [virtual]
```

Check if this VarExpr expression object contains a VarExpr object.

Returns

true because a VarExpr object always contains a VarExpr object

Implements Expr.

5.5.3.3 interp()

```
int VarExpr::interp ( )  [virtual]
```

Returns the int value of this VarExpr object

Returns

throw a runtime error since a varibale cannot be evaluated to a int value

Implements Expr.

5.5.3.4 pretty_print()

```
void VarExpr::pretty_print (
            std::ostream & out )  [virtual]
```

Print this object in a prettier way.

Parameters

out	the output stream used to print this object

Implements Expr.

5.5.3.5 print()

```
void VarExpr::print (
            std::ostream & out )  [virtual]
```

Print this object.

Parameters

out	the output stream used to print this object

Implements Expr.

5.5.3.6 subst()

```
Expr * VarExpr::subst (
            std::string s,
            Expr * expr )  [virtual]
```

Substitute the specified string in this object with a specified expression

Parameters

s	the specified string to be substituted
expr	the specified expression used to substitute the string

Returns

the specified expression if s equals val, otherwise return a new VarExpr object with the same val as this object

Implements Expr.

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

- /Users/sonia/Documents/MSD/CS6015GitRepo/MSDScript/expr.hpp
- /Users/sonia/Documents/MSD/CS6015GitRepo/MSDScript/expr.cpp

Chapter 6

File Documentation

6.1 /Users/sonia/Documents/MSD/CS6015GitRepo/MSDScript/cmdline.h

```
00001 //
00002 //  cmdline.h
00003 //  hw1
00004 //
00005 //  Created by Yue Sun on 1/12/23.
00006 //
00007
00008 #ifndef cmdline_h
00009 #define cmdline_h
00010
00011 void use_arguments(int argc, const char* argv[]);
00012
00013 #endif /* cmdline_h */
```

6.2 /Users/sonia/Documents/MSD/CS6015GitRepo/MSDScript/expr.hpp

```
00001 //
00002 //  expr.hpp
00003 //  HW2_Expression
00004 //
00005 //  Created by Yue Sun on 1/12/23.
00006 //
00007
00008 #ifndef expr_hpp
00009 #define expr_hpp
00010
00011 #include <sstream>
00012 #include <string>
00013
00014 enum precedence_t {
00015     prec_none, // 0
00016     prec_add, // 1
00017     prec_mult, // 2
00018 };
00019
00023 class Expr {
00024 public:
00025     precedence_t prec; // !< the precedence of an expression
00026     virtual bool equals(Expr* expr)=0;
00027     virtual int interp() = 0;
00028     virtual bool has_variable() = 0;
00029     virtual Expr* subst(std::string s, Expr* expr) = 0;
00030     virtual void print(std::ostream& out) = 0;
00031     virtual void pretty_print(std::ostream& out) = 0;
00032     std::string to_string();
00033     precedence_t get_precedence() {
00034         return prec;
00035     }
00036     std::string to_pretty_string();
00037 };
00038
00042 class NumExpr : public Expr {
00043 private:
00044     int val;
```

```
00045 public:
00046     NumExpr(int v);
00047     bool equals(Expr* expr);
00048     int interp();
00049     bool has_variable();
00050     Expr* subst(std::string s, Expr* expr);
00051     void print(std::ostream& out);
00052     void pretty_print(std::ostream& out);
00053 };
00054
00058 class AddExpr : public Expr {
00059 private:
00060     Expr* lhs;
00061     Expr* rhs;
00062 public:
00063     AddExpr(Expr *left, Expr *right);
00064     AddExpr(int left, int right);
00065     AddExpr(std::string left, int right);
00066     AddExpr(int left, std::string right);
00067     AddExpr(std::string left, std::string right);
00068     bool equals(Expr* expr);
00069     int interp();
00070     bool has_variable();
00071     Expr* subst(std::string s, Expr* expr);
00072     void print(std::ostream& out);
00073     void pretty_print(std::ostream& out);
00074 };
00075
00079 class MultExpr : public Expr {
00080 public:
00081     Expr* lhs;
00082     Expr* rhs;
00083 public:
00084     MultExpr(Expr *left, Expr *right);
00085     MultExpr(int left, int right);
00086     MultExpr(std::string left, int right);
00087     MultExpr(int left, std::string right);
00088     MultExpr(std::string left, std::string right);
00089     bool equals(Expr* expr);
00090     int interp();
00091     bool has_variable();
00092     Expr* subst(std::string s, Expr* expr);
00093     void print(std::ostream& out);
00094     void pretty_print(std::ostream& out);
00095 };
00096
00100 class VarExpr : public Expr {
00101 private:
00102     std::string val;
00103 public:
00104     VarExpr(std::string s);
00105     bool equals(Expr* expr);
00106     int interp();
00107     bool has_variable();
00108     Expr* subst(std::string s, Expr* expr);
00109     void print(std::ostream& out);
00110     void pretty_print(std::ostream& out);
00111 };
00112
00113 #endif /* expr_hpp */
```

Index