MSDscript

Generated by Doxygen 1.9.6

Chapter 1

MSDScript

Author

Lakshay Santosh Kucheriya

Date

Feburary 7, 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

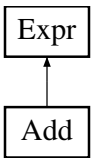
Class Documentation

5.1 Add Class Reference

Derived Class from Expr for representing a expression involving addition.

`#include <expr.hpp>`

Inheritance diagram for Add:



Public Member Functions

- **Add** (Expr *lhs, Expr *rhs)

  Constructor.
- bool equals (Expr *e)

  Checks for the equality between the left hand side and the right hand side.
- int interp ()

  This function interprets the value of the expression.
- bool has_variable ()

  This function determines if the the expression consists of a variable or not.
- Expr * subst (std::string s, Expr *e)

  This function substitutes the expression with the combination of sub-expressions if possible.
- void print (std::ostream &out)

  Prints the expression.
- void pretty_print (std::ostream &out)

  Prints the expression with more clarity.

Public Member Functions inherited from Expr

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

Public Attributes

- Expr ∗ **lhs**

  Left hand side Expression.
- Expr ∗ **rhs**

  Right hand side Expression.

5.1.1 Detailed Description

Derived Class from Expr for representing a expression involving addition.

5.1.2 Member Function Documentation

5.1.2.1 equals()

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

Checks for the equality between the left hand side and the right hand side.

Parameters

e	Expression

Returns

boolean value of LHS = RHS

Implements Expr.

5.1.2.2 has_variable()

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

This function determines if the the expression consists of a variable or not.

Returns

boolean value for the expression has a variable or not

Implements Expr.

5.1.2.3 interp()

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

This function interprets the value of the expression.

Returns

integer value of the computed expression

Implements Expr.

5.1.2.4 pretty_print()

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

Prints the expression with more clarity.

Parameters

out	*out*	output stream

Implements Expr.

5.1.2.5 print()

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

Prints the expression.

Parameters

out	*out*	output stream

Implements Expr.

5.1.2.6 subst()

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

This function substitutes the expression with the combination of sub-expressions if possible.

Parameters

s	first argument, string which can be replaced as a part of the expression
e	second argument, Expression

Returns

Expression which is modified if combination of sub expressions was possible

Implements Expr.

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

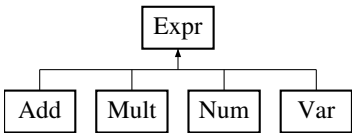
- /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.hpp
- /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.cpp

5.2 Expr Class Reference

Base Class for representing a expression.

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

Inheritance diagram for Expr:



Public Member Functions

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

5.2.1 Detailed Description

Base Class for representing a expression.

5.2.2 Member Function Documentation

5.2.2.1 equals()

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

Implemented in Num, Add, Mult, and Var.

5.2.2.2 has_variable()

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

Implemented in Num, Add, Mult, and Var.

5.2.2.3 interp()

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

Implemented in Num, Add, Mult, and Var.

5.2.2.4 pretty_print()

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

Implemented in Num, Add, Mult, and Var.

5.2.2.5 print()

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

Implemented in Num, Add, Mult, and Var.

5.2.2.6 subst()

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

Implemented in Num, Add, Mult, and Var.

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

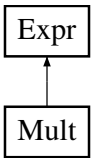
- /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.hpp

5.3 Mult Class Reference

Derived Class from Expr for representing a expression involving addition.

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

Inheritance diagram for Mult:



Public Member Functions

- **Mult** (Expr *lhs, Expr *rhs)

  Constructor.
- bool equals (Expr *e)

  Checks for the equality between the left hand side and the right hand side.
- int interp ()

  This function interprets the value of the expression.
- bool has_variable ()

  This function determines if the the expression consists of a variable or not.
- Expr * subst (std::string s, Expr *e)

  This function substitutes the expression with the combination of sub-expressions if possible.
- void print (std::ostream &out)

  Prints the expression.
- void pretty_print (std::ostream &out)

  Prints the expression with more clarity.

Public Member Functions inherited from Expr

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

Public Attributes

- Expr * **lhs**

  Left hand side Expression.
- Expr * **rhs**

  Right hand side Expression.

5.3.1 Detailed Description

Derived Class from Expr for representing a expression involving addition.

5.3.2 Member Function Documentation

5.3.2.1 equals()

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

Checks for the equality between the left hand side and the right hand side.

Parameters

e	Expression

Returns

boolean value of LHS = RHS

Implements Expr.

5.3.2.2 has_variable()

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

This function determines if the the expression consists of a variable or not.

Returns

boolean value for the expression has a variable or not

Implements Expr.

5.3.2.3 interp()

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

This function interprets the value of the expression.

Returns

integer value of the computed expression

Implements Expr.

5.3.2.4 pretty_print()

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

Prints the expression with more clarity.

Parameters

out	*out*	output stream

Implements Expr.

5.3.2.5 print()

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

Prints the expression.

Parameters

out	*out*	output stream

Implements Expr.

5.3.2.6 subst()

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

This function substitutes the expression with the combination of sub-expressions if possible.

Parameters

s	first argument, string which can be replaced as a part of the expression
e	second argument, Expressiont

Returns

Expression which is modified if combination of sub expressions was possible

Implements Expr.

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

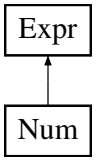
- /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.hpp
- /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.cpp

5.4 Num Class Reference

Derived Class from Expr for representing a number.

`#include <expr.hpp>`

Inheritance diagram for Num:



Public Member Functions

- **Num** (int val)

  Constructor.
- bool equals (Expr ∗e)

  Checks for the equality between the left hand side and the right hand side.
- int interp ()

  This function interprets the value of the integer.
- bool has_variable ()

  This function determines if the the expression consists of a variable or not.
- Expr ∗ subst (std::string s, Expr ∗e)

  This function substitutes the expression with the combination of sub-expressions if possible.
- void print (std::ostream &out)

  Prints the expression.
- void pretty_print (std::ostream &out)

  Prints the expression with more clarity.

Public Member Functions inherited from Expr

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

Public Attributes

- int **val**

  Value of the number.

5.4.1 Detailed Description

Derived Class from Expr for representing a number.

5.4.2 Member Function Documentation

5.4.2.1 equals()

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

Checks for the equality between the left hand side and the right hand side.

Parameters

e	Expression

Returns

boolean value of LHS = RHS

Implements Expr.

5.4.2.2 has_variable()

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

This function determines if the the expression consists of a variable or not.

Returns

boolean value for the number has a variable or not

Implements Expr.

5.4.2.3 interp()

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

This function interprets the value of the integer.

Returns

integer value of the number

Implements Expr.

5.4.2.4 pretty_print()

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

Prints the expression with more clarity.

Parameters

out	*out*	output stream

Implements Expr.

5.4.2.5 print()

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

Prints the expression.

Parameters

out	*out*	output stream

Implements Expr.

5.4.2.6 subst()

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

This function substitutes the expression with the combination of sub-expressions if possible.

Parameters

s	first argument, string which can be replaced as a part of the expression
e	second argument, Expression

Returns

Expression which is modified if combination of sub expressions was possible

Implements Expr.

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

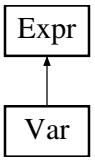
- /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.hpp
- /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.cpp

5.5 Var Class Reference

Derived Class from Expr for representing a expression involving Variable.

`#include <expr.hpp>`

Inheritance diagram for Var:



Public Member Functions

- **Var** (std::string name)

  Constructor.
- bool equals (Expr ∗e)

  Checks for the equality between the left hand side and the right hand side.
- int interp ()

  This function interprets the value of the Variable.
- bool has_variable ()

  This function determines if the the expression consists of a variable or not.
- Expr ∗ subst (std::string s, Expr ∗e)

  This function substitutes the expression with the combination of sub-expressions if possible.
- void print (std::ostream &out)

  Prints the expression.
- void pretty_print (std::ostream &out)

  Prints the expression with more clarity.

Public Member Functions inherited from Expr

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

Public Attributes

- std::string **name**

  name of the variable

5.5.1 Detailed Description

Derived Class from Expr for representing a expression involving Variable.

5.5.2 Member Function Documentation

5.5.2.1 equals()

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

Checks for the equality between the left hand side and the right hand side.

Parameters

e	Expression

Returns

boolean value of LHS = RHS

Implements Expr.

5.5.2.2 has_variable()

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

This function determines if the the expression consists of a variable or not.

Returns

boolean value for the variable is a variable or not

Implements Expr.

5.5.2.3 interp()

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

This function interprets the value of the Variable.

Returns

error message as value of the variable cannot be iterpreted

Implements Expr.

5.5.2.4 pretty_print()

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

Prints the expression with more clarity.

Parameters

out	*out*	output stream

Implements Expr.

5.5.2.5 print()

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

Prints the expression.

Parameters

out	*out*	output stream

Implements Expr.

5.5.2.6 subst()

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

This function substitutes the expression with the combination of sub-expressions if possible.

Parameters

s	first argument, string which can be replaced as a part of the expression
e	second argument, Expressiont

Returns

Expression which is modified if combination of sub expressions was possible

Implements Expr.

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

- /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.hpp
- /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/expr.cpp

Chapter 6

File Documentation

6.1 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/↩ MSDScript/MSDScript/cmdline.cpp File Reference

Contains the code for identifiying the argument given ahead of running the executable on the command line is valid and if it is valid executes the command.

```
#include "cmdline.hpp"
```

Functions

- void use_arguments (int argc, char ∗argv[ ])
  Identifies the argument given ahead of running the executable.

6.1.1 Detailed Description

Contains the code for identifiying the argument given ahead of running the executable on the command line is valid and if it is valid executes the command.

Author

Lakshay Santosh Kucheriya

6.1.2 Function Documentation

6.1.2.1 use_arguments()

```
void use_arguments (
            int argc,
            char * argv[] )
```

Identifies the argument given ahead of running the executable.

Parameters

argc	first parameter, stores the number of command line arguments passed by the user
argv	second parameter, is array of character pointers listing all the arguments.

6.2 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/cmdline.hpp File Reference

Header file for cmdline.cpp file.

```
#include "catch.h"
#include <stdio.h>
#include <iostream>
#include <string>
```

Functions

- void use_arguments (int argc, char *argv[])

  Identifies the argument given ahead of running the executable.

6.2.1 Detailed Description

Header file for cmdline.cpp file.

Author

Lakshay Santosh Kucheriya

6.2.2 Function Documentation

6.2.2.1 use_arguments()

```
void use_arguments (
            int argc,
            char * argv[] )
```

Identifies the argument given ahead of running the executable.

Parameters

argc	first parameter, stores the number of command line arguments passed by the user
argv	second parameter, is array of character pointers listing all the arguments.

6.3 cmdline.hpp

Go to the documentation of this file.

```
00001 //
00002 //  cmdline.hpp
00003 //  MSDScript
00004 //
00005 //  Created by Lakshay Santosh Kucheriya on 1/16/23.
00006 //
00007
00015 #ifndef cmdline_hpp
00016 #define cmdline_hpp
00017 //
00018 #include "catch.h"
00019 #include <stdio.h>
00020 #include <iostream>
00021 #include <string>
00022
00023 void use_arguments(int argc, char* argv[]) ;
00024
00025 #endif /* cmdline_hpp */
```

6.4 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/↩ MSDScript/MSDScript/expr.cpp File Reference

Contains expression class definition.

```
#include "expr.hpp"
```

Functions

- operator_precedence **pretty_print_at** (Expr *e)

6.4.1 Detailed Description

Contains expression class definition.

Author

Lakshay Santosh Kucheriya

6.5 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/↩ MSDScript/MSDScript/expr.hpp File Reference

Header file for expr.cpp file.

```
#include <stdio.h>
#include <string>
#include <iostream>
#include <stdexcept>
#include <sstream>
```

Classes

- class Expr

  Base Class for representing a expression.
- class Num

  Derived Class from Expr for representing a number.
- class Add

  Derived Class from Expr for representing a expression involving addition.
- class Mult

  Derived Class from Expr for representing a expression involving addition.
- class Var

  Derived Class from Expr for representing a expression involving Variable.

Enumerations

- enum operator_precedence { **precedence_none** = 0 , **precedence_add** = 1 , **precedence_mult** = 2 }

  Enumeration for assigning the precedence of the operators.

Functions

- operator_precedence **pretty_print_at** (Expr *e)

6.5.1 Detailed Description

Header file for expr.cpp file.

Author

Lakshay Santosh Kucheriya

6.6 expr.hpp

Go to the documentation of this file.

```
00001 //
00002 //  expr.hpp
00003 //  MSDScript
00004 //
00005 //  Created by Lakshay Santosh Kucheriya on 1/23/23.
00006 //
00007
00015 #ifndef expr_hpp
00016 #define expr_hpp
00017
00018 #include <stdio.h>
00019 #include <string>
00020 #include <iostream>
00021 #include <stdexcept>
00022 #include <sstream>
00023
00024
00027 class Expr  // Base class
00028 {
00029 public:
00030     virtual bool equals(Expr *e) = 0 ;  // This function checks for the equality for the LHS and the
      RHS.
00031     virtual int interp() = 0 ; // This function interprets the value of the expression/variable.
00032     virtual bool has_variable() = 0 ; // This function determines if the the expression consists of a
      variable or not.
```

```
00033     virtual Expr* subst(std::string s, Expr* e) = 0; // This function substitutes the expression with
      the combination of sub-expressions if possible.
00034     virtual void print(std::ostream &out) = 0; // This function prints the expression.
00035     virtual void pretty_print(std::ostream &out) = 0; // This is an extenstion of the print function
      with minor changes.
00036
00037     std::string to_string() {
00038         std::stringstream st("");
00039         this->print(st);
00040         return st.str();
00041     }
00042
00043
00044     std::string to_pretty_string() {
00045         std::stringstream st("");
00046         this->pretty_print(st);
00047         return st.str();
00048     }
00049
00050 };
00051
00052
00055 class Num : public Expr
00056 {
00057 public:
00058     int val;
00059
00060     Num(int val);
00061     bool equals(Expr *e) ;
00062     int interp();
00063     bool has_variable();
00064     Expr* subst(std::string s, Expr* e) ;
00065     void print(std::ostream &out);
00066     void pretty_print(std::ostream &out);
00067 };
00068
00069
00072 class Add : public Expr
00073 {
00074 public:
00075     Expr *lhs;
00076     Expr *rhs;
00077
00078     Add(Expr *lhs, Expr *rhs);
00079     bool equals(Expr *e) ;
00080     int interp();
00081     bool has_variable();
00082     Expr* subst(std::string s, Expr* e) ;
00083     void print(std::ostream &out);
00084     void pretty_print(std::ostream &out);
00085
00086 };
00087
00088
00091 class Mult : public Expr
00092 {
00093 public:
00094     Expr *lhs;
00095     Expr *rhs;
00096
00097     Mult(Expr *lhs, Expr *rhs);
00098     bool equals(Expr *e) ;
00099     int interp();
00100     bool has_variable();
00101     Expr* subst(std::string s, Expr* e) ;
00102     void print(std::ostream &out);
00103     void pretty_print(std::ostream &out);
00104 };
00105
00106
00109 class Var : public Expr
00110 {
00111 public:
00112
00113       std::string name;
00114       Var(std::string name);
00115
00116       bool equals(Expr *e);
00117       int interp();
00118       bool has_variable();
00119       Expr* subst(std::string s, Expr* e) ;
00120       void print(std::ostream &out);
00121       void pretty_print(std::ostream &out);
00122 };
00123
00124
00127 enum operator_precedence {
```

```
00128     precedence_none = 0,
00129     precedence_add = 1,
00130     precedence_mult = 2
00131 };
00132
00133 operator_precedence pretty_print_at(Expr *e);
00134
00135
00136 #endif /* expr_hpp */
```

6.7 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/main.cpp File Reference

Entry point for the project.

```
#include <iostream>
#include "cmdline.hpp"
```

Functions

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

6.7.1 Detailed Description

Entry point for the project.

Author

Lakshay Santosh Kucheriya

6.8 /Users/lakshaysantoshkucheriya/Desktop/MSD/Courses/MSDScript/MSDScript/MSDScript/tests.cpp File Reference

contains all the tests forl the methods present in this project

```
#include "catch.h"
#include "expr.hpp"
#include <stdio.h>
```

Functions

- **TEST_CASE** ("equals")
- **TEST_CASE** ("Tests for checking interp")
- **TEST_CASE** ("Test for has_variable")
- **TEST_CASE** ("Test for subst")
- **TEST_CASE** ("to_string")
- **TEST_CASE** ("to_pretty_string")

6.8.1 Detailed Description

contains all the tests forl the methods present in this project

Author

Lakshay Santosh Kucheriya

Index