MSDscript

Generated by Doxygen 1.9.6

1 MSDScript	1
2 Hierarchical Index	3
2.1 Class Hierarchy	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Class Documentation	9
5.1 Add Class Reference	9
5.1.1 Constructor & Destructor Documentation	10
5.1.1.1 Add()	10
5.1.2 Member Function Documentation	10
5.1.2.1 equals()	10
5.1.2.2 has_variable()	11
5.1.2.3 interp()	11
5.1.2.4 subst()	11
5.2 Expr Class Reference	12
5.2.1 Member Function Documentation	12
5.2.1.1 equals()	12
5.2.1.2 has_variable()	13
5.2.1.3 interp()	13
5.2.1.4 subst()	13
5.3 Mult Class Reference	13
5.3.1 Constructor & Destructor Documentation	14
5.3.1.1 Mult()	14
5.3.2 Member Function Documentation	14
5.3.2.1 equals()	14
5.3.2.2 has_variable()	15
5.3.2.3 interp()	15
5.3.2.4 subst()	15
5.4 Num Class Reference	16
5.4.1 Constructor & Destructor Documentation	17
5.4.1.1 Num()	17
5.4.2 Member Function Documentation	18
5.4.2.1 equals()	18
5.4.2.2 has_variable()	18
5.4.2.3 interp()	19
5.4.2.4 subst()	19
5.5 Var Class Reference	20
5.5.1 Constructor & Destructor Documentation	20

5.5.1.1 Var()	20
5.5.2 Member Function Documentation	21
5.5.2.1 equals()	21
5.5.2.2 has_variable()	21
5.5.2.3 interp()	22
5.5.2.4 subst()	22
6 File Documentation	23
6.1 /Users/reshmaraghavan/Desktop/msdscript/cmdline.cpp File Reference	23
6.1.1 Detailed Description	23
6.2 /Users/reshmaraghavan/Desktop/msdscript/cmdline.hpp File Reference	24
6.2.1 Detailed Description	24
6.3 /Users/reshmaraghavan/Desktop/msdscript/cmdline.hpp	24
6.4 /Users/reshmaraghavan/Desktop/msdscript/Expr.cpp File Reference	24
6.4.1 Detailed Description	25
6.5 /Users/reshmaraghavan/Desktop/msdscript/Expr.h File Reference	25
6.5.1 Detailed Description	25
6.6 /Users/reshmaraghavan/Desktop/msdscript/Expr.h	26
6.7 /Users/reshmaraghavan/Desktop/msdscript/TestExpr.cpp File Reference	26
6.7.1 Detailed Description	27
6.7.2 Function Documentation	27
6.7.2.1 TEST_CASE() [1/4]	27
6.7.2.2 TEST_CASE() [2/4]	27
6.7.2.3 TEST_CASE() [3/4]	27
6.7.2.4 TEST_CASE() [4/4]	27
Index	29

MSDScript

Author

Reshma Raghavan

Date

01-11-2023

2 MSDScript

Hierarchical Index

2.1 Class Hierarchy

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

Expr				 												 									12
Add			 	 																					9
Mult			 	 																					13
Num			 	 																		 			16
Var .		_	 	 				 								 									20

4 Hierarchical Index

Class Index

3.1 Class List

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

Add			 												 										 		9
Expr			 												 										 		12
Mult			 												 										 		13
Num			 												 										 		16
Var																											20

6 Class Index

File Index

4.1 File List

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

/Users/reshmaraghavan/Desktop/msdscript/cmdline.cpp	
Single method to run executable	23
/Users/reshmaraghavan/Desktop/msdscript/cmdline.hpp	
Declaration of a single method; more information available in complementary .cpp file	24
/Users/reshmaraghavan/Desktop/msdscript/Expr.cpp	
Method implementations of subclasses	24
/Users/reshmaraghavan/Desktop/msdscript/Expr.h	
Declaration of the abstract class's methods; more information available in complementary .cpp	
file	25
/Users/reshmaraghavan/Desktop/msdscript/TestExpr.cpp	
Test cases of Expression subclass methods	26

8 File Index

Class Documentation

5.1 Add Class Reference

Inheritance diagram for Add:



Public Member Functions

Add (Expr *left, Expr *right)

Constructor.

bool equals (Expr *e)

Overriding "equals" method in Expr (base) class Basically, we compare two Add objects and determine if they are equal to one another by comparing their member variables (both Expr* lhs and Expr* rhs)

• int interp ()

Add returns a number (an int) Basically, interp() interprets the value of the object. In this case, an Add object contains two expressions that should be added to one another; the sum is returned.

• bool has_variable ()

Add returns true if it HAS a variable The general rule of thumb is this function returns true if the Expression is a variable or has a variable In Add, we check if the lhs or the rhs HAVE a variable.

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

If the first param (str) is in either lhs or rhs, we replace it with e and return.

• virtual bool equals (Expr *e)=0

Expression class: it is an abstract class with virtual methods to be implemented in its derived classes.

- virtual int interp ()=0
- virtual bool has variable ()=0
- virtual Expr * subst (std::string str, Expr *e)=0

Public Attributes

- **Expr** * **Ihs**
- **Expr** * **rhs**

5.1.1 Constructor & Destructor Documentation

5.1.1.1 Add()

Constructor.

Parameters

left	a pointer to an Expression
right	a pointer to an Expression

Returns

nothing (it is a constructor so it just creates an object of type Add)

5.1.2 Member Function Documentation

5.1.2.1 equals()

Overriding "equals" method in Expr (base) class Basically, we compare two Add objects and determine if they are equal to one another by comparing their member variables (both Expr* lhs and Expr* rhs)

Parameters

е	Pointer to an expression
---	--------------------------

Returns

true or false (it is a boolean)

Implements Expr.

5.1 Add Class Reference 11

5.1.2.2 has_variable()

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

Add returns true if it HAS a variable The general rule of thumb is this function returns true if the Expression is a variable or has a variable In Add, we check if the lhs or the rhs HAVE a variable.

Parameters

none

Returns

true or false (it is a boolean)

Implements Expr.

5.1.2.3 interp()

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

Add returns a number (an int) Basically, interp() interprets the value of the object. In this case, an Add object contains two expressions that should be added to one another; the sum is returned.

Parameters

none

Returns

the sum of two expressions

Implements Expr.

5.1.2.4 subst()

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

If the first param (str) is in either lhs or rhs, we replace it with e and return.

Parameters

str	a String value
e	a pointer to an Expression

Returns

a new Add Expression

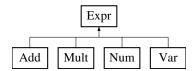
Implements Expr.

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

- /Users/reshmaraghavan/Desktop/msdscript/Expr.h
- /Users/reshmaraghavan/Desktop/msdscript/Expr.cpp

5.2 Expr Class Reference

Inheritance diagram for Expr:



Public Member Functions

• virtual bool equals (Expr *e)=0

Expression class: it is an abstract class with virtual methods to be implemented in its derived classes.

- virtual int interp ()=0
- virtual bool has_variable ()=0
- virtual Expr * subst (std::string str, Expr *e)=0

5.2.1 Member Function Documentation

5.2.1.1 equals()

Expression class: it is an abstract class with virtual methods to be implemented in its derived classes.

Implemented in Num, Add, Mult, and Var.

5.3 Mult Class Reference 13

5.2.1.2 has_variable()

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

Implemented in Num, Add, Mult, and Var.
```

5.2.1.3 interp()

```
virtual int Expr::interp () [pure virtual] Implemented in Num, Add, Mult, and Var.
```

5.2.1.4 subst()

Implemented in Num, Add, Mult, and Var.

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

/Users/reshmaraghavan/Desktop/msdscript/Expr.h

5.3 Mult Class Reference

Inheritance diagram for Mult:



Public Member Functions

- Mult (Expr *left, Expr *right)
 Constructor.
- bool equals (Expr *e)

Overriding "equals" method in Expr (base) class Basically, we compare two Mult objects and determine if they are equal to one another by comparing their member variables (both Expr* Ihs and Expr* rhs)

• int interp ()

Mult returns a number (an int) Basically, interp() interprets the value of the object. In this case, a Mult object contains two expressions that should be multiplied by one another; the product is returned.

• bool has variable ()

Mult returns true if it HAS a variable The general rule of thumb is this function returns true if the Expression is a variable or has a variable In Mult, we check if the lhs or the rhs HAVE a variable.

• Expr * subst (std::string str, Expr *e)

If the first param (str) is in either lhs or rhs, we replace it with e and return.

virtual bool equals (Expr *e)=0

Expression class: it is an abstract class with virtual methods to be implemented in its derived classes.

- virtual int interp ()=0
- virtual bool has_variable ()=0
- virtual Expr * subst (std::string str, Expr *e)=0

Public Attributes

- **Expr** * **Ihs**
- **Expr** * **rhs**

5.3.1 Constructor & Destructor Documentation

5.3.1.1 Mult()

Constructor.

Parameters

left	a pointer to an Expression
right	a pointer to an Expression

Returns

nothing (it is a constructor so it just creates an object of type Add)

5.3.2 Member Function Documentation

5.3.2.1 equals()

Overriding "equals" method in Expr (base) class Basically, we compare two Mult objects and determine if they are equal to one another by comparing their member variables (both Expr* lhs and Expr* rhs)

Parameters

e Pointer to an expression

Returns

true or false (it is a boolean)

Implements Expr.

5.3 Mult Class Reference 15

5.3.2.2 has_variable()

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

Mult returns true if it HAS a variable The general rule of thumb is this function returns true if the Expression is a variable or has a variable In Mult, we check if the Ihs or the rhs HAVE a variable.

Parameters

none

Returns

true or false (it is a boolean)

Implements Expr.

5.3.2.3 interp()

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

Mult returns a number (an int) Basically, interp() interprets the value of the object. In this case, a Mult object contains two expressions that should be multiplied by one another; the product is returned.

Parameters

none

Returns

the product of two subexpression classes

Implements Expr.

5.3.2.4 subst()

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

If the first param (str) is in either lhs or rhs, we replace it with e and return.

Parameters

str	a String value
e	a pointer to an Expression

Returns

a new Mult Expression

Implements Expr.

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

- /Users/reshmaraghavan/Desktop/msdscript/Expr.h
- /Users/reshmaraghavan/Desktop/msdscript/Expr.cpp

5.4 Num Class Reference

Inheritance diagram for Num:



Public Member Functions

• Num (int value)

Constructor.

bool equals (Expr *e)

Overriding "equals" method in Expr (base) class Basically, we compare two Num objects and determine if they are equal to one another by comparing their member variables (int val)

• int interp ()

Num returns a number (an int) Basically, interp() interprets the value of the object. In this case, a Num object contains an integer; this is returned.

• bool has_variable ()

Num returns false since it IS NOT a variable and DOES NOT HAVE a variable The general rule of thumb is this function returns true if the Expression is a variable or has a variable.

• Expr * subst (std::string str, Expr *e)

Num will never contain a string, so it just returns the original value.

• virtual bool equals (Expr *e)=0

Expression class: it is an abstract class with virtual methods to be implemented in its derived classes.

- virtual int interp ()=0
- virtual bool has_variable ()=0
- virtual Expr * subst (std::string str, Expr *e)=0

Public Attributes

int val

5.4 Num Class Reference 17

5.4.1 Constructor & Destructor Documentation

5.4.1.1 Num()

```
Num::Num (
     int value )
```

Constructor.

Parameters

Returns

nothing (it is a constructor so it just creates an object of type Num)

5.4.2 Member Function Documentation

5.4.2.1 equals()

Overriding "equals" method in Expr (base) class Basically, we compare two Num objects and determine if they are equal to one another by comparing their member variables (int val)

Parameters

e Pointer to an expression

Returns

true or false (it is a boolean)

Implements Expr.

5.4.2.2 has_variable()

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

Num returns false since it IS NOT a variable and DOES NOT HAVE a variable The general rule of thumb is this function returns true if the Expression is a variable or has a variable.

Parameters

none

Returns

true or false (it is a boolean)

Implements Expr.

5.4 Num Class Reference

5.4.2.3 interp()

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

Num returns a number (an int) Basically, interp() interprets the value of the object. In this case, a Num object contains an integer; this is returned.

Parameters

none

Returns

int

Implements Expr.

5.4.2.4 subst()

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

Num will never contain a string, so it just returns the original value.

Parameters

str	a String value
е	a pointer to an Expression

Returns

a new Num Expression (containing the same integer val)

Implements Expr.

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

- /Users/reshmaraghavan/Desktop/msdscript/Expr.h
- /Users/reshmaraghavan/Desktop/msdscript/Expr.cpp

5.5 Var Class Reference

Inheritance diagram for Var:



Public Member Functions

• Var (std::string value)

Constructor.

bool equals (Expr *e)

Overriding "equals" method in Expr (base) class Basically, we compare two Var objects and determine if they are equal to one another by comparing their member variables (String val)

• int interp ()

A Var has no int equivalent.

• bool has variable ()

Var returns true as it IS a variable The general rule of thumb is this function returns true if the Expression is a variable or has a variable.

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

If the first param (str) is in the original Var, we replace it with e and return.

• virtual bool equals (Expr *e)=0

Expression class: it is an abstract class with virtual methods to be implemented in its derived classes.

- virtual int interp ()=0
- virtual bool has_variable ()=0
- virtual Expr * subst (std::string str, Expr *e)=0

Public Attributes

· std::string val

5.5.1 Constructor & Destructor Documentation

5.5.1.1 Var()

Constructor.

5.5 Var Class Reference 21

Parameters

Returns

nothing (it is a constructor so it just creates an object of type Var)

5.5.2 Member Function Documentation

5.5.2.1 equals()

Overriding "equals" method in Expr (base) class Basically, we compare two Var objects and determine if they are equal to one another by comparing their member variables (String val)

Parameters

e Pointer to an expression

Returns

true or false (it is a boolean)

Implements Expr.

5.5.2.2 has_variable()

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

Var returns true as it IS a variable The general rule of thumb is this function returns true if the Expression is a variable or has a variable.

Parameters

none

Returns

true or false (it is a boolean)

Implements Expr.

5.5.2.3 interp()

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

A Var has no int equivalent.

Parameters

none

Returns

throws a runtime error

Implements Expr.

5.5.2.4 subst()

If the first param (str) is in the original Var, we replace it with e and return.

Parameters

str	a String value
e	a pointer to an Expression

Returns

a new Var Expression

Implements Expr.

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

- /Users/reshmaraghavan/Desktop/msdscript/Expr.h
- /Users/reshmaraghavan/Desktop/msdscript/Expr.cpp

File Documentation

6.1 /Users/reshmaraghavan/Desktop/msdscript/cmdline.cpp File Reference

contains single method to run executable

```
#include "cmdline.hpp"
#include <string>
#include <iostream>
#include <regex>
#include "catch.h"
```

Functions

• void use_arguments (int argc, char **argv)

6.1.1 Detailed Description

contains single method to run executable

Runs executable with 0 or more arguments passed in

Parameters

argc	first argument, number of arguments
argv	second argument, pointer to array of argument values

Returns

nothing (it is a void function)

24 File Documentation

Author

Reshma Raghavan

Date

01-11-2023

6.2 /Users/reshmaraghavan/Desktop/msdscript/cmdline.hpp File Reference

contains declaration of a single method; more information available in complementary .cpp file

Functions

void use_arguments (int argc, char **argv)

6.2.1 Detailed Description

contains declaration of a single method; more information available in complementary .cpp file

Author

Reshma Raghavan

Date

01-11-2023

6.3 /Users/reshmaraghavan/Desktop/msdscript/cmdline.hpp

Go to the documentation of this file.

```
00001

00009 #pragma once

00010

00011 void use_arguments(int argc, char **argv);
```

6.4 /Users/reshmaraghavan/Desktop/msdscript/Expr.cpp File Reference

contains method implementations of subclasses

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

6.4.1 Detailed Description

contains method implementations of subclasses

Author

Reshma Raghavan

Date

01-17-2023

6.5 /Users/reshmaraghavan/Desktop/msdscript/Expr.h File Reference

contains declaration of the abstract class's methods; more information available in complementary .cpp file

```
#include <string>
```

Classes

- class Expr
- · class Num
- class Add
- class Mult
- class Var

6.5.1 Detailed Description

contains declaration of the abstract class's methods; more information available in complementary .cpp file

Author

Reshma Raghavan

Date

01-17-2023

26 File Documentation

6.6 /Users/reshmaraghavan/Desktop/msdscript/Expr.h

Go to the documentation of this file.

```
00009 #ifndef EXPRESSION_CLASSES_EXPR_H
00010 #define EXPRESSION_CLASSES_EXPR_H
00012
00013 #include <string>
00014
00015 class Expr {
00016
00021 public:
         virtual bool equals(Expr* e) = 0;
00023
         virtual int interp() = 0;
00024
         virtual bool has_variable() = 0;
         virtual Expr* subst (std::string str, Expr* e) = 0;
00025
00026 };
00027
00028 class Num : public Expr {
00029 public:
00030
         int val;
00031
         Num(int value);
00032
         bool equals (Expr *e);
         int interp();
00033
00034
         bool has_variable();
00035
         Expr* subst (std::string str, Expr* e);
00036 };
00037
00038 class Add : public Expr {
00039 public:
       Expr* lhs;
00041
         Expr* rhs;
00042
         Add(Expr* left, Expr* right);
00043
         bool equals(Expr *e);
00044
         int interp();
00045
         bool has_variable();
00046
         Expr* subst (std::string str, Expr* e);
00047 };
00048
00049 class Mult: public Expr {
00050 public:
         Expr* lhs;
00051
         Expr* rhs;
         Mult(Expr* left, Expr* right);
00054
         bool equals(Expr *e);
00055
         int interp();
         bool has_variable();
00056
00057
         Expr* subst (std::string str, Expr* e);
00058 };
00060 class Var: public Expr {
00061 public:
00062
         std::string val;
         Var(std::string value);
00063
         bool equals(Expr *e);
00064
00065
         int interp();
00066
         bool has_variable();
00067
         Expr* subst (std::string str, Expr* e);
00068 };
00069
00070
00072 #endif //EXPRESSION_CLASSES_EXPR_H
```

6.7 /Users/reshmaraghavan/Desktop/msdscript/TestExpr.cpp File Reference

contains test cases of Expression subclass methods

```
#include "catch.h"
#include "Expr.h"
```

Functions

```
TEST_CASE ("equals")TEST_CASE ("interp")TEST_CASE ("has_variable")
```

6.7.1 Detailed Description

• TEST CASE ("subst")

contains test cases of Expression subclass methods

Author

Reshma Raghavan

Date

01-23-2023

6.7.2 Function Documentation

6.7.2.1 TEST_CASE() [1/4]

Testing the "equals" method. Asserting true or false as the case may be.

6.7.2.2 TEST_CASE() [2/4]

Testing the "has_variable" method. Asserting true or false as the case may be.

6.7.2.3 TEST_CASE() [3/4]

```
TEST_CASE (
    "interp" )
```

Testing the "interp" method. Asserting true or false as the case may be.

6.7.2.4 TEST_CASE() [4/4]

```
TEST_CASE (
     "subst" )
```

Testing the "subst" method. Asserting true or false as the case may be.

28 File Documentation

Index

```
/Users/reshmaraghavan/Desktop/msdscript/Expr.cpp,
                                                            equals, 18
                                                            has_variable, 18
/Users/reshmaraghavan/Desktop/msdscript/Expr.h, 25
                                                            interp, 19
/Users/reshmaraghavan/Desktop/msdscript/TestExpr.cpp,
                                                            Num, 17
                                                            subst, 19
/Users/reshmaraghavan/Desktop/msdscript/cmdline.cpp,
                                                        subst
                                                            Add, 11
/Users/reshmaraghavan/Desktop/msdscript/cmdline.hpp,
                                                            Expr, 13
                                                            Mult, 15
Add, 9
                                                            Num, 19
     Add, 10
                                                            Var, 22
     equals, 10
                                                        TEST_CASE
    has_variable, 10
                                                            TestExpr.cpp, 27
    interp, 11
    subst, 11
                                                        TestExpr.cpp
                                                            TEST_CASE, 27
equals
                                                        Var, 20
    Add, 10
                                                            equals, 21
     Expr, 12
                                                            has_variable, 21
     Mult, 14
                                                            interp, 22
    Num, 18
                                                            subst, 22
     Var, 21
                                                            Var, 20
Expr, 12
     equals, 12
    has_variable, 12
    interp, 13
    subst, 13
has_variable
     Add, 10
     Expr, 12
     Mult, 14
    Num, 18
     Var, 21
interp
     Add, 11
     Expr, 13
    Mult, 15
    Num, 19
    Var, 22
Mult, 13
     equals, 14
    has_variable, 14
    interp, 15
    Mult, 14
    subst, 15
```

Num, 16