LAB 08 EXPRESSION TREES

Generated by Doxygen 1.8.11

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 ExprTree< DataType > Class Template Reference

`#include <ExpressionTree.h>`

Public Member Functions

- **ExprTree** ()
  Initializes inside variables.
- **ExprTree** (const **ExprTree** &source)
  Copy Constructor.
- **ExprTree** & **operator=** (const **ExprTree** &source)
- ~**ExprTree** ()
  this is the Destructor
- void **build** ()
  This builds the tree of data values.
- void **expression** () const
  prints out the expression that the tree represents
- DataType **evaluate** () const throw (logic_error)
  this evaluates the value to which the expression represents
- void **clear** ()
  this clears the tree
- void **commute** ()
  commutes the tree (will not necessarily preserve evaluation values)
- bool **isEquivalent** (const **ExprTree** &source) const
  Evaluates whether two trees are equivalent or not.
- bool **isEmpty** () const
- void **showStructure** () const
  this shows the structure of the tree

3.1.1 Constructor & Destructor Documentation

3.1.1.1 template<typename DataType > ExprTree< DataType >::ExprTree ( )

Initializes inside variables.

3.1.1.2 template<typename DataType > ExprTree< DataType >::ExprTree (const ExprTree< DataType > & *source*)

Copy Constructor.

3.1.1.3 template<typename DataType > ExprTree< DataType >::~ExprTree ( )

this is the Destructor

3.1.2 Member Function Documentation

3.1.2.1 template<typename DataType > void ExprTree< DataType >::build ( )

This builds the tree of data values.

3.1.2.2 template<typename DataType > void ExprTree< DataType >::clear ( )

this clears the tree

3.1.2.3 template<typename DataType > void ExprTree< DataType >::commute ( )

commutes the tree (will not necessarily preserve evaluation values)

3.1.2.4 template<typename DataType > DataType ExprTree< DataType >::evaluate ( ) const throw logic_error)

this evaluates the value to which the expression represents

3.1.2.5 template<typename DataType > void ExprTree< DataType >::expression ( ) const

prints out the expression that the tree represents

3.1.2.6 template<typename DataType > bool ExprTree< DataType >::isEmpty ( ) const

3.1.2.7 template<typename DataType > bool ExprTree< DataType >::isEquivalent (const ExprTree< DataType > & *source*) const

Evaluates whether two trees are equivalent or not.

Parameters

const	**ExprTree** (p. 5)& source, which is the tree to be compared against

3.1.2.8 template<typename DataType > ExprTree< DataType > & ExprTree< DataType >::operator= (const ExprTree< DataType > & *other*)

equates the values of the two objects (deep copy)

Parameters

const	**ExprTree** (p. 5)& other

3.1.2.9 template<typename DataType > void ExprTree< DataType >::showStructure ( ) const

this shows the structure of the tree

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

- **ExpressionTree.h**
- **ExpressionTree.cpp**
- **show8.cpp**

Chapter 4

File Documentation

4.1 config.h File Reference

Macros

- #define **LAB8_TEST1** 0
- #define **LAB8_TEST2** 1
- #define **LAB8_TEST3** 1

4.1.1 Macro Definition Documentation

4.1.1.1 #define LAB8_TEST1 0

Expression Tree class (Lab 8) configuration file. Activate test #N by defining the corresponding LAB8_TESTN to have the value 1.

4.1.1.2 #define LAB8_TEST2 1

4.1.1.3 #define LAB8_TEST3 1

4.2 ExpressionTree.cpp File Reference

```
#include "ExpressionTree.h"
#include <string>
```

4.3 ExpressionTree.h File Reference

```
#include <stdexcept>
#include <iostream>
```

Classes

- class **ExprTree**< **DataType** >

4.4 show8.cpp File Reference

4.5 test8.cpp File Reference

```
#include <iostream>
#include <stdexcept>
#include "ExpressionTree.cpp"
#include "config.h"
```

Functions

- template<typename DataType >
  void **dummy** (**ExprTree**< DataType > copyTree)
- int **main** ()

4.5.1 Function Documentation

4.5.1.1 template<typename DataType > void dummy (ExprTree< DataType > *copyTree*)

4.5.1.2 int main ( )

Index