FuzzyLogicGenerator

Generated by Doxygen 1.8.3.1

Tue Oct 6 2015 19:27:00

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Class Documentation

4.1 Engine Class Reference

```
#include <engine.h>
```

Public Member Functions

- Engine (std::string inputFileName, unsigned int baseTicks) throw (std::string)

  Constructor creating an engine class.
- ∼Engine ()

  deconstructor
- void createKernel () throw (std::string)

  Methode to write the kernel source code in MaxJ to a file.
- void createManager () throw (std::string)

  Methode to write the manager source code in MaxJ to a file.
- void createEngineParameters () throw (std::string)

  Methode to write the engine parameter source code in MaxJ to a file.
- void createCPUCode () throw (std::string)

  Methode to write the CPU source code in c to a file.
- void createTestData () throw (std::string)

  Methode to generate a .fld file containing auto generated test data.
- void createProject ()

  Methode to copy all generated files into an directory structure usable by MaxIDE. Also generating Makefiles.
- void debugPrint ()

  Methode to create some debug information regarding the correct parsing of the fuzzy logic problem description.

4.1.1 Constructor & Destructor Documentation

4.1.1.1 Engine::Engine (std::string *inputFileName,* unsigned int *baseTicks*) throw (std::string)

Constructor creating an engine class.

Parameters

inputFileName	path to the file containing the description of the fuzzy logic problem.
baseTicks	Number of different input values for each inputVariable in the generated test data.

4.1.1.2 Engine::∼Engine ( )

deconstructor

4.1.2 Member Function Documentation

4.1.2.1 void Engine::createCPUCode ( ) throw (std::string)

Methode to write the CPU source code in c to a file.

4.1.2.2 void Engine::createEngineParameters ( ) throw (std::string)

Methode to write the engine parameter source code in MaxJ to a file.

4.1.2.3 void Engine::createKernel ( ) throw (std::string)

Methode to write the kernel source code in MaxJ to a file.

4.1.2.4 void Engine::createManager ( ) throw (std::string)

Methode to write the manager source code in MaxJ to a file.

4.1.2.5 void Engine::createProject ( )

Methode to copy all generated files into an directory structure usable by MaxIDE. Also generating Makefiles.

4.1.2.6 void Engine::createTestData ( ) throw (std::string)

Methode to generate a .fld file containing auto generated test data.

4.1.2.7 void Engine::debugPrint ( )

Methode to create some debug information regarding the correct parsing of the fuzzy logic problem description.

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

- engine.h
- engine.cpp

4.2 EvaluationRule Class Reference

```
#include <evaluationRule.h>
```

Public Member Functions

- EvaluationRule (std::string originalText)

  Constructor.
- void pushOperandList (Term ∗operand)

  pushes an operand onto the operand list
- void pushOperatorList (Operator op)

pushes an operator onto the operator list

- void createRuleEvaluation (std::ofstream &file)

  generates the MaxJ code to evaluate the rule and writes it to a file

- void debugPrint ()

  function to provide some debug information regarding the description parsing

4.2.1 Constructor & Destructor Documentation

4.2.1.1 EvaluationRule::EvaluationRule (std::string *originalText*)

Constructor.

Parameters

originalText	Original text from the description

4.2.2 Member Function Documentation

4.2.2.1 void EvaluationRule::createRuleEvaluation (std::ofstream & *file*)

generates the MaxJ code to evaluate the rule and writes it to a file

Parameters

file	The file to write to

4.2.2.2 void EvaluationRule::debugPrint ( )

function to provide some debug information regarding the description parsing

4.2.2.3 void EvaluationRule::pushOperandList (Term ∗ *operand*)

pushes an operand onto the operand list

Parameters

operand	The operand to push

4.2.2.4 void EvaluationRule::pushOperatorList (Operator *op*)

pushes an operator onto the operator list

Parameters

op	The operator to push

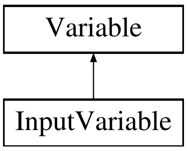
The documentation for this class was generated from the following files:

- evaluationRule.h
- evaluationRule.cpp

4.3 InputVariable Class Reference

```
#include <inputVariable.h>
```

Inheritance diagram for InputVariable:



Public Member Functions

- InputVariable (std::string name)

  Constructor.
- virtual void createFuzzySets (std::ofstream &file)

  Methode creating the fuzzy set definitions needed by the input variable.
- void createInputStream (std::ofstream &file)

  Methode creating the input streams in the Kernel code.
- void createFuzzyfication (std::ofstream &file)

  Methode creating the call to the correct fuzzification function for every term of the input variable.

Additional Inherited Members

4.3.1 Constructor & Destructor Documentation

4.3.1.1 InputVariable::InputVariable (std::string *name*)

Constructor.

Parameters

name	Name of the variable

4.3.2 Member Function Documentation

4.3.2.1 void InputVariable::createFuzzyfication (std::ofstream & *file*)

Methode creating the call to the correct fuzzification function for every term of the input variable.

Parameters

file	File to write to

4.3.2.2 void InputVariable::createFuzzySets (std::ofstream & *file*) `[virtual]`

Methode creating the fuzzy set definitions needed by the input variable.

Parameters

file	File to write the definitions to

Reimplemented from Variable.

4.3.2.3 void InputVariable::createInputStream (std::ofstream & *file*)

Methode creating the input streams in the Kernel code.

Parameters

file	File to write to

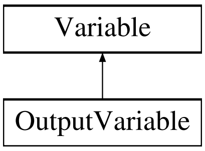
The documentation for this class was generated from the following files:

- inputVariable.h
- inputVariable.cpp

4.4 OutputVariable Class Reference

```
#include <outputVariable.h>
```

Inheritance diagram for OutputVariable:



Public Member Functions

- OutputVariable (std::string name)

  Constructor.
- virtual void createFuzzySets (std::ofstream &file)

  Methode creating the fuzzy set definitions needed by the output variable.
- void createDeclaration (std::ofstream &file)

  Methode used to create some definitions for every term to the MaxJ file that can be used later.
- void createDefuzzification (std::ofstream &file)

  Methode to call the correct defuzzification function for every term.
- void createOutputStream (std::ofstream &file)

  Methode to create the output streams.

Additional Inherited Members

4.4.1 Constructor & Destructor Documentation

4.4.1.1 OutputVariable::OutputVariable (std::string *name*)

Constructor.

Parameters

name	Name of the variable

4.4.2 Member Function Documentation

4.4.2.1 void OutputVariable::createDeclaration (std::ofstream & *file*)

Methode used to create some definitions for every term to the MaxJ file that can be used later.

Parameters

file	File to write to

4.4.2.2 void OutputVariable::createDefuzzification (std::ofstream & *file*)

Methode to call the correct defuzzification function for every term.

Parameters

file	File to write to

4.4.2.3 void OutputVariable::createFuzzySets (std::ofstream & *file*) [virtual]

Methode creating the fuzzy set definitions needed by the output variable.

Parameters

file	File to write the definitions to

Reimplemented from Variable.

4.4.2.4 void OutputVariable::createOutputStream (std::ofstream & *file*)

Methode to create the output streams.

Parameters

file	File to write to

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

- outputVariable.h
- outputVariable.cpp

4.5 Strings Class Reference

this class is used to save different strings which will than be coppied into the created source files

```
#include <strings.h>
```

Static Public Member Functions

- static std::string getImportString ()
- static std::string getDefuzzificationStructString ()
- static std::string getTriangleTypeString ()
- static std::string getTriangleDataStructureCreationMethodeString ()

- static std::string getTriangleFuzzificationMethodeString ()
- static std::string getTriangleDefuzzificationMethodeString ()
- static std::string getManagerImportString ()
- static std::string getManagerString1 ()
- static std::string getManagerString2 ()
- static std::string getManagerString3 ()
- static std::string getEngineParametersImportString ()
- static std::string getEngineParametersString1 ()
- static std::string getEngineParametersString2 ()
- static std::string getEngineParametersString3 ()
- static std::string getCPUIncludes ()

4.5.1 Detailed Description

this class is used to save different strings which will than be coppied into the created source files

4.5.2 Member Function Documentation

4.5.2.1 static std::string Strings::getCPUIncludes ( ) [inline], [static]

4.5.2.2 static std::string Strings::getDefuzzificationStructString ( ) [inline], [static]

4.5.2.3 static std::string Strings::getEngineParametersImportString ( ) [inline], [static]

4.5.2.4 static std::string Strings::getEngineParametersString1 ( ) [inline], [static]

4.5.2.5 static std::string Strings::getEngineParametersString2 ( ) [inline], [static]

4.5.2.6 static std::string Strings::getEngineParametersString3 ( ) [inline], [static]

4.5.2.7 static std::string Strings::getImportString ( ) [inline], [static]

4.5.2.8 static std::string Strings::getManagerImportString ( ) [inline], [static]

4.5.2.9 static std::string Strings::getManagerString1 ( ) [inline], [static]

4.5.2.10 static std::string Strings::getManagerString2 ( ) [inline], [static]

4.5.2.11 static std::string Strings::getManagerString3 ( ) [inline], [static]

4.5.2.12 static std::string Strings::getTriangleDataStructureCreationMethodeString ( ) [inline], [static]

4.5.2.13 static std::string Strings::getTriangleDefuzzificationMethodeString ( ) [inline], [static]

4.5.2.14 static std::string Strings::getTriangleFuzzificationMethodeString ( ) [inline], [static]

4.5.2.15 static std::string Strings::getTriangleTypeString ( ) [inline], [static]

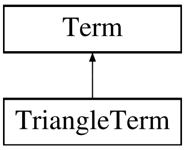
The documentation for this class was generated from the following files:

- strings.h
- strings.cpp

4.6 Term Class Reference

```
#include <term.h>
```

Inheritance diagram for Term:



Public Member Functions

- Term (std::string name, TermType type, bool isIn)

  constructor
- virtual ∼Term ()

  destructor
- virtual TermType getType ()=0

  Function returning the type of the term.
- void setVariableName (std::string variableName)

  Function used to set the name of the associated variable.
- std::string getMaxJName () const

  Function used to get the name of the term in MaxJ.
- std::string getName () const

  Function used to get the name of the term.
- std::string getDegreeOfMembershipName () const

  Function to get the name of the variable containing the degree of membership for the term.
- virtual std::string getDefuzzification () const =0

  Function used to get the name of the defuzzification function.
- virtual void createFuzzySet (std::ofstream &file)=0

  Function used to generate a fuzzy set equaling the term.
- virtual void createFuzzyfication (std::ofstream &file)

  Function used to create a the fuzzification for a given term.
- virtual void debugPrint ()

  Function providing some debug information regarding the correct parsing of the description.

Static Public Member Functions

- static void writeTypeDefsToFile (std::ofstream &file)

  Function used to write all needed Type definitions to the file.
- static void writeHelperFunctionsToFile (std::ofstream &file)

  Function used to write all needed helper function (like (de)fuzzification) to the file.

Protected Attributes

- std::string name

  Name of the term.
- std::string variableName

  Name of the variable the term is belonging to.

4.6.1 Constructor & Destructor Documentation

4.6.1.1 Term::Term (std::string *name*, TermType *type*, bool *isIn*)

constructor

Parameters

name	Name of the term
type	Type of the term
isIn	flag indicating if the term is used with an input or an output variable

4.6.1.2 virtual Term::∼Term ( ) `[inline]`, `[virtual]`

destructor

4.6.2 Member Function Documentation

4.6.2.1 void Term::createFuzzyfication (std::ofstream & *file*) `[virtual]`

Function used to create a the fuzzification for a given term.

Parameters

file	File to write to

Reimplemented in TriangleTerm.

4.6.2.2 virtual void Term::createFuzzySet (std::ofstream & *file*) `[pure virtual]`

Function used to generate a fuzzy set equaling the term.

Parameters

file	File to write to

Implemented in TriangleTerm.

4.6.2.3 void Term::debugPrint ( ) `[virtual]`

Function providing some debug information regarding the correct parsing of the description.

Reimplemented in TriangleTerm.

4.6.2.4 virtual std::string Term::getDefuzzification ( ) const `[pure virtual]`

Function used to get the name of the defuzzification function.

Returns

name of the defuzzification function

Implemented in TriangleTerm.

4.6.2.5 std::string Term::getDegreeOfMembershipName ( ) const

Function to get the name of the variable containing the degree of membership for the term.

Returns

name of the variable containing the degree of membership for the term

4.6.2.6 std::string Term::getMaxJName ( ) const

Function used to get the name of the term in MaxJ.

Returns

name of the term in MaxJ

4.6.2.7 std::string Term::getName ( ) const

Function used to get the name of the term.

Returns

name of the term

4.6.2.8 virtual TermType Term::getType ( ) [pure virtual]

Function returning the type of the term.

Returns

Type of the term

Implemented in TriangleTerm.

4.6.2.9 void Term::setVariableName (std::string *variableName*)

Function used to set the name of the associated variable.

Parameters

variableName	name of the variable

4.6.2.10 void Term::writeHelperFunctionsToFile (std::ofstream & *file*) [static]

Function used to write all needed helper function (like (de)fuzzification) to the file.

Parameters

file	File to write to

4.6.2.11 void Term::writeTypeDefsToFile (std::ofstream & *file*) [static]

Function used to write all needed Type definitions to the file.

Parameters

file	File to write to

4.6.3 Member Data Documentation

4.6.3.1 std::string Term::name [protected]

Name of the term.

4.6.3.2 std::string Term::variableName [protected]

Name of the variable the term is belonging to.

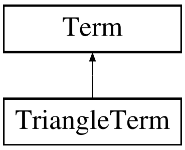
The documentation for this class was generated from the following files:

- term.h
- term.cpp

4.7 TriangleTerm Class Reference

`#include <triangleTerm.h>`

Inheritance diagram for TriangleTerm:



Public Member Functions

- TriangleTerm (std::string name, bool isIn)

  constructor
- virtual TermType getType ()

  Function returning the type of the term.
- void setValues (double start, double peak, double end)

  Function used to set all values needed to describe the triangle term.
- virtual void debugPrint ()

  Function providing some debug information regarding the correct parsing of the description.
- virtual void createFuzzySet (std::ofstream &file)

  Function used to generate the fuzzy set equaling the term.
- virtual void createFuzzyfication (std::ofstream &file)

  Function used to create a the fuzzification for a given term.
- virtual std::string getDefuzzification () const

  Function used to get the name of the defuzzification function.

Static Public Member Functions

- static void writeTypeDefsToFile (std::ofstream &file)

  Function used to write all needed Type definitions to the file.
- static void writeDataStructureCreationMethodeToFile (std::ofstream &file)

  Function to create a method used to gernerate triangle fuzzy sets.
- static void writeFuzzificationMethodeToFile (std::ofstream &file)

  Funtion to create a methode used for fuzzfication of triangle fuzzy sets.
- static void writeDefuzzificationMethodeToFile (std::ofstream &file)

  Funtion to create a methode used for defuzzification of triangle fuzzy sets.

Additional Inherited Members

4.7.1 Constructor & Destructor Documentation

4.7.1.1 TriangleTerm::TriangleTerm (std::string *name,* bool *isIn*)

constructor

Parameters

name	Name of the term
isIn	Flag indicating if the term is associated with an input or an output variable

4.7.2 Member Function Documentation

4.7.2.1 void TriangleTerm::createFuzzyfication (std::ofstream & *file*) [virtual]

Function used to create a the fuzzification for a given term.

Parameters

file	File to write to

Reimplemented from Term.

4.7.2.2 void TriangleTerm::createFuzzySet (std::ofstream & *file*) [virtual]

Function used to generate the fuzzy set equaling the term.

Parameters

file	File to write to

Implements Term.

4.7.2.3 void TriangleTerm::debugPrint ( ) [virtual]

Function providing some debug information regarding the correct parsing of the description.

Reimplemented from Term.

4.7.2.4 std::string TriangleTerm::getDefuzzification ( ) const [virtual]

Function used to get the name of the defuzzification function.

Returns

name of the defuzzification function

Implements Term.

4.7.2.5 TermType TriangleTerm::getType ( ) [virtual]

Function returning the type of the term.

Returns

Type of the term

Implements Term.

4.7.2.6 void TriangleTerm::setValues (double *start*, double *peak*, double *end*)

Function used to set all values needed to describe the triangle term.

Parameters

start	Start of the triangle
peak	Peak of the triangle
end	End of the triangle

4.7.2.7 void TriangleTerm::writeDataStructureCreationMethodeToFile (std::ofstream & *file*) [static]

Function to create a method used to gernerate triangle fuzzy sets.

Parameters

file	File to write to

4.7.2.8 void TriangleTerm::writeDefuzzificationMethodeToFile (std::ofstream & *file*) [static]

Funtion to create a methode used for defuzzification of triangle fuzzy sets.

Parameters

file	File to write to

4.7.2.9 void TriangleTerm::writeFuzzificationMethodeToFile (std::ofstream & *file*) [static]

Funtion to create a methode used for fuzzfication of triangle fuzzy sets.

Parameters

file	File to write to

4.7.2.10 void TriangleTerm::writeTypeDefsToFile (std::ofstream & *file*) [static]

Function used to write all needed Type definitions to the file.

Parameters

file	File to write to

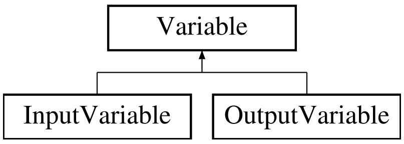
The documentation for this class was generated from the following files:

- triangleTerm.h
- triangleTerm.cpp

4.8 Variable Class Reference

```
#include <variable.h>
```

Inheritance diagram for Variable:



Public Member Functions

- Variable (std::string name)

    constructor

- virtual ∼Variable ()

    Deconstructor.

- void setRange (double rangeLow, double rangeHigh)

    Mehtode used to set the range of the variable.

- float getRangeLow ()

    Function to access the lower bound of the range of the variable.

- float getRangeHigh ()

    Function to access the upper bound of the range of the variable.

- void addTerm (Term ∗term)

    Function used to add a Term to the variable.

- std::string getName () const

    Function used to get the name of the variable.

- Term ∗ getTermToName (std::string name)

    Function used to get the term to a given name.

- virtual void createFuzzySets (std::ofstream &file)

    Function used to generate all fuzzy sets needed.

- virtual void debugPrint ()

    Function providing some debug information regarding the correct parsing of the description.

Protected Attributes

- std::string name

  Name of the variable.
- float rangeLow

  Lower bound of the range of the variable.
- float rangeHigh

  Upper bound of the range of the variable.
- std::list< Term * > termList

  List used to save all terms connected to the variable.

4.8.1 Constructor & Destructor Documentation

4.8.1.1 Variable::Variable (std::string *name*)

constructor

Parameters

name	Name of the variable to create

4.8.1.2 Variable::∼Variable ( ) [virtual]

Deconstructor.

4.8.2 Member Function Documentation

4.8.2.1 void Variable::addTerm (Term * *term*)

Function used to add a Term to the variable.

Parameters

term	Pointer to the term

4.8.2.2 void Variable::createFuzzySets (std::ofstream & *file*) [virtual]

Function used to generate all fuzzy sets needed.

Parameters

file	File to write to

Reimplemented in InputVariable, and OutputVariable.

4.8.2.3 void Variable::debugPrint ( ) [virtual]

Function providing some debug information regarding the correct parsing of the description.

4.8.2.4 std::string Variable::getName ( ) const

Function used to get the name of the variable.

Returns

Name of the variable

4.8.2.5 float Variable::getRangeHigh ( )

Function to access the upper bound of the range of the variable.

Returns

Upper bound of the range of the variable

4.8.2.6 float Variable::getRangeLow ( )

Function to access the lower bound of the range of the variable.

Returns

Lower bound of the range of the variable

4.8.2.7 Term * Variable::getTermToName (std::string *name*)

Function used to get the term to a given name.

Parameters

name	Name of the term

Returns

Pointer to the term or NULL if not found

4.8.2.8 void Variable::setRange (double *rangeLow,* double *rangeHigh*)

Mehtode used to set the range of the variable.

Parameters

rangeLow	Lower bound of the range of the variable
rangeHigh	Upper bound of the range of the variable

4.8.3 Member Data Documentation

4.8.3.1 std::string Variable::name `[protected]`

Name of the variable.

4.8.3.2 float Variable::rangeHigh `[protected]`

Upper bound of the range of the variable.

4.8.3.3 float Variable::rangeLow [protected]

Lower bound of the range of the variable.

4.8.3.4 std::list<Term*> Variable::termList [protected]

List used to save all terms connected to the variable.

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

- variable.h
- variable.cpp

Chapter 5

File Documentation

5.1 engine.cpp File Reference

```
#include <iostream>
#include <sstream>
#include <cstdlib>
#include <cmath>
#include <algorithm>
#include "engine.h"
#include "triangleTerm.h"
#include "strings.h"
```

Enumerations

- enum ParseStateEnum { Input, Output, Rule, Invalid }

5.1.1 Enumeration Type Documentation

5.1.1.1 enum ParseStateEnum

Enumerator

Input

Output

Rule

Invalid

5.2 engine.h File Reference

```
#include <string>
#include <fstream>
#include <list>
#include "inputVariable.h"
#include "outputVariable.h"
#include "evaluationRule.h"
```

Classes

- class Engine

5.3 evaluationRule.cpp File Reference

```
#include <iostream>
#include <list>
#include <sstream>
#include "evaluationRule.h"
```

5.4 evaluationRule.h File Reference

```
#include <list>
#include "term.h"
```

Classes

- class EvaluationRule

Enumerations

- enum Operator { AND, OR, EQUAL }

  enum containg all possible operators in evaluation rules

5.4.1 Enumeration Type Documentation

5.4.1.1 enum Operator

enum containg all possible operators in evaluation rules

Enumerator

AND

OR

EQUAL

5.5 FuzzyLogicGenerator.cpp File Reference

```
#include <string>
#include <iostream>
#include <cstdlib>
#include "engine.h"
```

Functions

- void showUsage ()
- int main (int argc, char ∗argv[ ])

5.5.1 Function Documentation

5.5.1.1 int main (int *argc,* char ∗ *argv[]*)

5.5.1.2 void showUsage ( )

5.6 inputVariable.cpp File Reference

```
#include "inputVariable.h"
```

5.7 inputVariable.h File Reference

```
#include "variable.h"
```

Classes

- class InputVariable

5.8 outputVariable.cpp File Reference

```
#include "outputVariable.h"
```

5.9 outputVariable.h File Reference

```
#include "variable.h"
```

Classes

- class OutputVariable

Macros

- #define OUTPUTARIABLE_H

5.9.1 Macro Definition Documentation

5.9.1.1 #define OUTPUTARIABLE_H

5.10 SimpleDimmerAppCpuCode.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include "MaxSLiCInterface.h"
#include "Maxfiles.h"
```

Functions

- void readFile (int size, float ∗inputAmbient, float ∗outputPower)
- void writeFile (int size, float ∗inputAmbient, float ∗outputPower)
- int main (void)

5.10.1 Function Documentation

5.10.1.1 int main (void)

5.10.1.2 void readFile (int *size,* float ∗ *inputAmbient,* float ∗ *outputPower*)

5.10.1.3 void writeFile (int *size,* float ∗ *inputAmbient,* float ∗ *outputPower*)

5.11 strings.cpp File Reference

```
#include "strings.h"
```

5.12 strings.h File Reference

```
#include <string>
```

Classes

- class Strings

  this class is used to save different strings which will than be coppied into the created source files

5.13 term.cpp File Reference

```
#include <iostream>
#include "term.h"
#include "triangleTerm.h"
#include "strings.h"
```

5.14 term.h File Reference

```
#include <string>
#include <fstream>
```

Classes

- class Term

Enumerations

- enum TermType { TRIANGLE, TermTypeCount }

  enum containing all possible (implemented) term types

5.14.1 Enumeration Type Documentation

5.14.1.1 enum TermType

enum containing all possible (implemented) term types

Enumerator

TRIANGLE

TermTypeCount

5.15 triangleTerm.cpp File Reference

```
#include <iostream>
#include "triangleTerm.h"
#include "strings.h"
```

5.16 triangleTerm.h File Reference

```
#include "term.h"
```

Classes

- class TriangleTerm

5.17 variable.cpp File Reference

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

5.18 variable.h File Reference

```
#include <string>
#include <list>
#include "term.h"
```

Classes

- class Variable

Index