IT UNIVERSITY OF COPENHAGEN

BDSA 2014

InheritanceRPC Documentation

ASSIGNMENT 38

Anders Wind Steffensen - awis@itu.dk
Christopher Blundell - cnbl@itu.dk
Pierre Mandas - ppma@itu.dk

Generated by Doxygen 1.8.8

Fri Sep 19 2014 22:00

Contents

Chapter 1

Namespace Index

1.1 Packages

Here are the packages with brief descriptions (if available):

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

Namespace Documentation

4.1 Package InheritanceRPC_Project

Classes

- class BinaryOperation

  An operation class which extends the IOperation interface, and therefore has an execute method which takes doubles as parametre. The class is meant to handle binary operations (operations with 2 doubles as input).
- class InheritanceRPC

  The class takes a string as input, and if the string is in the format of a reverse Polish calculator expression, then it will calculate and return the result. Features Can handle the operators: +, -, ∗, /, sqrt, cos, sin, abs and pow (^) Can handle negative numbers if written in the form -x Can handle doubles if written in the form x,x Tokens are separated by a whitespace. If the input is invalid an exception is thrown. The class uses a dictionary<string, IOperation> to handle the operations. To add an operation simply add it to the dictionary. The class currently has IOperation classes for unary and binary operations but tienary and so on can be added by creating an inner class which implements the IOperation interface.s
- class InheritanceRPCtests
- interface IOperation

  An interface which has an execute method which can take doubles as parametres.
- class UnaryOperation

  An operation class which extends the IOperation interface, and therefore has an execute method which takes doubles as parametre. The class is meant to handle unary operations (operations with 1 double as input).

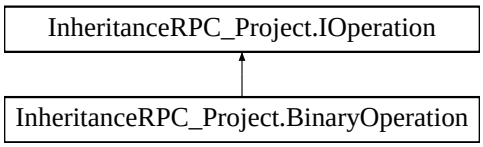
Chapter 5

Class Documentation

5.1 InheritanceRPC_Project.BinaryOperation Class Reference

An operation class which extends the IOperation interface, and therefore has an execute method which takes doubles as parametre. The class is meant to handle binary operations (operations with 2 doubles as input).

Inheritance diagram for InheritanceRPC_Project.BinaryOperation:



Public Member Functions

- BinaryOperation (Func< double, double, double > operation)

  The constructor of the operation, which is used to give the class the function which it will use in the execute method.
- double Execute (double arg1, params double[] argn)

  The Execute method which will use the operation field of the class to calculate the output given the 2 input doubles.

5.1.1 Detailed Description

An operation class which extends the IOperation interface, and therefore has an execute method which takes doubles as parametre. The class is meant to handle binary operations (operations with 2 doubles as input).

5.1.2 Constructor & Destructor Documentation

5.1.2.1 InheritanceRPC_Project.BinaryOperation.BinaryOperation (Func< double, double, double > *operation*)

The constructor of the operation, which is used to give the class the function which it will use in the execute method.

Parameters

operation	The operation wanted as a function(2 inputs 1 output)

5.1.3 Member Function Documentation

5.1.3.1 double InheritanceRPC_Project.BinaryOperation.Execute (double *arg1,* params double[ ] *argn*)

The Execute method which will use the operation field of the class to calculate the output given the 2 input doubles.

Parameters

arg1	The first double given
argn	A number of doubles in an array

Returns

The result of the operation field given arg1 and argn[0]

Implements InheritanceRPC_Project.IOperation.

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

- E:/User (E)/Programming (E)/BDSA-Exercises/BDSA2014/InheritanceRPC_Project/InheritanceRPC.cs

5.2 InheritanceRPC_Project.InheritanceRPC Class Reference

The class takes a string as input, and if the string is in the format of a reverse Polish calculator expression, then it will calculate and return the result. Features Can handle the operators: +, -, ∗, /, sqrt, cos, sin, abs and pow (^) Can handle negative numbers if written in the form -x Can handle doubles if written in the form x,x Tokens are separated by a whitespace. If the input is invalid an exception is thrown. The class uses a dictionary<string, IOperation> to handle the operations. To add an operation simply add it to the dictionary. The class currently has IOperation classes for unary and binary operations but tienary and so on can be added by creating an inner class which implements the IOperation interface.s

Public Member Functions

- InheritanceRPC ()

  Constructor which sets up the class.
- double CalculateExpression (string rpce)

  Calculates the input reverse calculator string expression. Throws exceptions if the input is not in the RPC format.

5.2.1 Detailed Description

The class takes a string as input, and if the string is in the format of a reverse Polish calculator expression, then it will calculate and return the result. Features Can handle the operators: +, -, ∗, /, sqrt, cos, sin, abs and pow (^) Can handle negative numbers if written in the form -x Can handle doubles if written in the form x,x Tokens are separated by a whitespace. If the input is invalid an exception is thrown. The class uses a dictionary<string, IOperation> to handle the operations. To add an operation simply add it to the dictionary. The class currently has IOperation classes for unary and binary operations but tienary and so on can be added by creating an inner class which implements the IOperation interface.s

5.2.2 Constructor & Destructor Documentation

5.2.2.1 InheritanceRPC_Project.InheritanceRPC.InheritanceRPC ( )

Constructor which sets up the class.

5.2.3 Member Function Documentation

5.2.3.1 double InheritanceRPC_Project.InheritanceRPC.CalculateExpression (string *rpce*)

Calculates the input reverse calculator string expression. Throws exceptions if the input is not in the RPC format.

Parameters

rpce	A String in the Reverse Polish Calculator expression format

Returns

0 if input is null or empty, otherwise returns the result

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

- E:/User (E)/Programming (E)/BDSA-Exercises/BDSA2014/InheritanceRPC_Project/InheritanceRPC.cs

5.3 InheritanceRPC_Project.InheritanceRPCtests Class Reference

Public Member Functions

- void **Setup** ()
- void TestNullInput ()

  Testcase where the input is null - should return 0.
- void TestEmptyInput ()

  Testcase where the input is "" - should return 0.
- void TestPlusOperator ()

  Testcase where the input contains a + operator (5 5 +) - should return 10.
- void TestMinusOperator ()

  Testcase where the input contains a - operator (3 5 -) - should return -2.
- void TestMultiplyOperator ()

  Testcase where the input contains a ∗ operator (3 4 ∗) - should return 12.
- void TestDivideOperator ()

  Testcase where the input contains a / operator (12 4 ∗) - should return 3.
- void TestPowOperator ()

  Testcase where the input contains a pov (^) operator (5 2 pov) - should return 25. (2 5 ^) - should return 32.
- void TestSinOperator ()

  Testcase where the input contains sin operator (30 sin) - should return Math.Sin(30).
- void TestCosOperator ()

  Testcase where the input contains cos operator (30 cos) - should return Math.Cos(30).
- void TestSqrtOperator ()

  Testcase where the input contains sqrt operator (25 sqrt) - should return Math.Cos(5).
- void TestAbsOperator ()

  Testcase where the input contains abs operator (25 abs, -25 abs, 0 abs, -0 abs) - should return Math.Cos(5).
- void TestComplexExpressionOperator ()

  Testcase where the input contains multiple operators and values.
- void TestExpressionWithLettersOperator ()

  Testcase where the input contains letters and letters combined with numbers.

5.3.1 Member Function Documentation

5.3.1.1 void InheritanceRPC_Project.InheritanceRPCtests.TestAbsOperator ( )

Testcase where the input contains abs operator (25 abs, -25 abs, 0 abs, -0 abs) - should return Math.Cos(5).

5.3.1.2 void InheritanceRPC_Project.InheritanceRPCtests.TestComplexExpressionOperator ( )

Testcase where the input contains multiple operators and values.

5.3.1.3 void InheritanceRPC_Project.InheritanceRPCtests.TestCosOperator ( )

Testcase where the input contains cos operator (30 cos) - should return Math.Cos(30).

5.3.1.4 void InheritanceRPC_Project.InheritanceRPCtests.TestDivideOperator ( )

Testcase where the input contains a / operator (12 4 $*$) - should return 3.

5.3.1.5 void InheritanceRPC_Project.InheritanceRPCtests.TestEmptyInput ( )

Testcase where the input is "" - should return 0.

5.3.1.6 void InheritanceRPC_Project.InheritanceRPCtests.TestExpressionWithLettersOperator ( )

Testcase where the input contains letters and letters combined with numbers.

5.3.1.7 void InheritanceRPC_Project.InheritanceRPCtests.TestMinusOperator ( )

Testcase where the input contains a - operator (3 5 -) - should return -2.

5.3.1.8 void InheritanceRPC_Project.InheritanceRPCtests.TestMultiplyOperator ( )

Testcase where the input contains a $*$ operator (3 4 $*$) - should return 12.

5.3.1.9 void InheritanceRPC_Project.InheritanceRPCtests.TestNullInput ( )

Testcase where the input is null - should return 0.

5.3.1.10 void InheritanceRPC_Project.InheritanceRPCtests.TestPlusOperator ( )

Testcase where the input contains a + operator (5 5 +) - should return 10.

5.3.1.11 void InheritanceRPC_Project.InheritanceRPCtests.TestPowOperator ( )

Testcase where the input contains a pov ($\hat{}$) operator (5 2 pov) - should return 25. (2 5 $\hat{}$) - should return 32.

5.3.1.12 void InheritanceRPC_Project.InheritanceRPCtests.TestSinOperator ( )

Testcase where the input contains sin operator (30 sin) - should return Math.Sin(30).

5.3.1.13 void InheritanceRPC_Project.InheritanceRPCtests.TestSqrtOperator ( )

Testcase where the input contains sqrt operator (25 sqrt) - should return Math.Cos(5).

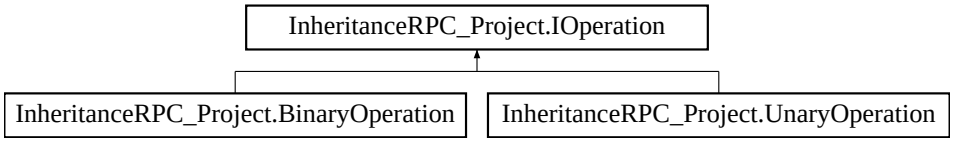
The documentation for this class was generated from the following file:

- E:/User (E)/Programming (E)/BDSA-Exercises/BDSA2014/InheritanceRPC_Project/InheritanceRPC.tests.cs

5.4 InheritanceRPC_Project.IOperation Interface Reference

An interface which has an execute method which can take doubles as parametres.

Inheritance diagram for InheritanceRPC_Project.IOperation:



Public Member Functions

- double **Execute** (double arg1, params double[] argn)

5.4.1 Detailed Description

An interface which has an execute method which can take doubles as parametres.

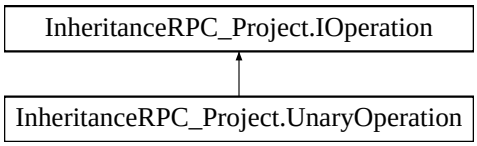
The documentation for this interface was generated from the following file:

- E:/User (E)/Programming (E)/BDSA-Exercises/BDSA2014/InheritanceRPC_Project/InheritanceRPC.cs

5.5 InheritanceRPC_Project.UnaryOperation Class Reference

An operation class which extends the IOperation interface, and therefore has an execute method which takes doubles as parametre. The class is meant to handle unary operations (operations with 1 double as input).

Inheritance diagram for InheritanceRPC_Project.UnaryOperation:



Public Member Functions

- UnaryOperation (Func< double, double > operation)

  The constructor of the operation, which is used to give the class the function which it will use in the execute method.
- double Execute (double arg1, params double[] argn)

  The Execute method which will use the operation field of the class to calculate the output given the input double.

5.5.1 Detailed Description

An operation class which extends the IOperation interface, and therefore has an execute method which takes doubles as parametre. The class is meant to handle unary operations (operations with 1 double as input).

5.5.2 Constructor & Destructor Documentation

5.5.2.1 InheritanceRPC_Project.UnaryOperation.UnaryOperation (Func< double, double > *operation*)

The constructor of the operation, which is used to give the class the function which it will use in the execute method.

Parameters

operation	The operation wanted as a function(1 input 1 output)

5.5.3 Member Function Documentation

5.5.3.1 double InheritanceRPC_Project.UnaryOperation.Execute (double *arg1*, params double[] *argn*)

The Execute method which will use the operation field of the class to calculate the output given the input double.

Parameters

arg1	The first double given
argn	A number of doubles in an array

Returns

The result of the operation field given arg1

Implements InheritanceRPC_Project.IOperation.

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

- E:/User (E)/Programming (E)/BDSA-Exercises/BDSA2014/InheritanceRPC_Project/InheritanceRPC.cs