MyVensim

Generated by Doxygen 1.9.4

1 Eng1 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 Body Class Reference 9

5.1.1 Detailed Description 9

5.1.2 Constructor & Destructor Documentation 10

5.1.2.1 Body() 10

5.1.2.2 ∼Body() 10

5.1.3 Member Function Documentation 10

5.1.3.1 attach() 10

5.1.3.2 detach() 10

5.1.3.3 refCount() 10

5.2 ComplexFlow Class Reference 11

5.2.1 Detailed Description 12

5.2.2 Constructor & Destructor Documentation 12

5.2.2.1 ComplexFlow() 12

5.2.3 Member Function Documentation 12

5.2.3.1 expression() 12

5.3 ExponencialFlow Class Reference 13

5.3.1 Detailed Description 14

5.3.2 Constructor & Destructor Documentation 14

5.3.2.1 ExponencialFlow() 14

5.3.3 Member Function Documentation 14

5.3.3.1 expression() 14

5.4 Flow Class Reference 15

5.4.1 Constructor & Destructor Documentation 15

5.4.1.1 ∼Flow() 15

5.4.2 Member Function Documentation 15

5.4.2.1 expression() 16

5.4.2.2 getName() 16

5.4.2.3 getSystemBegin() 16

5.4.2.4 getSystemEnd() 16

5.4.2.5 getValue() 16

5.4.2.6 setName() 16

5.4.2.7 setSystemBegin() 17

5.4.2.8 setSystemEnd() 17
5.4.2.9 setValue() 17
5.5 FlowBody Class Reference 18
5.5.1 Detailed Description 19
5.5.2 Constructor & Destructor Documentation 19
5.5.2.1 ∼FlowBody() 19
5.5.2.2 FlowBody() [1/2] 19
5.5.2.3 FlowBody() [2/2] 20
5.5.3 Member Function Documentation 20
5.5.3.1 expression() 20
5.5.3.2 getName() 20
5.5.3.3 getSystemBegin() 21
5.5.3.4 getSystemEnd() 21
5.5.3.5 getValue() 21
5.5.3.6 operator=() 21
5.5.3.7 setName() 21
5.5.3.8 setSystemBegin() 22
5.5.3.9 setSystemEnd() 22
5.5.3.10 setValue() 22
5.5.4 Member Data Documentation 22
5.5.4.1 name 22
5.5.4.2 systemBegin 23
5.5.4.3 systemEnd 23
5.5.4.4 value 23
5.6 FlowHandle< Flow_IMPL > Class Template Reference 23
5.6.1 Constructor & Destructor Documentation 24
5.6.1.1 ∼FlowHandle() 24
5.6.1.2 FlowHandle() 24
5.6.2 Member Function Documentation 24
5.6.2.1 expression() 24
5.6.2.2 getName() 25
5.6.2.3 getSystemBegin() 25
5.6.2.4 getSystemEnd() 25
5.6.2.5 getValue() 25
5.6.2.6 setName() 25
5.6.2.7 setSystemBegin() 26
5.6.2.8 setSystemEnd() 26
5.6.2.9 setValue() 26
5.7 Handle< T > Class Template Reference 27
5.7.1 Detailed Description 27
5.7.2 Constructor & Destructor Documentation 27
5.7.2.1 Handle() [1/2] 28

5.7.2.2 ∼Handle() 28
5.7.2.3 Handle() [2/2] 28
5.7.3 Member Function Documentation 28
5.7.3.1 operator=() 28
5.7.4 Member Data Documentation 28
5.7.4.1 pImpl_ 29
5.8 LogisticalFlow Class Reference 29
5.8.1 Detailed Description 30
5.8.2 Constructor & Destructor Documentation 30
5.8.2.1 LogisticalFlow() 30
5.8.3 Member Function Documentation 31
5.8.3.1 expression() 31
5.9 Model Class Reference 31
5.9.1 Constructor & Destructor Documentation 32
5.9.1.1 ∼Model() 32
5.9.2 Member Function Documentation 32
5.9.2.1 add() [1/2] 32
5.9.2.2 add() [2/2] 33
5.9.2.3 createFlow() 33
5.9.2.4 createModel() 33
5.9.2.5 createSystem() 33
5.9.2.6 endFlows() 34
5.9.2.7 endModels() 34
5.9.2.8 endSystems() 34
5.9.2.9 getFlowsIterator() 34
5.9.2.10 getModelsIterator() 34
5.9.2.11 getName() 34
5.9.2.12 getSystemsIterator() 35
5.9.2.13 getTime() 35
5.9.2.14 setName() 35
5.9.2.15 setTime() 35
5.9.2.16 simulate() 36
5.10 ModelBody Class Reference 36
5.10.1 Detailed Description 38
5.10.2 Constructor & Destructor Documentation 38
5.10.2.1 ∼ModelBody() 38
5.10.2.2 ModelBody() 38
5.10.3 Member Function Documentation 38
5.10.3.1 add() [1/2] 38
5.10.3.2 add() [2/2] 39
5.10.3.3 createModel() 39
5.10.3.4 createSystem() 39

5.10.3.5 endFlows() 39
5.10.3.6 endModels() 39
5.10.3.7 endSystems() 39
5.10.3.8 getFlowsIterator() 40
5.10.3.9 getModelsIterator() 40
5.10.3.10 getName() 40
5.10.3.11 getSystemsIterator() 40
5.10.3.12 getTime() 40
5.10.3.13 setName() 40
5.10.3.14 setTime() 40
5.10.3.15 simulate() 41
5.10.4 Member Data Documentation 41
5.10.4.1 flows 41
5.10.4.2 models 41
5.10.4.3 name 41
5.10.4.4 systems 42
5.10.4.5 time 42
5.11 ModelHandle Class Reference 42
5.11.1 Constructor & Destructor Documentation 43
5.11.1.1 ∼ModelHandle() 43
5.11.1.2 ModelHandle() 43
5.11.2 Member Function Documentation 43
5.11.2.1 add() [1/2] 43
5.11.2.2 add() [2/2] 44
5.11.2.3 createModel() 44
5.11.2.4 createSystem() 44
5.11.2.5 endFlows() 44
5.11.2.6 endModels() 45
5.11.2.7 endSystems() 45
5.11.2.8 getFlowsIterator() 45
5.11.2.9 getModelsIterator() 45
5.11.2.10 getName() 45
5.11.2.11 getSystemsIterator() 46
5.11.2.12 getTime() 46
5.11.2.13 setName() 46
5.11.2.14 setTime() 46
5.11.2.15 simulate() 47
5.12 System Class Reference 47
5.12.1 Constructor & Destructor Documentation 48
5.12.1.1 ∼System() 48
5.12.2 Member Function Documentation 48
5.12.2.1 getName() 48

5.12.2.2 getValue() 48
5.12.2.3 setName() 48
5.12.2.4 setValue() 49
5.13 SystemBody Class Reference 49
5.13.1 Detailed Description 50
5.13.2 Constructor & Destructor Documentation 50
5.13.2.1 ∼SystemBody() 51
5.13.2.2 SystemBody() [1/2] 51
5.13.2.3 SystemBody() [2/2] 51
5.13.3 Member Function Documentation 51
5.13.3.1 getName() 51
5.13.3.2 getValue() 52
5.13.3.3 operator=() 52
5.13.3.4 setName() 52
5.13.3.5 setValue() 52
5.13.4 Member Data Documentation 53
5.13.4.1 name 53
5.13.4.2 value 53
5.14 SystemHandle Class Reference 53
5.14.1 Constructor & Destructor Documentation 54
5.14.1.1 ∼SystemHandle() 54
5.14.1.2 SystemHandle() 55
5.14.2 Member Function Documentation 55
5.14.2.1 getName() 55
5.14.2.2 getValue() 55
5.14.2.3 setName() 55
5.14.2.4 setValue() 55
5.15 UnitTestFlow Class Reference 56
5.15.1 Detailed Description 57
5.15.2 Constructor & Destructor Documentation 57
5.15.2.1 UnitTestFlow() 57
5.15.3 Member Function Documentation 58
5.15.3.1 expression() 58
5.16 UnitTestFlow2 Class Reference 58
5.16.1 Detailed Description 59
5.16.2 Constructor & Destructor Documentation 59
5.16.2.1 UnitTestFlow2() 59
5.16.3 Member Function Documentation 60
5.16.3.1 expression() 60

6 File Documentation **61**

6.1 cmake-build-debug/CMakeCache.txt File Reference 61

6.2 cmake-build-debug/CMakeFiles/3.22.3/CompilerIdC/CMakeCCompilerId.c File Reference 61
6.2.1 Macro Definition Documentation 61
6.2.1.1 __has_include 62
6.2.1.2 ARCHITECTURE_ID 62
6.2.1.3 C_VERSION 62
6.2.1.4 COMPILER_ID 62
6.2.1.5 DEC 62
6.2.1.6 HEX 62
6.2.1.7 PLATFORM_ID 63
6.2.1.8 STRINGIFY 63
6.2.1.9 STRINGIFY_HELPER 63
6.2.2 Function Documentation 63
6.2.2.1 main() 63
6.2.3 Variable Documentation 63
6.2.3.1 info_arch 63
6.2.3.2 info_compiler 63
6.2.3.3 info_language_extensions_default 64
6.2.3.4 info_language_standard_default 64
6.2.3.5 info_platform 64
6.3 cmake-build-debug/CMakeFiles/3.22.3/CompilerIdCXX/CMakeCXXCompilerId.cpp File Reference . . 64
6.3.1 Macro Definition Documentation 65
6.3.1.1 __has_include 65
6.3.1.2 ARCHITECTURE_ID 65
6.3.1.3 COMPILER_ID 65
6.3.1.4 CXX_STD 65
6.3.1.5 DEC 65
6.3.1.6 HEX 66
6.3.1.7 PLATFORM_ID 66
6.3.1.8 STRINGIFY 66
6.3.1.9 STRINGIFY_HELPER 66
6.3.2 Function Documentation 66
6.3.2.1 main() 66
6.3.3 Variable Documentation 66
6.3.3.1 info_arch 67
6.3.3.2 info_compiler 67
6.3.3.3 info_language_extensions_default 67
6.3.3.4 info_language_standard_default 67
6.3.3.5 info_platform 67
6.4 cmake-build-debug/CMakeFiles/clion-environment.txt File Reference 68
6.5 cmake-build-debug/CMakeFiles/clion-log.txt File Reference 68
6.6 cmake-build-debug/CMakeFiles/TargetDirectories.txt File Reference 68
6.7 CMakeLists.txt File Reference 68

6.8 README.md File Reference 68
6.9 src/lib/Flow.h File Reference 68
6.10 Flow.h 69
6.11 src/lib/FlowImplementation.cpp File Reference 70
6.12 src/lib/FlowImplementation.h File Reference 70
6.13 FlowImplementation.h 71
6.14 src/lib/handlebody.h File Reference 73
6.14.1 Macro Definition Documentation 73
6.14.1.1 DEBUGING 73
6.14.2 Variable Documentation 74
6.14.2.1 numBodyCreated 74
6.14.2.2 numBodyDeleted 74
6.14.2.3 numHandleCreated 74
6.14.2.4 numHandleDeleted 74
6.15 handlebody.h 74
6.16 src/lib/Model.h File Reference 75
6.17 Model.h 76
6.18 src/lib/ModelImplementation.cpp File Reference 77
6.19 src/lib/ModelImplementation.h File Reference 78
6.20 ModelImplementation.h 80
6.21 src/lib/System.h File Reference 81
6.22 System.h 82
6.23 src/lib/SystemImplementation.cpp File Reference 83
6.24 src/lib/SystemImplementation.h File Reference 83
6.25 SystemImplementation.h 84
6.26 src/main.cpp File Reference 85
6.26.1 Function Documentation 85
6.26.1.1 main() 85
6.27 test/functional/main.cpp File Reference 86
6.27.1 Macro Definition Documentation 86
6.27.1.1 DEBUGING 87
6.27.2 Function Documentation 87
6.27.2.1 main() 87
6.27.3 Variable Documentation 87
6.27.3.1 numBodyCreated 87
6.27.3.2 numBodyDeleted 87
6.27.3.3 numHandleCreated 87
6.27.3.4 numHandleDeleted 87
6.28 test/unit/main.cpp File Reference 88
6.28.1 Macro Definition Documentation 89
6.28.1.1 DEBUGING 89
6.28.2 Function Documentation 89

6.28.2.1 main() 89
6.28.3 Variable Documentation 89
6.28.3.1 numBodyCreated 89
6.28.3.2 numBodyDeleted 89
6.28.3.3 numHandleCreated 89
6.28.3.4 numHandleDeleted 89
6.29 test/functional/FunctionalTests.cpp File Reference 90
6.29.1 Function Documentation 90
6.29.1.1 ComplexTest() 90
6.29.1.2 ExponencialTest() 90
6.29.1.3 LogisticalTest() 91
6.30 test/functional/FunctionalTests.h File Reference 91
6.30.1 Function Documentation 92
6.30.1.1 ComplexTest() 92
6.30.1.2 ExponencialTest() 92
6.30.1.3 LogisticalTest() 92
6.31 FunctionalTests.h 93
6.32 test/unit/unitFlow.cpp File Reference 94
6.32.1 Function Documentation 94
6.32.1.1 runUnitTestsFlow() 95
6.32.1.2 unitFlowDefaultConstructor() 95
6.32.1.3 unitFlowDestructor() 95
6.32.1.4 unitFlowExpression() 95
6.32.1.5 unitFlowGetName() 95
6.32.1.6 unitFlowGetSystemBegin() 95
6.32.1.7 unitFlowGetSystemEnd() 95
6.32.1.8 unitFlowGetValue() 95
6.32.1.9 unitFlowSetName() 96
6.32.1.10 unitFlowSetSystemBegin() 96
6.32.1.11 unitFlowSetSystemEnd() 96
6.32.1.12 unitFlowSetValue() 96
6.33 test/unit/unitFlow.h File Reference 96
6.33.1 Function Documentation 97
6.33.1.1 runUnitTestsFlow() 97
6.33.1.2 unitFlowDefaultConstructor() 98
6.33.1.3 unitFlowDestructor() 98
6.33.1.4 unitFlowExpression() 98
6.33.1.5 unitFlowGetName() 98
6.33.1.6 unitFlowGetSystemBegin() 98
6.33.1.7 unitFlowGetSystemEnd() 98
6.33.1.8 unitFlowGetValue() 98
6.33.1.9 unitFlowSetName() 98

6.33.1.10 unitFlowSetSystemBegin() . . . 99
6.33.1.11 unitFlowSetSystemEnd() . . . 99
6.33.1.12 unitFlowSetValue() . . . 99
6.34 unitFlow.h . . . 99
6.35 test/unit/unitModel.cpp File Reference . . . 100
6.35.1 Function Documentation . . . 100
6.35.1.1 runUnitTestsModel() . . . 101
6.35.1.2 unitModelAddFlow() . . . 101
6.35.1.3 unitModelAddSystem() . . . 101
6.35.1.4 unitModelCreateFlow() . . . 101
6.35.1.5 unitModelCreateModel() . . . 101
6.35.1.6 unitModelCreateSystem() . . . 101
6.35.1.7 unitModelDefaultConstructor() . . . 101
6.35.1.8 unitModelDestructor() . . . 101
6.35.1.9 unitModelGetName() . . . 102
6.35.1.10 unitModelGetTime() . . . 102
6.35.1.11 unitModelSetName() . . . 102
6.35.1.12 unitModelSetTime() . . . 102
6.35.1.13 unitModelSimulate() . . . 102
6.36 test/unit/unitModel.h File Reference . . . 102
6.36.1 Function Documentation . . . 104
6.36.1.1 runUnitTestsModel() . . . 104
6.36.1.2 unitModelAddFlow() . . . 104
6.36.1.3 unitModelAddSystem() . . . 104
6.36.1.4 unitModelCreateFlow() . . . 104
6.36.1.5 unitModelCreateModel() . . . 104
6.36.1.6 unitModelCreateSystem() . . . 105
6.36.1.7 unitModelDefaultConstructor() . . . 105
6.36.1.8 unitModelDestructor() . . . 105
6.36.1.9 unitModelGetName() . . . 105
6.36.1.10 unitModelGetTime() . . . 105
6.36.1.11 unitModelSetName() . . . 105
6.36.1.12 unitModelSetTime() . . . 105
6.36.1.13 unitModelSimulate() . . . 105
6.37 unitModel.h . . . 106
6.38 test/unit/unitSystem.cpp File Reference . . . 107
6.38.1 Function Documentation . . . 107
6.38.1.1 runUnitTestsSystem() . . . 108
6.38.1.2 unitSystemDefaultConstructor() . . . 108
6.38.1.3 unitSystemDestructor() . . . 108
6.38.1.4 unitSystemGetName() . . . 108
6.38.1.5 unitSystemGetValue() . . . 108

6.38.1.6 unitSystemSetName() 108
6.38.1.7 unitSystemSetValue() 108
6.39 test/unit/unitSystem.h File Reference 109
6.39.1 Function Documentation 110
6.39.1.1 runUnitTestsSystem() 110
6.39.1.2 unitSystemDefaultConstructor() 110
6.39.1.3 unitSystemDestructor() 110
6.39.1.4 unitSystemGetName() 110
6.39.1.5 unitSystemGetValue() 110
6.39.1.6 unitSystemSetName() 111
6.39.1.7 unitSystemSetValue() 111
6.40 unitSystem.h 111
6.41 test/unit/unitTests.cpp File Reference 112
6.41.1 Function Documentation 112
6.41.1.1 runGlobal() 112
6.42 test/unit/unitTests.h File Reference 113
6.42.1 Function Documentation 114
6.42.1.1 runGlobal() 114
6.43 unitTests.h 114

Index **115**

Chapter 1

Eng1

Projeto individual de Engenharia de Software 1

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Body 9
    FlowBody 18
        ComplexFlow 11
        ExponencialFlow 13
        LogisticalFlow 29
        UnitTestFlow 56
        UnitTestFlow2 58
    ModelBody 36
    SystemBody 49
Flow 15
    FlowHandle< Flow_IMPL > 23
Handle< T > 27
Handle< Flow_IMPL > 27
    FlowHandle< Flow_IMPL > 23
Handle< ModelBody > 27
    ModelHandle 42
Handle< SystemBody > 27
    SystemHandle 53
Model 31
    ModelHandle 42
System 47
    SystemHandle 53

Chapter 3

Class Index

3.1 Class List

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

Body
The class Implementation was implemented based on the class teCounted writed by Ricardo Cartaxo and Gilberto Câmara and founded in the geographic library TerraLib 9
ComplexFlow 11
ExponencialFlow 13
Flow 15
FlowBody 18
FlowHandle< Flow_IMPL > 23
Handle< T >
The classes Handle and Body implements the "bridge" design pattern (also known as "handle/body idiom") 27
LogisticalFlow 29
Model 31
ModelBody 36
ModelHandle 42
System 47
SystemBody 49
SystemHandle 53
UnitTestFlow 56
UnitTestFlow2 58

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

cmake-build-debug/CMakeFiles/3.22.3/CompilerIdC/CMakeCCompilerId.c 61
cmake-build-debug/CMakeFiles/3.22.3/CompilerIdCXX/CMakeCXXCompilerId.cpp 64
src/main.cpp 85
src/lib/Flow.h 68
src/lib/FlowImplementation.cpp 70
src/lib/FlowImplementation.h 70
src/lib/handlebody.h 73
src/lib/Model.h 75
src/lib/ModelImplementation.cpp 77
src/lib/ModelImplementation.h 78
src/lib/System.h 81
src/lib/SystemImplementation.cpp 83
src/lib/SystemImplementation.h 83
test/functional/FunctionalTests.cpp 90
test/functional/FunctionalTests.h 91
test/functional/main.cpp 86
test/unit/main.cpp 88
test/unit/unitFlow.cpp 94
test/unit/unitFlow.h 96
test/unit/unitModel.cpp 100
test/unit/unitModel.h 102
test/unit/unitSystem.cpp 107
test/unit/unitSystem.h 109
test/unit/unitTests.cpp 112
test/unit/unitTests.h 113

Chapter 5

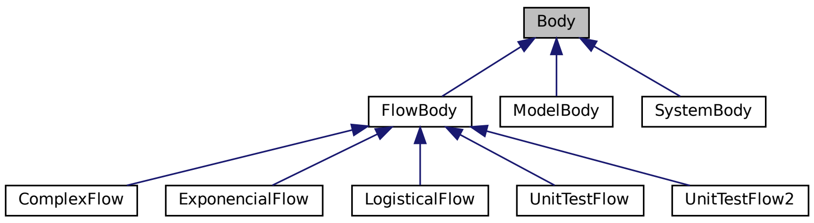
Class Documentation

5.1 Body Class Reference

The class Implementation was implemented based on the class teCounted writed by Ricardo Cartaxo and Gilberto Câmara and founded in the geographic library TerraLib.

```
#include <handlebody.h>
```

Inheritance diagram for Body:



Public Member Functions

- Body ()

  Constructor: zero references when the object is being built.
- void attach ()

  Increases the number of references to this object.
- void detach ()
- int refCount ()

  Returns the number of references to this object.
- virtual ∼Body ()

  Destructor.

5.1.1 Detailed Description

The class Implementation was implemented based on the class teCounted writed by Ricardo Cartaxo and Gilberto Câmara and founded in the geographic library TerraLib.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 Body()

```
Body::Body ( )  [inline]
```

Constructor: zero references when the object is being built.

5.1.2.2 ∼Body()

```
virtual Body::∼Body ( )  [inline], [virtual]
```

Destructor.

5.1.3 Member Function Documentation

5.1.3.1 attach()

```
void Body::attach ( )  [inline]
```

Increases the number of references to this object.

5.1.3.2 detach()

```
void Body::detach ( )  [inline]
```

Decreases the number of references to this object. Destroy it if there are no more references to it

5.1.3.3 refCount()

```
int Body::refCount ( )  [inline]
```

Returns the number of references to this object.

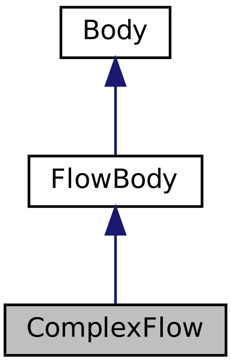
The documentation for this class was generated from the following file:

- src/lib/handlebody.h

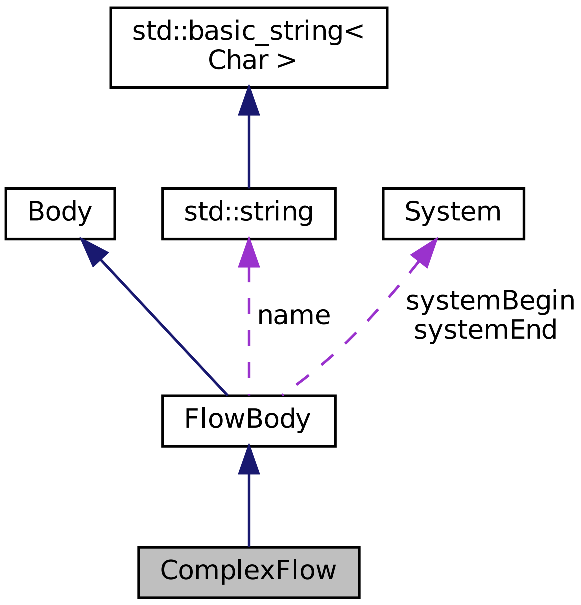
5.2 ComplexFlow Class Reference

```
#include <FunctionalTests.h>
```

Inheritance diagram for ComplexFlow:



Collaboration diagram for ComplexFlow:



Public Member Functions

- ComplexFlow (std::string name="", System ∗systemOut=NULL, System ∗systemIn=NULL)
- double expression () override

Additional Inherited Members

5.2.1 Detailed Description

Flow that converges energy from a model to another exponencialy with 1% of the end system per timestep

5.2.2 Constructor & Destructor Documentation

5.2.2.1 ComplexFlow()

```
ComplexFlow::ComplexFlow (
            std::string name = "",
            System * systemOut = NULL,
            System * systemIn = NULL )   [inline]
```

Default constructor

Parameters

name	Inital flow name
value	Inital flow value
systemBegin	Inital system where the flow comes from
systemEnd	Inital system where the flow goes to

Returns

Complex flow with initial name, value, systemBegin and systemEnd

5.2.3 Member Function Documentation

5.2.3.1 expression()

```
double ComplexFlow::expression ( )  [inline], [override], [virtual]
```

Complex expression

Implements FlowBody.

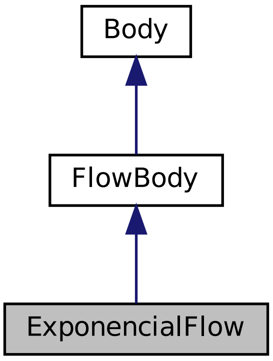
The documentation for this class was generated from the following file:

- test/functional/FunctionalTests.h

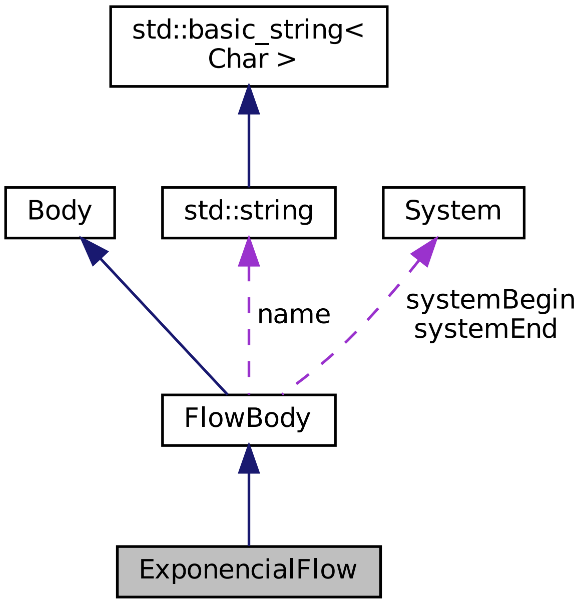
5.3 ExponencialFlow Class Reference

```
#include <FunctionalTests.h>
```

Inheritance diagram for ExponencialFlow:



Collaboration diagram for ExponencialFlow:



Public Member Functions

- ExponencialFlow (std::string name="", System ∗systemOut=NULL, System ∗systemIn=NULL)
- double expression () override

Additional Inherited Members

5.3.1 Detailed Description

Flow that converges energy from a model to another exponencialy with 1% of the initial system per timestep

5.3.2 Constructor & Destructor Documentation

5.3.2.1 ExponencialFlow()

```
ExponencialFlow::ExponencialFlow (
            std::string name = "",
            System * systemOut = NULL,
            System * systemIn = NULL )  [inline]
```

Default constructor

Parameters

name	Inital flow name
value	Inital flow value
systemBegin	Inital system where the flow comes from
systemEnd	Inital system where the flow goes to

Returns

Exponencial flow with initial name, value, systemBegin and systemEnd

5.3.3 Member Function Documentation

5.3.3.1 expression()

```
double ExponencialFlow::expression ( )  [inline], [override], [virtual]
```

Exponencial expression

Implements FlowBody.

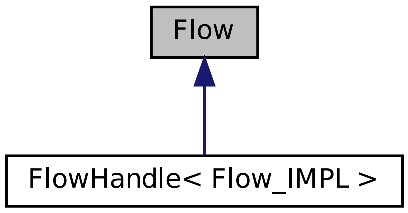
The documentation for this class was generated from the following file:

- test/functional/FunctionalTests.h

5.4 Flow Class Reference

```
#include <Flow.h>
```

Inheritance diagram for Flow:



Public Member Functions

- virtual ∼Flow ()=default
- virtual std::string getName () const =0
- virtual void setName (std::string)=0
- virtual double getValue () const =0
- virtual void setValue (double)=0
- virtual double expression ()=0
- virtual System ∗ getSystemBegin () const =0
- virtual void setSystemBegin (System ∗)=0
- virtual System ∗ getSystemEnd () const =0
- virtual void setSystemEnd (System ∗)=0

5.4.1 Constructor & Destructor Documentation

5.4.1.1 ∼Flow()

```
virtual Flow::∼Flow ( )  [virtual], [default]
```

Default destructor

5.4.2 Member Function Documentation

5.4.2.1 expression()

```
virtual double Flow::expression ( )  [pure virtual]
```

Sets the expression of the flow

Implemented in FlowHandle< Flow_IMPL >.

5.4.2.2 getName()

```
virtual std::string Flow::getName ( ) const  [pure virtual]
```

Get system name

Implemented in FlowHandle< Flow_IMPL >.

5.4.2.3 getSystemBegin()

```
virtual System * Flow::getSystemBegin ( ) const  [pure virtual]
```

Get systemBegin

Implemented in FlowHandle< Flow_IMPL >.

5.4.2.4 getSystemEnd()

```
virtual System * Flow::getSystemEnd ( ) const  [pure virtual]
```

Get systemEnd

Implemented in FlowHandle< Flow_IMPL >.

5.4.2.5 getValue()

```
virtual double Flow::getValue ( ) const  [pure virtual]
```

Get system value

Implemented in FlowHandle< Flow_IMPL >.

5.4.2.6 setName()

```
virtual void Flow::setName (
             std::string  )  [pure virtual]
```

Set system name

Parameters

n	Name for the flow

Implemented in FlowHandle< Flow_IMPL >.

5.4.2.7 setSystemBegin()

```
virtual void Flow::setSystemBegin (
             System * )  [pure virtual]
```

Set systemBegin

Parameters

system	SystemBegin for the flow

Implemented in FlowHandle< Flow_IMPL >.

5.4.2.8 setSystemEnd()

```
virtual void Flow::setSystemEnd (
             System * )  [pure virtual]
```

Set systemBegin

Parameters

system	SystemEnd for the flow

Implemented in FlowHandle< Flow_IMPL >.

5.4.2.9 setValue()

```
virtual void Flow::setValue (
             double  )  [pure virtual]
```

Set system value

Parameters

v	Value for the flow

Implemented in FlowHandle< Flow_IMPL >.

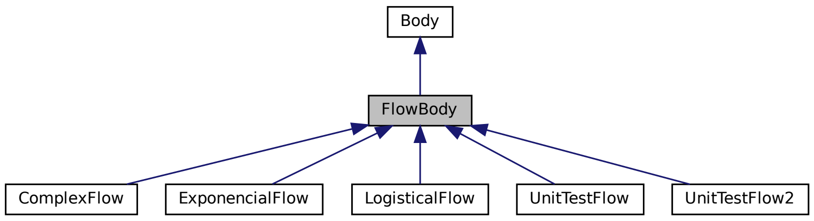
The documentation for this class was generated from the following file:

- src/lib/Flow.h

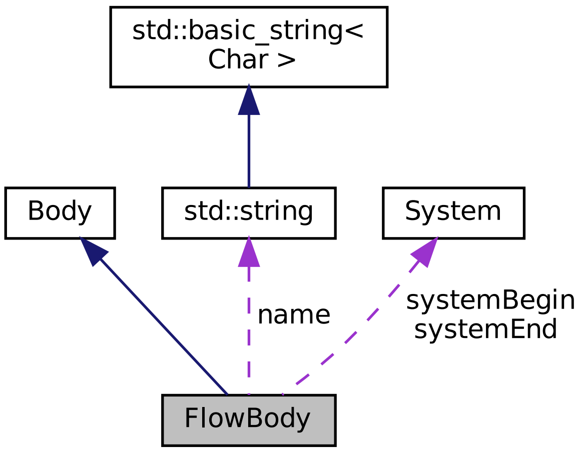
5.5 FlowBody Class Reference

```
#include <FlowImplementation.h>
```

Inheritance diagram for FlowBody:



Collaboration diagram for FlowBody:



Public Member Functions

- virtual ∼FlowBody ()
- FlowBody (std::string name="", System ∗systemBegin=NULL, System ∗systemEnd=NULL)
- FlowBody (const FlowBody &flow)
- FlowBody & operator= (const FlowBody &flow)
- virtual double expression ()=0
- std::string getName () const
- void setName (std::string n)
- double getValue () const
- void setValue (double v)
- System ∗ getSystemBegin () const
- void setSystemBegin (System ∗system)
- System ∗ getSystemEnd () const
- void setSystemEnd (System ∗system)

Protected Attributes

- std::string name
- double value
- System ∗ systemBegin
- System ∗ systemEnd

5.5.1 Detailed Description

Flow that converges energy from a model to another

5.5.2 Constructor & Destructor Documentation

5.5.2.1 ∼FlowBody()

```
FlowBody::∼FlowBody ( )  [virtual], [default]
```

Default destructor

5.5.2.2 FlowBody() [1/2]

```
FlowBody::FlowBody (
              std::string name = "",
              System ∗ systemBegin = NULL,
              System ∗ systemEnd = NULL )
```

Default constructor

Parameters

name	Inital flow name
value	Inital flow value
systemBegin	Inital system where the flow comes from
systemEnd	Inital system where the flow goes to

Returns

Flow with initial name, value, systemBegin and systemEnd

5.5.2.3 FlowBody() `[2/2]`

```
FlowBody::FlowBody (
              const FlowBody & flow )
```

Copy constructor

Parameters

flow	Flow to copy from

Returns

Copied flow

5.5.3 Member Function Documentation

5.5.3.1 expression()

```
virtual double FlowBody::expression ( ) [pure virtual]
```

Sets the expression of the flow

Implemented in ExponencialFlow, LogisticalFlow, ComplexFlow, UnitTestFlow, and UnitTestFlow2.

5.5.3.2 getName()

```
std::string FlowBody::getName ( ) const
```

Get system name

5.5.3.3 getSystemBegin()

```
System * FlowBody::getSystemBegin ( ) const
```

Get systemBegin

5.5.3.4 getSystemEnd()

```
System * FlowBody::getSystemEnd ( ) const
```

Get systemEnd

5.5.3.5 getValue()

```
double FlowBody::getValue ( ) const
```

Get system value

5.5.3.6 operator=()

```
FlowBody & FlowBody::operator= (
             const FlowBody & flow )
```

Copy Assignment Operator

Parameters

flow	Flow to copy from

Returns

Copied flow

5.5.3.7 setName()

```
void FlowBody::setName (
             std::string n )
```

Set system name

Parameters

n	Name for the flow

5.5.3.8 setSystemBegin()

```
void FlowBody::setSystemBegin (
             System * system )
```

Set systemBegin

Parameters

system	SystemBegin for the flow

5.5.3.9 setSystemEnd()

```
void FlowBody::setSystemEnd (
             System * system )
```

Set systemBegin

Parameters

system	SystemEnd for the flow

5.5.3.10 setValue()

```
void FlowBody::setValue (
             double v )
```

Set system value

Parameters

v	Value for the flow

5.5.4 Member Data Documentation

5.5.4.1 name

```
std::string FlowBody::name  [protected]
```

5.5.4.2 systemBegin

```
System* FlowBody::systemBegin  [protected]
```

5.5.4.3 systemEnd

```
System* FlowBody::systemEnd  [protected]
```

5.5.4.4 value

```
double FlowBody::value  [protected]
```

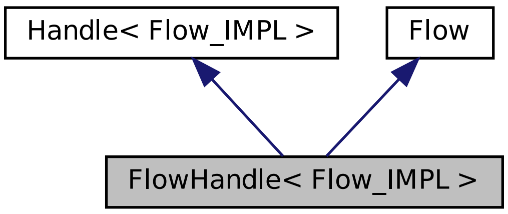
The documentation for this class was generated from the following files:

- src/lib/FlowImplementation.h
- src/lib/FlowImplementation.cpp

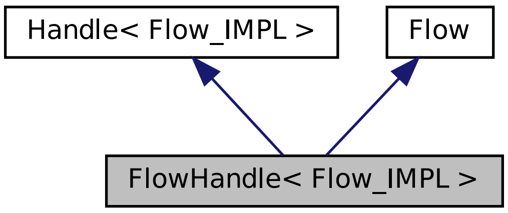
5.6 FlowHandle< Flow_IMPL > Class Template Reference

```
#include <FlowImplementation.h>
```

Inheritance diagram for FlowHandle< Flow_IMPL >:



Collaboration diagram for FlowHandle< Flow_IMPL >:



Public Member Functions

- ∼FlowHandle () override=default
- FlowHandle (std::string name="", System ∗systemBegin=NULL, System ∗systemEnd=NULL)
- double expression () override
- std::string getName () const override
- void setName (std::string name) override
- double getValue () const override
- void setValue (double v) override
- System ∗ getSystemBegin () const override
- void setSystemBegin (System ∗systemBegin) override
- System ∗ getSystemEnd () const override
- void setSystemEnd (System ∗systemEnd) override

Additional Inherited Members

5.6.1 Constructor & Destructor Documentation

5.6.1.1 ∼FlowHandle()

```
template<typename Flow_IMPL >
FlowHandle< Flow_IMPL >::∼FlowHandle ( ) [override], [default]
```

5.6.1.2 FlowHandle()

```
template<typename Flow_IMPL >
FlowHandle< Flow_IMPL >::FlowHandle (
             std::string name = "",
             System ∗ systemBegin = NULL,
             System ∗ systemEnd = NULL ) [inline]
```

5.6.2 Member Function Documentation

5.6.2.1 expression()

```
template<typename Flow_IMPL >
double FlowHandle< Flow_IMPL >::expression ( ) [inline], [override], [virtual]
```

Sets the expression of the flow

Implements Flow.

5.6.2.2 getName()

```
template<typename Flow_IMPL >
std::string FlowHandle< Flow_IMPL >::getName ( ) const  [inline], [override], [virtual]
```

Get system name

Implements Flow.

5.6.2.3 getSystemBegin()

```
template<typename Flow_IMPL >
System * FlowHandle< Flow_IMPL >::getSystemBegin ( ) const  [inline], [override], [virtual]
```

Get systemBegin

Implements Flow.

5.6.2.4 getSystemEnd()

```
template<typename Flow_IMPL >
System * FlowHandle< Flow_IMPL >::getSystemEnd ( ) const  [inline], [override], [virtual]
```

Get systemEnd

Implements Flow.

5.6.2.5 getValue()

```
template<typename Flow_IMPL >
double FlowHandle< Flow_IMPL >::getValue ( ) const  [inline], [override], [virtual]
```

Get system value

Implements Flow.

5.6.2.6 setName()

```
template<typename Flow_IMPL >
void FlowHandle< Flow_IMPL >::setName (
             std::string  )  [inline], [override], [virtual]
```

Set system name

Parameters

n	Name for the flow

Implements Flow.

5.6.2.7 setSystemBegin()

```
template<typename Flow_IMPL >
void FlowHandle< Flow_IMPL >::setSystemBegin (
             System * ) [inline], [override], [virtual]
```

Set systemBegin

Parameters

system	SystemBegin for the flow

Implements Flow.

5.6.2.8 setSystemEnd()

```
template<typename Flow_IMPL >
void FlowHandle< Flow_IMPL >::setSystemEnd (
             System * ) [inline], [override], [virtual]
```

Set systemBegin

Parameters

system	SystemEnd for the flow

Implements Flow.

5.6.2.9 setValue()

```
template<typename Flow_IMPL >
void FlowHandle< Flow_IMPL >::setValue (
             double ) [inline], [override], [virtual]
```

Set system value

Parameters

v	Value for the flow

Implements Flow.

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

- src/lib/FlowImplementation.h

5.7 Handle< T > Class Template Reference

The classes Handle and Body implements the "bridge" design pattern (also known as "handle/body idiom").

```
#include <handlebody.h>
```

Public Member Functions

- Handle ()
  constructor
- virtual ∼Handle ()
  Destructor.
- Handle (const Handle &hd)
  copy constructor
- Handle< T > & operator= (const Handle &hd)
  assignment operator

Protected Attributes

- T ∗ pImpl_
  referência para a implementação

5.7.1 Detailed Description

template<class T>
class Handle< T >

The classes Handle and Body implements the "bridge" design pattern (also known as "handle/body idiom").

5.7.2 Constructor & Destructor Documentation

5.7.2.1 Handle() [1/2]

```
template<class T >
Handle< T >::Handle ( )  [inline]
```

constructor

5.7.2.2 ∼Handle()

```
template<class T >
virtual Handle< T >::∼Handle ( )  [inline], [virtual]
```

Destructor.

5.7.2.3 Handle() [2/2]

```
template<class T >
Handle< T >::Handle (
            const Handle< T > & hd )  [inline]
```

copy constructor

5.7.3 Member Function Documentation

5.7.3.1 operator=()

```
template<class T >
Handle< T > & Handle< T >::operator= (
            const Handle< T > & hd )  [inline]
```

assignment operator

5.7.4 Member Data Documentation

5.7.4.1 pImpl_

```
template<class T >
T* Handle< T >::pImpl_  [protected]
```

referência para a implementação

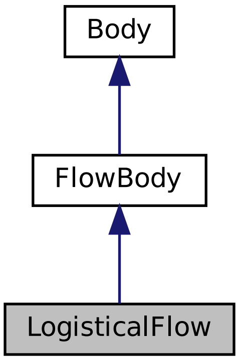
The documentation for this class was generated from the following file:

- src/lib/handlebody.h

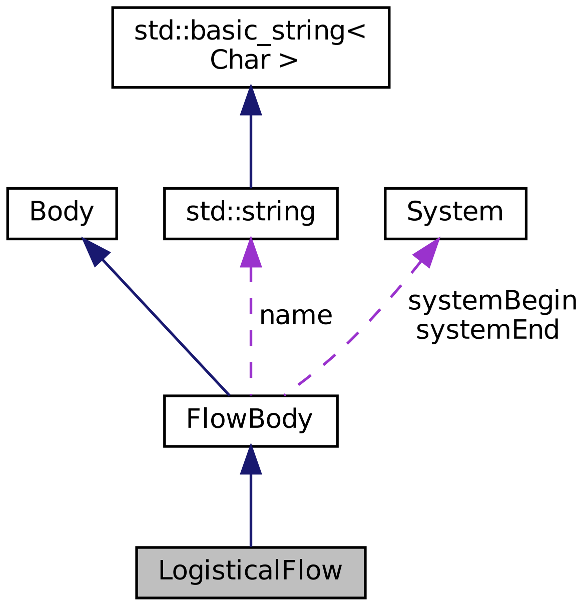
5.8 LogisticalFlow Class Reference

```
#include <FunctionalTests.h>
```

Inheritance diagram for LogisticalFlow:



Collaboration diagram for LogisticalFlow:



Public Member Functions

- LogisticalFlow (std::string name="", System ∗systemOut=NULL, System ∗systemIn=NULL)
- double expression () override

Additional Inherited Members

5.8.1 Detailed Description

Flow that converges energy from a model to another exponencialy with 1% of the end system per timestep times onde minus the end system divided by seventy

5.8.2 Constructor & Destructor Documentation

5.8.2.1 LogisticalFlow()

```
LogisticalFlow::LogisticalFlow (
             std::string name = "",
             System ∗ systemOut = NULL,
             System ∗ systemIn = NULL )  [inline]
```

Default constructor

Parameters

name	Inital flow name
value	Inital flow value
systemBegin	Inital system where the flow comes from
systemEnd	Inital system where the flow goes to

Returns

Logistical flow with initial name, value, systemBegin and systemEnd

5.8.3 Member Function Documentation

5.8.3.1 expression()

```
double LogisticalFlow::expression ( )  [inline], [override], [virtual]
```

Logistical expression

Implements FlowBody.

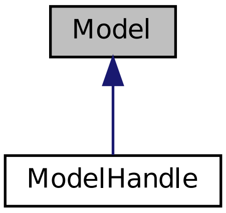
The documentation for this class was generated from the following file:

- test/functional/FunctionalTests.h

5.9 Model Class Reference

```
#include <Model.h>
```

Inheritance diagram for Model:



Public Member Functions

- virtual ∼Model ()=default
- virtual void simulate (double, double, double)=0
- virtual std::string getName () const =0
- virtual void setName (std::string)=0
- virtual double getTime () const =0
- virtual void setTime (double)=0
- virtual std::vector< System ∗ >::iterator getSystemsIterator ()=0
- virtual std::vector< Flow ∗ >::iterator getFlowsIterator ()=0
- virtual std::vector< Model ∗ >::iterator getModelsIterator ()=0
- virtual std::vector< System ∗ >::iterator endSystems ()=0
- virtual std::vector< Flow ∗ >::iterator endFlows ()=0
- virtual std::vector< Model ∗ >::iterator endModels ()=0
- virtual System ∗ createSystem (std::string name, double value)=0
- template<typename FlowType >
  Flow ∗ createFlow (std::string name, System ∗systemBegin, System ∗systemEnd)

Static Public Member Functions

- static Model ∗ createModel (std::string name, double time)

Protected Member Functions

- virtual void add (System ∗)=0
- virtual void add (Flow ∗)=0

5.9.1 Constructor & Destructor Documentation

5.9.1.1 ∼Model()

```
virtual Model::∼Model ( )  [virtual], [default]
```

Default destructor

5.9.2 Member Function Documentation

5.9.2.1 add() [1/2]

```
virtual void Model::add (
             Flow ∗  )  [protected], [pure virtual]
```

Add a flow to the model

Parameters

flow	Flow to be added to the model

Implemented in ModelHandle.

5.9.2.2 add() [2/2]

```
virtual void Model::add (
             System *  )  [protected], [pure virtual]
```

Add a flow to the model

Parameters

flow	Flow to be added to the model

Implemented in ModelHandle.

5.9.2.3 createFlow()

```
template<typename FlowType >
Flow * Model::createFlow (
             std::string name,
             System * systemBegin,
             System * systemEnd )  [inline]
```

5.9.2.4 createModel()

```
Model * Model::createModel (
             std::string name,
             double time )  [static]
```

5.9.2.5 createSystem()

```
virtual System * Model::createSystem (
             std::string name,
             double value )  [pure virtual]
```

Implemented in ModelHandle.

5.9.2.6 endFlows()

```
virtual std::vector< Flow * >::iterator Model::endFlows ( ) [pure virtual]
```

Implemented in ModelHandle.

5.9.2.7 endModels()

```
virtual std::vector< Model * >::iterator Model::endModels ( ) [pure virtual]
```

Implemented in ModelHandle.

5.9.2.8 endSystems()

```
virtual std::vector< System * >::iterator Model::endSystems ( ) [pure virtual]
```

Implemented in ModelHandle.

5.9.2.9 getFlowsIterator()

```
virtual std::vector< Flow * >::iterator Model::getFlowsIterator ( ) [pure virtual]
```

Get model flows iterator

Implemented in ModelHandle.

5.9.2.10 getModelsIterator()

```
virtual std::vector< Model * >::iterator Model::getModelsIterator ( ) [pure virtual]
```

Get model models iterator

Implemented in ModelHandle.

5.9.2.11 getName()

```
virtual std::string Model::getName ( ) const [pure virtual]
```

Get model name

Implemented in ModelHandle.

5.9.2.12 getSystemsIterator()

```
virtual std::vector< System * >::iterator Model::getSystemsIterator ( )  [pure virtual]
```

Get model systems iterator

Implemented in ModelHandle.

5.9.2.13 getTime()

```
virtual double Model::getTime ( ) const  [pure virtual]
```

Get model time

Implemented in ModelHandle.

5.9.2.14 setName()

```
virtual void Model::setName (
            std::string  )  [pure virtual]
```

Set model name

Parameters

n	Name for the system

Implemented in ModelHandle.

5.9.2.15 setTime()

```
virtual void Model::setTime (
            double  )  [pure virtual]
```

Set model time

Parameters

t	Name for the system

Implemented in ModelHandle.

5.9.2.16 simulate()

```
virtual void Model::simulate (
            double ,
            double ,
            double  )  [pure virtual]
```

Simulates the model during a period of time between start and end time values with a specified timestep

Parameters

start	Time where the simulation starts
end	Time where the simulation ends
timestep	Timestep value to simulate with

Implemented in ModelHandle.

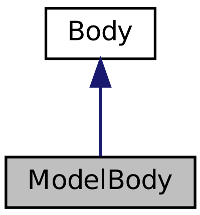
The documentation for this class was generated from the following files:

- src/lib/Model.h
- src/lib/ModelImplementation.cpp

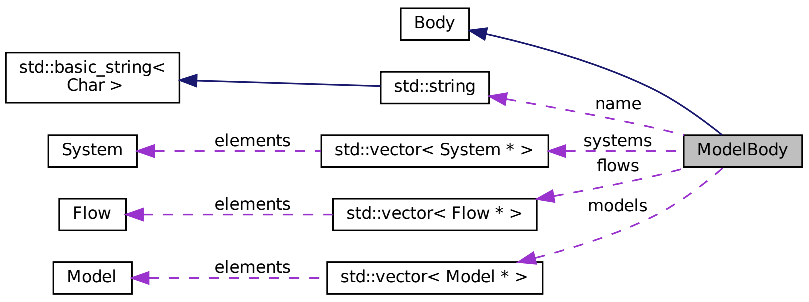
5.10 ModelBody Class Reference

```
#include <ModelImplementation.h>
```

Inheritance diagram for ModelBody:



Collaboration diagram for ModelBody:



Public Member Functions

- virtual ∼ModelBody ()
- ModelBody (std::string name="", double time=0.0)
- void simulate (double start, double end, double timestep)
- std::string getName () const
- void setName (std::string n)
- double getTime () const
- void setTime (double t)
- void add (System ∗system)
- void add (Flow ∗flow)
- std::vector< System ∗ >::iterator getSystemsIterator ()
- std::vector< Flow ∗ >::iterator getFlowsIterator ()
- std::vector< System ∗ >::iterator endSystems ()
- std::vector< Flow ∗ >::iterator endFlows ()
- System ∗ createSystem (std::string name, double value)

Static Public Member Functions

- static std::vector< Model ∗ >::iterator getModelsIterator ()
- static std::vector< Model ∗ >::iterator endModels ()
- static Model ∗ createModel (std::string name, double time)

Protected Attributes

- std::string name
- double time
- std::vector< System ∗ > systems
- std::vector< Flow ∗ > flows

Static Protected Attributes

- static std::vector< Model ∗ > models

5.10.1 Detailed Description

Model that simulates the energy flow through models

5.10.2 Constructor & Destructor Documentation

5.10.2.1 ∼ModelBody()

```
ModelBody::∼ModelBody ( )  [virtual]
```

Default destructor destrutor padrao

5.10.2.2 ModelBody()

```
ModelBody::ModelBody (
             std::string name = "",
             double time = 0.0 )
```

Default constructor

Parameters

name	Inital model name
time	Inital model time

Returns

Model with initial name and time

5.10.3 Member Function Documentation

5.10.3.1 add() [1/2]

```
void ModelBody::add (
             Flow * flow )
```

Add a flow to the model

Parameters

flow	Flow to be added to the model

5.10.3.2 add() [2/2]

```
void ModelBody::add (
             System * system )
```

Add a system to the model

Parameters

system	System to be added to the model

5.10.3.3 createModel()

```
Model * ModelBody::createModel (
             std::string name,
             double time ) [static]
```

5.10.3.4 createSystem()

```
System * ModelBody::createSystem (
             std::string name,
             double value )
```

5.10.3.5 endFlows()

```
std::vector< Flow * >::iterator ModelBody::endFlows ( )
```

5.10.3.6 endModels()

```
std::vector< Model * >::iterator ModelBody::endModels ( ) [static]
```

5.10.3.7 endSystems()

```
std::vector< System * >::iterator ModelBody::endSystems ( )
```

5.10.3.8 getFlowsIterator()

```
std::vector< Flow * >::iterator ModelBody::getFlowsIterator ( )
```

Get model flows iterator

5.10.3.9 getModelsIterator()

```
std::vector< Model * >::iterator ModelBody::getModelsIterator ( )  [static]
```

Get model models iterator

5.10.3.10 getName()

```
std::string ModelBody::getName ( ) const
```

Get model name

5.10.3.11 getSystemsIterator()

```
std::vector< System * >::iterator ModelBody::getSystemsIterator ( )
```

Get model systems iterator

5.10.3.12 getTime()

```
double ModelBody::getTime ( ) const
```

Get model time

5.10.3.13 setName()

```
void ModelBody::setName (
             std::string n )
```

Set model name

Parameters

n	Name for the system

5.10.3.14 setTime()

```
void ModelBody::setTime (
            double t )
```

Set model time

Parameters

t	Name for the system

5.10.3.15 simulate()

```
void ModelBody::simulate (
            double start,
            double end,
            double timestep )
```

Simulates the model during a period of time between start and end time values with a specified timestep

Parameters

start	Time where the simulation starts
end	Time where the simulation ends
timestep	Timestep value to simulate with

5.10.4 Member Data Documentation

5.10.4.1 flows

```
std::vector<Flow*> ModelBody::flows  [protected]
```

5.10.4.2 models

```
std::vector< Model * > ModelBody::models  [static], [protected]
```

5.10.4.3 name

```
std::string ModelBody::name  [protected]
```

5.10.4.4 systems

```
std::vector<System*> ModelBody::systems [protected]
```

5.10.4.5 time

```
double ModelBody::time [protected]
```

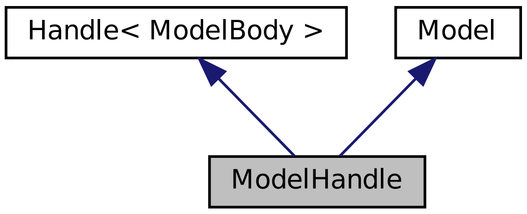
The documentation for this class was generated from the following files:

- src/lib/ModelImplementation.h
- src/lib/ModelImplementation.cpp

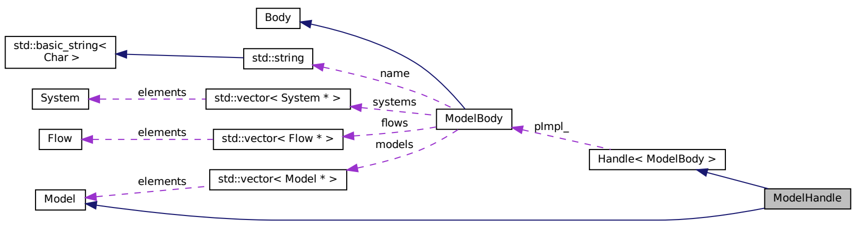
5.11 ModelHandle Class Reference

```
#include <ModelImplementation.h>
```

Inheritance diagram for ModelHandle:



Collaboration diagram for ModelHandle:



Public Member Functions

- ∼ModelHandle () override=default
- ModelHandle (std::string name="", double time=0.0)
- void simulate (double start, double end, double timestep) override
- System ∗ createSystem (std::string name, double value) override
- void setName (std::string n) override
- std::string getName () const override
- void setTime (double t) override
- double getTime () const override
- void add (System ∗system) override
- void add (Flow ∗flow) override
- std::vector< System ∗ >::iterator getSystemsIterator () override
- std::vector< Flow ∗ >::iterator getFlowsIterator () override
- std::vector< Model ∗ >::iterator getModelsIterator () override
- std::vector< System ∗ >::iterator endSystems () override
- std::vector< Flow ∗ >::iterator endFlows () override
- std::vector< Model ∗ >::iterator endModels () override

Static Public Member Functions

- static Model ∗ createModel (std::string name, double time)

Additional Inherited Members

5.11.1 Constructor & Destructor Documentation

5.11.1.1 ∼ModelHandle()

```
ModelHandle::∼ModelHandle ( )  [override], [default]
```

5.11.1.2 ModelHandle()

```
ModelHandle::ModelHandle (
             std::string name = "",
             double time = 0.0 )  [inline]
```

5.11.2 Member Function Documentation

5.11.2.1 add() [1/2]

```
void ModelHandle::add (
             Flow ∗  )  [inline], [override], [virtual]
```

Add a flow to the model

Parameters

flow	Flow to be added to the model

Implements Model.

5.11.2.2 add() [2/2]

```
void ModelHandle::add (
             System * ) [inline], [override], [virtual]
```

Add a flow to the model

Parameters

flow	Flow to be added to the model

Implements Model.

5.11.2.3 createModel()

```
static Model * ModelHandle::createModel (
             std::string name,
             double time ) [inline], [static]
```

5.11.2.4 createSystem()

```
System * ModelHandle::createSystem (
             std::string name,
             double value ) [inline], [override], [virtual]
```

Implements Model.

5.11.2.5 endFlows()

```
std::vector< Flow * >::iterator ModelHandle::endFlows ( ) [inline], [override], [virtual]
```

Implements Model.

5.11.2.6 endModels()

```
std::vector< Model * >::iterator ModelHandle::endModels ( )  [inline], [override], [virtual]
```

Implements Model.

5.11.2.7 endSystems()

```
std::vector< System * >::iterator ModelHandle::endSystems ( )  [inline], [override], [virtual]
```

Implements Model.

5.11.2.8 getFlowsIterator()

```
std::vector< Flow * >::iterator ModelHandle::getFlowsIterator ( )  [inline], [override], [virtual]
```

Get model flows iterator

Implements Model.

5.11.2.9 getModelsIterator()

```
std::vector< Model * >::iterator ModelHandle::getModelsIterator ( )  [inline], [override],
[virtual]
```

Get model models iterator

Implements Model.

5.11.2.10 getName()

```
std::string ModelHandle::getName ( ) const  [inline], [override], [virtual]
```

Get model name

Implements Model.

5.11.2.11 getSystemsIterator()

```
std::vector< System * >::iterator ModelHandle::getSystemsIterator ( )  [inline], [override],
[virtual]
```

Get model systems iterator

Implements Model.

5.11.2.12 getTime()

```
double ModelHandle::getTime ( ) const  [inline], [override], [virtual]
```

Get model time

Implements Model.

5.11.2.13 setName()

```
void ModelHandle::setName (
             std::string  )  [inline], [override], [virtual]
```

Set model name

Parameters

n	Name for the system

Implements Model.

5.11.2.14 setTime()

```
void ModelHandle::setTime (
             double  )  [inline], [override], [virtual]
```

Set model time

Parameters

t	Name for the system

Implements Model.

5.11.2.15 simulate()

```
void ModelHandle::simulate (
             double ,
             double ,
             double  )  [inline], [override], [virtual]
```

Simulates the model during a period of time between start and end time values with a specified timestep

Parameters

start	Time where the simulation starts
end	Time where the simulation ends
timestep	Timestep value to simulate with

Implements Model.

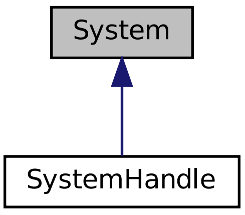
The documentation for this class was generated from the following file:

- src/lib/ModelImplementation.h

5.12 System Class Reference

```
#include <System.h>
```

Inheritance diagram for System:



Public Member Functions

- virtual ∼System ()=default
- virtual std::string getName () const =0
- virtual void setName (std::string)=0
- virtual double getValue () const =0
- virtual void setValue (double)=0

5.12.1 Constructor & Destructor Documentation

5.12.1.1 ∼System()

```
virtual System::∼System ( )  [virtual], [default]
```

Default destructor

5.12.2 Member Function Documentation

5.12.2.1 getName()

```
virtual std::string System::getName ( ) const  [pure virtual]
```

Get system name

Implemented in SystemHandle.

5.12.2.2 getValue()

```
virtual double System::getValue ( ) const  [pure virtual]
```

Get system value

Implemented in SystemHandle.

5.12.2.3 setName()

```
virtual void System::setName (
             std::string  )  [pure virtual]
```

Set system name

Parameters

n	Name for the system

Implemented in SystemHandle.

5.12.2.4 setValue()

```
virtual void System::setValue (
             double   )  [pure virtual]
```

Set system value

Parameters

v	Value for the system

Implemented in SystemHandle.

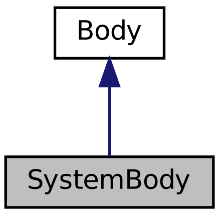
The documentation for this class was generated from the following file:

- src/lib/System.h

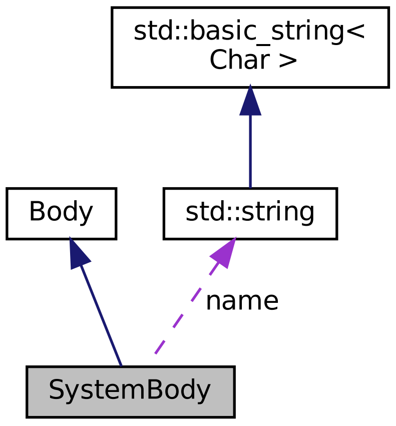
5.13 SystemBody Class Reference

```
#include <SystemImplementation.h>
```

Inheritance diagram for SystemBody:



Collaboration diagram for SystemBody:



Public Member Functions

- virtual ∼SystemBody () override
- SystemBody (std::string name="", double value=0.0)
- SystemBody (const SystemBody &system)
- SystemBody & operator= (const SystemBody &system)
- std::string getName () const
- void setName (std::string n)
- double getValue () const
- void setValue (double v)

Protected Attributes

- std::string name
- double value

5.13.1 Detailed Description

System that stores energy

5.13.2 Constructor & Destructor Documentation

5.13.2.1 ∼SystemBody()

```
SystemBody::∼SystemBody ( )  [override], [virtual], [default]
```

Default destructor

5.13.2.2 SystemBody() [1/2]

```
SystemBody::SystemBody (
            std::string name = "",
            double value = 0.0 )
```

Default constructor

Parameters

name	Inital system name
value	Inital system value

Returns

System with initial name and value

5.13.2.3 SystemBody() [2/2]

```
SystemBody::SystemBody (
            const SystemBody & system )
```

Copy constructor

Parameters

system	System to copy from

Returns

Copied system

5.13.3 Member Function Documentation

5.13.3.1 getName()

```
std::string SystemBody::getName ( ) const
```

Get system name

5.13.3.2 getValue()

```
double SystemBody::getValue ( ) const
```

Get system value

5.13.3.3 operator=()

```
SystemBody & SystemBody::operator= (
              const SystemBody & system )
```

Copy Assignment Operator

Parameters

system	System to copy from

Returns

Copied system

5.13.3.4 setName()

```
void SystemBody::setName (
              std::string n )
```

Set system name

Parameters

n	Name for the system

5.13.3.5 setValue()

```
void SystemBody::setValue (
              double v )
```

Set system value

Parameters

v	Value for the system

5.13.4 Member Data Documentation

5.13.4.1 name

```
std::string SystemBody::name  [protected]
```

5.13.4.2 value

```
double SystemBody::value  [protected]
```

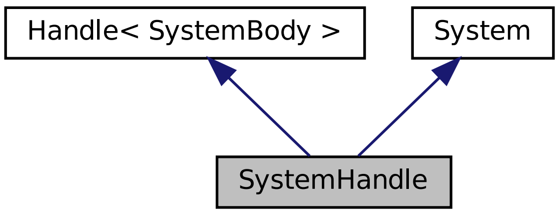
The documentation for this class was generated from the following files:

- src/lib/SystemImplementation.h
- src/lib/SystemImplementation.cpp

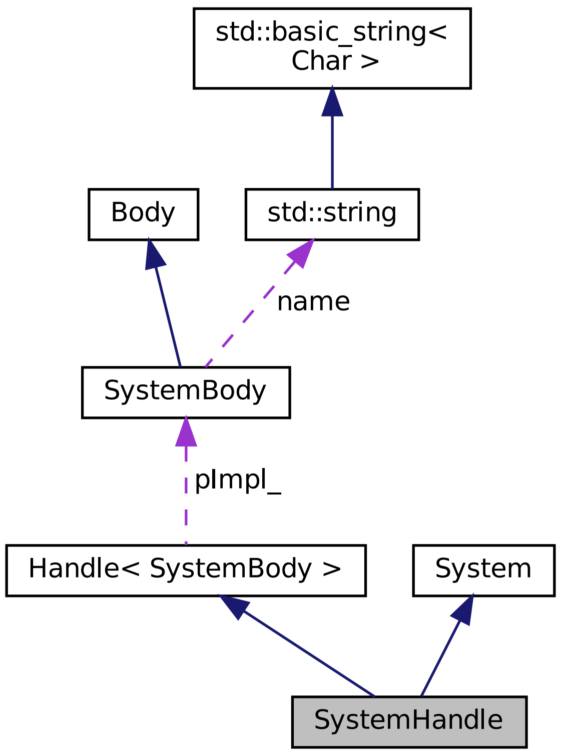
5.14 SystemHandle Class Reference

```
#include <SystemImplementation.h>
```

Inheritance diagram for SystemHandle:



Collaboration diagram for SystemHandle:



Public Member Functions

- ∼SystemHandle () override=default
- SystemHandle (std::string name="", double value=0.0)
- std::string getName () const override
- void setName (std::string sysName) override
- double getValue () const override
- void setValue (double sysValue) override

Additional Inherited Members

5.14.1 Constructor & Destructor Documentation

5.14.1.1 ∼SystemHandle()

```
SystemHandle::∼SystemHandle ( )  [override], [default]
```

5.14.1.2 SystemHandle()

```
SystemHandle::SystemHandle (
             std::string name = "",
             double value = 0.0 )  [inline]
```

5.14.2 Member Function Documentation

5.14.2.1 getName()

```
std::string SystemHandle::getName ( ) const  [inline], [override], [virtual]
```

Get system name

Implements System.

5.14.2.2 getValue()

```
double SystemHandle::getValue ( ) const  [inline], [override], [virtual]
```

Get system value

Implements System.

5.14.2.3 setName()

```
void SystemHandle::setName (
             std::string  )  [inline], [override], [virtual]
```

Set system name

Parameters

n	Name for the system

Implements System.

5.14.2.4 setValue()

```
void SystemHandle::setValue (
             double  )  [inline], [override], [virtual]
```

Set system value

Parameters

v	Value for the system

Implements System.

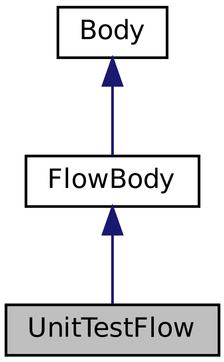
The documentation for this class was generated from the following file:

- src/lib/SystemImplementation.h

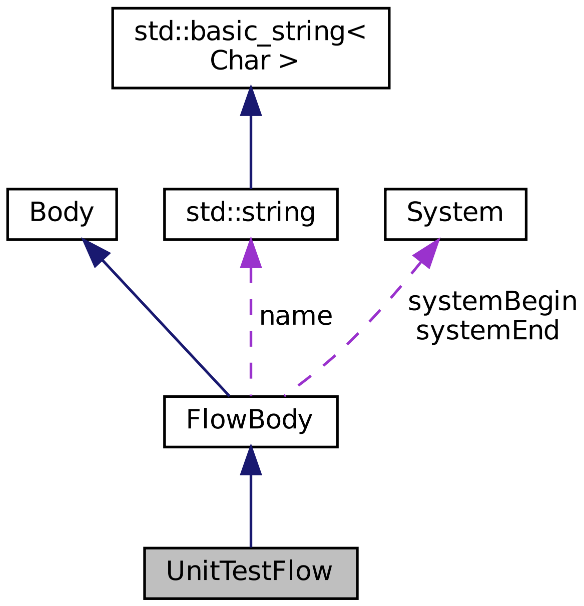
5.15 UnitTestFlow Class Reference

```
#include <unitFlow.h>
```

Inheritance diagram for UnitTestFlow:



Collaboration diagram for UnitTestFlow:



Public Member Functions

- UnitTestFlow (std::string name="", System ∗systemBegin=NULL, System ∗systemEnd=NULL)
- double expression () override

Additional Inherited Members

5.15.1 Detailed Description

Flow used for testing

5.15.2 Constructor & Destructor Documentation

5.15.2.1 UnitTestFlow()

```
UnitTestFlow::UnitTestFlow (
            std::string name = "",
            System ∗ systemBegin = NULL,
            System ∗ systemEnd = NULL )  [inline]
```

Default constructor

Parameters

name	Inital flow name
value	Inital flow value
systemBegin	Inital system where the flow comes from
systemEnd	Inital system where the flow goes to

Returns

UnitTestFlow with initial name, value, systemBegin and systemEnd

5.15.3 Member Function Documentation

5.15.3.1 expression()

```
double UnitTestFlow::expression ( )  [inline], [override], [virtual]
```

Flow expression method implementation for testing

Implements FlowBody.

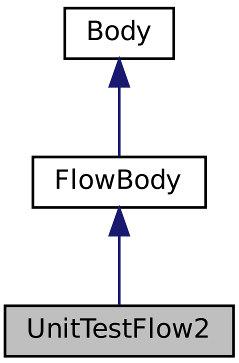
The documentation for this class was generated from the following file:

- test/unit/unitFlow.h

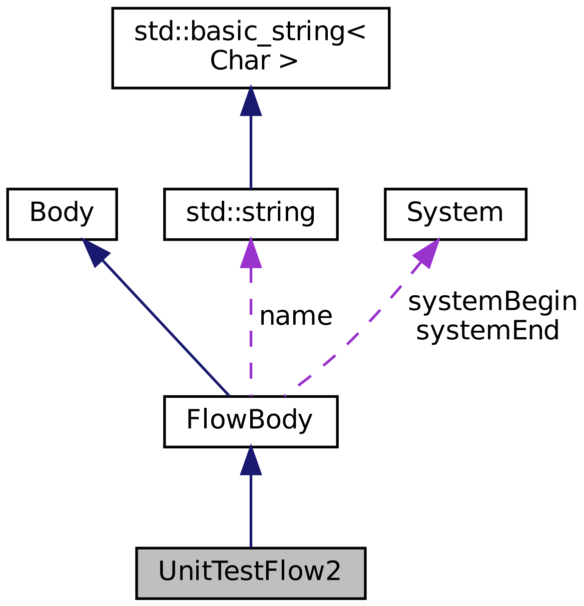
5.16 UnitTestFlow2 Class Reference

```
#include <unitModel.h>
```

Inheritance diagram for UnitTestFlow2:



Collaboration diagram for UnitTestFlow2:



Public Member Functions

- UnitTestFlow2 (std::string name="", System ∗systemBegin=NULL, System ∗systemEnd=NULL)
- double expression () override

Additional Inherited Members

5.16.1 Detailed Description

Flow used for testing

5.16.2 Constructor & Destructor Documentation

5.16.2.1 UnitTestFlow2()

```
UnitTestFlow2::UnitTestFlow2 (
              std::string name = "",
              System * systemBegin = NULL,
              System * systemEnd = NULL )  [inline]
```

Default constructor

Parameters

name	Inital flow name
value	Inital flow value
systemBegin	Inital system where the flow comes from
systemEnd	Inital system where the flow goes to

Returns

UnitTestFlow2 with initial name, value, systemBegin and systemEnd

5.16.3 Member Function Documentation

5.16.3.1 expression()

```
double UnitTestFlow2::expression ( )  [inline], [override], [virtual]
```

Flow expression method implementation for testing

Implements FlowBody.

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

- test/unit/unitModel.h

Chapter 6

File Documentation

6.1 cmake-build-debug/CMakeCache.txt File Reference

6.2 cmake-build-debug/CMakeFiles/3.22.3/CompilerIdC/CMake↩ CCompilerId.c File Reference

Macros

- #define __has_include(x) 0
- #define COMPILER_ID ""
- #define STRINGIFY_HELPER(X) #X
- #define STRINGIFY(X) STRINGIFY_HELPER(X)
- #define PLATFORM_ID
- #define ARCHITECTURE_ID
- #define DEC(n)
- #define HEX(n)
- #define C_VERSION

Functions

- int main (int argc, char ∗argv[ ])

Variables

- char const ∗ info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
- char const ∗ info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
- char const ∗ info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
- const char ∗ info_language_standard_default
- const char ∗ info_language_extensions_default

6.2.1 Macro Definition Documentation

6.2.1.1 __has_include

```
#define __has_include(
              x ) 0
```

6.2.1.2 ARCHITECTURE_ID

```
#define ARCHITECTURE_ID
```

6.2.1.3 C_VERSION

```
#define C_VERSION
```

6.2.1.4 COMPILER_ID

```
#define COMPILER_ID ""
```

6.2.1.5 DEC

```
#define DEC(
              n )
```

Value:

```
  ('0' + (((n) / 10000000)%10)), \
  ('0' + (((n) / 1000000)%10)),  \
  ('0' + (((n) / 100000)%10)),   \
  ('0' + (((n) / 10000)%10)),    \
  ('0' + (((n) / 1000)%10)),     \
  ('0' + (((n) / 100)%10)),      \
  ('0' + (((n) / 10)%10)),       \
  ('0' +  ((n) % 10))
```

6.2.1.6 HEX

```
#define HEX(
              n )
```

Value:

```
  ('0' + ((n)»28 & 0xF)), \
  ('0' + ((n)»24 & 0xF)), \
  ('0' + ((n)»20 & 0xF)), \
  ('0' + ((n)»16 & 0xF)), \
  ('0' + ((n)»12 & 0xF)), \
  ('0' + ((n)»8  & 0xF)), \
  ('0' + ((n)»4  & 0xF)), \
  ('0' + ((n)     & 0xF))
```

6.2.1.7 PLATFORM_ID

```
#define PLATFORM_ID
```

6.2.1.8 STRINGIFY

```
#define STRINGIFY(
              X ) STRINGIFY_HELPER(X)
```

6.2.1.9 STRINGIFY_HELPER

```
#define STRINGIFY_HELPER(
              X ) #X
```

6.2.2 Function Documentation

6.2.2.1 main()

```
int main (
             int argc,
             char * argv[] )
```

6.2.3 Variable Documentation

6.2.3.1 info_arch

```
char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
```

6.2.3.2 info_compiler

```
char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
```

6.2.3.3 info_language_extensions_default

```
const char* info_language_extensions_default
```

Initial value:
```
= "INFO" ":" "extensions_default["
  "OFF"
"]"
```

6.2.3.4 info_language_standard_default

```
const char* info_language_standard_default
```

Initial value:
```
=
  "INFO" ":" "standard_default[" C_VERSION "]"
```

6.2.3.5 info_platform

```
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
```

6.3 cmake-build-debug/CMakeFiles/3.22.3/CompilerIdCXX/CMake↩ CXXCompilerId.cpp File Reference

Macros

- #define __has_include(x) 0
- #define COMPILER_ID ""
- #define STRINGIFY_HELPER(X) #X
- #define STRINGIFY(X) STRINGIFY_HELPER(X)
- #define PLATFORM_ID
- #define ARCHITECTURE_ID
- #define DEC(n)
- #define HEX(n)
- #define CXX_STD __cplusplus

Functions

- int main (int argc, char ∗argv[ ])

Variables

- char const ∗ info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
- char const ∗ info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
- char const ∗ info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
- const char ∗ info_language_standard_default
- const char ∗ info_language_extensions_default

6.3.1 Macro Definition Documentation

6.3.1.1 __has_include

```
#define __has_include(
              x ) 0
```

6.3.1.2 ARCHITECTURE_ID

```
#define ARCHITECTURE_ID
```

6.3.1.3 COMPILER_ID

```
#define COMPILER_ID ""
```

6.3.1.4 CXX_STD

```
#define CXX_STD __cplusplus
```

6.3.1.5 DEC

```
#define DEC(
              n )
```

Value:

```
  ('0' + (((n) / 10000000)%10)), \
  ('0' + (((n) / 1000000)%10)),  \
  ('0' + (((n) / 100000)%10)),   \
  ('0' + (((n) / 10000)%10)),    \
  ('0' + (((n) / 1000)%10)),     \
  ('0' + (((n) / 100)%10)),      \
  ('0' + (((n) / 10)%10)),       \
  ('0' +  ((n) % 10))
```

6.3.1.6 HEX

```
#define HEX(
             n )
```

Value:

```
  ('0' + ((n)»28 & 0xF)), \
  ('0' + ((n)»24 & 0xF)), \
  ('0' + ((n)»20 & 0xF)), \
  ('0' + ((n)»16 & 0xF)), \
  ('0' + ((n)»12 & 0xF)), \
  ('0' + ((n)»8  & 0xF)), \
  ('0' + ((n)»4  & 0xF)), \
  ('0' + ((n)     & 0xF))
```

6.3.1.7 PLATFORM_ID

```
#define PLATFORM_ID
```

6.3.1.8 STRINGIFY

```
#define STRINGIFY(
             X ) STRINGIFY_HELPER(X)
```

6.3.1.9 STRINGIFY_HELPER

```
#define STRINGIFY_HELPER(
             X ) #X
```

6.3.2 Function Documentation

6.3.2.1 main()

```
int main (
            int argc,
            char * argv[] )
```

6.3.3 Variable Documentation

6.3.3.1 info_arch

```
char const* info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"
```

6.3.3.2 info_compiler

```
char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
```

6.3.3.3 info_language_extensions_default

```
const char* info_language_extensions_default
```

Initial value:

```
= "INFO" ":" "extensions_default["
  "OFF"
"]"
```

6.3.3.4 info_language_standard_default

```
const char* info_language_standard_default
```

Initial value:

```
= "INFO" ":" "standard_default["
  "98"
"]"
```

6.3.3.5 info_platform

```
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
```

6.4 cmake-build-debug/CMakeFiles/clion-environment.txt File Reference

6.5 cmake-build-debug/CMakeFiles/clion-log.txt File Reference

6.6 cmake-build-debug/CMakeFiles/TargetDirectories.txt File Reference

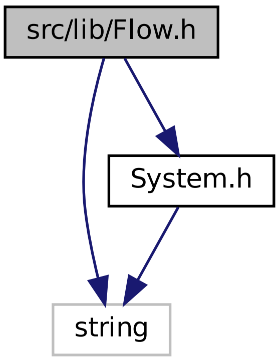
6.7 CMakeLists.txt File Reference

6.8 README.md File Reference

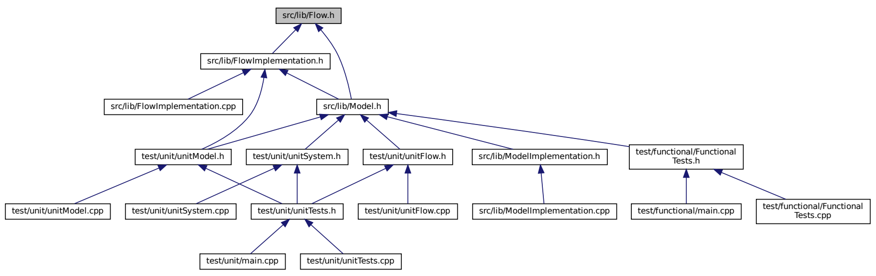
6.9 src/lib/Flow.h File Reference

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

Include dependency graph for Flow.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Flow

6.10 Flow.h

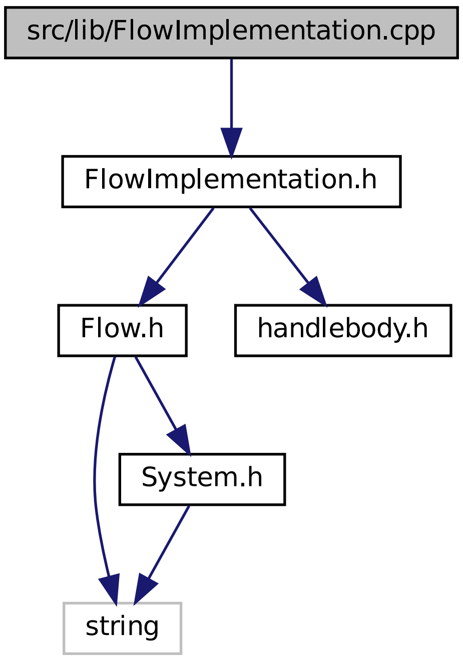
Go to the documentation of this file.

```
//
// Created by joaozenobio on 27/04/2022.
//

#ifndef ENG1_FLOW_H
#define ENG1_FLOW_H

#include <string>

#include "System.h"

class Flow {
public:
    virtual ~Flow() = default;
    virtual std::string getName() const = 0;
    virtual void setName(std::string) = 0;
    virtual double getValue() const = 0;
    virtual void setValue(double) = 0;
    virtual double expression() = 0;
    virtual System* getSystemBegin() const = 0;
    virtual void setSystemBegin(System*) = 0;
    virtual System* getSystemEnd() const = 0;
    virtual void setSystemEnd(System*) = 0;
};


#endif //ENG1_FLOW_H
```

6.11 src/lib/FlowImplementation.cpp File Reference

```
#include "FlowImplementation.h"
```

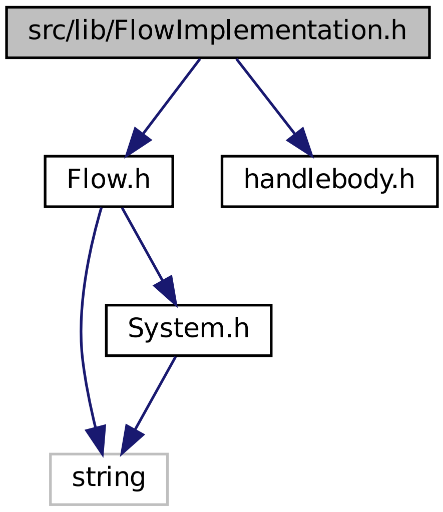
Include dependency graph for FlowImplementation.cpp:



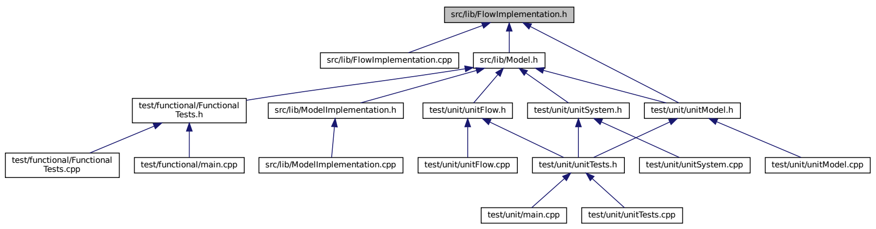
6.12 src/lib/FlowImplementation.h File Reference

```
#include "Flow.h"
#include "handlebody.h"
```

Include dependency graph for FlowImplementation.h:



This graph shows which files directly or indirectly include this file:



Classes

- class FlowBody
- class FlowHandle< Flow_IMPL >

6.13 FlowImplementation.h

Go to the documentation of this file.

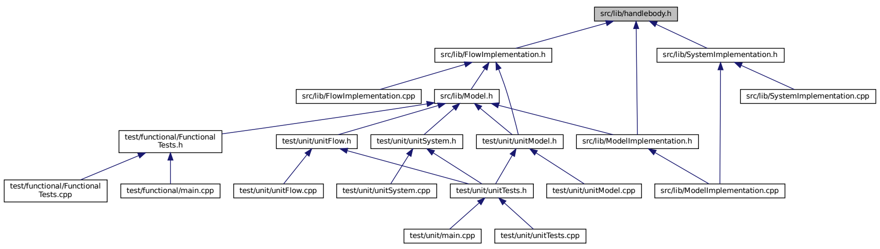
```
//
// Created by joaozenobio on 28/04/2022.
//

#ifndef ENG1_FLOWIMPLEMENTATION_H
#define ENG1_FLOWIMPLEMENTATION_H


#include "Flow.h"
```

```
#include "handlebody.h"

class FlowBody : public Body {
protected:
    std::string name;
    double value;
    System* systemBegin;
    System* systemEnd;

public:
    virtual ~FlowBody();

    FlowBody(std::string name="", System* systemBegin=NULL, System* systemEnd=NULL);

    FlowBody(const FlowBody& flow);

    FlowBody& operator=(const FlowBody& flow);

    virtual double expression() = 0;

    std::string getName() const;

    void setName(std::string n);

    double getValue() const;

    void setValue(double v);

    System* getSystemBegin() const;

    void setSystemBegin(System* system);

    System* getSystemEnd() const;

    void setSystemEnd(System* system);
};

template <typename Flow_IMPL>
class FlowHandle : public Handle<Flow_IMPL>, public Flow {
public:

    ~FlowHandle() override = default;

    FlowHandle<Flow_IMPL>(std::string name="", System* systemBegin=NULL, System* systemEnd=NULL){
        this->pImpl_->setName(name);
        this->pImpl_->setSystemBegin(systemBegin);
        this->pImpl_->setSystemEnd(systemEnd);
    }

    double expression() override {
        return this->pImpl_->expression();
    }

    std::string getName() const override {
        return this->pImpl_->getName();
    }

    void setName(std::string name) override {
        this->pImpl_->setName(name);
    }

    double getValue() const override {
        return this->pImpl_->getValue();
    }

    void setValue(double v) override {
        this->pImpl_->setValue(v);
    }

    System* getSystemBegin() const override {
        return this->pImpl_->getSystemBegin();
    }

    void setSystemBegin(System* systemBegin) override {
        this->pImpl_->setSystemBegin(systemBegin);
    }

    System* getSystemEnd() const override {
        return this->pImpl_->getSystemEnd();
    }

    void setSystemEnd(System* systemEnd) override {
        this->pImpl_->setSystemEnd(systemEnd);
    }
};


```

```
#endif //ENG1_FLOWIMPLEMENTATION_H
```

6.14 src/lib/handlebody.h File Reference

This graph shows which files directly or indirectly include this file:



Classes

- class Handle< T >

  The classes Handle and Body implements the "bridge" design pattern (also known as "handle/body idiom").
- class Body

  The class Implementation was implemented based on the class teCounted writed by Ricardo Cartaxo and Gilberto Câmara and founded in the geographic library TerraLib.

Macros

- #define DEBUGING

Variables

- int numHandleCreated
- int numHandleDeleted
- int numBodyCreated
- int numBodyDeleted

6.14.1 Macro Definition Documentation

6.14.1.1 DEBUGING

```
#define DEBUGING
```

6.14.2 Variable Documentation

6.14.2.1 numBodyCreated

```
int numBodyCreated  [extern]
```

6.14.2.2 numBodyDeleted

```
int numBodyDeleted  [extern]
```

6.14.2.3 numHandleCreated

```
int numHandleCreated  [extern]
```

6.14.2.4 numHandleDeleted

```
int numHandleDeleted  [extern]
```

6.15 handlebody.h

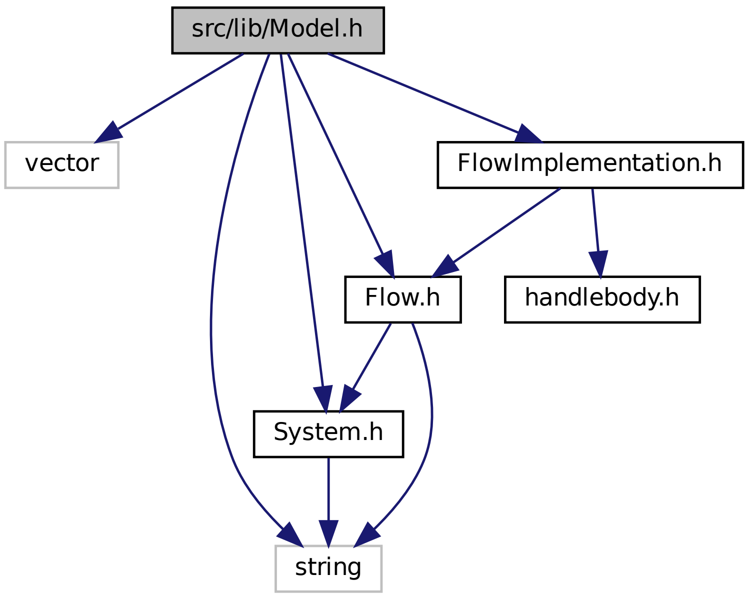
Go to the documentation of this file.

```

#if !defined(HANDLE_BODY)
#define HANDLE_BODY

#define DEBUGING
#ifdef DEBUGING
extern int numHandleCreated;
extern int numHandleDeleted;
extern int numBodyCreated;
extern int numBodyDeleted;
#endif

template <class T>
class Handle
{

public:

    Handle<T>( ){
        pImpl_ = new T;
        pImpl_->attach();

#ifdef DEBUGING
        numHandleCreated++;
#endif
    }

    virtual ~Handle<T>(){
        pImpl_->detach();

#ifdef DEBUGING
        numHandleDeleted++;
#endif
    }

    Handle<T>( const Handle& hd ):pImpl_( hd.pImpl_ ) { pImpl_->attach();  }

    Handle<T>& operator=( const Handle& hd) {
        if (  this != &hd )
        {
            hd.pImpl_->attach();
            pImpl_->detach();
            pImpl_ = hd.pImpl_;
        }
        return *this;
    }
protected:

    T *pImpl_;
};

class Body
{
public:
    Body(): refCount_(0){
#ifdef DEBUGING
        numBodyCreated++;
#endif
    }

    void attach ()    { refCount_++; }

    void detach (){
        if ( --refCount_ == 0 ) {
            delete this;
        }
    }

    int refCount(){ return refCount_; }

    virtual ~Body(){
#ifdef DEBUGING
        numBodyDeleted++;
#endif
    }

private:

    Body(const Body&);

    Body& operator=(const Body&){return *this;}

    int refCount_;

};

#endif
```

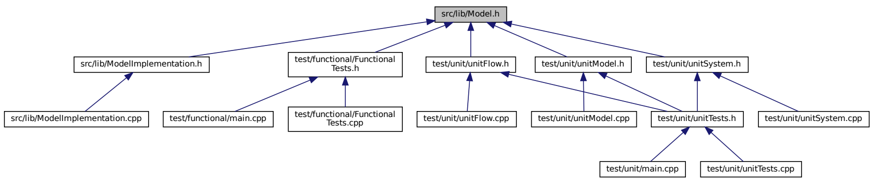
6.16 src/lib/Model.h File Reference

```
#include <vector>
#include <string>
#include "System.h"
#include "Flow.h"
#include "FlowImplementation.h"
```

Include dependency graph for Model.h:



This graph shows which files directly or indirectly include this file:



Classes

- class Model

6.17 Model.h

Go to the documentation of this file.

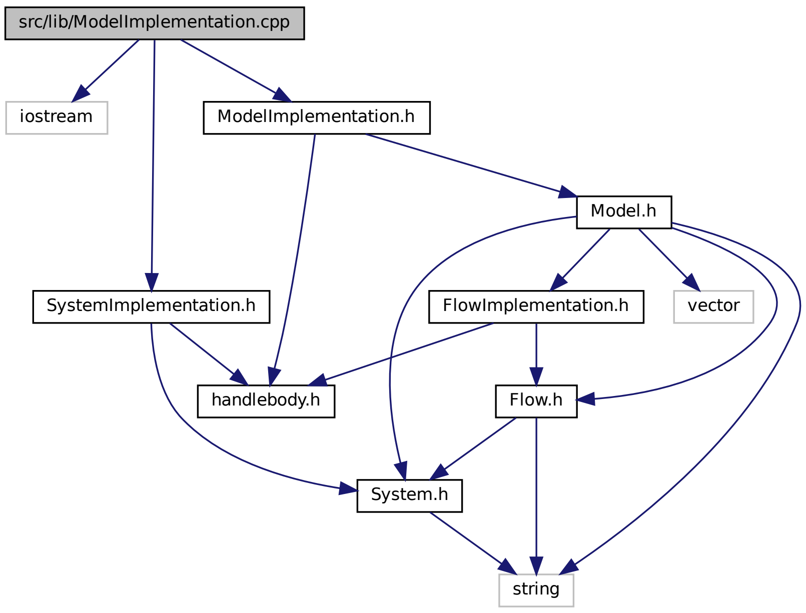
```
//
// Created by joaozenobio on 27/04/2022.
//

#ifndef ENG1_MODEL_H
#define ENG1_MODEL_H


#include <vector>
#include <string>
```

```

#include "System.h"
#include "Flow.h"
#include "FlowImplementation.h"

class Model {
protected:
    virtual void add(System*) = 0;
    virtual void add(Flow*) = 0;
public:
    virtual ~Model() = default;
    virtual void simulate(double, double, double) = 0;
    virtual std::string getName() const = 0;
    virtual void setName(std::string) = 0;
    virtual double getTime() const = 0;
    virtual void setTime(double) = 0;
    virtual std::vector<System*>::iterator getSystemsIterator() = 0;
    virtual std::vector<Flow*>::iterator getFlowsIterator() = 0;
    virtual std::vector<Model*>::iterator getModelsIterator() = 0;

    virtual std::vector<System*>::iterator endSystems() = 0;

    virtual std::vector<Flow*>::iterator endFlows() = 0;

    virtual std::vector<Model*>::iterator endModels() = 0;

    virtual System* createSystem(std::string name, double value) = 0;

    template<typename FlowType>
    Flow* createFlow(std::string name, System* systemBegin, System* systemEnd){
        Flow* flow = new FlowHandle<FlowType>(name, systemBegin, systemEnd);
        add(flow);
        return flow;
    }

    static Model* createModel(std::string name, double time);
};


#endif //ENG1_MODEL_H
```

6.18 src/lib/ModelImplementation.cpp File Reference

```
#include <iostream>
#include "ModelImplementation.h"
#include "SystemImplementation.h"
```

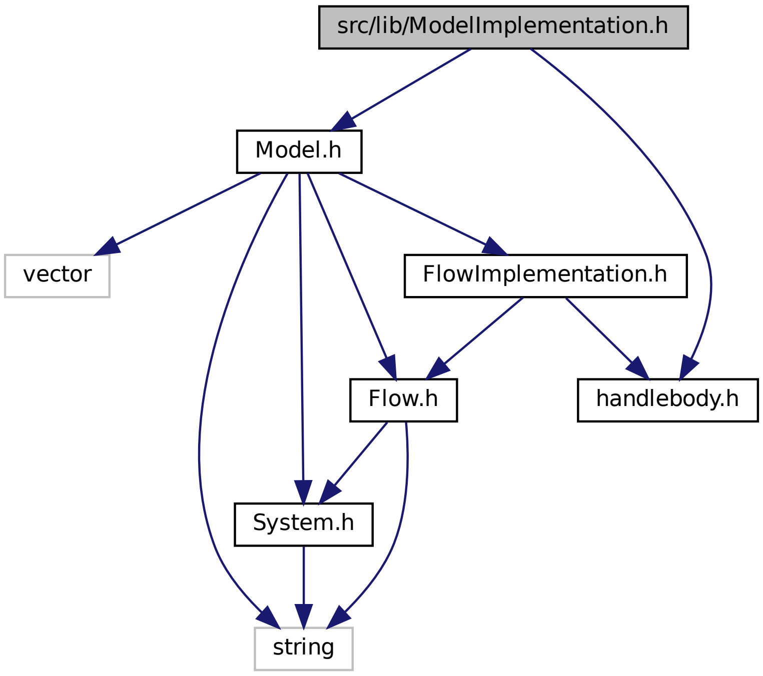
Include dependency graph for ModelImplementation.cpp:



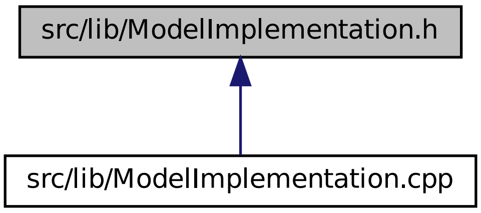
6.19 src/lib/ModelImplementation.h File Reference

```
#include "Model.h"
#include "handlebody.h"
```

Include dependency graph for ModelImplementation.h:



This graph shows which files directly or indirectly include this file:


src/lib/ModelImplementation.h
src/lib/ModelImplementation.cpp

Classes

- class ModelBody
- class ModelHandle

6.20 ModelImplementation.h

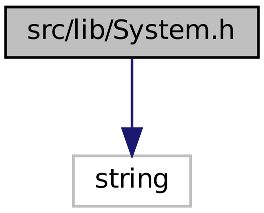
Go to the documentation of this file.

```
//
// Created by joaozenobio on 28/04/2022.
//

#ifndef ENG1_MODELIMPLEMENTATION_H
#define ENG1_MODELIMPLEMENTATION_H


#include "Model.h"
#include "handlebody.h"

class ModelBody: public Body {
private:
    ModelBody(const ModelBody& model);

    ModelBody& operator=(const ModelBody& model);

protected:
    std::string name;
    double time;
    std::vector<System*> systems;
    std::vector<Flow*> flows;
    static std::vector<Model*> models;

public:
    virtual ~ModelBody();

    ModelBody(std::string name="", double time=0.0);

    void simulate(double start, double end, double timestep);

    std::string getName() const;

    void setName(std::string n);

    double getTime() const;

    void setTime(double t);

    void add(System* system);

    void add(Flow* flow);

    std::vector<System*>::iterator getSystemsIterator();

    std::vector<Flow*>::iterator getFlowsIterator();

    static std::vector<Model*>::iterator getModelsIterator();

    std::vector<System*>::iterator endSystems();

    std::vector<Flow*>::iterator endFlows();

    static std::vector<Model*>::iterator endModels();

    System* createSystem(std::string name, double value);

    static Model* createModel(std::string name, double time);
};

class ModelHandle : public Handle<ModelBody>, public Model{
public:

    ~ModelHandle() override = default;

    ModelHandle(std::string name="", double time=0.0){
        pImpl_->setName(name);
        pImpl_->setTime(time);
    }

    void simulate(double start, double end, double timestep) override {
        pImpl_->simulate(start, end, timestep);
    }

    System* createSystem(std::string name, double value) override{
        return pImpl_->createSystem(name, value);
    }

    static Model* createModel(std::string name, double time){
        return ModelBody::createModel(name, time);
    }

    void setName(std::string n) override {
        pImpl_->setName(n);
    }

    std::string getName() const override {
        return pImpl_->getName();
    }

    void setTime(double t) override {
        pImpl_->setTime(t);
    }

    double getTime() const override {
        return pImpl_->getTime();
    }

    void add(System* system) override {
        return pImpl_->add(system);
    }

    void add(Flow* flow) override {
        return pImpl_->add(flow);
    }

    std::vector<System*>::iterator getSystemsIterator() override {
        return pImpl_->getSystemsIterator();
    }

    std::vector<Flow*>::iterator getFlowsIterator() override {
        return pImpl_->getFlowsIterator();
    }

    std::vector<Model*>::iterator getModelsIterator() override {
        return ModelBody::getModelsIterator();
    }

    std::vector<System*>::iterator endSystems() override {
        return pImpl_->endSystems();
    };

    std::vector<Flow*>::iterator endFlows() override {
        return pImpl_->endFlows();
    };

    std::vector<Model*>::iterator endModels() override {
        return ModelBody::endModels();
    };
};

#endif //ENG1_MODELIMPLEMENTATION_H
```

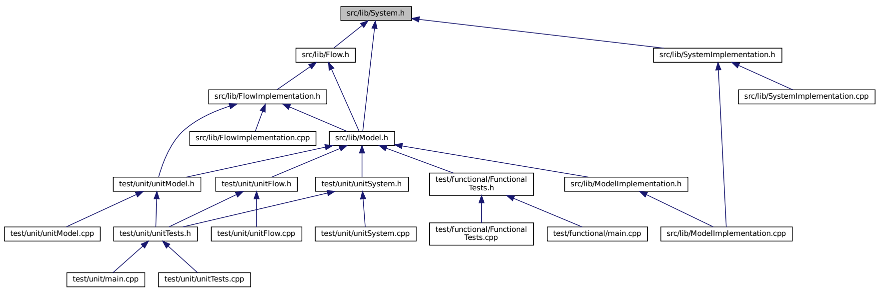
6.21 src/lib/System.h File Reference

`#include <string>`

Include dependency graph for System.h:



This graph shows which files directly or indirectly include this file:



Classes

- class System

6.22 System.h

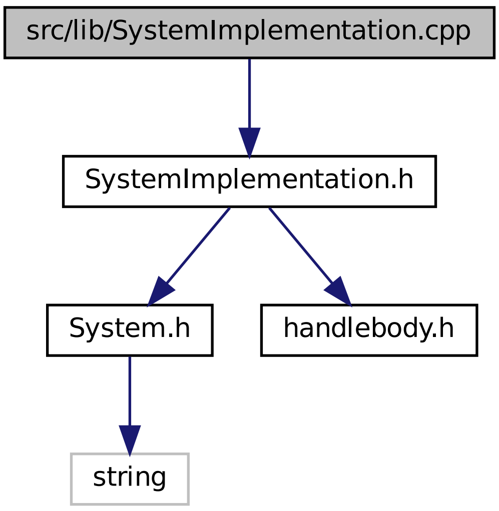
Go to the documentation of this file.

```
//
// Created by joaozenobio on 27/04/2022.
//

#ifndef ENG1_SYSTEM_H
#define ENG1_SYSTEM_H

#include <string>

class System {
public:
    virtual ~System() = default;
    virtual std::string getName() const = 0;
    virtual void setName(std::string) = 0;
    virtual double getValue() const = 0;
    virtual void setValue(double) = 0;
};


#endif //ENG1_SYSTEM_H
```

6.23 src/lib/SystemImplementation.cpp File Reference

```
#include "SystemImplementation.h"
```

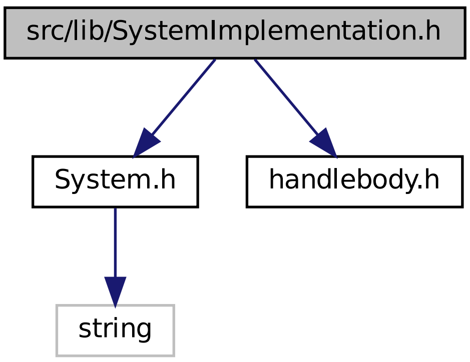
Include dependency graph for SystemImplementation.cpp:



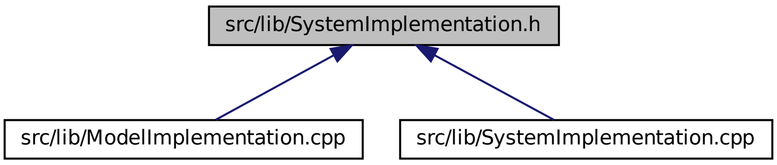
6.24 src/lib/SystemImplementation.h File Reference

```
#include "System.h"
#include "handlebody.h"
```

Include dependency graph for SystemImplementation.h:



This graph shows which files directly or indirectly include this file:


src/lib/SystemImplementation.h
src/lib/ModelImplementation.cpp
src/lib/SystemImplementation.cpp

Classes

- class SystemBody
- class SystemHandle

6.25 SystemImplementation.h

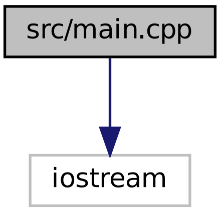
Go to the documentation of this file.

```
//
// Created by joaozenobio on 28/04/2022.
//

#ifndef ENG1_SYSTEMIMPLEMENTATION_H
#define ENG1_SYSTEMIMPLEMENTATION_H

#include "System.h"
#include "handlebody.h"

class SystemBody : public Body {
protected:
    std::string name;
    double value;

public:
    virtual ~SystemBody() override;

    SystemBody(std::string name="", double value=0.0);

    SystemBody(const SystemBody& system);

    SystemBody& operator=(const SystemBody& system);

    std::string getName() const;

    void setName(std::string n);

    double getValue() const;

    void setValue(double v);
};

class SystemHandle : public Handle<SystemBody>, public System{
public:
    ~SystemHandle() override = default;

    SystemHandle(std::string name="", double value=0.0){
        pImpl_->setName(name);
        pImpl_->setValue(value);
    }

    std::string getName() const override {
        return pImpl_->getName();
    }

    void setName(std::string sysName) override {
        pImpl_->setName(sysName);
    }

    double getValue() const override {
        return pImpl_->getValue();
    }

    void setValue(double sysValue) override {
        pImpl_->setValue(sysValue);
    }
};

#endif //ENG1_SYSTEMIMPLEMENTATION_H
```

6.26 src/main.cpp File Reference

`#include <iostream>`

Include dependency graph for main.cpp:



Functions

- int main ()

6.26.1 Function Documentation

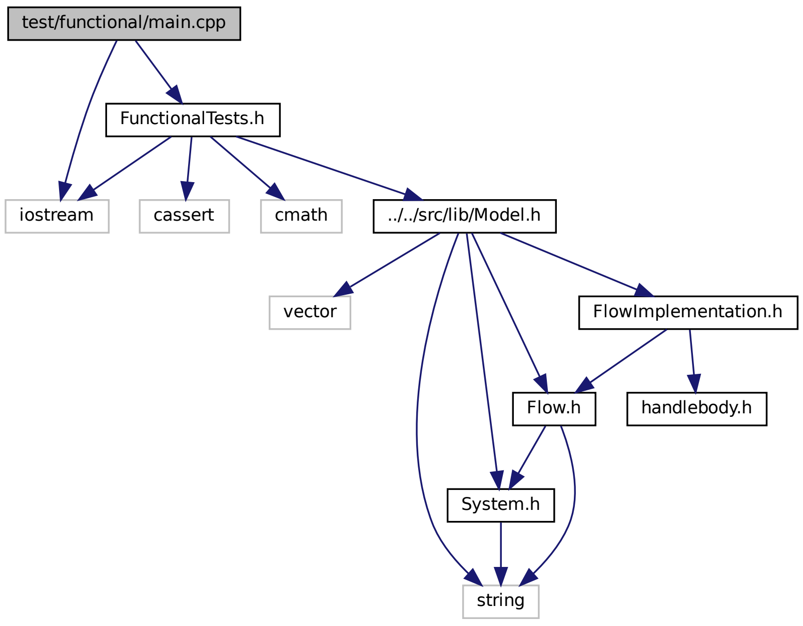
6.26.1.1 main()

```
int main ( )
```

6.27 test/functional/main.cpp File Reference

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

Include dependency graph for main.cpp:



Macros

- #define DEBUGING

Functions

- int main ()

Variables

- int numHandleCreated
- int numHandleDeleted
- int numBodyCreated
- int numBodyDeleted

6.27.1 Macro Definition Documentation

6.27.1.1 DEBUGING

```
#define DEBUGING
```

6.27.2 Function Documentation

6.27.2.1 main()

```
int main ( )
```

6.27.3 Variable Documentation

6.27.3.1 numBodyCreated

```
int numBodyCreated
```

6.27.3.2 numBodyDeleted

```
int numBodyDeleted
```

6.27.3.3 numHandleCreated

```
int numHandleCreated
```

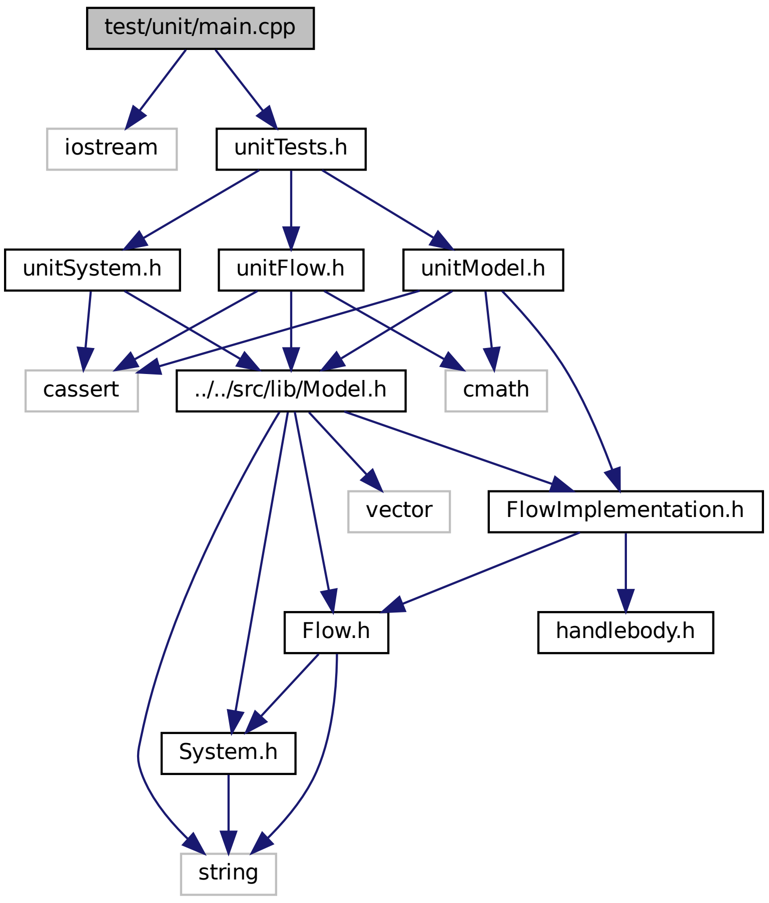
6.27.3.4 numHandleDeleted

```
int numHandleDeleted
```

6.28 test/unit/main.cpp File Reference

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

Include dependency graph for main.cpp:



Macros

- #define DEBUGING

Functions

- int main ()

Variables

- int numHandleCreated
- int numHandleDeleted
- int numBodyCreated
- int numBodyDeleted

6.28.1 Macro Definition Documentation

6.28.1.1 DEBUGING

```
#define DEBUGING
```

6.28.2 Function Documentation

6.28.2.1 main()

```
int main ( )
```

6.28.3 Variable Documentation

6.28.3.1 numBodyCreated

```
int numBodyCreated
```

6.28.3.2 numBodyDeleted

```
int numBodyDeleted
```

6.28.3.3 numHandleCreated

```
int numHandleCreated
```

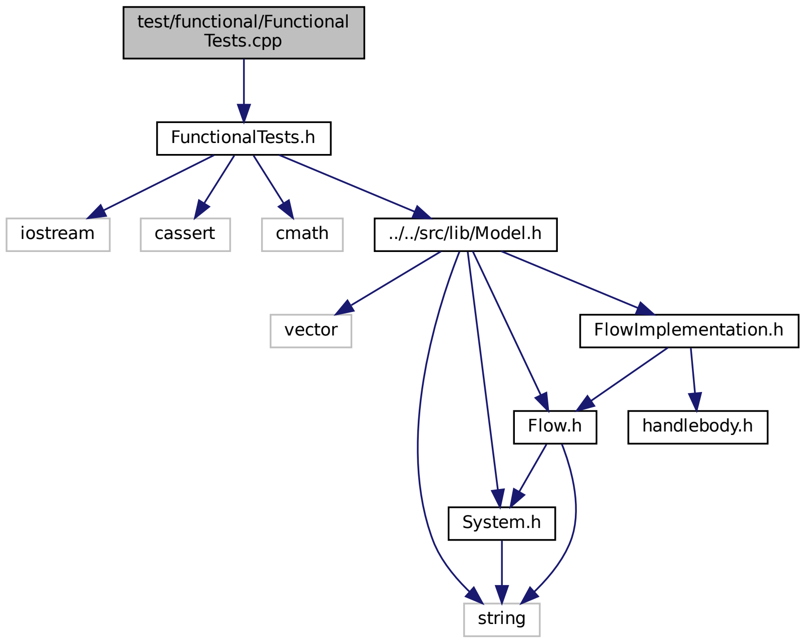
6.28.3.4 numHandleDeleted

```
int numHandleDeleted
```

6.29 test/functional/FunctionalTests.cpp File Reference

```
#include "FunctionalTests.h"
```

Include dependency graph for FunctionalTests.cpp:



Functions

- void ExponencialTest ()
- void LogisticalTest ()
- void ComplexTest ()

6.29.1 Function Documentation

6.29.1.1 ComplexTest()

```
void ComplexTest ( )
```

Function to test the complex flow.

6.29.1.2 ExponencialTest()

```
void ExponencialTest ( )
```

Function to test the exponencial flow.

6.29.1.3 LogisticalTest()

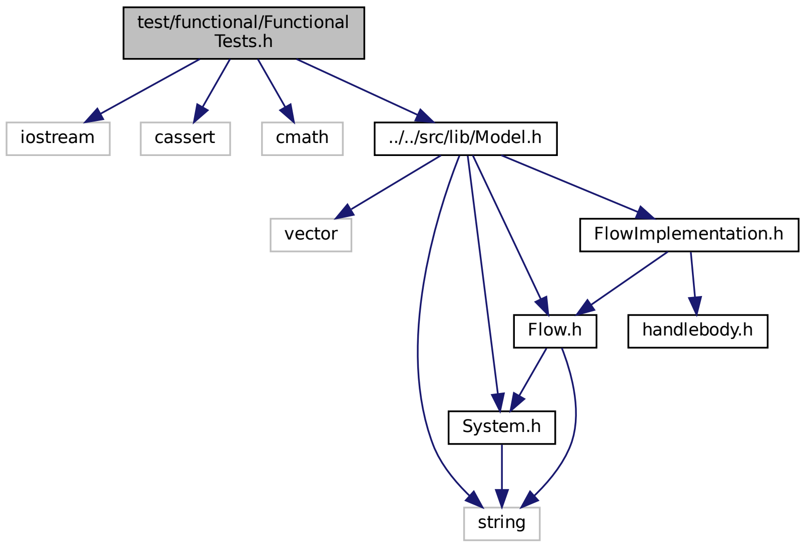
```
void LogisticalTest ( )
```

Function to test the logistical flow.

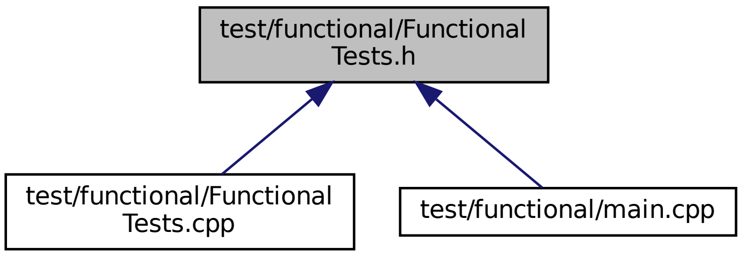
6.30 test/functional/FunctionalTests.h File Reference

```
#include <iostream>
#include <cassert>
#include <cmath>
#include "../../src/lib/Model.h"
```

Include dependency graph for FunctionalTests.h:



This graph shows which files directly or indirectly include this file:


test/functional/Functional Tests.h
test/functional/Functional Tests.cpp
test/functional/main.cpp

Classes

- class ExponencialFlow
- class LogisticalFlow
- class ComplexFlow

Functions

- void ExponencialTest ()
- void ComplexTest ()
- void LogisticalTest ()

6.30.1 Function Documentation

6.30.1.1 ComplexTest()

```
void ComplexTest ( )
```

Function to test the complex flow.

6.30.1.2 ExponencialTest()

```
void ExponencialTest ( )
```

Function to test the exponencial flow.

6.30.1.3 LogisticalTest()

```
void LogisticalTest ( )
```

Function to test the logistical flow.

6.31 FunctionalTests.h

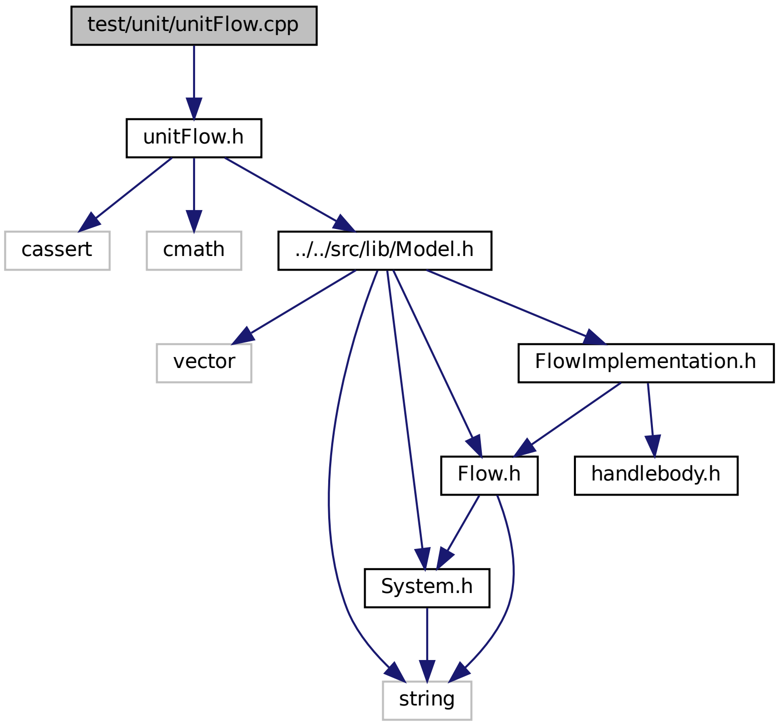
Go to the documentation of this file.

```
//
// Created by joaozenobio on 28/04/2022.
//

#ifndef ENG1_FUNCTIONALTESTS_H
#define ENG1_FUNCTIONALTESTS_H

#include <iostream>
#include <cassert>
#include <cmath>

#include "../../src/lib/Model.h"

class ExponencialFlow : public FlowBody{
public:
    ExponencialFlow(std::string name="", System* systemOut=NULL, System* systemIn=NULL) : FlowBody(name,
    systemOut, systemIn) {}
    double expression() override {
        return 0.01 * getSystemBegin()->getValue();
    }
};

class LogisticalFlow : public FlowBody{
public:
    LogisticalFlow(std::string name="", System* systemOut=NULL, System* systemIn=NULL) : FlowBody(name,
    systemOut, systemIn) {}
    double expression() override {
        return 0.01 * this->getSystemEnd()->getValue() * (1 - this->getSystemEnd()->getValue() / 70);
    }
};

class ComplexFlow : public FlowBody{
public:
    ComplexFlow(std::string name="", System* systemOut=NULL, System* systemIn=NULL) : FlowBody(name,
    systemOut, systemIn) {}
    double expression() override {
        return 0.01 * getSystemBegin()->getValue();
    }
};

void ExponencialTest();
void ComplexTest();
void LogisticalTest();

#endif //ENG1_FUNCTIONALTESTS_H
```

6.32 test/unit/unitFlow.cpp File Reference

```
#include "unitFlow.h"
```

Include dependency graph for unitFlow.cpp:



Functions

- void unitFlowDestructor ()
- void unitFlowDefaultConstructor ()
- void unitFlowExpression ()
- void unitFlowGetName ()
- void unitFlowSetName ()
- void unitFlowGetValue ()
- void unitFlowSetValue ()
- void unitFlowGetSystemBegin ()
- void unitFlowSetSystemBegin ()
- void unitFlowGetSystemEnd ()
- void unitFlowSetSystemEnd ()
- void runUnitTestsFlow ()

6.32.1 Function Documentation

6.32.1.1 runUnitTestsFlow()

```
void runUnitTestsFlow ( )
```

Run all unit tests for Flow

6.32.1.2 unitFlowDefaultConstructor()

```
void unitFlowDefaultConstructor ( )
```

Tests Flow default constructor

6.32.1.3 unitFlowDestructor()

```
void unitFlowDestructor ( )
```

Tests Flow destructor

6.32.1.4 unitFlowExpression()

```
void unitFlowExpression ( )
```

Tests Flow expression

6.32.1.5 unitFlowGetName()

```
void unitFlowGetName ( )
```

Tests Flow getName method

6.32.1.6 unitFlowGetSystemBegin()

```
void unitFlowGetSystemBegin ( )
```

Tests Flow getSystemBegin method

6.32.1.7 unitFlowGetSystemEnd()

```
void unitFlowGetSystemEnd ( )
```

Tests Flow getSystemEnd method

6.32.1.8 unitFlowGetValue()

```
void unitFlowGetValue ( )
```

Tests Flow getValue method

6.32.1.9 unitFlowSetName()

```
void unitFlowSetName ( )
```

Tests Flow setName method

6.32.1.10 unitFlowSetSystemBegin()

```
void unitFlowSetSystemBegin ( )
```

Tests Flow setSystemBegin method

6.32.1.11 unitFlowSetSystemEnd()

```
void unitFlowSetSystemEnd ( )
```

Tests Flow setSystemEnd method

6.32.1.12 unitFlowSetValue()

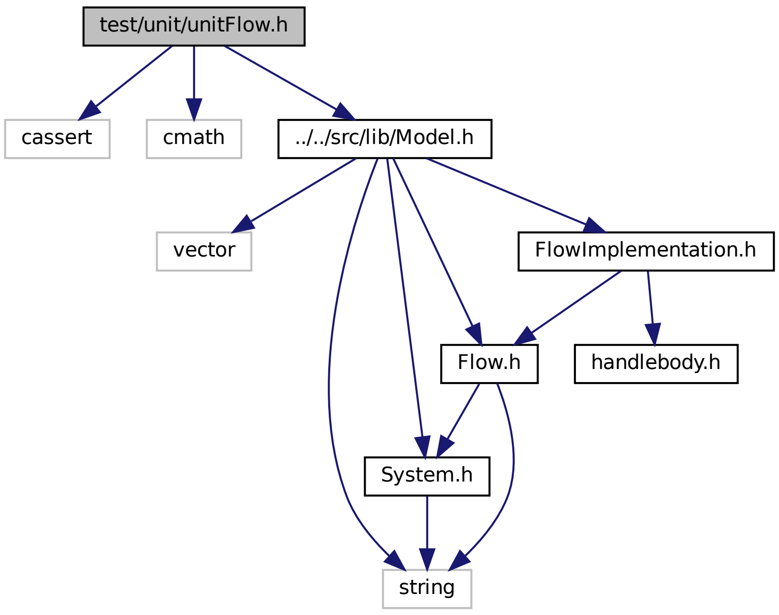
```
void unitFlowSetValue ( )
```

Tests Flow setValue method

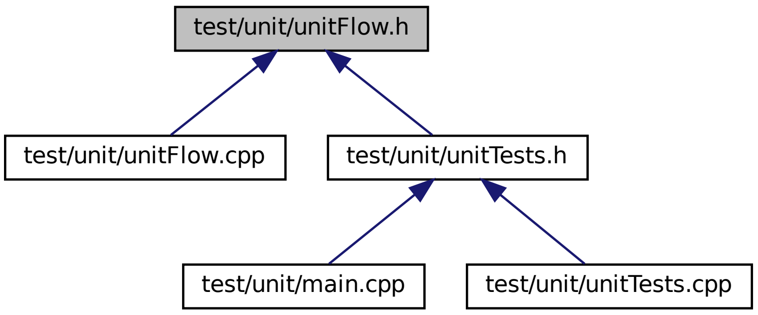
6.33 test/unit/unitFlow.h File Reference

```
#include <cassert>
#include <cmath>
#include "../../src/lib/Model.h"
```

Include dependency graph for unitFlow.h:



This graph shows which files directly or indirectly include this file:



Classes

- class UnitTestFlow

Functions

- void unitFlowDestructor ()
- void unitFlowDefaultConstructor ()
- void unitFlowExpression ()
- void unitFlowGetName ()
- void unitFlowSetName ()
- void unitFlowGetValue ()
- void unitFlowSetValue ()
- void unitFlowGetSystemBegin ()
- void unitFlowSetSystemBegin ()
- void unitFlowGetSystemEnd ()
- void unitFlowSetSystemEnd ()
- void runUnitTestsFlow ()

6.33.1 Function Documentation

6.33.1.1 runUnitTestsFlow()

```
void runUnitTestsFlow ( )
```

Run all unit tests for Flow

6.33.1.2 unitFlowDefaultConstructor()

```
void unitFlowDefaultConstructor ( )
```

Tests Flow default constructor

6.33.1.3 unitFlowDestructor()

```
void unitFlowDestructor ( )
```

Tests Flow destructor

6.33.1.4 unitFlowExpression()

```
void unitFlowExpression ( )
```

Tests Flow expression

6.33.1.5 unitFlowGetName()

```
void unitFlowGetName ( )
```

Tests Flow getName method

6.33.1.6 unitFlowGetSystemBegin()

```
void unitFlowGetSystemBegin ( )
```

Tests Flow getSystemBegin method

6.33.1.7 unitFlowGetSystemEnd()

```
void unitFlowGetSystemEnd ( )
```

Tests Flow getSystemEnd method

6.33.1.8 unitFlowGetValue()

```
void unitFlowGetValue ( )
```

Tests Flow getValue method

6.33.1.9 unitFlowSetName()

```
void unitFlowSetName ( )
```

Tests Flow setName method

6.33.1.10 unitFlowSetSystemBegin()

```
void unitFlowSetSystemBegin ( )
```

Tests Flow setSystemBegin method

6.33.1.11 unitFlowSetSystemEnd()

```
void unitFlowSetSystemEnd ( )
```

Tests Flow setSystemEnd method

6.33.1.12 unitFlowSetValue()

```
void unitFlowSetValue ( )
```

Tests Flow setValue method

6.34 unitFlow.h

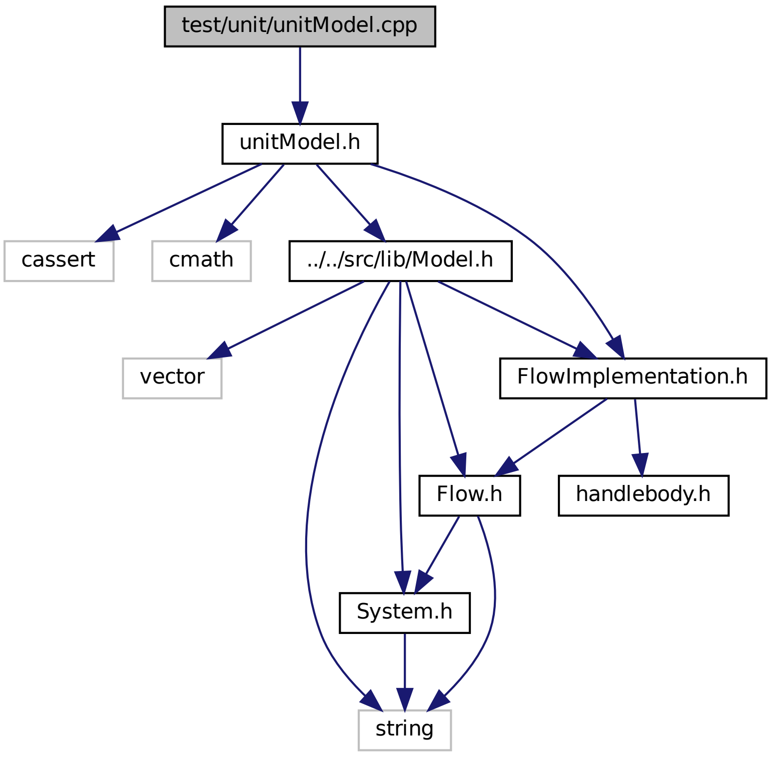
Go to the documentation of this file.

```
//
// Created by joaozenobio on 19/05/22.
//

#ifndef ENG1_UNITFLOW_H
#define ENG1_UNITFLOW_H


#include <cassert>
#include <cmath>

#include "../../src/lib/Model.h"

class UnitTestFlow : public FlowBody{
public:
    UnitTestFlow(std::string name="", System* systemBegin=NULL, System* systemEnd=NULL) : FlowBody(name,
    systemBegin, systemEnd) {}
    double expression() override {
        return 0.01 * getSystemBegin()->getValue();
    }
};

void unitFlowDestructor();
void unitFlowDefaultConstructor();
void unitFlowExpression();
void unitFlowGetName();
void unitFlowSetName();
void unitFlowGetValue();
void unitFlowSetValue();
void unitFlowGetSystemBegin();
void unitFlowSetSystemBegin();
void unitFlowGetSystemEnd();
void unitFlowSetSystemEnd();
void runUnitTestsFlow();


#endif //ENG1_UNITFLOW_H
```

6.35 test/unit/unitModel.cpp File Reference

```
#include "unitModel.h"
```

Include dependency graph for unitModel.cpp:



Functions

- void unitModelDestructor ()
- void unitModelDefaultConstructor ()
- void unitModelSimulate ()
- void unitModelGetName ()
- void unitModelSetName ()
- void unitModelGetTime ()
- void unitModelSetTime ()
- void unitModelAddSystem ()
- void unitModelAddFlow ()
- void unitModelCreateSystem ()
- void unitModelCreateFlow ()
- void unitModelCreateModel ()
- void runUnitTestsModel ()

6.35.1 Function Documentation

6.35.1.1 runUnitTestsModel()

```
void runUnitTestsModel ( )
```

Run all unit tests for Model

6.35.1.2 unitModelAddFlow()

```
void unitModelAddFlow ( )
```

Tests Model add to add a Flow

6.35.1.3 unitModelAddSystem()

```
void unitModelAddSystem ( )
```

Tests Model add to add a System

6.35.1.4 unitModelCreateFlow()

```
void unitModelCreateFlow ( )
```

Tests Model createFlow

6.35.1.5 unitModelCreateModel()

```
void unitModelCreateModel ( )
```

Tests Model createModel

6.35.1.6 unitModelCreateSystem()

```
void unitModelCreateSystem ( )
```

Tests Model createSystem

6.35.1.7 unitModelDefaultConstructor()

```
void unitModelDefaultConstructor ( )
```

Tests Model default constructor

6.35.1.8 unitModelDestructor()

```
void unitModelDestructor ( )
```

Tests Model destructor

6.35.1.9 unitModelGetName()

```
void unitModelGetName ( )
```

Tests Model getName method

6.35.1.10 unitModelGetTime()

```
void unitModelGetTime ( )
```

Tests Model getTime method

6.35.1.11 unitModelSetName()

```
void unitModelSetName ( )
```

Tests Model setName method

6.35.1.12 unitModelSetTime()

```
void unitModelSetTime ( )
```

Tests Model setTime method

6.35.1.13 unitModelSimulate()

```
void unitModelSimulate ( )
```

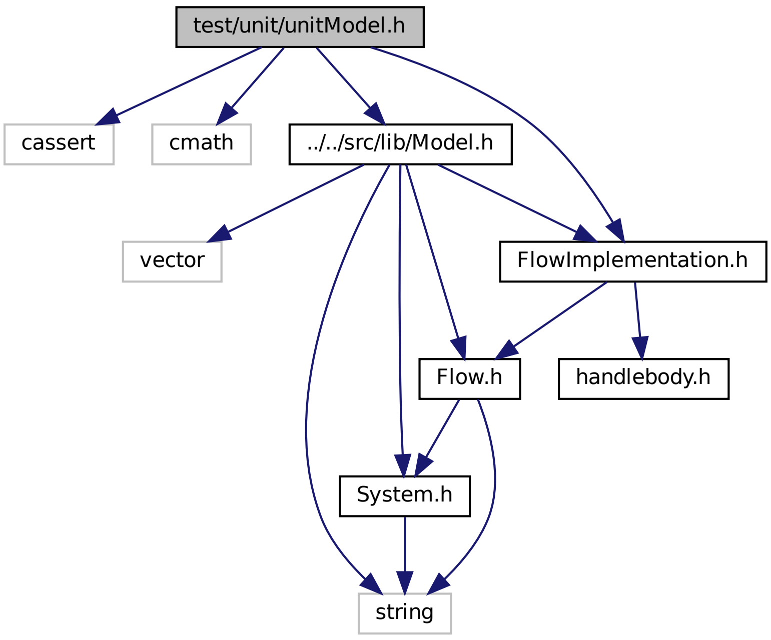
Tests Model simluate method

6.36 test/unit/unitModel.h File Reference

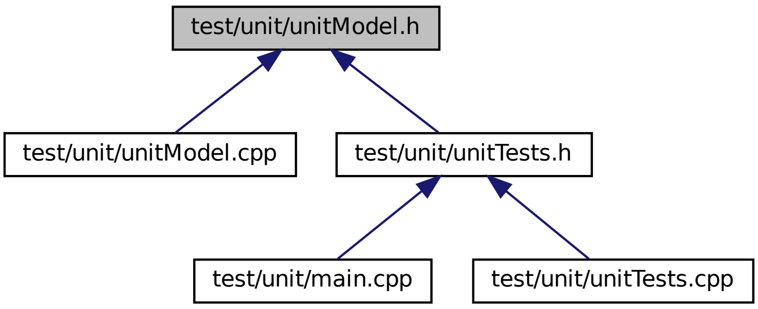
```
#include <cassert>
#include <cmath>
#include "../../src/lib/Model.h"
```

```
#include "../../src/lib/FlowImplementation.h"
```

Include dependency graph for unitModel.h:



This graph shows which files directly or indirectly include this file:



Classes

- class UnitTestFlow2

Functions

- void unitModelDestructor ()
- void unitModelDefaultConstructor ()
- void unitModelSimulate ()
- void unitModelGetName ()
- void unitModelSetName ()
- void unitModelGetTime ()
- void unitModelSetTime ()
- void unitModelAddSystem ()
- void unitModelAddFlow ()
- void unitModelCreateSystem ()
- void unitModelCreateFlow ()
- void unitModelCreateModel ()
- void runUnitTestsModel ()

6.36.1 Function Documentation

6.36.1.1 runUnitTestsModel()

```
void runUnitTestsModel ( )
```

Run all unit tests for Model

6.36.1.2 unitModelAddFlow()

```
void unitModelAddFlow ( )
```

Tests Model add to add a Flow

6.36.1.3 unitModelAddSystem()

```
void unitModelAddSystem ( )
```

Tests Model add to add a System

6.36.1.4 unitModelCreateFlow()

```
void unitModelCreateFlow ( )
```

Tests Model createFlow

6.36.1.5 unitModelCreateModel()

```
void unitModelCreateModel ( )
```

Tests Model createModel

6.36.1.6 unitModelCreateSystem()

```
void unitModelCreateSystem ( )
```

Tests Model createSystem

6.36.1.7 unitModelDefaultConstructor()

```
void unitModelDefaultConstructor ( )
```

Tests Model default constructor

6.36.1.8 unitModelDestructor()

```
void unitModelDestructor ( )
```

Tests Model destructor

6.36.1.9 unitModelGetName()

```
void unitModelGetName ( )
```

Tests Model getName method

6.36.1.10 unitModelGetTime()

```
void unitModelGetTime ( )
```

Tests Model getTime method

6.36.1.11 unitModelSetName()

```
void unitModelSetName ( )
```

Tests Model setName method

6.36.1.12 unitModelSetTime()

```
void unitModelSetTime ( )
```

Tests Model setTime method

6.36.1.13 unitModelSimulate()

```
void unitModelSimulate ( )
```

Tests Model simluate method

6.37 unitModel.h

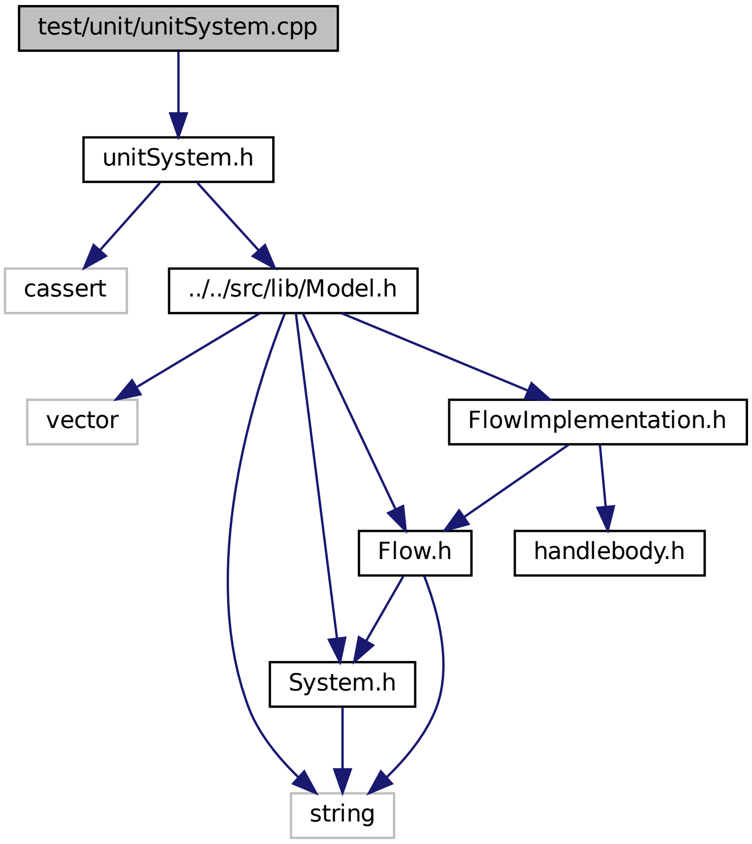
Go to the documentation of this file.

```
//
// Created by joaozenobio on 19/05/22.
//

#ifndef ENG1_UNITMODEL_H
#define ENG1_UNITMODEL_H


#include <cassert>
#include <cmath>

#include "../../src/lib/Model.h"
#include "../../src/lib/FlowImplementation.h"

class UnitTestFlow2 : public FlowBody{
public:
    UnitTestFlow2(std::string name="", System* systemBegin=NULL, System* systemEnd=NULL) : FlowBody(name,
    systemBegin, systemEnd) {}
    double expression() override {
        return 0.01 * getSystemBegin()->getValue();
    }
};

void unitModelDestructor();
void unitModelDefaultConstructor();
void unitModelSimulate();
void unitModelGetName();
void unitModelSetName();
void unitModelGetTime();
void unitModelSetTime();
void unitModelAddSystem();
void unitModelAddFlow();
void unitModelCreateSystem();
void unitModelCreateFlow();
void unitModelCreateModel();
void runUnitTestsModel();


#endif //ENG1_UNITMODEL_H
```

6.38 test/unit/unitSystem.cpp File Reference

```
#include "unitSystem.h"
```

Include dependency graph for unitSystem.cpp:



Functions

- void unitSystemDestructor ()
- void unitSystemDefaultConstructor ()
- void unitSystemGetName ()
- void unitSystemSetName ()
- void unitSystemGetValue ()
- void unitSystemSetValue ()
- void runUnitTestsSystem ()

6.38.1 Function Documentation

6.38.1.1 runUnitTestsSystem()

```
void runUnitTestsSystem ( )
```

Run all unit tests for System

6.38.1.2 unitSystemDefaultConstructor()

```
void unitSystemDefaultConstructor ( )
```

Tests System default constructor

6.38.1.3 unitSystemDestructor()

```
void unitSystemDestructor ( )
```

Tests System destructor

6.38.1.4 unitSystemGetName()

```
void unitSystemGetName ( )
```

Tests System getName method

6.38.1.5 unitSystemGetValue()

```
void unitSystemGetValue ( )
```

Tests System getValue method

6.38.1.6 unitSystemSetName()

```
void unitSystemSetName ( )
```

Tests System setName method

6.38.1.7 unitSystemSetValue()

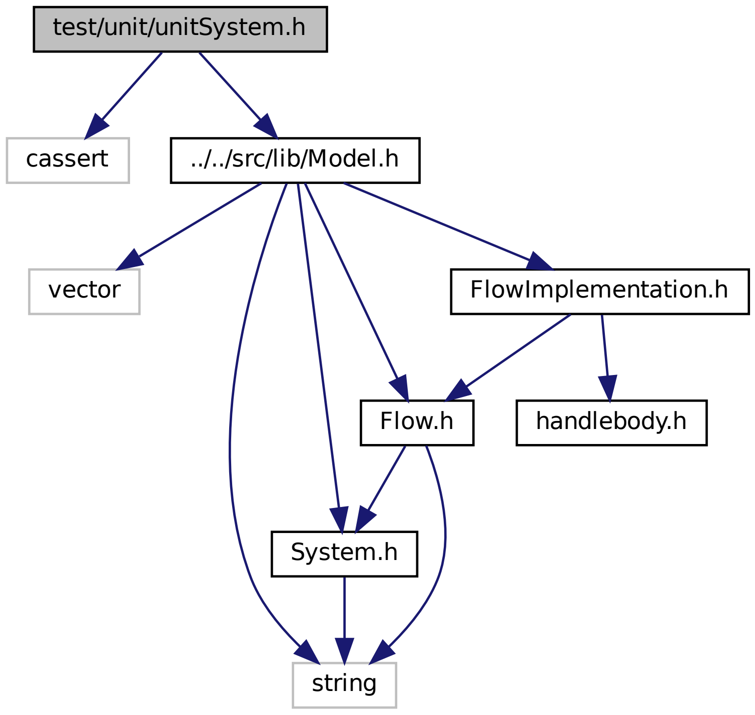
```
void unitSystemSetValue ( )
```

Tests System setValue method

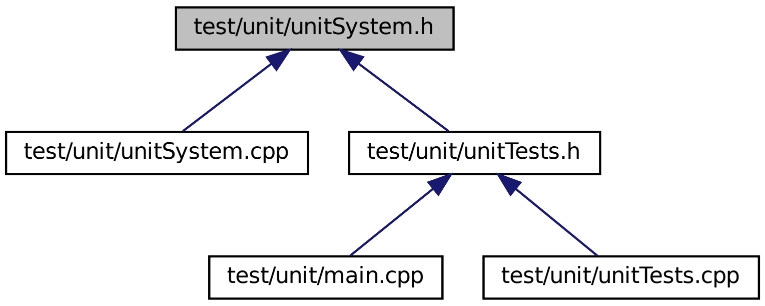
6.39 test/unit/unitSystem.h File Reference

```
#include <cassert>
#include "../../src/lib/Model.h"
```

Include dependency graph for unitSystem.h:



This graph shows which files directly or indirectly include this file:



Functions

- void unitSystemDestructor ()
- void unitSystemDefaultConstructor ()
- void unitSystemGetName ()
- void unitSystemSetName ()
- void unitSystemGetValue ()
- void unitSystemSetValue ()
- void runUnitTestsSystem ()

6.39.1 Function Documentation

6.39.1.1 runUnitTestsSystem()

```
void runUnitTestsSystem ( )
```

Run all unit tests for System

6.39.1.2 unitSystemDefaultConstructor()

```
void unitSystemDefaultConstructor ( )
```

Tests System default constructor

6.39.1.3 unitSystemDestructor()

```
void unitSystemDestructor ( )
```

Tests System destructor

6.39.1.4 unitSystemGetName()

```
void unitSystemGetName ( )
```

Tests System getName method

6.39.1.5 unitSystemGetValue()

```
void unitSystemGetValue ( )
```

Tests System getValue method

6.39.1.6 unitSystemSetName()

```
void unitSystemSetName ( )
```

Tests System setName method

6.39.1.7 unitSystemSetValue()

```
void unitSystemSetValue ( )
```

Tests System setValue method

6.40 unitSystem.h

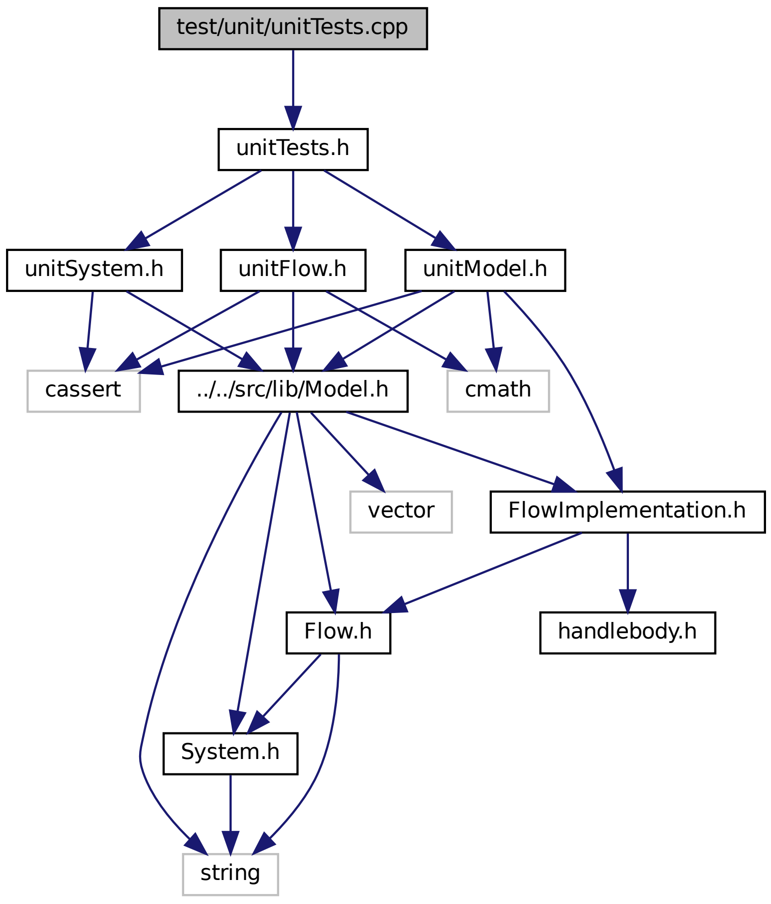
Go to the documentation of this file.

```
//
// Created by joaozenobio on 19/05/22.
//

#ifndef ENG1_UNITSYSTEM_H
#define ENG1_UNITSYSTEM_H

#include <cassert>

#include "../../src/lib/Model.h"

void unitSystemDestructor();
void unitSystemDefaultConstructor();
void unitSystemGetName();
void unitSystemSetName();
void unitSystemGetValue();
void unitSystemSetValue();
void runUnitTestsSystem();

#endif //ENG1_UNITSYSTEM_H
```

6.41 test/unit/unitTests.cpp File Reference

```
#include "unitTests.h"
```

Include dependency graph for unitTests.cpp:



Functions

- void runGlobal ()

6.41.1 Function Documentation

6.41.1.1 runGlobal()

```
void runGlobal ( )
```

Tests System methods

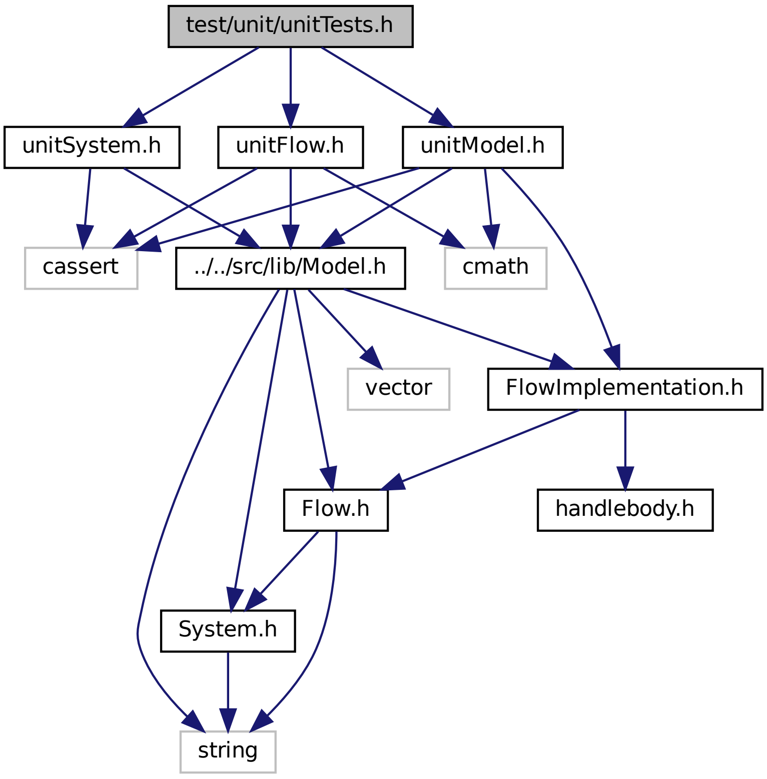
Tests Flow methods

Tests Model methods

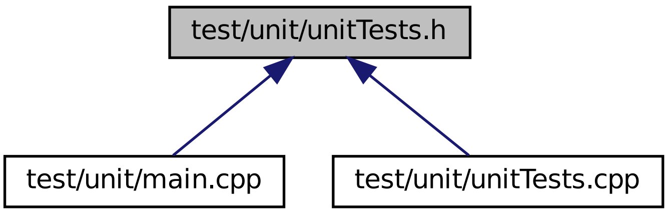
6.42 test/unit/unitTests.h File Reference

```
#include "unitModel.h"
#include "unitFlow.h"
#include "unitSystem.h"
```

Include dependency graph for unitTests.h:



This graph shows which files directly or indirectly include this file:



Functions

- void runGlobal ()

6.42.1 Function Documentation

6.42.1.1 runGlobal()

```
void runGlobal ( )
```

Tests System methods

Tests Flow methods

Tests Model methods

6.43 unitTests.h

Go to the documentation of this file.

```
//
// Created by joaozenobio on 19/05/22.
//

#ifndef ENG1_UNITTESTS_H
#define ENG1_UNITTESTS_H


#include "unitModel.h"
#include "unitFlow.h"
#include "unitSystem.h"

void runGlobal();


#endif //ENG1_UNITTESTS_H
```

Index

__has_include
    CMakeCCompilerId.c, 61
    CMakeCXXCompilerId.cpp, 65
∼Body
    Body, 10
∼Flow
    Flow, 15
∼FlowBody
    FlowBody, 19
∼FlowHandle
    FlowHandle< Flow_IMPL >, 24
∼Handle
    Handle< T >, 28
∼Model
    Model, 32
∼ModelBody
    ModelBody, 38
∼ModelHandle
    ModelHandle, 43
∼System
    System, 48
∼SystemBody
    SystemBody, 50
∼SystemHandle
    SystemHandle, 54

add
    Model, 32, 33
    ModelBody, 38, 39
    ModelHandle, 43, 44
ARCHITECTURE_ID
    CMakeCCompilerId.c, 62
    CMakeCXXCompilerId.cpp, 65
attach
    Body, 10

Body, 9
    ∼Body, 10
    attach, 10
    Body, 10
    detach, 10
    refCount, 10

C_VERSION
    CMakeCCompilerId.c, 62
cmake-build-debug/CMakeCache.txt, 61
cmake-build-debug/CMakeFiles/3.22.3/CompilerIdC/CMakeCCompilerId.c, 61
cmake-build-debug/CMakeFiles/3.22.3/CompilerIdCXX/CMakeCXXCompilerId.cpp, 64
cmake-build-debug/CMakeFiles/clion-environment.txt, 68
cmake-build-debug/CMakeFiles/clion-log.txt, 68
cmake-build-debug/CMakeFiles/TargetDirectories.txt, 68
CMakeCCompilerId.c
    __has_include, 61
    ARCHITECTURE_ID, 62
    C_VERSION, 62
    COMPILER_ID, 62
    DEC, 62
    HEX, 62
    info_arch, 63
    info_compiler, 63
    info_language_extensions_default, 63
    info_language_standard_default, 64
    info_platform, 64
    main, 63
    PLATFORM_ID, 62
    STRINGIFY, 63
    STRINGIFY_HELPER, 63
CMakeCXXCompilerId.cpp
    __has_include, 65
    ARCHITECTURE_ID, 65
    COMPILER_ID, 65
    CXX_STD, 65
    DEC, 65
    HEX, 65
    info_arch, 66
    info_compiler, 67
    info_language_extensions_default, 67
    info_language_standard_default, 67
    info_platform, 67
    main, 66
    PLATFORM_ID, 66
    STRINGIFY, 66
    STRINGIFY_HELPER, 66
CMakeLists.txt, 68
COMPILER_ID
    CMakeCCompilerId.c, 62
    CMakeCXXCompilerId.cpp, 65
ComplexFlow, 11
    ComplexFlow, 12
    expression, 12
ComplexTest
    FunctionalTests.cpp, 90
    FunctionalTests.h, 92
createFlow
    Model, 33

createModel
  Model, 33
  ModelBody, 39
  ModelHandle, 44
createSystem
  Model, 33
  ModelBody, 39
  ModelHandle, 44
CXX_STD
  CMakeCXXCompilerId.cpp, 65

DEBUGING
  handlebody.h, 73
  main.cpp, 86, 89
DEC
  CMakeCCompilerId.c, 62
  CMakeCXXCompilerId.cpp, 65
detach
  Body, 10

endFlows
  Model, 33
  ModelBody, 39
  ModelHandle, 44
endModels
  Model, 34
  ModelBody, 39
  ModelHandle, 44
endSystems
  Model, 34
  ModelBody, 39
  ModelHandle, 45
ExponencialFlow, 13
  ExponencialFlow, 14
  expression, 14
ExponencialTest
  FunctionalTests.cpp, 90
  FunctionalTests.h, 92
expression
  ComplexFlow, 12
  ExponencialFlow, 14
  Flow, 15
  FlowBody, 20
  FlowHandle< Flow_IMPL >, 24
  LogisticalFlow, 31
  UnitTestFlow, 58
  UnitTestFlow2, 60

Flow, 15
  ∼Flow, 15
  expression, 15
  getName, 16
  getSystemBegin, 16
  getSystemEnd, 16
  getValue, 16
  setName, 16
  setSystemBegin, 17
  setSystemEnd, 17
  setValue, 17
FlowBody, 18
  ∼FlowBody, 19
  expression, 20
  FlowBody, 19, 20
  getName, 20
  getSystemBegin, 20
  getSystemEnd, 21
  getValue, 21
  name, 22
  operator=, 21
  setName, 21
  setSystemBegin, 21
  setSystemEnd, 22
  setValue, 22
  systemBegin, 22
  systemEnd, 23
  value, 23
FlowHandle
  FlowHandle< Flow_IMPL >, 24
FlowHandle< Flow_IMPL >, 23
  ∼FlowHandle, 24
  expression, 24
  FlowHandle, 24
  getName, 24
  getSystemBegin, 25
  getSystemEnd, 25
  getValue, 25
  setName, 25
  setSystemBegin, 26
  setSystemEnd, 26
  setValue, 26
flows
  ModelBody, 41
FunctionalTests.cpp
  ComplexTest, 90
  ExponencialTest, 90
  LogisticalTest, 90
FunctionalTests.h
  ComplexTest, 92
  ExponencialTest, 92
  LogisticalTest, 92

getFlowsIterator
  Model, 34
  ModelBody, 39
  ModelHandle, 45
getModelsIterator
  Model, 34
  ModelBody, 40
  ModelHandle, 45
getName
  Flow, 16
  FlowBody, 20
  FlowHandle< Flow_IMPL >, 24
  Model, 34
  ModelBody, 40
  ModelHandle, 45
  System, 48
  SystemBody, 51

SystemHandle, 55
getSystemBegin
Flow, 16
FlowBody, 20
FlowHandle< Flow_IMPL >, 25
getSystemEnd
Flow, 16
FlowBody, 21
FlowHandle< Flow_IMPL >, 25
getSystemsIterator
Model, 34
ModelBody, 40
ModelHandle, 45
getTime
Model, 35
ModelBody, 40
ModelHandle, 46
getValue
Flow, 16
FlowBody, 21
FlowHandle< Flow_IMPL >, 25
System, 48
SystemBody, 51
SystemHandle, 55

Handle
Handle< T >, 27, 28
Handle< T >, 27
∼Handle, 28
Handle, 27, 28
operator=, 28
pImpl_, 28
handlebody.h
DEBUGING, 73
numBodyCreated, 74
numBodyDeleted, 74
numHandleCreated, 74
numHandleDeleted, 74
HEX
CMakeCCompilerId.c, 62
CMakeCXXCompilerId.cpp, 65

info_arch
CMakeCCompilerId.c, 63
CMakeCXXCompilerId.cpp, 66
info_compiler
CMakeCCompilerId.c, 63
CMakeCXXCompilerId.cpp, 67
info_language_extensions_default
CMakeCCompilerId.c, 63
CMakeCXXCompilerId.cpp, 67
info_language_standard_default
CMakeCCompilerId.c, 64
CMakeCXXCompilerId.cpp, 67
info_platform
CMakeCCompilerId.c, 64
CMakeCXXCompilerId.cpp, 67

LogisticalFlow, 29
expression, 31
LogisticalFlow, 30
LogisticalTest
FunctionalTests.cpp, 90
FunctionalTests.h, 92

main
CMakeCCompilerId.c, 63
CMakeCXXCompilerId.cpp, 66
main.cpp, 85, 87, 89
main.cpp
DEBUGING, 86, 89
main, 85, 87, 89
numBodyCreated, 87, 89
numBodyDeleted, 87, 89
numHandleCreated, 87, 89
numHandleDeleted, 87, 89
Model, 31
∼Model, 32
add, 32, 33
createFlow, 33
createModel, 33
createSystem, 33
endFlows, 33
endModels, 34
endSystems, 34
getFlowsIterator, 34
getModelsIterator, 34
getName, 34
getSystemsIterator, 34
getTime, 35
setName, 35
setTime, 35
simulate, 35
ModelBody, 36
∼ModelBody, 38
add, 38, 39
createModel, 39
createSystem, 39
endFlows, 39
endModels, 39
endSystems, 39
flows, 41
getFlowsIterator, 39
getModelsIterator, 40
getName, 40
getSystemsIterator, 40
getTime, 40
ModelBody, 38
models, 41
name, 41
setName, 40
setTime, 40
simulate, 41
systems, 41
time, 42
ModelHandle, 42
∼ModelHandle, 43
add, 43, 44

createModel, 44
createSystem, 44
endFlows, 44
endModels, 44
endSystems, 45
getFlowsIterator, 45
getModelsIterator, 45
getName, 45
getSystemsIterator, 45
getTime, 46
ModelHandle, 43
setName, 46
setTime, 46
simulate, 46
models
ModelBody, 41

name
FlowBody, 22
ModelBody, 41
SystemBody, 53
numBodyCreated
handlebody.h, 74
main.cpp, 87, 89
numBodyDeleted
handlebody.h, 74
main.cpp, 87, 89
numHandleCreated
handlebody.h, 74
main.cpp, 87, 89
numHandleDeleted
handlebody.h, 74
main.cpp, 87, 89

operator=
FlowBody, 21
Handle< T >, 28
SystemBody, 52

pImpl_
Handle< T >, 28
PLATFORM_ID
CMakeCCompilerId.c, 62
CMakeCXXCompilerId.cpp, 66

README.md, 68
refCount
Body, 10
runGlobal
unitTests.cpp, 112
unitTests.h, 114
runUnitTestsFlow
unitFlow.cpp, 94
unitFlow.h, 97
runUnitTestsModel
unitModel.cpp, 100
unitModel.h, 104
runUnitTestsSystem
unitSystem.cpp, 107
unitSystem.h, 110

setName
Flow, 16
FlowBody, 21
FlowHandle< Flow_IMPL >, 25
Model, 35
ModelBody, 40
ModelHandle, 46
System, 48
SystemBody, 52
SystemHandle, 55
setSystemBegin
Flow, 17
FlowBody, 21
FlowHandle< Flow_IMPL >, 26
setSystemEnd
Flow, 17
FlowBody, 22
FlowHandle< Flow_IMPL >, 26
setTime
Model, 35
ModelBody, 40
ModelHandle, 46
setValue
Flow, 17
FlowBody, 22
FlowHandle< Flow_IMPL >, 26
System, 49
SystemBody, 52
SystemHandle, 55
simulate
Model, 35
ModelBody, 41
ModelHandle, 46
src/lib/Flow.h, 68, 69
src/lib/FlowImplementation.cpp, 70
src/lib/FlowImplementation.h, 70, 71
src/lib/handlebody.h, 73, 74
src/lib/Model.h, 75, 76
src/lib/ModelImplementation.cpp, 77
src/lib/ModelImplementation.h, 78, 80
src/lib/System.h, 81, 82
src/lib/SystemImplementation.cpp, 83
src/lib/SystemImplementation.h, 83, 84
src/main.cpp, 85
STRINGIFY
CMakeCCompilerId.c, 63
CMakeCXXCompilerId.cpp, 66
STRINGIFY_HELPER
CMakeCCompilerId.c, 63
CMakeCXXCompilerId.cpp, 66
System, 47
~System, 48
getName, 48
getValue, 48
setName, 48
setValue, 49
systemBegin

FlowBody, 22
SystemBody, 49
~SystemBody, 50
getName, 51
getValue, 51
name, 53
operator=, 52
setName, 52
setValue, 52
SystemBody, 51
value, 53
systemEnd
FlowBody, 23
SystemHandle, 53
~SystemHandle, 54
getName, 55
getValue, 55
setName, 55
setValue, 55
SystemHandle, 54
systems
ModelBody, 41

test/functional/FunctionalTests.cpp, 90
test/functional/FunctionalTests.h, 91, 93
test/functional/main.cpp, 86
test/unit/main.cpp, 88
test/unit/unitFlow.cpp, 94
test/unit/unitFlow.h, 96, 99
test/unit/unitModel.cpp, 100
test/unit/unitModel.h, 102, 106
test/unit/unitSystem.cpp, 107
test/unit/unitSystem.h, 109, 111
test/unit/unitTests.cpp, 112
test/unit/unitTests.h, 113, 114
time
ModelBody, 42

unitFlow.cpp
runUnitTestsFlow, 94
unitFlowDefaultConstructor, 95
unitFlowDestructor, 95
unitFlowExpression, 95
unitFlowGetName, 95
unitFlowGetSystemBegin, 95
unitFlowGetSystemEnd, 95
unitFlowGetValue, 95
unitFlowSetName, 95
unitFlowSetSystemBegin, 96
unitFlowSetSystemEnd, 96
unitFlowSetValue, 96
unitFlow.h
runUnitTestsFlow, 97
unitFlowDefaultConstructor, 97
unitFlowDestructor, 98
unitFlowExpression, 98
unitFlowGetName, 98
unitFlowGetSystemBegin, 98
unitFlowGetSystemEnd, 98
unitFlowGetValue, 98
unitFlowSetName, 98
unitFlowSetSystemBegin, 98
unitFlowSetSystemEnd, 99
unitFlowSetValue, 99
unitFlowDefaultConstructor
unitFlow.cpp, 95
unitFlow.h, 97
unitFlowDestructor
unitFlow.cpp, 95
unitFlow.h, 98
unitFlowExpression
unitFlow.cpp, 95
unitFlow.h, 98
unitFlowGetName
unitFlow.cpp, 95
unitFlow.h, 98
unitFlowGetSystemBegin
unitFlow.cpp, 95
unitFlow.h, 98
unitFlowGetSystemEnd
unitFlow.cpp, 95
unitFlow.h, 98
unitFlowGetValue
unitFlow.cpp, 95
unitFlow.h, 98
unitFlowSetName
unitFlow.cpp, 95
unitFlow.h, 98
unitFlowSetSystemBegin
unitFlow.cpp, 96
unitFlow.h, 98
unitFlowSetSystemEnd
unitFlow.cpp, 96
unitFlow.h, 99
unitFlowSetValue
unitFlow.cpp, 96
unitFlow.h, 99
unitModel.cpp
runUnitTestsModel, 100
unitModelAddFlow, 101
unitModelAddSystem, 101
unitModelCreateFlow, 101
unitModelCreateModel, 101
unitModelCreateSystem, 101
unitModelDefaultConstructor, 101
unitModelDestructor, 101
unitModelGetName, 101
unitModelGetTime, 102
unitModelSetName, 102
unitModelSetTime, 102
unitModelSimulate, 102
unitModel.h
runUnitTestsModel, 104
unitModelAddFlow, 104
unitModelAddSystem, 104
unitModelCreateFlow, 104
unitModelCreateModel, 104

unitModelCreateSystem, 104
unitModelDefaultConstructor, 105
unitModelDestructor, 105
unitModelGetName, 105
unitModelGetTime, 105
unitModelSetName, 105
unitModelSetTime, 105
unitModelSimulate, 105
unitModelAddFlow
unitModel.cpp, 101
unitModel.h, 104
unitModelAddSystem
unitModel.cpp, 101
unitModel.h, 104
unitModelCreateFlow
unitModel.cpp, 101
unitModel.h, 104
unitModelCreateModel
unitModel.cpp, 101
unitModel.h, 104
unitModelCreateSystem
unitModel.cpp, 101
unitModel.h, 104
unitModelDefaultConstructor
unitModel.cpp, 101
unitModel.h, 105
unitModelDestructor
unitModel.cpp, 101
unitModel.h, 105
unitModelGetName
unitModel.cpp, 101
unitModel.h, 105
unitModelGetTime
unitModel.cpp, 102
unitModel.h, 105
unitModelSetName
unitModel.cpp, 102
unitModel.h, 105
unitModelSetTime
unitModel.cpp, 102
unitModel.h, 105
unitModelSimulate
unitModel.cpp, 102
unitModel.h, 105
unitSystem.cpp
runUnitTestsSystem, 107
unitSystemDefaultConstructor, 108
unitSystemDestructor, 108
unitSystemGetName, 108
unitSystemGetValue, 108
unitSystemSetName, 108
unitSystemSetValue, 108
unitSystem.h
runUnitTestsSystem, 110
unitSystemDefaultConstructor, 110
unitSystemDestructor, 110
unitSystemGetName, 110
unitSystemGetValue, 110
unitSystemSetName, 110
unitSystemSetValue, 111
unitSystemDefaultConstructor
unitSystem.cpp, 108
unitSystem.h, 110
unitSystemDestructor
unitSystem.cpp, 108
unitSystem.h, 110
unitSystemGetName
unitSystem.cpp, 108
unitSystem.h, 110
unitSystemGetValue
unitSystem.cpp, 108
unitSystem.h, 110
unitSystemSetName
unitSystem.cpp, 108
unitSystem.h, 110
unitSystemSetValue
unitSystem.cpp, 108
unitSystem.h, 111
UnitTestFlow, 56
expression, 58
UnitTestFlow, 57
UnitTestFlow2, 58
expression, 60
UnitTestFlow2, 59
unitTests.cpp
runGlobal, 112
unitTests.h
runGlobal, 114

value
FlowBody, 23
SystemBody, 53